

# PacketCable™ 1.5 Specification

## Audio Server Protocol

**PKT-SP-ASP1.5-I02-070412**

**ISSUED**

### **Notice**

This PacketCable specification is a cooperative effort undertaken at the direction of Cable Television Laboratories, Inc. (CableLabs®) for the benefit of the cable industry. Neither CableLabs, nor any other entity participating in the creation of this document, is responsible for any liability of any nature whatsoever resulting from or arising out of use or reliance upon this document by any party. This document is furnished on an AS-IS basis and neither CableLabs, nor other participating entity, provides any representation or warranty, express or implied, regarding its accuracy, completeness, or fitness for a particular purpose.

© Copyright 2004-2007 Cable Television Laboratories, Inc.

All rights reserved.

## Document Status Sheet

<b>Document Control Number:</b>	PKT-SP-ASP1.5-I02-070412			
<b>Document Title:</b>	Audio Server Protocol			
<b>Revision History:</b>	I01 – Issued January 28, 2005			
	I02 – Issued April 12, 2007			
<b>Date:</b>	April 12, 2007			
<b>Status:</b>	<del>Work in Progress</del>	<del>Draft</del>	Issued	Closed
<b>Distribution Restrictions:</b>	<del>Author Only</del>	<del>CL/Member</del>	<del>CL/ PacketCable/ Vendor</del>	Public

### Key to Document Status Codes:

<b>Work in Progress</b>	An incomplete document designed to guide discussion and generate feedback that may include several alternative requirements for consideration.
<b>Draft</b>	A document in specification format considered largely complete, but lacking reviews by Members and vendors. Drafts are susceptible to substantial change during the review process.
<b>Issued</b>	A stable document, reviewed, tested and validated, suitable to enable cross-vendor interoperability, and for certification testing.
<b>Closed</b>	A static document, reviewed, tested, validated, and closed to further engineering change requests to the specification through CableLabs.

### Trademarks

CableLabs®, DOCSIS®, EuroDOCSIS™, eDOCSIS™, M-CMTS™, PacketCable™, EuroPacketCable™, PCMM™, CableHome®, CableOffice™, OpenCable™, OCAP™, CableCARD™, M-Card™, and DCAS™ are trademarks of Cable Television Laboratories, Inc.

## Contents

<b>1</b>	<b>INTRODUCTION</b> .....	<b>1</b>
1.1	Purpose .....	1
1.2	Scope.....	1
1.3	Specification Language .....	2
<b>2</b>	<b>REFERENCES</b> .....	<b>4</b>
2.1	Normative.....	4
2.2	Informative .....	4
2.3	Reference Acquisition .....	4
<b>3</b>	<b>TERMS AND DEFINITIONS</b> .....	<b>5</b>
<b>4</b>	<b>ABBREVIATIONS AND ACRONYMS</b> .....	<b>9</b>
<b>5</b>	<b>TECHNICAL OVERVIEW</b> .....	<b>16</b>
<b>5.1</b>	<b>Architectural Requirements</b> .....	<b>16</b>
5.1.1	Call Destination .....	16
5.1.2	Media Formats.....	16
5.1.3	Security.....	16
5.1.4	Operational Support Systems.....	16
<b>5.2</b>	<b>Announcement Definitions</b> .....	<b>16</b>
5.2.1	Tones.....	16
5.2.2	Fixed-content Announcements.....	17
5.2.3	Variable Content Announcements.....	17
5.2.4	Interactive Announcements .....	17
5.2.5	Naming Conventions for Endpoint Identifiers .....	17
<b>5.3</b>	<b>Interfaces</b> .....	<b>17</b>
<b>6</b>	<b>ANN-1 INTERFACE: CMS-MTA AND MGC-MG</b> .....	<b>19</b>
<b>6.1</b>	<b>CMS-MTA Interface</b> .....	<b>19</b>
6.1.1	Announcement List.....	19
<b>6.2</b>	<b>MGC-MG Interface</b> .....	<b>20</b>
<b>7</b>	<b>ANN-2 INTERFACE: MPC-MP</b> .....	<b>21</b>
<b>7.1</b>	<b>Introduction</b> .....	<b>21</b>
<b>7.2</b>	<b>Audio Package Concepts</b> .....	<b>21</b>
7.2.1	Understanding Audio Segments.....	22
7.2.2	Segment Identifiers.....	22
7.2.3	Segment Life .....	23
7.2.4	Nested Sets And Sequences.....	23
7.2.5	Sequence Example .....	23
7.2.6	Set Example .....	23
7.2.7	Set With Nested Sequence Example .....	24
<b>7.3</b>	<b>Base Audio Package</b> .....	<b>24</b>
7.3.1	Abstract .....	24

7.3.2	Events.....	25
7.3.3	Signal Interactions .....	25
7.3.4	Parameters .....	26
7.3.5	Type-ahead .....	30
7.3.6	Return Parameters .....	30
7.3.7	Segment Descriptors .....	33
7.3.8	Variable Syntax .....	33
7.3.9	Variable Definitions.....	34
7.3.10	Timers.....	36
7.3.11	Examples.....	36
<b>7.4</b>	<b>Advanced Audio Package .....</b>	<b>38</b>
7.4.1	Abstract .....	38
7.4.2	Sets .....	38
7.4.3	Selectors.....	38
7.4.4	Selector Encoding .....	39
7.4.5	Variable Order .....	39
7.4.6	Overrides .....	39
7.4.7	Parameters .....	40
7.4.8	Return Codes .....	40
7.4.9	Examples.....	40
<b>7.5</b>	<b>Overview .....</b>	<b>41</b>
7.5.1	Speech Recognition Extensions to the BAU package.....	41
7.5.2	Signal Interactions .....	42
7.5.3	Parameters .....	42
7.5.4	Type-ahead .....	44
7.5.5	Return Parameters .....	45
7.5.6	Grammar Descriptors .....	46
7.5.7	Examples.....	46
<b>7.6</b>	<b>Formal Syntax Description.....</b>	<b>46</b>
<b>APPENDIX A</b>	<b>CALL FLOW FOR NETWORK ANNOUNCEMENT .....</b>	<b>52</b>
A.1	Call Flow Details.....	54
<b>APPENDIX B</b>	<b>CALL FLOW FOR MTA-STORED ANNOUNCEMENT .....</b>	<b>67</b>
B.1	Call Flow Details.....	67
<b>APPENDIX C</b>	<b>FUTURE CONSIDERATIONS .....</b>	<b>71</b>
<b>APPENDIX D</b>	<b>ACKNOWLEDGEMENTS.....</b>	<b>72</b>
<b>APPENDIX E</b>	<b>REVISION HISTORY .....</b>	<b>73</b>

## Figures

Figure 1. PacketCable™ Network Component Reference Model..... 2  
Figure 2. PacketCable Audio Server Components and Interfaces..... 18  
Figure 3. Sequence Example..... 23  
Figure 4. Set Example..... 24  
Figure 5. Set With Nested Sequence Example..... 24  
Figure 6. Call Flow for Network Announcement..... 53  
Figure 7. Call Flow for MTA-Stored Announcement ..... 67

## Tables

Table 1. MP Storage .....	19
Table 2. Sample Announcements .....	20
Table 3. Example URIs .....	22
Table 4. Events .....	25
Table 5. Parameters.....	26
Table 6. Return Parameters.....	31
Table 7. Return Codes.....	32
Table 8. Segment Descriptors.....	33
Table 9. Variable Types and Subtypes .....	34
Table 10. Sample Currency Codes .....	35
Table 11. Sample Language Codes.....	38
Table 12. Example URIs .....	39
Table 13. Parameters.....	40
Table 14. Return Codes .....	40
Table 15. S1 Events.....	41
Table 16. S2. Parameters .....	42
Table 17. S3. Return Parameters .....	45
Table 18. S4. Return Codes.....	45
Table 19. S5. Grammar Descriptors .....	46

# 1 INTRODUCTION

## 1.1 Purpose

This specification describes the architecture and protocols that are required for playing announcements in voice-over-IP (VoIP) PacketCable networks. Cable Television Laboratories (CableLabs®) has issued this specification to facilitate design and field-testing leading to the manufacture and interoperability of conforming hardware and software by multiple vendors. The will be referred to as the PacketCable Audio Server Specification.

Announcements are typically needed for calls that do not complete. Additionally, they may be used to provide enhanced information services to the caller (e.g., calling card, N11 services). Different carrier service feature sets require different announcement sets and announcement formats.

Announcements can be as basic as fixed-content announcements (e.g., all circuits busy) or as complex as those provided by intelligent IVR (Interactive Voice Response) systems. The PacketCable service model requires that all announcements be provisioned and signaled in a standard manner for all supported call features and use case scenarios.

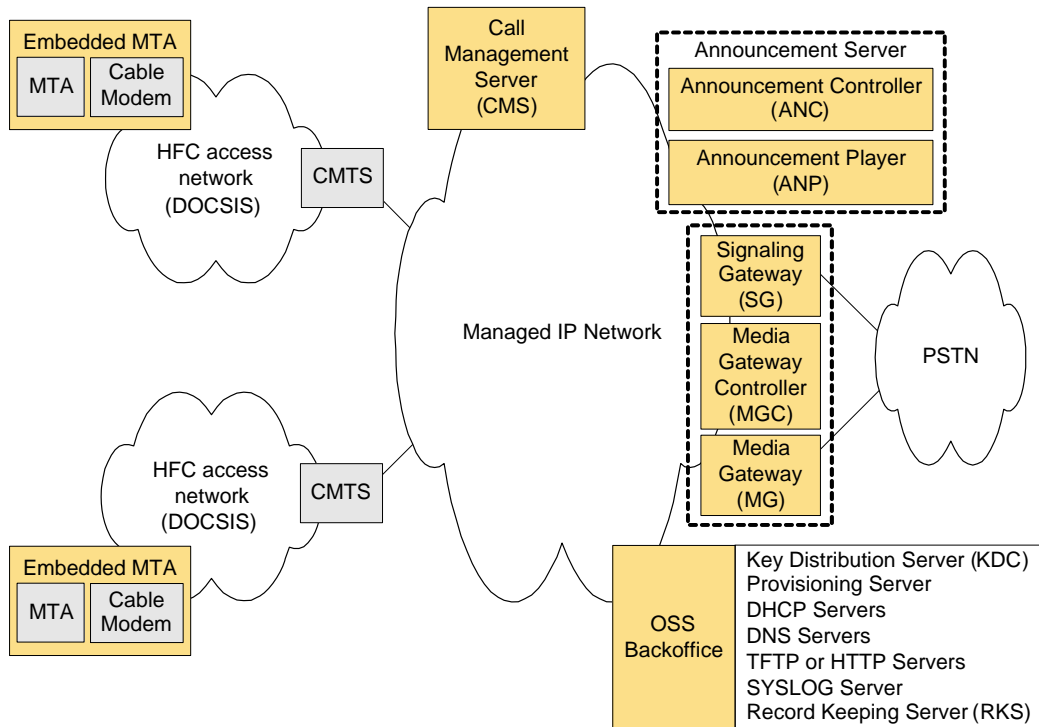
This specification defines a set of signaling protocols that are used to provide announcement services within a cable network. For one of these protocols, the PacketCable Network Call Signaling (NCS) protocol, this specification defines two new event packages:

- A Base Audio Package
- An Advanced Audio Package

## 1.2 Scope

PacketCable is a set of protocols developed to deliver enhanced communications services using packetized data transmission technology to a consumer's home over the cable network. The "PacketCable Architecture Framework" (PKT-ARCH-I01-991201) is the starting point for understanding PacketCable Interface Specifications, Technical reports, and other PacketCable documents.

The reference architecture for the PacketCable Network is shown in Figure 1 below.



**Figure 1. PacketCable Network Component Reference Model**

Announcement Servers, also known as Audio Servers are network components that manage and play informational tones and messages in response to events that occur in the network. Most announcements are media streams that originate from servers in the network. Some simple tones and short announcements can also reside at the MTA and in the MG.

### 1.3 Specification Language

Throughout this document words that are used to define the significance of particular requirements are capitalized. These words are:

"MUST"	This word or the adjective "REQUIRED" means that the item is an absolute requirement of this specification.
"MUST NOT"	This phrase means that the item is an absolute prohibition of this specification.
"SHOULD"	This word or the adjective "RECOMMENDED" means that there may exist valid reasons in particular circumstances to ignore this item, but the full implications should be understood and the case carefully weighed before choosing a different course.
"SHOULD NOT"	This phrase means that there may exist valid reasons in particular circumstances when the listed behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
"MAY"	This word or the adjective "OPTIONAL" means that this item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because it enhances the product, for example; another vendor may omit the same item.



Other text is descriptive or is explanatory.

The legal/regulatory classification of IP-based voice communications provided over cable networks and otherwise, and the legal/regulatory obligations, if any, borne by providers of such voice communications, are not yet fully defined by appropriate legal and regulatory authorities. Nothing in this specification is addressed to, or intended to affect, those issues. In particular, while this document uses standard terms such as "call," "call signaling," "telephony," etc., it will be evident from this document that while a PacketCable network performs activities analogous to these PSTN functions, the manner by which it does so differs considerably from the manner in which they are performed in the PSTN by telecommunications carriers. These differences may be significant for legal/regulatory purposes.

## 2 REFERENCES

### 2.1 Normative

In order to claim compliance with this specification, it is necessary to conform to the following standards and other works as indicated, in addition to the other requirements of this specification. Notwithstanding, intellectual property rights may be required to use or implement such normative references.

- [1] PacketCable 1.5 Network-Based Call Signaling Protocol Specification, PKT-SP-NCS1.5-I03-070412, April 12, 2007, Cable Television Laboratories, Inc.
- [2] IETF RFC 3435, Media Gateway Control Protocol (MGCP), January, 2003.
- [3] PacketCable 1.5 Audio/Video CODECS Specification, PKT-SP-CODEC1.5-I02-070412, April 12, 2007, Cable Television Laboratories, Inc.
- [4] PacketCable 1.5 Security Specification, PKT-SP-SEC- I02-070412, April 12, 2007, Cable Television Laboratories, Inc.

### 2.2 Informative

- [5] PacketCable 1.5 Architecture Framework Technical Report, PKT-TR-ARCH-1.5-V02-070412, April 12, 2007, Cable Television Laboratories, Inc.
- [6] PacketCable 1.5 PSTN Gateway Call Signaling Protocol Specification, PKT-SP-TGCP1.5-I03-070412, April 12, 2007, Cable Television Laboratories, Inc.
- [7] PacketCable 1.5 Event Messages, PKT-SP-EM1.5-I02-070412, April 12, 2007, Cable Television Laboratories, Inc.
- [8] IETF RFC 2396, Uniform Resource Identifiers (URI): Generic Syntax, August 1998.
- [9] IETF RFC 2234, Augmented BNF for Syntax Specifications: ABNF, November 1997.
- [10] ISO 639-2, Code For The Representation Of Names Of Languages, 1998.
- [11] ISO 4217, Codes for the Representation of Currencies and Funds, 2001.
- [12] ISO 8601, Representation of Dates and Times, 2000.
- [13] Sun Microsystems, Java Speech Grammar Format Specification, [JSGF], Copyright 1998-99.
- [14] Hunt, McGlashan, Speech Recognition Grammar Specification for the W3C Speech Interface Framework, [W3C], March 2004.

### 2.3 Reference Acquisition

- Cable Television Laboratories, Inc., 858 Coal Creek Circle, Louisville, CO 80027; Phone +1-303-661-9100; Fax +1-303-661-9199; Internet: [www.packetcable.com/](http://www.packetcable.com/)
- International Organization for Standardization (ISO) 1, rue de Varembé, Case postale 56, CH-1211 Geneva 20, Switzerland, Phone 41-22-749-01-11; Fax 41-22-733-34-30, Internet: [www.iso.org/](http://www.iso.org/)
- Internet Engineering Task Force (IETF) Secretariat c/o Corporation for National Research Initiatives, 1895 Preston White Drive, Suite 100, Reston, VA 20191-5434, Phone +1-703-620-8990, Fax +1-703-620-9071, Internet: [www.ietf.org/](http://www.ietf.org/)
- Sun Microsystems, Internet: [java.sun.com/products/java-edia/speech/forDevelopers/JSGF/JSGF.html](http://java.sun.com/products/java-edia/speech/forDevelopers/JSGF/JSGF.html)
- [W3C], Internet: <http://www.w3.org/TR/speech-grammar/>

### 3 TERMS AND DEFINITIONS

PacketCable specifications use the following terms:

<b>Access Control</b>	Limiting the flow of information from the resources of a system only to authorized persons, programs, processes, or other system resources on a network.
<b>Active</b>	A service flow is said to be "active" when it is permitted to forward data packets. A service flow must first be admitted before it is active.
<b>Admitted</b>	A service flow is said to be "admitted" when the CMTS has reserved resources (e.g., bandwidth) for it on the DOCSIS™ network.
<b>A-link</b>	A-Links are SS7 links that interconnect STPs and either SSPs or SCPs. 'A' stands for "Access."
<b>Asymmetric Key</b>	An encryption key or a decryption key used in public key cryptography, where encryption and decryption keys are always distinct.
<b>Audio Server</b>	An Audio Server plays informational announcements in PacketCable network. Media announcements are needed for communications that do not complete and to provide enhanced information services to the user. The component parts of Audio Server services are Media Players and Media Player Controllers.
<b>Authentication</b>	The process of verifying the claimed identity of an entity to another entity.
<b>Authenticity</b>	The ability to ensure that the given information is without modification or forgery and was in fact produced by the entity that claims to have given the information.
<b>Authorization</b>	The act of giving access to a service or device if one has permission to have the access.
<b>Cipher</b>	An algorithm that transforms data between plaintext and ciphertext.
<b>Ciphersuite</b>	A set which must contain both an encryption algorithm and a message authentication algorithm (e.g., a MAC or an HMAC). In general, it may also contain a key-management algorithm, which does not apply in the context of PacketCable.
<b>Ciphertext</b>	The (encrypted) message output from a cryptographic algorithm that is in a format that is unintelligible.
<b>Cleartext</b>	The original (unencrypted) state of a message or data. Also called plaintext.
<b>Confidentiality</b>	A way to ensure that information is not disclosed to anyone other than the intended parties. Information is encrypted to provide confidentiality. Also known as privacy.
<b>Cryptanalysis</b>	The process of recovering the plaintext of a message or the encryption key without access to the key.
<b>Cryptographic algorithm</b>	An algorithm used to transfer text between plaintext and ciphertext.
<b>Decipherment</b>	A procedure applied to ciphertext to translate it into plaintext.
<b>Decryption</b>	A procedure applied to ciphertext to translate it into plaintext.
<b>Decryption key</b>	The key in the cryptographic algorithm to translate the ciphertext to plaintext.
<b>Digital certificate</b>	A binding between an entity's public key and one or more attributes relating to its identity, also known as a public key certificate.

