INTERNATIONAL TELECOMMUNICATION UNION

TELECOMMUNICATION STANDARDIZATION SECTOR

STUDY PERIOD 1993 - 1996

COM15- -E
November 1995
Original: English

Question 2/15

STUDY GROUP 15 - CONTRIBUTION

SOURCE* : RAPPORTEURS FOR Q.2/15 (Sakae OKUBO, Richard SCHAPHORST)

TITLE: DRAFT RECOMMENDATION H.245

This contribution provides a text for Draft Recommendation H.245 "Control protocol for multimedia communication" which was determined at the Study Group 15 meeting in February 1995.

Summary

This Recommendation specifies syntax and semantics of terminal information messages as well as procedures to use them for in-band negotiation at the start of or during communication. The messages cover receiving and transmitting capabilities as well as mode preference from the receiving end, logical channel signalling, and Control & Indication. Acknowledged signalling procedures are specified to ensure reliable audiovisual and data communication.

* Contact: Mike Nilsson
BT Labs
Ipswich, UK

Tel.: +44 1473 645413
Fax: +44 1473 643791
E-mail: nilsson_m_e@bt-web.bt.co.uk

Bill Welsh
BT Labs
Ipswich, UK

Tel.: +44 1473 643810
Fax: +44 1473 643791
E-mail: welsh_w_j@bt-web.bt.co.uk

INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

DRAFT H.245

(20 October, 1995)

LINE TRANSMISSION OF NON-TELEPHONE SIGNALS

CONTROL PROTOCOL FOR MULTIMEDIA COMMUNICATION

DRAFT ITU-T Recommendation H.245

FOREWORD

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

The World Telecommunication Standardization Conference (WTSC), which meets every four years, established the topics for study by the ITU-T Study Groups which, in their turn, produce Recommendations on these topics.

ITU-T Recommendation H.245 was prepared by the ITU-T Study Group 15 (199x-199x) and was approved by the WTSC (Place, Month xx-xx, 199x).

NOTES

1 "Shall" is used in this Recommedation to specify a mandatory requirement. "Should" is used in this Recommendation to specify a suggested, but not required, course of action. "May" is used to specify an optional course of action, without expressing a preference.

CONTENTS

SUMMARY

This Recommendation specifies syntax and semantics of terminal information messages as well as procedures to use them for in-band negotiation at the start of or during communication. The messages cover receiving and transmitting capabilities as well as mode preference from the receiving end, logical channel signalling, and Control & Indication. Acknowledged signalling procedures are specified to ensure reliable audiovisual and data communication.

CONTROL PROTOCOL FOR MULTIMEDIA COMMUNICATION

(Place, 199x)

11 Scope

This Recommendation specifies syntax and semantics of terminal information messages as well as procedures to use them for in-band negotiation at the start of or during communication. The messages cover receiving and transmitting capabilities as well as mode preference from the receiving end, logical channel signalling, and Control & Indication. Acknowledged signalling procedures are specified to ensure reliable audiovisual and data communication.

This Recommendation covers a wide range of applications, including storage/retrieval, messaging and distribution services as well as conversational. It applies to, but is not limited to, multimedia systems that use the multiplexes defined in H.222.0 and H.223. These different systems share the same syntax and semantics, and are therefore bit-wise compatible. Some of the procedures are applicable to all systems, while the others are more specific to particular systems.

The different systems that make use of this Recommendation may specify the use of different transport protocols. However, it is intended to be used with a reliable transport layer, that is, one that provides guaranteed delivery of correct data.

Note: there should be no confusion with the T.120 management system, which is carried within the data stream, and covers different functionalities from those described here - the H.245 stream and the T.120-data stream are complementary.

22 References

The following ITU-T Recommendations, and other references, contain provisions which, through reference in this text, constitute the provisions of the Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; all users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published.

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

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

[33] ITU-T Recommendation G.723 (1995) - Dual rate speech coder for multimedia communication transmitting at 5.3 & 6.3 kbit/s

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

[55] ITU-T Recommendation H.221 (1993) - Frame structure for a 64 to 1920 kbit/s channel in audiovisual teleservices

[66] ITU-T Recommendation H.222.0 (1995) - Coding of Moving Pictures and Associated Audio: Systems - ISO/IEC 13818-1

[77] ITU-T Recommendation H.222.1 (1995) - Multimedia multiplex and synchronisation for audiovisual communication in ATM environments

[88] ITU-T Recommendation H.223 (1995) - Multiplexing protocol for low bitrate multimedia communication

[99] ITU-T Recommendation H.224 (1995) - A real time control protocol for simplex applications using the H.221 LSD/HSD/MLP channels

[1010] ITU-T Recommendation H.230 (1993) - Frame-synchronous control and indication signals for audiovisual systems

[1111] ITU-T Recommendation H.233 (1993) - Confidentiality system for audiovisual services

[1212] ITU-T Recommendation H.234 (1993) - Authentication and key management

[1313] ITU-T Recommendation H.261 (1993) - Video Codec for audiovisual services at px64 kbit/s

[1414] ITU-T Recommendation H.262 (1995) - Generic Coding of Moving Pictures and Associated Audio: Video - ISO/IEC 13818-2

[1515] ITU-T Recommendation H.263 (1995) - Video coding for low bitrate communication

[1616] ITU-T Recommendation H.281 (1995) - A far end camera control protocol for videoconferences using H.224

[1717] ITU-T Recommendation H.320 (1993) - Narrow-band ISDN visual telephone systems and terminal equipment

[1818] ITU-T Recommendation H.324 (1995) - Terminal for low bitrate multimedia communication

[1919] ITU-T Recommendation I.363 (1993) - B-ISDN ATM adaptation layer (AAL) specification

[2020] ITU-T Recommendation Q.2931 (1995) - Broadband integrated services digital network (B-ISDN) - Digital subscriber signalling No. 2 (DSS 2) - User network interface layer 3 specification for basic call/connection control

[2121] ITU-T Recommendation T.30 (1994) - Procedures for document facsimile transmission in the general switched telephone network

[2222] ITU-T Recommendation T.35 (1991) - Procedure for the allocation of CCITT defined codes for non-standard facilities

[2323] ITU-T Recommendation T.51 (1993) - Latin based coded character sets for telematic services

[2424] Draft ITU-T T.84 | ISO/IEC 10918-3 (199x): "Digital Compression and Coding of Continuous Tone Still Images - Extensions"

[2525] ITU-T Recommendation T.120 (199x) - Data protocols for multimedia conferencing - under development

[2626] ITU-T Recommendation T.434 (1992) - Binary File Transfer Format for the Telematic Services

[2727] ITU-T Recommendation V.14 (1993) - Transmission of start-stop characters over synchronous bearer channels

[2828] ITU-T Recommendation V.34 (1994) - A modem operating at data signalling rates of up to 28 800 bit/s for use on the general switched telephone network and on leased point-to-point 2-wire telephone-type circuits

[2929] ITU-T Recommendation V.42 (1993) - Error-correcting procedures for DCEs using asynchronous-to-synchronous conversion

[3030] ITU-T Recommendation X.680 (1994): Information Technology - Abstract Syntax Notation One (ASN.1) - Specification of basic notation

[3131] ITU-T Recommendation X.691 (1995): Information Technology - ASN.1 Encoding Rules - Specification of Packed Encoding Rules (PER)

[3232] ISO/IEC 3309 (1991): Information Technology - Telecommunications and information exchange between systems - High-level data link control (HDLC) procedures - Frame Structure

[3333] ISO/IEC 11172-3 (1993): Information Technology - Coding of Moving Pictures and Associated Audio for digital storage media at up to about 1,5 Mbit/s - Part 3: Audio

[3434] ISO/IEC 13818-3 (1995): Information Technology - Generic Coding of Moving Pictures and Associated Audio - Part 3: Audio

[3535] ISO/IEC 13818-6 (1996): Information Technology - Generic Coding of Moving Pictures and Associated Audio - Part 6: Digital Storage Media Command and Control

[3636] ISO/IEC TR9577 (1990): Information Technology - Telecommunications information exchange between systems - protocol identification in the network layer

33 Definitions

For the purpose of this Recommendation, the following definitions apply:

Bi-directional Logical Cchannel: A bi-directional logical channel consists of a pair of associated logical channels, one in each direction of transmission.

Capability: A terminal has a particular capability if it is able to encode and transmit or receive and decode that particular signal.

Channel: A channel is a uni-directional link between two end points.

Command: A command is a message that requires action but no explicit response.

Elementary Steam: Elementary Steam is a generic term for a coded video, coded audio or other coded bitstream.

In-band: In-band messages are those that are transported within the channel or logical channel to which they refer.

Indication: An indication is a message that contains information but does not require action or response.

Logical Channel: A logical channel is a uni-directional path or bi-directional path for the transmission of inofrmation.a single elementary stream.

Logical Channel Number: A logical channel number is a number that identifies a single logical channel.

Logical Channel Signalling: Logical channel signalling is a set of procedures that are used to open and close logical channels.

Master Terminal: A master terminal is the terminal that is determined as being the master terminal by the master-slave determination procedure defined in this Recommendation, or by some other procedure; a master terminal may initiate the opening of bi-directional channels.

Medium Type: A medium type is a single form of information that is presented to a user or the data representing that information: video, audio and text are example Medium Types.

Mode: A mode is a set of elementary streams that a terminal is transmitting, intends to transmit, or would like to receive.

Multimedia communication: Multimedia communication refers to the transmission and/or reception of signals of two or more Medium Types simultaneously.

Non-standard: Not conforming to a national or international standard referenced in this Recommendation.

Multipoint: Multipoint refers to the simultaneous interconnection of three or more terminals to allow communication among several sites through the use of multipoint control units (bridges) that centrally direct the flow of information.

Request: A request is a message that results in action by the remote terminal and requires an immediate response from it.

Response: A response is a message that is the response to a request.

Session: A session is a period of communication between two terminals which may be conversational or non-conversational (for example retrieval from a database).

Slave Terminal: A slave terminal is the terminal that is determined as being the slave terminal by the master-slave determination procedure defined in this Recommendation, or by some other procedure; a slave terminal may not initiate the opening of bi-directional channels, but may request the master terminal to do so.

Support: The ability to operate in a given mode, however a requirement to support a mode does not mean that the mode must actually be used at all times: unless prohibited, other modes may be used by mutual negotiation.

Terminal: A terminal is any endpoint and may be a user's terminal or some other communication system such as an MCU or an information server.

Uni-directional Logical Channel: A uni-directional logical channel is a path for the transmission of a single elementary stream from one terminal to another.

44 Abbreviations

For the purpose of this Recommendation, the following abbreviations are used:

AAL ATM Adaptation layer

AL1,2,3 H.223 Adaptation layers 1, 2 and 3

ASN.1 Abstract syntax notation 1

ATM Asynchronous transfer mode

B-LCSE Bi-directional Logical Channel Signalling Entity

CESE Capability Exchange Signalling Entity

CLCSE Close Logical Channel Signalling Entity

CIF Common Intermediate Format (of a video picture: refer to H.261 and H.263)

CPCS Common Part Convergence Sublayer (of ATM Adaptation Layer 5)

DTMF Dual tone multi-frequency

DSM-CC Digital storage media - command and control

GOB Group of blocks (of a video picture: refer to H.261 and H.263)

GSTN General switched telephone network

HDLC High-level data link control

HRD Hypothetical Reference Decoder (refer to H.261 and H.263)

IV Initialisation Vector (used for encryption: refer to H.233 and H.234)

LAPM Link access protocol for modems

LCSE Logical Channel Signalling Entity

MCU Multipoint control unit

MPI Minimum picture interval

MSDSE Master Slave Determination Signalling Entity

MTSE Multiplex Table Signalling Entity

MRSE Mode Request Signalling Entity

OBCSE Open Bi-directional Channel Signalling Entity

PCR Program Clock Reference (refer to ISO/IEC 13818-1 | H.222.0)

PDU Protocol data unit

PID Packet Identifier (refer to ISO/IEC 13818-1 | H.222.0)

QCIF Quarter CIF

RTDSE Round Trip Delay Signalling Entity

SDL Specification and description Language

SDU Service data unit

SE Session Exchange message (used for encryption: refer to H.233 and H.234)

SQCIF Sub QCIF

STD System Target Decoder (refer to ISO/IEC 13818-1 | H.222.0)

VC ATM Virtual channel

XID Exchange identification (frame)

55 General

This Recommendation provides a number of different services, some of which are expected to be applicable to all terminals that use it and some that are more specific to particular ones. Procedures are defined to allow the exchange of audiovisual and data capabilities; to request the transmission of a particular audiovisual and data mode; to manage the logical channels used to transport the audiovisual and data information; to establish which terminal is the master terminal and which is the slave terminal for the purposes of managing bi-directional logical channels; to carry various control and indication signals; to control the bit rate of individual logical channels and the whole multiplex; and to measure the round trip delay, from one terminal to the other and back. These procedures are explained in more detail below.

Following this general introduction, there are sections detailing the message syntax and semantics and the procedures. The syntax has been defined using ASN.1 notation [3030] and the semantics define the meaning of syntax elements as well as providing syntactic constraints that are not specified in the ASN.1 syntax. The procedures section defines the protocols that use the messages defined in the other sections.

Although not all of the messages and procedures defined in this Recommendation will be applicable to all terminals, no indication of such restrictions is given here. These restrictions are the responsibility of the recommendations that use this Recommendation.

This Recommendation has been defined to be independent of the underlying transport mechanism, but is intended to be used with a reliable transport layer, that is, one that provides guaranteed delivery of correct data.

55.11 Capability exchange

The capability exchange procedures are intended to ensure that the only multimedia signals to be transmitted are those that can be received and treated appropriately by the receive terminal. This requires that the capabilities of each terminal to receive and decode be known to the other terminal. It is not necessary that a terminal understand or store all in-coming capabilities; those that are not understood, or can not be used shall be ignored, and no fault shall be considered to have occurred.

The total capability of a terminal to receive and decode various signals is made known to the other terminal by transmission of its capability set.

Receive capabilities describe the terminal's ability to receive and process in-coming information streams. Transmitters shall limit the content of their transmitted information to that which the receiver has indicated it is capable of receiving. The absence of a receive capability indicates that the terminal cannot receive (is a transmitter only).

Transmit capabilities describe the terminal's ability to transmit information streams. Transmit capabilities serve to offer receivers a choice of possible modes of operation, so that the receiver may request the mode which it prefers to receive. The absence of a transmit capability indicates that the terminal is not offering a choice of preferred modes to the receiver (but it may still transmit anything within the capability of the receiver).

These capability sets provide for more than one stream of a given medium type to be sent simultaneously. For example, a terminal may declare its ability to receive (or send) two independent H.262 video streams and two independent G.722 audio streams at the same time. Capability messages have been defined to allow a terminal to indicate that it does not have fixed capabilities, but that they depend on which other modes are being used simultaneously. For example, it is possible to indicate that higher resolution video can be decoded when a simpler audio algorithm is used; or that either two low resolution video sequences can be decoded or a single high resolution one. It is also possible to indicate trade-offs between the capability to transmit and the capability to receive.

Non-standard capabilities and control messages may be issued using the NonStandardParameter structure. Note that while the meaning of non-standard messages is defined by individual organizations, equipment built by any manufacturer may signal any non-standard message, if the meaning is known.

Terminals may reissue capability sets at any time.

55.22 Audiovisual and data mode request

When the capability exchange protocol has been completed, both terminals will be aware of each other's capability to transmit and receive as specified in the capability descriptors that have been exchanged. It is not mandatory for a terminal to declare all its capabilities; it need only declare those that it wishes to be used.

A terminal may indicate its capabilities to transmit. A terminal that receives transmission capabilities from the remote terminal may request a particular mode to be transmitted to it. A terminal indicates that it does not want its transmission mode to be controlled by the remote terminal by sending no transmission capabilities.

55.33 Logical channel signalling procedures

An acknowledged protocol is defined for the opening and closing of logical channels which carry the audiovisual and data information. The aim of these procedures is to ensure that a terminal is capable of receiving and decoding the data that will be transmitted on a logical channel at the time the logical channel is opened rather than at the time the first data is transmitted on it; and to ensure that the receive terminal is ready to receive and decode the data that will be transmitted on the logical channel before that transmission starts. The logical channel open message includes a description of the data to be transported, for example, H.262 MP@ML at 6Mbit/s. Logical channels should only be opened when there is sufficient capability to receive data on all open logical channels simultaneously.

A part of this protocol is concerned with the opening of bi-directional channels. To avoid timing problems, one terminal is defined as the master terminal, and the other as the slave terminal. Only the master terminal can initiate the opening of bi-directional channels: however, the slave terminal may request the master terminal to do so.

A protocol is defined to establish which terminal is the master and which is the slave. However, systems that use this Recommendation may specify other means of determining which terminal is the master and which is the slave.

55.44 Commands and indications

Commands and indications are provided for various purposes: loops for maintenance; video/audio active/inactive signals to inform the user; fast update request for source switching in multipoint applications are some examples. Neither commands nor indications elicit response messages from the remote terminal. Commands force an action at the remote terminal whilst indications merely provide information and do not force any action.

A command is defined to allow the bit rate of logical channels and the whole multiplex to be controlled from the remote terminal. This has a number of purposes: interworking with terminals using multiplexes in which only a finite number of bit rates are available; multi-point applications where the rates from different sources should be matched; and flow control in congested networks.

66 Messages: syntax

This section specifies the syntax of messages using the notation defined in ASN.1 [3030]. Messages shall be encoded for transmission by applying the packed encoding rules specified in [3134] using the basic aligned variant. The first bit in each octet which is transmitted is the most significant bit of the octet as is specified in X.691.

```
MULTIMEDIA-SYSTEM-CONTROL DEFINITIONS AUTOMATIC TAGS ::=
BEGIN

-- Export all symbols

-- ================================================================================
-- Top level PDUs
-- ================================================================================

MultimediaSystemControlPDU          ::=CHOICE
{
    request                         RequestPDU,
    response                        ResponsePDU,
    command                         CommandPDU,
    indication                      IndicationPDU,
    ...
}

-- A RequestPDU results in action and requires an immediate response

RequestPDU                          ::=CHOICE
{
    nonStandard                     NonStandardPDU,

    masterSlaveDetermination        MasterSlaveDetermination,

    terminalCapabilitySet           TerminalCapabilitySet,
    terminalCapabilitySetRelease    TerminalCapabilitySetRelease,

    openLogicalChannel              OpenLogicalChannel,
    closeLogicalChannel             CloseLogicalChannel,

    openBiDirectionalChannelRequest OpenBiDirectionalChannelRequest,
    openBiDirectionalChannelRelease OpenBiDirectionalChannelRelease,

    requestChannelClose             RequestChannelClose,
    requestChannelCloseRelease      RequestChannelCloseRelease,

    multiplexEntrySend              MultiplexEntrySend,
    multiplexEntrySendRelease       MultiplexEntrySendRelease,

    requestMultiplexEntry           RequestMultiplexEntry,

    requestMode                     RequestMode,
    requestModeRelease              RequestModeRelease,

    roundTripDelayRequest           RoundTripDelayRequest,

    maintenanceLoopRequest          MaintenanceLoopRequest,

    ...
}

-- A ResponsePDU is the response to a request PDU

ResponsePDU                         ::=CHOICE
{
    nonStandard                     NonStandardPDU,
```

```
    masterSlaveDeterminationAck          MasterSlaveDeterminationAck,
    masterSlaveDeterminationReject       MasterSlaveDeterminationReject,

    terminalCapabilitySetAck             TerminalCapabilitySetAck,
    terminalCapabilitySetReject          TerminalCapabilitySetReject,

    openLogicalChannelAck                OpenLogicalChannelAck,
    openLogicalChannelReject             OpenLogicalChannelReject,
    closeLogicalChannelAck               CloseLogicalChannelAck,

    openBiDirectionalChannelAck          OpenBiDirectionalChannelAck,
    openBiDirectionalChannelReject       OpenBiDirectionalChannelReject,

    requestChannelCloseAck               RequestChannelCloseAck,
    requestChannelCloseReject            RequestChannelCloseReject,

    multiplexEntrySendAck                MultiplexEntrySendAck,
    multiplexEntrySendReject             MultiplexEntrySendReject,

    requestMultiplexEntryResponse        RequestMultiplexEntryResponse,

    requestModeAck                       RequestModeAck,
    requestModeReject                    RequestModeReject,

    roundTripDelayResponse               RoundTripDelayResponse,

    maintenanceLoopResponse              MaintenanceLoopResponse,

    ...
}

-- A CommandPDU requires action, but no explicit response

CommandPDU                               ::=CHOICE
{
    nonStandard                          NonStandardPDU,

    maintenanceLoopOffCommand            MaintenanceLoopOffCommand,

    sendTerminalCapabilitySet            SendTerminalCapabilitySet,

    encryptionCommand                    EncryptionCommand,

    flowControlCommand                   FlowControlCommand,

    endSessionCommand                    EndSessionCommand,

    miscellaneousCommand                 MiscellaneousCommand,

    ...
}

-- An IndicationPDU is information that does not require action or response

IndicationPDU                            ::=CHOICE
{
    nonStandard                          NonStandardPDU,

    functionNotSupported                 FunctionNotSupported,

    miscellaneousIndication              MiscellaneousIndication,

    jitterIndication                     JitterIndication,

    h223SkewIndication                   H223SkewIndication,

    userInput                            UserInputIndication,
    ...
```

```
}

-- SequenceNumber is defined here as it is used in a number of PDUs
SequenceNumber                  ::=INTEGER (0..255)

-- ================================================================================
-- Non standard PDU definitions
-- ================================================================================

NonStandardPDU                  ::=SEQUENCE
{
    nonStandardData             NonStandardParameter,
    ...
}

NonStandardParameter            ::=SEQUENCE
{
    nonStandardIdentifier       NonStandardIdentifier,
    data                        OCTET STRING
}

NonStandardIdentifier           ::=CHOICE
{
    object                      OBJECT IDENTIFIER,
    h221NonStandard             SEQUENCE
    {
        t35CountryCode          INTEGER (0..255),   -- country, per T.35
        t35Extension            INTEGER (0..255),   -- assigned nationally
        manufacturerCode        INTEGER (0..65535)  -- assigned nationally
    }
}

-- ================================================================================
-- Master-slave determination definitions
-- ================================================================================

MasterSlaveDetermination        ::=SEQUENCE
{
    statusDeterminationNumber   INTEGER (0..4294967295),
    ...
}

MasterSlaveDeterminationAck     ::=SEQUENCE
{
    decision                    CHOICE
    {
        master                  NULL,
        slave                   NULL
    },
    ...
}

MasterSlaveDeterminationReject  ::=SEQUENCE
{
    cause                       CHOICE
    {
        identicalNumbers        NULL,
        ...
    },
    ...
}
```

```
-- ================================================================================
-- Capability exchange definitions
-- ================================================================================

TerminalCapabilitySet                   ::=SEQUENCE
{
    sequenceNumber                      SequenceNumber,

    protocolIdentifier                  OBJECT IDENTIFIER,
                                        -- shall be set to the value
                                        -- {itu recommendation h 245 version (0) 1}

    multiplexCapability                 MultiplexCapability OPTIONAL,

    capabilityTable                     SET SIZE (1..256) OF CapabilityTableEntry OPTIONAL,

    capabilityDescriptors               SET SIZE (1..256) OF CapabilityDescriptor OPTIONAL,

    ...
}

CapabilityTableEntry                    ::=SEQUENCE
{
    capabilityTableEntryNumber          CapabilityTableEntryNumber,
    capability                          Capability OPTIONAL
}

CapabilityDescriptor                    ::=SEQUENCE
{
    capabilityDescriptorNumber          CapabilityDescriptorNumber,
    simultaneousCapabilities            SET SIZE (1..256) OF AlternativeCapabilitySet OPTIONAL
}

AlternativeCapabilitySet                ::=SEQUENCE SIZE (1..256) OF CapabilityTableEntryNumber

CapabilityTableEntryNumber              ::=INTEGER (1..655350..255)

CapabilityDescriptorNumber              ::=INTEGER (0..255)

TerminalCapabilitySetAck                ::=SEQUENCE
{
    sequenceNumber                      SequenceNumber,
    ...
}

TerminalCapabilitySetReject             ::=SEQUENCE
{
    sequenceNumber                      SequenceNumber,
    cause                               CHOICE
    {
        unspecified                     NULL,
        undefinedTableEntryUsed         NULL,
        descriptorCapacityExceeded      NULL,
        tableEntryCapacityExceeded      CHOICE
        {
            highestEntryNumberProcessed capabilityTableEntryNumber,
            noneProcessed               NULL
        }
        ...
    },
    ...
}

TerminalCapabilitySetRelease            ::=SEQUENCE
{
    ...
}
```

```
-- ================================================================================
-- Capability exchange definitions: top level capability description
-- ================================================================================

Capability                              ::=CHOICE
{
    nonStandard                         NonStandardParameter,

    receiveVideoCapability              VideoCapability,
    transmitVideoCapability             VideoCapability,
    receiveAndTransmitVideoCapability   VideoCapability,

    receiveAudioCapability              AudioCapability,
    transmitAudioCapability             AudioCapability,
    receiveAndTransmitAudioCapability   AudioCapability,

    receiveDataApplicationCapability    DataApplicationCapability,
    transmitDataApplicationCapability   DataApplicationCapability,
    receiveAndTransmitDataApplicationCapability   DataApplicationCapability,

    h233EncryptionTransmitCapability    BOOLEAN,
    h233EncryptionReceiveCapability     SEQUENCE
    {
        h233IVResponseTime              INTEGER (0..255),       -- units milliseconds
        ...
    },

    ...
}

-- ================================================================================
-- Capability exchange definitions: Multiplex capabilities
-- ================================================================================

MultiplexCapability                     ::=CHOICE
{
    nonStandard                         NonStandardParameter,
    h222Capability                      H222Capability,
    h223Capability                      H223Capability,
    ...
}

H222Capability                          ::= SEQUENCE
{
    numberOfVCs                         INTEGER (1..256),
    vcCapability                        SET OF VCCapability,
    ...
}

VCCapability                            ::=SEQUENCE
{
    aal1    SEQUENCE
    {
        nullClockRecovery               BOOLEAN,
        srtsClockRecovery               BOOLEAN,
        adaptiveClockRecovery           BOOLEAN,
        nullErrorCorrection             BOOLEAN,
        longInterleaver                 BOOLEAN,
        shortInterleaver                BOOLEAN,
        errorCorrectionOnly             BOOLEAN,
        structuredDataTransfer          BOOLEAN,
        partiallyFilledCells            BOOLEAN,
        ...
    },
    aal5    SEQUENCE
    {
        forwardMaximumSDUSize           INTEGER (0..65535),     -- units octets
        backwardMaximumSDUSize          INTEGER (0..65535),     -- units octets
```

```
        ...
    },
    transportStream                 BOOLEAN,
    programStream                   BOOLEAN,
    bitRateINTEGER (0..65535),      -- units 64 kbits per second
    ...
}

H223Capability                      ::=SEQUENCE
{
    transportWithI-frames           BOOLEAN,            -- I-frame transport of H.245

    videoWithAL1                    BOOLEAN,
    videoWithAL2                    BOOLEAN,
    videoWithAL3                    BOOLEAN,
    audioWithAL1                    BOOLEAN,
    audioWithAL2                    BOOLEAN,
    audioWithAL3                    BOOLEAN,
    dataWithAL1                     BOOLEAN,
    dataWithAL2                     BOOLEAN,
    dataWithAL3                     BOOLEAN,

    maximumAl2SDUSize               INTEGER (0..65535),     -- units octets
    maximumAl3SDUSize               INTEGER (0..65535),     -- units octets

    maximumDelayJitter              INTEGER (0..1023),      -- units milliseconds

    h223MultiplexTableCapability    CHOICE
    {
        basic                       NULL,
        enhanced                    SEQUENCE
        {
            maximumNestingDepth     INTEGER (1..15),
            maximumElementListSize  INTEGER (2..255),
            maximumSubElementListSize INTEGER (2..255),
            ...
        }
    },

    ...
}

-- ================================================================================
-- Capability exchange definitions: Video capabilities
-- ================================================================================

VideoCapability                     ::=CHOICE
{
    nonStandard                     NonStandardParameter ,
    h261VideoCapability             H261VideoCapability,
    h262VideoCapability             H262VideoCapability,
    h263VideoCapability             H263VideoCapability,
    ...
}

H261VideoCapability                 ::=SEQUENCE
{
    qcifMPI                         INTEGER (1..4) OPTIONAL,    -- units 1/29.97 Hz
    cifMPI                          INTEGER (1..4) OPTIONAL,    -- units 1/29.97 Hz
    temporalSpatialTradeOffCapability BOOLEAN,
    ...
}

H262VideoCapability                 ::=SEQUENCE
{
    profileAndLevel-SPatML          BOOLEAN,
    profileAndLevel-MPatLL          BOOLEAN,
    profileAndLevel-MPatML          BOOLEAN,
```

```
    profileAndLevel-MPatH-14        BOOLEAN,
    profileAndLevel-MPatHL          BOOLEAN,
    profileAndLevel-SNRatLL         BOOLEAN,
    profileAndLevel-SNRatML         BOOLEAN,
    profileAndLevel-SpatialatH-14   BOOLEAN,
    profileAndLevel-HPatML          BOOLEAN,
    profileAndLevel-HPatH-14        BOOLEAN,
    profileAndLevel-HPatHL          BOOLEAN,
    videoBitRate                    INTEGER (0.. 1073741823) OPTIONAL,  -- units 400 bits/sec
    vbvBufferSize                   INTEGER (0.. 262143) OPTIONAL,      -- units   16384 bits
    samplesPerLine                  INTEGER (0..16383) OPTIONAL,        -- units samples/line
    linesPerFrame                   INTEGER (0..16383) OPTIONAL,        -- units lines/frame
    framesPerSecond                 INTEGER (0..15) OPTIONAL,           -- frame_rate_code
    luminanceSampleRate             INTEGER (0..4294967295) OPTIONAL,   -- units samples/sec
    ...
}

H263VideoCapability                 ::=SEQUENCE
{
    sqcifMPI                        INTEGER (1..32) OPTIONAL,           -- units 1/29.97 Hz
    qcifMPI                         INTEGER (1..32) OPTIONAL,           -- units 1/29.97 Hz
    cifMPI                          INTEGER (1..32) OPTIONAL,           -- units 1/29.97 Hz
    cif4MPI                         INTEGER (1..32) OPTIONAL,           -- units 1/29.97 Hz
    cif16MPI                        INTEGER (1..32) OPTIONAL,           -- units 1/29.97 Hz
    unrestrictedVector              BOOLEAN,
    arithmeticCoding                BOOLEAN,
    advancedPrediction              BOOLEAN,
    pbFrames                        BOOLEAN,
    temporalSpatialTradeOffCapability BOOLEAN,
    hrd-BmaxKb                      INTEGER (0..65535) OPTIONAL,        -- units 1024 bits
    ...
}

-- ================================================================================
-- Capability exchange definitions: Audio capabilities
-- ================================================================================

-- For an H.222 multiplex, the integers indicate the size of the STD buffer in units of 256 octets
-- For an H.223 multiplex, the integers indicate the maximum number of audio frames per AL-SDU

AudioCapability                     ::=CHOICE
{
    nonStandard                     NonStandardParameter,
    g711Alaw64k                     INTEGER (1..256),
    g711Alaw56k                     INTEGER (1..256),
    g711Ulaw64k                     INTEGER (1..256),
    g711Ulaw56k                     INTEGER (1..256),

    g722-64k                        INTEGER (1..256),
    g722-56k                        INTEGER (1..256),
    g722-48k                        INTEGER (1..256),

    g723                            INTEGER (1..256),

    g7238                           SEQUENCE
    {
        maxAl-sduAudioFrames        INTEGER (1..256),
        silenceSuppression          BOOLEAN
    }

    g728                            INTEGER (1..256),
    g729                            INTEGER (1..256),
    g-dsvd                          INTEGER (1..256),

    is11172AudioCapability          IS11172AudioCapability,
    is13818AudioCapability          IS13818AudioCapability,
    ...
}
```

```
IS11172AudioCapability                  ::=SEQUENCE
{
    audioLayer1                         BOOLEAN,
    audioLayer2                         BOOLEAN,
    audioLayer3                         BOOLEAN,

    audioSampling32k                    BOOLEAN,
    audioSampling44k1                   BOOLEAN,
    audioSampling48k                    BOOLEAN,

    singleChannel                       BOOLEAN,
    twoChannels                         BOOLEAN,

    bitRate                             INTEGER (1..448),       -- units kbits/sec
    ...
}

IS13818AudioCapability                  ::=SEQUENCE
{
    audioLayer1                         BOOLEAN,
    audioLayer2                         BOOLEAN,
    audioLayer3                         BOOLEAN,

    audioSampling16k                    BOOLEAN,
    audioSampling22k05                  BOOLEAN,
    audioSampling24k                    BOOLEAN,
    audioSampling32k                    BOOLEAN,
    audioSampling44k1                   BOOLEAN,
    audioSampling48k                    BOOLEAN,

    singleChannel                       BOOLEAN,
    twoChannels                         BOOLEAN,
    threeChannels2-1                    BOOLEAN,
    threeChannels3-0                    BOOLEAN,
    fourChannels2-0-2-0                 BOOLEAN,
    fourChannels2-2                     BOOLEAN,
    fourChannels3-1                     BOOLEAN,
    fiveChannels3-0-2-0                 BOOLEAN,
    fiveChannels3-2                     BOOLEAN,

    lowFrequencyEnhancement             BOOLEAN,

    multilingual                        BOOLEAN,

    bitRate                             INTEGER (1..1130),      -- units kbits/sec
    ...
}

-- ================================================================================
-- Capability exchange definitions: Data capabilities
-- ================================================================================

DataApplicationCapability               ::=CHOICE
{
    nonStandard                         NonStandardParameter,
    t120                                DataProtocolCapability,
    dsm-cc                              DataProtocolCapability,
    userData                            DataProtocolCapability,
    t84                                 SEQUENCE
    {
        t84Protocol                     DataProtocolCapability,
        t84Profile                      T84Profile
    },
    t434                                DataProtocolCapability,
    h224                                DataProtocolCapability,
    nlpid                               SEQUENCE
    {
```

```
        nlpidProtocol               DataModeProtocol,
        nlpidData                   OCTET STRING
    },
    ...
}

DataProtocolCapability              ::=CHOICESEQUENCE
{
    nonStandard                     NonStandardParameter,
    v14buffered                     NULLBOOLEAN,
    v42lapm                         NULLBOOLEAN,        -- may negotiate to V.42bis
    hdlcFrameTunneling              NULLBOOLEAN,
    transparent                     NULLBOOLEAN,
    ...
}

T84Profile                          ::=CHOICE
{
    t84Unrestricted                 NULL,
    t84Restricted                   SEQUENCE
    {
        qcif                        BOOLEAN,
        cif                         BOOLEAN,
        ccir601Seq                  BOOLEAN,
        ccir601Prog                 BOOLEAN,
        hdtvSeq                     BOOLEAN,
        hdtvProg                    BOOLEAN,

        g3FacsMH200x100             BOOLEAN,
        g3FacsMH200x200             BOOLEAN,
        g4FacsMMR200x100            BOOLEAN,
        g4FacsMMR200x200            BOOLEAN,
        jbig200x200Seq              BOOLEAN,
        jbig200x200Progr            BOOLEAN,
        jbig300x300Seq              BOOLEAN,
        jbig300x300Prog             BOOLEAN,

        digPhotoLow                 BOOLEAN,
        digPhotoMedSeq              BOOLEAN,
        digPhotoMedProg             BOOLEAN,
        digPhotoHighSeq             BOOLEAN,
        digPhotoHighProg            BOOLEAN,

        ...
    }
}
```

```
-- ================================================================================
-- Logical channel signalling definitions
-- ================================================================================

OpenLogicalChannel                  ::=SEQUENCE
{
    logicalChannelNumber            LogicalChannelNumber,

    forwardLogicalChannelParameters SEQUENCELogicalChannelParameters,
    {
        portNumber                  INTEGER (0..65535),
        dataType                    DataType,
        multiplexParameters         CHOICE
        {
            h222LogicalChannelParameters  H222LogicalChannelParameters,
            h223LogicalChannelParameters  H223LogicalChannelParameters,
            ...
        },
        ...
    },
    bidirectionalChannelParameters  CHOICE
    {
        -- Used to open the reverse channel for bi-directional open request
        reverseLogicalChannelParameters  SEQUENCELogicalChannelParameters OPTIONAL,
    -- Used by master to specify requirements of reverse channel for bi-directional open.
        -- Note that H.222 parameters are not present in the reverse direction.
        {
            dataType                    DataType,
            multiplexParameters         CHOICE
            {
                -- H.222 parameters are never present in reverse direction
                h223LogicalChannelParameters   H223LogicalChannelParameters,
                ...
            } OPTIONAL,                 -- Not present for H.222
            ...
        },    -- used when opening the reverse channel of a bi-directional channel

        associatedLogicalChannelNumber  LogicalChannelNumber OPTIONAL,
    -- Used by responding terminalslave to indicate logical channel number of forward channel opened
    -- by initiating terminalmaster.
    }OPTIONAL,      --Not present for uni-directional channel request
    ...
}

LogicalChannelNumber                ::=INTEGER (1..65535)

LogicalChannelParameters            ::=SEQUENCE
{
    portNumber                      INTEGER (0..65535),
    dataType                        DataType,
    logicalChannelMultiplexParameters  LogicalChannelMultiplexParameters,
    ...
}

DataType                            ::=CHOICE
{
    nonStandard                     NonStandardParameter,
    nullData                        NULL,
    videoData                       VideoCapability,
    audioData                       AudioCapability,
    data                            DataApplicationCapability,
    encryptionData                  EncryptionMode,
    ...
}

LogicalChannelMultiplexParameters   ::=CHOICE
{
    h222LogicalChannelParameters    H222LogicalChannelParameters,
```

```
    h223LogicalChannelParameters        H223LogicalChannelParameters,
    ...
}

H222LogicalChannelParameters            ::=SEQUENCE
{
    virtualChannelID                    INTEGER (0..65535),
    subChannelID                        INTEGER (0..8191),
    pcr-pid                             INTEGER (0..8191) OPTIONAL,
    programDescriptors                  OCTET STRING OPTIONAL,
    streamDescriptors                   OCTET STRING OPTIONAL,
    ...
}

H223LogicalChannelParameters            ::=SEQUENCE
{
    adaptationLayerType                 CHOICE
    {
        nonStandard                     NonStandardParameter,
        al1Framed                       NULL,
        al1NotFramed                    NULL,
        al2WithoutSequenceNumbers       NULL,
        al2WithSequenceNumbers          NULL,
        al3                             SEQUENCE
        {
            controlFieldOctets          INTEGER (0..2),
            sendBufferSize              INTEGER (0..16777215)       -- units octets
        },

        ...
    },

    segmentableFlag                     BOOLEAN,
    ...
}


OpenLogicalChannelAck                   ::=SEQUENCE
{
    logicalChannelNumber                LogicalChannelNumber,
    ...
}

OpenLogicalChannelReject                ::=SEQUENCE
{
    logicalChannelNumber                LogicalChannelNumber,
    cause                               CHOICE
    {
        unspecified                     NULL,
        dataTypeNotSupported            NULL,
        dataTypeNotAvailable            NULL,
        unknownDataType                 NULL,
        dataTypeALCombinationNotSupported   NULL,
        ...
    },
    ...
}

CloseLogicalChannel                     ::=SEQUENCE
{
    logicalChannelNumber                LogicalChannelNumber,
    source                              CHOICE
    {
        user                            NULL,
        lcse                            NULL
    },
    ...
}
```

```
CloseLogicalChannelAck                  ::=SEQUENCE
{
    logicalChannelNumber                LogicalChannelNumber,
    ...
}

OpenBiDirectionalChannelRequest         ::= SEQUENCE
{
    sequenceNumber                      SequenceNumber,
    biDirectionalRequests               SET SIZE (1..256) OF OpenLogicalChannel,
    ...
}
-- Used only by the slave to request the master to open bi-directional channels

OpenBiDirectionalChannelAck             ::=SEQUENCE
{
    sequenceNumber                      SequenceNumber,
    ...
}

OpenBiDirectionalChannelReject          ::=SEQUENCE
{
    sequenceNumber                      SequenceNumber,
    cause                               CHOICE
    {
        willOpenSomeChannels            NULL,
        willOpenNoChannels              NULL,
        busy                            NULL,
        ...
    },
    ...
}

OpenBiDirectionalChannelRelease         ::=SEQUENCE
{
    ...
}

RequestChannelClose                     ::=SEQUENCE
{
    logicalChannelNumber                LogicalChannelNumber,
    ...
}

RequestChannelCloseAck                  ::=SEQUENCE
{
    logicalChannelNumber                LogicalChannelNumber,
    ...
}

RequestChannelCloseReject               ::=SEQUENCE
{
    logicalChannelNumber                LogicalChannelNumber,
    cause                               CHOICE
    {
        unspecified                     NULL,
        ...
    },
    ...
}

RequestChannelCloseRelease              ::=SEQUENCE
{
    logicalChannelNumber                LogicalChannelNumber,
    ...
}
```

```
-- ================================================================================
-- H.223 multiplex table definitions
-- ================================================================================

MultiplexEntrySend                  ::=SEQUENCE
{
    sequenceNumber                  SequenceNumber,
    multiplexEntryDescriptors       SET SIZE (1..15) OF MultiplexEntryDescriptor,
    ...
}

MultiplexEntryDescriptor            ::=SEQUENCE
{
    multiplexTableEntryNumber       MultiplexTableEntryNumber,
    elementList                     SEQUENCE SIZE (1..256) OF MultiplexElement OPTIONAL
}

MultiplexElement                    ::=SEQUENCE
{
    type                            CHOICE
    {
        logicalChannelNumber        LogicalChannelNumber,
        subElementList              SEQUENCE SIZE (2..255) OF MultiplexElement
    },
    repeatCount                     CHOICE
    {
        finite                      INTEGER (1..65535),     -- repeats of type
        untilClosingFlag            NULL                    -- used for last element
    }
}

MultiplexTableEntryNumber           ::=INTEGER (1..15)

MultiplexEntrySendAck               ::=SEQUENCE
{
    sequenceNumber                  SequenceNumber,
    multiplexTableEntryNumber       SET SIZE (1..15) OF MultiplexTableEntryNumber,
    ...
}

MultiplexEntrySendReject            ::=SEQUENCE
{
    sequenceNumber                  SequenceNumber,
    rejectionDescriptions           SET SIZE (1..15) OF MultiplexEntryRejectionDescriptions,
    ...
}

MultiplexEntryRejectionDescriptions ::=SEQUENCE
{
    multiplexTableEntryNumber       MultiplexTableEntryNumber,
    cause                           CHOICE
    {
        unspecifiedCause            NULL,
        descriptorTooComplex        NULL,
        ...
    },
    ...
}

MultiplexEntrySendRelease           ::=SEQUENCE
{
    multiplexTableEntryNumber       SET SIZE (1..15) OF MultiplexTableEntryNumber,
    ...
}

RequestMultiplexEntry               ::=SEQUENCE
{
    sequenceNumber                  SequenceNumber,
```

```
    entryNumbers                    SET SIZE (1..15) OF MultiplexTableEntryNumber,
    ...
}

RequestMultiplexEntryResponse       ::=SEQUENCE
{
    sequenceNumber                  SequenceNumber,
    response                        CHOICE
    {
        willSendAllEntries          NULL,
        willSendSomeEntries         NULL,
        willSendNoEntries           NULL,
        ...
    },
    ...
}
```

```
-- ================================================================================
-- Request mode definitions
-- ================================================================================

-- RequestMode is a list, in order or preference, of modes that a terminal would like
-- to have transmitted to it.

RequestMode                         ::=SEQUENCE
{
    sequenceNumber                  SequenceNumber,
    requestedModes                  SEQUENCE SIZE (1..256) OF ModeDescription,
    ...
}

RequestModeAck                      ::=SEQUENCE
{
    sequenceNumber                  SequenceNumber,
    response                        CHOICE
    {
        willTransmitMostPreferredMode   NULL,
        willTransmitLessPreferredMode   NULL,
        ...
    },
    ...
}

RequestModeReject                   ::=SEQUENCE
{
    sequenceNumber                  SequenceNumber,
    cause                           CHOICE
    {
        modeUnavailable             NULL,
        multipointConstraint        NULL,
        requestDenied               NULL,
        ...
    },
    ...
}

RequestModeRelease                  ::=SEQUENCE
{
    ...
}

-- ================================================================================
-- Request mode definitions: Mode description
-- ================================================================================

ModeDescription                     ::=SET SIZE (1..256) OF ModeElement

ModeElement                         ::= SEQUENCE
{
    type                            CHOICE
    {
        nonStandard                 NonStandardParameter,
        videoMode                   VideoMode,
        audioMode                   AudioMode,
        dataMode                    DataMode,
        encryptionMode              EncryptionMode,
        ...
    },

    h223ModeParameters              H223ModeParameters OPTIONAL,

    ...
}

H223ModeParameters                  ::=SEQUENCE
```

```
{
    adaptationLayerType                 CHOICE
    {
        nonStandard                     NonStandardParameter,
        al1Framed                       NULL,
        al1NotFramed                    NULL,
        al2WithoutSequenceNumbers       NULL,
        al2WithSequenceNumbers          NULL,
        al3                             SEQUENCE
        {
            controlFieldOctets          INTEGER(0..2),
            sendBufferSize              INTEGER(0..16777215)        -- units octets
        },

        ...
    },

    segmentableFlag                     BOOLEAN,
    ...
}

-- ================================================================================
-- Request mode definitions: Video modes
-- ================================================================================

VideoMode                               ::=CHOICE
{
    nonStandard                         NonStandardParameter,
    h261VideoMode                       H261VideoMode,
    h262VideoMode                       H262VideoMode,
    h263VideoMode                       H263VideoMode,
    ...
}

H261VideoMode                           ::=SEQUENCE
{
    resolution                          CHOICE
    {
        qcif                            NULL,
        cif                             NULL
    },
    ...
}

H262VideoMode                           ::=SEQUENCE
{
    profileAndLevel                     CHOICE
    {
        profileAndLevel-SPatML          NULL,
        profileAndLevel-MPatLL          NULL,
        profileAndLevel-MPatML          NULL,
        profileAndLevel-MPatH-14        NULL,
        profileAndLevel-MPatHL          NULL,
        profileAndLevel-SNRatLL         NULL,
        profileAndLevel-SNRatML         NULL,
        profileAndLevel-SpatialatH-14   NULL,
        profileAndLevel-HPatML          NULL,
        profileAndLevel-HPatH-14        NULL,
        profileAndLevel-HPatHL          NULL,
        ...
    },
    videoBitRate                        INTEGER(0..1073741823) OPTIONAL,    -- units 400bits/sec
    vbvBufferSize                       INTEGER(0..262143) OPTIONAL,        -- units 16384bits
    samplesPerLine                      INTEGER(0..16383) OPTIONAL,         -- units samples/line
    linesPerFrame                       INTEGER(0..16383) OPTIONAL,         -- units lines/frame
    framesPerSecond                     INTEGER(0..15) OPTIONAL,            -- frame_rate_code
    luminanceSampleRate                 INTEGER(0..4294967295) OPTIONAL,    -- units samples/sec
    ...
```

```
}

H263VideoMode                   ::=SEQUENCE
{
    resolution                  CHOICE
    {
        sqcif                   NULL,
        qcif                    NULL,
        cif                     NULL,
        cif4                    NULL,
        cif16                   NULL,
        ...
    },
    unrestrictedVector          BOOLEAN,
    arithmeticCoding            BOOLEAN,
    advancedPrediction          BOOLEAN,
    pbFrames                    BOOLEAN,
    ...
}

-- ================================================================================
-- Request mode definitions: Audio modes
-- ================================================================================

AudioMode::=CHOICE
{
    nonStandard                 NonStandardParameter,
    g711Alaw64k                 NULL,
    g711Alaw56k                 NULL,
    g711Ulaw64k                 NULL,
    g711Ulaw56k                 NULL,

    g722-64k                    NULL,
    g722-56k                    NULL,
    g722-48k                    NULL,

    g728                        NULL,
    g729                        NULL,
    g-dsvd                      NULL,

    g723                        CHOICE
    {
        noSilenceSuppressionLowRate     NULL,
        noSilenceSuppressionHighRate    NULL,
        silenceSuppressionLowRate       NULL,
        silenceSuppressionHighRate      NULL
    },

    is11172AudioMode            IS11172AudioMode,
    is13818AudioMode            IS13818AudioMode,

    ...
}

IS11172AudioMode                ::=SEQUENCE
{
    audioLayer                  CHOICE
    {
        audioLayer1             NULL,
        audioLayer2             NULL,
        audioLayer3             NULL
    },

    audioSampling               CHOICE
    {
        audioSampling32k        NULL,
        audioSampling44k1       NULL,
        audioSampling48k        NULL
```

```
    },

    multichannelType                    CHOICE
    {
        singleChannel                   NULL,
        twoChannels                     NULL
    },

    bitRate                             INTEGER (1..448),       --units kbit/sec
    ...
}

IS13818AudioMode                        ::=SEQUENCE
{
    audioLayer                          CHOICE
    {
        audioLayer1                     NULL,
        audioLayer2                     NULL,
        audioLayer3                     NULL
    },

    audioSampling                       CHOICE
    {
        audioSampling16k                NULL,
        audioSampling22k05              NULL,
        audioSampling24k                NULL,
        audioSampling32k                NULL,
        audioSampling44k1               NULL,
        audioSampling48k                NULL
    },

    multichannelType                    CHOICE
    {
        singleChannel                   NULL,
        twoChannels                     NULL,
        threeChannels2-1                NULL,
        threeChannels3-0                NULL,
        fourChannels2-0-2-0             NULL,
        fourChannels2-2                 NULL,
        fourChannels3-1                 NULL,
        fiveChannels3-0-2-0             NULL,
        fiveChannels3-2                 NULL
    },

    lowFrequencyEnhancement             BOOLEAN,

    multilingual                        BOOLEAN,

    bitRate                             INTEGER (1..1130),      --units kbit/sec
    ...
}
-- This is to be specified.

-- ================================================================================
-- Request mode definitions: Data modes
-- ================================================================================

DataMode                                ::=CHOICE
{
    nonStandard                         NonStandardParameter,
    t120                                DataModeProtocol,
    dsm-cc                              DataModeProtocol,
    userData                            DataModeProtocol,
    t84                                 DataModeProtocol,
    t434                                DataModeProtocol,
    h224                                DataModeProtocol,
    nlpid                               SEQUENCE
    {
```

```
        nlpidProtocol                   DataModeProtocol,
        nlpidData                       OCTET STRING
    },
    ...
}

DataModeProtocol                        ::= CHOICE
{
    nonStandard                         NonStandardParameter,
    v14buffered                         NULL,
    v42lapm                             NULL,
    hdlcFrameTunneling                  NULL,
    transparent                         NULL,
    ...
}

-- ================================================================================
-- Request mode definitions: Encryption modes
-- ================================================================================

EncryptionMode                          ::=CHOICE
{
    nonStandard                         NonStandardParameter,
    h233Encryption                      NULL,
    ...
}
```

```
-- ================================================================================
-- Round Trip Delay definitions
-- ================================================================================

RoundTripDelayRequest                ::=SEQUENCE
{
    sequenceNumber                   SequenceNumber,
    ...
}

RoundTripDelayResponse               ::=SEQUENCE
{
    sequenceNumber                   SequenceNumber,
    ...
}

-- ================================================================================
-- Maintenance Loop definitions
-- ================================================================================

MaintenanceLoopRequest               ::=SEQUENCE
{
    sequenceNumber                   SequenceNumber,
    type                             CHOICE
    {
        systemLoop                   NULL,
        mediaLoop                    LogicalChannelNumber,
        logicalChannelLoop           LogicalChannelNumber,
        ...
    },
    ...
}

MaintenanceLoopResponse              ::=SEQUENCE
{
    sequenceNumber                   SequenceNumber,
    response                         CHOICE
    {
        willPerformLoop              NULL,
        willNotPerformLoop           NULL,
        ...
    },
    ...
}

MaintenanceLoopOffCommand            ::=SEQUENCE
{
    ...
}
```

```
-- ================================================================================
-- Command PDU definitions
-- ================================================================================

-- ================================================================================
-- Command PDU : Send Terminal Capability Set
-- ================================================================================

SendTerminalCapabilitySet               ::=CHOICESEQUENCE
{
    specificRequest                     SEQUENCE
    {
        multiplexCapability             BOOLEAN,

        capabilityTableEntryNumbers     SET SIZE (1..65535256) OF CapabilityTableEntryNumber OPTIONAL,

        capabilityDescriptorNumbers     SET SIZE (1..256) OF CapabilityDescriptorNumber OPTIONAL,
        ...
    },
    genericRequest                      NULL,
    ...
}

-- ================================================================================
-- Command PDU : Encryption
-- ================================================================================

EncryptionCommand                       ::=CHOICE
{
    encryptionSE                        OCTET STRING,    -- per H.233, but no error protection
    encryptionIVRequest                 NULL,            -- requests new IV
    encryptionAlgorithmID               SEQUENCE
    {
        h233AlgorithmIdentifier         SequenceNumber,
        associatedAlgorithm             NonStandardParameter
    },
    ...
}

-- ================================================================================
-- Command PDU : Flow Control
-- ================================================================================

FlowControlCommand                      ::=SEQUENCE
{
    scope                               CHOICE
    {
        logicalChannelNumber            LogicalChannelNumber,
        virtualChannelID                INTEGER (0..65535),
        wholeMultiplex                  NULL
    },
    restriction                         CHOICE
    {
        maximumBitRate                  INTEGER (0..16777215),      -- units 100 bits per second
        noRestriction                   NULL
    },
    ...
}

-- ================================================================================
-- Command PDU : Change or End Session
-- ================================================================================

EndSessionCommand                       ::=CHOICE
{
    nonStandard                         NonStandardParameter,

    disconnect                          NULL,
```

```
    v34Options                      CHOICE
    {
        telephonyMode               NULL,
        v34DSVD                     NULL,
        v34DuplexFAX                NULL,
        v34H324                     NULL,
        ...
    },

    ...
}

-- ================================================================================
-- Command PDU : Miscellaneous H.230-like commands
-- ================================================================================

MiscellaneousCommand                ::=SEQUENCE
{
    logicalChannelNumber            LogicalChannelNumber,
    type                            CHOICE
    {
        equaliseDelay               NULL,                   -- like H.230 ACE
        zeroDelay                   NULL,                   -- like H.230 ACZ

        videoFreezePicture          NULL,
        videoFastUpdatePicture      NULL,

        videoFastUpdateGOB          SEQUENCE
        {
            firstGOB                INTEGER (0..17),
            numberOfGOBs            INTEGER (1..18)
        },

        videoTemporalSpatialTradeOff    INTEGER (0..31),    -- commands a trade-off value

        videoSendSyncEveryGOB       NULL,
        videoSendSyncEveryGOBCancel NULL,

        ...
    },

    ...
}
```

```
-- ================================================================================
-- Indication PDU definitions
-- ================================================================================

-- ================================================================================
-- Indication PDU : Function not supported
-- ================================================================================

-- This is used to return a complete request, response or command that is not recognised

FunctionNotSupported                ::=CHOICE
{
    request                         RequestPDU,
    response                        ResponsePDU,
    command                         CommandPDU
}

-- ================================================================================
-- Indication PDU : Miscellaneous H.230-like indication
-- ================================================================================

MiscellaneousIndication             ::=SEQUENCE
{
    logicalChannelNumber            LogicalChannelNumber,
    type                            CHOICE
    {
        logicalChannelActive        NULL,           -- like H.230 AIA and VIA
        logicalChannelInactive      NULL,           -- like H.230 AIM and VIS

        multipointConference        NULL,           -- like H.230 MCIC
        cancelMultipointConference  NULL,           -- like H.230 cancel MCIC

        multipointZeroComm          NULL,           -- like H.230 MIZ
        cancelMultipointZeroComm    NULL,           -- like H.230 cancel MIZ

        multipointSecondaryStatus   NULL,           -- like H.230 MIS
        cancelMultipointSecondaryStatus NULL,       -- like H.230 cancel MIS

        videoIndicateReadyToActivate NULL,          -- like H.230 VIR

        videoTemporalSpatialTradeOff INTEGER (0..31), -- indicates current trade-off

        ...
    },
    ...
}

-- ================================================================================
-- Indication PDU : Jitter Indication
-- ================================================================================

JitterIndication                    ::=SEQUENCE
{
    scope                           CHOICE
    {
        logicalChannelNumber        LogicalChannelNumber,
        virtualChannelID            INTEGER (0..65535),
        wholeMultiplex              NULL
    },
    estimatedReceivedJitterMantissa INTEGER (0..3),
    estimatedReceivedJitterExponent INTEGER (0..7),
    skippedFrameCount               INTEGER (0..15) OPTIONAL,
    additionalDecoderBuffer         INTEGER (0.. 262143) OPTIONAL,      -- 262143 is 2^18 -
1
    ...
}

-- ================================================================================
```

```
-- Indication PDU : H.223 logical channel skew
-- ========================================================================

H223SkewIndication                  ::=SEQUENCE
{
    logicalChannelNumber1           LogicalChannelNumber,
    logicalChannelNumber2           LogicalChannelNumber,
    skew                            INTEGER (0..4095),          -- units milliseconds
    ...
}

-- ========================================================================
-- Indication PDU : user input
-- ========================================================================

UserInputIndication                 ::=CHOICE
{
    nonStandard                     NonStandardParameter,
    alphanumeric                    GeneralString,
    ...
}


END
```

