INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

H.248.1 Version 2 Implementors' Guide

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

(2 May 2008)

SERIES H: AUDIOVISUAL AND MULTIMEDIA SYSTEMS

Infrastructure of audiovisual services – Communication procedures

---

Implementors' Guide for Recommendation H.248.1 Version 2 (Media Gateway Control Protocol) and its Corrigendum 1 (03/2004)

Summary

This document is a compilation of reported defects identified in ITU-T Recommendation H.248.1 Version 2 (05/2002) subsequent to its Corrigendum 1 (01/2004). It must be read in conjunction with the Recommendation to serve as an additional authoritative source of information for implementors.

This revision contains all updates submitted up to and including those at Study Group 16 meeting in January 2008 and May 2008.

This document was approved by ITU-T Study Group 16 on 2 May 2007 (TD 521/Plen) and obsoletes the earlier version of this Implementors' Guide approved on 6 July 2007.

NOTE: Changes relative to H.248.1 Version 3 (05/2002) and onwards are found in another document.

Change Log

(All changes that were included in H.248.1 V2 (05/2002) Corrigendum 1 (03/2004) are omitted here.)

V7 (Seoul, January 2008)

Add new secetion

New:

6.47 ALL Wildcard in Topology Descriptor

6.48 Octet String usage in Annex B.2

6.49 H.248 Data type ServiceChangeMgcID

6.50 Text encoding of Octet Strings

6.51 Profile definition - Syntactical vs Semantical Profiles

6.52 Example Call Flow updates

6.53 Multiplexing and Maximum number of terminations per context property

6.54 Clarification of the behavior of sub-lists

6.55 Orthogonal capabilities “Transcoding” and “Codec Negotiation/Determination”

6.56 Signal Type Override

6.57 rtp/pltrans event clarification

6.58 The use of Ignore with SDP

6.59 Base packages and Extended Packages

6.60 Connection Endpoint Naming Conventions

V8 (Geneva, April 2008)

Modified

6.59 Base packages and Extended Packages

New:

6.61 Error in Context Audit example

6.62 Simple value as allowed as a sub-list of length one

6.63 Empty Statistics Descriptor

6.64 Incorrect RTP Profile ID

Contact Information

ITU-T Study Group 16 / Question 3 Rapporteur	Christian Groves Australia	Tel: +61 3 9301 6116 E-mail: Christian.Groves@nteczone.com
H.248 Sub-series Implementors' Guide Editor	Lanlan Li ZTE, P.R. China	Tel: +86 25 52877609 E-mail: li.lanlan@zte.com.cn

Table of Contents

Implementors’ Guide for
Recommendation H.248.1 Version 2

1 Scope

This guide resolves defects in the following categories:

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

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

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

2 Introduction

The H.248.1 Version 2 Implementors’ Guide is a compilation of reported defects for version 2 of Recommendation H.248.1 (05/2002) Corrigendum 1 (03/2004). This edition of the Guide contains reported defects identified as of 04/2008.

The Guide must be read in conjunction with Recommendation H.248.1 version 2 to serve as an additional source of information for implementors. For changes to version 3 or later of H.248.1 or for other Recommendations in the H.248.x sub-series, please reference the H.248 Sub-series Implementors’ Guide.

3 Defect Resolution Procedure

Upon discovering technical defects with Recommendation H.248.1 Version 2, please provide a written description directly to the editor with a copy to the Q.3/16 Rapporteur. The template for a defect report is located at the end of the Guide. Contact information for these parties is included at the front of the document. Return contact information should also be supplied so a dialogue can be established to resolve the matter and an appropriate reply to the defect report can be conveyed. This defect resolution process is open to any interested party. Formal membership in the ITU is not required to participate in this process.

4 References

This document refers to the following Recommendation:

- ITU-T Recommendation H.248.1 Version 2 (05/2002) Corr.1 (03/2004), *Gateway control protocol: Version 2*

5 Nomenclature

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

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

6 Technical and Editorial Corrections to H.248.1 (05/2002) Corr.1 (03/2004)

6.1 Error response when processing a ContextID

Description:	H.248.1 § 8.2.2 discusses the action to be taken when a receiver encounters an error parsing a ContextID. It specifies that the ContextID is returned in an Action response with error code 422 "Syntax Error in action". It is very likely however that if a receiver cannot parse the context ID it will not be able to return the ContextID back to the sender. It is proposed below to allow sending of a response at a transaction level without including the ContextID.
Reference:	AVD-2467

[Begin Correction]

8.2.2 TransactionReply

…

If the receiver encounters an error in processing a ContextID but can parse the ContextID, the requested Action response will consist of the Context ID and a single error descriptor, 422 (Syntax Error in Action). If the receiver cannot parse the ContextID, it shall return a TransactionReply consisting of the TransactionID and a single error descriptor, 422 (Syntax Error in Action).

…

[End Correction]

6.2 Support of packages

Description:	There has been some confusion of what is meant by the term "To support a particular package the MG must support all properties, signals, events and statistics defined in a package. It must also support all Signal and Event parameters." The term "Must Support" is different from "it is mandatory to implement all the functionality in the package." This term was added so that the MG had to support the reception of properties, signal, events and statistics. The MG must not return a syntax error or unknown ID for any of these elements. However the MG could give a meaningful response such as 501 not implemented. This shows that the MG has at least considered the complete package. It is encouraged to support the complete functionality of a package however it is better to return error code 501 and using an existing package rather than creating new packages with the same functionality as contained in existing packages. It is proposed to clarify this.
Reference:	Subject: RE: [Megaco] Support of a H.248 package Date: Tue, 23 Mar 2004 14:39:43 -0500 From: Kevin Boyle <kboyle@nortelnetworks.com> and AVD-2467

[Begin Correction]

6.2.3 Packages

…

To support a particular package the MG must ~~support~~ recognize all properties, signals, events and statistics defined in a package. It must also support all Signal and Event parameters. If the functionality behind these properties, signals, events and statistics is not implemented, the MG shall not return a syntax error or unknown ID error for any of these elements but should return error 501 "Not implemented".

The MG may support a subset of the values listed in a package for a particular Property or Parameter. If an unsupported value is specified by the MGC, the MG shall return error 501 "Not implemented".

…

[End Correction]

6.3 Mismatch between RFC2377 support and one "m=" line restriction

Description:	H.248.1 § 7.1.8 Local and Remote descriptor mandates that there shall only be one m= line per SDP. However further in the section it states that "Implementations shall accept session descriptions that are fully conformant to RFC 2327." There could be some confusion over whether or not multiple m=lines should be accepted if they conform to RFC2327. Multiple m=lines per SDP should not be accepted by the MG. This should be clarified.
Reference:	AVD-2467

[Begin Correction]

6.3.1 7.1.8 Local and Remote descriptors

…

A Stream Descriptor specifies a single bidirectional media stream and so a single session description MUST NOT include more than one media description ("m = " line). A Stream Descriptor may contain additional session descriptions as alternatives. Each media stream for a termination must appear in distinct Stream Descriptors. When multiple session descriptions are provided in one descriptor, the "v = " lines are required as delimiters; otherwise they are optional in session descriptions sent to the MG. Implementations shall accept session descriptions that are fully conformant to RFC 2327 according to the above restrictions. When binary encoding the protocol, the descriptor consists of groups of properties (tag-value pairs) as specified in Annex C. Each such group may contain the parameters of a session description.

…

[End Correction]

6.4 Annex C codepoints for RTCP

Description:	H.248.30 introduces a package for the support of RTCP XR. As part of the functioning of RTCP SDP bandwidth modifiers may be sent. This is currently supported in SDP but not in the Binary version of H.248.1. Thus the bandwidth modifiers RTCP(b=RS:xx, b=RR:xx) as defined in RFC3556 need to be added to H.248.1 Annex C.
Reference:	Subject: [Megaco] RTCP bandwidth modifiers Date: Thursday, March 25, 2004 5:41 AM From: Kalleitner Franz [mailto:franz.kalleitner@siemens.com]

[Begin Correction]

C.6 IP

PropertyID	Property tag	Type	Value
IPv4	6001	32 bits Ipv4Address	Ipv4Address Ref.: IETF RFC 791
IPv6	6002	128 bits	IPv6 Address Ref.: IETF RFC 2460
Port	6003	Unsigned integer	0..65535
Porttype	6004	Enumerated	TCP(0), UDP(1), SCTP(2)
RtcpbwRS	6005	Integer	RS RTCP bandwidth modifier indicates the RTCP bandwidth allocated to active data senders (as defined by the RTP spec) Ref.: IETF RFC 3556
RtcpbwRR	6006	Integer	RR RTCP bandwidth modifier indicates the RTCP bandwidth allocated to other participants in the RTP session (i.e., receivers) Ref.: IETF RFC 3556

[End Correction]

6.5 Clarification of PackageID and name for Annex C

Description:	The binary encoding of H.248.1 uses Annex C for a number of local and remote properties. The identities of the Annex C properties use the normal package name/property id construct. It is not clear from reading Annex C what the Package name is. The Annex A ASN.1 syntax must be read to find out that the binary package name is H'0000. No text package name has been defined for Annex C for use with the H.248 property SDP element. Annex C should be modified to make the package name/id clear.
Reference:	-------- Original Message -------- Subject: Re: Signaling of UDI rsp Clearmode Bearer Service; Re: TDM Hairpinning; Re: [Megaco] Hairpin case A/u Law conversion Date: Tue, 27 Apr 2004 13:55:12 +0200 From: Carsten Waitzmann <cwaitzmann@alcatel.de>

[Begin Correction]

6.5.1 Annex C Tags for media stream properties

Parameters for Local, Remote and LocalControl descriptors are specified as tag-value pairs if binary encoding is used for the protocol. This annex contains the property names (PropertyID), the tags (Property tag), type of the property (Type) and the values (Value). Values presented in the Value field when the field contains references shall be regarded as "information". The reference contains the normative values. If a value field does not contain a reference, then the values in that field can be considered as "normative".

The referencing of Annex C properties follows the PackageID/PropertyID structure; however Annex C is not in itself a package. Annex C is considered to have PackageID 0x0000 for binary encoding and "anxc" for text encoding. For text encoding of H.248.1, Annex C shall only be used in the case that the required property is not already defined by a package or represented by SDP. The nesting of one Annex C property inside another is forbidden.

Tags are given as hexadecimal numbers in this annex. When setting the value of a property, a MGC may underspecify the value according to one of the mechanisms specified in 7.1.1.

…

[End Correction]

6.6 Clarification of ReserveGroup and ReserveValue Properties

Description:	The use of the Reserve property is described in H.248.1 sections 7.1.7 and 7.1.8. Currently there is a mismatch between the stated behaviour of Reserve Group. Section 7.1.7 states: "If the value of a Reserve property is True, the MG SHALL reserve resources for all alternatives specified in the Local and/or Remote descriptors for which it currently has resources available." however section 7.1.8 states "If ReserveGroup is True, the MG reserves the resources required to support any of the requested property group alternatives that it can currently support. If ReserveValue is True, the MG reserves the resources required to support any of the requested property value alternatives that it can currently support." It is proposed to update 7.1.8 to align with the definition in 7.1.7.
Reference:	AVD-2569

[Begin Correction]

6.6.1 7.1.8 Local and remote descriptors

…

If ReserveGroup is True, the MG reserves the resources required to support ~~any~~ as many as possible of the requested property group alternatives that it can currently support. If ReserveValue is True, the MG reserves the resources required to support ~~any~~ as many as possible of the requested property value alternatives that it can currently support.

…

[End Correction]

6.7 Clarification of Provisional Response Timer Values

Description:	H.248.1 has 2 sections that discuss the setting of the (MG)provisionalResponseTimer value. Section D.1.4 states: "The root Termination has a property (ProvisionalResponseTimerValue), which can be set to the requested maximum number of milliseconds between receipt of a command and transmission of the TransactionPending response." Section E.2.1 states: "Initially set to normalMGExecutionTime plus network delay, but may be lowered."\ These two sections give conflicting advice of what to set the timer to, D.1.4 does not take into account network delay which may result in messages being repeated unnecessary. It is proposed to remove the text from D.1.4 and rely on the definition in E.2.1. Furthermore section 8.2.3 discusses procedures for the sending of Transaction Pending. It however has omitted the (MG/MGC)ProvisionalResponseTimerValue the use of which should be documented.
Reference:	AVD-2569

[Begin Correction]

6.7.1 8.2.3 TransactionPending

…

TransactionPending(TransactionID { })

The TransactionID parameter must be the same as that of the corresponding TransactionRequest. A property of root (normalMGExecutionTime) is settable by the MGC to indicate the interval within which the MGC expects a response to any transaction from the MG (exclusive of network delay). Another property (normalMGCExecutionTime) is settable by the MGC to indicate the interval within which the MG should expect a response to any transaction from the MGC (exclusive of network delay). MGProvisionalResponseTimerValue indicates the time within which the MGC should expect a Pending Response from the MG if a Transaction cannot be completed (initially set to normalMGExecutionTime plus network delay, but may be lowered). MGCProvisionalResponseTimerValue has the corresponding meaning to the MG. Senders may receive more than one TransactionPending for a command. If a duplicate request is received when pending, the responder may send a duplicate pending immediately, or continue waiting for its timer to trigger another TransactionPending.

…

6.7.2 D.1.4 Provisional responses

Executing some transactions may require a long time. Long execution times may interact with the timer-based retransmission procedure. This may result either in an inordinate number of retransmissions, or in timer values that become too long to be efficient. Entities that can predict that a transaction will require a long execution time may send a provisional response, "Transaction Pending". They SHOULD send this response if they receive a repetition of a transaction that is still being executed.

Entities that receive a Transaction Pending shall switch to a different repetition timer for repeating requests. ~~The root Termination has a property (ProvisionalResponseTimerValue), which can be set to the requested maximum number of milliseconds between receipt of a command and transmission of the TransactionPending response.~~ Upon receipt of a final response following receipt of provisional responses, an immediate confirmation shall be sent, and normal repetition timers shall be used thereafter. An entity that sends a provisional response, SHALL include the immAckRequired field in the ensuing final response, indicating that an immediate confirmation is expected. Receipt of a Transaction Pending after receipt of a reply shall be ignored.