<b>Digital signature</b>	A data value generated by a public-key algorithm based on the contents of a block of data and a private key, yielding an individualized cryptographic checksum.
<b>Downstream</b>	The direction from the headend toward the subscriber location.
<b>Encipherment</b>	A method used to translate plaintext into ciphertext.
<b>Encryption</b>	A method used to translate plaintext into ciphertext.
<b>Encryption Key</b>	The key used in a cryptographic algorithm to translate the plaintext to ciphertext.
<b>Endpoint</b>	A Terminal, Gateway or Multipoint Conference Unit (MCU).
<b>Errored Second</b>	Any 1-second interval containing at least one bit error.
<b>Event Message</b>	A message capturing a single portion of a connection.
<b>F-link</b>	F-Links are SS7 links that directly connect two SS7 end points, such as two SSPs. 'F' stands for "Fully Associated."
<b>Flow [DOCSIS Flow]</b>	(a.k.a. DOCSIS-QoS "service flow") A unidirectional sequence of packets associated with a Service ID (SID) and a QoS. Multiple multimedia streams may be carried in a single DOCSIS Flow.
<b>Flow [IP Flow]</b>	A unidirectional sequence of packets identified by OSI Layer 3 and Layer 4 header information. This information includes source/destination IP addresses, source/destination port numbers, protocol ID. Multiple multimedia streams may be carried in a single IP Flow.
<b>Gateway</b>	Devices bridging between the PacketCable IP Voice Communication world and the PSTN. Examples are the Media Gateway, which provides the bearer circuit interfaces to the PSTN and transcodes the media stream, and the Signaling Gateway, which sends and receives circuit switched network signaling to the edge of the PacketCable network.
<b>H.323</b>	An ITU-T recommendation for transmitting and controlling audio and video information. The H.323 recommendation requires the use of the ITU-T H.225 and ITU-T H.245 protocol for communication control between a "gateway" audio/video endpoint and a "gatekeeper" function.
<b>Header</b>	Protocol control information located at the beginning of a protocol data unit.
<b>Integrity</b>	A way to ensure that information is not modified except by those who are authorized to do so.
<b>IntraLATA</b>	Within a Local Access Transport Area.
<b>Jitter</b>	Variability in the delay of a stream of incoming packets making up a flow such as a voice communication.
<b>Kerberos</b>	A secret-key network authentication protocol that uses a choice of cryptographic algorithms for encryption and a centralized key database for authentication.
<b>Key</b>	A mathematical value input into the selected cryptographic algorithm.
<b>Key Exchange</b>	The swapping of public keys between entities to be used to encrypt communication between the entities.
<b>Key Management</b>	The process of distributing shared symmetric keys needed to run a security protocol.
<b>Key Pair</b>	An associated public and private key where the correspondence between the two are mathematically related, but it is computationally infeasible to derive the private key from the public key.
<b>Keying Material</b>	A set of cryptographic keys and their associated parameters, normally associated with a particular run of a security protocol.

<b>Keyspace</b>	The range of all possible values of the key for a particular cryptographic algorithm.
<b>Latency</b>	The time, expressed in quantity of symbols, taken for a signal element to pass through a device.
<b>Link Encryption</b>	Cryptography applied to data as it travels on data links between the network devices.
<b>Network Layer</b>	Layer 3 in the Open System Interconnection (OSI) architecture that provides network information that is independent from the lower layers.
<b>Network Management</b>	The functions related to the management of data across the network.
<b>Network Management OSS</b>	The functions related to the management of data link layer and physical layer resources and their stations across the data network supported by the hybrid fiber/coax system.
<b>Nonce</b>	A random value used only once that is sent in a communications protocol exchange to prevent replay attacks.
<b>Non-Repudiation</b>	The ability to prevent a sender from denying later that he or she sent a message or performed an action.
<b>Off-Net Call</b>	A communication connecting a PacketCable subscriber out to a user on the PSTN.
<b>On-Net Call</b>	A communication placed by one customer to another customer entirely on the PacketCable Network.
<b>One-way Hash</b>	A hash function that has an insignificant number of collisions upon output.
<b>Plaintext</b>	The original (unencrypted) state of a message or data. Also called cleartext.
<b>Pre-shared Key</b>	A shared secret key passed to both parties in a communication flow, using an unspecified manual or out-of-band mechanism.
<b>Privacy</b>	A way to ensure that information is not disclosed to any one other than the intended parties. Information is usually encrypted to provide confidentiality. Also known as confidentiality.
<b>Private Key</b>	The key used in public key cryptography that belongs to an individual entity and must be kept secret.
<b>Proxy</b>	A facility that indirectly provides some service or acts as a representative in delivering information, thereby eliminating the need for a host to support the service.
<b>Public Key</b>	The key used in public key cryptography that belongs to an individual entity and is distributed publicly. Other entities use this key to encrypt data to be sent to the owner of the key.
<b>Public Key Certificate</b>	A binding between an entity's public key and one or more attributes relating to its identity, also known as a digital certificate.
<b>Public Key Cryptography</b>	A procedure that uses a pair of keys, a public key and a private key, for encryption and decryption, also known as an asymmetric algorithm. A user's public key is publicly available for others to use to send a message to the owner of the key. A user's private key is kept secret and is the only key that can decrypt messages sent encrypted by the user's public key.
<b>Root Private Key</b>	The private signing key of the highest-level Certification Authority. It is normally used to sign public key certificates for lower-level Certification Authorities or other entities.
<b>Root Public Key</b>	The public key of the highest level Certification Authority, normally used to verify digital signatures generated with the corresponding root private key.

<b>Secret Key</b>	The cryptographic key used in a symmetric key algorithm, which results in the secrecy of the encrypted data depending solely upon keeping the key a secret, also known as a symmetric key.
<b>Session Key</b>	A cryptographic key intended to encrypt data for a limited period of time, typically between a pair of entities.
<b>Signed and Sealed</b>	An "envelope" of information which has been signed with a digital signature and sealed using encryption.
<b>Subflow</b>	A unidirectional flow of IP packets characterized by a single source and destination IP address and single source and destination UDP/TCP port.
<b>Symmetric Key</b>	The cryptographic key used in a symmetric key algorithm, which results in the secrecy of the encrypted data depending solely upon keeping the key a secret, also known as a secret key.
<b>Systems Management</b>	Functions in the application layer related to the management of various Open Systems Interconnection (OSI) resources and their status across all layers of the OSI architecture.
<b>Transit Delays</b>	The time difference between the instant at which the first bit of a Protocol Data Unit (PDU) crosses one designated boundary, and the instant at which the last bit of the same PDU crosses a second designated boundary.
<b>Trunk</b>	An analog or digital connection from a circuit switch that carries user media content and may carry voice signaling (M <sub>F</sub> , R <sub>2</sub> , etc.).
<b>Tunnel Mode</b>	An IPsec (ESP or AH) mode that is applied to an IP tunnel, where an outer IP packet header (of an intermediate destination) is added on top of the original, inner IP header. In this case, the ESP or AH transform treats the inner IP header as if it were part of the packet payload. When the packet reaches the intermediate destination, the tunnel terminates and both the outer IP packet header and the IPsec ESP or AH transform are taken out.
<b>Upstream</b>	The direction from the subscriber location toward the headend.
<b>X.509 certificate</b>	A public key certificate specification developed as part of the ITU-T X.500 standards directory.

## 4 ABBREVIATIONS AND ACRONYMS

PacketCable specifications use the following abbreviations.

<b>AAA</b>	Authentication, Authorization and Accounting.
<b>AES</b>	Advanced Encryption Standard. A block cipher, used to encrypt the media traffic in PacketCable.
<b>AF</b>	Assured Forwarding. This is a DiffServ Per Hop Behavior.
<b>AH</b>	Authentication header. An IPsec security protocol that provides message integrity for complete IP packets, including the IP header.
<b>AMA</b>	Automated Message Accounting. A standard form of call detail records (CDRs) developed and administered by Bellcore (now Telcordia Technologies).
<b>ASD</b>	Application-Specific Data. A field in some Kerberos key management messages that carries information specific to the security protocol for which the keys are being negotiated.
<b>AT</b>	Access Tandem.
<b>ATM</b>	Asynchronous Transfer Mode. A protocol for the transmission of a variety of digital signals using uniform 53-byte cells.
<b>BAF</b>	Bellcore AMA Format, also known as AMA.
<b>BCID</b>	Billing Correlation ID.
<b>BPI+</b>	Baseline Privacy Plus Interface Specification. The security portion of the DOCSIS 1.1 standard that runs on the MAC layer.
<b>CA</b>	Certification Authority. A trusted organization that accepts certificate applications from entities, authenticates applications, issues certificates and maintains status information about certificates.
<b>CA</b>	Call Agent. The part of the CMS that maintains the communication state, and controls the line side of the communication.
<b>CBC</b>	Cipher Block Chaining mode. An option in block ciphers that combine (XOR) the previous block of ciphertext with the current block of plaintext before encrypting that block of the message.
<b>CBR</b>	Constant Bit Rate.
<b>CDR</b>	Call Detail Record. A single CDR is generated at the end of each billable activity. A single billable activity may also generate multiple CDRs.
<b>CIC</b>	Circuit Identification Code. In ANSI SS7, a two-octet number that uniquely identifies a DS0 circuit within the scope of a single SS7 Point Code.
<b>CID</b>	Circuit ID (Pronounced "kid"). This uniquely identifies an ISUP DS0 circuit on a Media Gateway. It is a combination of the circuit's SS7 gateway point code and Circuit Identification Code (CIC). The SS7 DPC is associated with the Signaling Gateway that has domain over the circuit in question.
<b>CIF</b>	Common Intermediate Format.
<b>CIR</b>	Committed Information Rate.
<b>CM</b>	DOCSIS Cable Modem.
<b>CMS</b>	Cryptographic Message Syntax.
<b>CMS</b>	Call Management Server. Controls the audio connections. Also called a Call Agent in MGCP/SGCP terminology. This is one example of an Application Server.

<b>CMTS</b>	Cable Modem Termination System. The device at a cable headend which implements the DOCSIS RFI MAC protocol and connects to CMs over an HFC network.
<b>CMSS</b>	Call Management Server Signaling.
<b>Codec</b>	COder-DECoder.
<b>COPS</b>	Common Open Policy Service protocol. Currently an internet draft, which describes a client/server model for supporting policy control over QoS Signaling Protocols and provisioned QoS resource management.
<b>CoS</b>	Class of Service. The type 4 tuple of a DOCSIS configuration file.
<b>CRCX</b>	Create Connection.
<b>CSR</b>	Customer Service Representative.
<b>DA</b>	Directory Assistance.
<b>DE</b>	Default. This is a DiffServ Per Hop Behavior.
<b>DES</b>	Data Encryption Standard.
<b>DF</b>	Delivery Function.
<b>DHCP</b>	Dynamic Host Configuration Protocol.
<b>DHCP-D</b>	DHCP Default. Network Provider DHCP Server.
<b>DNS</b>	Domain Name Service.
<b>DOCSIS™</b>	Data-Over-Cable Service Interface Specifications.
<b>DPC</b>	Destination Point Code. In ANSI SS7, a 3-octet number which uniquely identifies an SS7 Signaling Point, either an SSP, STP, or SCP.
<b>DQoS</b>	Dynamic Quality-of-Service. Assigned on the fly for each communication depending on the QoS requested.
<b>DSA</b>	Dynamic Service Add.
<b>DSC</b>	Dynamic Service Change.
<b>DSCP</b>	DiffServ Code Point. A field in every IP packet that identifies the DiffServ Per Hop Behavior. In IP version 4, the TOS byte is redefined to be the DSCP. In IP version 6, the Traffic Class octet is used as the DSCP.
<b>DTMF</b>	Dual-tone Multi Frequency (tones).
<b>EF</b>	Expedited Forwarding. A DiffServ Per Hop Behavior.
<b>E-MTA</b>	Embedded MTA. A single node that contains both an MTA and a cable modem.
<b>EO</b>	End Office.
<b>ESP</b>	IPsec Encapsulating Security Payload. Protocol that provides both IP packet encryption and optional message integrity, not covering the IP packet header.
<b>ETSI</b>	European Telecommunications Standards Institute.
<b>F-link</b>	F-Links are SS7 links that directly connect two SS7 end points, such as two SSPs. 'F' stands for "Fully Associated."
<b>FEID</b>	Financial Entity ID.
<b>FGD</b>	Feature Group D signaling.
<b>FQDN</b>	Fully Qualified Domain Name. Refer to IETF RFC 2821 for details.
<b>GC</b>	Gate Controller.
<b>GTT</b>	Global Title Translation.
<b>HFC</b>	Hybrid Fiber/Coaxial. An HFC system is a broadband bi-directional shared media transmission system using fiber trunks between the headend and the fiber nodes, and coaxial distribution from the fiber nodes to the customer locations.



<b>HMAC</b>	Hashed Message Authentication Code. A message authentication algorithm, based on either SHA-1 or MD5 hash and defined in IETF RFC 2104.
<b>HTTP</b>	Hypertext Transfer Protocol. Refer to IETF RFC 1945 and RFC 2068.
<b>IANA</b>	Internet Assigned Numbered Authority. See <a href="http://www.ietf.org">www.ietf.org</a> for details.
<b>IC</b>	Inter-exchange Carrier.
<b>IETF</b>	Internet Engineering Task Force. A body responsible, among other things, for developing standards used on the Internet. See <a href="http://www.ietf.org">www.ietf.org</a> for details.
<b>IKE</b>	Internet Key Exchange. A key-management mechanism used to negotiate and derive keys for SAs in IPsec.
<b>IKE-</b>	A notation defined to refer to the use of IKE with pre-shared keys for authentication.
<b>IKE+</b>	A notation defined to refer to the use of IKE with X.509 certificates for authentication.
<b>IP</b>	Internet Protocol. An Internet network-layer protocol.
<b>IPsec</b>	Internet Protocol Security. A collection of Internet standards for protecting IP packets with encryption and authentication.
<b>ISDN</b>	Integrated Services Digital Network.
<b>ISTP</b>	Internet Signaling Transport Protocol.
<b>ISUP</b>	ISDN User Part. A protocol within the SS7 suite of protocols that is used for call signaling within an SS7 network.
<b>ITU</b>	International Telecommunication Union.
<b>ITU-T</b>	International Telecommunication Union–Telecommunication Standardization Sector.
<b>IVR</b>	Interactive Voice Response system.
<b>KDC</b>	Key Distribution Center.
<b>LATA</b>	Local Access and Transport Area.
<b>LD</b>	Long Distance.
<b>LIDB</b>	Line Information Database. Contains customer information required for real-time access such as calling card personal identification numbers (PINs) for real-time validation.
<b>LLC</b>	Logical Link Control. The Ethernet Packet header and optional 802.1P tag which may encapsulate an IP packet. A sublayer of the Data Link Layer.
<b>LNP</b>	Local Number Portability. Allows a customer to retain the same number when switching from one local service provider to another.
<b>LSSGR</b>	LATA Switching Systems Generic Requirements.
<b>MAC</b>	Message Authentication Code. A fixed-length data item that is sent together with a message to ensure integrity, also known as a MIC.
<b>MAC</b>	Media Access Control. It is a sublayer of the Data Link Layer. It normally runs directly over the physical layer.
<b>MC</b>	Multipoint Controller.
<b>MCU</b>	Multipoint Conferencing Unit.
<b>MD5</b>	Message Digest 5. A one-way hash algorithm that maps variable length plaintext into fixed-length (16 byte) ciphertext.
<b>MDCP</b>	Media Device Control Protocol. A media gateway control specification submitted to IETF by Lucent. Now called SCTP.
<b>MDCX</b>	Modify Connection.
<b>MDU</b>	Multi-Dwelling Unit. Multiple units within the same physical building. The term is usually associated with high-rise buildings.