77 Messages: semantic definitions

This section provides semantic definitions and constraints on the syntax elements defined in the previous section.

MultimediaSystemControlPDU: is a choice of message types. Messages defined in this Recommendation are classified as request, response, command and indication messages.

RequestPDU: a request message results in an action by the remote terminal and requires an immediate response from it. The nonStandardRequest may be used to send non-standard requests.

ResponsePDU: a response message is the response to a request message. The nonStandardResponse may be used to send non-standard responses.

CommandPDU: a command message requires action but no explicit response. The nonStandardCommand may be used to send non-standard commands.

IndicationPDU: an indication contains information that does not require action or response. The nonStandardIndication may be used to send non-standard indications.

NonStandardParameter: this may be used to indicate a non standard parameter. It consists of an identity and the actual parameters, which are coded as an octet string.

NonStandardIdentifier: is used to identify the type of non-standard parameter. It is either an object identifier, or an H.221 type of identifier that is an octet string consisting of exactly four octets which are country code (octet 1 as in T.35 [2222]; octet 2*), manufacturer code (next two octets*), *=assigned nationally. The manufacturer codes are the same as those assigned for use in H.320 [1717].

77.11 Master Slave Determination messages

This set of messages is used by a protocol to determine which terminal is the master terminal and which is the slave terminal.

77.11.11 Master Slave Determination

This is sent from a MSDSE to a peer MSDSE.

statusDeterminationNumber is a random number.

77.11.22 Master Slave Determination Acknowledge

This is used to confirm whether the terminal is the master terminal or the slave terminal, as indicated by decision. When decision is of type master, the terminal receiving this message is the master terminal and when decision is of type slave, it is the slave terminal.

77.11.33 Master Slave Determination Reject

This is used to reject the MasterSlaveDetermination message. When the cause is of type identicalNumbers, the rejection was due to both random numbers being the same.

7.22 Terminal capability messages

This set of messages is for the secure exchange of capabilities between the two terminals.

7.22.1 Overview

The transmitting terminal assigns each individual mode the terminal is capable of operating in a number in a capabilityTable. For example, G.723 audio, G.728 audio, and CIF H.263 video would each be assigned separate numbers.

These capability numbers are grouped into AlternativeCapabilitySet structures. Each AlternativeCapabilitySet indicates that the terminal is capable of operating in exactly one mode listed in the set. For example, an AlternativeCapabilitySet listing {G.711, G.723, G.728} means that the terminal can operate in any one of those audio modes, but not more than one.

These AlternativeCapabilitySet structures are grouped into simultaneousCapabilities structures. Each simultaneousCapabilities structure indicates a set of modes the terminal is capable of using simultaneously. For example, a simultaneousCapabilities structure containing the two AlternativeCapabilitySet structures {H.261, H.263} and {G.711, G.723, G.728} means that the terminal can operate either of the video codecs simultaneously with any one of the audio codecs. The simultaneousCapabilities set {{H.261}, {H.261, H.263}, {G.711, G.723, G.728} } means the terminal can operate two video channels and one audio channel simultaneously: one video channel per H.261, another video channel per either H.261 or H.263, and one audio channel per either G.711, G.723, or G.728.

Note: the actual capabilities stored in the capabilityTable are often more complex than presented here. For example, each H.263 capability indicates details including ability to support various picture formats at given minimum picture intervals, and ability to use optional coding modes.

The terminal's total capabilities are described by a set of CapabilityDescriptor structures, each of which is a single simultaneousCapabilities structure and a capabilityDescriptorNumber. By sending more than one CapabilityDescriptor, the terminal may signal dependencies between operating modes by describing different sets of modes which it can simultaneously use. For example, a terminal issuing two CapabilityDescriptor structures, one {{H.261, H.263}, {G.711, G.723, G.728} } as in the previous example, and the other { {H.262}, {G.711} }, means the terminal can also operate the H.262 video codec, but only with the low-complexity G.711 audio codec.

Terminals may dynamically add capabilities during a communication session by issuing additional CapabilityDescriptor structures, or remove capabilities by sending revised CapabilityDescriptor structures. All terminals shall transmit at least one CapabilityDescriptor structure.

7.22.2 Terminal Capability Set

This message contains information about the terminal's capability to transmit and receive. It also indicates the version of this Recommendation that is in use. It is sent from an out-going CESE to a peer in-coming CESE.

sequenceNumber is used to label instances of TerminalCapabilitySet so that the corresponding response can be identified.

protocolIdentifier is used to indicate the version of this Recommendation that is in use. Annex A lists the object identifiers defined for use by this Recommendation.

multiplexCapability indicates capabilities relating to multiplexing and network adaptation. A terminal shall include multiplexCapability in the first TerminalCapabilitySet sent.at least one TerminalCapabilitySet, including the first one that it transmits.

7.22.2.1 Capability Table

A capability table is a numbered list of capabilities. A terminal shall be capable of everything that it lists in its capability table, but shall not necessarily be capable of simultaneously performing more than one of them.

A TerminalCapabilitySet may contain zero or more CapabilityTableEntrys. At the start, no table entries are defined. When a CapabilityTableEntry is received, it replaces the previously received CapabilityTableEntry with the same CapabilityTableEntryNumber. A CapabilityTableEntry without a Capability may be used to remove the previously received CapabilityTableEntry with the same CapabilityTableEntryNumber.

7.22.2.2 Capability Descriptors

CapabilityDescriptors are used to indicate a terminal's capability to transmit and receive. Each CapabilityDescriptor provides an independent statement about the terminal's capabilities.

capabilityDescriptorNumber is used to number CapabilityDescriptors. If a terminal has a preference for the mode it would like to transmit or receive, and wishes to express this when transmitting its capabilities, it may do so by giving CapabilityDescriptors that relate to its preferred mode or modes small values of capabilityDescriptorNumber.

simultaneousCapabilities is a set of AlternativeCapabilitySet. It is used to list the simultaneous capabilities of the terminal.

An AlternativeCapabilitySet is a sequence of CapabilityTableEntryNumbers. Only those CapabilityTableEntrys that have been defined shall be present in an AlternativeCapabilitySet, although it is possible to define CapabilityTableEntrys and refer to them in the same TerminalCapabilitySet. If a terminal has a preference for the mode it would like to transmit or receive, and wishes to express this when transmitting its capabilities, it may do so by listing elements in AlternativeCapabilitySets in order of decreasing preference.

A terminal shall be capable of simultaneously performing any one capability from each AlternativeCapabilitySet listed in simultaneousCapabilities.

At least one capability descriptor shall have the following structure: there shall be at least one AlternativeCapabilitySet containing only capabilities of a single medium type for each medium type that the terminal can support. This is to ensure that the remote terminal can select a mode of transmission that includes at least one instance of each medium type that the receiver can support.

Note: a repetition of a capability in an AlternativeCapabilitySet is redundant and conveys no further information, while the repetition of a capability in different AlternativeCapabilitySets in the same CapabilityDescriptor indicates the possibility of an additional, simultaneous, instance of the particular capability.

Note: terminals that can not vary the allocation of resources can indicate their capability completely by use of a single CapabilityDescriptor.

~~7.22.22.33~~ Capability

The choices receiveVideoCapability, receiveAudioCapability and receiveDataApplicationCapability indicate the capability to receive according to the respective VideoCapability, AudioCapability and DataApplicationCapability.

The choices transmitVideoCapability, transmitAudioCapability and transmitDataApplicationCapability indicate the capability to transmit according to the respective VideoCapability, AudioCapability and DataApplicationCapability.

The choices receiveAndTransmitVideoCapability, receiveAndTransmitAudioCapability and receiveAndTransmitDataApplicationCapability indicate the capability to receive and transmit according to the respective VideoCapability, AudioCapability and DataApplicationCapability. These code points may be useful for indicating that the receive and transmit capabilities are not independent.

The boolean h233EncryptionTransmitCapability, when true, indicates that the terminal supports encryption according to H.233 and H.234 [11~~+~~][12~~+~~].

h233IVResponseTime is measured in units of milliseconds, and indicates the minimum time the receiver requires the transmitter to wait after the completion of transmission of an IV message before starting to use the new IV. The means of transmitting the IV is not defined in this Recommendation.

~~7.22.22.44~~ Multiplex Capabilities

MultiplexCapability indicates capabilities relating to multiplexing and network adaptation. A terminal shall send MultiplexCapability in at least one TerminalCapabilitySet, including the first one that it transmits. the first TerminalCapability sent.

Unless stated otherwise, these are capabilities to receive.

H222Capability: indicates multiplexing and network adaptation capabilities that are specific to the multiplex defined in H.222.1 [7~~7~~].

numberOfVCs indicates how many simultaneous ATM Virtual Channels (VCs) can be supported by the terminal. This includes any VCs that transport H.245, T.120, DSM-CC or any other data, and all VCs that carry audiovisual information. It does not include the VC used for Q.2931 signalling [20~~20~~].

vcCapability is a set, of size equal to the value of numberOfVCs, that indicates the capabilities present for each available VC.

The sequence aal1 indicates which of the options for ATM adaptation layer 1, as specified in I.363 [19~~19~~], are supported. The codepoints are defined in Table 1~~4~~.

TABLE 11/H.245

ATM Adaptation Layer 1 codepoints

ASN.1 Codepoint	Semantic meaning of codepoint
nullClockRecovery	Null source clock frequency recovery method: synchronous circuit transport.
srtsClockRecovery	Synchronous residual timestamp source clock frequency recovery method.
adaptiveClockRecovery	Adaptive clock source clock frequency recovery method.
nullErrorCorrection	No error correction is supported.
longInterleaver	The forward error correction method for loss sensitive signal transport is supported.
shortInterleaver	The forward error correction method for delay sensitive signal transport is supported.
errorCorrectionOnly	The forward error correction method without cell interleaving is supported.
structuredDataTransfer	Structured data transfer is supported.
partiallyFilledCells	Partially filled cells is supported.

The sequence aal5 indicates which of the options for ATM adaptation layer 5, as specified in I.363 [1949], are supported. forwardMaximumSDUSize and backwardMaximumSDUSize indicate the maximum CPCS-SDU size in the forward and reverse directions, measured in octets.

The booleans transportStream and programStream, when equal to true, indicate the capability to support the Transport Stream and Program Stream multiplexes respectively [66]; and bitRate indicates the maximum bit rate capability for the VC.

H223Capability: indicates capabilities specific to the H.223 multiplex [88].

The boolean transportWithI-frames, when true, indicates that the terminal is capable of sending and receiving control channel messages using LAPM I-frames as defined in V.42 [2929].

The booleans videoWithAL1, videoWithAL2, videoWithAL3, audioWithAL1, audioWithAL2, audioWithAL3, dataWithAL1, dataWithAL2 and dataWithAL3, when true, indicate the capability to receive the stated medium type (video, audio, or data) using the stated adaptation layer (AL1, AL2, or AL3).

The integers maximumAl2SDUSize and maximumAl3SDUSize indicate the maximum number of octets in each SDU that the terminal can receive when using adaptation layer types 2 and 3 respectively.

maximumDelayJitter indicates the maximum peak-to-peak multiplexing jitter that the transmitter shall cause. It is measured in milliseconds. Multiplexing jitter is defined as the difference in time of delivery of the first octet of an audio frame when delivered in the multiplexed stream and when it would be delivered at constant bit rate without a multiplex.

h223MultiplexTableCapability: indicates the terminals ability to receive and process multiplex table entries.

basic indicates that the multiplex can only receive basic MultiplexEntryDescriptors as defined in H.223 [8].

enhanced indicates that the multiplex can receive enhanced MultiplexEntryDescriptors with the additional parameters defined below.

maximumNesting depth indicates the maximum nesting depth of recursively invoked subElementList fields. MultiplexEntryDescriptors which do not use the subElementList field shall be considered to have a nesting depth of zero.

maximumElementListSize indicates the maximum number of fields in the ASN.1 SEQUENCE.

maximumSubElementListSize indicates the maximum number of subelements in the SubElementList.

7.22.22.55 Video Capabilities

This indicates video capabilities. The indication of more than a single capability within a single VideoCapability does not indicate simultaneous processing capability. Simultaneous processing capability can be indicated by instances of VideoCapability in different AlternativeCapabilitySets in a single CapabilityDescriptor.

H261VideoCapability: indicates H.261 [1313] capabilities.

If present, qcifMPI indicates the minimum picture interval in units of 1/29.97 for the encoding and/or decoding of QCIF pictures, and if not present, no capability for QCIF pictures is indicated.

If present, cifMPI indicates the minimum picture interval in units of 1/29.97 for the encoding and/or decoding of CIF pictures, and if not present, no capability for CIF pictures is indicated.

The boolean temporalSpatialTradeOffCapability, when true, indicates that the encoder is able to vary its trade-off between temporal and spatial resolution as commanded by the remote terminal. It has no meaning when part of a receive capability.

H262VideoCapability: indicates H.262 [14] capabilities.

The list of booleans indicate the capability of processing the particular profiles and levels: a value of true indicates that such operation is possible, while a value of false indicates that such operation is not possible. An encoder shall produce bitstreams compliant to the specifications of a profile and level for which it has indicated capability, but also within the limitations imposed by the optional fields (see below). A decoder shall be able to accept all bit streams conforming to a profile and level for which it has indicated capability, provided it is within the limitations indicated by the optional fields. The optional fields are integers with units defined in Table 22.

TABLE 22/H.245

Units for H.262 codepoints

ASN.1 Codepoint	Units for referenced parameter
videoBitRate	400 bits per second
vbvBufferSize	16384 bits
samplesPerLine	samples per line
linesPerFrame	lines per frame
framesPerSecond	The index, frame_rate_code, into table 6-4/H.262
luminanceSampleRate	samples per second

H263VideoCapability: indicates H.263 [15] capabilities.

If present, sqcifMPI indicates the minimum picture interval in units of 1/29.97 for the encoding and/or decoding of SQCIF pictures, and if not present, no capability for SQCIF pictures is indicated.

If present, qcifMPI indicates the minimum picture interval in units of 1/29.97 for the encoding and/or decoding of QCIF pictures, and if not present, no capability for QCIF pictures is indicated.

If present, cifMPI indicates the minimum picture interval in units of 1/29.97 for the encoding and/or decoding of CIF pictures, and if not present, no capability for CIF pictures is indicated.

If present, cif4MPI indicates the minimum picture interval in units of 1/29.97 for the encoding and/or decoding of 4CIF pictures, and if not present, no capability for 4CIF pictures is indicated.

If present, cif16MPI indicates the minimum picture interval in units of 1/29.97 for the encoding and/or decoding of 16CIF pictures, and if not present, no capability for 16CIF pictures is indicated.

The booleans unrestrictedVector, arithmeticCoding, advancedPrediction, and pbFrames, when true, indicate the capability to transmit and/or receive these optional modes defined in the annexes of H.263.

The boolean temporalSpatialTradeOffCapability, when true, indicates that the encoder is able to vary its trade-off between temporal and spatial resolution as commanded by the remote terminal. It has no meaning when part of a receive capability.

The integer hrd-BmaxKb, when present, indicates the available HRD buffer, and is measured in units of 1024 bits. When not present, the default value defined in H.263 applies.

The values of MPI are applicable when all of the optional modes, for which capability is indicated, are being used, as well as when any combination of them is used. A terminal may signal the capability for a smaller MPI when some options are not used by transmitting another VideoCapability including this smaller MPI and indicating the reduced set of options.

7.22.22.66 Audio Capabilities

This indicates audio capabilities. The indication of more than a single capability within a single AudioCapability does not indicate simultaneous processing capability. Simultaneous processing capability can be indicated by instances of AudioCapability in different AlternativeCapabilitySets in a single CapabilityDescriptor.

The capability to transmit and/or receive G-series audio is indicated by a choice of integers. When an H.222.1 multiplex is used, these numbers refer to the available STD buffer size in units of 256 octets. When an H.223 multiplex is used, these numbers refer to the maximum number of audio frames per AL-SDU. The exact meaning of the codepoints is given in Table 33.

TABLE 33/H.245

G-series audio codepoints

ASN.1 Codepoint	Semantic meaning of codepoint
g711Alaw64k	G.711 audio at 64 kbit/s, A-law
g711Alaw56k	G.711 audio at 56 kbit/s, A-law, truncated to 7 bits
g711Ulaw64k	G.711 audio at 64 kbit/s, μ-law
g711Ulaw56k	G.711 audio at 56 kbit/s, μ-law, truncated to 7 bits
g722-64k	G.722 7 KHz audio at 64 kbit/s
g722-56k	G.722 7 KHz audio at 56 kbit/s
g722-48k	G.722 7 KHz audio at 48 kbit/s
g723	G.723 at either 5.3 or 6.4 kbit/s
g728	G.728 audio at 16 kbit/s
g729	G.729 audio at 8 kbit/s
g-dsvd	G.dsvd audio (under development)

IS11172AudioCapability: indicates the ability to process audio coded according to ISO/IEC 11172-3 [3333].

Booleans that have the value of true indicate that the particular mode of operation is possible, while a value of false indicates that it is not. The booleans audioLayer1, audioLayer2 and audioLayer3 indicate which audio coding layers can be processed. The booleans audioSampling32k, audioSampling44k1 and audioSampling48k indicate which of the audio sample rates, 32KHz, 44.1KHz and 48KHz respectively, can be processed. The booleans singleChannel and twoChannels indicate capability for single channel and stereo/dual channel operation respectively. The integer bitRate indicates the maximum audio bit rate capability, and is measured in units of kbits per second.

IS13818AudioCapability: indicates the ability to process audio coded according to ISO/IEC 13818-3 [3434].