…

[End Correction]

6.8 Clarification of NULL Context Usage

Description:	There has been some confusion over what terminations in the NULL context actually represent. H.248.1 § 6. Connection Model gives some guidance: “A Context is an association between a collection of Terminations. There is a special type of Context, the null Context, which contains all Terminations that are not associated to any other Termination. For instance, in a decomposed access gateway, all idle lines are represented by Terminations in the null Context.” Terminations in the NULL context typically have default descriptor values associated with them. For example: the termination may have an event detecting “off-hook”. Therefore according to the above text it could be stated that terminations with default descriptor values could be considered as IDLE lines. H.248.1 § 7.2.3 Subtract gives further guidance on what happens to property values when a termination is SUBTRACTED back to the NULL context: “When a provisioned Termination is Subtracted from a Context, its property values shall revert to: • the default value, if specified for the property and not overridden by provisioning; • otherwise, the provisioned value.” It is the contributor’s contention that the term “property values” is a generic term and what is really meant is “descriptor values”. Therefore the conclusion that when a termination is subtracted back to the NULL context this represents an IDLE line.
Reference:	AVD-2570

[Begin Correction]

6.8.1 6.2.4 Termination properties and descriptors

Terminations have properties. The properties have unique PropertyIDs. Most properties have default values, which are explicitly defined in this protocol specification or in a package (see clause 12) or set by provisioning. If not provisioned otherwise, the properties in all descriptors except TerminationState and LocalControl default to empty/"no value" when a Termination is first created or returned to the null Context. When a termination is first created or returned to the null context, this state represents an "idle" line, trunk or other entity. The default contents of the two exceptions are described in 7.1.5 and 7.1.7.

The provisioning of a property value in the MG will override any default value, be it supplied in this protocol specification or in a package. Therefore, if it is essential for the MGC to have full control over the property values of a Termination, it should supply explicit values when ADDing the Termination to a Context. Alternatively, for a physical Termination the MGC can determine any provisioned property values by auditing the Termination while it is in the NULL Context.

There are a number of common properties for Terminations and properties specific to media streams. The common properties are also called the Termination state properties. For each media stream, there are local properties and properties of the received and transmitted flows.

Properties not included in the base protocol are defined in Packages. These properties are referred to by a name consisting of the PackageName and a PropertyId. Most properties have default values described in the Package description. Properties may be read-only or read/write. The possible values of a property may be audited, as can their current values. For properties that are read/write, the MGC can set their values. A property may be declared as "Global" which has a single value shared by all Terminations realizing the package. Related properties are grouped into descriptors for convenience.

When a Termination is added to a Context, the value of its read/write properties can be set by including the appropriate descriptors as parameters to the Add command. Similarly, a property of a Termination in a Context may have its value changed by the Modify command. Properties may also have their values changed when a Termination is moved from one Context to another as a result of a Move command. In some cases, descriptors are returned as output from a command.

In general, if a Descriptor is completely omitted from one of the aforementioned Commands, the properties in that Descriptor retain their prior values for the Termination(s) upon which the Command acts. On the other hand, if some read/write properties are omitted from a Descriptor in a Command (i.e., the Descriptor is only partially specified), those properties will be reset to their default values for the Termination(s) upon which the Command acts, unless the package specifies other behavior. For more details, see 7.1 dealing with the individual Descriptors.

The above behavior applies equally to Signals, Events and their parameters.

…

6.8.2 7.2.3 Subtract

…

When a provisioned Termination is Subtracted from a Context, its descriptor values shall revert to:

- the default value, if specified for the descriptor and not overridden by provisioning;
- otherwise, the provisioned value.

…

[End Correction]

6.9 Multiple Individual Auditing in ABNF

Description:	**In the ASN.1 specification of IndAudLocalControlDescriptor it is allowed to use one each of stream mode, reserve value and reserve group plus any number of package names:** **IndAudLocalControlDescriptor ::= SEQUENCE {** **streamMode NULL OPTIONAL,** **reserveValue NULL OPTIONAL,** **reserveGroup NULL OPTIONAL,** **propertyParms SEQUENCE OF IndAudPropertyParm OPTIONAL,** **... }** **IndAudPropertyParm ::= SEQUENCE {** **name PkgdName,** **...}** **while in the ABNF specification of indAudlocalControlDescriptor it is limited to be exactly one of stream mode, reserve value, reserve group and package name:** **indAudlocalControlDescriptor = LocalControlToken LBRKT indAudlocalParm RBRKT** **; at-most-once per item** **indAudlocalParm = (ModeToken / pkgdName / ReservedValueToken / ReservedGroupToken)** **For example this means that it is possible to have both stream mode and package name in ASN.1, while this is not allowed in ABNF. The ASN.1 encoding and ABNF encoding is not semantically equivalent in this case.** **H.248.1 sect 7.2.5 states: "It is possible to audit multiple individual items in one request."** **The ABNF should be changed to the following in order to be compliant with ASN.1:** **indAudlocalControlDescriptor = LocalControlToken LBRKT indAudlocalParm *(COMMA indAudlocalParm) RBRKT**
Reference:	Date: Mon, 9 Aug 2004 17:55:26 +0200 (CEST) From: Hakan Mattsson *<hakan@cslab.ericsson.se>* To: *megaco@ietf.org* Subject: IndAudlocalControlDescriptor inconsistency

[Begin Correction]

B.2 ABNF specification

…

```
; at-most-once
indAudstreamParm           = ( indAudlocalControlDescriptor )
; SDP too complex to pull out individual pieces for audit,
; hence no individual audit for Local and Remote

indAudstreamDescriptor     = StreamToken EQUAL StreamID
                                LBRKT indAudstreamParm RBRKT

indAudlocalControlDescriptor = LocalControlToken LBRKT indAudlocalParm
                                *(COMMA indAudlocalParm) RBRKT
```

…

[End Correction]

6.10 Loopback usage clarification

Description:	**There are 4 stream mode properties in H.248.1: send, receive, inactive and loopback. Send, receive and inactive are described by section 7.1.7 however H.248.1 is largely silent on the operation of loopback. The use of loopback should be clarified.**
Reference:	COM 16 D-44

[Begin Correction]

7.1.7 LocalControl Descriptor

The LocalControl descriptor contains the Mode property, the ReserveGroup and ReserveValue properties and properties of a Termination (defined in Packages) that are stream specific, and are of interest between the MG and the MGC. Values of properties may be specified as in 7.1.1.

The allowed values for the mode property are send-only, receive-only, send/receive, inactive and loop-back. "Send", ~~and~~ "receive" and “loopback” are with respect to the exterior of the Context, so that, for example, a stream set to mode = sendOnly does not pass received media into the Context. When a stream is set to "loop-back" on a termination, media received (local descriptor) on that termination will be looped back to the sending side (remote descriptor) of the termination and no media is passed between that termination and other terminations in the context. The looped back media shall be sent according to the remote descriptor. The default value for the mode property is "Inactive". Signals and Events are not affected by mode.

[End Correction]

6.11 Commands in ServiceChange on root transaction

Description:	H.248.1 allows multiple commands to be grouped per Transaction. An exception to this is a transaction containing a ServiceChange command specifying the “Root” terminationID and ServiceChangeMethod equal to Restart or Failover. This is because subsequent transactions shall use any ServiceChangeAddress supplied in the transaction response and the negotiated protocol version. This is already implied in the H.248.1 text however the prevention of multiple commands in this scenario is not explicit and may lead to confusion. It is proposed to make this explicit in H.248.1.
Reference:	COM 16 D-44

[Begin Correction]

7.2.8 ServiceChange

…

A ServiceChange Command specifying the "Root" for the TerminationID and ServiceChangeMethod equal to Restart is a registration command by which a Media Gateway announces its existence to the Media Gateway Controller. The Media Gateway may register by specifying the "Root" for the TerminationID and ServiceChangeMethod equal to Failover when the MG detects MGC failures. The Media Gateway is expected to be provisioned with the name of one primary and optionally some number of alternate Media Gateway Controllers. Acknowledgement of the ServiceChange Command completes the registration process, except when the MGC has returned an alternative ServiceChangeMgcId as described in the following paragraph. The MG may specify the transport ServiceChangeAddress to be used by the MGC for sending messages in the ServiceChangeAddress parameter in the input ServiceChangeDescriptor. The MG may specify an address in the ServiceChangeAddress parameter of the ServiceChange request, and the MGC may also do so in the ServiceChange reply. In either case, the recipient must use the supplied address as the destination for all subsequent transaction requests within the association. At the same time, as indicated in clause 9, transaction replies and pending indications must be sent to the address from which the corresponding requests originated. This must be done even if it implies extra messaging because commands and responses cannot be packed together. The TimeStamp parameter shall be sent with a registration command and its response. A message containing a ServiceChange Command specifying “Root” for the TerminationID and a ServiceChangeMethod equal to Restart or Failover shall not contain other commands as these commands should use the new ServiceChangeAddress and negotiated protocol version.

…

[End Correction]

6.12 Annex C and SDP parameters

Description:	H.248.1 Annex C.11 allows the use of SDP equivalents for use in binary implementations of the protocol. One issue that is not clear is whether the SDP is 100% RFC2327 compliant or is subject to the exceptions of H.248.1 section 7.1.8 “Local and Remote Descriptor”. 3GPP in their technical specification TS29.332 “Media Gateway Control Function (MGCF) – IM Media Gateway; Mn Interface”, V6.0.0 (2004-09) make use of SDP equivalents. The note to Table 10.1: “required parameters” makes the assumption that the exceptions of H.248.1 apply. It is believed that 3GPP2 also make this assumption. It is proposed to clarify the text in H.248.1 Annex C.1 that the SDP is subject to the behavior of H.248.1 section 7.1.8.
Reference:	AVD-2663

[Begin Correction]

C.11 SDP equivalents

The SDP equivalents are subject to the SDP exceptions of 7.1.8 described for text encoding of the protocol. Also the CHOOSE wildcard is used as in text encoding of the protocol.

PropertyID	Property tag	Type	Value
SDP_V	B001	String	Protocol Version Ref.: RFC 2327
SDP_O	B002	String	Owner/creator and session ID Ref.: RFC 2327
SDP_S	B003	String	Session name Ref.: RFC 2327
SDP_I	B004	String	Session identifier Ref.: RFC 2327
SDP_U	B005	String	URI of descriptor Ref.: RFC 2327
SDC_E	B006	String	email address Ref.: RFC 2327
SDP_P	B007	String	phone number Ref.: RFC 2327
SDP_C	B008	String	Connection information Ref.: RFC 2327
SDP_B	B009	String	Bandwidth Information Ref.: RFC 2327
SDP_Z	B00A	String	Time zone adjustment Ref.: RFC 2327
SDP_K	B00B	String	Encryption Key Ref.: RFC 2327
SDP_A	B00C	String	Zero or more session attributes Ref.: RFC 2327
SDP_T	B00D	String	Active Session Time Ref.: RFC 2327
SDP_R	B00E	String	Zero or more repeat times Reference: RFC 2327
SDP_M	B00F	String	Media type, port, transport and format Ref.: RFC 2327

[End Correction]

6.13 Case Sensitivity of Profile Names

Description:	Profiles of H.248 are becoming more widely used. One of the important features of the registration of profiles is that the profile name is unique and able to be agreed upon by a MGC and MG. There is a potential interoperability problem in that some implementers may assume that the "Profile Name" is case sensitive, where others may assume that the name is case insensitive. This could cause failure at profile negotiation. H.248.1 is largely silent of the case sensitivity of profile names. However as other protocol constructs such as package names are case-insensitive it is believed that profile names are also case insensitive. Thus H.248.1 § 13 should be updated to reflect this.
Reference:	AVD-2663

[Begin Correction]

6.13.1 13 Profile Definition

Profiles may be specified to further define how the H.248.1 protocol is used and what functionality is supported by a MG. It only describes the capabilities of the MGC/MG H.248 interface. The profile itself specifies what options associated with H.248.1 have been used. For example: transport and packages used for an application.

A profile is identified by a name (IANA registered) and a Version. A name shall be a case-insensitive string up to 64 characters long. Version shall be 1 to 99.

…

[End Correction]

6.14 Profile Negotiation

Description:	Profile negotiation is ambiguous in the situation where the MG issues a ServiceChange request without the ServiceChangeProfile parameter (ie. NoProfile) and the MGC requires a profile to work. It is assumed that the MGC responds with the profile that it wants to support in this situation. However the current text in 7.2.8.1.11 indicates that the profile is "only" returned when the MGC cannot support the profiles in the ServiceChangeRequest. The contributors contend that if no ServiceChangeProfile is added to the request this is the MG indicated "NoProfile". This should be clarified. Furthermore on reception of the Profile in the ServiceChangeResponse it is unclear whether the MG should: a) reissue a ServiceChange registration with the MGC indicated profile to accept the profile, or, simply await commands from the MGC.
Reference:	AVD-2663

[Begin Correction]

7.2.8.1.11 ServiceChange Command and Response

…

- ServiceChangeProfile, if the responder wishes to negotiate the profile to be used for the association. The profile (name and version) is ~~only~~ returned in reply in the case that the MGC cannot support the specified profiles in the ServiceChangeRequest. The profile "NoProfile" is assumed if no ServiceChangeProfile is included in the ServiceChangeRequest. The returned reply shall indicate the profile and version supported or "NoProfile" if no profile is supported. Upon reception of a profile in the reply the MG may continue the relationship with the current MGC by issuing a subsequent ServiceChangeRequest with the appropriate profile or contact secondary MGCs and establish a relationship with them. If the profile is not returned the MGC will use the capabilities specified by the Profile indicated in the service change request;

…

[End Correction]

6.15 Conflict between H.248.1 Version 2 Corrigendum 1 and H.248.8

Description:	H.248.1 contains a contradiction with H.248.8. H.248.1 Section 6.3.2 (added in H.248.1 V2 Corrigendum 1) indicates that error code 431 "No TerminationID matched a wildcard" is used when a specifically named termination does not exist in any context (besides NULL) when the ContextID is wildcarded. However, in the instance that there are no contexts at all, error code 411 "The transaction refers to an unknown ContextID" is clearly the more correct choice, since the MG can determine that an error has occurred without progressing to the command level of the message to process the TerminationID.
Reference:	AVD-2706

[Begin Correction]

6.15.1 6.3.2 ContextID wildcarded (ALL) with TerminationID specific

In the case where the ContextID is wildcarded (i.e., ContextID = ALL) and the TerminationID is fully specified, the effect is identical to a command specifying the non-NULL context that contains the specified termination. Thus a search must be made to find the context and only one instance of the command is executed. No errors are reported for Contexts that do not contain the specified termination. If the termination is not contained in any (non-NULL) context, then error 431 is returned. If there are no contexts other than NULL in existence, error code 411 is returned. Use of this form of action rather than one specifying the ContextID is discouraged but may be useful, for example in correcting conflicting state between MG and MGC.