<b>MEGACO</b>	Media Gateway Control IETF working group. See <a href="http://www.ietf.org">www.ietf.org</a> for details.
<b>MF</b>	Multi-Frequency.
<b>MG</b>	Media Gateway. Provides the bearer circuit interfaces to the PSTN and transcodes the media stream.
<b>MGC</b>	Media Gateway Controller. The overall controller function of the PSTN gateway. Receives, controls and mediates call-signaling information between the PacketCable and PSTN.
<b>MGCP</b>	Media Gateway Control Protocol. Protocol follow-on to SGCP. Refer to IETF 3435.
<b>MIB</b>	Management Information Base.
<b>MIC</b>	Message Integrity Code. A fixed-length data item that is sent together with a message to ensure integrity, also known as a Message Authentication Code (MAC).
<b>MMC</b>	Multi-Point Mixing Controller. A conferencing device for mixing media streams of multiple connections.
<b>MSB</b>	Most Significant Bit.
<b>MSO</b>	Multi-System Operator. A cable company that operates many headend locations in several cities.
<b>MSU</b>	Message Signal Unit.
<b>MTA</b>	Multimedia Terminal Adapter. Contains the interface to a physical voice device, a network interface, CODECs, and all signaling and encapsulation functions required for VoIP transport, class features signaling, and QoS signaling.
<b>MTP</b>	The Message Transfer Part. A set of two protocols (MTP 2, MTP 3) within the SS7 suite of protocols that are used to implement physical, data link, and network-level transport facilities within an SS7 network.
<b>MWD</b>	Maximum Waiting Delay.
<b>NANP</b>	North American Numbering Plan.
<b>NANPNAT</b>	North American Numbering Plan Network Address Translation.
<b>NAT Network Layer</b>	Network Address Translation. Layer 3 in the Open System Interconnection (OSI) architecture. This layer provides services to establish a path between open systems.
<b>NCS</b>	Network Call Signaling.
<b>NPA-NXX</b>	Numbering Plan Area (more commonly known as area code) NXX (sometimes called exchange) represents the next three numbers of a traditional phone number. The N can be any number from 2-9 and the Xs can be any number. The combination of a phone number's NPA-NXX will usually indicate the physical location of the call device. The exceptions include toll-free numbers and ported number (see LNP).
<b>NTP</b>	Network Time Protocol. An internet standard used for synchronizing clocks of elements distributed on an IP network.
<b>NTSC</b>	National Television Standards Committee. Defines the analog color television broadcast standard used today in North America.
<b>OID</b>	Object Identification.
<b>OSP</b>	Operator Service Provider.
<b>OSS</b>	Operations Systems Support. The back-office software used for configuration, performance, fault, accounting, and security management.
<b>OSS-D</b>	OSS Default. Network Provider Provisioning Server.
<b>PAL</b>	Phase Alternate Line. The European color television format that evolved from the American NTSC standard.
<b>PCES</b>	PacketCable Electronic Surveillance.

<b>PCM</b>	Pulse Code Modulation. A commonly employed algorithm to digitize an analog signal (such as a human voice) into a digital bit stream using simple analog-to-digital conversion techniques.
<b>PDU</b>	Protocol Data Unit.
<b>PHS</b>	Payload Header Suppression. A DOCSIS technique for compressing the Ethernet, IP, and UDP headers of RTP packets.
<b>PKCROSS</b>	Public-Key Cryptography for Cross-Realm Authentication. Utilizes PKINIT for establishing the inter-realm keys and associated inter-realm policies to be applied in issuing cross-realm service tickets between realms and domains in support of Intradomain and Interdomain CMS-to-CMS signaling (CMSS).
<b>PKCS</b>	Public-Key Cryptography Standards. Published by RSA Data Security Inc. These Standards describe how to use public key cryptography in a reliable, secure and interoperable way.
<b>PKI</b>	Public-Key Infrastructure. A process for issuing public key certificates, which includes standards, Certification Authorities, communication between authorities and protocols for managing certification processes.
<b>PKINIT</b>	Public-Key Cryptography for Initial Authentication. The extension to the Kerberos protocol that provides a method for using public-key cryptography during initial authentication.
<b>PSC</b>	Payload Service Class Table, a MIB table that maps RTP payload Type to a Service Class Name.
<b>PSFR</b>	Provisioned Service Flow Reference. An SFR that appears in the DOCSIS configuration file.
<b>PSTN</b>	Public Switched Telephone Network.
<b>QCIF</b>	Quarter Common Intermediate Format.
<b>QoS</b>	Quality of Service. Guarantees network bandwidth and availability for applications.
<b>RADIUS</b>	Remote Authentication Dial-In User Service. An internet protocol (IETF RFC 2865 and RFC 2866) originally designed for allowing users dial-in access to the internet through remote servers. Its flexible design has allowed it to be extended well beyond its original intended use.
<b>RAS</b>	Registration, Admissions and Status. RAS Channel is an unreliable channel used to convey the RAS messages and bandwidth changes between two H.323 entities.
<b>RC4</b>	Rivest Cipher 4. A variable length stream cipher. Optionally used to encrypt the media traffic in PacketCable.
<b>RFC</b>	Request for Comments. Technical policy documents approved by the IETF which are available on the World Wide Web at <a href="http://www.ietf.cnri.reston.va.us/rfc.html">http://www.ietf.cnri.reston.va.us/rfc.html</a> .
<b>RFI</b>	The DOCSIS Radio Frequency Interface specification.
<b>RJ-11</b>	Registered Jack-11. A standard 4-pin modular connector commonly used in the United States for connecting a phone unit into a wall jack.
<b>RKS</b>	Record Keeping Server. The device, which collects and correlates the various Event Messages.
<b>RSA</b>	A public-key, or asymmetric, cryptographic algorithm used to provide authentication and encryption services. RSA stands for the three inventors of the algorithm; Rivest, Shamir, Adleman.
<b>RSA Key Pair</b>	A public/private key pair created for use with the RSA cryptographic algorithm.
<b>RSVP</b>	Resource Reservation Protocol.
<b>RTCP</b>	Real-Time Control Protocol.
<b>RTO</b>	Retransmission Timeout.

<b>RTP</b>	Real-time Transport Protocol. A protocol for encapsulating encoded voice and video streams. Refer to IETF RFC3550.
<b>SA</b>	Security Association. A one-way relationship between sender and receiver offering security services on the communication flow.
<b>SAID</b>	Security Association Identifier. Uniquely identifies SAs in the DOCSIS Baseline Privacy Plus Interface (BPI+) security protocol.
<b>SCCP</b>	Signaling Connection Control Part. A protocol within the SS7 suite of protocols that provides two functions in addition to those provided within MTP. The first function is the ability to address applications within a signaling point. The second function is Global Title Translation.
<b>SCP</b>	Service Control Point. A Signaling Point within the SS7 network, identifiable by a Destination Point Code that provides database services to the network.
<b>SCTP</b>	Stream Control Transmission Protocol.
<b>SDP</b>	Session Description Protocol.
<b>SDU</b>	Service Data Unit. Information delivered as a unit between peer service access points.
<b>SF</b>	Service Flow. A unidirectional flow of packets on the RF interface of a DOCSIS system.
<b>SFID</b>	Service Flow ID. A 32-bit integer assigned by the CMTS to each DOCSIS Service Flow defined within a DOCSIS RF MAC domain. SFIDs are considered to be in either the upstream direction (USFID) or downstream direction (DSFID). Upstream Service Flow IDs and Downstream Service Flow IDs are allocated from the same SFID number space.
<b>SFR</b>	Service Flow Reference. A 16-bit message element used within the DOCSIS TLV parameters of Configuration Files and Dynamic Service messages to temporarily identify a defined Service Flow. The CMTS assigns a permanent SFID to each SFR of a message.
<b>SG</b>	Signaling Gateway. An SG is a signaling agent that receives/sends SCN native signaling at the edge of the IP network. In particular, the SS7 SG function translates variants ISUP and TCAP in an SS7-Internet Gateway to a common version of ISUP and TCAP.
<b>SGCP</b>	Simple Gateway Control Protocol. Earlier draft of MGCP.
<b>SHA – 1</b>	Secure Hash Algorithm 1. A one-way hash algorithm.
<b>SID</b>	Service ID. A 14-bit number assigned by a CMTS to identify an upstream virtual circuit. Each SID separately requests and is granted the right to use upstream bandwidth.
<b>SIP</b>	Session Initiation Protocol. An application-layer control (signaling) protocol for creating, modifying, and terminating sessions with one or more participants.
<b>SIP+</b>	Session Initiation Protocol Plus. An extension to SIP.
<b>S-MTA</b>	Standalone MTA. A single node that contains an MTA and a non-DOCSIS MAC (e.g., ethernet).
<b>SNMP</b>	Simple Network Management Protocol.
<b>SOHO</b>	Small Office/Home Office.
<b>SS7</b>	Signaling System number 7. An architecture and set of protocols for performing out-of-band call signaling with a telephone network.
<b>SSP</b>	Service Switching Point. SSPs are points within the SS7 network that terminate SS7 signaling links and also originate, terminate, or tandem switch calls.

<b>STP</b>	Signal Transfer Point. A node within an SS7 network that routes signaling messages based on their destination address. This is essentially a packet switch for SS7. It may also perform additional routing services such as Global Title Translation.
<b>TCAP</b>	Transaction Capabilities Application Protocol. A protocol within the SS7 stack that is used for performing remote database transactions with a Signaling Control Point.
<b>TCP</b>	Transmission Control Protocol.
<b>TD</b>	Timeout for Disconnect.
<b>TFTP</b>	Trivial File Transfer Protocol.
<b>TFTP-D</b>	Default – Trivial File Transfer Protocol.
<b>TGS</b>	Ticket Granting Server. A sub-system of the KDC used to grant Kerberos tickets.
<b>TGW</b>	Telephony Gateway.
<b>TIPHON</b>	Telecommunications and Internet Protocol Harmonization Over Network.
<b>TLV</b>	Type-Length-Value. A tuple within a DOCSIS configuration file.
<b>TN</b>	Telephone Number.
<b>ToD</b>	Time-of-Day Server.
<b>TOS</b>	Type of Service. An 8-bit field of every IP version 4 packet. In a DiffServ domain, the TOS byte is treated as the DiffServ Code Point, or DSCP.
<b>TSG</b>	Trunk Subgroup.
<b>UDP</b>	User Datagram Protocol. A connectionless protocol built upon Internet Protocol (IP).
<b>VAD</b>	Voice Activity Detection.
<b>VBR</b>	Variable Bit Rate.
<b>VoIP</b>	Voice-over-IP.

## 5 TECHNICAL OVERVIEW

The PacketCable Audio Server Specification defines a suite of signaling protocols for providing announcement and media services in a PacketCable network. This section of the document:

- defines the architectural requirements for providing PacketCable announcement and media services,
- defines and categorizes announcement and media types,
- defines the components and their roles in the PacketCable Audio Server Architecture, as described in the PacketCable Architecture Technical Report [5], and
- describes the signaling and media interfaces in the PacketCable Audio Server Specification.

### 5.1 Architectural Requirements

The architectural requirements and assumptions for providing Audio and Media Services for a PacketCable Network are listed below. These requirements are based upon the specifications and technical reports that define the PacketCable architecture.

#### 5.1.1 Call Destination

The Audio Server Specification must define how announcements are provided for PacketCable on-net to off-net and on-net to on-net calls<sup>1</sup>.

#### 5.1.2 Media Formats

A Media Player must be able to generate the required announcements in any of the code formats required by the PacketCable Codecs specification [3].

#### 5.1.3 Security

Audio MUST be signaled and played in a secure manner. Media Player Controllers and Media Players MUST support the security protocols defined in the PacketCable Security specification [4].

#### 5.1.4 Operational Support Systems

Media Player Controllers MUST support the PacketCable billing and event message protocols as defined in [7], as applicable to the applications they support. No requirement has been identified for support of event reporting by the Media Player.

### 5.2 Announcement Definitions

Announcements can be divided into four distinct categories: tones, fixed-content, variable content, and interactive announcements.

#### 5.2.1 Tones

Includes tones such as reorder, busy, and ringback. See [1] and [6] for details.

---

<sup>1</sup> Announcements for Off-net to on-net calls will usually be handled by the PSTN as a result of SS7 clearing messages. However when appropriate, they also may be played from the PacketCable Media Gateway (MG).

### 5.2.2 Fixed-content Announcements

Fixed-content Announcements consist of audio messages with fixed-content that require no user interaction. For example, "Your call did not go through. Please hang up and try your call again."

### 5.2.3 Variable Content Announcements

Variable Content Announcements are messages that contain a customizing parameter(s) yet require no user interaction. For example, "The number you have dialed, 321-9876, has been changed. The new number is 321-6789."

### 5.2.4 Interactive Announcements

Interactive Announcements are announcements that require user interaction, DTMF (Dual Tone Multi-Frequency) or IVR. For example, "The number you have dialed, 541-321-9876, has been changed. The new number is 541-321-6789. To be connected to the new number, at a cost of thirty-five cents, please press 1."

### 5.2.5 Naming Conventions for Endpoint Identifiers

A flat name space for endpoints is used, with audio ports indicated by the prefix *aud* and the port number, e.g., *aud/12@audio-server-3.whatever.net*. Wildcards (\$, \*) may be used in place of the port numbers in accordance with standard NCS rules for wild card use.

Systems that support announcements only (i.e., no digit collection, no recording, and no speech recognition ability) may use the prefix *ann* instead of *aud*.

Some systems may use one additional level in the naming scheme to support the identification of specific cards. In this case the naming would look like *aud/<card number>/<port number>@audio-server-3.whatever.net*

## 5.3 Interfaces

The PacketCable Audio Server Specification defines a set of interfaces between the components responsible for providing audio services. The following figure illustrates the interfaces between these components. Only where an interface is exposed is it expected to meet PacketCable specification requirements.

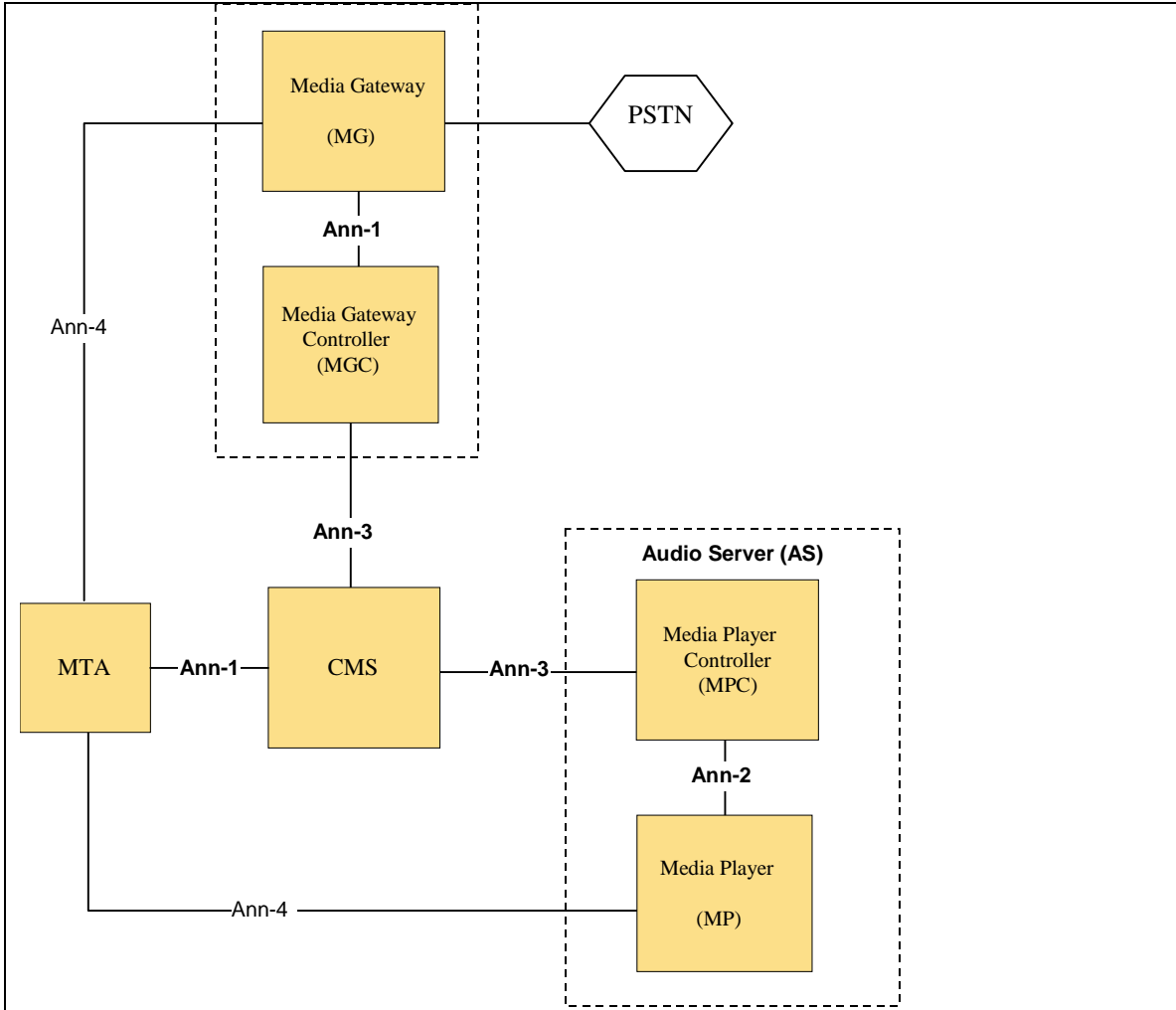


Figure 2. PacketCable Audio Server Components and Interfaces



## 6 ANN-1 INTERFACE: CMS-MTA AND MGC-MG

The CMS-MTA and MGC-MG announcement interfaces are implemented by the Legacy Audio Package of the NCS/TGCP protocol, which provides the playback of tones and fixed-content announcements to the end-users.

### 6.1 CMS-MTA Interface

Each MTA in the network MAY store a predefined set of simple announcements locally. When an announcement is needed, the CMS will decide if it should instruct the MTA to play a local announcement or set up a connection between the MTA and a Network MP and have the announcement played over the network. Playing simple announcements from the MTA saves network resources.

The MTA MAY store announcements in either static or dynamic memory. If announcements are stored in dynamic memory then the announcements will not be available until the MTA has accessed them from the network.

These simple announcements will require only a small amount of storage on the MTA. The table below illustrates the storage requirements for such announcements. The example uses an average announcement time of 10 seconds.

*Table 1. MP Storage*

Number of Announcements	Announcement Length (seconds)	Encoding bytes/second	Bytes required
11	10	2000 (G.728)	220 K
11	10	8000 (PCMU/PCMA)	880 K

MTAs require the ability to be updated dynamically with announcements so that the same MTA can move from service provider to service provider without requiring complete firmware upgrades. This capability is for further study and will need to be worked jointly with the PacketCable architecture, security, and provisioning teams.