Booleans that have the value of true indicate that the particular mode of operation is possible, while a value of false indicates that it is not. The booleans audioLayer1, audioLayer2 and audioLayer3 indicate which audio coding layers can be processed. The booleans audioSampling16k, audioSampling22k05, audioSampling24k, audioSampling32k, audioSampling44k1 and audioSampling48k indicate which of the audio sample rates, 16KHz, 22.05KHz, 24KHz, 32KHz, 44.1KHz and 48KHz respectively, can be processed.

The booleans concerned with multi-channel operation indicate capability to operate in the particular modes, as specified in Table 44.

TABLE 44/H.245

ISO/IEC 13818-3 multi-channel codepoints

ASN.1 Codepoint	Semantic meaning of codepoint
singleChannel	One channel, using the 1/0 configuration. Single channel mode (as in ISO/IEC 11172-3)
twoChannels	Two channels, using the 2/0 configuration. Stereo or dual channel mode (as in ISO/IEC 11172-3)
threeChannels2-1	Three channels, using the 2/1 configuration. Left, Right and single surround channel
threeChannels3-0	Three channels, using the 3/0 configuration. Left, Centre and Right, without surround channel
fourChannels2-0-2-0	Four channels, using the 2/0 + 2/0 configuration. Left and Right of the first programme and Left and Right of the second programme
fourChannels2-2	Four channels, using the 2/2 configuration. Left, Right, Left surround and Right surround
fourChannels3-1	Four channels, using the 3/1 configuration. Left, Centre, Right, and a single surround channel
fiveChannels3-0-2-0	Five channels, using the 3/0 + 2/0 configuration. Left, Centre and Right of the first programme and Left and Right of the second programme
fiveChannels3-2	Five channels, using the 3/2 configuration. Left, Centre, Right, Left surround and Right surround

The boolean lowFrequencyEnhancement indicates the capability for a low frequency enhancement channel.

The boolean multilingual, when true, indicates the capability to support up to seven multilingual channels, and when false that no multilingual channel is supported.

The integer bitRate indicates the maximum audio bit rate capability, and is measured in units of kbits per second.

7.22.22.7 Data Application Capabilities

This indicates data capabilities. The indication of more than a single capability within a single DataApplicationCapability does not indicate simultaneous processing capability. Simultaneous processing capability can be indicated by instances of DataApplicationCapability in different AlternativeCapabilitySets in a single CapabilityDescriptor.

Recommendations that use this Recommendation may place restrictions on which of these modes may be signalled.

Some of the data capabilities require bi-directional logical channels, for example, to run a retransmission protocol. This requirement is implicitly included in the appropriate capability codepoints.

DataApplicationCapability: is a list of data applications. Each data application indicated shall be supported by one or more DataCapProtocols.

t120 indicates the capability to support the T.120 [25~~25~~] protocol.

dsm-cc indicates the capability to support the DSM-CC [35~~35~~] protocol.

userData indicates the capability to support unspecified user data from external data ports.

t84 indicates the capability to support the transfer of T.84 [24~~24~~] type images (JPEG, JBIG, Facsimile Gr.3/4).

t434 indicates the capability to support the transfer of T.434 [26~~26~~] telematic binary files.

h224 indicates the capability to support the real-time simplex device control protocol H.224 [99].

nlpid indicates the capability to support the network layer protocol as specified by nlpidData as defined in ISO/IEC TR9577 [36~~36~~]. These protocols include Internet protocol (IP) and IETF Point-to-Point protocol (PPP), among others.

Note: the use of the NLPID is extensively described in IETC RFC1490, 'Multiprotocol Interconnect over Frame Relay'.

DataProtocolCapability: contains a list of data protocols.

v14buffered indicates the capability to support a specified data application using buffered V.14 [2727].

v42lapm indicates the capability to support a specified data application using the LAPM protocol defined in V.42 [2929].

hdlcFrameTunneling indicates the capability to support a specified data application using HDLC Frame Tunneling. Refer to clause 4.5.2 of ISO/IEC 3309 [3232].

transparent indicates the capability to support a specified data application using transparent data transfer.

T84Profile: indicates the types of still image profile that the terminal is able to support

T84Unrestricted provides no indication of the type of T.84 still image that the terminal is able to support: information in the T.84 layer should be used to determine whether a particular image can be received.

T84Restricted indicates the type of T.84 still image that the terminal is able to support.

qcif indicates the support of a sequential colour YCrCb type image with QCIF resolution

cif indicates the support of a sequential colour YCrCb type image with CIF resolution

ccir601Seq indicates the support of a sequential colour YCrCb type image with CCIR601 resolution

ccir601Prog indicates the support of a progressive colour YCrCb type image with CCIR601 resolution

hdtvSeq indicates the support of a sequential colour YCrCb type image with HDTV resolution

hdtvProg indicates the support of a progressive colour YCrCb type image with HDTV resolution

g3FacsMH200x100 indicates the support of a sequential Facsimile Gr. 3 MH (Modified Huffman) coded bi-level image at the normal (200x100ppi) resolution

g3FacsMH200x200 indicates the support of a sequential Facsimile Gr. 3 MH (Modified Huffman) coded bi-level image at the high (200x200ppi) resolution

g4FacsMMR200x100 indicates the support of a sequential Facsimile Gr. 4 MMR (Modified Modified Reed) coded bi-level image at the normal (200x100ppi) resolution

g4FacsMMR200x200 indicates the support of a sequential Facsimile Gr. 4 MMR (Modified Modified Reed) coded bi-level image at the high (200x200ppi) resolution

jbig200x200Seq indicates the support of a sequential bi-level JBIG coded bi-level image at the 200x200ppi resolution

jbig200x200Progr indicates the support of a progressive bi-level JBIG coded bi-level image at the 200x200ppi resolution

jbig300x300Seq indicates the support of a sequential bi-level JBIG coded bi-level image at the 300x300ppi resolution

jbig300x300Prog indicates the support of a progressive bi-level JBIG coded bi-level image at the 300x300ppi resolution

digPhotoLow indicates the support of a sequential JPEG coded colour image of 1400x1000ppi {to be checked for exact figures...} resolution

digPhotoMedSeq indicates the support of a sequential JPEG coded colour image of ..00x..00ppi {to be checked for exact figures...} resolution

digPhotoMedProg indicates the support of a progressive JPEG coded colour image of ..00x..00ppi {to be checked for exact figures...} resolution

digPhotoHighSeq indicates the support of a sequential JPEG coded colour image of ..00x..00ppi {to be checked for exact figures...} resolution

digPhotoHighProg indicates the support of a progressive JPEG coded colour image of ..00x..00ppi {to be checked for exact figures...} resolution

{Ed. There are resolutions to be included here.}

77.22.33 Terminal Capability Set Acknowledge

This is used to confirm receipt of a TerminalCapabilitySet from the peer CESE.

The sequenceNumber shall be the same as the sequenceNumber in the TerminalCapabilitySet for which this is the confirmation.

7.22.4 Terminal Capability Set Reject

This is used to reject a TerminalCapabilitySet from the peer CESE.

The sequenceNumber shall be the same as the sequenceNumber in the TerminalCapabilitySet for which this is the negative acknowledgement.

The reasons for sending this message are given in Table 5.

TABLE 5/H.245

Reasons for rejecting a TerminalCapabilitySet

ASN.1 codepoint	Cause
unspecified	No cause for rejection specified.
undefinedTableEntryUsed	A capability descriptor made reference to a capabilityTable entry that is not defined.
descriptorCapacityExceeded	The terminal was incapable of storing all of the information in the TerminalCapabilitySet.
descriptorCapacityExceededtableEntryCapacityExceed	The terminal was incapable of storing all of the information in the TerminalCapabilitySet. The terminal was incapable of storing more entries than that indicated in highestEntryNumberProcessed or else could not store any.

7.22.5 Terminal Capability Set Release

This is sent in the case of a time out.

7.3 Logical channel signalling messages

This set of messages is for logical channel signalling. The same set of messages is used for uni-directional and bi-directional logical channel signalling; however, some parameters are only present in the case of bi-directional logical channel signalling.

'Forward' is used to refer to transmission in the direction from the terminal making the original request for a logical channel to the other terminal, and 'reverse' is used to refer to the opposite direction of transmission, in the case of a bi-directional channel request.

7.3.1 Open Logical Channel

This is used to attempt to open a logical channel connection between an out-going LCSE and a peer in-coming LCSE.

logicalChannelNumber indicates the logical channel number of the forwardlogical channel that is to be opened.

forwardLogicalChannelParameters: include parameters associated with the logical channel in the case of attempting to open a uni-directional channel and parameters associated with the forward logical channel in the case of attempting to open a bi-directional channel. The forward logical channel is the logical channel to be used for transmission in the same direction as this message is being transmitted.

reverseLogicalChannelParameters: include parameters associated with the reverse logical channel in the case of attempting to open a bi-directional channel. The reverse logical channel is the logical channel to be used for transmission in the opposite direction to that in which this message is being transmitted. It is not present when attempting to open a uni-directional channel. Its presence indicates that the request is for a bi-directional logical channel with the stated parameters, and its absence indicates that the request is for a uni-directional logical channel or that the request is to open the reverse channel of a bi-directional channel.

Note: H.222 parameters are not included in reverseLogicalChannelParameters as their values are not known to the terminal initiating the request.

PortNumber is a user to user parameter that may be used by a user for such purposes as associating an input or output port, or higher layer channel number, with the logical channel.

dataType indicates the data that is to be carried on the logical channel.

logicalChannelParameters indicate system specific parameters associated with the logical channel.

associatedLogicalChannelNumber is used when attempting to open a bi-directional channel. It shall only be present in the OpenLogicalChannel that is sent from the slave terminal responding to a request to open a bi-directional channelto the master terminal; and shall be equal to the logicalChannelNumber used in the corresponding OpenLogicalChannel that is sent from the initiatingmaster terminal to the respondingslave terminal. It is not present when attempting to open a uni-directional channel.

DataType: is used to indicate the data that is to be carried on the logical channel.

If it is nullData, the logical channel will not be used for the transport of elementary stream data, but only for adaptation layer information - if video is to be transmitted in one direction only, but a retransmission protocol is to be used, such as AL3 defined in H.223, a return channel is needed to transport the retransmission requests - it may also be used to describe a logical channel that only contains PCR values in the case of H.222.1 Transport Streams [7].

If it is of type VideoCapability, AudioCapability or DataApplicationCapability, the logical channel may be used for any of the variations indicated by each individual capabilitydescribed; and it shall be possible to switch between these variations using only signalling that is in-band to the logical channel - for example, in the case of H.261 video, if both QCIF and CIF are indicated, it shall be possible to switch between these on a picture by picture basis. In the case of DataApplicationCapability, only one instance of a capability can be indicated since there is no in-band signalling allowing a switch between variations

If it is encryptionData, the logical channel will be used for the transport of encryption information as specified.

LogicalChannelParameters: is used to indicate system specific parameters associated with the logical channel.

H222LogicalChannelParameters: is used to indicate parameters specific to using H.222.1 [7]. It shall be present in forwardLogicalChannelParameters and shall not be present in reverseLogicalChannelParameters.

virtualChannelID indicates in which ATM Virtual Channel the logical channel is to be transported. The means by which this parameter is associated with an ATM Virtual Channel is not specified in this Recommendation.

subChannelID indicates which H.222.1 sub-channel is used for the logical channel. It shall be equal to the PID in a Transport Stream and the stream_id in a Program Stream.

pcr-pid indicates the PID used for the transport of Program Clock References when the Transport Stream is used. It shall be present when the ATM virtual channel carries a Transport Stream and shall not be present when the ATM virtual channel carries a Program Stream.

programDescriptors is an optional octet string, which, if present, contains one or more descriptors, as specified in H.222.0 and H.222.1, that describe the program that the information to be carried in the logical channel is a part of.

streamDescriptors is an optional octet string, which, if present, contains one or more descriptors, as specified in H.222.0 and H.222.1, that describe the information that is to be carried in the logical channel.

H223LogicalChannelParameters: is used to indicate parameters specific to using H.223 [88]. It shall be present in forwardLogicalChannelParameters and reverseLogicalChannelParameters.

adaptationLayerType indicates which adaptation layer and options will be used on the logical channel. The codepoints are as follows: nonStandard, al1Framed (AL1 framed mode), al1NotFramed (AL1 unframed mode), al2WithoutSequenceNumbers (AL2 with no sequence numbers present), al2WithSequenceNumbers (AL2 with sequence numbers present), and al3 (AL3, indicating the number of control field octets that will be present and the size of the send buffer, B_S, that will be used, the size being measured in octets).

segmentableFlag, when equal to true indicates that the channel is designated to be segmentable, and when equal to false indicates that the channel is designated to be non-segmentable.

77.33.22 Open Logical Channel Acknowledge

This is used to confirm acceptance of the logical channel connection request from the peer LCSE. In the case of a request for a uni-directional logical channel, it indicates acceptance of that uni-directional logical channel. In the case of a request for a bi-directional logical channel, it indicates acceptance of that bi-directional logical channel.

logicalChannelNumber indicates the logical channel number of the logical channel that is being opened.

77.33.33 Open Logical Channel Reject

This is used to reject the logical channel connection request from the peer LCSE.

Note: In the case of a bi-directional channel request, rejection applies to both forward and reverse channels. It is not possible to accept one and reject the other.

logicalChannelNumber indicates the logical channel number of the logical channel that is being rejected.

The reasons for sending this message are given in Table 66.

TABLE 66/H.245

Reasons for rejecting a OpenLogicalChannel

ASN.1 codepoint	Cause
unspecified	No cause for rejection specified.
dataTypeNotSupported	The terminal was not capable of supporting the dataType indicated in OpenLogicalChannel.
dataTypeNotAvailable	The terminal was not capable of supporting the dataType indicated in OpenLogicalChannel simultaneously with the dataTypes of logical channels that are already open..
unknownDataType	The terminal did not understand the dataType indicated in OpenLogicalChannel.
dataTypeALCombinationNotSupported	The terminal was not capable of supporting the dataType indicated in OpenLogicalChannel simultaneously with the Adaptation Layer type indicated in H233LogicalChannelParameters

77.33.44 Close Logical Channel

This is used to by the out-going LCSE to close a logical channel connection between two peer LCSEs.

Note. In the case of a bi-directional logical channel, this closes both forward and reverse channels. It is not possible to accept one and reject the other.

logicalChannelNumber indicates the logical channel number of the logical channel that is to be closed.

The source of the logical channel release is given in Table 77.

TABLE 77/H.245

Sources of logical channel release

ASN.1 codepoint	Cause
user	The LCSE-user is the source of the release.
lcse	The LCSE is the source of the release. This may occur as a result of a protocol error.

77.33.55 Close Logical Channel Acknowledge

This is used to confirm the closing of a logical channel connection.

logicalChannelNumber indicates the logical channel number of the logical channel that is being closed.

77.33.66 Open Bi-directional Channel Request

This is sent by a slave terminal to request a master terminal to open one or more bi-directional channels. It is a set of OpenLogicalChannel, the parameters of which are arbitrary except that both forwardLogicalChannelParameters and reverseLogicalChannelParameters shall be present, and dataType and logicalChannelParameters shall indicate the type of bi-directional channel that is being requested.

sequenceNumber is used to label instances of OpenBiDirectionalChannelRequest so that the corresponding response can be identified.

77.33.77 Open Bi-directional Channel Acknowledge

This is sent by a master terminal to a slave terminal to confirm that it intends to open the bi-directional channels requested by the slave terminal.

The sequenceNumber shall be the same as the sequenceNumber in the OpenBiDirectionalChannelRequest for which this is the confirmation.

77.33.88 Open Bi-directional Channel Reject

This is sent by a master terminal to a slave terminal to reject the request by the slave terminal to open the bi-directional channels requested.

The sequenceNumber shall be the same as the sequenceNumber in the OpenBiDirectionalChannelRequest for which this is the rejection.

The possible responses are given in Table 88.

TABLE 88/H.245

Responses to bi-directional channel request

ASN.1 codepoint	Cause
willOpenSomeChannels	The master terminal will attempt to open some of the bi-directional channels that the slave terminal requested. No indication of which or how many of these is given.
willOpenNoChannels	The master terminal will not attempt to open any of the bi-directional channels that the slave terminal requested.
busy	The master terminal is currently in the process of opening bi-directional channels. It is therefore unable to satisfy the request of the slave terminal.

77.33.99 Open Bi-directional Channel Release

This is sent by a master terminal to a slave terminal in the case of a time out.

77.33.1010 Request Channel Close

This is used to by the out-going CLCSE to request the closing of a logical channel connection between two peer LCSEs.

logicalChannelNumber indicates the logical channel number of the logical channel that is requested to close.

77.33.1111 Request Channel Close Acknowledge

This is used by the in-coming CLCSE to indicate that the logical channel connection will be closed.

logicalChannelNumber indicates the logical channel number of the logical channel that it has been requested to close.

77.33.1212 Request Channel Close Reject

This is used by the in-coming CLCSE to indicate that the logical channel connection will not be closed.

logicalChannelNumber indicates the logical channel number of the logical channel that it has been requested to close.

77.33.1313 Request Channel Close Release

This is sent by the out-going CLCSE in the case of a time out.

logicalChannelNumber indicates the logical channel number of the logical channel that it has requested to close.

77.44 Multiplex Table signalling messages

This set of messages is for the secure transmission of H.223 multiplex table entries from the transmitter to the receiver.

77.44.11 Multiplex Entry Send

This is used to send H.223 multiplex table entries from the transmitter to the receiver. It is sent from an out-going MTSE and a peer in-coming MTSE.

sequenceNumber is used to label instances of MultiplexEntrySend so that the corresponding response can be identified.

MultiplexEntryDescriptors is a set of 1 to 15 MultiplexEntryDescriptors.

MultiplexEntryDescriptor: describes a single multiplex table entry. It includes the MultiplexTableEntryNumber and a list of MultiplexElements. A missingzero length (empty) element list indicates that the entry is deactivated.

MultiplexElement: is a recursive structure that describes a single element and a repeat count. If of type logicalChannelNumber, the element indicates a single segment from the given logical channel, and the repeat count indicates the length of the segment in octets. If of type subElementList, the element indicates a sequence of nested MultiplexElements, and the repeat count indicates the number of times to repeat the sequence. In either case, if the repeatCount field is untilClosingFlag, this means to repeat the element indefinitely until the closing flag of the MUX-PDU.

In each MultiplexEntryDescriptor, the repeatCount of the final MultiplexElement in the elementList shall be set to "untilClosingFlag", and the repeatCount of all other MultiplexElements in the elementList shall be set to "finite". This ensures that all multiplex table entries define a multiplex sequence pattern of indefinite length, repeating until the closing flag of the MUX-PDU.

A MultiplexEntryDescriptor with a missingEmpty (size 0) elementList fields shall indicate a deactivated entryies.

Each MultiplexEntrySend requestindication may contain up to 15 MultiplexEntryDescriptors, each describing a single multiplex table entry. Multiplex entries may be sent in any order.

77.44.22 Multiplex Entry Send Acknowledge

This is used to confirm receipt of one or more multiplexEntryDescriptors from a MultiplexEntrySend from the peer MTSE.

The sequenceNumber shall be the same as the sequenceNumber in the MultiplexEntrySend for which this is the confirmation.

multiplexTableEntryNumber indicates which multiplex table entries are being confirmed.

77.44.33 Multiplex Entry Send Reject

This is used to reject one or more multiplexEntryDescriptors from a MultiplexEntrySend from the peer MTSE.

The sequenceNumber shall be the same as the sequenceNumber in the MultiplexEntrySend for which this is the rejection.

MultiplexEntryRejectionDescriptions specifies which table entries are being rejected, and why. The causes of rejection are given in Table 99.

TABLE 99/H.245

Reasons for rejecting a MultiplexEntrySend

ASN.1 codepoint	Cause
unspecified	No cause for rejection specified.
descriptorTooComplex	The MultiplexEntryDescriptor exceeded the capability of the receive terminal.

77.44.44 Multiplex Entry Send Release

This is sent by the out-going MTSE in the case of a time out.

multiplexTableEntryNumber indicates which multiplex table entries have timed out.

77.44.55 Request Multiplex Entry

This is used to request the retransmission of one or more MultiplexEntryDescriptors.

sequenceNumber is used to label instances of RequestMultiplexEntry so that the corresponding response can be identified.

entryNumbers is a list of the MultiplexTableEntryNumbers of the MultiplexEntryDescriptors for which retransmission is requested.

77.44.66 Request Multiplex Entry Response

This is used to indicate whether the terminal will send the requested MultiplexEntryDescriptors.

The sequenceNumber shall be the same as the sequenceNumber in the RequestMultiplexEntry for which this is the response.

The possible responses are given in Table 1040.

TABLE 1040/H.245

Responses to Request Multiplex Entry

ASN.1 codepoint	Response
willSendAllEntries	The out-going LCSE will send all the requested MultiplexEntryDescriptors.
willSendSomeEntries	The out-going LCSE will send some of the requested MultiplexEntryDescriptors. No indication of which or how many of these is given.
willSendNoEntries	The out-going LCSE will send none of the requested MultiplexEntryDescriptors.

77.55 Request Mode messages

This set of messages is used by a receive terminal to request particular modes of transmission from the transmit terminal.

77.55.14 Request Mode

This is used to request particular modes of transmission from the transmit terminal. It is a list, in order or preference (most preferable first), of modes that the terminal would like to receive. Each mode is described using a ModeDescription.

sequenceNumber is used to label instances of RequestMode so that the corresponding response can be identified.

ModeDescription: is a set of one or more ModeElements.

ModeElement: is used to describe a mode element, that is, one of the constituent parts of a complete mode description. It indicates the type of elementary stream that is requested and optionally how it is requested to be multiplexed.

type is used to indicate the type of elementary stream that is requested. It is a choice of VideoMode, AudioMode, DataMode, and EncryptionMode.

h223ModeParameters is used to indicate parameters specific to using H.223 [88].

adaptationLayerType indicates which adaptation layer and options are requested for the requested type. The codepoints are as follows: nonStandard, al1Framed (AL1 framed mode), al1NotFramed (AL1 unframed mode), al2WithoutSequenceNumbers (AL2 with no sequence numbers present), al2WithSequenceNumbers (AL2 with sequence numbers present), and al3 (AL3, indicating the number of control field octets that will be present and the size of the send buffer, B_S, that will be used, the size being measured in octets).

segmentableFlag, when equal to true indicates that segmentable multiplexing is requested, and when equal to false indicates that non-segmentable multiplexing is requested.

77.55.14.14 Video Mode

This is a choice of VideoModes.

H261VideoMode: Indicates the requested picture resolution (either QCIF or CIF).

H262VideoMode: Indicates the requested profile and level, and the optional fields, if present, indicate the requested values of the parameters given. The optional fields are integers with units defined in Table 22.

H263VideoMode: Indicates the requested picture resolution (SQCIF, QCIF, CIF, 4CIF and 16CIF).

The booleans unrestrictedVector, arithmeticCoding, advancedPrediction, and pbFrames, when true, indicate that is it requested to use these optional modes that are defined in the annexes of H.263.

77.55.14.22 Audio Mode

This is a choice of AudioModes.

The exact meaning of the G-series audio codepoints is given in Table 33. There are four options for G.723 audio, to allow either of the bit rates (the low bit rate of 5.3 kbit/s or the high bit rate of 6.3 kbit/s) to be requested with or without the use of silence suppression.

IS11172AudioMode: is used to request audio coded according to ISO/IEC 11172-3 [3333].

audioLayer indicates which coding layer is requested: either audioLayer1, audioLayer2 or audioLayer3.

audioSampling indicates which sample rate is requested: audioSampling32k, audioSampling44k1 and audioSampling48k indicate the audio sample rates 32KHz, 44.1KHz and 48KHz respectively.

multichannelType indicates which multi-channel mode is requested: singleChannel and twoChannels request single channel and stereo/dual channel operation respectively.

bitRate indicates the requested audio bit rate, and is measured in units of kbits per second.

IS13818AudioMode: is used to request audio coded according to ISO/IEC 13818-3 [3434].

audioLayer indicates which coding layer is requested: either audioLayer1, audioLayer2 or audioLayer3.

audioSampling indicates which sample rate is requested: audioSampling16k, audioSampling22k05, audioSampling24k, audioSampling32k, audioSampling44k1 and audioSampling48k indicate the audio sample rates 16KHz, 22.05KHz, 24KHz, 32KHz, 44.1KHz and 48KHz respectively.

multichannelType indicates which multi-channel mode is requested as specified in Table 44.

The boolean lowFrequencyEnhancement, when true, requests a low frequency enhancement channel.

The boolean multilingual, when true, requests up to seven multilingual channels.

bitRate indicates the requested audio bit rate, and is measured in units of kbits per second.

77.55.14.33 Data Mode

This is a choice of data applications.

t120 requests the use of the T.120 [2525] protocol.

dsm-cc requests the use of the DSM-CC [3535] protocol.

userData requests the use of unspecified user data from external data ports.

t84 requests the use of T.84 [2424] for the transfer of such images (JPEG, JBIG, Facsimile Gr.3/4).

t434 requests the use of T.434 [2626] for the transfer of telematic binary files.

h224 requests the use of the real-time simplex device control protocol H.224 [99].

nlpid requests the use of the use of the specified network link layer data application.

DataModeProtocol: is a choice of data protocols. This is used to request the data protocol to be used with the requested data application.

v14buffered requests the use of buffered V.14 [2727].

v42lapm requests the use of the LAPM protocol defined in V.42 [2929].

hdlcFrameTunneling requests the use of HDLC Frame Tunneling. Refer to clause 4.5.2 of ISO/IEC 3309 [3232].

transparent requests the use of transparent data transfer.

77.55.14.44 Encryption Mode

This is a choice of encryption modes.

h233Encryption requests the use of encryption according to H.233 and H.234 [1111][1212].

77.55.22 Request Mode Acknowledge

This is sent to confirm that the transmit terminal intends to transmit in one of the modes requested by the receive terminal.

The sequenceNumber shall be the same as the sequenceNumber in the RequestMode for which this is the confirmation.

The possible responses are given in Table 1111.

TABLE 1111/H.245

Confirmation responses to Request Mode

ASN.1 codepoint	Response
willTransmitMostPreferredMode	The transmit terminal will change to the receiver's most preferred mode.
willTransmitLessPreferredMode	The transmit terminal will change to one of the receiver's preferred mode, but not the most preferred mode.

7.55.3 Request Mode Reject

This is sent to reject the request by the receive terminal.

The sequenceNumber shall be the same as the sequenceNumber in the RequestMode for which this is the response.

The possible responses are given in Table 12.

TABLE 12/H.245

Rejection responses to Request Mode

ASN.1 codepoint	Response
modeUnavailable	The transmit terminal will not change its mode of transmission as the requested modes are not available.
multipointConstraint	The transmit terminal will not change its mode of transmission due to a multipoint constraint.
requestDenied	The transmit terminal will not change its mode of transmission.

7.55.4 Request Mode Release

This is used by the out-going MRSE in the case of a time out.

7.6 Round Trip Delay messages

This set of messages is used by a terminal to determine the round trip delay between two communicating terminals. It also enables a H.245 user to determine whether the peer H.245 protocol entity is alive.

7.6.1 Round Trip Delay Request

This is sent from the out-going RTDSE to the in-coming RTDSE.

sequenceNumber is used to label instances of RoundTripDelayRequest so that the corresponding response can be identified.

7.6.2 Round Trip Delay Response

This is sent from the in-coming RTDSE to the out-going RTDSE.

The sequenceNumber shall be the same as the sequenceNumber in the RoundTripDelayRequest for which this is the response.

7.7 Maintenance Loop messages

This set of messages is used by a terminal to perform maintenance loop functions.

7.7.1 Maintenance Loop Request

This is sent to request a particular type of loop back. The types mediaLoop and logicalChannelLoop request the loopback of only one logical channel as indicated by LogicalChannelNumber, while the type systemLoop refers to all logical channels. The exact definition of these types is system specific and outside the scope of this Recommendation.

sequenceNumber is used to label instances of MaintenanceRequest so that the corresponding response can be identified.

7.7.2 Maintenance Loop Response

This is used to indicate how the terminal will respond to the request for a maintenance loop.

The sequenceNumber shall be the same as the sequenceNumber in the MaintenanceRequest for which this is the response.

The possible responses are given in Table 13.

TABLE 13/H.245

Responses to Maintenance Loop Request

ASN.1 codepoint	Response
willPerformLoop	The terminal will perform the loop as requested.
willNotPerformLoop	The terminal will not perform the loop as requested.

7.7.3 Maintenance Loop Command Off

On receipt of this command, the terminal shall disconnect all loops and restore audio, video and data paths to their normal condition.

77.88 Commands

A command message requires action but no explicit response.

77.88.11 Send Terminal Capability Set

specificRequestThis commands the far end terminal to indicate its transmit and receive capabilities by sending one or more TerminalCapabilitySets that contain the information requested, as specified below. This command may be sent at any time to elicit the capabilities of the remote terminal, for example, following an interruption or other cause for uncertainty; however, such messages should not be sent repetitively without strong cause.

A terminal shall only request the transmission of capability TableEntryNumbers and capability DescriptorNumbers that it has previously received. A terminal shall ignore any requests to transmit capability TableEntryNumbers and capabilityDescriptorNumbers that it has not previously transmitted and no fault shall be considered to have occurred.

The boolean multiplexCapability, when true, requests the transmission of the MultiplexCapability.

capabilityTableEntryNumbers is a set of the CapabilityTableEntryNumbers that indicate the CapabilityTableEntrys that the terminal requests to be transmitted.

capabilityDescriptorNumbers is a set of the CapabilityDescriptorNumbers that indicate the CapabilityDescriptors that the terminal requests to be transmitted.

genericRequest commands the far end terminal to send its entire terminal capability set.

77.88.22 Encryption

This command is used to exchange encryption capabilities and to command the transmission of an initialisation vector (IV), refer to H.233 and H.234 [1144][1242].

encryptionSE is an H.233 Session Exchange (SE) message, except that the error protection bits described in H.233 shall not be applied.

encryptionIVRequest commands the far-end encryptor to transmit a new IV in a logical channel opened for encryptionData.

encryptionAlgorithmIdentifier indicates to the receiver that the sending terminal will associate the given H.233 Algorithm Identifier value with the non-standard encryption algorithm.

77.88.33 Flow Control

This command is used to specify the upper limit of bit rate of either a single logical channel or the whole multiplex. A terminal may send this command to restrict the bit rate that the far-end terminal sends. A terminal that receives this command shall comply with it.

When scope is of type logicalChannelNumber the limit applies to the given logical channel, when scope is of type virtualChannelID the limit applies to the given ATM virtual channel, and when scope is of type wholeMultiplex the limit applies to the whole multiplex.

maximumBitRate is measured in units of 100 bit/s averaged over non-overlapping consecutive periods of one second. When this is present, the specified limit supersedes any previous limit, whether higher or lower. When it is not present any previous restriction on the bit rate for the channel is no longer applicable.

The point at which the bit rate limit is applied, and the specification of which bits are included in the calculation of bit rate is not specified in this Recommendation, but should be specified by recommendations that use this Recommendation.

Each transmission of this command affects a specific logical channel or the entire multiplex. More than one such command may be in effect at the same time, up to the number of open logical channels plus one, for the overall multiplex limitation.

Note. When the bit rate that can be transmitted on a logical channel is constrained to particular values, for example G.723 audio, and the request is to transmit at a rate lower than the lowest rate at which it would normally operate, it shall respond by stopping transmission on the logical channel.

77.88.44 End session

This command indicates the end of the H.245 session. After transmitting EndSessionCommand, the terminal shall not send any more of the messages defined in this Recommendation.

disconnect indicates that the connection will be dropped.

v34Options: is a choice of alternatives that will occur after ending the H.245 session when a V.34 [2828] modem is used.

The possible options are given in Table 1414.

TABLE 1414/H.245

Options after EndSessionCommand when using an V.34 Modem

ASN.1 codepoint	Option
telephonyMode	The terminal shall initiate the cleardown procedures defined in V.34, except that it shall not physically disconnect the GSTN connection.
v34DSVD	The terminal shall preserve the V.34 modem connection, but use it to support V.DSVD.
v34DuplexFAX	The terminal shall preserve the V.34 modem connection, but use it to support T.30 FAX [2124].
v34H324	The terminal shall preserve the V.34 modem connection, but use it to support H.324 [1818].

77.88.55 Miscellaneous Command

This is used for a variety of commands, some of which are present in H.221 and H.230 [55][3740].

logicalChannelNumber indicates the logical channel number to which the command applies. It shall indicate a logical channel opened for video data when the type is one of videoFreezePicture, videoFastUpdatePicture, videoFastUpdateGOB, videoTemporalSpatialTradeOff, videoSendSyncEveryGOB, and videoSendSyncEveryGOBCancel.

equaliseDelay and zeroDelay shall have the same meaning as the commands ACE and ACZ defined in H.230[3740].

videoFreezePicture commands the video decoder to complete updating the current video frame and subsequently display the frozen picture until receipt of the appropriate freeze-picture release control signal.

videoFastUpdatePicture commands the video encoder to enter the fast-update mode at its earliest opportunity.

videoFastUpdateGOB commands the far-end video encoder to perform a fast update of one or more GOBs. firstGOB indicates the number of the first GOB to be updated, and numberOfGOBs indicates the number of GOBs to be updated. It shall only be used with video compression algorithms that define GOBs, for example, H.261 and H.263.

videoTemporalSpatialTradeOff commands the far-end video encoder to change its trade-off between temporal and spatial resolution. A value of 0 commands a high spatial resolution and a value of 31 commands a high frame rate. The values from 0 to 31 indicate monotonically a desire for higher frame rate. Actual values do not correspond to precise values of spatial resolution or frame rate.

videoSendSyncEveryGOB commands the far-end video encoder to use sync for every GOB as defined in H.263 [1515], until the command videoSendSyncEveryGOBCancel is received, from which time the far-end video encoder may decide the frequency of GOB syncs. These commands shall only be used with video encoded according to H.263.

7.9 Indications

An indication contains information that does not require action or response.

7.9.1 Function Not Supported

This is used to return requests, responses and commands that are not understood to the transmitter of them.

The whole of the RequestPDU, ResponsePDU or CommandPDU is returned.

If a terminal receives a request, response or command that it does not understand, either because it is non-standard or has been defined in a subsequent revision of this Recommendation, it shall respond by sending FunctionNotSupported.

7.9.2 Miscellaneous Indication

This is used for a variety of indications, some of which are present in H.221 and H.230 [5][37].

logicalChannelNumber indicates the logical channel number to which the indication applies. It shall indicate a logical channel opened for video data when the type is videoIndicateReadyToActivate, and videoTemporalSpatialTradeOff.

logicalChannelInactive is used to indicate that the content of the logical channel does not represent a normal signal. It is analogous to AIM and VIS defined in H.230.

logicalChannelActive is complementary to logicalChannelInactive. It is analogous to AIA and VIA defined in H.230.

multipointConference, cancelMultipointConference, multipointZeroComm, cancelMultipointZeroComm, multipointSecondaryStatus, and cancelMultipointSecondaryStatus shall have the same meaning as MIC, cancelMIC, MIZ, cancelMIZ, MIS and cancelMIS respectively, as defined in H.230.

videoIndicateReadyToActivate shall have the same meaning as VIR defined in H.230, that is, it is transmitted by a terminal whose user has decided not to send video unless he will also receive video from the other end.

videoTemporalSpatialTradeOff indicates to the far-end video decoder its current trade-off between temporal and spatial resolution. A value of 0 indicates a high spatial resolution and a value of 31 indicates a high frame rate. The values from 0 to 31 indicate monotonically a higher frame rate. Actual values do not correspond to precise values of spatial resolution or frame rate. A terminal that has indicated temporalSpatialTradeOffCapability shall transmit this indication whenever it changes its trade-off and when a video logical channel is initially opened.

7.9.3 Jitter Indication

This is used to indicate the amount of jitter, as estimated by the receive terminal, of a logical channel. It may be useful for choice of bit-rate and buffer control in video channels, or to determine an appropriate rate of transmission of timing information, etc.. The video encoder will then have the option of using this information to restrict the video bit-rate or the video decoder buffer fluctuations to help prevent decoder buffer underflow or overflow, given the occurring jitter. If the encoder takes this option, it will enable correct operation for existing designs of video decoder buffers, regardless of the amplitude of received jitter, as well as allow correct operation with minimum delay.

When scope is of type logicalChannelNumber the information applies to the given logical channel, when scope is of type virtualChannelID the information applies to the given ATM virtual channel, and when scope is of type wholeMultiplex the information applies to the whole multiplex.

estimatedReceivedJitterMantissa and estimatedReceivedJitterExponent provide an estimate of the jitter that has been received by the terminal that has sent the message.

estimatedReceivedJitterMantissa indicates the mantissa of the jitter estimate as given in Table 15.

TABLE 15/H.245

Mantissa of estimatedReceivedJitterMantissa in JitterIndication

estimatedReceivedJitterMantissa	Mantissa
0	1
1	2.5
2	5
3	7.5

estimatedReceivedJitterExponent indicates the exponent of the jitter estimate as given in Table 1616.

TABLE 1616/H.245

Exponent of estimatedReceivedJitterExponent in JitterIndication

estimatedReceivedJitterExponent	Exponent
0	Out of Range
1	1 µs
2	10 µs
3	100 µs
4	1 ms
5	10 ms
6	100 ms
7	1 s

The jitter estimate is obtained by multiplying the mantissa by the exponent, unless estimatedReceivedJitterExponent is equal to zero, in which case the estimate is just known to be more than 7.5 seconds.

skippedFrameCount indicates how many frames have been skipped by the decoder since the last JitterControlPDU was received. Since the maximum value that can be encoded is 15, if this option is implemented, this information must be transmitted before more than 15 frames have been skipped.