…

[End Correction]

6.16 AuditCapability of Signals

Description:	H.248.1 section 7.2.6 currently states that it is possible to perform an AuditCapabilities of a signal and that the reply contains a range of values for: the keepactive indication, signal type, duration, signal completion indication and package defined properties. However the encoding in Annex A and B does not allow a range of values to be returned for keepactive, signal type, duration, and signal completion. The contributors contend that to provide such a range provides little value as the values are defined in the H.248.1 protocol and signals have well defined defaults. Thus it proposed to delete the ability to return a range for the above parameters. This would not pose any backward compatibility problems as because the encoding was not defined noone has been able to use this function.
Reference:	COM 16 D-119

[Begin Correction]

6.16.1

6.16.2 7.2.6 AuditCapabilities

Descriptors or individual properties, signals, events and statistics can be audited.

- An audit of a entire descriptor may be requested by identifying the desired descriptor in the AuditDescriptor with no further information.
- To audit an individual property in the media descriptor the relevant stream ID (optional) and propertyID are included. A list of possible values of the property are returned.
- To audit a signal the relevant signal list ID and/or signal ID are provided. A list of possible values associated with each signal parameter is returned (including: package defined properties). ~~t~~The keepactive indication, signal type, duration and~~,~~ signal completion indication~~ and package defined properties~~ are not returned.~~).~~

…

[End Correction]

6.17 Media Type Mismatch

Description:	H.248.1 allows media descriptor parameters to be set independently on terminations in different commands. The H.248 connection model also makes use of this fact to be able to describe different functions on an MG. For example if Codec=GSM AMR is specified on TerminationA and Codec=G.711 is specified on TerminationB then the MG can assume that transcoding will take place. The problem comes when the MGC tries to change the codec on one or more of the terminations. For example: the MGC wants to change Terminations A and B to both to G.729. It issues a MOD.req on Termination A Codec=G729. What does the MG do? Does it try to insert another transcoder to transcode G.729 to G.711? This would be a waste of resources as no transcoding will be required when TerminationB is set. Does the MG reject the command as it can't support G.729 to G.711 transcoding? The MG must have unambiguous knowledge of when to apply a function eg. transcoding. Therefore it is proposed that the MG only applies the function when the streammode associated with the function is NOT inactive.
Reference:	COM 16 D-119

[Begin Correction]

6.17.1 6.2.4 Termination properties and descriptors

…

When a Termination is added to a Context, the value of its read/write properties can be set by including the appropriate descriptors as parameters to the Add command. Similarly, a property of a Termination in a Context may have its value changed by the Modify command. Properties may also have their values changed when a Termination is moved from one Context to another as a result of a Move command. In some cases, descriptors are returned as output from a command.

By setting properties on different terminations in the same context the MG can be instructed to perform certain functions. For example: if a G.711 codec is set on Termination A and a G.729 codec is set on Termination B then the MG performs a transcoding function. This transcoding function is activated when the Mode Property (see Local Control Descriptor 7.1.7) on the termination/streams affected by the function are not set to "Inactive".

In general, if a Descriptor is completely omitted from one of the aforementioned Commands, the properties in that Descriptor retain their prior values for the Termination(s) upon which the Command acts. On the other hand, if some read/write properties are omitted from a Descriptor in a Command (i.e., the Descriptor is only partially specified), those properties will be reset to their default values for the Termination(s) upon which the Command acts, unless the package specifies other behavior. For more details, see 7.1 dealing with the individual Descriptors.

…

[End Correction]

6.18 Notify Avalanche

Description:	During a failure of the MGC-MG control link a large number of events could occur on the MG. During this period the MG may still continue to generate Notify requests that need to be sent to the MGC. However as the control link has failed these Notify.reqs would be queued. Once the link is re-established (ServiceChangeMethod = Disconnected) then these Notify requests would be sent. This could result in a potential signaling avalanche problem. An avalanche problem has previously been identified in H.248.1 section "9.2 Protection against restart avalanche" however this was restricted to ServiceChange commands. It is proposed to add some advisory text to minimize the chances of Notify avalanches.
Reference:	COM 16 D-119

[Begin Correction]

9.3 Protection against Notify avalanche

In the event that a control association goes down the MG may continue to generate notify messages. These Notify messages must be buffered until the control association comes back into service (ServiceChangeMethod = Disconnected). When the control association comes back into service the rapid sending of the Notifications may result in a notification avalanche. To prevent this from occurring the MG should send the Notifications in a restricted manner until the buffer is cleared.

…

[End Correction]

6.19 Topology Reply

Description:	H.248.1 section 7.1.18 states that "It is possible to have an action containing only a Topology descriptor". In the text encoding, what should the response be to such an action? The ABNF requires something in the actionReply - at least one context property, command reply or error descriptor. The behaviour for v1 and v2 needs to be described. E.g. the MG should echo back what was sent. If CHOOSE $ was indicated you should send back the chosen TerminationID. *Editor's Note: It is not permissible to have $ in a TP descriptor without an Add command in the same action.*
Reference:	Subject: RE: [Megaco] Topology Questions Date: Wed, 29 Jun 2005 05:35:39 +1000 From: Kevin Boyle <kboyle@nortel.com> To: Christian Groves (BR/EPA) <christian.groves@ericsson.com>, Frank Reno <hmegaco@yahoo.com> CC: megaco@ietf.org

[Begin Correction]

6.19.1 7.1.18 Topology descriptor

…

The Topology descriptor occurs before the commands in an action. It is possible to have an action containing only a Topology descriptor, provided that the Context to which the action applies already exists. In the instance where there are no commands in an action containing a Topology Descriptor, the Topology Descriptor is echoed back to the MGC.

…

[End Correction]

6.20 Statistics and Sub-lists

Description:	Section 12.1.5 indicates that a possible type for the value of a Statistic is "Sub-list-of".
	However in the ABNF encoding a statistic parameter may only have a single value. Ie.
	ABNF
	`statisticsDescriptor = StatsToken LBRKT statisticsParameter *(COMMA statisticsParameter) RBRKT`
	`;at-most-once per item`
	`statisticsParameter = pkgdName [EQUAL VALUE]`
	The ASN.1 Syntax supports multiple values that may be used to encode a sub-list. However it does not support a flag as per the property relation field.
	ASN.1
	StatisticsDescriptor ::= SEQUENCE OF StatisticsParameter
	StatisticsParameter ::= SEQUENCE
	{
	statName PkgdName,
	statValue Value OPTIONAL
	}
	Value ::= SEQUENCE OF OCTET STRING
	This apparent conflict between the encoding and the procedures was probably introduced when the value of Statistic was changed from UNIT to Type in H.248.1v2 IG. Therefore it is proposed to update the ASN.1 and ABNF so that the types of statistics are consistent with existing types for properties and parameters.

Reference:	COM 16 D-119

[Begin Correction]

A.2 ASN.1 Specification

…

```
StatisticsDescriptor ::= SEQUENCE OF StatisticsParameter

StatisticsParameter ::= SEQUENCE
{
         statName                        PkgdName,
         statValue                       Value OPTIONAL
                              -- A sequence of values indicates a sub-list
}
```

…

B.2 ABNF Specification

…

```
statisticsDescriptor    = StatsToken LBRKT statisticsParameter *(COMMA
             statisticsParameter) RBRKT

;at-most-once per item
statisticsParameter     = pkgdName [EQUAL ( VALUE /
                               LSBRKT VALUE *(COMMA VALUE) RSBRKT]
```

…

[End Correction]

6.21 Protocol version negotiation

Description:	H.248.1 is silent on what to do when either the MGC or the MG fails to abide by the negotiated protocol version within a control association. Consider the following: The MG offers Version 2, which the MGC accepts. The MG then starts sending all messages as Version 1. This is clearly not what was intended in the version negotiation procedures. H.248.1 should allow the receiver of the "off-version" messaging to reject it as not in line with the negotiated version. The most appropriate error code is 406, "Version not supported".
Reference:	AVD-2820

[Begin Correction]

11.3 Negotiation of protocol version

…

If the MGC supports the version indicated by the MG, ~~it~~ both the MGC and MG shall conform to that version in all subsequent messages. In this case it is optional for the MGC to return a version in the ServiceChange Reply. Any subsequent messaging that does not conform to the negotiated version shall be rejected with Error Code 406 ("Version Not Supported").

…

[End Correction]

6.22 Clarification of error code usage in wildcarding procedures

Description:	It appears that the error code required in clause 6.3.2/H.248.1 is a cut and paste error, as error code 435 makes more sense. However, the long-standing existence of 431 in that clause may lead to backwards compatibility problems if the error code is just changed. The text needs to be updated to allow either possibility.
Reference:	COM16 D-223

[Begin Correction]

6.3.2 ContextID wildcarded (ALL) with TerminationID specific

In the case where the ContextID is wildcarded (i.e., ContextID = ALL) and the TerminationID is fully specified, the effect is identical to a command specifying the non-NULL context that contains the specified termination. Thus a search must be made to find the context and only one instance of the command is executed. No errors are reported for Contexts that do not contain the specified termination. If the termination is not contained in any (non-NULL) context, then error code 435 ("TerminationID is not in specified context") is returned, though error code 431 ("No TerminationID matched a wildcard") may be returned in order to maintain backward compatibility~~is returned~~. Use of this form of action rather than one specifying the ContextID is discouraged but may be useful, for example in correcting conflicting state between MG and MGC.

…

[End Correction]

6.23 ServiceStates clarification for continuity testing

Description:	The continuity package does not specify whether or not a termination must be placed in the Test state prior to conducting a continuity test.
Reference:	COM16 D-224

[Begin Correction]

6.23.1 E.10.5 Procedures

…

When a continuity test is performed on a Termination, no echo devices or codecs shall be active on that Termination. The termination under test does not need to have its ServiceStates Property set to Test.

…

[End Correction]

6.24 Clarification of termination service state upon restart of MG

Description:	During discussion on the 3GPP and IETF Megaco mailing lists it became apparent that there is a source of confusion on the default states of all terminations after a ServiceChange restart. It is widely agreed that all terminations including physical and ephemeral terminations are default "InService" after the ServiceChange. However H.248.1 doesn't explicitly make this statement.
Reference:	COM16 D-274

[Begin Correction]

6.24.1 7.2.8 ServiceChange

…

3) Restart – indicates that service will be restored on the specified t~~T~~erminations after expiration of the ServiceChangeDelay. The S~~s~~erviceStates Property should be set to "~~inService~~InService" upon expiry of ServiceChangeDelay. Upon receipt of a ServiceChange Command on Root with ServiceChangeMethod Restart all terminations are assumed to be “InService”. This includes physical and ephemeral terminations. Those terminations which are “OutOfService” may be reported by subsequent ServiceChange Commands with ServiceChangeMethod Forced.

…

[End Correction]

6.25 Alignment of text among events in the Tone Detection Package

Description:	The Tone Detection Package specifies three different events for tone detection: ‘Start Tone Detected’, ‘End Tone Detected’ and ‘Long Tone Detected’. While the ‘*’ wildcard in the EventsDescriptor parameter ‘tl’ is allowed for the ‘Start Tone Detected’ and ‘Long Tone Detected’ events, H.248 currently doesn’t allow it in the ‘End Tone Detected’ event. There is no reason why the wildcard should not be allowed in the ‘End Tone Detected’ event. In fact, a very common use of this event, as of the other two, is the detection of DTMF tones. For DTMF it is common to order the MGW to detect any DTMF digit, as it is not known in advance which DTMF digit will be received in the line.
Reference:	COM16 D-303

[Begin Correction]

6.25.1 E.4.2 Events

…

End tone detected

EventID: etd, 0x0002

Detects the end of a tone.

EventDescriptor parameters:

Tone id list

ParameterID: tl (0x0001)

Type: enumeration or list of enumerated types

Possible values: The only tone id defined in this package is "wild card" which is "*" in text encoding and 0x0000 in binary. ~~No possible values are specified in this package.~~ Extensions to this package would add possible values for tone id. If tl is "wild card", any tone id is detected.

…

[End Correction]

6.26 Missing header in Long Tone Detected Event

Description:	The 'Long Tone Detected' event in H.248.1 v2 lists three parameters under the EventsDescriptor. However the last one, the Tone Id, is not an EventsDescriptor parameter, but an ObservedEventsDescriptor parameter.
Reference:	COM16 D-303

[Begin Correction]

6.26.1 E.4.2 Events

…

Long tone detected

EventID: ltd, 0x0003

Detects that a tone has been playing for at least a certain amount of time

EventDescriptor parameters:

Tone id list

ParameterID: tl (0x0001)

Type: enumeration or list

Possible values: "wildcard" as defined above is the only value defined in this package. Extensions to this package would add possible values for tone id.

Duration:

ParameterID: dur (0x0002)

Type: integer, duration to test against

Possible values: any legal integer, expressed in milliseconds

ObservedEventsDescriptor parameters:

Tone id:

ParameterID: tid (0x0003)

Type: Enumeration

Possible values: No possible values are specified in this package. Extensions to this package would add possible values for tone id.

…

[End Correction]

6.27 Clarification of package definition requirements for enumerations

Description:	Packages may define properties, statistics and parameters for signals and events of enumeration type. As stated in the guideline for package definition in H.248.1, the possible values for these parameters must be also specified in the package. Once the values are specified, the binary encoding is unambiguous, as with ASN.1 each of the values of an enumeration type is associated to an integer. However, with text encoding, the encoded values may use any character or character string, not only integers. Therefore it is important that the package specifies not only the possible values that a property, statistic or parameter of type enumeration may take, but also the strings to be used to encode each of the values if ABNF, Annex B/H.248.1 is used.
Reference:	COM16 D-303

[Begin Correction]

6.27.1 12.1.2 Properties

Properties defined by the package, specifying:

…

Type: One of:

Boolean

String: UTF-8 string

Octet String: A number of octets. See Annexes A and B.3 for encoding.

Integer: 4 byte signed integer

Double: 8 byte signed integer

Character: ~~unicode~~Unicode UTF-8 encoding of a single letter. Could be more than one octet.

Enumeration: one of a list of possible unique values (see 12.3). Packages MUST define the text and binary encodings for each value in the enumeration.

Sub-list: a list of several values from a list. The type of sub-list SHALL also be specified. The type shall be chosen from the types specified in this clause (with the exception of sub-list). For example, Type: sublist of enumeration. The encoding of sub-lists is specified in Annexes A and B.3.

…

12.1.5 Statistics

Statistics defined by the package, specifying:

…

Type: One of:

Boolean

String: UTF-8 string

Octet String: A number of octets. See Annex A and B.3 for encoding

Integer: 4 byte signed integer

Double: 8 byte signed integer

Character: Unicode UTF-8 encoding of a single letter. Could be more than one octet.