#### 6.1.1 Announcement List

The MTA MAY store and play a defined set of announcements for common network situations. These announcements may be played using the Announcement Server Package defined in RFC 3435 [2], using URI (Universal Resource Identifiers) to identify the announcements. Cached versions of all announcement URIs should be refreshed every time the MTA connects to the network. Other methods of propagating new announcements to MTAs, for instance while the MTA remains in service, are for further study. The second column of the table below is a list of some announcements that MAY be supported in the MTA. The first column contains wording that may be used.

**Table 2. Sample Announcements**

<b>Sample Announcement</b>	<b>Name</b>
Your call cannot be completed as dialed. Please check the number and dial again.	Vacant Code
You must first dial a one or zero when calling this number. Please hang up and try your call again.	Dial One or Zero
You must first dial a one when calling this number. Please hang up and try your call again.	Dial One First
It is not necessary to dial a one when calling this number. Please hang up and try your call again.	No Dial One
If you'd like to make a call, please hang up and try again. If you need help, hang up and dial the operator.	No Digits
Your call cannot be completed as dialed. Please read the instruction card or call your operator for assistance.	Assisted Dialing
Your call did not go through. Please try your call again.	Reorder
All circuits are busy now. Please try your call later.	No Circuit
Due to facility trouble in the area you are calling, your call cannot be completed at this time. Please try your call later.	Domestic Facility
The party you are calling has declined to receive this call. Please try your call again with caller ID enabled.	Unidentified Call Reject
Thank you for using [carrier's name]	Branding

## 6.2 MGC-MG Interface

The MG announcement interface (Ann-1) allows for the MGC to request the MG play fixed-content announcements to PSTN end-users. The MGC/MG announcement interface package does not specify any standard announcements to be stored locally in the MG. All announcements are provisioned dynamically and are referenced accordingly.

This MG announcement provisioning capability is for further study and will need to be worked jointly with the PacketCable architecture, security, and provisioning teams.

## 7 ANN-2 INTERFACE: MPC-MP

### 7.1 Introduction

An MP (Media Player) is a shared resource in the PacketCable Network that can be instructed to provide media services to an end-user or terminal. These services include streaming fixed-content, variable content and interactive announcements to PacketCable subscribers. For example, the MP is responsible for playing prompts and collecting digits when charging a call to a calling card.

The MP is controlled by an external element, the MPC (Media Player Controller). The MPC-MP Interface defines a two new NCS announcement packages used to control the Media Player. The Base Audio Package provides a standard set of IVR functions such as Play, PlayCollect, and PlayRecord. The Advanced Audio Package is a superset of the Base Audio Package and provides additional capabilities.

The MP is responsible for managing its own resources. When accepting a request, the MP **MUST** make sure that the required resources are available before accepting the request. When a single session involves multiple requests to the Media Player, the MP may experience a shortage of resources preventing it from accepting one given request belonging to that session. In this case, the MP user (i.e., the MPC) is responsible for re-sending the request or terminating the end-user session elegantly.

### 7.2 Audio Package Concepts

The Base and Advanced Audio Packages support both simple and complex audio structures. A simple audio structure might be a single announcement such as "Welcome to Bell South's Automated Directory Assistance Service." A more complex audio structure might consist of an announcement followed by voice variable followed by another announcement, for example "There are thirty seven minutes remaining on your prepaid calling card," where "There are" is a prompt, the number of minutes is a voice variable, and "minutes remaining on your prepaid calling card" is another prompt.

It is also possible to define complex audio structures that are qualified by user defined selectors such as language, audio file format, gender, accent, customer, or voice talent. For instance, if the above example were qualified by language and accent selectors, it would be possible to play "There are thirty seven minutes remaining on your prepaid calling card" in English spoken with a southern accent or in English spoken with a mid-western accent, providing that the audio to support this had been provisioned.

There are two methods of specifying complex audio. The first is to directly reference the individual components. This requires a complete description of each component to be specified via the protocol. The second method is to provision the components on the Audio Server as a single entity and to export a reference to that entity to the call agent. In this case, only the reference (plus any dynamic data required, such as a variable data) is passed via the protocol, and no specification of individual components is necessary.

These packages provide significant functionality most of which is controlled via protocol parameters. Most parameters are optional, and where ever possible default to reasonable values. An audio application that references to complex provisioned audio structures can specify audio events using a minimum of syntax by taking advantage of parameter optionality and parameter defaults.

### 7.2.1 Understanding Audio Segments

An audio segment is a reference that resolves to one or more audio recordings. There are four types of audio segments:

**Physical:** A physical segment is the simplest type of segment, a single recording. The recording could be a single word, such as "one", or an extended block of speech, such as "Our office is closed at this time. Please call back during normal business hours." Every physical segment is assigned a unique URI (Universal Resource Identifier) which among other things can be a hierarchical name, or a simple name or number.

**Sequence:** A sequence is a provisioned ordered list of audio segments. Every sequence is assigned a unique URI. A sequence can contain any of the four segment types (physical segments, other sequences, sets, and variables). On playback a sequence identifier is resolved to an ordered list of physical segments which are played in order.

**Set:** A set is a provisioned collection of semantically related audio segments and an associated selector. Each set is assigned a unique URI. A set can contain physical segments, sequences, other sets, or variables. At runtime the value of the selector is used to determine which element of the set is played.

Individual selector types are not defined in the syntax (except for the pre-defined language selector) and are instead defined by the provisioner. A provisioner could define one or more of the following selector types: language, accent, gender, accent, customer, or day of the week. For each selector type, the provisioner must define a range of valid values. The provisioner may also choose to define a default value. At runtime if a selector value is not supplied the default value is used.

**Variable:** A voice variable represents a single semantic concept (such as date or number) and dynamically produces the appropriate speech based on information supplied at runtime. Each provisioned voice variable is assigned a unique URI. For example, if an application needs to play a date, rather than telling the AudioServer to play each individual component of the date (e.g., "March" "twenty" "second" "nineteen" "ninety" "nine"), it can specify a voice variable of type date with value "19990322". The variable then assembles and plays the component audio needed to speak the date. Specification of variables is considered in more detail in a later section of this document.

### 7.2.2 Segment Identifiers

Provisioned segments and segments recorded at runtime are identified by URIs as defined in RFC 2396, [8] Uniform Resource Identifiers: Generic Syntax.

A URI can be a simple name or it can be a URL. Three URL schemes are allowed: the file: scheme, the ftp:scheme, and the http: scheme. The file: scheme is used for audio local to the Audio Server. The ftp: scheme is used for audio remote to the Audio Server. The http: scheme can be used for audio local to the Audio Server using the http://localhost convention or audio remote to the Audio Server. All audio references that require parameters encoded in the URL (e.g., set selectors) MUST use the http: scheme. The following table shows some of the possibilities.

**Table 3. Example URIs**

Reference to local audio (flat file):
S: pa(an=file://welcome)
Reference to local audio (flat file):
S: pa(an=file://12354)
Reference to local audio:
S: pa(an=file://audio/xyztel/welcome)
Reference to remote audio:
S: pa(an=http://audio/xyztel/welcome)

### 7.2.3 Segment Life

Physical segments may be provisioned or they may be recorded during the course of a call. A physical segment recorded during the course of a call can be either transient or persistent. A transient physical segment lasts only for the duration of the call during which it was recorded. A persistent physical segment lasts beyond the duration of the call during which it was recorded.

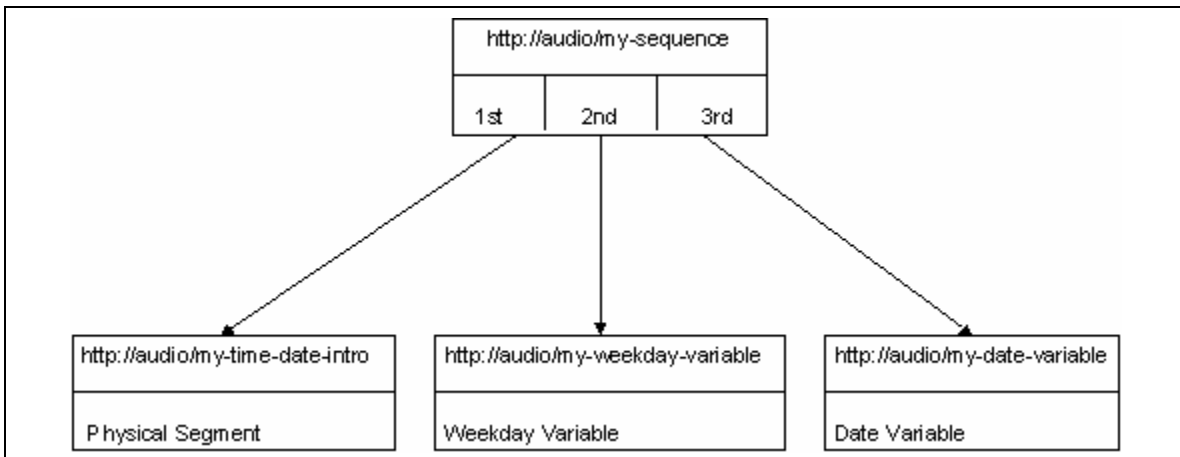
### 7.2.4 Nested Sets And Sequences

Nested definition of both sets and sequences is allowed, i.e., it is legal to define a set of sets or a sequence of sequences. In addition, audio structures may also be specified by intermixing sets and sequences, and it is possible to specify a set of sequences or a sequence containing one or more set elements. Direct or transitive definition of a set or segment in terms of itself is not allowed.

Nesting of sets and sequences should be restricted to two or three levels.

### 7.2.5 Sequence Example

In the following example, a provisioner has provisioned one physical segment and two variable segments and has provisioned a sequence, `http://mysegment`, which is an ordered list of the three segments. The sequence when played speaks the following: "Today's date is <weekday> <date>."



**Figure 3. Sequence Example**

### 7.2.6 Set Example

To support an application which plays a particular piece of audio in either Arabic, Welsh, or Tibetan, a provisioner could define a set with the pre-defined selector, "lang", and would use define three of the possible values for that selector, "ara", "cym", and "tib". The provisioner would provision three audio segments, one in each language, and would associate the Arabic segment with the "ara" selector value, etc. The provisioner also could define a default value of the selector when no selector value is supplied, "ara" for instance. The entire set would be assigned a unique URI.

At runtime a reference to the set with the selector set to "cym" would result in the Welsh version of the prompt being played. A reference to the set with no selector would result in the Arabic version of the prompt being played since English has been set as the default selector value.

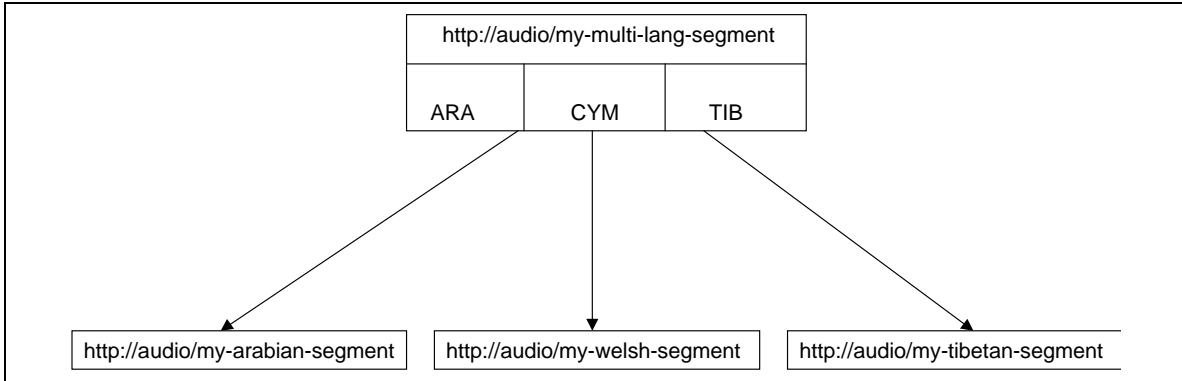


Figure 4. Set Example

**7.2.7 Set With Nested Sequence Example**

In this example, the provisioner has provisioned three physical segments, one in Arabic, one in Welsh, and one in Tibetan, and the provisioner has also provisioned three date variables. Using these six segments the provisioner has provisioned three sequences, each consisting of a physical segment followed by a date variable. Finally the provisioner has provisioned a set consisting of the three sequences and with language as the set selector.

At runtime a reference to the set with the selector set to "ara" and a variable value of "20001015" would result in the following being played in Arabic: "Today's date is October 15<sup>th</sup>, 2000."

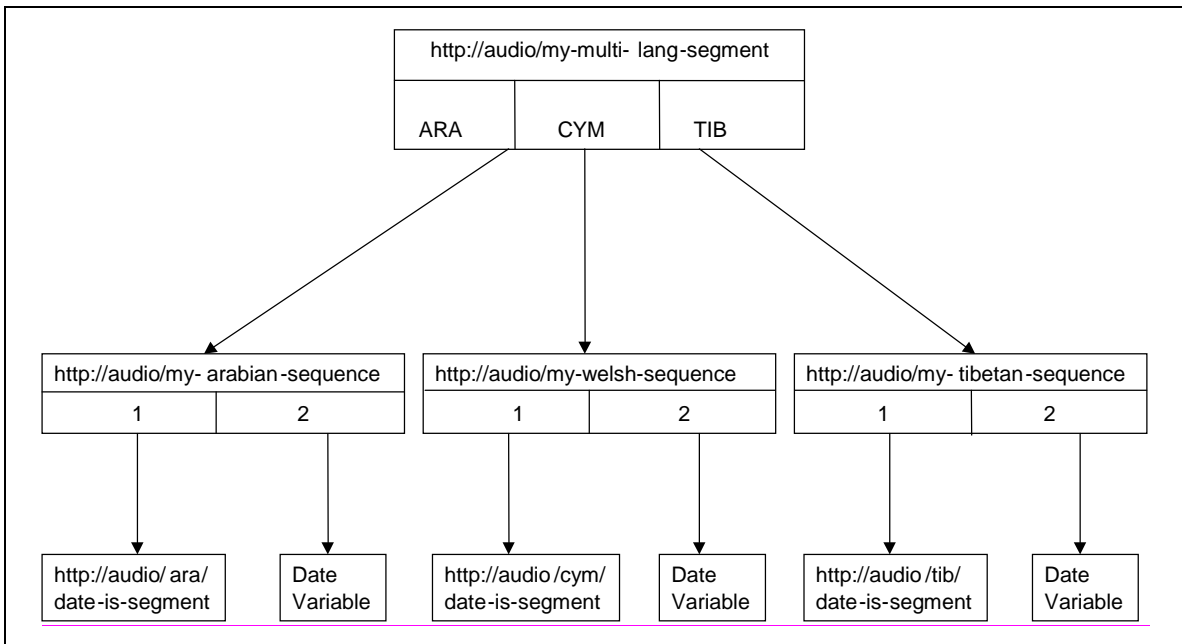


Figure 5. Set With Nested Sequence Example

**7.3 Base Audio Package**

**7.3.1 Abstract**

This event package provides support for the standard IVR operations of PlayAnnouncement, PlayCollect, and PlayRecord. It supports direct references to simple audio as well as indirect references to simple and

complex audio. It provides audio variables, control of audio interruptibility, digit buffer control, special key sequences, and support for reprompting during data collection.

**Package Name:** BAU

### 7.3.2 Events

*Table 4. Events*

Symbol	Definition	R	S	Duration
ma(parms)	ManageAudio		BR	variable
oc	OperationComplete	x		
of(parms)	OperationFailed	x		
pa(parms)	PlayAnnouncement		TO	variable
pc(parms)	PlayCollect		TO	variable
pr(parms)	PlayRecord		TO	variable

**PlayAnnouncement:** Plays an announcement in situations where there is no need for interaction with the user. Because there is no need to monitor the incoming media stream this event is an efficient mechanism for treatments, informational announcements, etc.

**PlayCollect:** Plays a prompt and collects DTMF digits entered by a user. If no digits are entered or an invalid digit pattern is entered, the user may be reprompted and given another chance to enter a correct pattern of digits. The following digits are supported: 0-9, \*, and #. By default PlayCollect does not play an initial prompt, makes only one attempt to collect digits, and therefore functions as a simple Collect operation. Various special purpose keys, key sequences, and key sets can be defined for use during the PlayCollect operation.

**PlayRecord:** Plays a prompt and records user speech. If the user does not speak, the user may be reprompted and given another chance to record. By default PlayRecord does not play an initial prompt, makes only one attempt to record, and therefore functions as a simple Record operation. The call agent may specify a URI to be associated with the recording or the call agent may ask the Audio Server to allocate a URI and return it to the call agent as part of the OperationComplete event. Digits entered by the user during a recording that are not defined as special key sequences are ignored and become part of the recording.

**ManageAudio:** Performs audio management operations on persistent audio which is typically not related to a current interaction with a user, e.g., "delete an audio segment" or "change volume for duration of connection".

**OperationComplete:** Detected upon the successful completion of a Play, PlayRecord, Play Collect, or ManageAudio signal.

**OperationFailed:** Detected upon the failure of a Play, PlayRecord, PlayCollect, or ManageAudio signal.

### 7.3.3 Signal Interactions

If an Audio Package signal is active on an endpoint and another signal of the same type is applied, the two signals including parameters and parameter values will be compared. If the signals are identical, the signal in progress will be allowed to continue and the new signal will be discarded. Because of this behavior the Advanced Audio Package may not interoperate well with some other packages such as the Line and Trunk packages.

### 7.3.4 Parameters

The PlayAnnouncement, PlayRecord, and PlayCollect events may each be qualified by a string of parameters, most of which are optional. Where appropriate, parameters default to reasonable values. If a required parameter is not supplied an error is returned to the application.