Note. Since frames are skipped when the decoder buffer underflows, additional jitter may cause the decoder buffer to underflow more or less often than the encoder expects frame skips to happen.

additionalDecoderBuffer indicates the additional size of the video decoder buffer over and above that required by the indicated profile and level. This is defined in the same way as vbv_buffer_size H.262 [1414].

77.99.44 H.223 Skew Indication

This is used to indicate to the far-end terminal the average amount of time skew between two logical channels.

logicalChannelNumber1 and logicalChannelNumber2 are logical channel numbers of opened logical channels.

skew is measured in milliseconds, and indicates the delay that must be applied to data belonging to logicalChannelNumber2 as measured at the multiplex, to achieve synchronisation with logicalChannelNumber1 as measured at the multiplex. The actual delay necessary for synchronisation is dependent on decoder implementation, and is a local matter for the receiver.

77.99.55 User Input

This is used for User Input messages.

alphanumeric is a string of characters coded according to T.51 [2323]. This could be used for key-pad input, an equivalent to DTMF.

Note. Any data that is carried in H.245, including user input messages, will not be encrypted.

88 Procedures

88.11 Introduction

This section defines generic multimedia system control procedures that use the messages defined in this Recommendation. Recommendations using this Recommendation shall indicate which of these procedures are applicable, as well as defining any specific requirements.

Procedures to perform the following functions are described in this section:

- master-slave determination
- terminal capability exchange
- logical channel signalling
- bi-directional logical channel signalling
- receive terminal close logical channel request
- slave terminal open bi-directional logical channel request
- H.223 multiplex table entry modification
- receiver to transmitter transmit mode request
- round trip delay determination

88.11.11 Method of specification

Procedures are generally specified in this section using SDLs. The SDL provides a graphical specification of the procedures, and includes specification of actions in the event of exception conditions.

88.11.22 Communication between protocol entity and protocol user

The interaction with the user of a particular function is specified in terms of primitives transferred at the interface between the protocol entity and the protocol user. Primitives are for the purpose of defining protocol procedures and are not intended to specify or constrain implementation. There may be a number of parameters associated with each primitive.

To assist in the specification, protocol states are defined. These states are conceptual and reflect general conditions of the protocol entity in the sequences of primitives exchanged between the protocol entity and the user, and the exchange of messages between the protocol entity and its peer.

For each protocol entity the allowed sequence of primitives between the user and the protocol entity is defined. The allowed sequence constrains the actions of the user, and defines the possible responses from the protocol entity.

A primitive parameter described as being null, is equivalent to the parameter not being present.

88.11.33 Peer-to-peer communication

Protocol information is transferred to the peer protocol entity via the relevant messages defined in section 6. In the SDL diagrams these messages are referred to as PDUs.

Some protocol entities described have state variables associated with them. A number of protocol entities described also have timers associated with them.

A primitive is defined to report protocol error conditions to some unspecified management entity.

In the SDLs the messages defined in section 6 are represented by shorthand SDL names.

88.11.44 SDL Diagrams

The SDL diagrams show actions to the allowed interactions with the protocol user, and to reception of messages from the peer protocol entity. Primitives which are not allowed for a given state are not shown in the SDL diagrams. However the responses to the reception of inappropriate messages are described in the SDL diagrams.

88.11.55 SDL Key

The SDL key is shown in Figure 11.

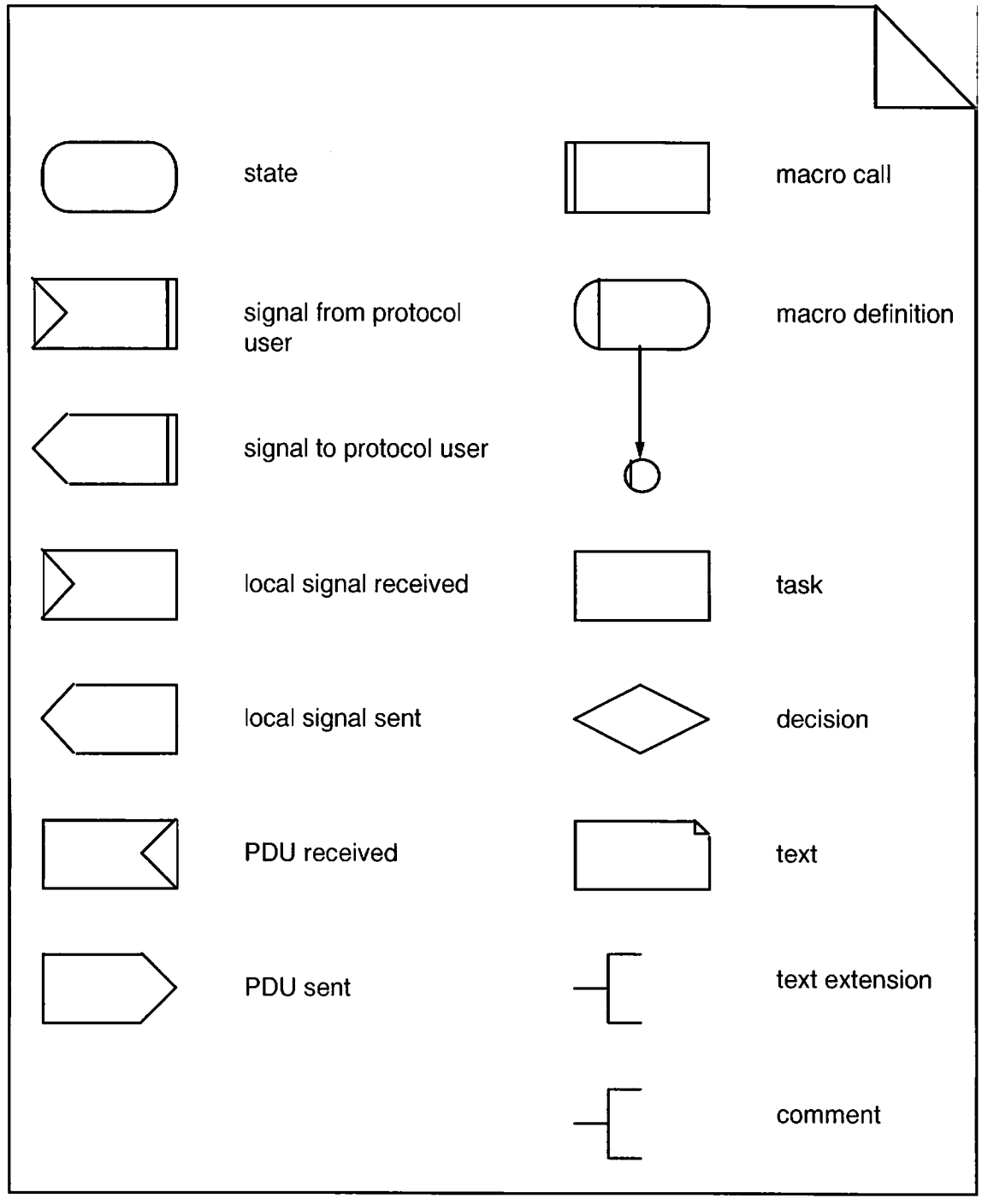

FIGURE 11/H.245

SDL key

88.22 Master slave determination procedures

88.22.1 Introduction

Bi-directional logical channel signalling requires one of the terminals involved in a communications call to act, with respect to the opening and closing of bi-directional logical channels, as a master terminal and the other terminal(s) involved in the call to act as slave terminals. Procedures described here allow terminals in the call to determine which is the master terminal and which is the slave terminal.

The function described here is referred to as the Master Slave Determination Signalling Entity (MSDSE). There is one instance of the MSDSE in each terminal involved in a call.

Either terminal may initiate the master slave determination process by transmitting its statusDeterminationNumber. When a terminal receives such a number, it compares it with its own number, and the terminal with the larger number is determined as the master. The terminal that receives the statusDeterminationNumber responds by sending the decision, that is, the status of the terminal that sent the statusDeterminationNumber.

If both terminals have equal statusDeterminationNumbers, the terminal that receives the other's number responds with the rejection message indicating that the numbers were identical. In this case, either terminal may transmit a new statusDeterminationNumber. This process shall be repeated until the two numbers are different or one or other terminal ends the connection.

A terminal shall change its status determination number only on receipt of a MSDREJ PDU from the remote. If a terminal has received an MSD PDU from the remote before it has initiated its MSD procedure, it should not start its MSD procedure.

88.22.2 Communication between the MSDSE and the MSDSE user

88.22.2.1 Primitives between the MSDSE and the MSDSE user

Communication between the MSDSE, and MSDSE user, is performed using the primitives shown in Table 17.

TABLE 17/H.245

Primitives and parameters

generic name	type			
	request	indication	response	confirm
DETERMINE	- 1USE_PREV	TYPE	not defined [2]	TYPE
REJECT	not defined	SOURCE CAUSE	not defined	not defined

Notes:

1. "-" means no parameters
2. "not defined" means that this primitive is not defined.

88.22.2.2 Primitive definition

The definition of these primitives is as follows:

a) The DETERMINE primitive is used to initiate, and to return the result from, the master slave determination procedure.

b) The REJECT primitive indicates that the master slave determination procedure was unsuccessful.

88.22.2.3 Parameter definition

The definition of the primitive parameters shown in Table 17 are as follows:

a) The USE_PREV parameter indicates if set to TRUE that the same status determination number shall be used when sending an MSD PDU else if set to FALSE a new value shall be used.

ba) The TYPE primitive indicates the terminal type. It has the value of "MASTER" or "SLAVE".

cb) The SOURCE parameter indicates the source of the REJECT.indication primitive. The SOURCE parameter has the value of "USER" or "PROTOCOL". The latter case may occur as the result of a timer expiry.

dc) The CAUSE parameter indicates the reason for failure of the master slave determination procedure. The CAUSE parameter is not present when the SOURCE parameter indicates "PROTOCOL".

88.22.22.44 MSDSE states

The following states are used to specify the allowed sequence of primitives between the MSDSE and the MSDSE user.

State 0: IDLE

The local MSDSE user has not initiated a master slave determination procedure.

State 1: AWAITING RESPONSE

The local MSDSE user has requested a master slave determination procedure. A response from the peer MSDSE is awaited.

88.22.22.55 State transition diagram

The allowed sequence of primitives between the MSDSE and the MSDSE user is defined here. The allowed sequences are shown in Figure 22.

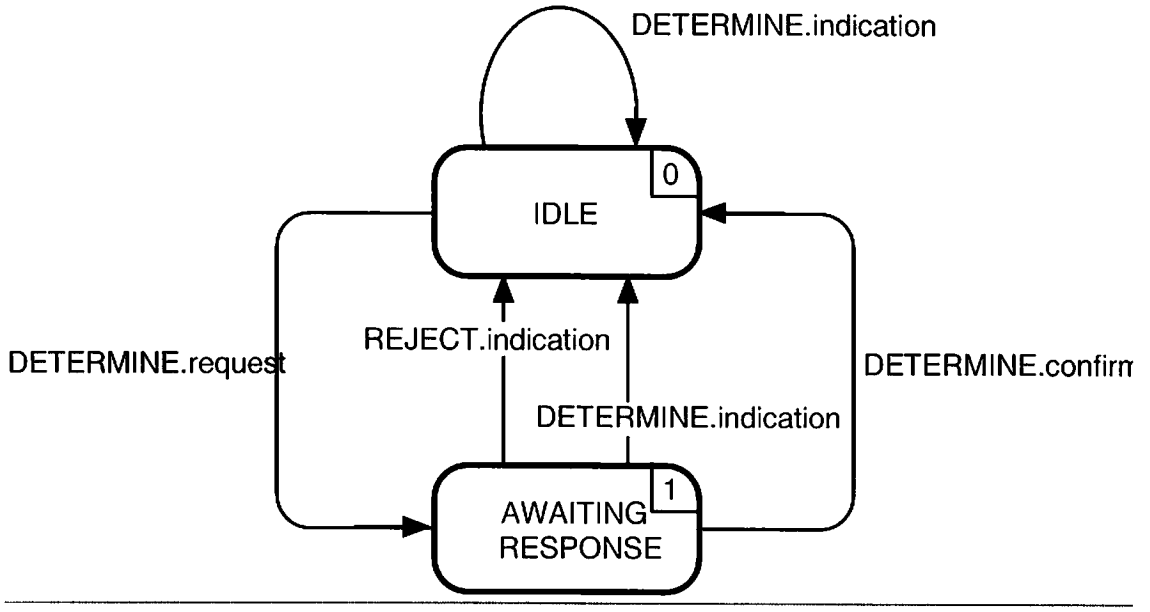

FIGURE 22/H.245

State transition diagram for sequence of primitives at MSDSE

88.22.33 Peer to peer MSDSE communication

88.22.33.11 MSDSE messages

Table 1818 shows the MSDSE messages and fields, defined in section 6, which are relevant to the MSDSE protocol.

TABLE 1818/H.245

MSDSE message names and fields

function	message	direction	SDL name [2]	field
determination	MasterSlaveDetermination	O -> I [1]	MSD	statusDeterminationNumber
	MasterSlaveDeterminationAck	O <- I	MSDACK	decision
	MasterSlaveDeterminationReject	O <- I	MSDREJ	cause

Notes:

1. Direction: O - out-going, I - in-coming.
2. The SDL name is a short hand notation used to indicate the message in the SDL figures.

88.22.33.22 MSDSE state variables

The following MSDSE state variables are defined:

sv_SDNUM

This state variable holds the status determination number for this terminal.

sv_RT

This state variable indicates the number of times an MSD PDU has been transmitted. It is incremented by one after each transmission of the MSD.

88.22.33.33 MSDSE timers

The following timer is specified for the out-going MSDSE:

T106

This timer is used during the AWAITING RESPONSE state. It specifies the maximum allowed time in which no MSDACK or MSDREJ has been received.

88.22.33.43 MSDSE counters

The following counter is specified for the out-going MSDSE:

N206

This counter is used during the AWAITING RESPONSE state. It specifies the maximum number of times an MSD shall be sent if no MSDACK or MSDREJ has been received.

88.22.44 MSDSE procedures

88.22.44.11 Introduction

Figure 33 summarises the MSDSE primitives and their parameters, and messages.

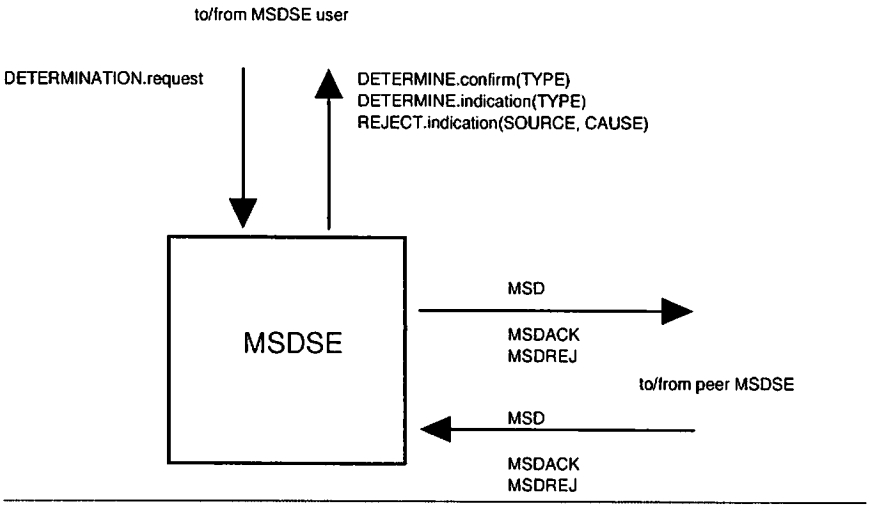

FIGURE 33/H.245

Primitives and messages in the MSDSE.

88.22.44.22 Primitive parameter default values

Where not explicitly stated in the SDL diagrams the parameters of the indication and confirm primitives assume values as shown in Table 1919.

TABLE 1919/H.245

Default primitive parameter values

primitive	parameter	default value
DETERMINE.request	USE_PREV	FALSE
DETERMINE.confirm	TYPE	MSDACK.decision
DETERMINE.indication	TYPE	MASTER
REJECT.indication	SOURCE	USER
	CAUSE	null

88.22.44.33 Message field default values

Where not explicitly stated in the SDL diagrams the message fields assume values as shown in Table 2020.

TABLE 2020/H.245

Default message field values

message	field	default value
MSD	statusDeterminationNumber	sv_SDNUM
MSDACK	decision	slave
MSDREJ	cause	identicalNumbers

88.22.44.44 SDLs

The MSDSE procedures are expressed in SDL form in Figure 44.

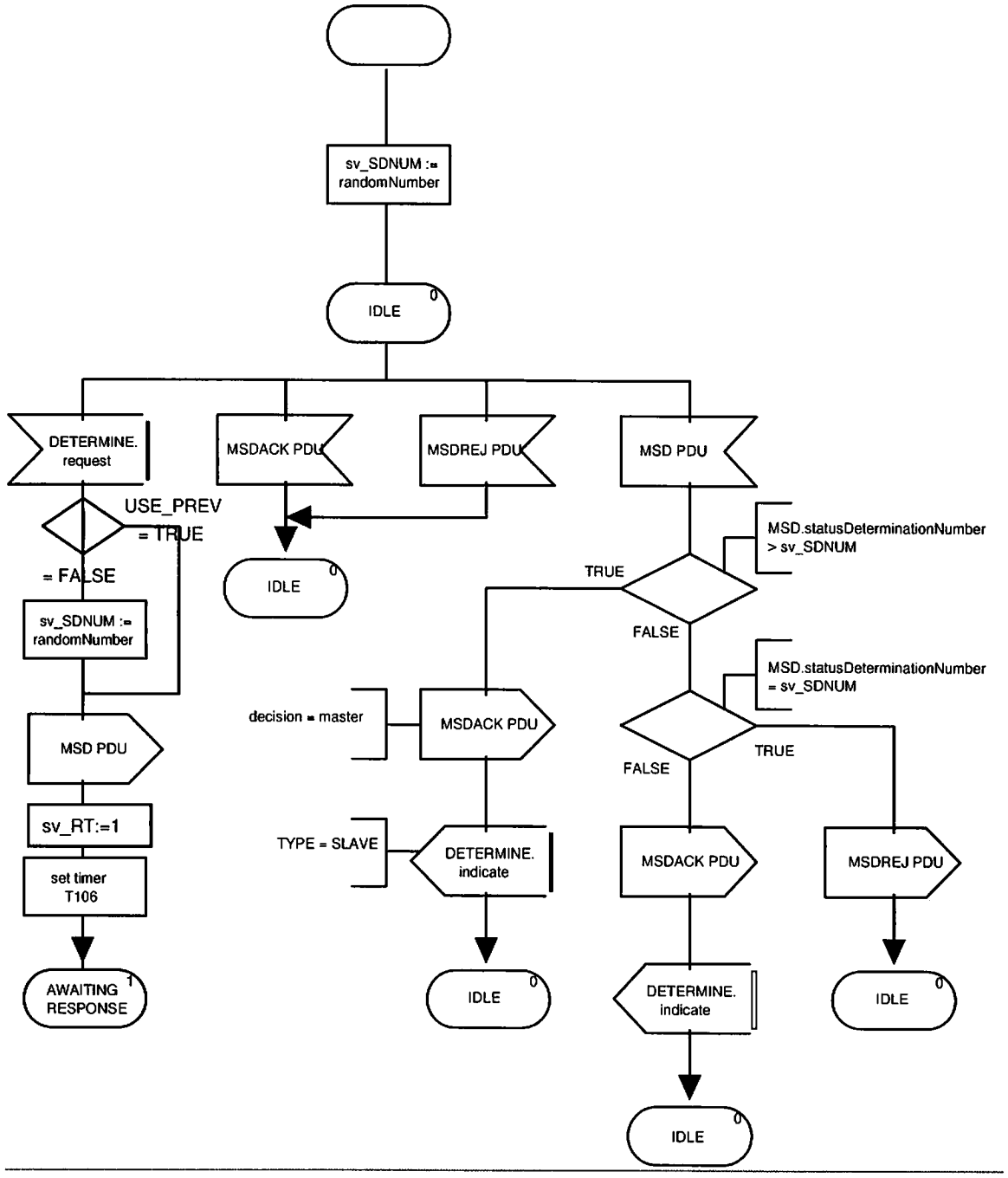

FIGURE 44(i)/H.245

MSDSE SDL

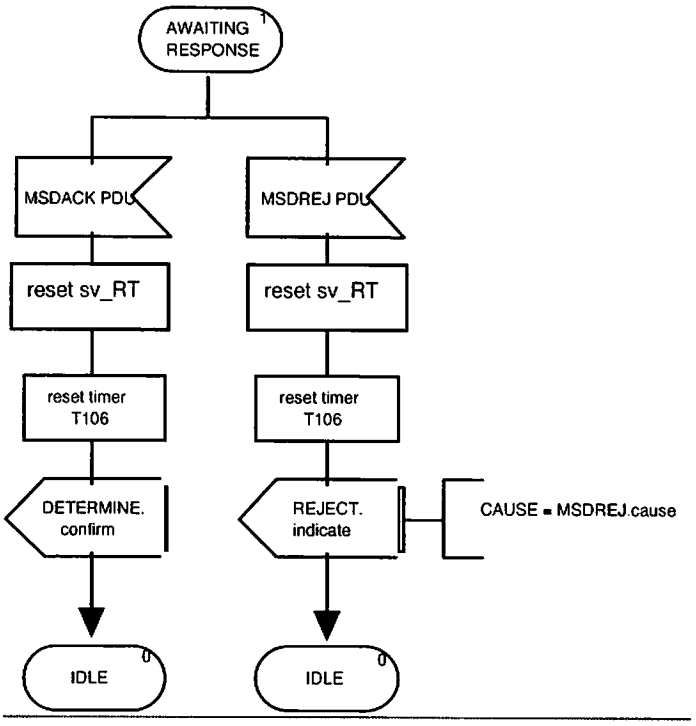

FIGURE 44(ii)/H.245

MSDSE SDL (continued)

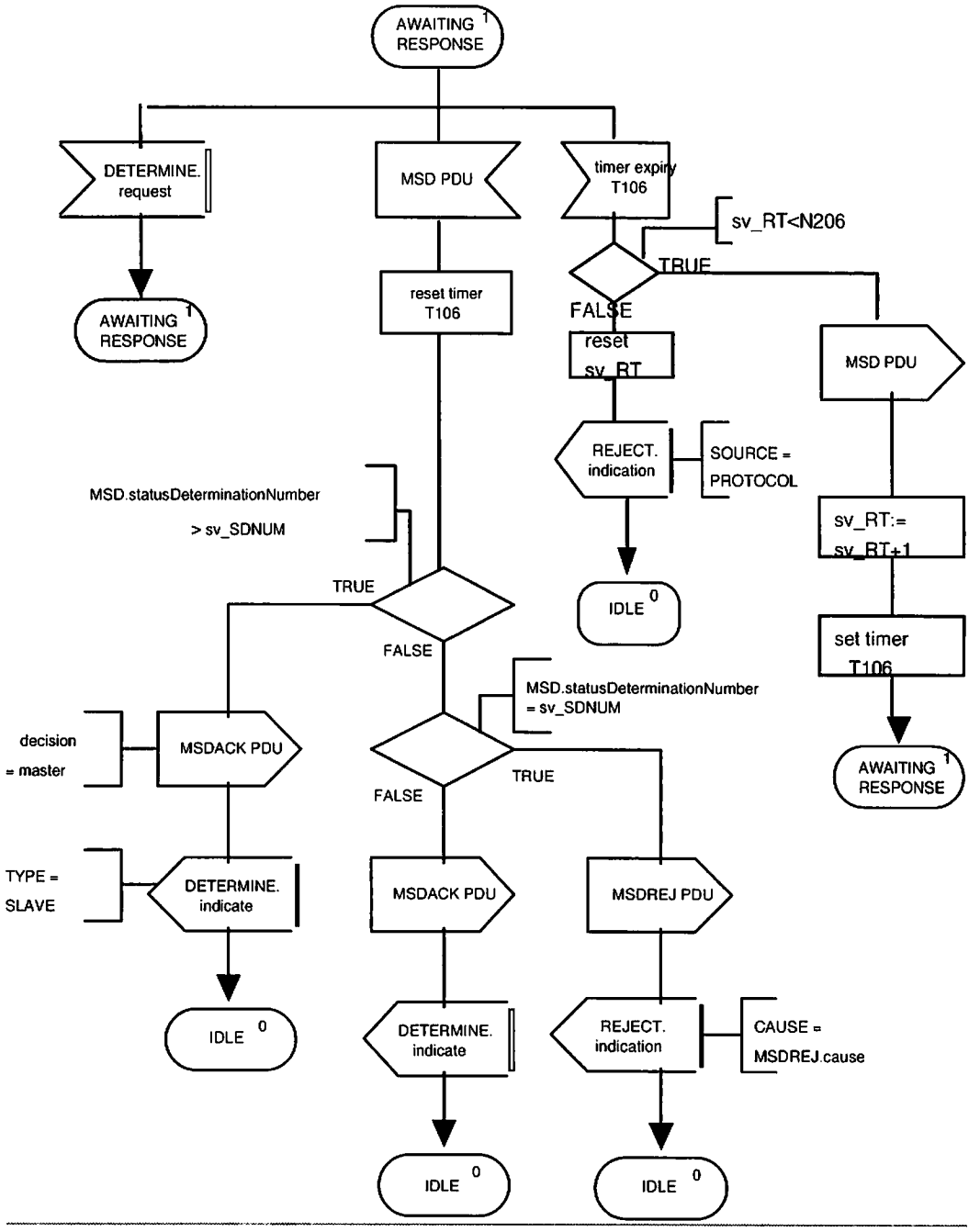

FIGURE 4T(iii)/H.245

MSDSE SDL (concluded)

88.33 Capability exchange procedures

88.33.14 Introduction

These procedures are used by terminals to communicate their capabilities, and are referred to as the Capability Exchange Signalling Entity (CESE). Procedures are specified in terms of primitives and states at the interface between the CESE and the CESE user. Protocol information is transferred to the peer CESE via relevant messages defined in section 6. There is an out-going CESE and an in-coming CESE. At each of the out-going and in-coming ends there is one instance of the CESE for each call.

A terminal shall indicate its capabilities by sending one or more TerminalCapabilitySets. A terminal that receives a TerminalCapabilitySet, shall respond by sending a TerminalCapabilitySetAck or TerminalCapabilitySetReject.

All terminals intended for use in point-to-point applications or those connected to an MCU shall be able to identify a TerminalCapabilitySet and its structure, and such capability values therein that are mandatory for those applications; any unrecognised capability values shall be ignored, and no fault shall be implied.

The capability exchange may be performed at any time. The capability exchange may signal both changed and unchanged capabilities. Unchanged capabilities should not be sent repetitively without strong cause.

88.33.22 Communication between CESE and CESE user

88.33.22.11 Primitives between CESE and CESE user

Communication between the CESE and CESE user, is performed using the primitives shown in Table 2124.

TABLE 2124/H.245

Primitives and parameters

generic name	type			
	request	indication	response	confirm
TRANSFER	MUXCAP CAPTABLE CAPDESCRIPTORS	MUXCAP CAPTABLE CAPDESCRIPTORS	- [1]	-
REJECT	CAUSE	SOURCE CAUSE	not defined [2]	not defined
REJECTERROR	CAUSEnot defined [2]	SOURCE CAUSE	not defined [2]not defined [2]	not definednot defined [2]

Notes:

1. "-" means no parameters
2. "not defined" means that this primitive is not defined.

88.33.22.22 Primitive definition

The definition of these primitives is as follows:

a) The TRANSFER primitives are used for transfer of the capability exchange.

b) The REJECT primitives are used to reject a capability descriptor entry, and to terminate a current capability transfer.

c) The ERROR primitive reports CESE errors to the CESE user.

88.33.22.33 Parameter definition

The definition of the primitive parameters shown in Table 2124 are as follows:

a) The MUXCAP parameter is the multiplex capability parameter. This parameter is carried transparently to the peer CESE user. This parameter is optional.

b) The CAPTABLE parameter is the capability table parameter. There may be onezero or more capability table entries described within this parameter. This parameter is carried transparently to the peer CESE user. This parameter is optional.

c) The CAPDESCRIPTORS parameter is the capability descriptors parameter. There may be onezero or more capability descriptors described within in this parameter. This parameter is carried transparently to the peer CESE user. this parameter is optional.

d) The SOURCE parameter indicates the source of the REJECT.indication primitive. The SOURCE parameter has the value of "USER" or "PROTOCOL". The latter case may occur as the result of a timer expiry.

e) The CAUSE parameter indicates the reason for rejection of a CAPTABLE or CAPDESCRIPTORS parameter. The CAUSE parameter is not present when the SOURCE parameter indicates "PROTOCOL".

88.33.22.44 CESE states

The following states are used to specify the allowed sequence of primitives between the CESE and the CESE user.

The states for an out-going CESE are:

State 0: IDLE

The CESE is idle.

State 1: AWAITING RESPONSE

The CESE is waiting for a response from the remote CESE.

The states for an in-coming CESE are:

State 0: IDLE

The CESE is idle.

State 1: AWAITING RESPONSE

The CESE is waiting for a response from the CESE user.

88.33.22.55 State transition diagram

The allowed sequence of primitives between the CESE and the CESE user is defined here. The allowed sequence of primitives relates to states of the CESE as viewed from the CESE user. The allowed sequences are specified separately for each of an out-going CESE and an in-coming CESE, as shown in Figure 55 and Figure 66 respectively.

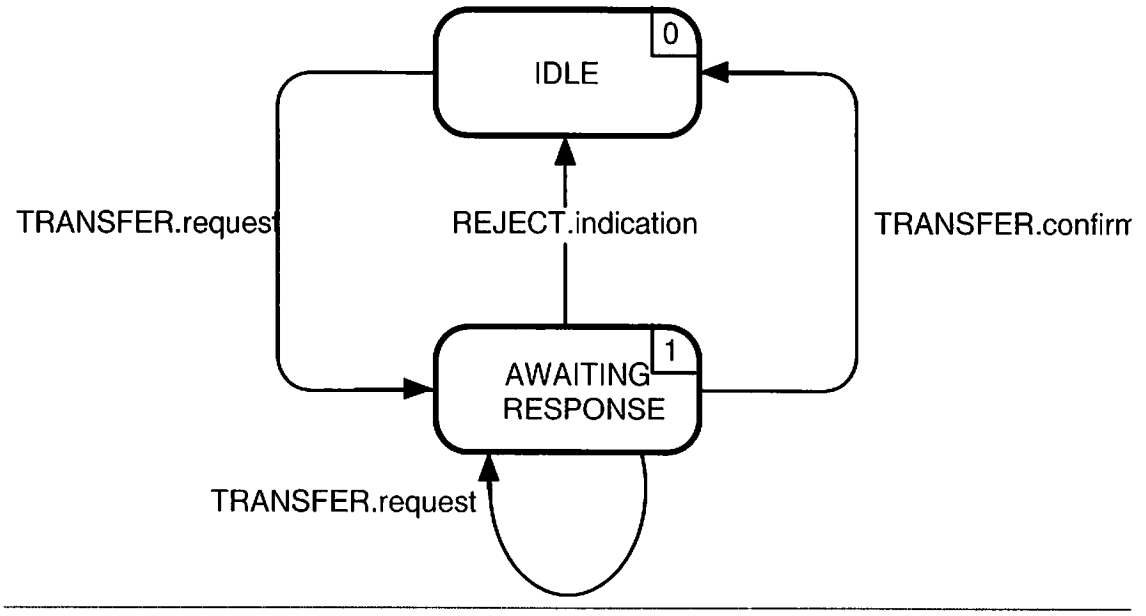

FIGURE 55/H.245

State transition diagram for sequence of primitives at CESE out-going

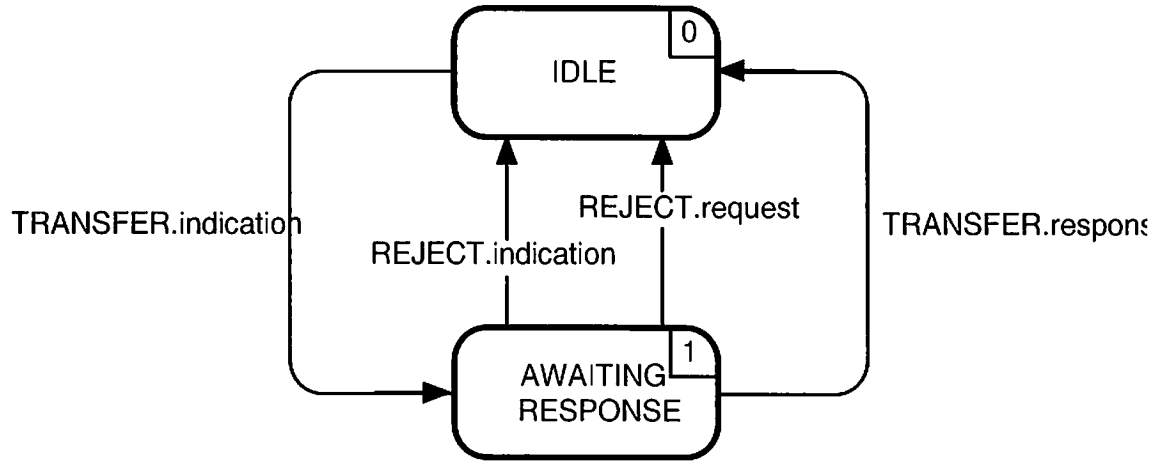

FIGURE 66/H.245

State transition diagram for sequence of primitives at CESE in-coming

88.33.33 Peer to peer CESE communication

88.33.33.11 Messages

Table 2222 shows the CESE messages and fields, defined in section 6, which are relevant to the CESE protocol.

TABLE 2222/H.245

CESE message names and fields

function	message	direction	SDL name [2]	field
transfer	TerminalCapabilitySet	O -> I [1]	SEND	sequenceNumber multiplexCapability capabilityTable capabilityDescriptors
	TerminalCapabilitySetAck	O <- I	ACK	sequenceNumber
reject	TerminalCapabilitySetReject	O <- I	REJ	sequenceNumber cause
reset	TerminalCapabilitySetRelease	O -> I	RELEASE	-

Notes:

1. Direction: O - out-going, I - in-coming.
2. The SDL name is a short hand notation used to indicate the message in the SDL figures.

88.33.33.22 CESE state variables

The following state variables are defined at the out-going CESE:

out_SQ

This state variable is used to indicate the most recent SEND message. It is incremented by one and mapped to the SEND message sequenceNumber field before transmission of the SEND message. Arithmetic performed on out_SQ is modulo 256.

out_RT

This state variable is used to indicate the number of times a SEND message has been transmitted. It is incremented by one after each transmission of the SEND message.

The following state variables are defined at the in-coming CESE:

in_SQ

This state variable is used to store the value of the sequenceNumber field of the most recently received SEND message. The ACK and REJ messages have their sequenceNumber fields set to the value of in_SQ, before being sent to the peer CESE.

88.33.33.33 CESE timers

The following timer is specified for the out-going CESE:

T101

This timer is used during the AWAITING RESPONSE state. It specifies the time before an error message is generated, during which no ACK or REJ message has been received.

88.33.33.44 CESE counters

The following counter is specified for the out-going CESE:

N201

This counter is used during the AWAITING RESPONSE state. It specifies the numbers of times a SEND message can be transmitted if no ACK or REJ message has been received.

88.33.44 CESE procedures

Figure 77 summarises the CESE primitives and their parameters, and messages, for each of the out-going and in-coming CESE.

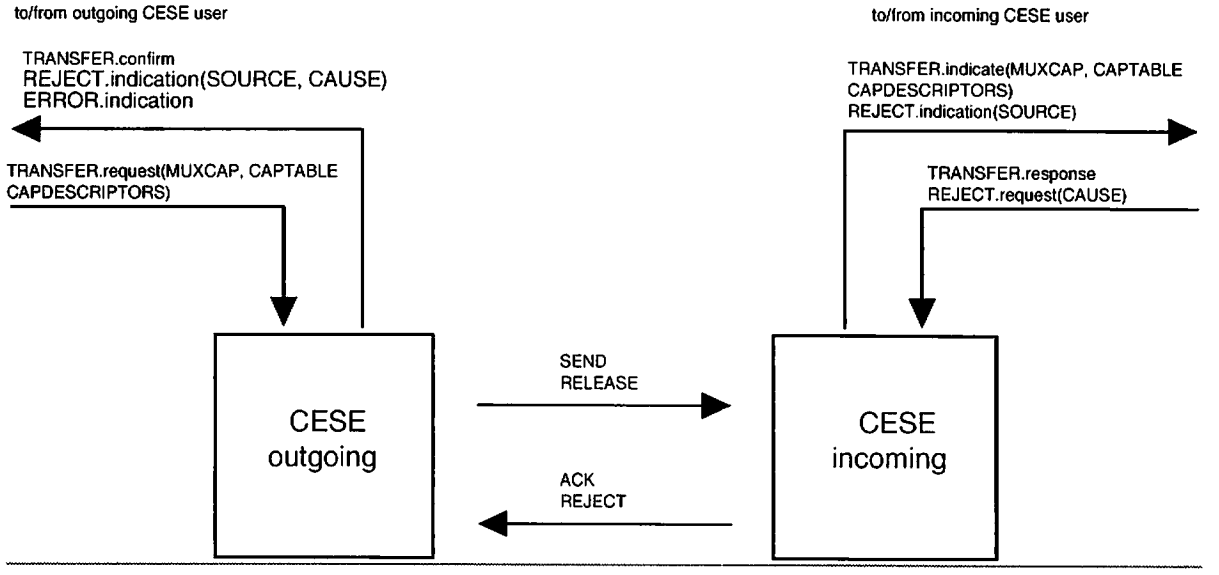

FIGURE 77/H.245

Primitives and messages in the Capability Exchange Signalling Entity

88.33.44.11 Primitive parameter default values

Where not explicitly stated in the SDL diagrams the parameters of the indication and confirm primitives assume values as shown in Table 2323.