Enumeration: One of a list of possible unique values (see 12.3). Packages MUST define the text and binary encodings for each value in the enumeration.

Sub-list: A list of several values from a list. The type of sub-list SHALL also be specified. The type shall be chosen from the types specified in this clause (with the exception of sub-list). For example, Type: sub-list of enumeration. The encoding of sub-lists is specified in Annex A and B.3.

…

6.27.2 12.2 Guidelines to defining Parameters to Events and Signals

…

Type: One of:

Boolean

String: UTF-8 octet string

Octet String: A number of octets. See Annexes A and B.3 for encoding.

Integer: 4-octet signed integer

Double: 8-octet signed integer

Character: ~~unicode~~Unicode UTF-8 encoding of a single letter. Could be more than one octet.

Enumeration: one of a list of possible unique values (see 12.3). Packages MUST define the text and binary encodings for each value in the enumeration.

Sub-list: a list of several values from a list (not supported for statistics). The type of sub-list SHALL also be specified. The type shall be chosen from the types specified in this clause (with the exception of sub-list). For example, Type: sub-list of enumeration. The encoding of sub-lists is specified in Annexes A and B.3.

…

[End Correction]

6.28 Clarification of use of ABNF encodings of octet strings

Description:	Properties, statistics and signal and events parameters can be defined as of type Octet String, among other types. This is described in H.248.1 12.1.2, 12.1.5 and 12.2. These chapters refer to Annex B.3 for how the actual encoding of the Octet String shall be done. Annex B.3 does indeed describe a method for the encoding of strings, but fails to make a precise reference to the type Octet String, as the object of the method it is describing. Instead, it talks about "representing a string of octets" or "encoding octet strings". As ABNF defines still another type called "octetString" to describe SDP lines, and which is different to the type Octet String defined above (is not compatible), there is a risk to misinterpret the applicability of B.3.
Reference:	COM16 D-303

[Begin Correction]

6.28.1 B.3 Hexadecimal octet coding

Hexadecimal octet coding is a means of representing ~~a string of octets~~package elements of type Octet String as a string of hexadecimal digits, with two digits representing each octet. This octet encoding should be used when encoding ~~octet strings~~values of type Octet String in the text version of the protocol.

For each octet, the 8-bit sequence is encoded as two hexadecimal digits. Bit 0 is the first transmitted; bit 7 is the last.

Bits 7-4 are encoded as the first hexadecimal digit, with Bit 7 as MSB and Bit 4 as LSB. Bits 3-0 are encoded as the second hexadecimal digit, with Bit 3 as MSB and Bit 0 as LSB.

Examples:

Octet bit pattern	Hexadecimal coding
00011011	D8
11100100	27
10000011 10100010 11001000 00001001	C1451390

This encoding is not applicable to the octetString construct defined in section B.2

[End Correction]

6.29 Clarification of encoding for packet loss statistic in Annex E.12

Description:	H.248.1 E.12.4 defines the statistics packet loss rtp/pl to describe the packet loss rate, as a percentage. Although this statistics element is defined as type double, it is meant to hold both the whole part and the fractional part of the percentage. The ASN.1 “double” encoding of this element entails multiplying the percentage by 2 ^ 32 in order to obtain an integer and then use 4 octets to encode the resulting integer. In ABNF is questionable if the same applies, as that would lead to a long string. This seems unnecessary, especially considering that RFC 3550, to which E.12.4 refers to when defining rtp/pl, defines the packet fraction lost with only 8 bits. Therefore it is proposed to clarify that the notation x.y is allowed when encoding rtp/pl with ABNF.
Reference:	COM16 D-303

EDITOR’S NOTE: This item was modified by AVD-2916 at the August 2006 Ottawa meeting.

EDITOR’S NOTE: This item was modified by COM16 TD-354/WP2 at the November 2006 Geneva meeting.

[Begin Correction]

E.12.4.3 Packet Loss

…

Possible values: a 32-bit whole number and a 32-bit fraction. ~~The value shall be encoded in ABNF as “x.y” where x is the whole part and y the fractional part of the percentage.~~

The actual data type is a fixed point number, which is mapped on the H.248 type “double”. The whole number and the fractional part shall be interpreted as 32-bit integer each, thus the “double” type for rtp/pl shall be encoded as the concatenation of two integers. The fractional part must be therefore first converted into an integer, i.e. multiplied by 2^{32}.

For example, given the percentage 23.625, to express this in the Packet Loss Statistic we perform the following steps:

1. Convert the number to binary: 23.625_{10} equals 10111.101_{2}.

2. Multiply the binary number by 2^{32}. Note that multiplying a binary number by a power of two is the same as shifting it to the left the same number of places as the power of two by which it is being multiplied. 1011110100000000000000000000000000000 is the binary value of our statistic.

3. Convert the binary value to decimal double: the statistic is reported as having value 101468602368.

To return the statistic back to its fractional representation, the steps are reversed. Once the double is converted back to its binary form, the lower 32 bits represent the fraction and the rest are the whole number. From there, conversion back to a percentage is fairly straightforward.

Binary encoding shall be as described in clause A.2 for type “integer”.

…

[End Correction]

6.30 Clarification of package versions versus protocol versions

Description:	SG16 received a liaison from ETSI TISPAN which highlighted possible confusion between package versioning and protocol versioning. Since only certain packages are linked to the protocol version (those with syntactical requirements in the protocol, like segmentation for example), some text clarifying that this linkage is limited to only certain packages would be of benefit.
Reference:	AVD-2945

[Begin Correction]

12 Package definition

The primary mechanism for extension is by means of Packages. Packages define additional properties that may occur on terminations and contexts and events, signals and statistics that may occur on terminations.

Packages and their versions are not generally related to the version of the protocol being used. If packages defined outside H.248.1 have a dependence upon the syntax or features of a particular version of the protocol, they must explicitly state the minimum version of the protocol upon which they are dependent.

Packages defined by IETF will appear in separate RFCs.

…

[End Correction]

[Begin Correction]

6.30.1 E.2 Base Root Package

Base Root Package

PackageID: root (0x0002)

Version: 2

Extends: None

Description:

This package defines Gateway wide properties.

Version 2 of the base root package requires at least version 2 of the protocol.

[End Correction]

6.31 Correction to profile negotiation in Appendix I

Description:	The example in Appendix I shows a profile negotiation. It differs from the profile negotiation text, however, in that it shows the MGC including the same profile parameter in the response to the registration. In this instance, the MGC is agreeing to the profile requested by the MG, and so does not need to return this parameter.
Reference:	AVD-2843

[Begin Correction]

6.31.1 I.1.1 Programming residential GW analog line terminations for idle behaviour

…

2) The MGC sends a reply:

```
MGC to MG1:
MEGACO/1 [123.123.123.4]:55555
Reply = 9998 {
   Context = - {ServiceChange = ROOT {
     Services {ServiceChangeAddress=55555~~, Profile=ResGW/1~~} } }
}
```

…

[End Correction]

6.32 Termination ID in Add Reply with error

Description:	In a typical ADD.request the MGC will use $ (CHOOSE) for the TerminationID. However if there is an error (e.g. 234 Max number of Terminations in a Context exceeded" the MG may not be able to provide a "real" TerminationID. H.248.1 is silent on what TerminationID should be returned in this case. The contributors believe that following the principle that in cases where the MG is unable to determine a TerminationId from a command request this should be returned in the command reply should be followed.
Reference:	Subject: [Megaco] Max terms in context Date: Thu, 13 Jul 2006 14:50:22 -0700 From: Wainwright, John (Com US) <john.wainwright@siemens.com> To: <megaco@ietf.org>

[Begin Correction]

6.32.1 8 Transactions

…

At the first failing command in a transaction, processing of the remaining commands in that transaction stops. If a command contains a wildcarded TerminationID, the command is attempted with each of the actual TerminationIDs matching the wildcard. A response within the TransactionReply is included for each matching TerminationID, even if one or more instances generated an error. If the MG cannot return a TerminationID in response to a wildcarded termination then the original wildcard TerminationID should be returned. If any TerminationID matching a wildcard results in an error when executed, any commands following the wildcarded command are not attempted.

…

[End Correction]

6.33 Codec Selection

Description:	H.248.1 describes procedures for determines capabilities to be supported in the Local and Remote descriptors. These rules can be used to determine the codec/s that is/are to be supported on a stream. The rules allow for symmetric codec selection OR asymmetric codec selection. This may not be clear from the text. It is proposed to add a note to this effect.
Reference:	AVD-2890

[Begin Correction]

6.33.1 7.1.8 Local and Remote Descriptors

…

If ReserveGroup is "False" and ReserveValue is "False", then the MG should apply the following rules to resolve Local and Remote to a single alternative each:

- The MG chooses the first alternative in Local for which it is able to support at least one alternative in Remote.
- If the MG is unable to support at least one Local and one Remote alternative, it returns Error Code 510 ("Insufficient resources").
- The MG returns its selected alternative in each of Local and Remote.

NOTE: The above rules allow the MG to prioritize the selection of the same codec in both the Local and Remote Descriptors; however, it also permits the MG to choose different codecs in each descriptor.

A new setting of a Local or Remote Descriptor completely replaces the previous setting of that descriptor in the MG. Thus, to retain information from the previous setting, the MGC must include that information in the new setting. If the MGC wishes to delete some information from the existing descriptor, it merely resends the descriptor (in a Modify Command) with the unwanted information stripped out.

[End Correction]

6.34 Clarification of use of ServiceChangeAddress

Description:	The ServiceChange parameter "ServiceChangeAddress" was discussed in the April 2006 meeting in the context of H.Sup7 (see D-219 & D-217). Potential uses cases were identified and described in § 5.5.1.2/H.Sup7.
Reference:	AVD-2917

[Begin Correction]

6.34.1 7.2.8.1.3 ServiceChangeAddress

The optional ServiceChangeAddress parameter specifies the address (e.g. IP port number for IP networks) to be used for subsequent communications. It can be specified in the input parameter descriptor or the returned result descriptor. ServiceChangeAddress and ServiceChangeMgcID parameters must not both be present in the ServiceChange Descriptor or the ServiceChangeResult Descriptor. The ServiceChangeAddress provides an address to be used within the context of the association currently being negotiated, while the ServiceChangeMgcID provides an alternate address where the MG should seek to establish another association. ~~Note that the use of ServiceChangeAddress is not encouraged.~~ MGCs and MGs must be able to cope with the ServiceChangeAddress being either a full address or just a port number in the case of ~~TCP~~ IP-based transports, such as UDP, TCP or SCTP.

[End Correction]

6.35 Clarification for context attribute values when omitted from actions

Description:	Clause 6.2.4/H.248.1 clarifies the handling of properties and descriptors for terminations, and clause 7.1.19/H.248.1 clarifies the handling of properties specified through the ContextAttribute Descriptor. Such a clarification would be also beneficial for attributes on context-level.
Reference:	AVD-2890

[Begin Correction]

6.35.1 6.1.1 Context attributes and descriptors

The attributes of Contexts are:

- ContextID.
- The topology (who hears/sees whom).

  The topology of a Context describes the flow of media between the Terminations within a Context. In contrast, the mode of a Termination (send/receive/…) describes the flow of the media at the ingress/egress of the media gateway.
- The priority is used for a Context in order to provide the MG with information about a certain precedence handling for a Context. The MGC can also use the priority to control autonomously the traffic precedence in the MG in a smooth way in certain situations (e.g. restart), when a lot of Contexts must be handled simultaneously. Priority 0 is the lowest priority and a priority of 15 is the highest priority.
- An indicator for an emergency call is also provided to allow a preference handling in the MG.

In general, if a context attribute is completely omitted from a H.248 action, the attribute of the corresponding context retains its prior value.

[End Correction]

6.36 Clarification of use of unsigned integer

Description:	It was discovered when discussing statistics an their encoding that unsigned integer was not specified in clause 12, but is defined in both the ASN.1 and ABNF encodings. Since the protocol is capable of encoding this, it should be available for use in packages.
Reference:	Discussions at August 2006 Ottawa meeting

[Begin Correction]

6.36.1 12.1.2 Properties

Properties defined by the package, specifying:

Property Name: only descriptive

PropertyID: is an identifier

Description:

Type: One of:

Boolean

String: UTF-8 string

Octet String: A number of octets. See Annex A and B.3 for encoding

Integer: 4 byte signed integer

Unsigned Integer: 4-octet unsigned integer

Double: 8 byte signed integer

Character: unicode UTF-8 encoding of a single letter. Could be more than one octet.

Enumeration: one of a list of possible unique values (see 12.3)

Sub-list: a list of several values from a list. The type of sub-list SHALL also be specified. The type shall be chosen from the types specified in this section (with the exception of sub-list). For example, Type: sub-list of enumeration. The encoding of sub-lists is specified in Annexes A and B.3.

…

[End Correction]

[Begin Correction]

6.36.2 12.2 Guidelines to defining parameters to events and signals

Parameter Name: only descriptive

ParameterID: is an identifier. The textual ParameterID of parameters to Events and Signals shall not start with "EPA" and "SPA", respectively. The textual ParameterID shall also not be "ST", "Stream", "SY", "SignalType", "DR", "Duration", "NC", "NotifyCompletion", "KA", "Keepactive", "EB", "Embed", "DM" or "DigitMap".

Type: One of:

Boolean

String: UTF-8 octet string

Octet String: A number of octets. See Annex A and B.3 for encoding

Integer: 4-octet signed integer

Unsigned Integer: 4-octet unsigned integer

Double: 8-octet signed integer

Character: unicode UTF-8 encoding of a single letter. Could be more than one octet.

Enumeration: one of a list of possible unique values (see 12.3)

Sub-list: a list of several values from a list (not supported for statistics). The type of sub-list SHALL also be specified. The type shall be chosen from the types specified in this section (with the exception of sub-list). For example, Type: sub-list of enumeration. The encoding of sub-lists is specified in Annex A and B.3.