These parameters are shown in the following table:

**Table 5. Parameters**

Symbol	Definition	pa	pc	pr	ma
an	announcement	O	F	F	F
ap	append	F	F	O	F
cb	clear digit buffer	F	O	O	F
dm	digit map	F	O	O	F
dpa	delete persistent audio	F	F	F	O
du	duration	O	F	F	F
edt	extra digit timer	F	O	F	F
fa	failure announcement	F	O	O	F
fdt	first digit timer	F	O	F	F
ict	inter digit critical timer	F	O	O	F
idt	inter digit timer	F	O	O	F
ip	initial prompt	F	O	O	F
it	iterations	O	F	F	F
iv	interval	O	F	F	F
na	number of attempts	F	O	O	F
nd	no digits reprompt	F	O	F	F
ni	non-interruptible play	F	O	O	F
ns	no speech reprompt	F	F	O	F
off	offset	O	O	O	F
prt	prespeech timer	F	F	O	F
pst	postspeech timer	F	F	O	F
rid	recording id	F	F	M	F
rik	reinput key	F	O	O	F



Symbol	Definition	pa	pc	pr	ma
rlt	recording length timer	F	F	M	F
rp	reprompt	F	O	O	F
rpa	record persistent audio	F	F	O	F
rsk	restart key	F	O	O	F
rtk	return key	F	O	O	F
sa	success announcement	F	O	O	F
sp	speed	O	O	O	F
vl	volume	O	O	O	F
pv	persistent volume	F	F	F	O
mt	mute	F	F	F	O
lm	lecture mode	F	F	F	O

O = Optional M = Mandatory F = Forbidden

**Persistent volume:** The relative volume of the input audio stream is specifiable as a positive or negative decibel variation from the original volume. Support for this parameter is optional.

Persistent volume is, by default, applied to the endpoint. Persistent volume may be applied to a specific connection. When applied to the endpoint, persistent volume modifies the volume level of the endpoint for the remainder of the time that a connection is attached to the endpoint (i.e., in the case of a conference bridge, the volume of the mix is modified; in the case of any other endpoint the volume of the audio input to the endpoint is modified). If at any time the endpoint has no connections, the volume level is returned to the default volume level for that endpoint.

When applied to a connection, persistent volume modifies the volume level of the connection (i.e., the volume of the audio flowing from the endpoint to the connection is modified). The modification persists until another 'pv' explicitly changes it or the connection is deleted.

**Mute:** The values true and false are supported. Support for this parameter is optional.

Mute is, by default, applied to the endpoint. It may be applied to a specific connection. When applied to the endpoint, mute modifies the volume level of the endpoint for the remainder of the time that a connection is attached to the endpoint (i.e., in the case of a conference bridge, the volume of the mix is muted; in the case of any other endpoint the volume of the audio input to the endpoint is muted) or until a mt=false is received for the endpoint. If at any time the endpoint has no connections, the volume level is returned to the default volume level for that endpoint.

When applied to a connection, mute modifies the volume level of the connection (i.e., the volume of the audio flowing from the endpoint to the connection is muted). The modification persists until another 'mt' explicitly changes it or the connection is deleted.

Mute arguments persist across persistent volume changes. Persistent volume changes the base volume, mute changes the existence or absence of audio. Thus three consecutive invocations specifying: a) mt=true, b) pv=-2, c) mt=false would result in the audio stream reappearing with a slightly lower volume after step c.

**Lecture mode:** Lecture mode takes as an argument either the keyword "off" or the connection identifier of the lecturing connection. Lecture mode is always applied to an endpoint. Support for this parameter is optional.

When lecture mode is sent with a connection identifier as the argument, all connections on the endpoint other than the identified connection are muted, i.e., the audio arriving from the other connections is not included in e.g., the mix of a conference bridge. When lecture mode is set to "off", all connections are restored to their pre-lecture mode values. If the pre-lecture mode values have been subsequently modified by a persistent volume change, that change persists.

**Announcement:** An announcement to be played. Consists of one or more audio segments.

**Append:** If set to true, the audio recording will append to any existing content in the Recording ID. It MAY not be used with wildcarded Recording Ids. Valid values are "true" and "false".

**Clear Digit Buffer:** If set to true, clears the digit buffer before playing the initial prompt. Defaults to false. Valid values are the text strings "true" and "false."

**Delete Persistent Audio:** Indicates that the specified persistent audio segment is to be deleted. This parameter is carried by the Manage Audio event.

**Digit Map:** A digit map as specified in RFC 3435 [2], Media Gateway Control Protocol (MGCP) Version 1.0, which specifies one or more digit patterns to be collected. Valid digits are 0-9, \*, and #.

**Duration:** The maximum amount of time to play and possibly replay an announcement. Takes precedence over iteration and interval. Specified in units of 100 milliseconds. No default.

**Extra Digit Timer:** The amount of time to wait for a user to enter a final digit once the maximum expected amount of digits have been entered. Typically this timer is used to wait for a terminating key in applications where a specific key has been defined to terminate input. Specified in units of 100 milliseconds. If not specified, this timer is not activated. If an extra digit is entered it is returned to the application along with the other collected digits.

The Extra Digit Timer can be used to implement a consistent human interface when collecting a variable number of digits where collection can be terminated by a Return Key, typically the # key. For example, suppose an application has asked for a minimum of three digits and a maximum of six. If the user consistently uses the # key to terminate collection following digit strings are acceptable: xxx#, xxxx#, xxxxx#, and xxxxxx. The inconsistency arises when the user enters six digits. Because the maximum number of digits have been entered the Audio Server returns the digits immediately without waiting for the # key. If the type ahead is allowed (the default Audio Server behavior) and if user then enters the # key, the application has to decide whether the user meant the # key to terminate the six digits already collected or if the user meant to enter the # key to begin the next digit collection. The Extra Digit Timer tells the Audio Server to wait for an additional period of time after the maximum number of digits have been entered to see if the user is going to enter another key.

**Failure Announcement:** Played when all data entry attempts have failed. Consists of one or more audio segments. No default.

**First Digit Timer:** The amount of time allowed for the user to enter the first digit. The first digit starts after the announcements end. Specified in units of 100 milliseconds. Defaults to 50 (five seconds).

**Initial Prompt:** The initial announcement prompting the user to either enter DTMF digits or to speak. Consists of one or more audio segments. If not specified (the default), the event immediately begins digit collection or recording.

**Inter Digit Timer:** The amount of time allowed for the user to enter each subsequent digit when no alternative within a digit map has been matched. Specified units of 100 milliseconds seconds. Defaults to 50 (five seconds). The Inter Digit Timer is used when a partial dial timer is needed (see Tpar in [1]).

**Interdigit Critical Timer:** The amount of time allowed for the user to enter each subsequent digit when a dialed string matches both a complete digit map and a partial dial of another alternative within the digit map. Specified units of 100 milliseconds. Defaults to 30 (three seconds). The Inter Digit Timer is used when a critical dial timer is needed (see `Tcrit` in [1]).

**Interval:** The interval of silence to be inserted between iterative plays. Specified in units of 100 milliseconds. Defaults to 10 (one second).

**Iterations:** The maximum number of times an announcement is to be played. A value of minus one (-1) indicates the announcement is to be repeated forever. Defaults to one (1).

**No Digits Reprompt:** Played after the user has failed to enter a valid digit pattern during a `PlayCollect` event. Consists of one or more audio segments. Defaults to the `Reprompt`.

**No Speech Reprompt:** Played after the user has failed to speak during a `PlayRecord` event. Consists of one or more audio segments. Defaults to the `Reprompt`.

**NonInterruptible Play:** If set to true, the initial prompt of the `PlayCollect` or `PlayRecord` event is not interruptible by either voice or digits. Defaults to false. Valid values are the text strings "true" and "false." Digits entered during a non-interruptible initial prompt are accumulated and are treated as they would if they had been entered during the second (collect or record) phase of the event.

**Number Of Attempts:** The number of attempts the user is allowed to enter a valid digit pattern or to make a recording. Defaults to 1. Also used as a return parameter to indicate the number of attempts the user made.

**Offset:** Specifies the offset into an announcement to start playing. Offset MUST only be used with the initial prompt of the `PlayCollect` or `PlayRecord` events where the initial prompt is a single physical segment. An offset may be either positive or negative. A positive offset is the offset going forward from the beginning of the prompt. A negative offset is the offset going backwards from the end of the prompt. Offsets are specified in 10 millisecond units. Defaults to 0.

Offsets are useful when digit handling is done by the call agent, e.g., the user hits a DTMF key, the key is sent to the call agent, the call agent decides to ignore the key and tells the Audio Server to resume playing at the point of interrupt. Another application is to allow the user to skip back and forward in a physical segment.

**Postspeech Timer:** The amount of silence necessary after the end of the last speech segment for the recording to be considered complete. Specified in units of 100 milliseconds. Defaults to 50 (five seconds).

**Prespeech Timer:** The amount of time to wait for the user to initially speak. Specified in units of 100 milliseconds. Defaults to 30 (three seconds).

**Record Persistent Audio:** If set to true, the recording that is made is persistent instead of temporary. Defaults to false. Valid values are the text strings "true" and "false." This parameter is carried by the `PlayRecord` event, although nothing is either played or recorded in this case.

**Recording ID:** A URI to be assigned to the physical segment which is to be recorded by the `PlayRecord` event. If this parameter is set to the ANY wildcard, "\$", the Audio Server will allocate the URI, associate it with the newly recorded segment, and return it to the call agent with the `OperationComplete` event.

**Recording Length Timer:** The maximum allowable length of the recording, not including pre or post speech silence. Specified in units of 100 milliseconds. This parameter is mandatory for the `PlayRecord` signal. A value of -1 (minus one) means there is no limit to recording length. In this case the recording is open ended, and it is up to the application to manage the storage resources for recordings.

**Reinput Key:** Defines a digit map that, if matched, has the following action: discard any digits collected or recording in progress and resume digit collection or recording. No default.

The use of this key does not constitute an attempt to enter user input (i.e., it does not count against the number of attempts specified by the Number Of Attempts parameter). Reinput keys are handled locally by the Audio Server and are not returned to the call agent. During a recording, all digits except for the restart, reinput, and return keys (if defined) are ignored and become part of the recording.

**Reprompt:** Played after the user has made an error such as entering an invalid digit pattern or not speaking. Consists of one or more audio segments. Defaults to the Initial Prompt.

**Restart Key:** Defines a digit map that, if matched, has the following action: discard any digits collected or recording in progress, replay the prompt, and resume digit collection or recording. No default.

The use of this key does not constitute an attempt to enter user input (i.e., it does not count against the number of attempts specified by the Number Of Attempts parameter). Restart Keys are handled locally by the Audio Server and are not returned to the call agent. During a recording, all digits except for the restart, reinput, and return keys (if defined) are ignored and become part of the recording.

**Return Key:** Defines a digit map that, if matched, has the following action: stop digit collection or recording. If the return key is hit during a PlayCollect event, all keys collected prior to detection of the return key are returned to the call agent. If the return key is hit during a PlayRecord event, the recording is saved, all keys collected prior to the return key are returned, and a Recording ID is returned if appropriate. (See definition of RecordingID for details.) Detection of the return key constitutes successful completion of the collection operation even if no digit map match has occurred.

**Speed:** The relative playback speed of announcement specifiable as a positive or negative percentage of the original playback speed.

**Success Announcement:** Played when data collection has succeeded. Consists of one or more audio segments. No default.

**Volume:** The relative playback volume of announcement specifiable as a positive or negative decibel variation from the original playback volume.

### 7.3.5 Type-ahead

The Audio Server supports type-ahead by default. Type-ahead is not supported for the Play event because by definition no digit collection is done during this event. Type-ahead can be turned off for all prompts associated with a PlayCollect or PlayRecord event by setting the Clear Digit Buffer parameter.

### 7.3.6 Return Parameters

Each event has an associated set of possible return parameters that are returned with either the OperationComplete or OperationFailed events. These parameters are listed in the following table:

**Table 6. Return Parameters**

Symbol	Definition	pa	pc	pr	ma
ap	amount played	F	C	C	F
dc	digits collected	F	O	O	F
na	number of attempts	F	M	M	F
rc	return code	O	O	O	O
rid	recording id	F	F	O	F
rl	Recording length	F	F	M	F

O = Optional M = Mandatory F = Forbidden C = Conditional (see expanded definition)

**Amount Played:** The length played of an initial prompt if the prompt was interrupted, in 10 ms. units. This parameter is mandatory if the prompt was interrupted, and forbidden otherwise.

**Digits Collected:** If returned with an oc event, this parameter contains the DTMF digits that were collected during a PlayCollect operation. If returned with an of event, this parameter contains the DTMF digits that were collected during an unsuccessful PlayCollect or PlayRecord operation up until the point of failure.

**Number Of Attempts:** The number of attempts the user actually needed to enter a valid digit pattern or to make a recording. Defaults to 1. Also used as an input parameter to specify the number of attempts the user will be allowed to enter a valid digit pattern or make a recording. This parameter is returned only if a na parameter was specified on the PlayCollect or PlayRecord.

**Return Code:** A return code giving the final status of the operation, followed optionally by a comma and the offending item. The possible return codes are:

**Table 7. Return Codes**

Return Code	Meaning
600	Illegal syntax
601	Unknown segment ID
602	Variable type not supported
603	Variable subtype not supported
604	Invalid variable name
605	Variable value out of range
606	Inconsistent variable specification
607	Extra sequence data
608	Missing sequence data
609	Mismatch between play specification and provisioned data
610	Delete audio error
611	Unable to record temporary audio
612	Unable to delete temporary audio
613	Unable to record persistent audio
614	Unable to delete persistent audio
615	Unable to override non-existent segment id
616	Unable to remove override from non-existent segment id
617	Provisioning error
618	Hardware failure
619	Unspecified failure
620	No digits
621	No speech
622	Spoke too long
623	Digit map not matched
624	Max attempts exceeded
625	No free segment ids
626	Required parameter not set
627	Inconsistent parameter set
628	Value out of range
629	Invalid offset
630	Invalid digit map

Examples:

The PlayAnnouncement event completed successfully. Note that no return code is necessary:

O: BAU/oc

The PlayAnnouncement event failed the parameters supplied were inconsistent:

O: BAU/of(rc=627)

The PlayCollect event completed successfully on the user's second attempt when the user entered the digits 04375182:

O: BAU/oc(na=2 dc=04375182)

The PlayRecord event was successful on the user's first attempt; the id of the recording made by the user is 983:

```
O: BAU/oc(na=1 ri=983)
```

The PlayRecord event was successful on the user's first attempt; the id of the recording made by the user is 983 and the duration was 27.5 seconds:

```
O: BAU/oc(na=1 ri=983 rl=275)
```

The PlayAnnouncement event failed the variable name requested was not recognized:

O: BAU/of(rc=604,zwq)**Recording ID:** A URI assigned to physical segment recorded by the PlayRecord operation. This parameter is returned only if the RecordingID parameter to the PlayRecord event has been set to the ANY wildcard, "\$". If this is the case the Audio Server allocates a unique URI, associates it with the newly recorded segment, and returns it to the call agent.

**Recording Length:** The length of the recording, not including pre or post speech silence. Specified in units of 100 milliseconds. This parameter is mandatory for the PlayRecord signal. In the case where the append operation was used, this is the length of the new recording, not the total length.

### 7.3.7 Segment Descriptors

Segment descriptors are used with the an, ip, rp, nd, ns, fa, and sa parameters to define the segments that make up an announcement. There are two kinds of segment descriptors:

**Table 8. Segment Descriptors**

Symbol	Definition
<URI>	segment identifier
vb	variable

**Segment Identifier:** A URI identifying a provisioned entity, i.e., a physical segment, sequence, or variable.

**Variable:** Specifies a voice variable by type, subtype, and value, and used when the application specifies a variable on the fly as opposed to referencing a provisioned variable. Does not apply to provisioned variables. Variables are more completely defined in a subsequent section of the document.

### 7.3.8 Variable Syntax

The syntax supports two kinds of variables. Embedded variables are variables that have been provisioned as part of an audio segment. At runtime the call agent references the segment and specifies a value for the variable. Typically embedded variables are provisioned along with recorded speech, e.g., "A representative will be with you in approximately 5 minutes. If you would prefer to leave a voice message, press 1 now." where the variable is the number of minutes. Standalone variables are variables that are not provisioned and therefore MUST be completely specified on the fly by the call agent or MPC. Variables are specified by the following parameters: type, subtype, and value. Variable types include Date, Money, Number, Time, etc. Subtype is a refinement of type. For example the variable type Money might have an associated range of subtypes such as Dollar, Rupee, Dinar, etc. Not all variables require a subtype, and for these variables the subtype parameter should be set to null.

For embedded variables, the type and subtype MUST be provisioned. The value may be provisioned. If it is not provisioned it MUST be specified as part of the variable reference. In a list of segments, an embedded variable value specification applies only to the segment that directly precedes it. If a segment has multiple embedded variables, the values MUST be given in the order in which the variables are encountered when the segment is played. Some examples:

```
Standalone variable: S: pa(an=vb(mny,usd,1153))
Embedded variable: S: pa(an=file://ann1<1153>)
```

A variable segment is played in the default language and voice of the server. When an application desires to select an alternative language or voice, the syntax of the URL query shown in Section 7.4.4 and Table 12 is used. Example: if the application wishes to speak a monetary value in English with a male voice, it would send

```
pa(an=vb(mny,usd,1153)?lang=english&gender=male)
```

Not all variables, such as the date variable shown in the next example, require a subtype. In that case, the subtype is encoded with the value "null":

```
S: pa(an=vb(dat,null,101598))
```

In some cases it may be desirable to play an announcement that contains an embedded variable without playing the variable itself. To do this a single "null" is provided for the value:

```
S: pa(an=file://ann1<null>)
```

### 7.3.9 Variable Definitions

Variable types and subtypes are specified in the following table:

**Table 9. Variable Types and Subtypes**

Type	Subtype	Definition
dat	mdy, dmy, etc.	Date
	mdy	Month-Day-Year
	dym	Day-Year-Month
dig	gen, ndn	Digits
	gen	Generic
	ndn	North American DN
dur		Duration
mth		Month
mny	<ISO 4217 three letter codes>	Money
num	crd, ord	Number
	crd	Cardinal
	ord	Ordinal
sil		Silence
str		String
tme	t12, t24	Time
	t12	Twelve hour format
	t24	Twenty four hour format
ton	vpackage name	Tone identifier
wkd		Weekday

**Date:** Speaks a date specified as YYYYMMDD (per ISO 8601 [12]). If the subtype is Month-Date-Year "20001015", for example would be spoken as "October Fifteenth Two Thousand." If the subtype is Date-Month\_Year the same date would be spoken as "Fifteen October Two Thousand." Date subtypes may be



extended as needed as long as they are patterned after the existing subtypes (i.e., they MUST be a three letter combination of the letters m, d, and y).