TABLE 2323/H.245

Default primitive parameter values

primitive	parameter	default value
TRANSFER.indication	MUXCAP	SEND.multiplexCapability
	CAPTABLE	SEND.capabilityTable
	CAPDESCRIPTORS	SEND.capabilityDescriptors
REJECT.indication	SOURCE	USER
	CAUSE	null

88.33.44.22 Message field default values

Where not explicitly stated in the SDL diagrams the message fields assume values as shown in Table 2424.

TABLE 2424/H.245

Default message field values

message	field	default value [1]
SEND	sequenceNumber	out_SQ
	multiplexCapability	TRANSFER.request(MUXCAP)
	capabilityTable	TRANSFER.request(CAPTABLE)
	capabilityDescriptors	TRANSFER.request(CAPDESCRIPTORS)
ACK	sequenceNumber	in_SQ
REJ	sequenceNumber	in_SQ
	cause	REJECT.request(CAUSE)
RELEASE	-	-

Notes:

1. A message field shall not be coded, if the corresponding primitive parameter is null i.e. not present.

88.33.44.33 SDLs

The out-going CESE and the in-coming CESE procedures are expressed in SDL form in Figure 88 and Figure 99, respectively. In the SDL Figures timer Tn indicates timer T101.

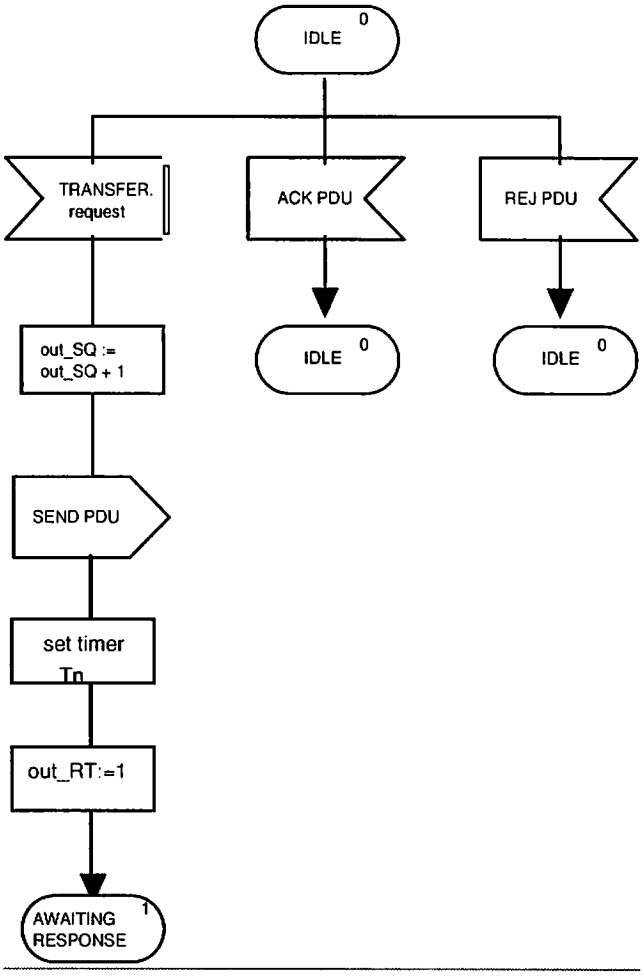

FIGURE 88(i)/H.245

Out-going CESE SDL

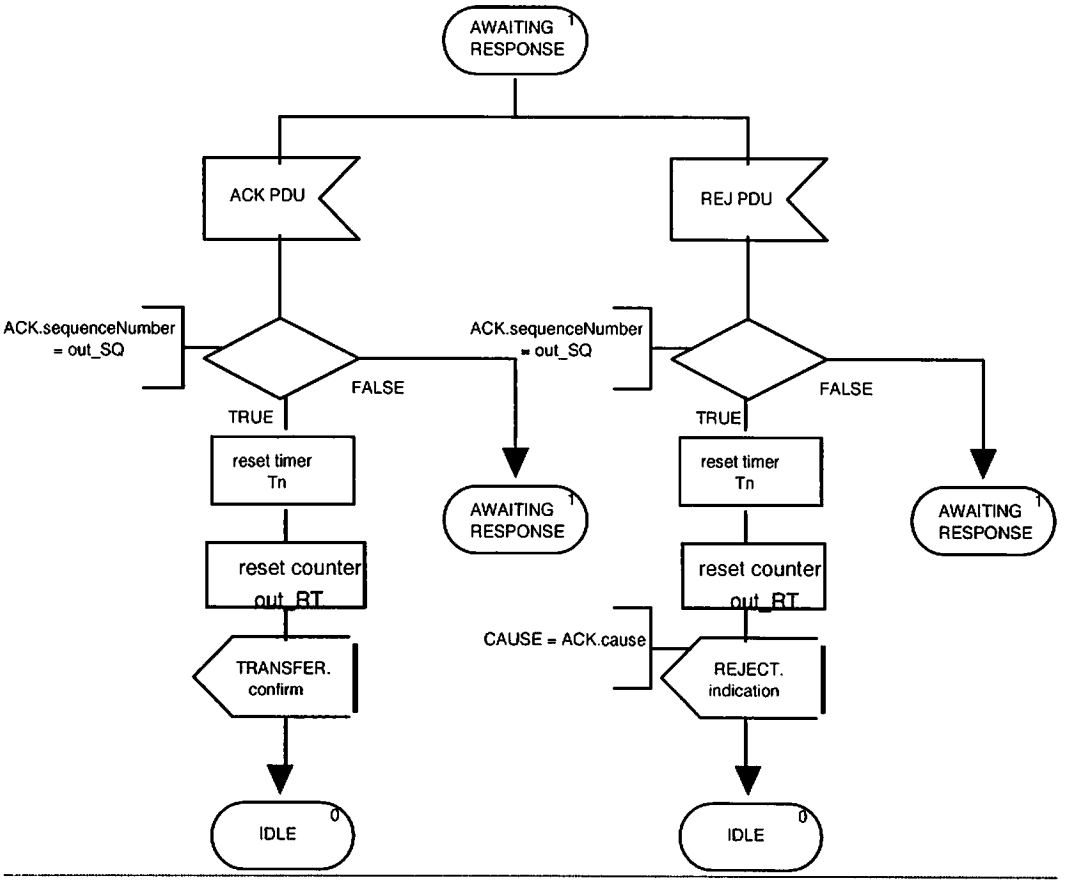

FIGURE 88(ii)/H.245

Out-going CESE SDL (continued)

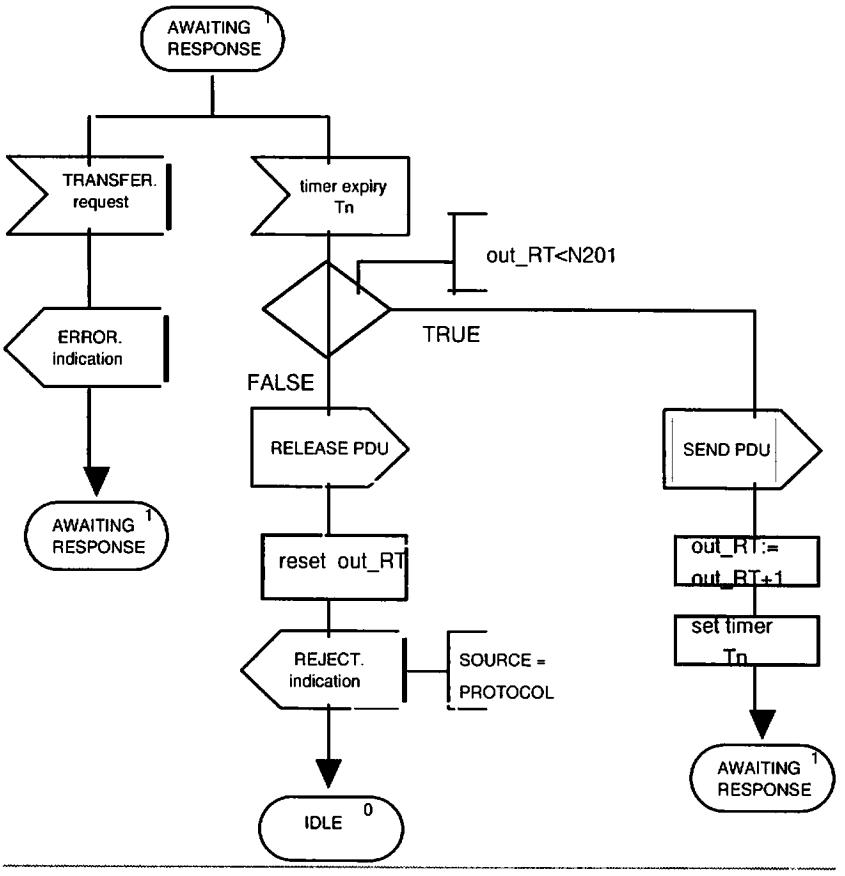

FIGURE 8~~8~~(iii)/H.245

Out-going CESE SDL (concluded)

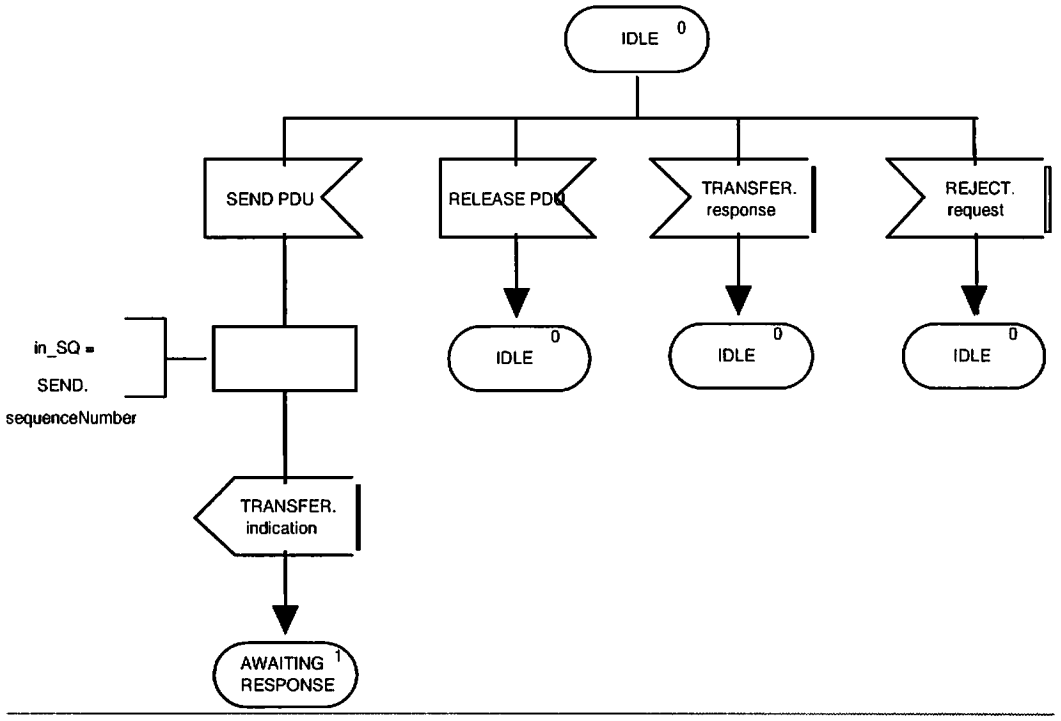

FIGURE 9~~9~~(i)/H.245

In-coming CESE SDL

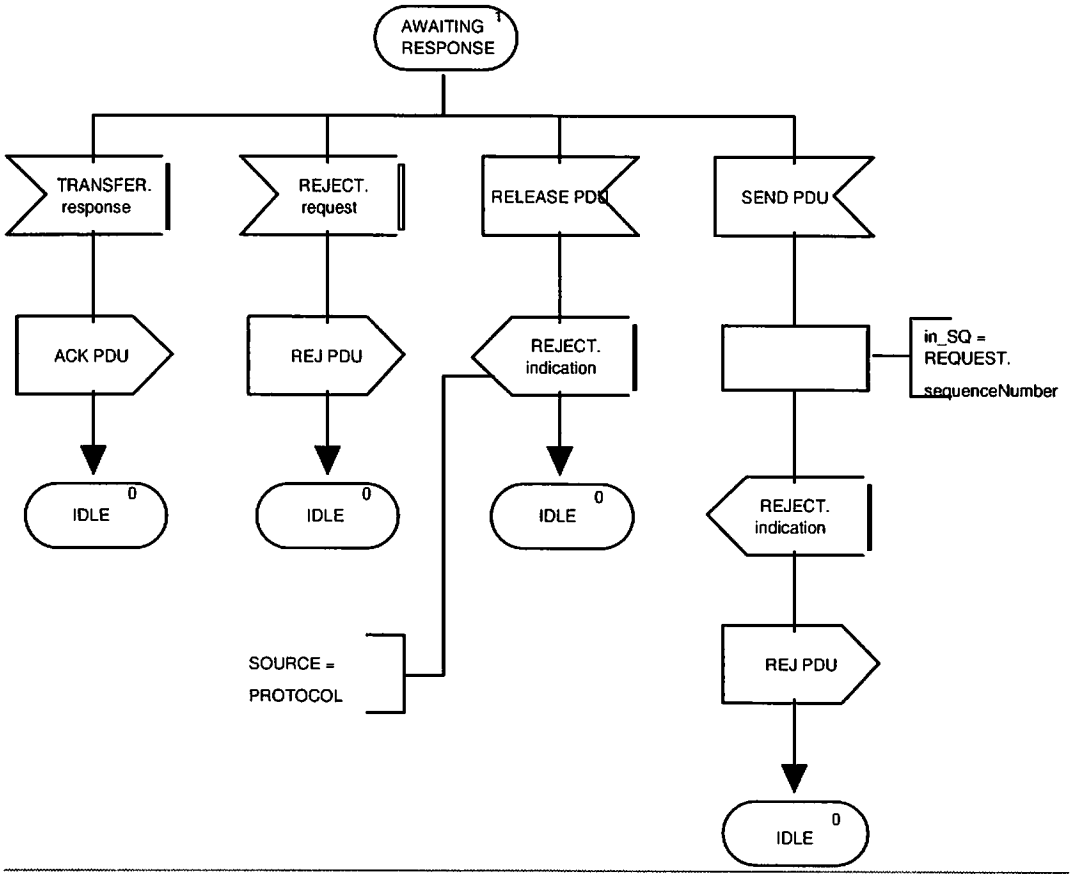

FIGURE 9~~9~~(ii)/H.245

In-coming CESE SDL (concluded)

88.44 Logical Channel signalling procedures

88.44.11 Introduction

The protocol specified here provides reliable opening and closing of uni-directional and bi-directional logical channels using acknowledgement procedures.

The protocol specified here It is referred to as the Logical Channel Signalling Entity (LCSE). Procedures are specified in terms of primitives at the interface between the LCSE and the LCSE user, and LCSE states . Protocol information is transferred to the peer LCSE via relevant messages defined in section 6.

There is an out-going LCSE and an in-coming LCSE. At each of the out-going and in-coming sides there is one instance of the LCSE for each logical channel. LCSE error conditions are reported.

The LCSE procedures provide reliable set up and release of logical channels using acknowledgement procedures. The logical channel shall only be used for audiovisual and data communication while in the ESTABLISHED state.

Either terminal may initiate the opening of a logical channel by issuing the ESTABLISH.request primitive to its LCSE. Successful opening is indicated by the ESTABLISH.confirm primitive and failure by the RELEASE.confirm primitive.

A terminal is made aware of a request to open a logical channel by receipt of the ESTABLISH.indication primitive. A terminal accepts the request by issuing the ESTABLISH.response primitive and rejects it by issuing the RELEASE.request primitive.

The logical channel shall only be used for audiovisual and data communication while in the ESTABLISHED state.

The procedures are specified separately for out-going and in-coming logical channel connections. Since a logical channel is defined as being unidirectional, two way audiovisual and data communication requires the procedures to be performed twice; once in the forward calling direction, and once in the backward calling direction.

There is no connection between the in-coming LCSE and the out-going LCSE at one side, other than via primitives to and from the LCSE user.

LCSE error conditions are reported.

Mode switching is performed by closing logical channels and opening new logical channels.

Note. Some recommendations that use this Recommendation may define some default logical channels. These shall be considered ESTABLISHEDopened from the start of communication and shall not be opened using these procedures. They may, however, be closed by these procedures, and subsequently be re-opened for the same or a different purpose.

On occasions there may be conflicting requests for logical channels, caused by both terminals initiating requests at about the same time. It may be possible to determine that there is conflict from knowledge of exchanged capabilities. On other occasions, when opening bi-directional logical channels, both terminals may initiate the opening of a channel for the same purpose, even though the exact parameters requested may be different, and both terminals have sufficient capability for both requests. Terminals shall be capable of detecting when both of these situations have arisen, any shall act as follows.

Before logical channels can be opened, one terminal must be determined as the master terminal, and the other as the slave. The protocol defined in 8.2 provides one means to make this decision. The master terminal shall reject immediately any request from the slave that it identifies as a conflicting request. The slave terminal may identify such conflicts, but shall ignore the conflict for the purposes of responding to the requests of the master.

Note: In the second type of conflict defined above, it is impossible to distinguish when two bi-directional channels are actually wanted from the case when only one is wanted. Terminals shall respond assuming that only one is wanted, but a terminal may subsequently repeat its request if the assumption was incorrect.

The following text provides an overview of the operation of the protocol. In the case of any discrepancy with the formal specification of the protocol that follows, the formal specification will supersede.

For a given logical channel, either terminal may initiate the opening of the channel by transmitting OpenLogicalChannel, indicating the data that will be conveyed on the logical channel. Transmission on the logical channel shall not start up until OpenLogicalChannelAck has been received.

A terminal may change the data type that is transmitted on a logical channel by sending another OpenLogicalChannel without first having to send CloseLogicalChannel. But no data shall be be transmitted between the time of sending the OpenLogicalChannel and when OpenLogicalChannelAck is received, as the LCSE is no longer in the ESTABLISHED state.

A logical channel may be closed by sending CloseLogicalChannel. The transmission of data on the logical channel shall stop before CloseLogicalChannel is sent.

On receiving OpenLogicalChannel, a terminal shall send OpenLogicalChannelAck if it is able to satisfy the request and OpenLogicalChannelReject if it is unable to satisfy the request. If an OpenLogicalChannel is received requesting opening of a bi-directional channel and the terminal accepts the request, it shall send an OpenLogicalChannelAck followed by OpenLogicalChannel in order to open the channel in the reverse direction. If the terminal which originated the request for a bi-directional channel sends an OpenLogicalChannelReject to the remote terminal, it shall also close down the forward logical channel which it opened previously.

A terminal that is no longer capable of processing the signals on a logical channel should take appropriate action: this should include closing the logical channel and transmitting the relevant (changed) capability information to the remote terminal.

88.44.22 Communication between the LCSE and the LCSE user

88.44.22.11 Primitives between the LCSE and the LCSE user

Communication between the LCSE and the LCSE user is performed using the primitives shown in Table 2525.

TABLE 2525/H.245

Primitives and parameters

generic name	type			
	request	indication	response	confirm
ESTABLISH	PORTNUMBER DATATYPE LC_PARAM R_PORTNUMBER R_DATATYPE R_LC_PARAM ASSOCIATED_LCN	PORTNUMBER DATATYPE LC_PARAM R_PORTNUMBER R_DATATYPE R_LC_PARAM ASSOCIATED_LCN	- [1]	-
RELEASE	CAUSE	SOURCE CAUSE	not defined [2]	-
ERROR	not defined	ERRCODE	not defined	not defined

Notes:

1. "-" means no parameters.
2. "not defined" means that this primitive does not exist.

88.44.22.22 Primitive definition

The definition of these primitives is as follows:

a) The ESTABLISH primitives are used to establish a logical channel for audiovisual and data communication.

b) The RELEASE primitives are used to release a logical channel.

c) The ERROR primitive reports LCSE errors to some management entity.

88.44.22.33 Parameter definition

The definition of the primitive parameters shown in Table 2525 are as follows:

a) The PORTNUMBER parameter specifies a port associated with this logical channel.

b) The DATATYPE parameter specifies the type of data which is to be transferred in the logical channel.

c) The LC_PARAM parameter specifies logical channel multiplexing parameters.

d) The R_PORTNUMBER parameter specifies a port associated with a logical channel to be opened in the reverse signalling direction. The presence of this parameter indicates the desire to open an associated logical channel in the reverse signalling direction.

e) The R_DATATYPE parameter specifies the type of data which is to be transferred in a logical channel to be opened in the reverse signalling direction. The presence of this parameter indicates the desire to open an associated logical channel in the reverse signalling direction.

f) The R_LC_PARAM parameter specifies logical channel multiplexing parameters of a logical channel to be opened in the reverse direction. The presence of this parameter indicates the desire to open an associated logical channel in the reverse signalling direction.

g) The ASSOCIATED_LCN parameter specifies the logical channel number of a logical channel which has already been established in the opposite signalling direction. This parameter is optional. It shall not be present when R_PORTNUMBER, R_DATATYPE, or R_LC_PARAM are present.

h) The SOURCE parameter indicates to the LCSE user the source of the logical channel release. The SOURCE parameter has the value of "USER" or "LCSE", indicating either the LCSE user, or the LCSE. The latter may occur as the result of a protocol error.

i) The CAUSE parameter indicates the reason as to why the peer LCSE user rejected a request to establish a logical channel. The CAUSE parameter is not present when the SOURCE parameter indicates "LCSE".

j) The ERRCODE parameter indicates the type of LCSE error. TABLE 3030 shows the allowed values of the ERRCODE parameter.

88.44.22.44 Uni-directional and bi-directional logical channel establishment

Table 2626 shows the ESTABLISH.request primitive parameters required when opening a uni-directional and a bi-directional logical channel.

TABLE 2626/H.245

ESTABLISH.request primitive parameters required for uni-directional and bi-directional logical channel establishment

primitive	parameter	logical channel [1]	
		uni-directional	bi-directional
ESTABLISH.request	PORTNUMBER	M	M
	DATATYPE	M	M
	LC_PARAM	M	M
	R_PORTNUMBER	null	M
	R_DATATYPE	null	M
	R_LC_PARAM	null	M
	ASSOCIATED_LCN	O [2]	null

Notes:

1. "M" - mandatory, "O" - optional, "null" - parameter shall not be present.
2. The ASSOCIATED_LCN parameter shall be present if this ESTABLISH.request primitive is in response to an earlier request to establish a bi-directional logical channel. Otherwise it shall not be present.

88.44.22.55 LCSE states

The following states are used to specify the allowed sequence of primitives between the LCSE and the LCSE user, and the exchange of messages between peer LCSEs. The states are specified separately for each of an out-going LCSE and an in-coming LCSE. The states for an out-going LCSE are:

State 0: RELEASED

The logical channel is released. The logical channel shall not be used to send out-going data.

State 1: AWAITING ESTABLISHMENT

The out-going LCSE is waiting to establish a logical channel with a peer in-coming LCSE. The logical channel shall not be used to send out-going data.

State 2: ESTABLISHED

The LCSE peer-to-peer logical channel connection has been established. The logical channel may be used to send out-going data.

State 3: AWAITING RELEASE

The out-going LCSE is waiting to release a logical channel with the peer in-coming LCSE. The logical channel shall not be used to send out-going data.

The states for an in-coming LCSE are:

State 0: RELEASED

The logical channel is released. The logical channel shall not be used to receive in-coming data.

State 1: AWAITING ESTABLISHMENT

The in-coming LCSE is waiting to establish a logical channel with a peer out-going LCSE. The logical channel shall not be used to receive in-coming data.

State 2: ESTABLISHED

An LCSE peer-to-peer logical channel connection has been established. The logical channel may be used to receive in-coming data.

88.44.22.66 State transition diagram

The allowed sequence of primitives between the LCSE and the LCSE user is defined here. The allowed sequence of primitives relates to states of the LCSE as viewed from the LCSE user. The allowed sequences are specified separately for each of an out-going LCSE and an in-coming LCSE, as shown in Figure 10~~10~~ and Figure 11~~11~~ respectively.

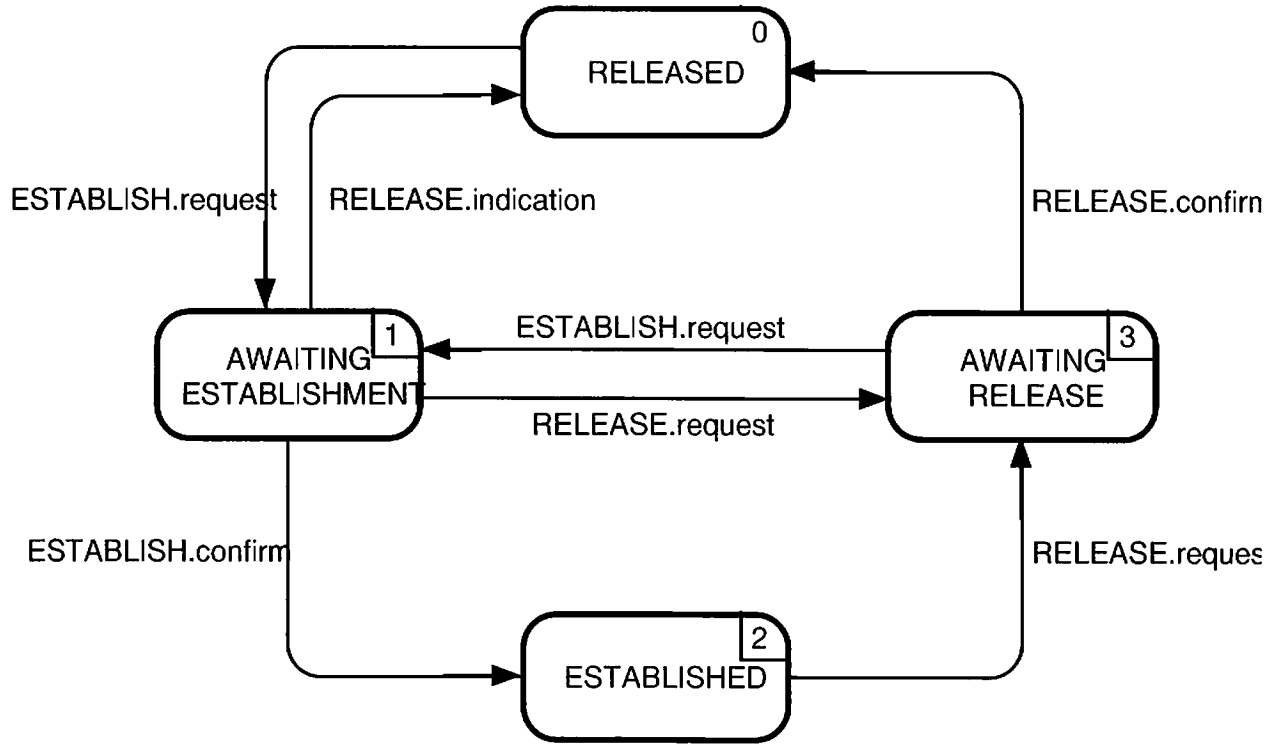

FIGURE 10~~10~~/H.245

State transition diagram for sequence of primitives at out-going LCSE

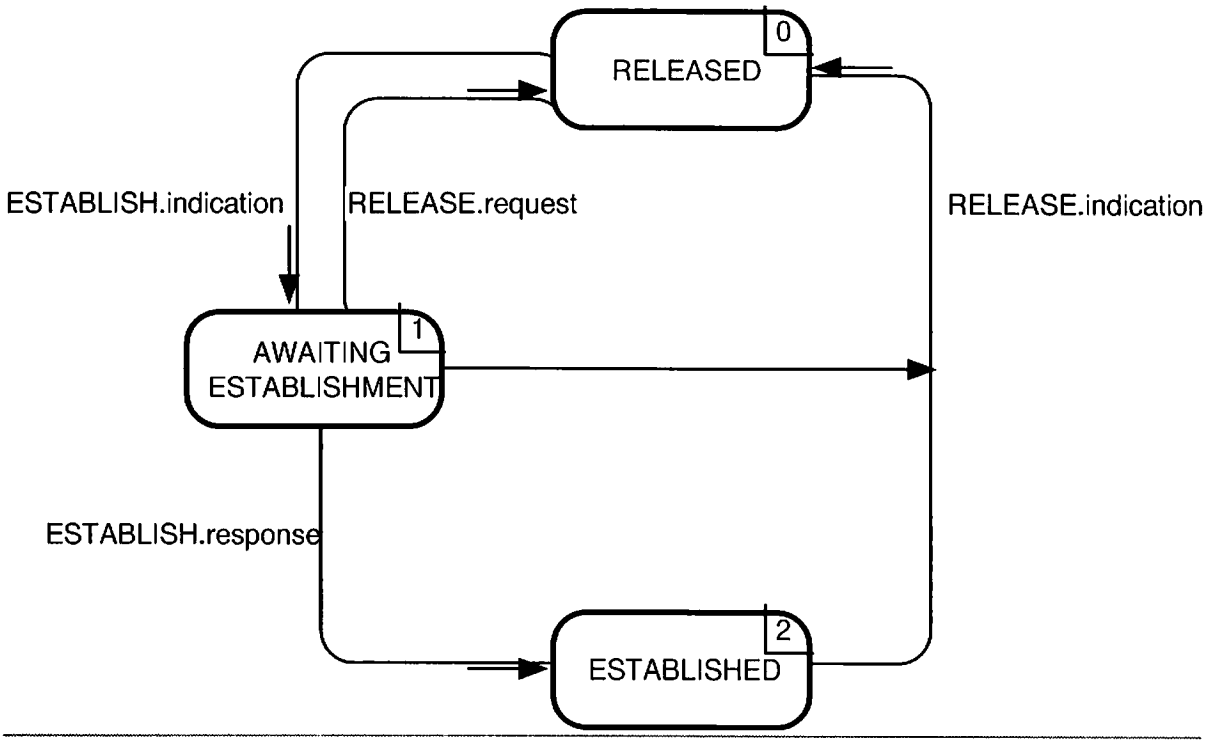

FIGURE 11~~11~~/H.245

State transition diagram for sequence of primitives at in-coming LCSE

8.44.3 Peer to peer LCSE communication

8.44.3.1 LCSE messages

Table 2~~7~~ shows the LCSE messages and fields, defined in section 6, which are relevant to the LCSE protocol.

TABLE 2727/H.245

LCSE message names and fields

function	message	direction	SDL name [2]	field
establishment	OpenLogicalChannel	O -> I [1]	OPEN	logicalChannelNumber forwardLogicalChannelParameters. portNumber forwardLogicalChannelParameters. dataType forwardLogicalChannelParameters. logicalChannelMultiplexParameters reverseLogicalChannelParameters. portNumber reverseLogicalChannelParameters. dataType reverseLogicalChannelParameters. logicalChannelMultiplexParameters associatedLogicalChannelNumber
	OpenLogicalChannelAck	O <- I	OPAK	logicalChannelNumber
	OpenLogicalChannelReject	O <- I	OPREJ	logicalChannelNumber cause
release	CloseLogicalChannel	O -> I	CLOSE	logicalChannelNumber source
	CloseLogicalChannelAck	O <- I	CLAK	logicalChannelNumber

Notes:

1. Direction: O - out-going, I - in-coming.
2. The SDL name is a short hand notation used to indicate the message in the SDL figures.

88.44.33.22 LCSE state variables

The following state variable is defined at the out-going LCSE:

out_LCN

This state variable distinguishes between out-going LCSEs. It is initialised at out-going LCSE initialisation. The value of out_LCN is used to set the logicalChannelNumber field of LCSE messages sent from an out-going LCSE. For LCSE messages received at an out-going LCSE, the message logicalChannelNumber field value is identical to the value of out_LCN.

The following state variable is defined at the in-coming LCSE:

in_LCN

This state variable distinguishes between in-coming LCSEs. It is initialised at in-coming LCSE initialisation. The value of in_LCN is used to set the logicalChannelNumber field of LCSE messages sent from an in-coming LCSE. For LCSE messages received at an in-coming LCSE, the message logicalChannelNumber field value is identical to the value of in_LCN.

88.44.33.33 LCSE timers

The following timer is specified for the out-going LCSE:

T103

This timer is used during the AWAITING ESTABLISHMENT and AWAITING RELEASE states. It specifies the maximum allowed time during which no OPEN or CLOSE message has been received, respectively.

8.4.4 LCSE procedures

8.4.4.1. Introduction

FIGURE 12 summarises the primitives and their parameters, and the messages, for each of the out-going and in-coming LCSE.

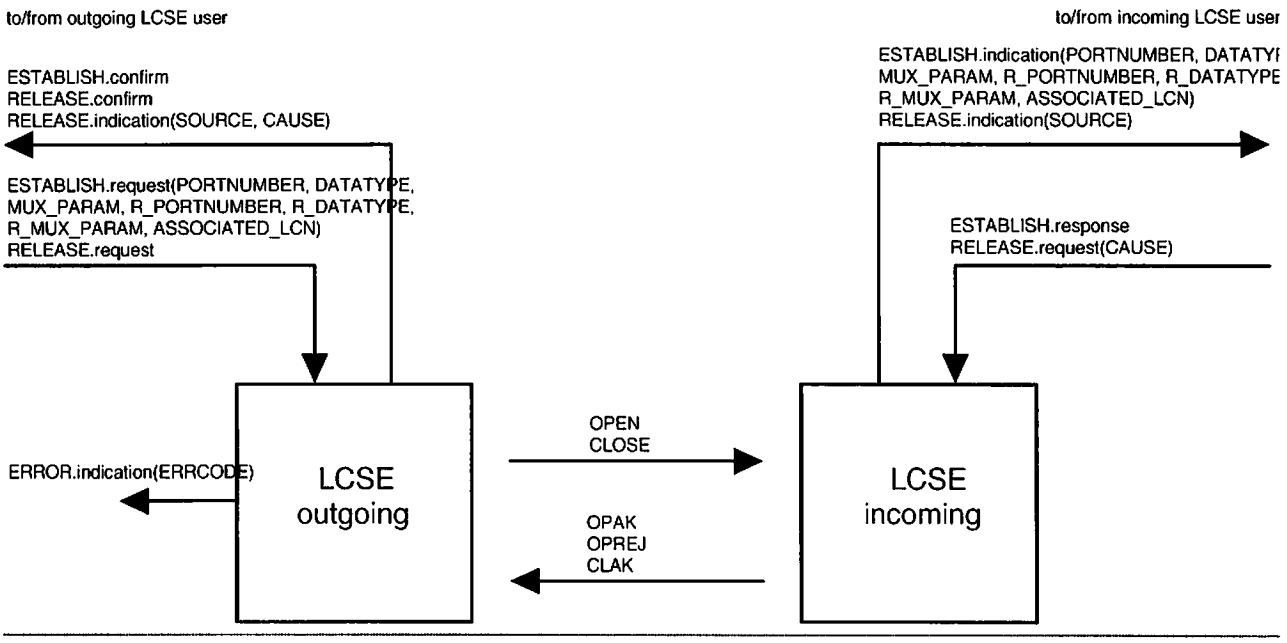

FIGURE 12/H.245

Primitives and messages in the Logical Channel Signalling Entity

8.4.4.2. Primitive parameter default values

Where not explicitly stated in the SDL diagrams the parameters of the indication and confirm primitives assume values as shown in Table 28.

TABLE 28/H.245

Default primitive parameter values

primitive	parameter	default value [1]
ESTABLISH.indication	PORTNUMBER	OPEN.forwardLogicalChannelParameters.portNumber
	DATATYPE	OPEN.forwardLogicalChannelParameters.dataType
	LC_PARAM	OPEN.forwardLogicalChannelParameters. logicalChannelMultiplexParameters
	R_PORTNUMBER	OPEN.reverseLogicalChannelParameters.portNumber
	R_DATATYPE	OPEN.reverseLogicalChannelParameters.dataType
	R_LC_PARAM	OPEN.reverseLogicalChannelParameters. logicalChannelMultiplexParameters
	ASSOCIATED_LCN	OPEN.associatedLogicalChannelNumber
RELEASE.indication	SOURCE	CLOSE.source
	CAUSE	null

Notes:

1. A primitive parameter shall be coded as null, if an indicated message field is not present in the message.

8.4.4.3. Message field default values

Where not explicitly stated in the SDL diagrams the message fields assume values as shown in Table 2929.

TABLE 2929/H.245

Default message field values

message	field	default value [1]
OPEN	logicalChannelNumber	out_LCN
	forwardLogicalChannelParameters.portNumber	ESTABLISH.request(PORTNUMBER)
	forwardLogicalChannelParameters.dataType	ESTABLISH.request(DATATYPE)
	forwardLogicalChannelParameters. logicalChannelMultiplexParameters	ESTABLISH.request(LC_PARAM)
	reverseLogicalChannelParameters.portNumber	ESTABLISH.request(R_PORTNUMBER)
	reverseLogicalChannelParameters.dataType	ESTABLISH.request(R_DATATYPE)
	reverseLogicalChannelParameters. logicalChannelMultiplexParameters	ESTABLISH.request(R_LC_PARAM)
	associatedLogicalChannelNumber	ESTABLISH.request(ASSOCIATED_LCN)
OPAK	logicalChannelNumber	in_LCN
OPREJ	logicalChannelNumber	in_LCN
	cause	RELEASE.request(CAUSE)
CLOSE	logicalChannelNumber	out_LCN
	source	user
CLAK	logicalChannelNumber	in_LCN

Notes:

1. A message field shall not be coded, if the corresponding primitive parameter is null i.e. not present.

8.4.4.4. ERRCODE parameter values

The ERRCODE parameter of the ERROR.indication primitive indicates a particular error condition. Table 3030 shows the values that the ERRCODE parameter may take at the out-going LCSE. There is no ERROR.indication primitive associated with the in-coming LCSE.