…

[End Correction]

6.37 Clarification of SDP requirements for the MG

Description:	There have been many questions about whether the MG is required to be compliant to RFC 2327 with regards to SDP it sends. This is due to the exceptions outlined in clause 7.1.8 that apply only to the MGC. The MG requirement needs to be explicit.
Reference:	Discussions at November 2006 Geneva meeting based on COM16 TD-271/GEN

[Begin Correction]

6.37.1 7.1.8 Local and Remote Descriptors

…

When text encoding the protocol, the descriptors consist of session descriptions as defined in SDP (RFC 2327). In session descriptions sent from the MG to the MGC, the SDP must comply with RFC 2327. In session descriptions sent from the MGC to the MG, the following exceptions to the syntax of RFC 2327 are allowed:

…

[End Correction]

6.38 Correction to SDP examples in Appendix I

Description:	The SDP used in the example in Appendix I is not compliant to RFC 2327 or RFC 4566 in that the t= line is appearing before the c= line, even though the c= line applies to the entire description.
Reference:	COM16 C-69

[Begin Correction]

6.38.1 I.1.2 Collecting originator digits and initiating termination

…

13) MG1 acknowledges the new Termination and fills in the Local IP address and UDP port. It also makes a choice for the codec based on the MGC preferences in Local. MG1 sets the RTP port to 2222.

```
MEGACO/2 [124.124.124.222]:55555
Reply = 10003 {
   Context = 2000 {
      Add = A4444,
      Add=A4445{
         Media {
             Stream = 1 {
                 Local {
v=0
o=- 2890844526 2890842807 IN IP4 124.124.124.222
s=-
~~t= 0 0~~
c=IN IP4 124.124.124.222
t= 0 0
m=audio 2222 RTP/AVP 4
a=ptime:30
a=recvonly
                 } ; RTP profile for G.723.1 is 4
             }
         }
      }
   }
}
```

…

15) This is acknowledged. The stream port number is different from the control port number. In this case it is 1111 (in SDP).

```
MG2 to MGC:
MEGACO/1 [125.125.125.111]:55555
Reply = 50003 {
   Context = 5000 {
     Add = A5555,
      Add = A5556{
         Media {
            Stream = 1 {
                Local {
v=0
o=- 7736844526 7736842807 IN IP4 125.125.125.111
s=-
~~t= 0 0~~
c=IN IP4 125.125.125.111
t= 0 0
m=audio 1111 RTP/AVP 4
}
            } ; RTP profile for G.723.1 is 4
         }
       }
   }
}
```

16) The above IPAddr and UDPport need to be given to MG1 now.

```
MGC to MG1:
MEGACO/1 [123.123.123.4]:55555
Transaction = 10005 {
  Context = 2000 {
    Modify = A4444 {
      Signals {cg/rt}
    },
    Modify = A4445 {
       Media {
            Stream = 1 {
                Remote {
v=0
o=- 7736844526 7736842807 IN IP4 125.125.125.111
s=-
~~t= 0 0~~
c=IN IP4 125.125.125.111
t= 0 0
m=audio 1111 RTP/AVP 4
                }
            } ; RTP profile for G.723.1 is 4
        }
    }
  }
}
```

…

20) The MG2 replies.

```
MEGACO/1 [125.125.125.111]:55555
Reply = 50007 {
   Context = - {
      AuditValue = A5556 {
          Media {
             TerminationState { ServiceStates = InService,
                 Buffer = OFF },
             Stream = 1 {
                 LocalControl { Mode = SendReceive,
                    nt/jit=40 },
                 Local {
v=0
o=- 7736844526 7736842807 IN IP4 125.125.125.111
s=-
~~t= 0 0~~
c=IN IP4 125.125.125.111
t= 0 0
m=audio 1111 RTP/AVP  4
a=ptime:30
                },
                 Remote {
v=0
o=- 2890844526 2890842807 IN IP4 124.124.124.222
s=-
~~t= 0 0~~
c=IN IP4 124.124.124.222
t= 0 0
m=audio 2222 RTP/AVP  4
a=ptime:30
                 } } },
          Events,
          Signals,
          DigitMap,
          Packages {nt-1, rtp-1},
          Statistics { rtp/ps=1200,  ; packets sent
                       nt/os=62300, ; octets sent
                       rtp/pr=700, ; packets received
                       nt/or=45100, ; octets received
                       rtp/pl=0.2,  ; % packet loss
                       rtp/jit=20,
                       rtp/delay=40 } ; avg latency
       }
    }
}
```

…

[End Correction]

6.39 Correction of package name in Signals Descriptor text

Description:	The text in clause 7.1.11 refers to the "Continuity Test" package. The correct name of the package is the "Basic Continuity Package".
Reference:	AVD-2972a

[Begin Correction]

6.39.1 7.1.11 Signals descriptor

…

Signals are MG-generated media such as tones and announcements as well as bearer-related signals such as hookswitch. More complex signals may include a sequence of such simple signals interspersed with and conditioned upon the receipt and analysis of media or bearer-related signals. Examples include echoing of received data as in the Basic Continuity ~~Test p~~Package (clause E.10). Signals may also request preparation of media content for future signals.

…

[End Correction]

6.40 Clarification of construction of Annex C types

Description:	Annex C indicates that values that are less than one octet shall appear in the low-order bytes of the octet, but does not indicate what the high-order bits should be set to (0 or 1). These should be set to 0.
Reference:	AVD-3067

[Begin Correction]

6.40.1 Annex C Tags for media stream properties

…

When a type is smaller than one octet, the value shall be stored in the low-order bits of an octet string of size 1. 0-padding should be used for the remaining high-order bits.

[End Correction]

6.41 Clarification on the Move Command

Description:	The current H.248.1 does not explicitly mention that signal playout is maintained when a termination is moved from a context via a MOVE command. However the H.248.1 is clear that all descriptors maintain their current values thus this confirms that signal playout is maintained. Explicit text may help clarify this situation.
Reference:	C-291

[Begin Correction]

6.41.1 7.2.4 Move

…

The Move command does not affect the properties/signals/events/statistics of the Termination on which it operates, except those properties explicitly modified by descriptors included in the Move command. As such, unless the Signals Descriptor is modified the playout of signals continues. The AuditDescriptor with the Statistics option, for example, would return statistics on the Termination just prior to the Move. Possible descriptors returned from Move are the same as for Add.

[End Correction]

6.42 Clarification on maximum number of Contexts

Description:	Property *root/maxNumberOfContexts* has read-only access and may be used by the MGC as part of his management concerning MG resources. The managed resource component would be then of type "H.248 Context". The MGC could beneficially manage such MG resources in a deterministic manner. Such an MGC resource management is appropriate in general for MG resources, which may *not* be used in a *statistical multiplexed* manner (e.g. like resources behind physical terminations), or in case that worst-case assumptions concerning MG resource consumption per Context are made. But the usage of this property is questionable or needs more careful consideration for cases like: – a number of different connection models supported by a H.248 profile, or – MG termination or context admission controls on basis of statistical multiplexed models for resources required for a stream, termination or context realization, or – IP-to-IP interworking with a number of possible instantiations of the single (IP, IP) connection model, e.g. media-agnostic, media-aware or transport-protocol agnostic etc. The semantic of property *root/maxNumberOfContexts* may benefit from further clarification.
Reference:	C-236

[Begin Correction]

6.42.1 E.2.1 Properties

MaxNrOfContexts

PropertyID: maxNumberOfContexts (0x0001)

The value of this property gives the maximum number of contexts that can exist at any time. The NULL context is not included in this number.

The maximum number of contexts may be calculated by a MG in different ways, e.g. dependent on the number of to be supported connection models, dependent on applied stream, termination and/or context admission control functions, or specific, MG-internal QoS support mechanisms (e.g. best effort versus over-provisioning models), or other reasons.

The value of the property must be therefore interpreted as a best-case figure. Requests for additional contexts over and above this number may receive an error response.

Type: double

Possible values: 1 and up

Defined in: TerminationState

Characteristics: read only

[End Correction]

6.43 Clarification on ABNF comment update for indAudTerminationStateParm

Description:	There is still an oversight in v2 regarding termination state audit.
	`indAudterminationStateDescriptor = TerminationStateToken LBRKT`
	`indAudterminationStateParm RBRKT`
	`; at-most-once per item`
	`indAudterminationStateParm = (pkgdName /`
	`ServiceStatesToken / BufferToken)`
	"at-most-once" is superfluous as one instance of indAudterminationStateParm is allowed. The at most once statement could be removed without causing as syntax or backwards compatibility problems.

Reference:	C-277

[Begin Correction]

B.2 ABNF specification

…

```
indAudterminationStateDescriptor = TerminationStateToken LBRKT
                                   indAudterminationStateParm RBRKT
```

~~; at-most-once per item~~

```
indAudterminationStateParm =(pkgdName / ServiceStatesToken / BufferToken)
```

[End Correction]

6.44 Clarification for command responses

Description:	TransactionID in §6.3.4 should be TerminationID
Reference:	C.292

[Begin Correction]

6.3.4 Wildcarded responses

…

If an error occurs during the execution of a wildcarded request that specifies a wildcarded response, special handling is required to provide useful information about the error(s) while still maintaining a modest sized response. When a wildcarded response is requested, all instances (as specified above) of the command shall be executed even if one or more result in errors, but later commands in the transaction will not be executed (unless optional was specified). Multiple command responses shall be returned for the command that encountered the error. The first command response shall be the normal wildcard response containing the UNION of responses for those commands that succeeded. If none of them succeeded the UNION shall be empty. Additional command responses for each ~~transactionID~~ TerminationID that failed shall be returned with the appropriate Error Descriptor.

[End Correction]

6.45 Clarification for ServiceChange at failure of an MGC

Description:	Method in §11.5 should be command
Reference:	C.292

[Begin Correction]

6.45.1 11.5 Failure of an MGC

…

In partial failure, or for manual maintenance reasons, an MGC may wish to direct its controlled MGs to use a different MGC. To do so, it sends a ServiceChange ~~method~~ Command to the MG with a "HandOff" method, and its designated replacement in ServiceChangeMgcId. If "HandOff" is supported, the MG shall send a ServiceChange message with a "Handoff" method and a "MGC directed change" reason to the designated MGC. If it fails to get a reply from the designated MGC, the MG shall behave as if its MGC failed, and start contacting secondary MGCs as specified in the previous paragraph. If the MG is unable to establish a control relationship with any MGC, it shall wait a random amount of time as described in 9.2 and then start contacting its primary, and if necessary, its secondary MGCs again.

[End Correction]

6.46 Clarification for Text OctetString encoding

Description:	There has been some confusion over the encoding of binary octets using the text encoding of H.248. The rules for hexadecimal encoding of octets is defined in H.248.1 Annex B.3. These rules result in the ordering of the binary octet being reversed. Examples are given however this reversal may not be apparent at first glance. Thus it is proposed to add a note to this effect.
Reference:	C.266

[Begin Correction]

6.46.1 B.3 Hexadecimal octet coding

Hexadecimal octet coding is a means for representing a string of octets as a string of hexadecimal digits, with two digits representing each octet. This octet encoding should be used when encoding octet strings in the text version of the protocol.

For each octet, the 8-bit sequence is encoded as two hexadecimal digits. Bit 0 is the first transmitted; bit 7 is the last.

Bits 7-4 are encoded as the first hexadecimal digit, with Bit 7 as MSB and Bit 4 as LSB. Bits 3-0 are encoded as the second hexadecimal digit, with Bit 3 as MSB and Bit 0 as LSB.

Note: The above encoding results in the reversal of bits from the original octet.

Examples:

Octet bit pattern	**Hexadecimal coding**
00011011	D8
11100100	27
10000011 10100010 11001000 00001001	C1451390

[End Correction]

6.47 ALL Wildcard in Topology Descriptor

Description:	The semantic for ALL is related to TerminationIDs, thus outlined by clause 6.2.2/H.248.1 ("…The two wildcards are ALL and CHOOSE. The former is used to address multiple terminations at once, while the latter is used to indicate to a media gateway that it must select a termination satisfying the partially specified TerminationID."). The Topology Descriptor is applied on Context level, wildcard "ALL" could basically mean either A. the set of "all" Terminations {Ta, Tb, … Tn} of the Context, or B. the sub-set of all Terminations without the Termination "T1" and/or "T2". Only semantic "B" is meaningful with regards to "topology associations" between terminations within a context. The handling of the topology descriptor when adding a termination to a Context is further clarified.
Reference:	AVD-3204 & AVD-3205

[Begin Correction]

[Begin Proposal]

7.1.18 Topology Descriptor

7.1.18.1 General

A Topology Descriptor is used to specify flow directions between terminations in a context. Contrary to the descriptors in previous clauses, the Topology Descriptor applies to a context instead of a termination. The default topology of a context is that each termination's transmission is received by all other terminations. When a Termination is added to a Context its default Topology is "bothway" to the other Terminations in the Context. If another Topology is required this shall be indicated in any updated Topology Descriptor. The Topology Descriptor is optional to implement. An MG that does not support Topology Descriptors, but receives a command containing one, returns Error Code 444 ("Unsupported or unknown descriptor"), and optionally includes a string containing the name of the unsupported descriptor ("Topology") in the error text in the Error Descriptor.

The Topology Descriptor occurs before the commands in an action. It is possible to have an action containing only a Topology Descriptor, provided that the context to which the action applies already exists.

7.1.18.2 Structure (syntax) of the Topology Descriptor

A Topology Descriptor consists of a sequence of associated terminations of the form (*T1*, *T2*, *association[,StreamID]*). *T1* and *T2* specify terminations within the context, possibly using the ALL or CHOOSE wildcard. If the optional StreamID field is used, the association applies only to the particular stream between *T1* and *T2* labeled by the StreamID. If the StreamID field is omitted, the topology applies to all streams in the termination.

7.1.18.3 Descriptor element "association"

The *association* specifies how media flows between these two terminations as follows:

- (*T1*, *T2*, Isolate) means that the terminations matching *T2* do not receive media from the terminations matching *T1*, nor vice versa.
- (*T1*, *T2*, Oneway) means that the terminations that match *T2* receive media from the terminations matching *T1*, but not vice versa. In this case, use of the ALL wildcard such that there are terminations that match either *T1* or *T2* *but not both* is allowed.
- (*T1*, *T2*, OnewayExternal) means the terminations that match *T2*, receive media sent externally by terminations matching *T1*, but not vice versa. In this case, use of the ALL wildcard for *T1* is not allowed.
- (*T1*, *T2*, OnewayBoth) means the terminations that match *T2*, receive media sent and received externally by terminations matching *T1*, but not vice versa. In this case, use of the ALL wildcard for *T1* and/or *T2* is not allowed.
- (*T1*, *T2*, Bothway) means that the terminations matching *T2* receive media from the terminations matching *T1*, and vice versa. In this case it is allowed to use wildcards such that there are terminations that match both *T1* and *T2*. However, if there is a termination that matches both, no loopback is introduced.

7.1.18.4 Wildcarding of TerminationID elements

7.1.18.4.1 Wildcard CHOOSE

CHOOSE wildcards may be used in *T1* and *T2* as well, under the following restrictions:

- the action (see clause 8) of which the Topology Descriptor is part contains an Add Command in which a CHOOSE wildcard is used;
- if a CHOOSE wildcard occurs in *T1* or *T2*, then a partial name (underspecified Termination ID) shall not be specified.