**Digits:** Speaks a string of digits one at a time. If the subtype is North American DN, the format of which is NPA-NXX-XXXX, the digits are spoken with appropriate pauses between the NPA and NXX and between the NXX and XXXX. If the subtype is generic, the digits are spoken no pauses.

**Duration:** Duration is specified in seconds and is spoken in one or more units of time as appropriate, e.g., "3661" is spoken as "One hour, one minute, and one second", "3360" is spoken as "One hour and one minute", and "3600" is spoken as "One minute."

**Money:** Money is specified in the smallest units of a given currency and is spoken in one or more units of currency as appropriate, e.g., "110" in U.S. Dollars would be spoken "one dollar and ten cents." The three letter codes defined in ISO 4217, Currency And Funds Code List [11] are used to specify the currency subtype. A small excerpt from ISO 4217 follows:

**Table 10. Sample Currency Codes**

Code	Currency	Entity
GQE	Ekwele	Equatorial Guinea
GRD	Drachma	Greece
GTQ	Quetzal	Guatemala

Money can be specified in negative or positive units of currency. In the above example "-110" would be spoken as "minus one dollar and ten cents."

**Month:** Speaks the specified month, e.g., "10" is spoken as "October." Specification is in MM format with "01" denoting January, "02" denoting February, etc.

**Number:** Speaks a number in cardinal form or in ordinal form. For example, "100" is spoken as "one hundred" in cardinal form and "one hundredth" in ordinal form. Cardinal numbers can be specified as negative or positive.

**Silence:** Plays a specified period of silence. Specification is in 100 millisecond units.

**String:** Speaks each character of a string, e.g., "a34bc" is spoken "A, three, four, b, c." Valid characters are a-z, A-Z, 0-9, #, and \*.

**Time:** Speaks a time in either twelve hour format or twenty four hour format depending on the specified subtype. For example "1700" is spoken as "Five p.m." in twelve hour format or as "Seventeen hundred hours" in twenty four hour format. Specification is in HHMM format per ISO 8601 [12].

**Tone:** The tone variable is used to cause the audio player to generate a defined tone from any other standard package as part of the sequence of audio segments. If the package referenced in the request is not known to (or not supported by) the audio player an error code of 603 *Variable subtype not supported* shall be returned. **Caution:** Only tones of known duration should be used. Examples:

```
vb(ton,L,ci(1942,3036619100,CableLabs))
```

```
vb(ton,D,2) -or- vb(ton,L,2)
```

```
vb(ton,SL,(D/1,D/5,D/7))
```

**Weekday:** Speaks the day of the week, e.g., "Monday." Weekdays are specified as single digits, with "1" denoting Sunday, "2" denoting Monday, etc.

### 7.3.10 Timers

Four timers are defined in this package:

- First digit timer (FDT)
- Interdigit timer (IDT)
- Interdigit critical timer (ICT)
- Extra digit timer (EDT)

Consistent implementation of the interaction between timers is important for applications that use audio servers that meet this specification. The following guidelines are strongly recommended:

1. There is no need for more than one of the timers to run at any given time in processing of a digit map.
2. The first digit timer (FDT) will be started on receipt of the collection request if no initial prompt is present, at completion of the playing of the initial prompt, and on completion of any reprompts. If a digit is collected during the playing of the initial prompt or reprompt, the FDT is not started.
3. The interdigit timer (IDT) will be started when the end of a tone is detected if there are no possible matches and there are still possible matches. The IDT will not run if the collected tone (digit) completes a match or if the collected tone (digit) completes a match except for a terminal "T".
4. The interdigit critical timer (ICT) will be started when the digit map includes a terminal "T" and the matched string is a subset of a longer string. If an additional digit/tone is detected during the running of the ICT, it is examined to determine whether it creates a possible match (or partial match) of an alternative in the digit map. Thus in the digit map "123T|12345", the "T" represents the running of the ICT. If a "4" arrives before ICT expires, the digit map matching algorithm selects option two and continues the process.
5. The extra digit timer (EDT) runs after a match has been completed, even if the completion of the match required another timer to run (e.g., the ICT). Any digit that is detected while the EDT is running is returned in the observed events string, and the detection of that event will result in a OF response with RC=623 and the DC= parameter showing all digits detected prior to and during the EDT period. The EDT does not run if the digit map matching algorithm determines an error condition exists (no match possible, no digits entered, etc.).

Some examples with commentary:

```
dm=123|1234
```

Option two (1234) can not be matched – the algorithm will return immediately on detection of 123. If specified, EDT may run after the match, but the 4, if entered, will be ignored.

```
dm=123T|1234
```

The ICT will run after the 3 is entered. If the timer expires, the match (123T) is returned. If a 4 is detected before the expiration of ICT, the match (1234) is returned. If a different digit is detected, error processing (return, reprompt, as appropriate) is started.

### 7.3.11 Examples

This section presents a number of syntax examples. Play an announcement that consists of a single segment:

```
S: pa(an=file://12333)
```

Play an announcement that consists of multiple segments:

```
S: pa(an=file://ann798,file://ann300,file://ann4747)
```

Play an announcement that consists of a recording followed by three seconds of silence followed by a standalone voice variable:

```
S: pa(an=file://ann357,vb(sil,null,30),vb(my,usd,3999))
```

Play an announcement with an embedded variable. If the separate segments of the previous announcement were provisioned as a sequence with a segment id of ann43321, the following would be exactly equivalent to the previous example:

```
S: pa(an=file://ann43321<3999>)
```

Play an announcement with two embedded variables:

```
S: pa(an=http://jackstraw/audio/xyztel/hello  
<3999,10151998>)
```

Play a prompt and collect a single digit. If need be, play a reprompt, a no digits prompt, and a success or failure announcement. Give the user three attempts to enter a digit:

```
S: pc(ip=file://ann27 rp=file://ann19 nd=file://ann102  
fa=file://ann8 sa=file://ann777 na=file://ann31  
dm=x)
```

Play a prompt and collect a single digit. If the user does not enter a digit replay the initial prompt. Give the user three attempts to enter a digit:

```
S: pc(ip=file://audio/ann77775 na=3 dm=x)
```

Play a prompt and record voice. If the user does not speak play a no speech prompt. Give the user two attempts to record:

```
S: pr(ip=http://brenda/audio/ann070500  
ns=http://althea/audio/no-speech na=2)
```

Play an announcement at ninety percent of its original speed and five decibels softer than its original volume. Play the announcement three times with two seconds of silence between plays.

```
S: pa(an=file://ann276 sp=90 vl=-5 it=3 iv=20)
```

Give the user two attempts to enter a three digit pattern. Clear the digit buffer before playing the prompt.

```
S: pc(ip=file://438975 cb=true dm=xxx na=2)
```

Give the user three attempts to enter a three digit pattern. If the user enters one digit or two digits on the first or second attempts a reprompt is played. If the user enters no digits on the first or second attempts a no digits reprompt is played. If all three attempts fail, a failure announcement is played. If one of the attempts is successful, a success announcement is played and the collected digits are returned to the call agent.

```
S: pc(ip=file://ann493 rp=5 nd=409 fa=file://ann923  
sa=file://ann18337 dm=xxx)
```

Give the user three chances to enter an 11 digit number that begins with 0 or 1. If the user makes a mistake while entering digits, he can press the \* key to discard any digits already collected, replay the prompt, and resume collection.

```
S: pc(ip=http://stella/blue/audio/ann5684  
dm=0xxxxxxxxxxx|1xxxxxxxxxxx rsk=* na=3)
```

Give the user two chances to make a recording. After playing the prompt, wait 5 seconds for the user to speak, otherwise replay the initial prompt and try again. If the user does speak, wait for seven seconds after speech stops to make sure the user is finished. If the recording is successful, return a reference to the recording to the call agent.

```
S: pr(ip=file://ann432 prt=50 pst=70 na=2)
```

## 7.4 Advanced Audio Package

### 7.4.1 Abstract

The Advanced Audio package extends the Base Audio package by adding the set capability which the user can use to create an arbitrary number of user defined qualifiers to be used in resolving complex audio structures. For example, the user could define qualifiers for any or all of the following: language, accent, audio file format, gender, speaker, or customer.

**Package Name:** AAU

### 7.4.2 Sets

A set is a provisioned collection of semantically related audio segments with an associated selector. Each set is assigned a unique URI. A set can contain physical segments, sequences, other sets, or variables. At runtime the value of the selector is used to determine which element of the set is played.

Individual selector types are not defined in the syntax (except for the pre-defined language selector, "lang") and are instead defined by the provisioner. A provisioner could define one or more of the following selector types: language, accent, gender, accent, customer, or day of the week. For each selector type, the provisioner must define a range of valid values. The provisioner may also choose to define a default value. At runtime if a selector value is not supplied the default value is used.

### 7.4.3 Selectors

Selector types, except for the predefined "lang" (language) selector, are defined by the user. For each selector type, the user must define a range of values that the selector can assume.

Selectors apply to individual audio segments. If an event specifies multiple segments, each segment have its own set of selectors. If selectors are not specified for an audio segment, provisioned defaults are used.

For example, if the user defines a selector of type "phaseofthemoon", he might also define the legal values for that selector to be "new", "half", "full", "harvest", and "blue". For the selector to actually work at runtime, audio associated with each of the selector values be provisioned.

The three letter codes defined in ISO standard 639-2, Code For The Representation Of Names Of Languages [10] MUST be used as values for user defined language selectors. For languages that have both a bibliographic and a terminology code, both codes should be supported. A small excerpt from ISO 639-2 follows:

**Table 11. Sample Language Codes**

Code	Language
cze	Czech
cym	Welsh
dan	Danish

Selectors are applied to variables only after the variable has been resolved. For instance if a date variable resolved to "October 15th, 1998" the voice with which the variable is spoken could resolve to either male or female if a gender selector had been defined.

Selectors are encoded as parameters to the URI segment id. If the URI refers to a physical segment on a node other than the Audio Server, to fetch the audio from the remote node the URI must contain the information necessary for this node to resolve the URI to a specific physical segment. This does not imply that the remote node needs the same capability as the Audio Server to resolve complex audio references.

The remote node could for example use a simple scheme such as encoding the hierarchical directory path to the physical in the URI.

#### 7.4.4 Selector Encoding

Provisioned segments and segments recorded at runtime are identified by URIs as defined in RFC 2396, Uniform Resource Identifiers: Generic Syntax.

A URI can be a simple name or it can be a URL. If a URL refers to audio stored on a node other than the Audio Server it must contain all the information necessary to resolve the URL to a physical segment. If the URL refers to a set, the selector types and values necessary to resolve the URL to a physical segment must be encoded in the query field of the URL. URLs for audio local to the Audio Server should use the file: scheme. URLs for audio remote to the Audio Server should use the http: scheme. The following table shows some of the possibilities.

**Table 12. Example URIs**

Reference to local audio (set):
S: pa(an=http://localhost/audio/xyztel/welcome?lang=eng&gender=female)
Reference to remote audio (set):
S: pa(an=http://audio/xyztel/welcome?lang=eng&gender=female)

#### 7.4.5 Variable Order

When a provisioned segment containing more than one variable is referenced at runtime, the variable values **MUST** be supplied in the order in which they occur in the provisioned segment. This principle extends to sets. If the elements of a set contain more than one variable then for all elements of the set the variables **MUST** occur in the same order. Sets with elements containing variables that do not appear in the same order are not supported.

#### 7.4.6 Overrides

A provisioned physical segment may be replaced (or overridden) by a persistent physical segment. The URI of the provisioned physical segment will then resolve to the persistent physical segment. The overriding persistent audio can subsequently be deleted and the original provisioned audio can be restored.

A provisioned physical segment may be overridden more than once. In this case, the URI of the provisioned physical segment refers to the latest overriding physical segment. When the overriding physical segment is deleted, the original provisioned physical segment is restored, even if the segment has been overridden multiple times.

Segment override could be used for a feature where a standard greeting is played to all customers calling a retail store. Occasionally the store manager may want to call a special number and record a temporary greeting that overrides the standard greeting, for instance a greeting that announces a sale or maybe a seasonal greeting of some kind. When the greeting is no longer wanted, the manager can call the special number, cancel the temporary greeting, and restore the standard greeting.

### 7.4.7 Parameters

**Table 13. Parameters**

Symbol	Definition	pa	pc	pr	ma
oa	override persistent audio	F	F	F	O
ra	restore persistent audio	F	F	F	O

O = Optional M = Mandatory F = Forbidden

Override Persistent Audio: The id of the segment to be overridden and the id of the overriding segment.

Restore Persistent Audio: The id of the segment to be restored.

### 7.4.8 Return Codes

The following return codes are defined for the Advanced Audio Package:

**Table 14. Return Codes**

Return Code	Meaning
650	Bad selector type
651	Bad selector value
652	Missing selector
653	Missing selector value
654	Wrong number of selector
655	Remove override error
656	Override error
657	Unable to override non-existent segment id
658	Unable to remove override from non-existent segment id

### 7.4.9 Examples

This section presents a number of examples of how sets and selectors are used.

Play an announcement in English.

```
S: pa(an=file://audio/xyztel/hello?lang=eng)
```

Play an announcement in a Danish, female voice with a Cajun accent.

```
S: pa(an=file://audio/xyztel/hello?lang=dan&
gender=female&accent=cajun)
```

Play the first part of an announcement in English, the second part in the default language, and the third part in French.

```
S: pa(an=file://ann1?lang=eng,file://ann2,
file://ann2?lang=fra)
```

Play an announcement with an embedded variable in English (the embedded variable is also played in English):

```
S: pa(an=file://ann4?lang=eng<101599>)
```

## 7.5 Overview

The purpose of this document is to provide extensions to support speech recognition, natural language understanding, and dialog manager resources in the MGCP (NCS) based media server framework.

This is a superset of the PASS BAU package, and the BAU package is assumed present with this package. The package is also compatible with the AAU package, but the presence of the AAU capabilities is not assumed.

Package name is "Speech Recognition" -SPR. The presence of the package name SPR as a signal descriptor incorporates all of the BAU package. A signal in the SPR package may simultaneously use parameters from both the BAU and SPR package (see examples below).

### 7.5.1 Speech Recognition Extensions to the BAU package

This table extends Table 4, Section 7.3.2

**Table 15. S1 Events**

Symbol	Definition	R	S	Duration
prg(parms)	PlayRecognize		TO	variable
mac(parms)	ManageASRContext		BR	variable

**PlayRecognize:** PlayRecognize extends the PlayCollect signal. PlayRecognize plays a prompt and recognizes spoken utterance and/or collects DTMF digits entered by a user. If the user does not speak, or enter no digits or an invalid digit pattern the user may be prompted again and given another chance to respond.

The grammars and vocabulary used for recognition may be provisioned ahead of time or configured using ManageASRContext signal.

The following digits are supported: 0-9, \*, and #. By default PlayRecognize does not play an initial prompt, makes only one attempt to collect digits or recognize utterance, and therefore functions as a simple PlayCollect operation. The various special purpose keys, key sequences, and key sets defined for use in the PlayCollect operation are equally valid in the PlayRecognize operation.

**ManageASRContext:** Performs context management to support the subsequent PlayRecognize signal. An ASR context is completely specified by its network – a composition of grammars, dictionary, acoustic models and associated parameters. A network is a fully composed entity that ASR engine decodes on. A compiled grammar is a network consisting of sentence rules, vocabulary words and their pronunciations. The grammars can be a fully composed network entity and may be referenced using its name. In addition, using this signal we will make it possible for the call agent to manipulate the part of the grammar called rules. The JSGF (Java Speech Grammar Format Specification) [13] and/or W3C defined syntax format [14] will be used to define these rules. Updating rules in a grammar will allow an application to bind dynamic grammar fragments at a call time to a pre-composed surround grammar network.

The default pre-defined grammars available to an application will include the VoiceXML built-in grammars and other pre-defined application specific grammars that are provisioned and configured upon initialization of the ASR resource.

To specify inline grammars the W3C defined grammar format (Speech Recognition Grammar Specification for the W3C Speech Interface Framework [14]) will be supported.

## 7.5.2 Signal Interactions

If an Audio Package signal is active on an endpoint and another signal of the same type is applied, the two signals including parameters and parameter values will be compared if the signals are identical, the signal in progress will be allowed to continue and the new signal will be discarded. Because of this behavior the Advanced Audio Package may not interoperate well with some other packages such as the Line and Trunk packages.

## 7.5.3 Parameters

The PlayRecognize and ManageASRContext events may each be qualified by a string of parameters, most of which are optional. Where appropriate, reasonable default parameter values have been defined. If a required parameter is not supplied an error is returned to the application.

The PlayRecognize command re-uses the following digit and prompt related parameters as defined for PlayCollect in the BAU: Initial Prompt (ip), Reprompt (rp), No digit reprompt (nd), No response reprompt (nr), Failure announcement (fa), Success announcement (sa), Offset (off), Non-interruptible play (ni), Iterations (it), Interval (iv), Duration (du), Speed (sp), Volume (vl), Clear digit buffer (cb), Digit map (dm), Restart Key (rsk), Reinput Key (rik), Return Key (rtk), First digit timer (fdt), inter digit timer (idt) and extra digit timer (edt), Number of attempts (na). They are not described in the following table.