TABLE 3030/H.245

ERRCODE parameter values at out-going LCSE

error type	error code	error condition	state
inappropriate message	A	OPAK message	RELEASED
	B	OPREJ message	RELEASED ESTABLISHED
	C	CLAK	ESTABLISHED
no response from peer LCSE	D	timer T103 expiry	AWAITING ESTABLISHMENT AWAITING RELEASE

8.4.4.5. SDLs

The out-going LCSE and the in-coming LCSE procedures are expressed in SDL form in Figure 1343 and Figure 1414 respectively.

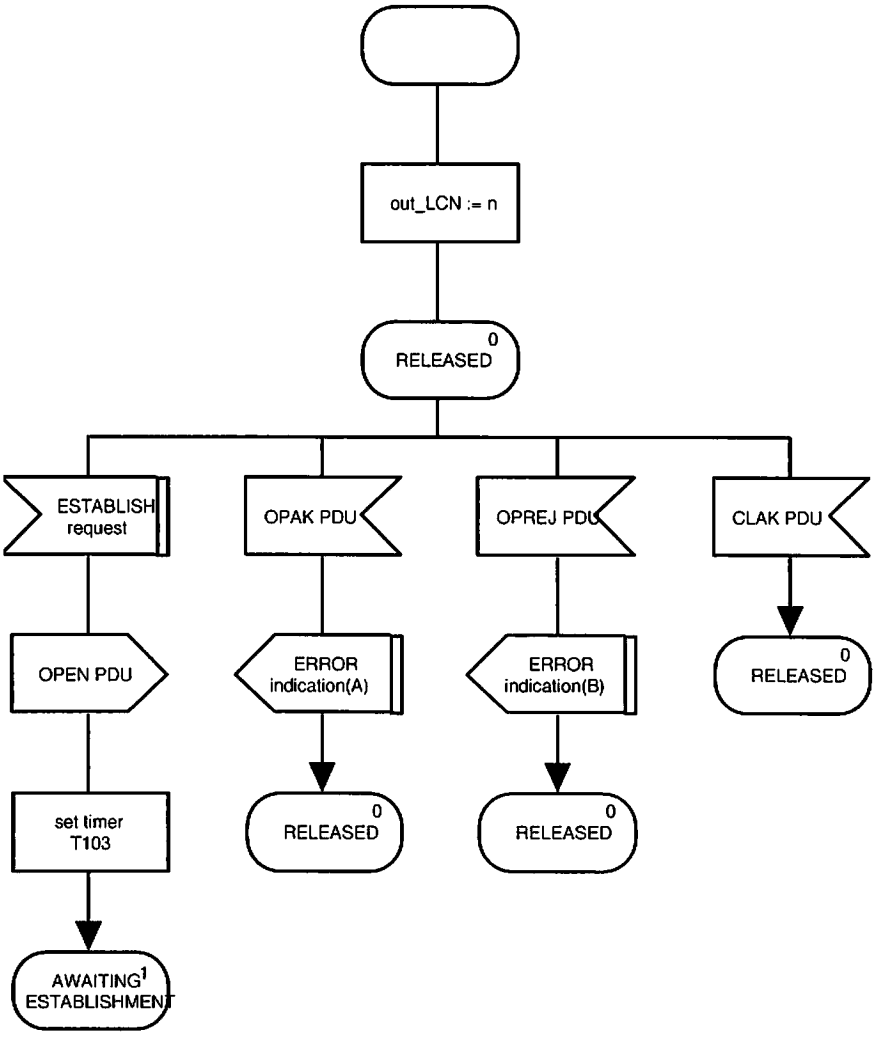

FIGURE 1343(1)/H.245

Out-going LCSE SDL

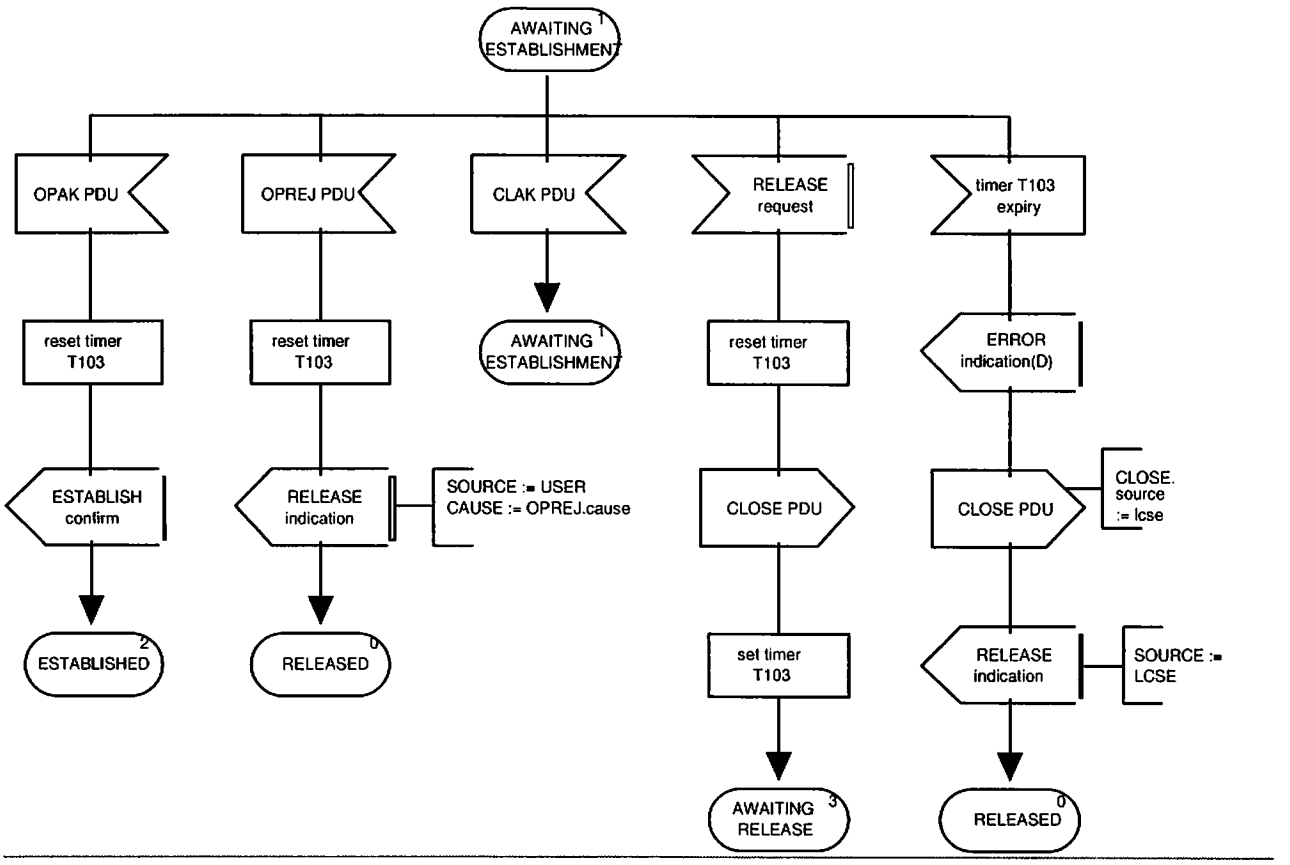

FIGURE 13~~13~~(ii)/H.245

Out-going LCSE SDL (continued)

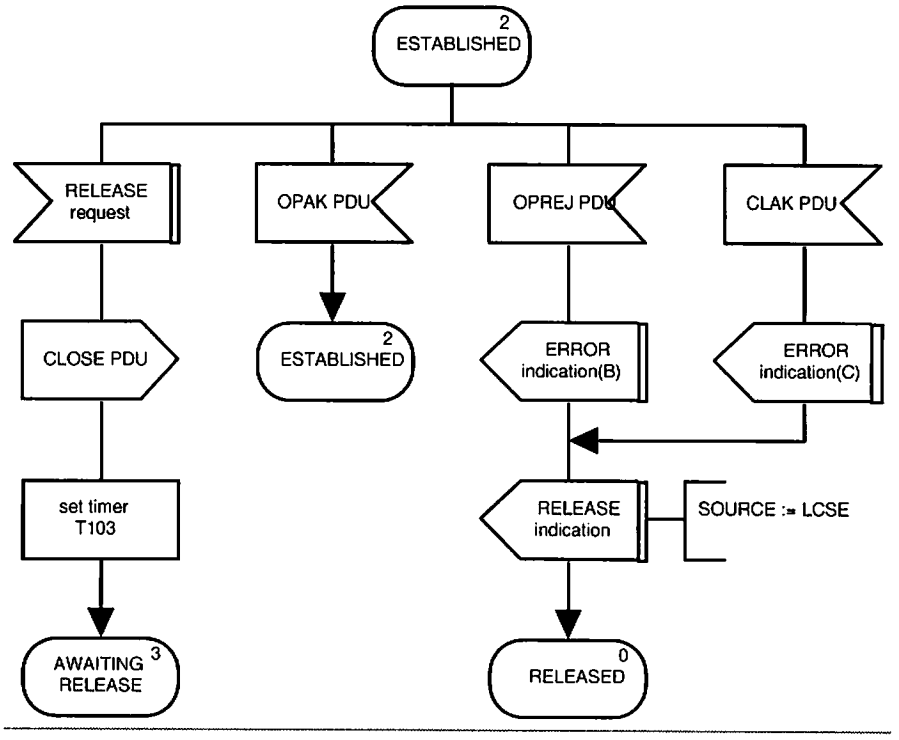

FIGURE 13~~13~~(iii)/H.245

Out-going LCSE SDL (continued)

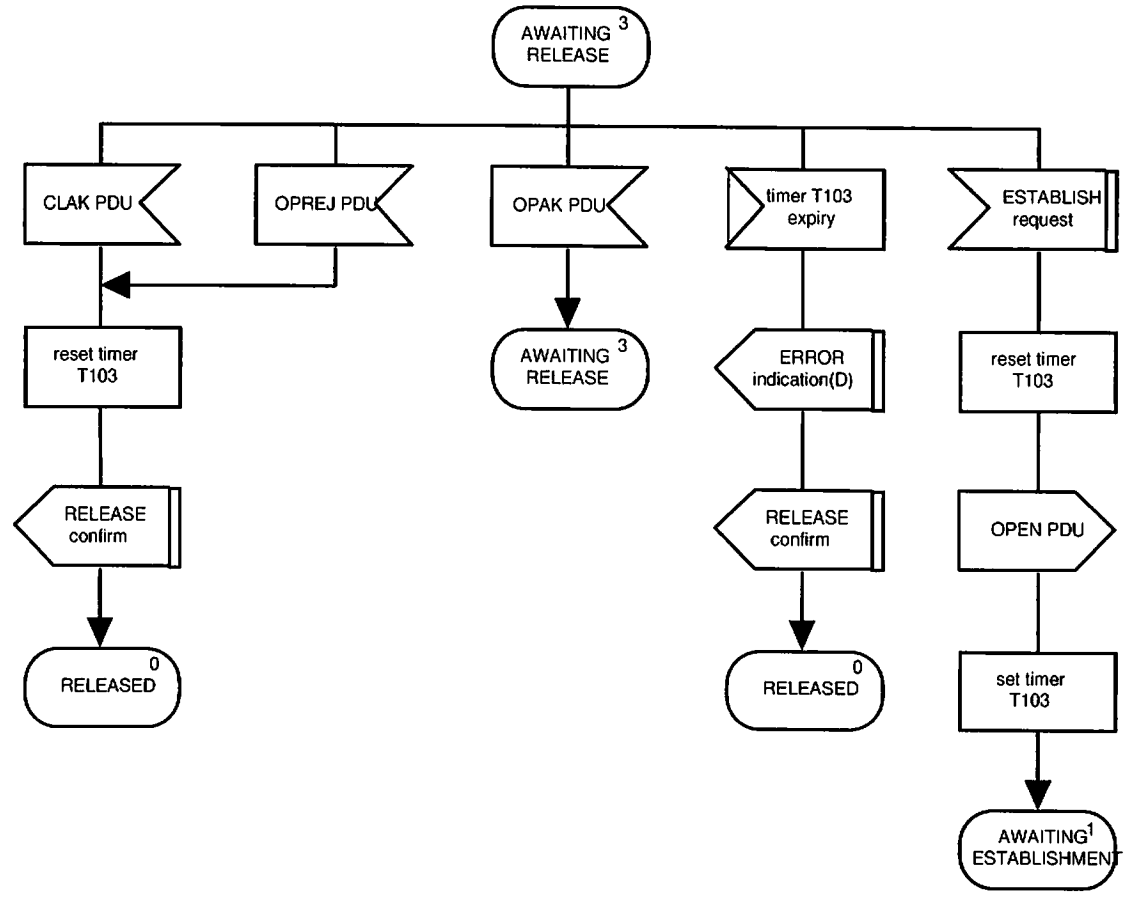

FIGURE 13~~13~~(iv)/H.245

Out-going LCSE SDL (concluded)

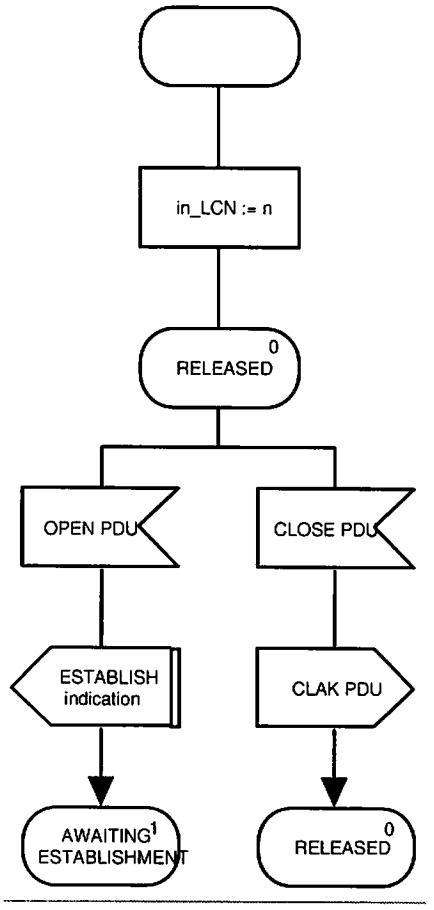

FIGURE 14~~14~~(i)/H.245

In-coming LCSE SDL

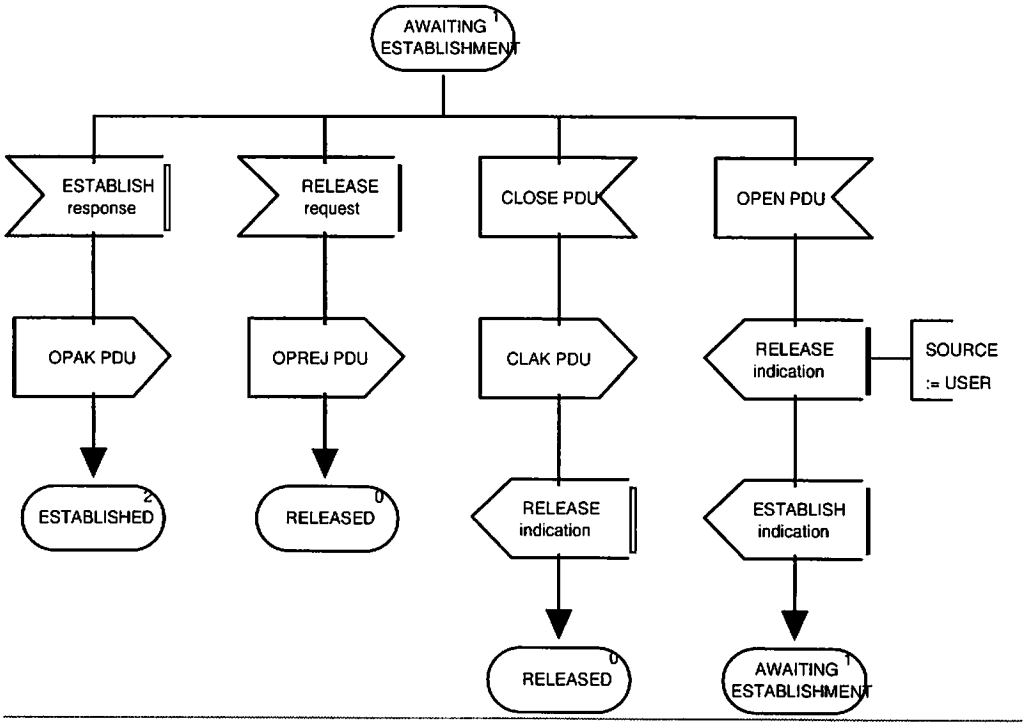

FIGURE 14~~11~~(ii)/H.245

In-coming LCSE SDL (continued)

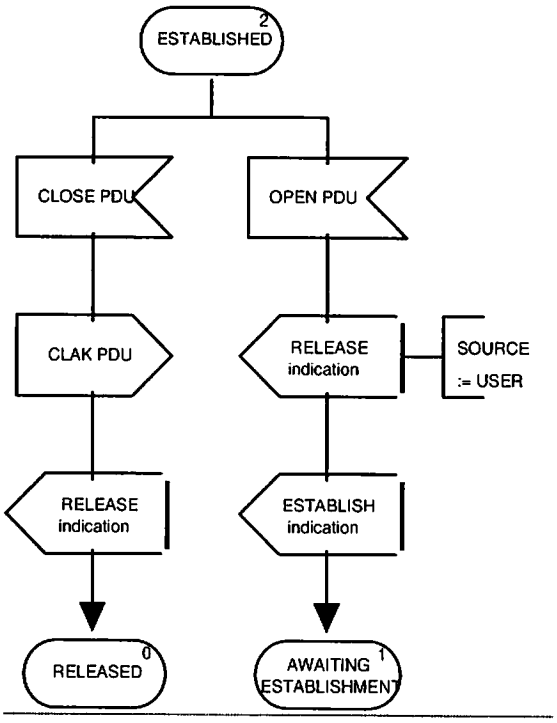

FIGURE 14~~11~~(iii)/H.245

In-coming LCSE SDL (concluded)

88.55 Bi-directional logical channel signalling procedures

88.55.11 Introduction

The protocol specified here is referred to as the Bi-directional Logical Channel Signalling Entity (B-LCSE). Procedures are specified in terms of primitives and states at the interface between the B-LCSE and the B-LCSE user. The B-LCSE uses the services of the LCSE. The B-LCSE maps primitives between the B-LCSE user and an in-coming and an out-going LCSE.

There is a master B-LCSE and a slave B-LCSE. At each of the master and slave sides there is one instance of the B-LCSE for each bi-directional logical channel. The bi-directional logical channel may only be used for audiovisual and data communication while in the ESTABLISHED state.

88.55.22 Communication between B-LCSE and the B-LCSE user

88.55.22.11 Primitives between B-LCSE and the B-LCSE user

Communication between the B-LCSE, and the B-LCSE user, is performed using the primitives shown in Table 3134. The B-LCSE primitives have a similar meaning to the LCSE primitives, except that the B-LCSE primitives relate to bi-directional logical channels.

TABLE 3134/H.245

Primitives and parameters

generic name	type			
	request	indication	response	confirm
B-ESTABLISH	PORTNUMBER DATATYPE LC_PARAM R_PORTNUMBER R_DATATYPE R_LC_PARAM	PORTNUMBER DATATYPE LC_PARAM R_PORTNUMBER R_DATATYPE R_LC_PARAM	- 1	-
B-RELEASE	CAUSE	SOURCE CAUSE	not defined 2	-

Notes:

1. "-" means no parameters.
2. "not defined" means that this primitive does not exist.

88.55.22.22 Primitive definition

The definition of these primitives is as follows:

a) The B-ESTABLISH primitives are used to establish a bi-directional logical channel for audiovisual and data communication.

b) The B-RELEASE primitives are used to release a bi-directional logical channel.

88.55.22.33 Parameter definition

The definition of the primitive parameters shown in Table 3134 are as follows:

a) The PORTNUMBER parameter specifies a port associated with the master to slave direction logical channel. This parameter is mandatory.

b) The DATATYPE parameter specifies the type of data which is to be transferred in the master to slave direction of the bi-directional logical channel. This parameter is mandatory.

c) The LC_PARAM parameter specifies master to slave direction logical channel parameters. This parameter is mandatory.

d) The R_PORTNUMBER parameter specifies a port associated with the slave to master direction of the bi-directional logical channel. This parameter is mandatory.

e) The R_DATATYPE parameter specifies the type of data which is to be transferred in the slave to master direction of the bi-directional logical channel. This parameter is mandatory.

f) The R_LC_PARAM parameter specifies slave to master direction logical channel parameters. This parameter is mandatory.

g) The SOURCE parameter indicates to the B-LCSE user the source of the logical channel release. The SOURCE parameter has the value of "USER" or "B-LCSE", indicating either the B-LCSE user, or the B-LCSE. The latter may occur as the result of a protocol error.

h) The CAUSE parameter indicates the reason as to why the peer B-LCSE user rejected a request to establish a bi-directional logical channel. The CAUSE parameter is only present when the SOURCE parameter indicates B-LCSE user.

88.55.22.44 B-LCSE states

States are used to specify the allowed sequence of primitives between the B-LCSE and the B-LCSE user, and the exchange of primitives between the B-LCSE and the out-going and in-coming LCSEs. The states are specified separately for each of a master B-LCSE and a slave B-LCSE.

The states for a master B-LCSE are identical to those of the out-going LCSE, except that the states relate to bi-directional logical channel establishment and release. The states for a master B-LCSE are:

State 0: RELEASED

State 1: AWAITING ESTABLISHMENT

State 2: ESTABLISHED

State 3: AWAITING RELEASE

The states for a slave B-LCSE are identical to those of the in-coming LCSE, except that the states relate to bi-directional logical channel establishment and release. The states for a slave B-LCSE are:

State 0: RELEASED

State 1: AWAITING ESTABLISHMENT

State 2: ESTABLISHED

88.55.22.55 State transition diagram

The allowed sequence of primitives between the B-LCSE and the B-LCSE user is defined here. The allowed sequences are specified separately for each of a master B-LCSE and a slave B-LCSE.

The state transition diagram for the sequence of primitives exchanged between a master B-LCSE and a master B-LCSE user is identical to that of the out-going LCSE (Figure 1010), except that primitives and states in the case of the B-LCSE relate to bi-directional logical channels.

The state transition diagram for the sequence of primitives exchanged between a slave B-LCSE and a slave B-LCSE user is identical to that of the in-coming LCSE (Figure 1111), except that primitives and states in the case of the B-LCSE relate to bi-directional logical channels.

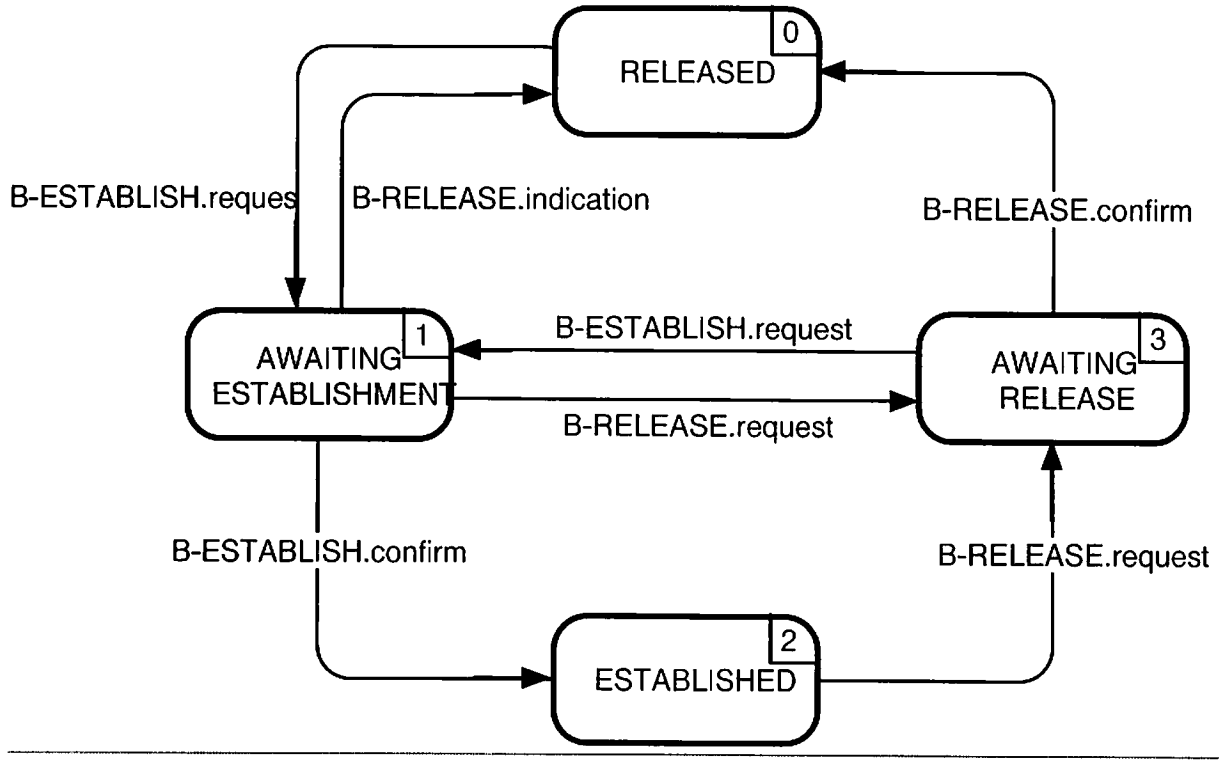

FIGURE 15~~15~~/H.245

State transition diagram for allowed sequence of primitives between master B-LCSE and master B-LCSE user

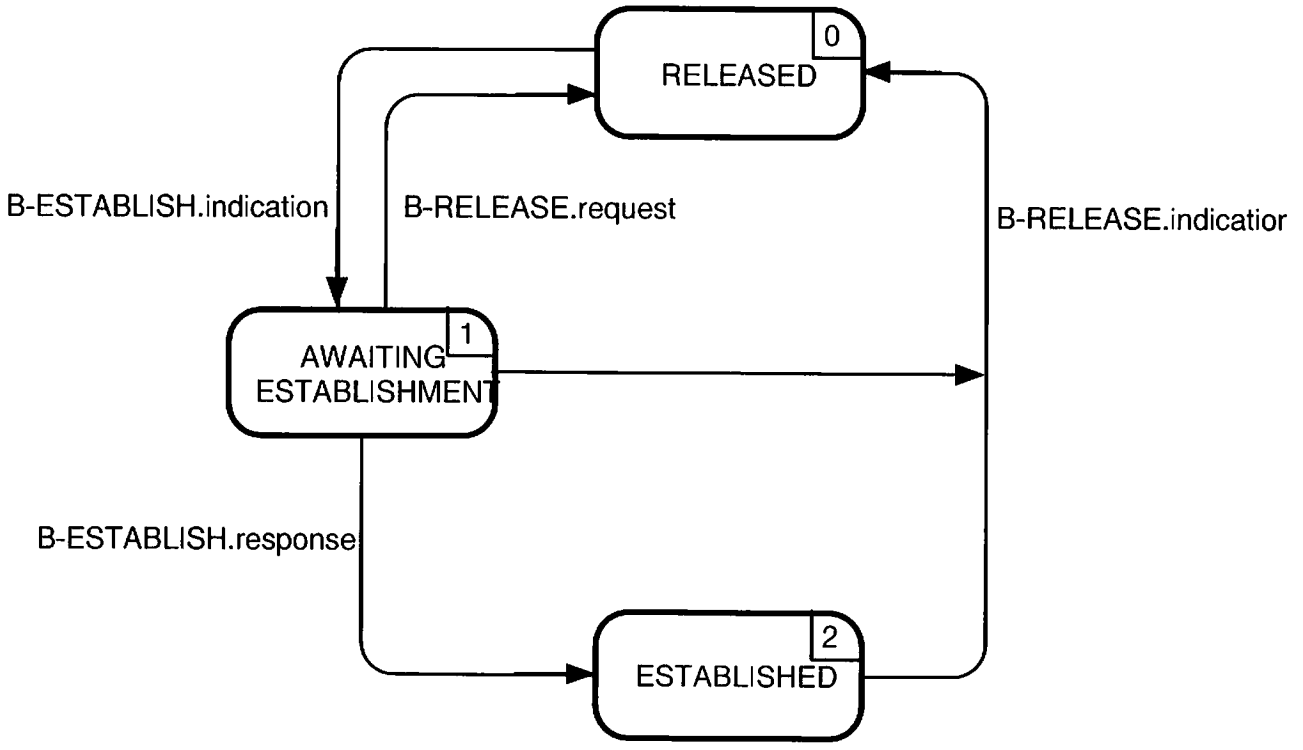

FIGURE 16~~16~~/H.245

State transition diagram for allowed sequence of primitives between slave B-LCSE and slave B-LCSE user

88.55.22.66 B-LCSE state variables

The following state variables are defined at the slave B-LCSE:

slave_R_DATATYPE

This state variable is used to store the R_DATATYPE parameter of the B-ESTABLISH.indication primitive.

slave_R_LC_PARAM

This state variable is used to store the R_LC_PARAM parameter of the B-ESTABLISH.indication primitive.

slave_R_PORTNUMBER

This state variable is used to store the R_PORTNUMBER parameter of the B-ESTABLISH.indication primitive.

The three preceding state variables are used to set the corresponding parameter of the slave B-LCSE out-going ESTABLISH.request primitive, at the time of establishment of the slave to master logical channel.

slave_in_LCN

This state variable is the logical channel number of the in-coming logical channel at the slave B-LCSE. It is set equal to the logical channel number of the in-coming LCSE at B-LCSE initiation. It is used to set the ASSOCIATED_LCN parameter in the slave B-LCSE out-going ESTABLISH.request primitive, at the time of establishment of the slave to master logical channel.

88.55.33 B-LCSE procedures

88.55.33.11 Introduction

The B-LCSE uses the services of an out-going LCSE and an in-coming LCSE. Figure 17~~1~~7 summarises the primitives for each of the master B-LCSE and slave B-LCSE.

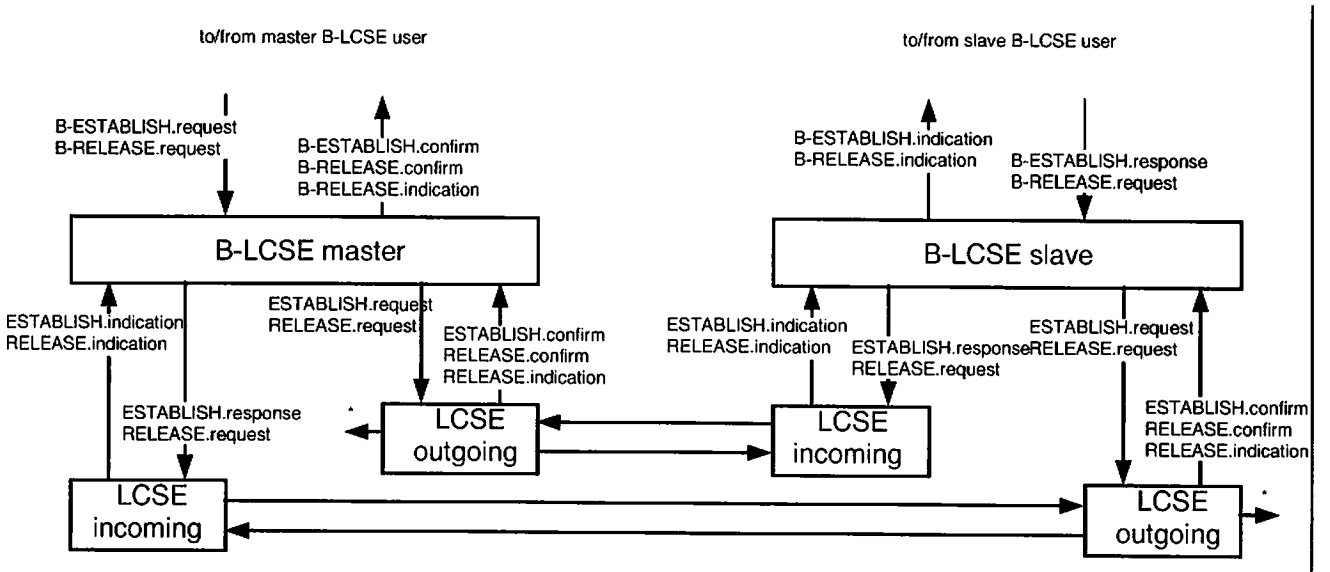

FIGURE 17~~1~~7 /H.245

Primitives in the B-LCSE

The B-LCSE procedures are specified as a compound state transition table, where the compound state is the state of the out-going LCSE and the in-coming LCSE. The master B-LCSE and the slave B-LCSE compound state transition tables are shown in Table 34~~3~~4 and Table 35~~3~~5 respectively.

88.55.33.22 B-LCSE primitive parameter default values

Where not explicitly stated in the Table 34~~3~~4 and Table 35~~3~~5 the parameters of the LCSE indication and confirm primitives assume values as shown in Table 32~~3~~2.

TABLE 3232/H.245

B-LCSE default LCSE request and response primitive parameter values

B-LSCE	primitive	parameter	default value [1]
master	B-ESTABLISH.confirm	- [2]	
	B-RELEASE.confirm	-	
	B-RELEASE.indication	SOURCE	RELEASE.indication(SOURCE) [3]
		CAUSE	RELEASE.indication(CAUSE)
slave	B-ESTABLISH.indication	PORTNUMBER	ESTABLISH.indication(PORTNUMBER)
		DATATYPE	ESTABLISH.indication(DATATYPE)
		LC_PARAM	ESTABLISH.indication(LC_PARAM)
		R_PORTNUMBER	ESTABLISH.indication(R_PORTNUMBER)
		R_DATATYPE	ESTABLISH.indication(R_DATATYPE)
		R_LC_PARAM	ESTABLISH.indication(R_LC_PARAM)
	B-RELEASE.indication	SOURCE	RELEASE.indication(SOURCE) [4]

Notes:

1. Some of these parameters are optional i.e not present, in which case the B-LCSE primitive parameter is also not present.
2. "-" means that there are no parameters associated with this parameter.
3. At the master B-LCSE the RELEASE.indication primitive is always generated by the out-going LCSE.
4. At the slave B-LCSE the RELEASE.indication primitive is generated by either the in-coming or the out-going LCSE.

8.55.33.33 LCSE primitive parameter default values

Where not explicitly stated in the Table 3434 and Table 3535 the parameters of the LCSE indication and response primitives for each of the out-going LCSE and the in-coming LCSE, used in both the master B-LCSE and the slave B-LCSE, assume the values shown in Table 3333.

TABLE 3333/H.245

B-LCSE default LCSE request and response primitive parameter values

B-LSCE	LCSE	primitive	parameter	default value
master	out-going	ESTABLISH.request	PORTNUMBER	B-ESTABLISH.request(PORTNUMBER)
			DATATYPE	B-ESTABLISH.request(DATATYPE)
			LC_PARAM	B-ESTABLISH.request(LC_PARAM)
			R_PORTNUMBER	B-ESTABLISH.request(R_PORTNUMBER)
			R_DATATYPE	B-ESTABLISH.request(R_DATATYPE)
			R_LC_PARAM	B-ESTABLISH.request(R_LC_PARAM)
			ASSOCIATED_LCN	null [1]
		RELEASE.request	- [2]	
	in-coming	ESTABLISH.response	-	
		RELEASE.request	CAUSE	dataTypeNotAvailable
slave	out-going	ESTABLISH.request	PORTNUMBER	slave_R_PORTNUMBER
			DATATYPE	slave_R_DATATYPE
			LC_PARAM	slave_R_LC_PARAM
			R_PORTNUMBER	null
			R_DATATYPE	null
			R_LC_PARAM	null
			ASSOCIATED_LCN	slave_in_LCN
		RELEASE.request	-	
	in-coming	ESTABLISH.response	-	
		RELEASE.request	CAUSE	B-RELEASE.request(CAUSE)

Notes:

1. A parameter with a value of null means that the parameter is not present.
2. "-" means that there are no parameters associated with this parameter.

88.55.33.44 B-LCSE compound state transition tables

The master B-LCSE and the slave B-LCSE compound state transition tables are shown in Table 3434 and Table 3535 respectively.

For each compound state that exists, each table expresses which B-LCSE and/or LCSE primitives are to be issued in response to a specified B-LCSE or LCSE primitive. Non-existent states are not shown. The "next state" simply identifies the compound state of the in-coming and out-going LCSE, as a result of issuing the specified primitive(s).