The CHOOSE wildcard in a Topology Descriptor matches the TerminationID that the MG assigns in the first Add Command that uses a CHOOSE wildcard in the same action. An existing termination that matches *T1* or *T2* in the context to which a termination is added is connected to the newly added termination as specified by the Topology Descriptor. If a termination is not mentioned within a Topology Descriptor, any topology associated with it remains unchanged. If, however, a new termination is added into a context, its association with the other terminations within the context defaults to Bothway, unless a Topology Descriptor is given to change this (e.g., if *T3* is added to a context with *T1* and *T2* with topology (*T3*, *T1*, Oneway) it will be connected Bothway to *T2*).

7.1.18.4.2 Wildcard ALL

ALL wildcard may be used in *T1 or T2* as well, under the following restrictions:

- "all" excludes the other termination in the Topology Descriptor (this semantic is in-line with clause 6.2.2);

ALL wildcard may be not used for both T1 and T2 in a single descriptor with a one way association.

7.1.18.4.3 Combination of Wildcard ALL and CHOOSE in a single Topology Descriptor

ALL wildcard may be used in *T1 and* CHOOSE wildcard may be used in *T2*, or vice versa. The *choosen* termination is then excluded from the "*all*" set of the remaining terminations.

7.1.18.5 Toplogies on Steam-level

If the topology is applied to one particular stream (*T1*, *T2*, association, StreamID), the topology of other streams between the terminations does not change.

A Topology Descriptor shall not include a combination of associations between two terminations (Ti, Tj) with and without the optional StreamID field, to avoid undefined behaviour. For example (*T1*, *T2*, Bothway) and (*T1*, *T2*, Isolate, S1) shall not appear in the same descriptor. Upon receipt of such a Topology Descriptor, a MG shall respond with an error response, including Error Code 421 ("Unknown action or illegal combination of actions").

~~A oneway connection must be implemented in such a way that the other terminations in the context are not aware of the change in topology.~~

7.1.18.6 Example topologies

Figure 7, the table following it and Figure 8 following it show some examples of the effect of including Topology Descriptors in actions. In these examples it is assumed that the Topology Descriptors are applied in sequence. Figures 9 and 10 are stand-alone examples showing the specific effects of the OnewayExternal and OnewayBoth topology settings.

…

[End Correction]

6.48 Octet String usage in Annex B.2

Description:

H.248 package definitions in terms of Properties, Events and Signals descriptor parameters all based on:

```
propertyParm     = pkgdName parmValue
eventOther       = eventParameterName parmValue
sigOther      = sigParameterName parmValue
parmValue     = (EQUAL alternativeValue / INEQUAL VALUE)
VALUE       = quotedString / 1*(SafeChar / %x80-FF)
```

However packages specify data types to be used for properties, events and signals descriptor parameters. In some cases parameters are defined as “octet string” as well. In H.248 B.2 “octet string” is currently defined as:

```
octetString     = *(nonEscapeChar)
```

and the usage of “octet string” is defined to be related to SDP defined in RFC2327 used within H.248 Local and Remote Descriptors.

The distinction between the two needs to be specified.

Reference: AVD-3206

[Begin Correction]

B.2 ABNF specification

…

```
; Boolean:  Boolean values are encoded as "on" and "off" and are
; case insensitive.  The SafeChar form of VALUE must be used.
; OctetString: Where a package specifies the data type –“OctetString” the
; hexadecimal form defined in B.3 shall be used. Note: the OctetString BNF
; is only for the carriage of SDP in the Local and Remote Descriptors.
;
; Future types:  Any defined types must fit within
; the ABNF specification of VALUE.  Specifically, if a type's encoding
```

…

[End Correction]

6.49 H.248 Data type ServiceChangeMgcID

Description:

The ServiceChangeMgcID is used to identify an “alternate address” for a MGC entity. This implies an interim address resolution step in case that the parameter format is using a *name* format, i.e. either a domain name or a device name. The current H.248.1 text is not explicit on that aspect and only mentions the *address* format.

Reference: AVD-3207

[Begin Correction]

▪ 7.2.8 Servicechange

…

The optional ServiceChangeAddress parameter specifies the address (e.g., IP port number for IP networks) to be used for subsequent communications. It can be specified in the input parameter descriptor or the returned result descriptor.

The optional ServiceChangeMgcID parameter is of type *Message Identifier* (MID, see clause 8.3), the parameter format represents thus either an *address* (IP version 4 or 6 *domain address*, or broadband *MTP3 address*) or a *name* (IP *domain name* or a generic *device name*). In both cases is the parameter used for an unambiguous identification of an MGC entity (i.e. a primary or secondary MGC).

NOTE - There is following difference between name and address format: the network address is routable, thus may be directly inserted as destination address in the signalling transport protocol data unit, whereas names requiring firstly a resolution into a routable address. The name-to-address resolution by the MG requires a local or remote DNS query request in case of the domain format, or a local mapping table in case of the device format.

ServiceChangeAddress and ServiceChangeMgcID parameters must not both be present in the ServiceChange Descriptor or the ServiceChangeResult Descriptor. The ServiceChangeAddress provides an address to be used within the context of the association currently being negotiated, while the ServiceChangeMgcID provides an *alternate address* (NOTE: in case of a name format is firstly a name-to-address resolution implied, see above) where the MG should seek to establish another association. MGCs and MGs must be able to cope with the ServiceChangeAddress being either a full address or just a port number in the case of IP-based transports such as UDP, TCP or SCTP.

…

[End Correction]

6.50 Text encoding of Octet Strings

Description:	The coding of Octet String in Annex B.3 can be seen as confusing on transmission order versus the bit sending order. The should be further clarified.
Reference:	AVD-3313

[Begin Correction]

B.3 Hexadecimal octet coding

Hexadecimal octet coding is a means for representing package elements of type Octet String as a string of hexadecimal digits, with two digits representing each octet. This octet encoding should be used when encoding values of type Octet String in the text version of the protocol.

For each octet, the 8-bit sequence is encoded as two hexadecimal digits. Bit 0 is the first transmitted or left-most; bit 7 is the last or right-most.

Note: this means that Bit 0 is the MSB of the original octet and Bit 7 is the LSB of the original octet, according to the common convention used to specify octet values.

Bits 7-4 are encoded as the first hexadecimal digit, with Bit 7 as MSB of the first hexadecimal digit and Bit 4 as LSB of the first hexadecimal digit.. Bits 3-0 are encoded as the second hexadecimal digit, with Bit 3 as MSB of the second hexadecimal digit and Bit 0 as LSB of the second hexadecimal digit.

Note: The above encoding results in the reversal of bits from the original octet.

…

[End Correction]

6.51 Profile definition - Syntactical vs Semantical Profiles

Description:	It was not clear what the behaviour of Profile Specification should be with regards to extending and profiling the H.248 core protocol and its associated packages. This needs to be further clarified.
Reference:	AVD-3210

[Begin Correction]

---

13 Profile definition

Profiles may be specified to further define how the H.248.1 protocol is used and what functionality is supported by a MG. It only describes the capabilities of the MGC/MG H.248 interface. The profile itself specifies what options associated with H.248.1 have been used. For example: transport and packages used for an application.

Any profile definition shall not extend the H.248.1 protocol by e.g. adding new syntactical elements, revision of semantics, or additional procedures (like e.g. cross-package level procedures) in a profile. Possible profile procedures are thus given only by possible core protocol level (1) and package level (2) procedures.

A profile is identified by a name (IANA registered) and a version. A name shall be a case-insensitive string up to 64 characters long. Version shall be 1 to 99.

The profile itself is a document that indicates the options for a particular application. There is no set format for this document. The only mandatory element is that there should be a section indicating the Profile Name and Version and a summary of the profile.

The two points below are the only mandatory sections of a profile:

- Profile Identification: The name and version of the profile that is sent in the ServiceChange Command.
- Summary: A description of what the profile is.

Appendix III contains a template for the definition of profiles. It should be used as the basis of a profile definition.

[End Correction]

6.52 Example Call Flow updates

Description:	Some bracket types in the example call flows or Appendix I are not consistent with the protocol syntax for text encoding. Also is some cases the Streammode should be set to Inactive when the termination is in the NULL context.
Reference:	AVD-3211 & AVD-3357

[Begin Correction]

Appendix I
Example call flows

…

3) The MGC programs a termination in the NULL Context. The TerminationID is A4444, the streamID is 1, the requestID in the Events Descriptor is 2222. The MID is the identifier of the sender of this message; in this case, it is the IP address and port [123.123.123.4]:55555. The Mode Property for this stream is set to Inactive. "al" is the analog line supervision package. The Local and Remote Descriptors are assumed to be provisioned.

```
MGC to MG1:
MEGACO/3 [123.123.123.4]:55555
Transaction = 9999 {
    Context = - {
        Modify = A4444 {
            Media { Stream = 1 {
                     LocalControl {
                         Mode = Inactive,
                         tdmc/gain=2,  ; in dB,
                         tdmc/ec=on
                     },
                 }
            },
            Events = 2222 {al/of {strict=state}}
        }
    }
}
```

…

I.1.2 Collecting originator digits and initiating termination

The following builds upon the previously shown conditions. It illustrates the transactions from the Media Gateway Controller and originating Media Gateway (MG1) to get the originating termination (A4444) through the stages of digit collection required to initiate a connection to the terminating Media Gateway (MG2).

6) MG1 detects an off-hook event from User 1 and reports it to the Media Gateway Controller via the Notify Command.

```
MG1 to MGC:
MEGACO/3 [124.124.124.222]:55555
Transaction = 10000 {
   Context = - {
       Notify = A4444 {ObservedEvents =2222 {
         19990729T22000000:al/of{init=OFF}}}
   }
}
```

…

8) The MGC modifies the termination to play dial tone, to look for digits according to Dialplan0 and to look for the on-hook event now.

```
MGC to MG1:
MEGACO/3 [123.123.123.4]:55555
Transaction = 10001 {
    Context = - {
        Modify = A4444 {
            Events = 2223 {
                al/on{strict=state}, dd/ce {DigitMap=Dialplan0}
            },
            Signals {cg/dt},
            DigitMap= Dialplan0{
(0| 00|[1-7]xxx|8xxxxxxx|Fxxxxxxx|Exx|91xxxxxxxxxx|9011x.)}
        }
    }
}
```

…

12) The controller then analyses the digits and determines that a connection needs to be made from MG1 to MG2. Both the TDM termination A4444, and an RTP termination are added to a new context in MG1. Mode is RecvOnly since Remote Descriptor values are not yet specified. Preferred codecs are in the MGC's preferred order of choice.

```
MGC to MG1:
MEGACO/3 [123.123.123.4]:55555
Transaction = 10003 {
    Context = $ {
       Add = A4444{ LocalControl {
                      Mode = SendRecv} },
       Add = $ {
           Media {
             Stream = 1 {
                  LocalControl {
                      Mode = RecvOnly,

                      nt/jit=40 ; in ms
                  },
                  Local {
v=0
c=IN IP4 $
m=audio $ RTP/AVP 4
a=ptime:30
v=0
c=IN IP4 $
m=audio $ RTP/AVP 0
                  }
             }
          }
       }
    }
}
```

…

14) The MGC will now associate A5555 with a new context on MG2, and establish an RTP Stream (i.e., A5556 will be assigned), SendRecv connection through to the originating user, User 1. The MGC also sets ring on A5555.

```
MGC to MG2:
MEGACO/3 [123.123.123.4]:55555
Transaction = 50003 {
    Context = $ {
       Add = A5555  { Media {
            Stream = 1 {
                 LocalControl {Mode = SendRecv} }},
         Events=1234{al/of{strict=state}},
            Signals {al/ri}
            },
       Add  = $ {Media {
            Stream = 1 {
                 LocalControl {
                    Mode = SendRecv,
                    nt/jit=40 ; in ms
                 },
                 Local {
v=0
c=IN IP4 $
m=audio $ RTP/AVP 4
a=ptime:30
                 },
                 Remote {
v=0
c=IN IP4 124.124.124.222
m=audio 2222 RTP/AVP 4
a=ptime:30
                 } ; RTP profile for G.723.1 is 4
             }
          }
      }
   }
}
```

…

17) The two gateways are now connected and User 1 hears the ringback. The MG2 now waits until User2 picks up the receiver and then the two-way call is established.

```
From MG2 to MGC:

MEGACO/3 [125.125.125.111]:55555
Transaction = 50005 {
   Context = 5000 {
       Notify = A5555 {ObservedEvents =1234 {
         19990729T22020002:al/of{init=off}}}
   }
}

From MGC to MG2:

MEGACO/3 [123.123.123.4]:55555
Reply = 50005 {
    Context = - {Notify = A5555}
}

From MGC to MG2:

MEGACO/3 [123.123.123.4]:55555
Transaction = 50006 {
   Context = 5000 {
      Modify = A5555 {
         Events = 1235 {al/on{strict=state}},
         Signals ; to turn off ringing
      }
   }
}

From MG2 to MGC:

MEGACO/3 [125.125.125.111]:55555
Reply = 50006 {
 Context = 5000 {Modify = A4445}
}
```

…

21) When the MGC receives an on-hook signal from one of the MGs, it brings down the call. In this example, the user at MG2 hangs up first.

```
From MG2 to MGC:

MEGACO/3 [125.125.125.111]:55555
Transaction = 50008 {
   Context = 5000 {
       Notify = A5555 {ObservedEvents =1235 {
          19990729T24020002:al/on{init=off}}
       }
   }
}

From MGC to MG2:

MEGACO/3 [123.123.123.4]:55555
Reply = 50008 {
    Context = - {Notify = A5555}
}
```

…

[End Correction]

6.53 Multiplexing and Maximum number of terminations per context property

Description:	The Base Root package provides the `MaxTerminationsPerContext` property. Termination types which are counted are physical and ephemeral terminations, but what about "multiplexing terminations"? Multiplexing terminations should be included.
Reference:	AVD-3212

[Begin Correction]

E.2.1 Properties

MaxNrOfContexts

PropertyID: maxNumberOfContexts (0x0001)

The value of this property gives the maximum number of contexts that can exist at any time. The NULL context is not included in this number.

Type: double

Possible values: 1 and up

Defined in:

TerminationState

Characteristics: read only

MaxTerminationsPerContext

PropertyID: maxTerminationsPerContext (0x0002)

The maximum number of allowed terminations in a context, see clause 6.1 This is not related to the capacity of the NULL Context. This is related to non-Root termination types (physical, ephemeral), inclusive multiplexing terminations.