The parameters specific to the PlayRecognize operation are shown in the following table:

**Table 16. S2. Parameters**

Symbol	Definition	prg	mac
pst	prespeech timer	O	F
ptt	postspeech timer	O	F
iwt	Inter word timer	O	F
mut	Max utterance timer	O	F
miut	Minimum utterance timer	O	F
idt	Initial delay timer	O	F
cfl	Confidence level	O	F
esl	Energy sensitivity level	O	F
jsgp	JSGF Path	O	F
nbst	Nbest candidates	O	F
bth	Barge-in threshold	O	F
dtmf	DTMF allowed	O	F
lwt	Listen window length timer	O	F
sva	SpeedvsAccuracy	O	F
rsp	restart phrase	O	F
rip	reinput phrase	O	F
rtp	return phrase	O	F
ldg	Load grammar	F	O
sdg	Set dynamic grammar	O	O
ag	Activate grammar	O	O
dag	Deactivate grammar(s)	O	O

O = Optional      F = Forbidden

**Prespeech Timer:** The amount of time to wait for the user to initially speak. Specified in units of 100 milliseconds. Defaults to 30 (three seconds).



**Postspeech Timer:** The amount of silence necessary after the end of the last detected utterance segment for the recognition to be considered complete. Specified in units of 100 milliseconds. Defaults to 50 (five seconds).

**Inter Word Timer:** The amount of silence necessary before declaring end of the last word. It is also the maximum pause allowed between two consecutive spoken words in an utterance.

**Listen Window Length Timer:** The maximum allowable listen window, not including pre or post speech silence. Specified in units of 100 milliseconds. This parameter is mandatory for the PlayRecognize signal. The default value, -1 (minus one), means there is no limit to recognition listen window length. In this case the recognition is open ended, and it is up to the application and or recognition engine based on the active grammar to decide when to stop listening.

**Maximum utterance timer:** The maximum length of detected speech utterance before recognizer quits decoding of a spoken speech utterance. Prevents recognizer from getting stuck in falsely detected speech frames forever. This should be less than Listen window timer.

**Minimum utterance timer:** The minimum length of detected speech segment to validate that the recognizer is in the speech part of the spoken utterance.

**Initial delay timer:** This timer acts as an offset into the incoming speech stream to indicate when to start the recognition process. This timer helps to overcome echo canceller algorithm's delay in adapting to the channel. All the other speech timers start upon expiration of this timer.

**Confidence level:** It is a normalize parameter in the range of 0 to 1.0, used to tell recognizer to reject the recognized hypothesis with confidence level less than the value of this parameter. Default value 0.5 is used, if not explicitly specified.

**Energy sensitivity level:** This parameter is used to tell recognizer the minimum energy level of the signal before considering it to be beginning of speech. The normalized range for this parameter is 0.0 to 1.0 and a default value of 0.5 is used.

**JSGF Path:** If the specified grammar path is not absolute then it is considered to be the path name relative to the root directory specified by the JSGF Path parameter.

**Nbest candidates:** Number of recognition results requested by an application from the recognition resource. It is an integer with a default value of 1.

**Bargein threshold:** For a smart barge in feature, where speech recognizer is used to detect the beginning of an utterance the value of this parameter provides the minimum confidence level in such detected speech fragments before recognizer can declare that barge in has occurred. The normalized range for this parameter is 0.0 to 1.0 and a default value of 0.5 is used.

**DTMF Allowed:** When present and set to true, DTMF digits may be detected in lieu of spoken digits. Default value is false.

**SpeedvsAccuracy:** This parameter allows application to trade the CPU usage at the cost of recognition accuracy. The normalized range for this parameter is 0.0 to 1.0 with a default value of 0.5.

**Restart Phrase:** Defines a JSGF syntax grammar that, if matched, has the following action: discard any digits collected or recognition in progress, replay the prompt, and resume digit collection and/or recognition. No default.

The use of this phrase does not constitute an attempt to enter user input (i.e., it does not count against the number of attempts specified by the Number Of Attempts parameter). Restart Phrase are handled locally by the Audio Server and are not returned to the call agent.

**Reinput Phrase:** Defines a JSGF syntax grammar that, if matched, has the following action: discard any digits collected or recognition in progress and resume digit collection and/or recognition. No default.

The use of this phrase does not constitute an attempt to enter user input (i.e., it does not count against the number of attempts specified by the Number Of Attempts parameter). Just like Reinput keys the Reinput Phrases are handled locally by the Audio Server and are not returned to the call agent.

**Return Phrase:** Defines a JSGF syntax grammar that, if matched, has the following action: stop digit collection or recognition. If the return phrase is hit during a PlayRecognize event, all keys collected or phrases recognized are returned to the call agent. No default.

Some recognition engines allow a tag to represent multiple variants of a phrase. In the case that such a tag was used, the return value will be tagged in accordance with the standard operation of that grammar.

**Number Of Attempts:** The number of attempts the user is allowed to enter a valid digit pattern or to speak an utterance. Defaults to 1. Also used as a return parameter to indicate the number of attempts the user made.

**Load Grammar:** Used to load a grammar on an allocated ASR resource. The grammar may be an in-line JSGF BNF or a pre-compiled vendor specific native format grammar. The format is `ldg={grammar_name|grammar_bnf}`.

This operation may cause delays for complex and large grammars.

**Activate Grammar:** associated with ManageASRContext to specify which preloaded grammar and rule(s) to activate. If rule name is not specified the entire grammar is activated. When specified with "prg" it may cause delays in starting the recognizer. For large grammars it is desired to have this parameter set by using the "mac" signal ahead of the "prg" signal. The format is `ag=grammar_name,rule1,...,rulen`.

**Set Dynamic Grammar:** Used to associate a dynamic grammar fragment (e.g., a list of words or address book names or commands) to a predefined rule name in a pre-loaded grammar. The grammar fragment may be an in-line JSGF BNF or a pre-compiled vendor specific native format grammar. The format is `sdg=rule_name,grammar_bnf`.

This operation may cause delays for complex and large grammars. When specified with "prg" it may cause delays in starting the recognizer. For large grammars it is desired to have this parameter set by using the "mac" signal ahead of the "prg" signal.

**Deactivate Grammars:** Deactivate grammar signal without any parameters will result in the deactivation of all grammars and it can be used by application to bring ASR resource into a known state. Individual grammars can be deactivated by specifying their name. When specified with "prg" signal, it implies which grammars to deactivate upon completion of the speech recognition to bring the recognizer back in the original state. The format is `dg=grammar1,...,grammarn`.

#### 7.5.4 Type-ahead

The Audio Server supports type-ahead by default. Type-ahead is not supported for the Play event because by definition no digit collection is done during this event. Type-ahead can be turned off for all prompts associated with a PlayCollect or PlayRecord event by setting the Clear Digit Buffer parameter.

### 7.5.5 Return Parameters

Each event has an associated set of possible return parameters, which are returned with either the OperationComplete or OperationFailed events. These parameters are listed in the following table:

This table extends Section 7.3.6, Table 6. Values not specified are forbidden.

**Table 17. S3. Return Parameters**

Symbol	Definition	prg	ma	mac
ap	amount played	C	F	F
dc	digits collected	O	F	F
ru	Recognized utterance(s)	O	F	F
na	number of attempts	M	F	F
rc	return code	O	O	O

O = Optional M = Mandatory F = Forbidden C = Conditional (see expanded definition)

**Amount Played:** The length played of an initial prompt if the prompt was interrupted, in 10 ms. units. This parameter is mandatory if the prompt was interrupted, and forbidden otherwise.

**Digits Collected:** If returned with an oc event, this parameter contains the DTMF digits that were collected during a PlayCollect or PlayRecognize operation. If returned with an of event, this parameter contains the DTMF digits that were collected during an unsuccessful PlayCollect, PlayRecognize or PlayRecord operation up until the point of failure.

**Recognized utterance:** If returned with an oc event, this parameter contains the recognition result(s) from the PlayRecognize operation. If returned with an of event, this parameter contains the recognition results that were collected during an unsuccessful PlayRecognize operation up until the point of failure.

**Number Of Attempts:** The number of attempts the user actually needed to enter or speak a valid response to a played prompt or to make a recording. Defaults to 1. Also used, as an input parameter to specify the number of attempts the user will be allowed to respond to a played prompt or make a recording. This parameter is returned only if a na parameter was specified on the PlayCollect, PlayRecognize or PlayRecord.

**Return Code:** A return code giving the final status of the operation:

**Table 18. S4. Return Codes**

Return Code	Meaning
700	Failure to recognize
730	Grammar not found
731	Invalid grammar fragment
732	Failure to activate grammar or rule(s)

Examples:

The PlayRecognize event completed successfully on the user's second attempt when the user entered the digits 04375182 and DTMF was allowed:

```
O: SPR/oc(na=2 dc=04375182)
```

The PlayRecognize event completed successfully on the user's second attempt when the user spoke the digits "0 4 3 7 5 1 8 2":

```
O: SPR/oc(na=2 ru=04375182)
```

The ManageASRContext event completed successfully:

```
O: SPR/oc
```

The ManageASRContext event failed the parameters supplied were inconsistent:

```
O: SPR/of(rc=627)
```

### 7.5.6 Grammar Descriptors

Grammar descriptors are used with the ldg, sdg, ag, rsp, rip and rtp parameters to define the grammars to be managed for the subsequent PlayRecognize signal. There are two kinds of grammar descriptors:

**Table 19. S5. Grammar Descriptors**

Symbol	Definition
<URI>	Grammar identifier
Inline	Inline definition of the grammar

**Grammar Identifier:** A URI identifying a provisioned grammar entity.

**Inline Grammars:** Is an inline grammar represented using JSGF or W3C defined syntax.

### 7.5.7 Examples

This section presents samples of automatic speech recognition interactions.

Play a prompt and collect the speaker's voice input for PIN code validation...

```
S:mac(ldg ag= file://grammar-digits,rulePIN).
```

```
S:prg(lwt=1000 sva=0.5 dtmf=true bth=0.9)
```

```
O: SPR/oc(na=1 ru=1234)
```

## 7.6 Formal Syntax Description

This description uses ABNF (RFC 2234 [9]) to formally describe the syntax of the Basic Audio Package and the Advanced Audio Package. The two packages have the same syntax except for the encoding of selector types and selector values in the query field of the URI and the persistent audio override capabilities. See RFC 2396 for the syntax of encoding parameter value pairs in the query field of the URL.

AudPkgEvent = PlayAnnouncement / PlayCollect / PlayRecord / ManageAudio / OperationComplete / OperationFailed

PlayAnnouncement = [ AudioPkgToken SLASH ] PlayAnnToken

LPAREN PlayAnnParmList RPAREN

PlayCollect = [ AudioPkgToken SLASH ] PlayColToken

LPAREN [ PlayColParmList ] RPAREN

PlayRecord = [ AudioPkgToken SLASH ] PlayRecToken

LPAREN [ PlayRecParmList ] RPAREN

ManageAudio = [AudioPkgToken SLASH] ManageAudToken LPAREN ManageAudParmList RPAREN

OperationComplete = [ AudioPkgToken SLASH ] OpCompleteToken  
 LPAREN [OpCompleteParmList ] RPAREN  
 OperationFailed = [ AudioPkgToken SLASH ] OpFailedToken  
 LPAREN ReturnCodeParm RPAREN  
 PlayAnnParmList = PlayAnnParm \*( WSP PlayAnnParm )  
 PlayColParmList = PlayColParm \*( WSP PlayColParm )  
 PlayRecParmList = PlayRecParm \*( WSP PlayRecParm )  
 ManageAudParmList = ManageAudParm \*( WSP ManageAudParm )  
 OpCompleteParmList = OpCompleteParm \*( WSP OpCompleteParm )  
 PlayAnnParm = ( AnnouncementParm / IterationsParm / IntervalParm /  
 DurationParm / SpeedParm / VolumeParm )  
 PlayColParm = ( InitPromptParm / RepromptParm / NoDigitsParm / FailAnnParm /  
 SuccessAnnParm / NoInterruptParm / SpeedParm / VolumeParm / ClearBufferParm /  
 DigitMapParm / FirstDigitParm / InterDigitParm / InterDigitCritParm /  
 ExtraDigitParm / RestartKeyParm / ReinputKeyParm /  
 ReturnKeyParm / NumAttemptsParm )  
 PlayRecParm = ( InitPromptParm / RepromptParm / NoSpeechParm / FailAnnParm /  
 SuccessAnnParm / NoInterruptParm / SpeedParm / VolumeParm /  
 ClearBufferParm / PreSpeechParm / PostSpeechParm /  
 RecordLenTimerParm / RestartKeyParm / ReinputKeyParm /  
 ReturnKeyParm / NumAttemptsParm )  
 ManageAudParm = ( RecPersistParm / DeletePersistParm / OverrideAudioParm /  
 RestoreAudioParm )  
 OpCompleteParm = ( NumAttemptsParm / AmtPlayedParm / DigitsColParm  
 RecordingIdParm / ReturnCodeParm / RecordLenParm )  
 AnnouncementParm = AnParmToken EQUALS Segmentlist  
 InitPromptParm = IpParmToken EQUALS Segmentlist  
 RepromptParm = RpParmToken EQUALS Segmentlist  
 NoDigitsParm = NdParmToken EQUALS Segmentlist  
 NoSpeechParm = NsParmToken EQUALS Segmentlist  
 FailAnnParm = FaParmToken EQUALS Segmentlist  
 SuccessAnnParm = SaParmToken EQUALS Segmentlist  
 OffsetParm = OffParmToken EQUALS OPTSIGNEDINT  
 DurationParm = DuParmToken EQUALS NUMBER  
 IterationsParm = ItParmToken EQUALS ( NUMBER / MINUSONE )  
 IntervalParm = IvParmToken EQUALS NUMBER  
 SpeedParm = SpParmToken EQUALS SIGNEDINT

VolumeParm = ViParamToken EQUALS SIGNEDINT  
 NoInterruptParm = NiParmToken EQUALS BOOLSTR  
 ClearBufferParm = CbParmToken EQUALS BOOLSTR  
 DigitMapParm = DmParmToken EQUALS DigitMap  
 DigitMap = <defined in RFC 3435>  
 FirstDigitParm = FdtParmToken EQUALS NUMBER  
 InterDigitParm = IdtParmToken EQUALS NUMBER  
 InterDigitCritParm = IctParmToken EQUALS NUMBER  
 ExtraDigitParm = EdtParmToken EQUALS NUMBER  
 PreSpeechParm = PrtParmToken EQUALS NUMBER  
 PostSpeechParm = PstParmToken EQUALS NUMBER  
 RecordLenTimerParm = RltParmToken EQUALS NUMBER  
 RecordLenParm = RIParmToken EQUALS NUMBER  
 RestartKeyParm = RskParmToken EQUALS DigitMap  
 ReinputKeyParm = RikParmToken EQUALS DigitMap  
 ReturnKeyParm = RtkParmToken EQUALS DigitMap  
 RecPersistParm = RpaParmToken EQUALS BOOLSTR  
 DeletePersistParm = DpaParmToken EQUALS SegmentId  
 OverrideAudioParm = OaParmToken EQUALS OverridenSegId OverridingSegId  
 OverridenSegId = SegmentId  
 OverridingSegId = SegmentId  
 RestoreAudioParm = RaParmToken EQUALS SegmentId  
 NumAttemptsParm = NaParmToken EQUALS NUMBER  
 AmtPlayedParm = ApParmToken EQUALS NUMBER  
 DigitsColParm = DcParmToken EQUALS KeySequence  
 RecordingIdParm = RidParmToken EQUALS UniversalResourceIdentifier  
 ReturnCodeParm = RcParmToken EQUALS 3\*3(DIGIT)  
 KeyPadKey = "0" / "1" / "2" / "3" / "4" / "5" / "6" / "7" / "8" / "9" / "\*" / "#"  
 KeySequence = 1\*64(KeyPadKey)  
 KeySet = 1\*11(KeyPadKey)  
 Segmentlist = SegmentDescriptor \*( COMMA SegmentDescriptor )  
 SegmentDescriptor = SegmentId [ EmbedVarList ] / VariableSeg  
 SegmentId = UniversalResourceIdentifier  
 UniversalResourceIdentifier = <defined in RFC 2396>  
 VariableSeg = VariableSegToken LPAREN FullSpecVar RPAREN  
 EmbedVarList = LANGLE NAME \*( COMMA NAME ) RANGLE  
 FullSpecVar = ( DateVariable / DigitsVariable / DurationVariable /

MonthVariable / MoneyVariable / NumberVariable /  
SilenceVariable / StringVariable / TextVariable /  
TimeVariable / ToneVariable / WeekdayVariable )  
DateVariable = DateVarToken COMMA NullStrToken COMMA Date  
Date = 8\*8(DIGIT)  
DigitsVariable = DigitsVarToken COMMA (NorthAmericanDnToken /  
GenericDigitsToken) COMMA NUMBER  
DurationVariable = DurationVarToken COMMA NullStrToken COMMA NUMBER  
MoneyVariable = MoneyVarToken COMMA 3\*3(ALPHA) COMMA OPTSIGNEDINT  
MonthVariable = MonthVarToken COMMA NullStrToken COMMA Month  
Month = "01" / "02" / "03" / "04" / "05" / "06" / "07" / "08" / "09" / "10" / "11" / "12"  
NumberVariable =  
(NumberVarToken COMMA CardinalNumberToken COMMA OPTSIGNEDINT) /  
(NumberVarToken COMMA OrdinalNumberToken COMMA NUMBER)  
SilenceVariable = SilenceVarToken COMMA NullStrToken COMMA NUMBER  
StringVariable = StringVarToken COMMA NullStrToken COMMA \*(KeyPadKey)  
SilenceVariable = SilenceVarToken COMMA NullStrToken COMMA NUMBER  
StringVariable = StringVarToken COMMA NullStrToken COMMA \*(KeyPadKey)  
TimeVariable = TimeVarToken COMMA (TwelveHourFormatToken /  
TwentyFourHourFormatToken) COMMA 4\*4(DIGIT)  
WeekdayVariable = WeekdayVarToken COMMA NullStrToken COMMA NAME  
AudioPkgToken = BaseAudPkgToken / AdvAudPkgToken  
BaseAudPkgToken = "BAU"  
AdvAudPkgToken = "AAU"  
PlayAnnToken = "pa"  
PlayColToken = "pc"  
PlayRecToken = "pr"  
ManageAudToken = "ma"  
OpCompleteToken = "oc"  
OpFailedToken = "of"  
VariableSegToken = "vb"  
AnParmToken = "an"  
IpParmToken = "ip"  
RpParmToken = "rp"  
NdParmToken = "nd"  
NsParmToken = "ns"