TABLE 3434/H.245

B-LCSE compound state transition table - master

event	compound state: in-coming LCSE/out-going LCSE		
	0/0	0/1	0/3
B-ESTABLISH.request	O: ESTABLISH.request Next state: 0/1	- Note 1	O: ESTABLISH.request Next state: 0/1
B-RELEASE.request	-	O: RELEASE.request Next state: 0/3	-
I: ESTABLISH.indication Note 2	-	 Next state: 1/1	RELEASE.request Next state: 0/3
I: RELEASE.indication	-	-	-
O: ESTABLISH.confirm	-	-	-
O: RELEASE.confirm	-	-	B-RELEASE.confirm Note 5 Next state: 0/0
O: RELEASE.indication	-	B-RELEASE.indication Note 6 Next state: 0/0	-

TABLE 3434/H.245

B-LCSE compound state transition table - master (continued)

TABLE 3434/H.245

B-LCSE compound state transition table - master (concluded)

TABLE 3535/H.245

B-LCSE compound state transition table - slave

TABLE 3535/H.245

B-LCSE compound state transition table - slave (concluded)

Notes:

1. "-" means that this event in this compound state is illegal.
2. LCSE primitives are qualified with "O:" or "I:" to indicate whether they relate to the in-coming LCSE or the out-going LCSE.
3. Compound states not shown are illegal.
4. LCSE states:

5. There may be an ERROR.indication(D) primitive associated with the out-going RELEASE.confirm primitive in this state.

6. There may be an ERROR.indication(D) primitive associated with the out-going RELEASE.indication primitive in this state.

7. The in-coming ESTABLISH.indication primitive in this state was immediately preceded by an in-coming RELEASE.indication primitive. The RELEASE.indication primitive resulted in no change of state.

8. There is either an ERROR.indication(B) or an ERROR.indication(C) primitive associated with the out-going RELEASE.indication primitive in this state. A management signal should inform the peer B_LCSE that an error has occurred.

9. In the master B-LCSE state 1/1 is a transit state. Upon entry to this state, an in-coming LCSE primitive is immediately issued. No events are serviced in this state.

10. When an in-coming ESTABLISH.indication primitive is received at the slave B-LCSE, the slave side state variables are set as shown in Table 3636.

TABLE 3636/H.245

Slave B-LCSE state variable values

88.66 Close Logical Channel procedures

88.66.11 Introduction

These procedures are used by a terminal to request the closure of an in-coming uni-directional logical channel. The procedures are referred to here as the Close Logical Channel Signalling Entity (CLCSE). Procedures are specified in terms of primitives and states at the interface between the CLCSE and the CLCSE user. Protocol information is transferred to the peer CLCSE via relevant messages defined in section 6. There is an out-going CLCSE and an in-coming CLCSE. At each of the out-going and in-coming ends there is one instance of the CLCSE per call.

If a terminal is incapable of processing the in-coming signals, it may use these procedures to request the closing of the relevant logical channels.

Note: this is not the only use of these procedures.

Note. If a change of capability has the result that the current mode is no longer receivable/decodable, there shall be a mode switch as soon as possible to a mode that can be received and decoded.

88.66.22 Communication between CLCSE and CLCSE user

88.66.22.11 Primitives between CLCSE and CLCSE user

Communication between the CLCSE and CLCSE user, is performed using the primitives shown in Table 3737.

TABLE 3737/H.245

Primitives and parameters

generic name	type			
	request	indication	response	confirm
TRANSFER	LC-NUMBER	LC-NUMBER	LC-NUMBER	LC-NUMBER
REJECT	LC-NUMBER CAUSE	LC-NUMBER SOURCE CAUSE	not defined [1]	not defined

Notes:

1. "not defined" means that this primitive is not defined.

88.66.22.22 Primitive definition

The definition of these primitives is as follows:

a) The TRANSFER primitives are used for the transfer of the close logical channel request.

b) The REJECT primitives are used to reject the closing of a logical channel.

88.66.22.33 Parameter definition

The definition of the primitive parameters shown in Table 3737 are as follows:

a) The LC-NUMBER parameter is the logical channel number. This parameter is mandatory in the TRANSFER primitives.

b) The SOURCE parameter indicates the source of the REJECT.indication primitive. The SOURCE parameter has the value of "USER" or "PROTOCOL". The latter case may occur as the result of a timer expiry.

c) The CAUSE parameter indicates the reason for refusal to close a logical channel. The CAUSE parameter is not present when the SOURCE parameter indicates "PROTOCOL".

88.66.22.44 CLCSE states

The following states are used to specify the allowed sequence of primitives between the CLCSE and the CLCSE user.

The states for an out-going CLCSE are:

State 0: IDLE

The CLCSE is idle.

State 1: AWAITING RESPONSE

The CLCSE is waiting for a response from the remote CLCSE.

The states for an in-coming CLCSE are:

State 0: IDLE

The CLCSE is idle.

State 1: AWAITING RESPONSE

The CLCSE is waiting for a response from the CLCSE user.

88.66.22.55 State transition diagram

The allowed sequence of primitives between the CLCSE and the CLCSE user is defined here. The allowed sequences are specified separately for each of an out-going CLCSE and an in-coming CLCSE, as shown in Figure 1818 and Figure 1919 respectively.

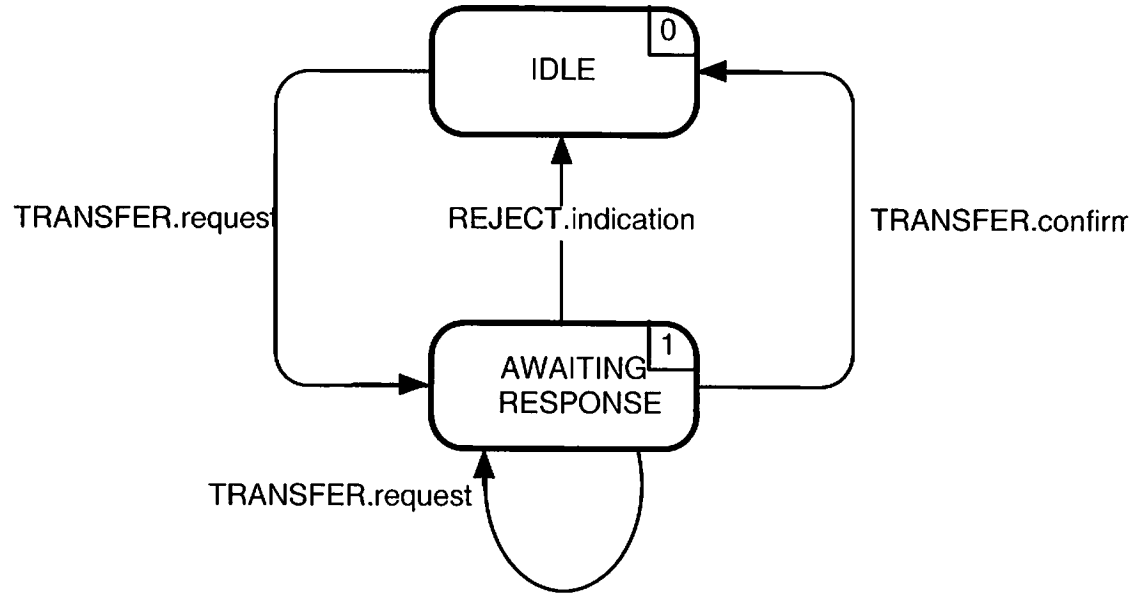

FIGURE 1818/H.245

State transition diagram for sequence of primitives at CLCSE out-going

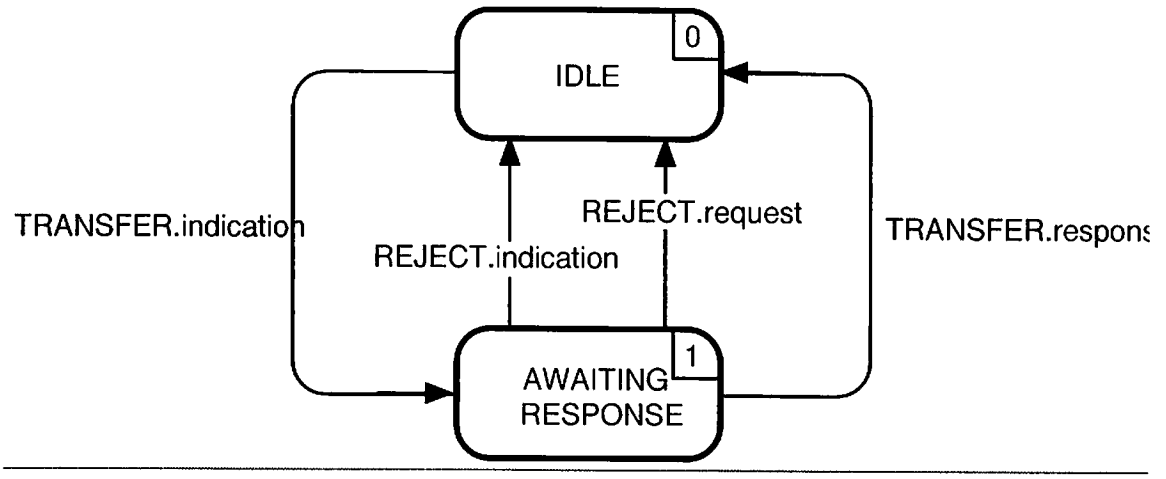

FIGURE 1949/H.245

State transition diagram for sequence of primitives at CLCSE in-coming

88.66.33 Peer to peer CLCSE communication

88.66.33.11 Messages

Table 3838 shows the CLCSE messages and fields, defined in section 6, which are relevant to the CLCSE protocol.

TABLE 3838/H.245

CLCSE message names and fields

function	message	direction	SDL name [2]	field
transfer	RequestChannelClose	O -> I [1]	SEND	logicalChannelNumber
	RequestChannelCloseAck	O <- I	ACK	logicalChannelNumber
	RequestChannelCloseReject	O <- I	REJ	logicalChannelNumber
reset	RequestChannelCloseRelease	O -> I	RELEASE	-

Notes:

1. Direction: O - out-going, I - in-coming.
2. The SDL name is a short hand notation used to indicate the message in the SDL figures.

88.66.33.22 CLCSE timers

The following timer is specified for the out-going CLCSE:

T108

This timer is used during the AWAITING RESPONSE state. It specifies the time before an error message is generated, during which no ACK or REJ message has been received.

88.66.44 CLCSE procedures

Figure 2020 summarises the CLCSE primitives and their parameters, and messages, for each of the out-going and in-coming CLCSE.

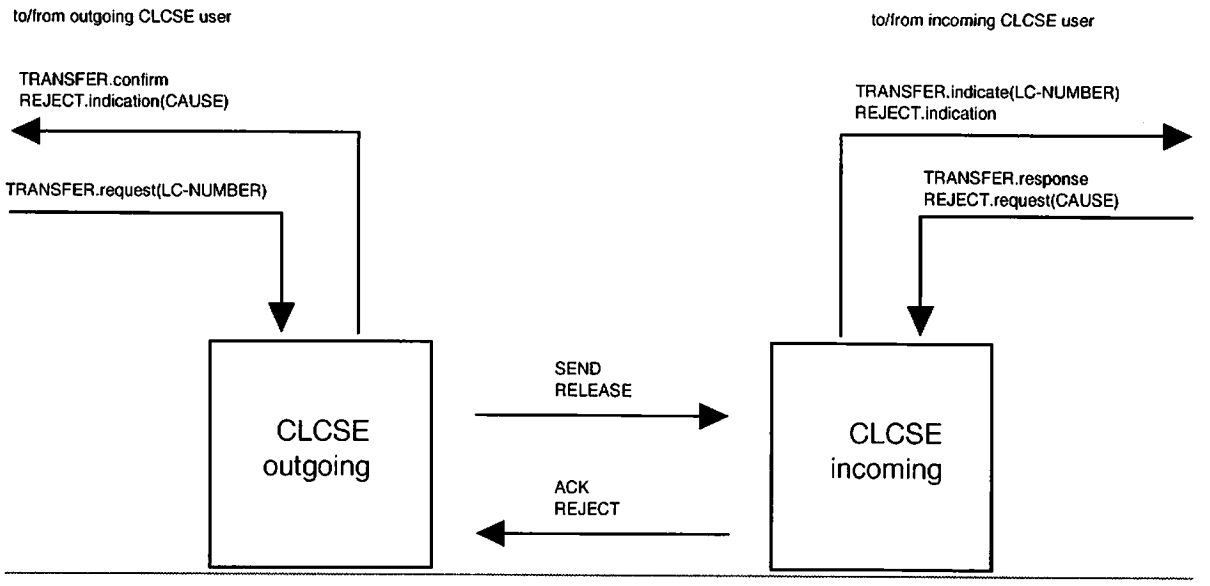

FIGURE 2020/H.245

Primitives and messages in the Close Logical Channel Signalling Entity

88.66.44.11 Primitive parameter default values

Where not explicitly stated in the SDL diagrams the parameters of the indication and confirm primitives assume values as shown in Table 3939.

TABLE 3939/H.245

Default primitive parameter values

primitive	parameter	default value
TRANSFER.indication	LC-NUMBER	SEND.logicalChannelNumber
TRANSFER.confirm	LC-NUMBER	ACK.logicalChannelNumber
REJECT.indication	LC-NUMBER	REJ.logicalChannelNumber
	SOURCE	USER
	CAUSE	null

88.66.44.22 Message field default values

Where not explicitly stated in the SDL diagrams the message fields assume values as shown in Table 4040.

TABLE 4040/H.245

Default message field values

message	field	default value
SEND	logicalChannelNumber	TRANSFER.request(LC-NUMBER)
ACK	logicalChannelNumber	TRANSFER.response(LC-NUMBER)
REJ	logicalChannelNumber	TRANSFER.response(LC-NUMBER)
	cause	REJECT.request(CAUSE)
RELEASE	-	-

88.66.44.33 SDLs

The out-going CLCSE and the in-coming CLCSE procedures have the same SDLs as for the CESE, which are shown in Figure 88 and Figure 99 respectively. In the case of the CLCSE, timer Tn in Figure 88 and Figure 99 indicates timer T107. There is no sequence number in the CLCSE. Actions relating to sequence numbers in Figure 88 and Figure 99 should be ignored.

88.77 Open Bi-directional Channel procedures

88.77.11 Introduction

These procedures are used by slave terminals to request a master terminal to open a bi-directional logical channel. They are referred to here as the Open Bi-directional Channel Signalling Entity (OBCSE). Procedures are specified in terms of primitives and states at the interface between the OBCSE and the OBCSE user. Protocol information is transferred to the peer OBCSE via relevant messages defined in section 6. There is an out-going OBCSE and an in-coming OBCSE. At each of the out-going and in-coming ends there is one instance of the OBCSE for each call.

88.77.22 Communication between OBCSE and OBCSE user

88.77.22.11 Primitives between OBCSE and OBCSE user

Communication between the OBCSE and OBCSE user, is performed using the primitives shown in Table 4141.

TABLE 4141/H.245

Primitives and parameters

generic name	type			
	request	indication	response	confirm
TRANSFER	B-LC-PARAM	B-LC-PARAM	- [1]	-
REJECT	CAUSE	SOURCE CAUSE	not defined [2]	not defined

Notes:

1. "-" means no parameters
2. "not defined" means that this primitive is not defined.

88.77.22.22 Primitive definition

The definition of these primitives is as follows:

a) The TRANSFER primitives are used for transfer of the bi-directional channel information request.

b) The REJECT primitives are used to reject the opening of a bi-directional logical channel.

88.77.22.33 Parameter definition

The definition of the primitive parameters shown in Table 4141 are as follows:

a) The B-LC-PARAM parameter is the bi-directional logical channel parameters, containing the forward and reverse logical channel parameters. This parameter is mandatory. There may be multiple B-LC-PARAM associated with the TRANSFER primitives.

b) The SOURCE parameter indicates the source of the REJECT.indication primitive. The SOURCE parameter has the value of "USER" or "PROTOCOL". The latter case may occur as the result of a timer expiry.

c) The CAUSE parameter indicates the reason for refusal to open a bi-directional logical channel. The CAUSE parameter is not present when the SOURCE parameter indicates "PROTOCOL".

88.77.22.44 OBCSE states

The following states are used to specify the allowed sequence of primitives between the OBCSE and the OBCSE user.

The states for an out-going OBCSE are:

State 0: IDLE

The OBCSE is idle.

State 1: AWAITING RESPONSE

The OBCSE is waiting for a response from the remote OBCSE.

The states for an in-coming OBCSE are:

State 0: IDLE

The OBCSE is idle.

State 1: AWAITING RESPONSE

The OBCSE is waiting for a response from the OBCSE user.

88.77.22.55 State transition diagram

The allowed sequence of primitives between the OBCSE and the OBCSE user is defined here. The allowed sequences are specified separately for each of an out-going OBCSE and an in-coming OBCSE, as shown in Figure 2124 and Figure 2222 respectively.

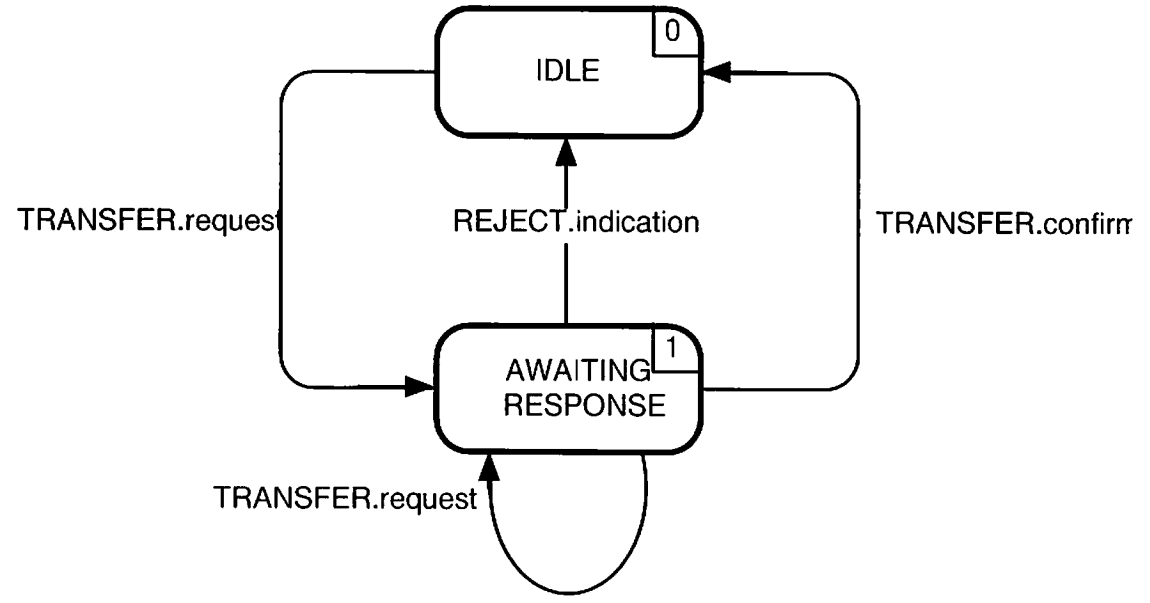

FIGURE 2124/H.245

State transition diagram for sequence of primitives at OBCSE out-going

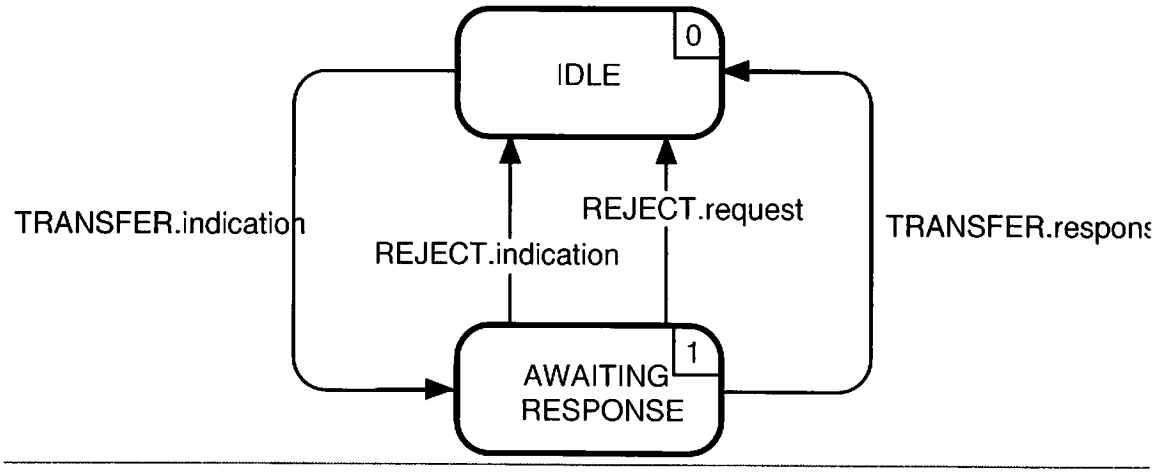

FIGURE 2222/H.245

State transition diagram for sequence of primitives at OBCSE in-coming

88.77.33 Peer to peer OBCSE communication

88.77.33.11 Messages

Table 4242 shows the OBCSE messages and fields, defined in section 6, which are relevant to the OBCSE protocol.

TABLE 4242/H.245

OBCSE message names and fields

function	message	direction	SDL name [2]	field
transfer	OpenBiDirectionalChannelRequest	O -> I [1]	SEND	sequenceNumber biDirectionalRequests
	OpenBiDirectionalChannelAck	O <- I	ACK	sequenceNumber
reject	OpenBiDirectionalChannelReject	O <- I	REJ	sequenceNumber cause
reset	OpenBiDirectionalChannelRelease	O -> I	RELEASE	-

Notes:

1. Direction: O - out-going, I - in-coming.
2. The SDL name is a short hand notation used to indicate the message in the SDL figures.

88.77.33.22 OBCSE state variables

The following state variables are defined at the out-going OBCSE:

out_SQ

This state variable is used to indicate the most recent SEND message. It is incremented by one and mapped to the SEND message sequenceNumber field before transmission of the SEND message. Arithmetic performed on out_SQ is modulo 256.

The following state variables are defined at the in-coming OBCSE:

in_SQ

This state variable is used to store the value of the sequenceNumber field of the most recently received SEND message. The ACK and REJ messages have their sequenceNumber fields set to the value of in_SQ, before being sent to the peer OBCSE.

88.77.33.33 OBCSE timers

The following timer is specified for the out-going OBCSE:

T107

This timer is used during the AWAITING RESPONSE state. It specifies the time before an error message is generated, during which no ACK or REJ message has been received.

88.77.44 OBCSE procedures

Figure 2323 summarises the OBCSE primitives and their parameters, and messages, for each of the out-going and in-coming OBCSE.

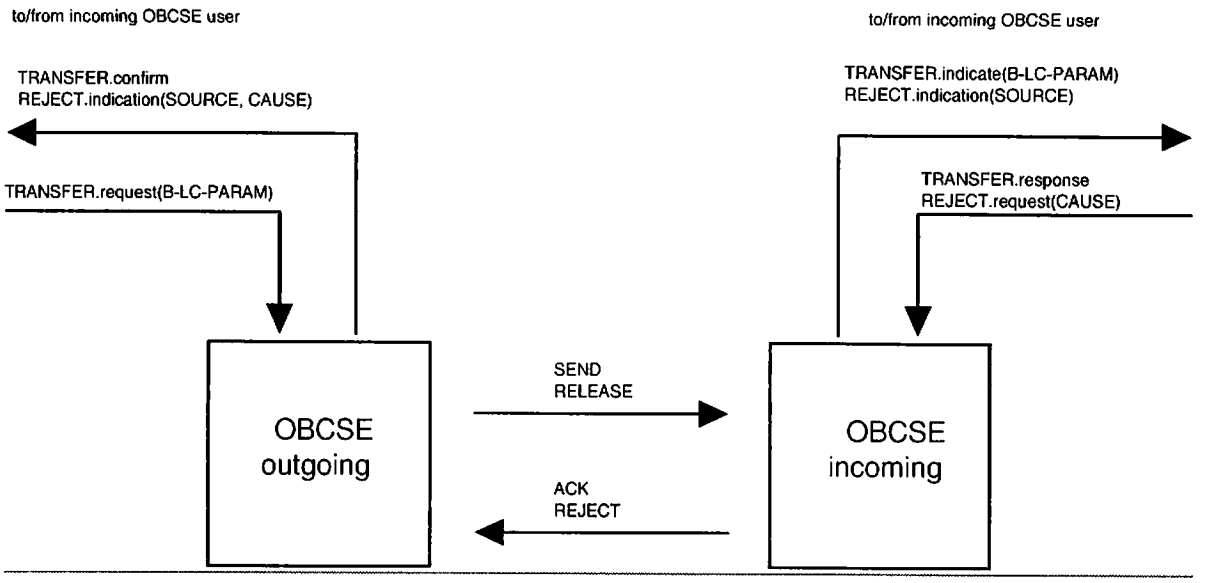

FIGURE 2323/H.245

Primitives and messages in the Open Bi-Directional Logical Channel Signalling Entity

88.77.44.11 Primitive parameter default values

Where not explicitly stated in the SDL diagrams the parameters of the indication and confirm primitives assume values as shown in Table 4343.

TABLE 4343/H.245

Default primitive parameter values

primitive	parameter	default value
TRANSFER.indication	B-LC-PARAM	SEND.biDirectionalRequests
REJECT.indication	SOURCE	USER
	CAUSE	null

88.77.44.22 Message field default values

Where not explicitly stated in the SDL diagrams the message fields assume values as shown in Table 4444.

TABLE 4444/H.245

Default message field values

message	field	default value [1]
SEND	sequenceNumber	out_SQ
	biDirectionalRequests	TRANSFER.request(B-LC-PARAM)
ACK	sequenceNumber	in_SQ
REJ	sequenceNumber	in_SQ
	cause	REJECT.request(CAUSE)
RELEASE	-	-

Notes:

1. A message field shall not be coded, if the corresponding primitive parameter is null i.e. not present.

8.7.4.3 SDLs

The out-going OBCSE and the in-coming OBCSE procedures have the same SDLs as for the CESE, shown in Figure 8 and Figure 9 respectively. In the case of the OBCSE, timer Tn in Figure 8 and Figure 9 indicates timer T107.

88.88 H.223 Multiplex Table Procedures

88.88.11 Introduction

The multiplex table serves to associate each octet within a H.223 MUX-message [8] with a particular logical channel number. The H.223 multiplex table may have up to 16 entries, numbered from 0 to 15. Table entries 1 to 15 shall be sent from transmitters to receivers as specified in the following procedures.

The procedures described here are referred to as the Multiplex Table Signalling Entity (MTSE). Procedures are specified in terms of primitives and states at the interface between the MTSE and the MTSE user. Protocol information is transferred to the peer MTSE via relevant messages defined in section 6.

There is an out-going MTSE and an in-coming MTSE. There is one instance of the MTSE for each multiplex table entry.

88.88.11.11 Multiplex Table Entry procedure

A transmit terminal uses the MultiplexEntrySend message to signal to a remote terminal one or more new multiplex table entries. The remote terminal may accept or reject the new multiplex table entries. If the remote terminal accepts a multiplex table entry, the previous entry at the given entry number is replaced with the new entry. Acceptance or rejection of the entry is signalled to the originating terminal using the MultiplexEntrySendAck or the MultiplexEntrySendReject message, respectively. If a MultiplexEntrySendAck or a MultiplexEntrySendReject message is not received within the interval specified by the timer T104, then an error message is generated., the transmitter shall send another MultiplexEntrySend that references the same set of multiplex table entries as in the previous attempt, in order to ensure that the multiplex table entries in the transmitter and receiver are kept in synchronism.

88.88.11.22 The use of Multiplex Table Entries

A multiplex table entry that is currently being updated i.e. AWAITING RESPONSE state, may not be used by the transmitter.

The transmitter may deactivate a multiplex table entry. The transmitter shall at no time use a multiplex table entry that is deactivated. Before transmitting a MultiplexEntrySend, the transmitter shall stop using the entries that are described by it. It shall not restart using those entries until it has received a MultiplexEntrySendAck. This procedure is used because if the use of these multiplex table entries is not stopped before sending the MultiplexEntrySend, errors may cause an ambiguity in the receiver.

The transmitter shall stop using deactivated entries before sending the MultiplexEntrySend indicating that they have been deactivated. Deactivated entries may by used again at any time by transmitting a MultiplexEntrySend message for activating that entry. Deactivating entries that are no longer required by the transmitter may increase the probability of detecting errors in the H.223 Multiplex Code field.

Note: While some multiplex table entries are being updated, other (active) entries may continue to be used. Also, a multiplex table entry may be deleted in the same MultiplexEntry send that is used to modify other multiplex table entries.

At the start of communication, unless specified otherwise in an appropriate recommendation, only table entry 0 is available for transmission, and table entries 1 to 15 are deactivated.

88.88.11.33 Request to retransmit mux table entries

A RequestMultiplexEntry may be sent at any time to elicit retransmission of specified multiplex table entries from the remote terminal, for example, following an interruption or other cause for uncertainty.

A terminal that receives RequestMultiplexEntry shall send RequestMultiplexEntryResponse indicating the action it intends to take.

88.88.22 Communication between the MTSE and MTSE user

88.88.22.11 Primitives between MTSE and MTSE user

Communication between the MTSE, and MTSE user, is performed using the primitives shown in Table 4545.

TABLE 4545/H.245

Primitives and parameters

generic name	type			
	request	indication	response	confirm
TRANSFER	MUX-DESCRIPTOR	MUX-DESCRIPTOR	- [1]	-
REJECT	CAUSE	SOURCE CAUSE	not defined [2]	not defined

Notes:

1. "-" means no parameters
2. "not defined" means that this primitive is not defined.

88.88.22.22 Primitive definition

The definition of these primitives is as follows:

a) The TRANSFER primitives are used to transfer multiplex table entries.

b) The REJECT primitives are used to reject a multiplex table entry, and to terminate a multiplex table entry transfer.

88.88.22.33 Parameter definition

The definition of the primitive parameters shown in Table 4545 are as follows:

a) The MUX-DESCRIPTOR parameter is a multiplex table entry. It is carried transparently from the MTSE user at the out-going MTSE to the MTSE user at the in-coming MTSE. There may be multiple MUX-DESCRIPTORs associated with the TRANSFER primitive.

b) The SOURCE parameter indicates the source of the REJECT.indication primitive. The SOURCE parameter has the value of "USER" or "PROTOCOL". The latter case may occur as the result of a timer expiry.

c) The CAUSE parameter indicates the reason for rejection of a multiplex table entry. The CAUSE parameter is not present when the SOURCE parameter indicates "PROTOCOL".

88.88.22.44 MTSE states

The following states are used to specify the allowed sequence of primitives between the MTSE and the MTSE user. The states are specified separately for each of an out-going MTSE and an in-coming MTSE. The states for an out-going MTSE are:

State 0: IDLE

There is no MTSE transfer is in progress. The multiplex table entry may be used by the transmitter.

State 1: AWAITING RESPONSE

The MTSE user has requested the transfer of a multiplex table entry, and a response from the peer MTSE is awaited. The multiplex table entry shall not be used by the transmitter.

The states for an in-coming MTSE are:

State 0: IDLE

There is no MTSE transfer in progress. The multiplex table entry may be in use by the transmitter.

State 1: AWAITING RESPONSE

The peer MTSE has transferred a multiplex table entry, and a response from the MTSE user is awaited. The multiplex table entry may not be in use by the transmitter.

88.88.22.55 State transition diagram

The allowed sequence of primitives between the MTSE and the MTSE user is defined here. The allowed sequences are specified separately for each of an out-going MTSE and an in-coming MTSE, as shown in Figure 2424 and Figure 2525 respectively.

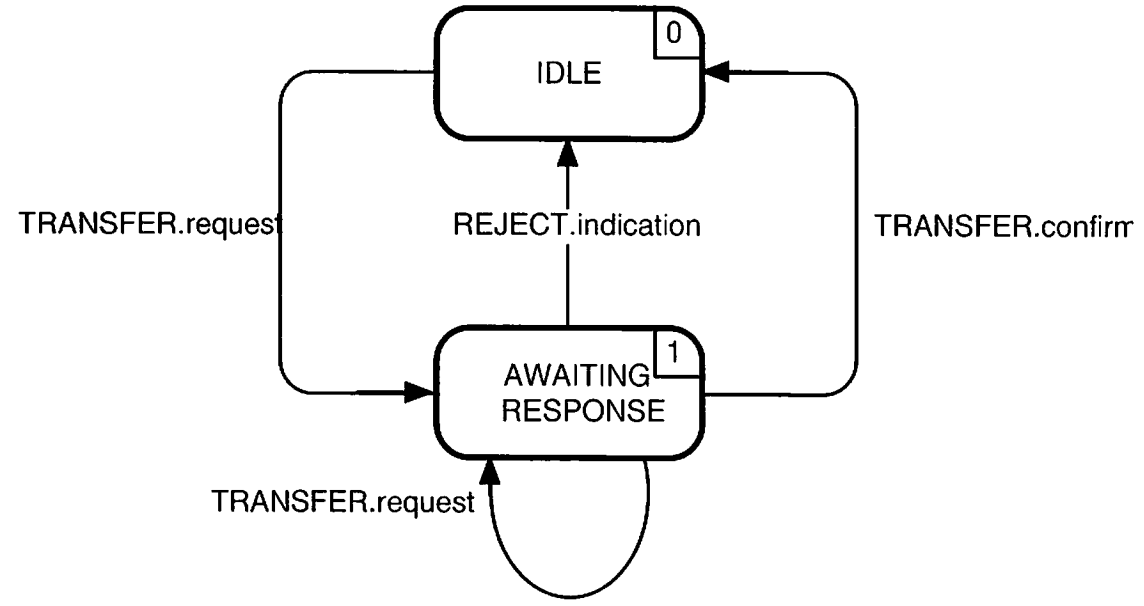

FIGURE 2424/H.245

State transition diagram for sequence of primitives at out-going MTSE

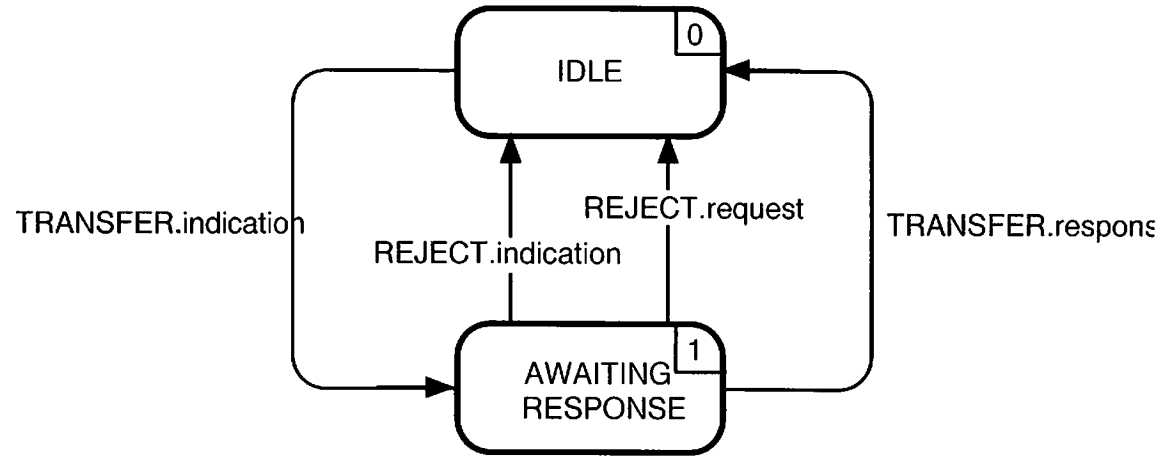

FIGURE 2525/H.245

State transition diagram for sequence of primitives at in-coming MTSE

88.88.33 Peer to peer MTSE communication

88.88.33.11 Messages

Table 4646 shows the MTSE messages and fields, defined in section 6, which are relevant to the MTSE protocol.

TABLE 4646/H.245

MTSE message names and fields

function	message	direction	SDL name [2]	field
transfer	MultiplexEntrySend	O -> I [2]	SEND	sequenceNumber multiplexTableEntryNumber MultiplexElement
	MultiplexEntrySendAck	O <- I	ACK	sequenceNumber multiplexTableEntryNumber
reject	MultiplexEntrySendReject	O <- I	REJ	sequenceNumber multiplexTableEntryNumber cause
reset	releaseMultiplexEntrySend	O -> I	RELEASE	multiplexTableEntryNumber

Notes:

1. Direction: O - out-going, I - in-coming.
2. The SDL name is a short hand notation used to indicate the message in the SDL figures.

88.88.33.22 MTSE state variables

The following state variables are defined at the out-going MTSE:

out_ENUM

This state variable distinguishes between out-going MTSEs. It is initialised at out-going MTSE initialisation. The value of out_ENUM is used to set the multiplexTableEntryNumber field of MTSE messages sent from an out-going MTSE. For MTSE messages received at an out-going MTSE, the message multiplexTableEntryNumber field value is identical to the value of out_ENUM.

out_SQ

This state variable is used to indicate the most recently sent SEND message. It is incremented by one and mapped to the SEND message sequenceNumber field before transmission of a SEND message. Arithmetic performed on out_SQ is modulo 256.

The following state variables are defined at the in-coming MTSE:

in_ENUM

This state variable distinguishes between in-coming MTSEs. It is initialised at in-coming MTSE initialisation. The value of in_ENUM is used to set the multiplexTableEntryNumber field of MTSE messages sent from an in-coming MTSE. For MTSE messages received at an in-coming MTSE, the message multiplexTableEntryNumber field value is identical to the value of in_ENUM.

in_SQ

This state variable is used to store the value of the sequenceNumber field of the most recently received SEND message. The ACK and REJ messages have their sequenceNumber fields set to the value of in_SQ, before being sent to the peer MTSE.

88.88.33.33 MTSE timers

The following timer is specified for the out-going MTSE:

T104

This timer is used during the AWAITING RESPONSE state. It specifies the time before an error message is generated, during which no ACK or REJ message has been received.