Type: integer

Possible values: any integer

Defined in: TerminationState

Characteristics: readonly

…

[End Correction]

6.54 Clarification of the behavior of sub-lists

Description:	There is an inconsistancy between the ASN.1 and Text encodings with regards to the behaviour of sub-lists. This should be rectified.
Reference:	AVD-3279

[Begin Correction]

A.2 ASN.1 syntax specification

…

```
-- In PropertyParm, value is a SEQUENCE OF octet string. When sent
-- by an MGC the interpretation is as follows:
-- empty sequence means CHOOSE
-- one element sequence specifies value
-- If the sublist field is not selected, a longer sequence means
-- "choose one of the values" (i.e., value1 or~~OR~~ value2 etc from a
-- collection of values).
-- If the sublist field is selected,
-- a sequence with more than one element encodes the value of a
-- list-valued property (i.e., a collection of values, value1 and value2
-- and ...).~~).~~
-- The relation field may only be selected if the value sequence
-- has length 1.  It indicates that the MG has to choose a value
-- for the property. E.g., x > 3 (using the greaterThan
-- value for relation) instructs the MG to choose any value larger
-- than 3 for property x.
-- The range field may only be selected if the value sequence
-- has length 2.  It indicates that the MG has to choose a value
-- in the range between the first octet in the value sequence and
-- the trailing octet in the value sequence, including the
-- boundary values.
-- When sent by the MG, only responses to an AuditCapability request
-- may contain multiple values, a range, or a relation field.
```

…

[End Correction]

[Begin Correction]

B.2 ABNF specification

…

```
propertyParm     = pkgdName parmValue

; the (Safe)Char '$' means CHOOSE
; the (Safe)Char '*' means ALL
parmValue    = (EQUAL alternativeValue / INEQUAL VALUE)

alternativeValue  = (VALUE
              / LSBRKT VALUE *(COMMA VALUE) RSBRKT
                 ; sublist                    / LBRKT VALUE *(COMMA VALUE)
RBRKT
                 ; alternatives (i.e., A OR B OR ...)
              / LSBRKT VALUE COLON VALUE RSBRKT)
                 ; range
; If the sublist field is not selected (alternative or range), a longer
; sequence means "choose one of the values" (i.e., value1 or value2 etc
; from a collection of values).
; If the sublist field is selected,
; a sequence with more than one element encodes the value of a
; list-valued property (i.e., a collection of values, value1 and
; value2 and ...).
```

…

[End Correction]

6.55 Orthogonal capabilities "Transcoding" and "Codec Negotiation/Determination"

Description:	Transcoding, i.e. the conversion of different media formats between two

	associated H.248 stream endpoints within a context, may be the result of either fully specified H.248 media descriptors (“single step resource reservation procedure”), or due to a previous media negotiation phase (“multi step resource reservation procedure”). The capability of transcoding is therefore disjoint to media negotiation principles.
Reference:	AVD-3280

[Begin Correction]

7.1.7 LocalControl descriptor

The LocalControl descriptor contains the Mode property, the ReserveGroup and ReserveValue properties and properties of a Termination (defined in Packages) that are stream specific, and are of interest between the MG and the MGC. Values of properties may be specified as in 7.1.1.

The allowed values for the mode property are send-only, receive-only, send/receive, inactive and loop-back. "Send" and "receive" are with respect to the exterior of the Context, so that, for example,a stream set to mode = sendOnly does not pass received media into the Context. The default valuefor the mode property is "Inactive". Signals and Events are not affected by mode.

The boolean-valued Reserve properties, ReserveValue and ReserveGroup, of a Termination indicate what the MG is expected to do when it receives a Local and/or Remote descriptor.

If the value of a Reserve property is True, the MG SHALL reserve resources for all alternatives specified in the Local and/or Remote descriptors for which it currently has resources available. It SHALL respond with the alternatives for which it reserves resources. If it cannot not support any of the alternatives, it SHALL respond with a reply to the MGC that contains empty Local and/or Remote descriptors. If media begins to flow while more than a single alternative is reserved, media packets may be sent/received on any of the alternatives and must be processed, although only a single alternative may be active at any given time.

If the value of a Reserve property is False, the MG SHALL choose one of the alternatives specified in the Local descriptor (if present) and one of the alternatives specified in the Remote descriptor (if present). If the MG has not yet reserved resources to support the selected alternative, it SHALL reserve the resources. If, on the other hand, it already reserved resources for the Termination addressed (because of a prior exchange with ReserveValue and/or ReserveGroup equal to True), it SHALL release any excess resources it reserved previously. Finally, the MG shall send a reply to
the MGC containing the alternatives for the Local and/or Remote descriptor that it selected. If the MG does not have sufficient resources to support any of the alternatives specified, it SHALL respond with error 510 (insufficient resources).

The default value of ReserveValue and ReserveGroup is False. More information on the use of the two Reserve properties is provided in 7.1.8.

Note: A *single set of property values* may consist of a single *media type* (e.g. audio or video) related *media format* complemented by a list of supplementary media formats. Supplementary media formats are for example:

- Comfort noise ([IETF RFC 3389]),

- RTP Payload for DTMF digits, telephony tones and telephony signals ([IETF RFC 4733]),

- Voiceband data (VBD) services (according to [ITU-T V.152]),
- or other auxiliary media (e.g. inband signaling) associated to the main media flow.

[End Correction]

6.56 Signal Type Override

Description:	When signals are defined in packages, a default type is also provided: Brief, TimeOut or OnOff. Still H.248.1 allows the MGC to change the type of a signal at the moment of activating it, by including the signal type in the SignalsDescriptor. The effect of changing the signal type is straightforward in most cases, most notably in H.248 signals that produce audible or visible media: tones, announcements, video, etc. However, there are H.248 signals whose effect is of instantaneous nature, rather than having an associated duration controlled by the MGC, and that are normally specified as type 'Brief' . Examples of these signals are the IPBCP tunnelling signal defined in Q.1950, or the latching signal defined in H.248.37. Changing the type of these signals, although allowed by H.248, may not have any effect in some cases without changing the semantics of the signal, and that is certainly not possible.
Reference:	AVD-3204

[Begin Correction]

7.1.11 Signals Descriptor

…

There are three types of signals:

- OnOff (OO): the signal lasts until it is turned off;
- TimeOut (TO): the signal lasts until it is turned off or a specific period of time elapses;
- Brief (BR): the signal will stop on its own unless a new Signals Descriptor is applied that causes it to stop; no timeout value is needed.

If a signal of default type other than TO has its type overridden to type TO in the Signals Descriptor, the duration parameter must be present.

~~NOTE: It is not possible to change the semantics of a signal by overriding the signal type.~~

If the signal type is specified in a Signals Descriptor, it overrides the default signal type (see clause 12.1.4). It is not possible to change the semantics of a signal by overriding the signal type. If duration is specified for an on/off signal, it shall be ignored.

…

[End Correction]

6.57 rtp/pltrans event clarification

Description:	When signals are defined in packages, a default type is also provided: Brief, TimeOut or OnOff. Still H.248.1 allows the MGC to change the type of a signal at the moment of activating it, by including the signal type in the SignalsDescriptor. The effect of changing the signal type is straightforward in most cases, most notably in H.248 signals that produce audible or visible media: tones, announcements, video, etc. However, there are H.248 signals whose effect is of instantaneous nature, rather than having an associated duration controlled by the MGC, and that are normally specified as type 'Brief' . Examples of these signals are the IPBCP tunnelling signal defined in Q.1950, or the latching signal defined in H.248.37. Changing the type of these signals, although allowed by H.248, may not have any effect in some cases without changing the semantics of the signal, and that is certainly not possible.
Reference:	AVD-3316

[Begin Correction]

…

E.12.5 Procedures

When RTCP is associated with an RTP stream, RTCP shall remain unaffected by the H.248.1 Mode Property in the LocalControl Descriptor.

When RTCP is associated with an RTP stream and the MG receives an Empty Remote Descriptor for that stream, the MG shall stop the RTCP stream along with the corresponding RTP stream.

Where the Payload Transition (*pltrans*) event is used the codec currently used by the relevant Stream on the MG is the start codec. In the case where a MG has not selected a codec for use it can be assumed that the initial codec is the first one listed in the Local descriptor (e.g. SDP). Therefore an MG receiving the *pltrans* event from the MGC in should consider the first codec listed as the "start codec". If the MG chooses another codec it should send a Notify.req with the *pltrans* observed event.

…

[End Correction]

6.58 The use of Ignore with SDP

Description:	Where SDP is used, some specifications indicate that if part of the SDP is not understood it is simply ignored. That is no error message is returned. SDP is used in the H.248 Local and Remote descriptors. Currently H.248 does not discuss this “Ignore” behavior. As a master/slave protocol H.248 always confirms any request with a success or failure indication. A failure usually halts execution unless the command is specified as “optional”. Therefore the use of the “Ignore” semantic is not appropriate as this does not fit the general H.248 behaviour. Also supporting an “Ignore” semantic would mean the process of Local and Remote descriptors would differ between the binary and text versions of the protocol as the binary version does not have an ignore semantic. The use of ignore should be clarified in the H.248.1 text.
Reference:	AVD-3316

[Begin Correction]

7.1.8 Local and remote descriptors

The MGC uses Local and Remote descriptors to reserve and commit MG resources for media decoding and encoding for the given Stream(s) and Termination to which they apply. The MG includes these descriptors in its response to indicate what it is actually prepared to support. The MG SHALL include additional properties and their values in its response if these properties are mandatory yet not present in the requests made by the MGC (e.g. by specifying detailed video encoding parameters where the MGC only specified the payload type).

Local refers to the media received by the MG and Remote refers to the media sent by the MG.

When text encoding the protocol, the descriptors consist of session descriptions as defined in SDP (RFC 2327). In session descriptions sent from the MG to the MGC, the SDP must comply with RFC 2327. In session descriptions sent from the MGC to the MG, the following exceptions to the syntax of RFC 2327 are allowed:

- the "s = ", "t = " and "o = " lines are optional;
- the use of CHOOSE is allowed in place of a single parameter value; and
- the use of alternatives is allowed in place of a single parameter value.

A Stream Descriptor specifies a single bidirectional media stream and so a single session description must not include more than one media description ("m = " line). A Stream Descriptor may contain additional session descriptions as alternatives. Each media stream for a termination must appear in distinct Stream Descriptors. When multiple session descriptions are provided in one descriptor, the "v = " lines are required as delimiters; otherwise they are optional in session descriptions sent to the MG. Implementations shall accept session descriptions that are fully conformant to RFC 2327 according to the above restrictions.

If a MG does not understand the SDP provided rather than ignoring that piece of SDP an appropriate error response shall be provided.

…

[End Correction]

6.59 Base packages and Extended Packages

Description:	The MG in response to a Command with wildcarded PackageID returns the PackageIDs of the Packages where the H.248 Package Elements (properties, signal, events and statistics) were originally defined. This results in the return of a mix of base and extended PackageIDs. However there is an issue on whether the MG would make this information available, as it is optional for the MG to publish this information. If it did not publish both the base and extended PackageIDs then the MGC may not be aware of the base PackageIDs. Likewise if we mandate that the MG should return a mix or base and extended PackageIDs then this means that the optional publishing of 6.2.3 is removed. This interaction needs to be specified.
Reference:	AVD-3382 & C.446

6.2.3 Packages

…

When packages are extended, the properties, events, signals and statistics defined in the base package can be referred to using either the extended package name or the base package name. For example, if Package A defines Event e1, and Package B extends Package A, then B/e1 is an event for a termination implementing Package B. By definition, the MG must also implement the base package, but it is optional to publish the base package as an allowed interface. If it does publish A, then A would be reported on the Packages Descriptor in AuditValue as well as B, and Event A/e1 would be available on a termination. If the MG does not publish A, then only B/e1 would be available. If published through AuditValue, A/e1 and B/e1 are the same event.

For improved interoperability and backward compatibility, an MG may publish all packages supported by its terminations, including base packages from which extended packages are derived. An exception to this is in cases where the base packages are expressly defined as "Designed to be extended only".

In the case that the MG publishes the base PackageIDs in response to a Packages Descriptor Audit, it will also respond with the PackageID of the H.248 Package Element where the element was originally defined in response to a wildcarded PackageID Audit.

In the case that the MG only publishes the extended PackageID in response to a Packages Descriptor Audit then the MG will respond to a wildcarded PackageID Audit using the extended H.248 PackageIDs.

Note: Whilst a Subtract request does not explicitly request an AuditValue.req of Statistics with a wildcarded PackageID, it is implicit in the command. Similarly, an audit of a complete descriptor is equivalent to an audit with a wildcarded PackageID. Thus the procedure described above will be used in both these cases.

[End Correction]

6.60 Connection Endpoint Naming Conventions

Description:	The terminology aspects of naming conventions for IP connection endpoint addresses has appeared in many discussions in past meetings. For instances in the work areas of H.248.30A1, H.Sup5, H.Sup7, H.248.37A1, H.248.48, H.248.43 or with regards to H.248 ServiceChange parameters ServiceChangeAddress and/or ServiceChangeMgcID (in case of IP based Control Associations). Furthermore there could be IP connectivity (for H.248 entities) in a) control plane: IP-based H.248 Control Association, and/or b) user/media plane: IP-based ephemeral Termination. The address naming issue is the same for both. The general assumption of address asymetry is also valid for both. Unfortunately terminology is not consistent concerning the naming conventions for endpoint addresses. This concept should be better explained.
Reference:	AVD-3214

[Begin Correction]

5 Conventions

5.1 Key words to indicate requirement levels

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

5.2 Connection endpoint naming conventions

5.2.1 Generic concept

There are basically four connection endpoints (concerning sources and sinks of traffic) in case of a bidirectional connection. H.248 uses a specific terminology for naming these endpoints *from the perspective of an H.248 entity*. Figure 1 provides a conceptual overview. The naming scheme is generic because it is technology-independent.

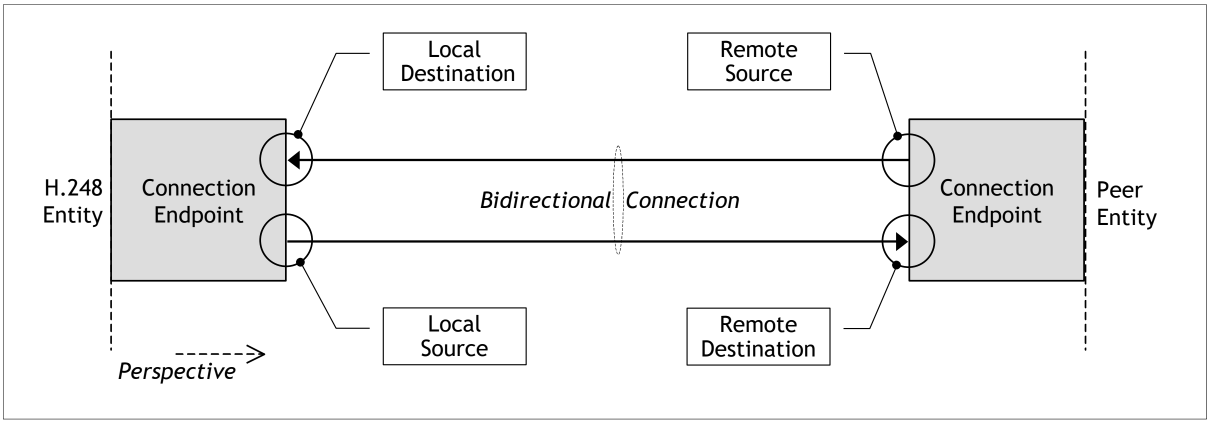

Figure 1 – Connection endpoint naming conventions – Generic concept

The *geographical locations* of the endpoints are termed as '*local*' (also known as '*near-end*') and as '*remote*' (also known as '*far-end*'). The *traffic directions* are termed as '*source*' (also known as ‘*egress*’, ‘*outgoing*’, ‘*outbound*’, ‘*transmit*’, or ‘*Tx*’) and as ‘*destination*’ (also known as ‘*ingress*’, ‘*incoming*’, ‘*inbound*’, ‘*receive*’, or ‘*Rx*’).

5.2.2 Scope of Local and Remote Descriptors

The *Local* and *Remote Descriptors* are defined in clause 7.1.8. Both descriptors have different connection endpoints in scope (see Figure 2). The *Local Descriptor* (LD) is related to the “*local destination*” endpoint, the *Remote Descriptor* (RD) is related to the “*remote destination*” endpoint. The media gateway is primarily responsible for the resources and configuration of the “*local source*” endpoint (Note: there are a few protocol elements which allow the MGC to control certain endpoint aspects). The “*remote source*” endpoint is completely out of scope of H.248 entities.

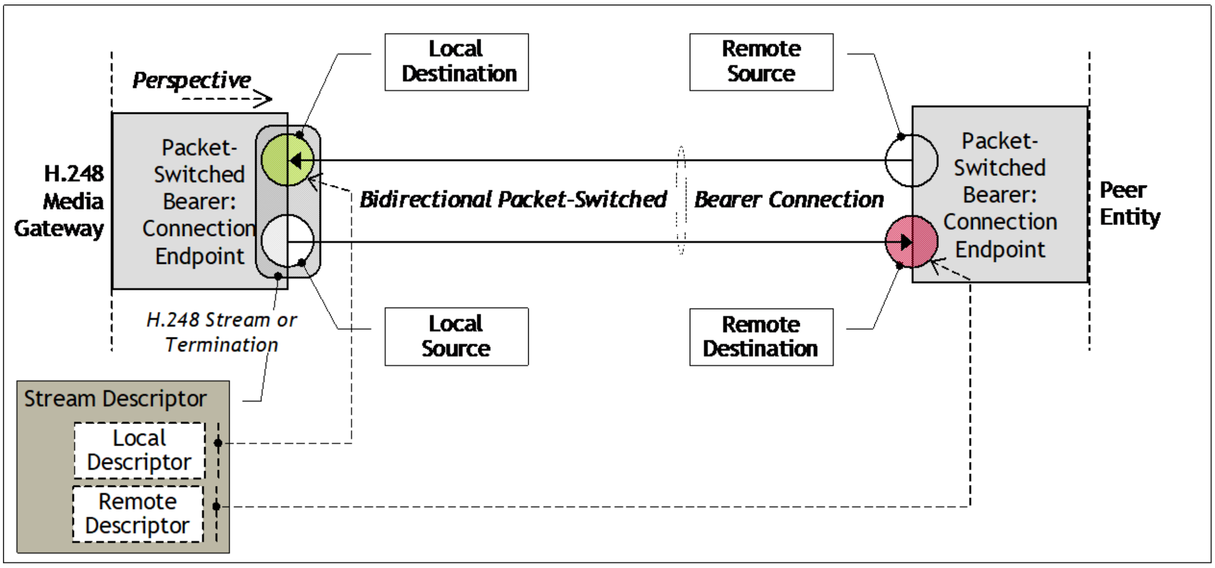

Figure 2 – Connection endpoint naming conventions – Scope of Local and Remote Descriptors

Figure 2 underlines that just the traffic sink endpoints of H.248 streams are controlled via the *Local* and *Remote Descriptors.*

5.2.3 Concrete “connection” types

An example shall illustrate the application of the naming conventions for a specific combination of a bearer technology and protocol layer.

5.2.3.1 Example “IP connection”

Figure 3 shows an IP connection, e.g. behind an H.248 IP stream or termination, or below an IP-based H.248 Control Association.

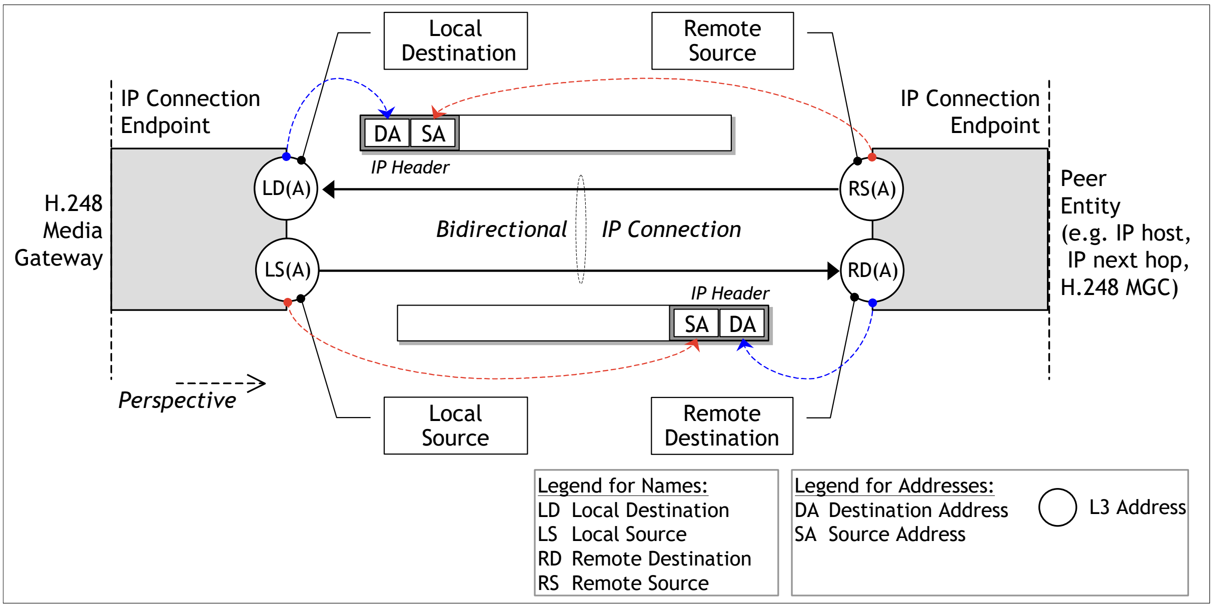

Figure 3 – Connection endpoint naming conventions – IP connection

The two local or remote connection endpoints may be assigned to the same or different IP interfaces. They are then so called *symmetrical* or *asymmetrical* (connection) addresses.

…

[End Correction]

6.61 Error in Context Audit example

Description:	Regarding the following example given in H.248.1v2, section 7.2.5: The command: Context=*{ContextAttr{ContextList={*}},AuditValue=Root{Audit{}}} returns: Context=*{ContextAttr{ContextList={1,2,3,4}},AuditValue=Root{}} When looking at the ABNF for auditOther, the {} cannot be empty and must contain an auditReturnParameter: auditOther = EQUAL termIDList [LBRKT terminationAudit RBRKT] terminationAudit = auditReturnParameter *(COMMA auditReturnParameter) Therefore the extraneous "{}" should be removed from the reply.
Reference:	Subject: Compact list of ContextIDs Date: Wed, 16 Apr 2008 16:59:46 +0100 From: Miri Epstein mepstein@juniper.net To: Christian Groves Christian.Groves@nteczone.com

	CC: megaco ietf <megaco@ietf.org>

[Begin Correction]

7.2.5 AuditValue

…

The command:

Context=*{ContextAttr{ContextList={*}},AuditValue=Root{Audit{}}}

returns:

Context=*{ContextAttr{ContextList={1,2,3,4}},AuditValue=Root~~{}~~}

The following illustrates other information that can be obtained with the AuditValue Command:

…

[End Correction]

6.62 Simple value as allowed as a sub-list of length one

Description:	There seems to be some lack of clarity regarding the encoding of sub-lists of length one. According to Annexes A.2/H.248.1 and B.2/H.248.1, it appears that in such a case, the single value must still be encoded as a sub-list. However recent package drafts (e.g. H.248.53) seem to imply that a sub-list of length one can be encoded as a simple value (e.g. without the brackets when using text encoding). This later approach has the advantage that future versions of a package may change the type of a property from X to sub-list of X without breaking backward compatibility.
Reference:	C.341 and C.381

[Begin Correction]

12.1.2 Properties

Properties defined by the package, specifying:

Property Name: only descriptive

PropertyID: is an identifier.

Description: is a description of the function of the property

Type: One of:

Boolean

String: UTF-8 string

Octet String: A number of octets. See Annex A and B.3 for encoding

Integer: 4-byte signed integer

Unsigned Integer: 4-octet unsigned integer

Double: 8-byte signed integer

Character: Unicode UTF-8 encoding of a single letter; could be more than one octet.

Enumeration: one of a list (NOTE 1) of possible unique values. Packages MUST define the text and binary encodings for each value in the enumeration.

Sub-list: a list of several values from a list (NOTE 1). The type of sub-list shall also be specified. The type shall be chosen from the types specified in this section (with the exception of sub-list). For example, Type: sub-list of enumeration. The encoding of sub-lists is specified in Annexes A.2 and in B.2.

NOTE 1 – A Sub-list may contain a single list item. See Annex A.2 and B.2 for the encoding options.

…

[End Correction]

[Begin Correction]

A.2 ASN.1 specification

…

```
-- In PropertyParm, value is a SEQUENCE OF octet string. When sent
-- by an MGC the interpretation is as follows:
-- empty sequence means CHOOSE
-- one element sequence specifies value
-- If the sublist field is not selected, a longer sequence means
-- "choose one of the values" (i.e., value1 OR value2 OR ...)
-- If the sublist field is selected,
-- a sequence with more than one element encodes the value of a
-- list-valued property (i.e., value1 AND value2 AND ...).
-- Note that when encoding a sub-list of length one, the sublist field
-- may be left unselected.
-- The relation field may only be selected if the value sequence
-- has length 1. It indicates that the MG has to choose a value
-- for the property. E.g., x > 3 (using the greaterThan
-- value for relation) instructs the MG to choose any value larger
-- than 3 for property x.
-- The range field may only be selected if the value sequence
-- has length 2. It indicates that the MG has to choose a value
-- in the range between the first octet in the value sequence and
-- the trailing octet in the value sequence, including the
-- boundary values.
-- When sent by the MG, only responses to an AuditCapability request
-- may contain multiple values, a range, or a relation field.
```

…

[End Correction]

[Begin Correction]

B.2 ABNF specification

…

```
; Boolean:  Boolean values are encoded as "on" and "off" and are
; case insensitive.  The SafeChar form of VALUE must be used.
;
; Sub-list:  Sub-lists are encoded using the "sublist" branch of
; alternativeValue. A sub-list with a length of one may also be encoded
; simply as a single VALUE of the appropriate type.
;
; Future types:  Any defined types must fit within
; the ABNF specification of VALUE.  Specifically, if a type's encoding
; allows characters other than SafeChars, the quotedString form must
; be used for all values of that type, even for specific values that
; consist only of SafeChars.
```

…

[End Correction]

6.63 Empty Statistics Descriptor

Description:	The text of clause 7.1.15/H.248.1 indicates that the MGC may send an empty Statistics Descriptor to the MG: "*The receipt of an empty descriptor means that no statistics shall be collected for the specified termination*". Sending an empty Statistics Descriptor is possible using the binary encoding of Annex A; but impossible using the text encoding of Annex B. The text encoding needs to be fixed to allow an empty Statistics Descriptor.
Reference:	Subject: RE: H.248.1 Sub-series IG for review Date: Sun, 27 Apr 2008 21:35:56 +0100 From: Elad Chomsky <elad@juniper.net>

[Begin Correction]

- B.2 ABNF specification

```
; Time = hhmmssss
Time       = 8(DIGIT)


statisticsDescriptor= StatsToken [LBRKT statisticsParameter
                     *(COMMAstatisticsParameter) RBRKT]


;at-most-once per item
statisticsParameter = pkgdName [EQUAL VALUE /
```

```
          (LSBRKT VALUE *(COMMA VALUE) RSBRKT)]
                          ...
```

[End Correction]

6.64 Incorrect RTP Profile ID

Description:	The introductory section of Appendix I "Example Call flows" of H.248.1 V3 (09/2005). There it says that "...For example, G.711 A-law is called PCMA in SDP, and is assigned profile 0. G.723.1 is called G723 and is profile 4; H.263 is called H263 and is profile 34. See also http://www.iana.org/assignments/rtp-parameters." The use of 0 for G.711 A-law is incorrect and should be 8.
Reference:	Subject: [Megaco] G.711 A-law assigned profile in H.248.1 Date: Tue, 26 Feb 2008 16:30:16 +0100 From: Beis, Grigoris (NSN - GR/Athens) <grigoris.beis@nsn.com> To: <megaco@ietf.org>

[Begin Correction]

Appendix I

Example call flows

All H.248.1 implementors must read the normative part of this Recommendation carefully before implementing from it. No one should use the examples in this appendix as stand-alone explanations of how to create protocol messages.

The examples in this appendix use SDP for encoding of the Local and Remote Descriptors. SDP is defined in RFC 2327. If there is any discrepancy between the SDP in the examples and RFC 2327, the RFC should be consulted for verification. Audio profiles used are those defined in RFC ~~1890~~3551, and others registered with IANA. For example, G.711 A-law is called PCMA in SDP, and is assigned profile ~~0~~8. G.723.1 is called G723 and is profile 4; H.263 is called H263 and is profile 34. See also http://www.iana.org/assignments/rtp-parameters.

…

[End Correction]

Annex: Defect Report Form for H.248.1 Version 2

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

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