88.88.44 MTSE procedures

88.88.44.11 Introduction

Figure 2626 summarises the primitives and their parameters, and the messages and relevant fields, for each of the outgoing and in-coming MTSE.

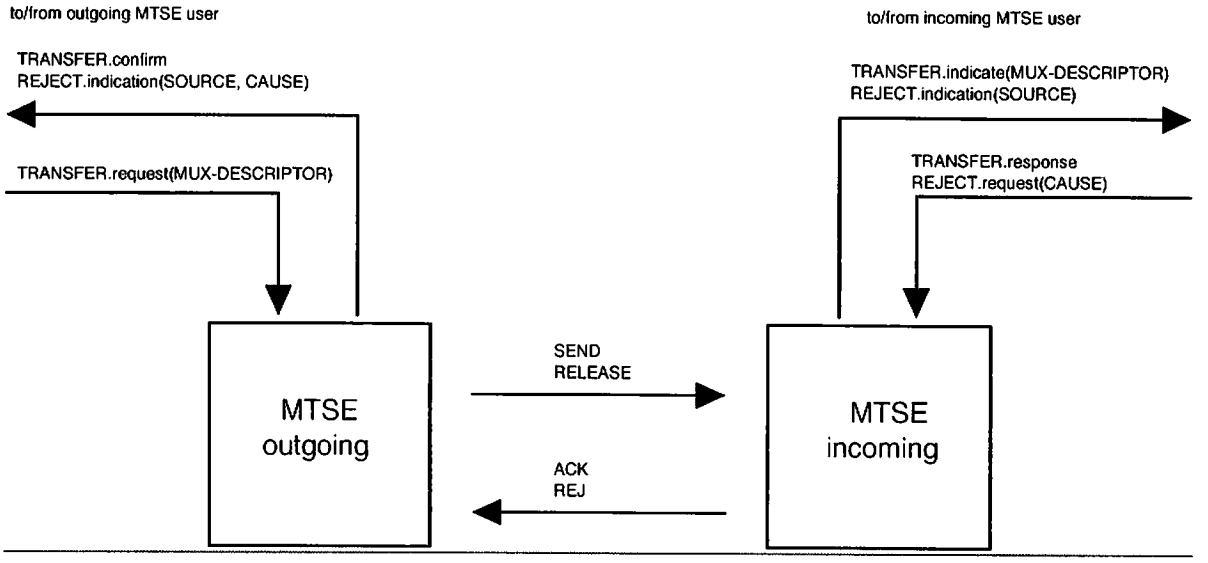

FIGURE 2626/H.245

Primitives and messages in the Multiplex Table Signalling Entity

88.88.44.22 Primitive parameter default values

Where not explicitly stated in the SDL diagrams the parameters of the indication and confirm primitives assume values as shown in Table 4747.

TABLE 4747/H.245

Default primitive parameter values

primitive	parameter	default value
TRANSFER.indication	MUX-DESCRIPTOR	SEND.MultiplexElement
REJECT.indication	SOURCE	USER
	CAUSE	null

88.88.44.33 Message field default values

Where not explicitly stated in the SDL diagrams the message fields assume values as shown in Table 4848.

TABLE 4848/H.245

Default message field values

message	field	default value [1]
SEND	sequenceNumber	out_SQ
	multiplexTableEntryNumber	out_ENUM
	MultiplexElement	TRANSFER.request(MUX-DESCRIPTOR)
ACK	sequenceNumber	in_SQ
	multiplexTableEntryNumber	in_ENUM
REJ	sequenceNumber	in_SQ
	multiplexTableEntryNumber	in_ENUM
	cause	REJECT.request(CAUSE)
RELEASE	multiplexTableEntryNumber	out_ENUM

Notes:

1. A message field shall not be coded, if the corresponding primitive parameter is null i.e. not present.

88.88.44.44 SDLs

The out-going MTSE and the in-coming MTSE procedures have the same SDLs as for the CESE, shown in Figure 88 and Figure 99 respectively. In the case of the MTSE, timer Tn in Figure 88 and Figure 99 indicates timer T104.

88.99 Mode Request procedures

88.99.11 Introduction

The procedures described here allow a terminal to request a remote terminal to use a particular mode of operation in its transmit direction. The procedures are referred to here as the Mode Request Signalling Entity (MRSE). Procedures are specified in terms of primitives and states at the interface between the MRSE and the MRSE user. Protocol information is transferred to the peer MRSE via relevant messages defined in section 6. There is an out-going MRSE and an in-coming MRSE. At each of the out-going and in-coming ends there is one instance of the MRSE per call.

If the currently valid capabilities received from the remote terminal contain one or more transmission capabilities, a terminal may select a mode that it prefers to have transmitted to it by performing the Mode Request procedures. A terminal whose currently valid capabilities contain one or more transmission capabilities and which is in receipt of such a request, should comply with the request.

A mode request shall not be sent to a terminal whose currently valid capabilities contain no transmission capabilities, that is, the terminal does not wish to, and shall not, be remotely controlled. If such a terminal does however receive a mode request, it may comply.

A terminal that receives RequestMode shall send RequestModeAck or RequestModeReject to indicate its intentions.

The requested mode may include channels which are already open. For example, if a channel for G.723 was currently open and a terminal wished to receive an additional G.728 channel, it would send a mode request containing both the G.723 and the G.728 channel. If the G.723 channel request were absent, this would indicate that G.723 was no longer desired.

Where one source is feeding several receivers it may be unable to respond to any received signals such as requests to transmit in a particular mode.

88.99.22 Communication between MRSE and MRSE user

88.99.22.11 Primitives between MRSE and MRSE user

Communication between the MRSE and MRSE user, is performed using the primitives shown in Table 4949.

TABLE 4949/H.245

Primitives and parameters

generic name	type			
	request	indication	response	confirm
TRANSFER	MODE-ELEMENT	MODE-ELEMENT	MODE-PREF	MODE-PREF
REJECT	CAUSE	SOURCE CAUSE	not defined [1]	not defined

Notes:

1. "not defined" means that this primitive is not defined.

88.99.22.22 Primitive definition

The definition of these primitives is as follows:

a) The TRANSFER primitives are used for the transfer of the mode request.

b) The REJECT primitives are used to reject a mode request.

88.99.22.33 Parameter definition

The definition of the primitive parameters shown in Table 4949 are as follows:

a) The MODE-ELEMENT parameter specifies a mode element. This parameter is mandatory. There may be multiple MODE-ELEMENTS associated with the TRANSFER primitives.

b) The MODE-PREF parameter informs the user as to whether the most preferred mode requested will be used or not. It is carried transparently from the in-coming RMSE user to the out-going RMSE user. It has two values being "MOST-PREFERRED" and "LESS-PREFERRED".

c) The SOURCE parameter indicates the source of the REJECT.indication primitive. The SOURCE parameter has the value of "USER" or "PROTOCOL". The latter case may occur as the result of a timer expiry.

d) The CAUSE parameter indicates the reason for refusal to close a logical channel. The CAUSE parameter is not present when the SOURCE parameter indicates "PROTOCOL".

8.99.22.44 MRSE states

The following states are used to specify the allowed sequence of primitives between the MRSE and the MRSE user. The states for an out-going MRSE are:

State 0: IDLE

The MRSE is idle.

State 1: AWAITING RESPONSE

The MRSE is waiting for a response from the remote MRSE.

The states for an in-coming MRSE are:

State 0: IDLE

The MRSE is idle.

State 1: AWAITING RESPONSE

The MRSE is waiting for a response from the MRSE user.

8.99.22.55 State transition diagram

The allowed sequence of primitives between the MRSE and the MRSE user is defined here. The allowed sequences are specified separately for each of an out-going MRSE and an in-coming MRSE, as shown in Figure 27~~27~~ and Figure 28~~28~~ respectively.

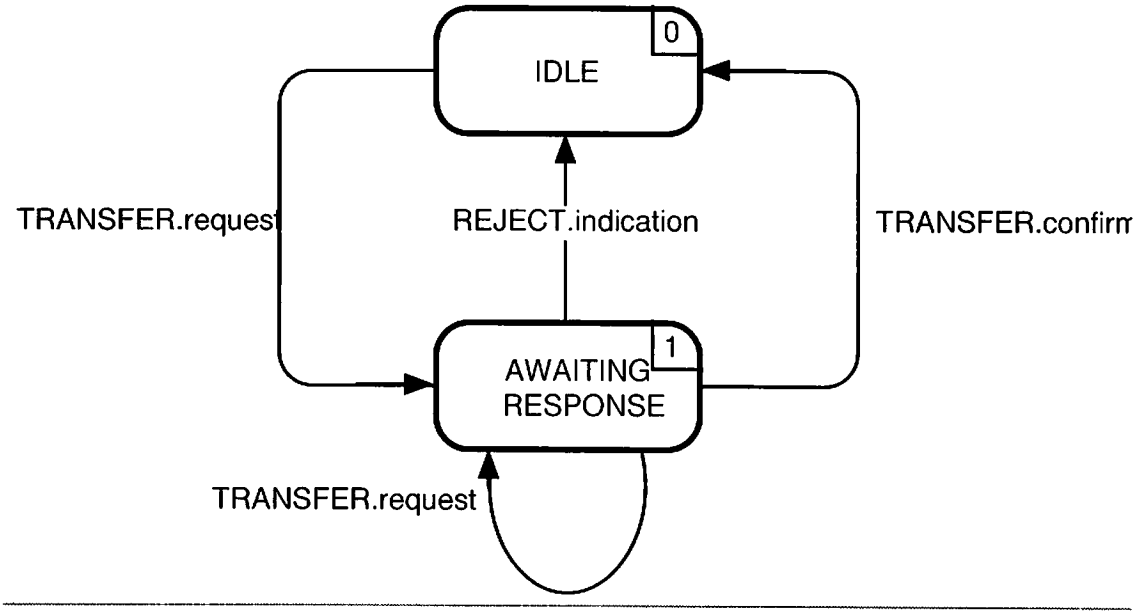

FIGURE 27~~27~~/H.245

State transition diagram for sequence of primitives at MRSE out-going

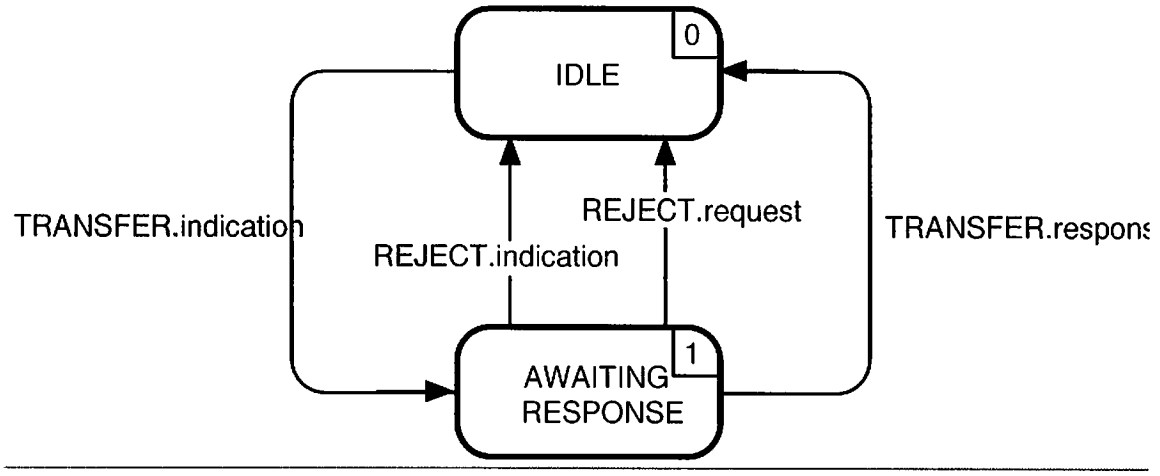

FIGURE 2828/H.245

State transition diagram for sequence of primitives at MRSE in-coming

88.99.33 Peer to peer MRSE communication

88.99.33.11 Messages

Table 5050 shows the MRSE messages and fields, defined in section 6, which are relevant to the MRSE protocol.

TABLE 5050/H.245

MRSE message names and fields

function	message	direction	SDL name [2]	field
mode request	RequestMode	O -> I [1]	SEND	sequenceNumber requestedModes
	RequestModeAck	O <- I	ACK	sequenceNumber response
	RequestModeReject	O <- I	REJ	sequenceNumber cause
reset	RequestModeRelease	O -> I	RELEASE	-

Notes:

1. Direction: O - out-going, I - in-coming.
2. The SDL name is a short hand notation used to indicate the message in the SDL figures.

88.99.33.22 MRSE state variables

The following state variables are defined at the out-going MRSE:

out_SQ

This state variable is used to indicate the most recent SEND message. It is incremented by one and mapped to the SEND message sequenceNumber field before transmission of the SEND message. Arithmetic performed on out_SQ is modulo 256.

The following state variables are defined at the in-coming MRSE:

in_SQ

This state variable is used to store the value of the sequenceNumber field of the most recently received SEND message. The ACK and REJ messages have their sequenceNumber fields set to the value of in_SQ, before being sent to the peer MRSE.

88.99.33.33 MRSE timers

The following timer is specified for the out-going MRSE:

T109

This timer is used during the AWAITING RESPONSE state. It specifies the time before an error message is generated, during which no ACK or REJ message has been received.

88.99.44 MRSE procedures

Figure 2929 summarises the MRSE primitives and their parameters, and messages, for each of the out-going and in-coming MRSE.

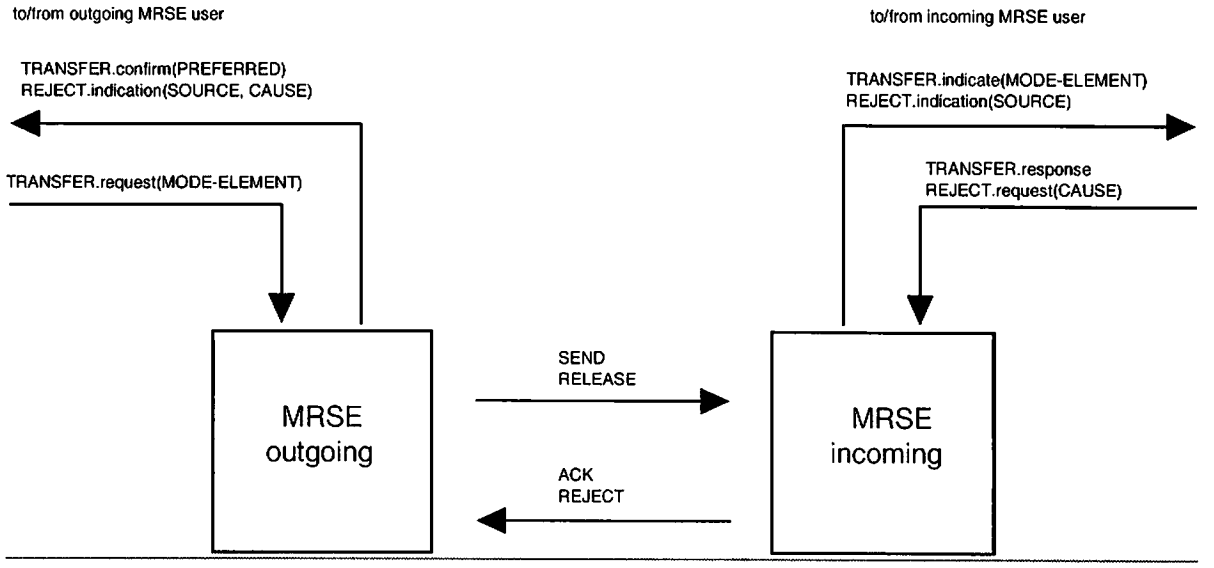

FIGURE 2929/H.245

Primitives and messages in the Mode Request Signalling Entity

88.99.44.11 Primitive parameter default values

Where not explicitly stated in the SDL diagrams the parameters of the indication and confirm primitives assume values as shown in Table 5151.

TABLE 5151/H.245

Default primitive parameter values

primitive	parameter	default value
TRANSFER.indication	MODE-ELEMENT	SEND.requestedModes
TRANSFER.confirm	MODE-PREF	ACK.response
REJECT.indication	SOURCE	USER
	CAUSE	null

88.99.44.22 Message field default values

Where not explicitly stated in the SDL diagrams the message fields assume values as shown in Table 5252.

TABLE 5252/H.245

Default message field values

message	field	default value
SEND	sequenceNumber	out_SQ
	requestedModes	TRANSFER.request(MODE-ELEMENT)
ACK	sequenceNumber	in_SQ
	response	TRANSFER.response(MODE-PREF)
REJ	sequenceNumber	in_SQ
	cause	REJECT.request(CAUSE)
RELEASE	-	-

88.99.44.33 SDLs

The out-going MRSE and the in-coming MRSE procedures have the same SDLs as for the CESE, which are shown in Figure 88 and Figure 99 respectively. In the case of the MRSE, timer Tn in Figure 88 and Figure 99 indicates timer T109.

88.1040 Round trip delay procedures

88.1040.11 Introduction

Procedures are described here that allow the determination of the round trip delay between two communicating terminals. This function also enables a H.245 user to determine if the peer H.245 protocol entity is still alive.

The function described here is referred to as the Round Trip Delay Signalling Entity (RTDSE). Procedures are specified in terms of primitives and states at the interface between the RTDSE and the RTDSE user. There is one instance of the RTDSE in each of the master and slave terminals. Both master and slave terminals may perform the round trip delay determination.

88.1040.22 Communication between the RTDSE and the RTDSE user

88.1040.22.11 Primitives between the RTDSE and the RTDSE user

Communication between the RTDSE, and RTDSE user, is performed using the primitives shown in Table 5353. These primitives are for the purpose of defining RTDSE procedures and are not meant to specify or constrain implementation.

TABLE 5353/H.245

Primitives and parameters

generic name	type			
	request	indication	response	confirm
TRANSFER	- [1]	not defined [2]	not defined	DELAY
EXPIRY	not defined	-	not defined	not defined

Notes:

1. "-" means no parameters
2. "not defined" means that this primitive is not defined.

88.1040.22.22 Primitive definition

The definition of these primitives is as follows:

a) The TRANSFER primitive is used to request, and report upon, the round trip delay determination.

b) The EXPIRY primitive indicates that no response has been received from the peer terminal.

88.1040.22.33 Parameter definition

The definition of the primitive parameters shown in Table 5353 are as follows:

a) The DELAY parameter returns the measured round trip delay.

88.1040.22.44 RTDSE states

The following states are used to specify the allowed sequence of primitives between the RTDSE and the RTDSE user.

State 0: IDLE

There is no RTDSE transfer in progress.

State 1: AWAITING RESPONSE

The RTDSE user has requested the measurement of the round trip delay. A response from the peer RTDSE is awaited.

88.1040.22.55 State transition diagram

The allowed sequence of primitives between the RTDSE and the RTDSE user is defined here. The allowed sequences are shown in Figure 3030.

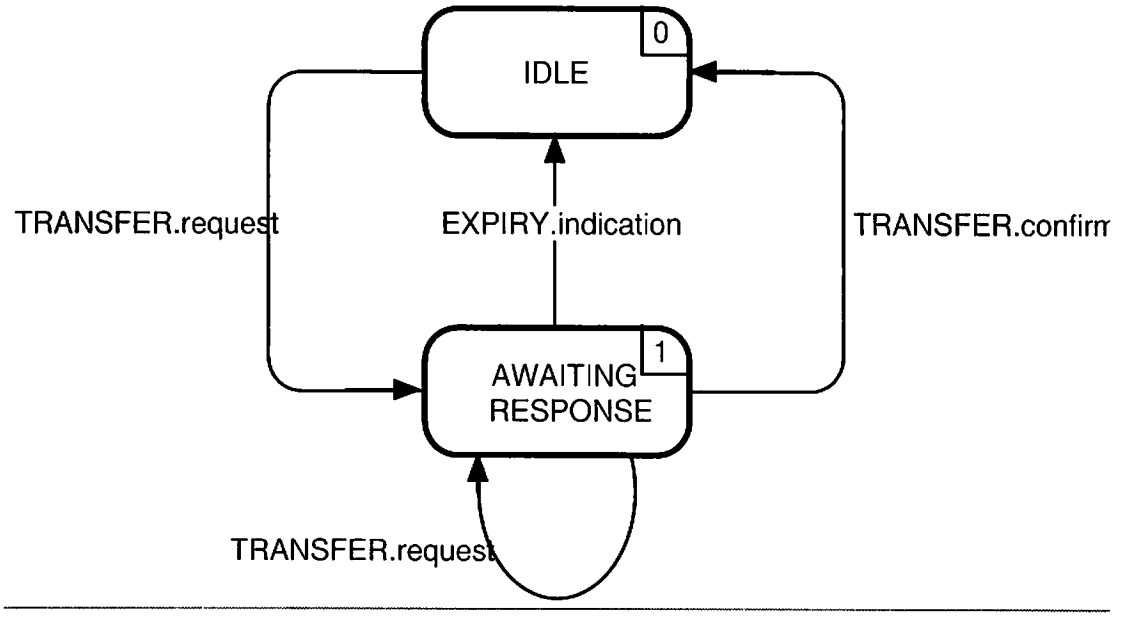

FIGURE 3030/H.245

State transition diagram for sequence of primitives at RTDSE

88.1010.33 Peer to peer RTDSE communication

88.1010.33.11 Protocol Data Units

Table 5454 shows the RTDSE PDUs and fields, defined in section 6, which are relevant to the RTDSE protocol.

TABLE 5454/H.245

RTDSE PDU names and fields

function	PDU	direction	SDL name [2]	field
transfer	RoundTripDelayRequest	O -> I [1]	RTDREQ	sequenceNumber
	RoundTripDelayResponse	O <- I	RTDACK	sequenceNumber

Notes:

1. Direction: O - out-going, I - in-coming.
2. The SDL name is a short hand notation used to indicate the message in the SDL figures.

88.1010.33.22 RTDSE state variables

The following RTDSE state variables are defined:

out_SQ

This state variable is used to indicate the most recent RTDREQ PDU. It is incremented by one and mapped to the RTDREQ PDU sequenceNumber field before transmission of an RTDREQ PDU. Arithmetic performed on out_SQ is modulo 256.

88.1010.33.33 RTDSE timers

The following timer is specified for the out-going RTDSE:

T105

This timer is used during the AWAITING RESPONSE state. It specifies the time before an error message is generated, during which no RTDACK has been received.

88.1010.44 RTDSE procedures

88.1010.44.11 Introduction

Figure 3131 summarises the RTDSE primitives and their parameters, and PDUs.

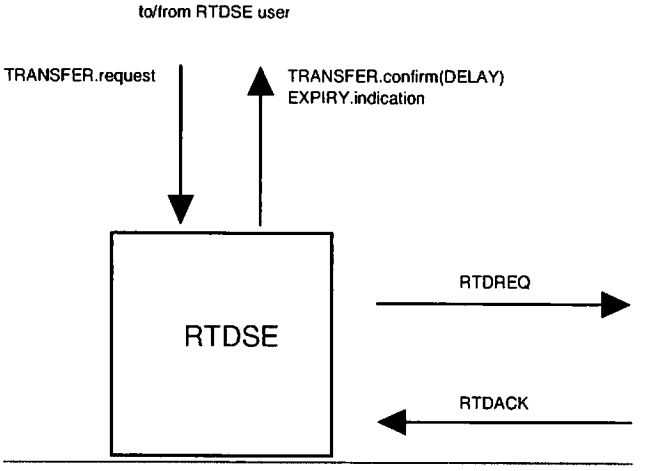

FIGURE 3131/H.245

Primitives and PDUs in the RTDSE.

88.1010.44.22 Primitive parameter default values

Where not explicitly stated in the SDL diagrams the parameters of the indication and confirm primitives assume values as shown in Table 5555.

TABLE 5555/H.245

Default primitive parameter values

primitive	parameter	default value
TRANSFER.confirm	DELAY	value of timer T105
EXPIRY.indication	-	-

88.1010.44.33 PDU field default values

Where not explicitly stated in the SDL diagrams the PDU fields assume values as shown in Table 5656.

TABLE 5656/H.245

Default PDU field values

PDU	field	default value
RTDREQ	sequenceNumber	out_SQ
RTDACK	sequenceNumber	RTDREQ.sequenceNumber

88.1040.44.44 SDLs

The RTDSE procedures are expressed in SDL form in Figure 3232.

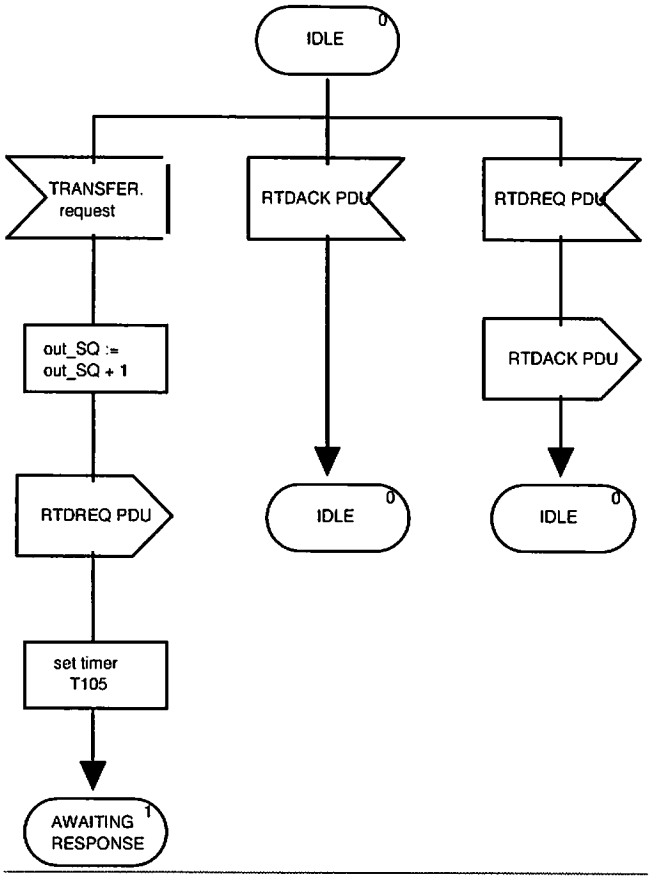

FIGURE 3232(i)/H.245

RTDSE SDL

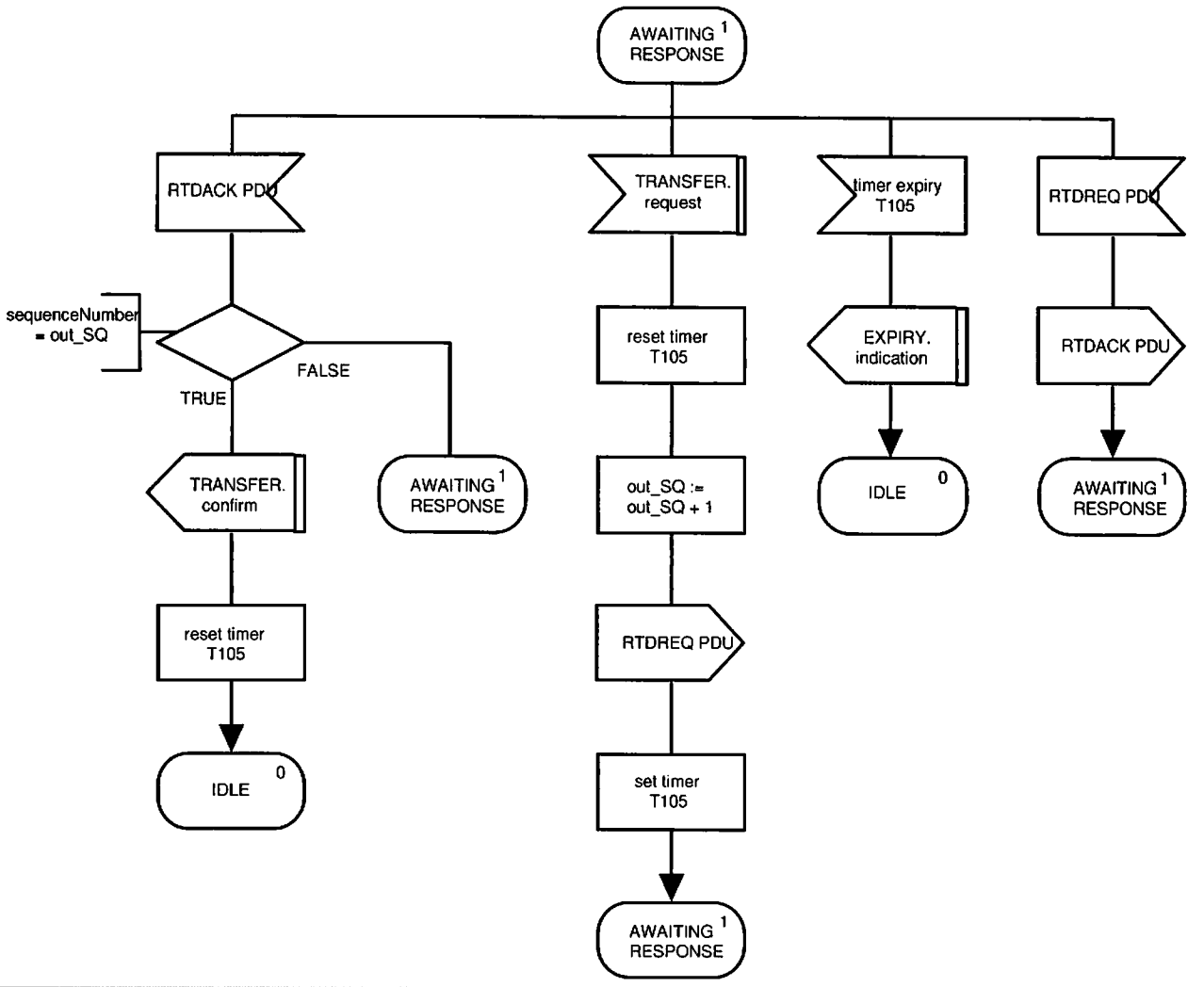

FIGURE 3232(ii)/H.245

RTDSE SDL (concluded)

ANNEX A

OBJECT IDENTIFIER ASSIGNMENTS

(This annex forms an integral part of this Recommendation)

Table A-1 lists the assignment of Object Identifiers defined for use by this Recommendation.

Table A-1

Object Identifier Value	Description
{itu recommendation h 245 version(0) 1}	This Object Identifier is used to indicate the version of this Recommendation in use as a multimedia system control protocol. At this time there is a single standardised version defined.

APPENDIX I

(This appendix does not form an integral part of this Recommendation)

Overview of ASN.1 syntax

I.1~~1~~ Introduction to ASN.1

Abstract Syntax Notation One (ASN.1) is a data specification language. It was originally standardized as part of the X.400 electronic mail series as X.409. This evolved to X.208 and most recently X.680. ASN.1 allows unambiguous specification of complex data structures including those with variable-length fields, optional fields and recursion.

The above Recommendations deal only with the syntax and semantics of ASN.1 specifications. The binary encoding of data structures is covered in other Recommendations, notably X.690 (basic encoding rules or BER) and X.691 (packed encoding rules or PER). BER allows data to be deciphered by systems that have general knowledge of ASN.1 but do not know the details of the specification used to form the data. In other words, the data types are encoded along with the data values. PER is much more efficient since only data values are encoded and the coding is designed with very little redundancy. This method can be used when both the transmitter and the receiver expect data to adhere to a known structure.

H.245 is implemented using the packed encoding rules. Since both sides of a call know that messages will conform to the H.245 specification it is not necessary to encode that specification into the messages. For decoding simplicity, the aligned variant of PER is used. This forces fields that require eight or more bits to be aligned on octet boundaries and to consume an integral number of octets. Alignment is done by padding the data with zeros before large fields.

I.2~~2~~ Basic ASN.1 data types

The simplest data type is BOOLEAN, which represents the values FALSE and TRUE. These are encoded in a single bit as 0 and 1 respectively. For example, **segmentableFlag BOOLEAN** is coded

Value	Encoding
FALSE	0
TRUE	1

The most fundamental data type is INTEGER, which represents whole number values. Integers can be unconstrained as in:

bitRateINTEGER

or they can be constrained to a range of values, for example:

maximumAl2SDUSize INTEGER (0..65535)

Constrained integers are encoded differently depending on the size of the range. Suppose N is the number of integers in the range, i.e. the upper limit minus the lower limit plus one. Depending on N, the constrained integer will be encoded in one of five ways:

N	Encoding
1	no bits needed
2-255	an unaligned field of 1 to 8 bits
256	an aligned 8-bit field
257-65536	an aligned 16-bit field
larger	as the minimum number of aligned octets preceded by the above encoding of the number of octets

In all cases, the number that is actually used is the value to be encoded minus the lower limit of the range. In these examples "pad" represents zero to seven 0 bits that are added to the encoding so that the following field will start on a 8-bit boundary.

firstGOB **INTEGER (0..17)**

Value	Encoding
0	00000
3	00011

h233IVResponseTime **INTEGER (0..255)**

Value	Encoding
3	pad 00000011
254	pad 11111110

skew **INTEGER (0..4095)**

Value	Encoding
3	pad 00000000 00000011
4095	pad 00001111 11111111

Unconstrained (2's complement) integer values that can be represented in 127 octets or less are encoded in the minimum number of octets needed. The number of octets (the length) is encoded as an aligned octet that precedes the number itself. For example,

-1	pad 00000001 11111111
0	pad 00000001 00000000
128	pad 00000010 00000000 10000000
1000000	pad 00000011 00001111 01000010 01000000

ASN.1 supports a variety of string data types. These are variable-length lists of bits, octets or other short data types. They are typically encoded as a length followed by the data. The length can be encoded as an unconstrained integer or as a constrained integer if the SIZE of the string is specified. For example,

data **OCTET STRING**

Since the length of the octet string is not bounded, it will have to be encoded as a *semi-constrained whole number* (has a lower bound, but no upper bound). First, the data is padded so that the encoding will be aligned. The rest of the code is as follows:

Length	Encoding
0 to 127	8-bit length followed by the data
128 to 16K-1	16-bit length with the MSB set, then the data
16K to 32K-1	11000001, 16K octets of data, then code the rest
32K to 48K-1	11000010, 32K octets of data, then code the rest
48K to 64K-1	11000011, 48K octets of data, then code the rest
64K or more	11000100, 64K octets of data, then code the rest

This method is called "fragmentation". Note that if the length is a multiple of 16K, then the representation will end with an octet of zero indicating a zero-length string.

I.33 Aggregate data types

ASN.1 includes several aggregate or container data types that are similar in concept to C's union, struct and array types. These are, respectively, CHOICE, SEQUENCE and SEQUENCE OF. In all cases they are encoded with some bits specific to the container followed by the normal encodings of the contents.

CHOICE is used to select exactly one of a group of data types. For example,

```
VideoCapability                 ::= CHOICE
{
    nonStandard                 NonStandardParameter,
    h261VideoCap                H261VideoCapability,
    h262VideoCap                H262VideoCapability,
    h263VideoCap                H263VideoCapability,
    ...
}
```

An index number is assigned to each choice, starting with zero. The index of the actual choice is encoded as a constrained integer. The index is followed by the encoding of the actual selection or nothing if the selection is **NULL**. If the extension marker is present (as above), the index is preceded by a bit that is zero if the actual choice is from the original list.

SEQUENCE is simply a grouping of dissimilar data types. Individual elements of the sequence may be OPTIONAL. The encoding is very simple. If there is an extension marker the first bit indicates the presence of additional elements. This is followed by a series of bits, one for each optional element that indicates if that data is present. This is followed by the encodings of the components of the sequence. For example,

```
H261VideoCapability             ::= SEQUENCE
{
    qcifMPI                     INTEGER (1..4) OPTIONAL, -- units 1/29.97 Hz
    cifMPI                      INTEGER (1..4) OPTIONAL,        -- units 1/29.97 Hz
    temporalSpatialTradeOffCapability   BOOLEAN,
    ...
}
```

The encoding has one bit for the extension marker, two bits for the optional fields, two bits each for any optional field that is present, one bit for the boolean and then any extension data. Note that in this sequence has no padding for octet alignment.

The SEQUENCE OF and SET OF types describe a collection of similar components (an array). SEQUENCE OF implies that the order of the elements is significant, with SET OF the element order is arbitrary. The PER encoding is the same for both types.

These types can have a SIZE constraint or an unconstrained number of elements. If the number is known a priori and is less than 64K, it is not encoded. Otherwise the actual number of components is encoded as a constrained or semi-constrained length. This is followed by the encoding of the data. If the length is at least 16K and is encoded then the list of data will be broken into fragments like the octet string. In this case the fragments are broken after some number of component fields (16K, 32K, etc.), not after some number of octets.

I.4 Object Identifier type

Normally the type of a value is given in the ASN.1 specification so that the only information that needs to be coded and transmitted is the data itself. Occasionally, however, it is desirable to encode the data type as well as the data value. For example, **NonStandardIdentifier** contains

object **OBJECT IDENTIFIER**

This is encoded as the data encoded with the BER (X.690) preceded by the length of that encoding in octets. The length is encoded as a semi-constrained whole number (see the OCTET STRING example above).

APPENDIX II

(This appendix does not form an integral part of this Recommendation)

Example Multiplex Table Descriptors

The table below shows an example of MultiplexEntryDescriptors which could be used in the multiplex table. The nesting depth, elementList size, and maximum subElementList size of each is given for illustration. Note that entry zero is not transmitted, but is shown here for clarity. In the example, four logical channels are being transmitted (plus LCN 0 for control): LCN 1 is data, LCN 2 is audio I, LCN 3 is video, and LCN 4 is audio II.

The following abbreviations are used: LCN - logicalChannelNumber; RC - repeatCount; UCF - untilClosingFlag.