FaParmToken = "fa"  
SaParmToken = "sa"  
OffParmToken = "off"  
NiParmToken = "ni"  
ItParmToken = "it"  
IvParmToken = "iv"  
DuParmToken = "du"  
SpParmToken = "sp"  
VlParmToken = "vl"  
CbParmToken = "cb"  
DmParmToken = "dm"  
FdtParmToken = "fdt"  
IdtParmToken = "idt"  
IctParmToken = "ict"  
EdtParmToken = "edt"  
PrtParmToken = "prt"  
PstParmToken = "pst"  
RltParmToken = "rlt"  
RlParmToken = "rl"  
RskParmToken = "rsk"  
RikParmToken = "rik"  
RtkParmToken = "rtk"  
RpaParmToken = "rpa"  
DpaParmToken = "dpa"  
OaParmToken = "oa"  
RaParmToken = "ra"  
ApParmToken = "ap"  
DcParmToken = "dc"  
NaParmToken = "na"  
RcParmToken = "rc"  
RidParmToken = "rid"  
DateVarToken = "dat"  
DigitsVarToken = "dig"  
DurationVarToken = "dur"  
DayYrMonthToken = "dym"  
MonthDayYrToken = "mdy"  
MoneyVarToken = "mny"



MonthVarToken = "mth"  
NumberVarToken = "num"  
SilenceVarToken = "sil"  
StringVarToken = "str"  
TimeVarToken = "tme"  
ToneVarToken = "ton"  
PackageNameToken = <defined in the package specifications>  
PackageSignalToken = <defined in the package specifications>  
GenericDigitsToken = "gen"  
NorthAmericanDnSToken = "ndn"  
CardinalNumberToken = "crd"  
OrdinalNumberToken = "ord"  
TwelveHourFormatToken = "t12"  
TwentyFourHourFormatToken = "t24"  
WeekdayVarToken = "wkd"  
NullStrToken = "null"  
BOOLSTR = "true" / "false"  
NAMECHAR = ALPHA / DIGIT / "\_" / "-"  
NAME = 1\*64(NAMECHAR)  
NUMBER = DIGIT \*31(DIGIT)  
SIGNEDINT = ("+" / "-") DIGIT \*31(DIGIT)  
OPTSIGNEDINT = ["+" / "-"] DIGIT \*31(DIGIT)  
MINUSONE = "-1"  
EQUALS = "="  
COMMA = ","  
LSQUARE = "["  
RSQUARE = "]"  
LANGLE = "<"  
RANGLE = ">"  
LPAREN = "("  
RPAREN = ")"  
SLASH = "/"  
WSP = SP / HTAB

## Appendix A Call Flow for Network Announcement

This section provides an example call flow where a caller (MTA-o) invokes the "Last Number Redial" feature to determine the phone number of the dialing party (MTA-t). An Audio Server is used to play an announcement to the caller containing the previous caller's number and to present the option to the caller for completing a return call to MTA-t. It should be noted that this call flow, although a valid one, is merely an example that may or may not be used in practice.

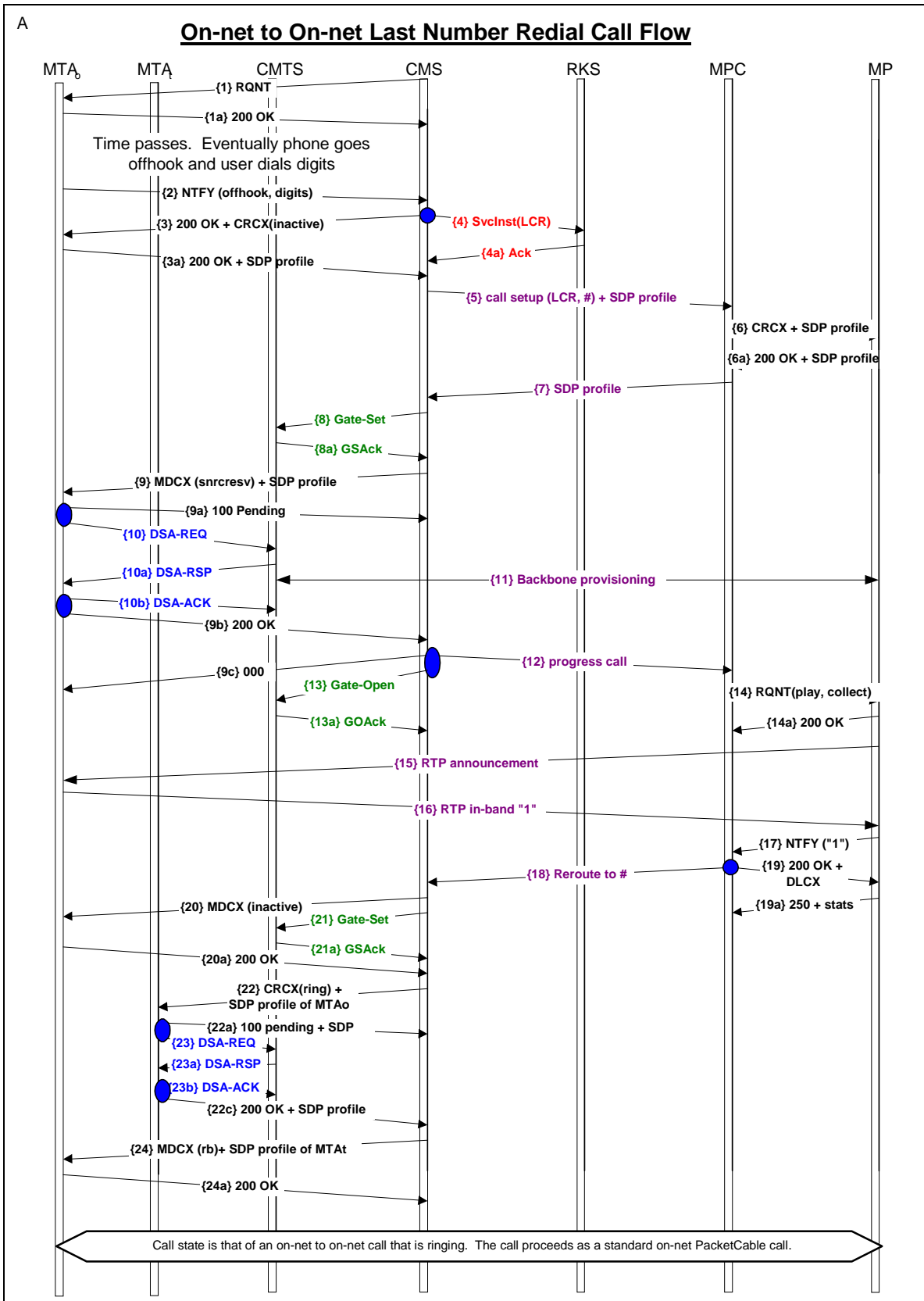


Figure 6. Call Flow for Network Announcement

**A.1 Call Flow Details**

Flow	Flow Description
1 <NCS>	<p>CMS sends MTA-o a NotificationRequest instructing MTA to look for an off-hook event, and to report it.</p> <p>RQNT 1201 aaln/1@ec-1.mso.net MGCP 1.0 NCS 1.0                      N: ca@ca1.mso.net:5678                      X: 0123456789AB                      R: hd(A, E(R(hu, [0-9# *T](D)),S(dl)))                      D: (0T   00T   303[2-9]xxxxxx   720[2-9]xxxxxx   1[2-9]xxxxxxxxxx   [3469]11   0[2-9]xxxxxxxxxx   01[2-9]xxxxxxxxxxxxxxT   011xxxxxxxxxxxxxxT)</p>
1a <NCS>	<p>MTA sends CMS an ACK in response to the command, repeating in the response the transaction id that the Call Agent attached to the query and providing a return code indicating success:</p> <p>200 1201 OK</p>
2 <NCS>	<p>MTA sends CMS a Notification message indicating that an off-hook was observed and that the user requested the phone number of the Last Call Received (LCR).</p> <p>NTFY 2001 aaln/1@ec-1.mso.net MGCP 1.0 NCS 1.0                      N: ca@ca1.mso.net:5678                      X: 0123456789AB                      O: *,6,9</p>
3 <NCS>	<p>CMS sends MTA an acknowledgement of the notification. Piggybacked on the acknowledgment, the CMS sends MTA-o a Create connection message. The connection is created in inactive mode. Packetization parameters are passed in the CRCX message.</p> <p>200 2001 OK</p> <p>.                      CRCX 1202 aaln/1@ec-1.mso.net MGCP 1.0 NCS 1.0                      C: A3C47F21456789F0                      L: p:10, a:PCMU, sc-rtp: 00/51; 62/51, sc-rtcp: 02/03; 01/03 sc-st: base64:</p> <p>pV6BIIHWt+0gDkpgnuxgTfROxYAemhYJTHWgHNt1crTtEUKFatJfSdEFVQueueo                      ==                      M: inactive                      N: ca@ca1.mso.net:5678                      X: 0123456789AC                      R: hu</p>
4 <Event Messages>	<p>CMS creates the BillingCorrelationID for this transaction.                      CMS sends RKS a Svcinst(LCR) Message.</p> <p>RADIUS Message Header:                      &lt;Code = Accounting-Request(1 octet, value = 4)&gt;                      &lt;Identifier (1 octet, value = 10)&gt;                      &lt;Length (2 octets, min value = 20, max value = 4096)&gt;                      &lt;Authenticator (16 octets, value = 0)&gt;</p> <p>PacketCable Event Message Header VSA:                      &lt;Type = vendor specific (1 octet, value = 26)&gt;</p>

Flow	Flow Description
	<p>&lt;Length (1 octet, value = ???)&gt;            &lt;vendor-ID = CableLabs (4 octets, value = 4491)&gt;            &lt;Vendor Attribute Type = Event Message Header (1 octet, value = 1)&gt;            &lt;Vendor Attribute Length (1 octet, value = 56)&gt;            &lt;Vendor Attribute Value =            &lt;Version ID = PacketCable 1.0 (2 octets, value = 1)&gt;            &lt;Billing Correlation ID (16 octets, value = TTTTXXXXXCMSCCCC)&gt;            &lt;Event Message Type = Call_Signaling_Start (2 octets, value = 1)&gt;            &lt;Element Type = CMS (2 octets, value = 1)&gt;            &lt;Element ID (8 octets, value = xxxxxCMS)&gt;            &lt;Sequence ID (4 octets, value = AA05)&gt;            &lt;Event Message Time and Date (17 octets, value = yyyyymmddhhmmss.mm)&gt;            &lt;Message Status = no known errors, message from trusted element (4 octets, value = ???)&gt;            &lt;Message Priority = user-defined (1 octet, value = any)&gt;            &lt;Attribute Count (2 octets, value = 4)&gt;            &lt;Event Object = reserved (1 octet, value = 0)&gt;            &gt;</p>
3a <NCS>	<p>MTA sends CMS an acknowledgement of the CRCX, adding its own SDP profile.</p> <pre> 200 1202 OK I: FDE234C8  v=0 o=- 25678 753849 IN IP4 128.96.41.1 s=- c=IN IP4 128.96.41.1 t=0 0 m=audio 3456 RTP/AVP 0 a=X-pc-csuites-rtp: 62/51 a=X-pc-csuites-rtcp: 02/03 01/03 a=X-pc-spi-rtcp: A7843B2 a=X-pc-secret: base64: pV6BIIHWt+0gDkpgnuxgTfROxYAemhYJTHWgHNt1crTtEUKFatJfSdEFVQueuo === </pre>
4a <EM>	<p>RKS sends CMS a RADIUS ACK in response to Service Instance message - Svcinst(LCR).</p> <p>RADIUS Message Header:            &lt;Code = Accounting-Response (1 octet, value = 5)&gt;            &lt;Identifier (1 octet, value = 10)&gt;            &lt;Length (2 octets, min value = 20, max value = 4096)&gt;            &lt;Authenticator (16 octets, value = 0)&gt;</p>
5 <proprietary>	<p>CMS sends MPC all call setup information (LCR, #) including MTA-o's SDP profile.            [proprietary]</p>
6 <ASP>	<p>MPC sends MP a CreateConnection request in send-receive mode.</p> <pre> CRCX 5050 ds/12/1@ec-2.mso.net MGCP 1.0 NCS 1.0 N:ca@ca2.mso.net:5678 C: A3C47F21456789F0 L: p:10, a:PCMU, dg-gi: 1273 sc-rtp: 62/51, sc-rtcp: 02/03; 01/03 sc-st: base64: pV6BIIHWt+0gDkpgnuxgTfROxYAemhYJTHWgHNt1crTtEUKFatJfSdEFVQueuo === </pre>

Flow	Flow Description
	<p>M: sendrcv                      X: 0123456789B0                      R: hd</p> <p>v=0                      o=- 25678 753849 IN IP4 128.96.41.1                      s=-                      c=IN IP4 128.96.41.1                      t=0 0                      m=audio 3456 RTP/AVP 0                      a=X-pc-csuites-rtp: 62/51                      a=X-pc-csuites-rtcp: 02/03 01/03                      a=X-pc-spi-rtcp: A7843B2                      a=X-pc-secret: base64:                      pV6BIIHWt+0gDkpgnuxgTfROxYAemhYJTHWgHNt1crTtEUKFatJfSdEFVQueueo                      ==</p>
<p>6a                      &lt;ASP&gt;</p>	<p>MP sends MPC an acknowledgement of receipt of Create Connection message.</p> <p>200 5050 OK                      K:                      I: 32F345E2                      DQ-RI:D32B8593</p> <p>v=0                      o=- 25678 753849 IN IP4 128.96.41.1                      s=-                      c=IN IP4 128.96.63.25                      t=0 0                      m=audio 1296 RTP/AVP 0                      a=X-pc-csuites-rtp: 62/51                      a=X-pc-csuites-rtcp: 02/03                      a=X-pc-spi-rtcp: 453A78F1                      a=X-pc-secret: base64:                      pV6BIIHWt+0gDkpgnuxgTfROxYAemhYJTHWgHNt1crTtEUKFatJfSdEFVQueueo                      ==</p>
<p>7&lt;proprietary&gt;</p>	<p>MPC sends CMS MP's SDP profile. [proprietary]</p>
<p>8                      &lt;DQoS&gt;</p>	<p>CMS sends CMTS a Gate-Set message including a local ID for use with gate coordination.</p> <p>Transaction ID – 3177                      Subscriber - MTA</p> <p>Remote Gate Info –                      CMS address – 128.96.22.15                      CMS Port - 2562                      Authentication Algorithm=0x64                      Security Key=FourScoreAndSevenYearsAgo                      Remote Gate ID – 8096</p> <p>GateSpec                      Direction upstream                      Protocol UDP                      SourceAddress 129.96.41.1 (MTA-o)                      DestinationAddress ????.????.????.??? (MG)</p>

Flow	Flow Description
	<p>SourcePort 0                      Destination Port 6540                      b 120                      r 12000                      p 12000                      m 120                      M 120                      R 12000                      S 0</p> <p>GateSpec                      Direction downstream                      Protocol UDP                      SourceAddress ????.????.????.??? (MG)                      DestinationAddress 129.96.41.1 (MTA-o)                      SourcePort 0                      Destination Port 3456                      b 120                      r 12000                      p 12000                      m 120                      M 120                      R 12000                      S 0                      Flag = Auto commit</p> <p>Billing Info –                      Billing Correlation ID – TTTTXXXXXCMSCCCC                      RKS_Primary - 128.96.60.110, 5000                      RKS_Secondary - 128.96.60.210, 5001                      Real_time_Flag - 0 (false)</p>
<p>8a                      &lt;DQoS&gt;</p>	<p>CMTS sends CMS an acknowledgment of the GateSet</p> <p>Transaction ID – 3177                      Subscriber - MTA                      Gate ID - 37125                      Activity Count - 2</p>

Flow	Flow Description
9 <NCS>	<p>CMS sends MTA-o an MDCX message. This message indicates that the MTA should go into send-receive mode. This message also contains the session description of the Media Player.</p> <pre>MDCX 1203 aaln/1@ec-1.mso.net MGCP 1.0 NCS 1.0 N:ca@ca1.mso.net:5678 C: A3C47F21456789F0 I: FDE234C8 M: sendrecv X: 0123456789AE R: hu L: dq-qi:37125  v=0 o=- 25678 753849 IN IP4 128.96.41.1 s=- c=IN IP4 128.96.63.25 t=0 0 m=audio 1296 RTP/AVP 0 a=X-pc-csuites-rtp: 62/51 a=X-pc-csuites-rtcp: 02/03 a=X-pc-spi-rtcp: 453A78F1 a=X-pc-secret: base64: pV6BIIHWt+0gDkpgnuxgTfROxYAemhYJTHWgHNt1crTtEUKFatJfSdEFVQueuo ==</pre>
9a <NCS>	<p>MTA-o sends CMS an acknowledgement of the MDCX message.</p> <pre>100 1203 PENDING</pre>
10 <DOCSIS>	<p>MTA-o sends CMTS a DSA request asking for bandwidth commitment in the access network.</p> <pre>DSAREQ TransactionID 1  UpstreamServiceFlow ServiceFlowReference 1 QoSParameterSetType Admitted(2) TimeoutAdmitted 200 ServiceFlowScheduling UGS(6) NominalGrantInterval 10ms ToleratedGrantJitter 2ms GrantsPerInterval 1 UnsolicitedGrantSize 111 AuthBlock 37125  DownStreamServiceFlow ServiceFlowReference 2 QoSParameterSetType Admitted(2) TimeoutAdmitted 200 TrafficPriority 5 MaximumSustainedRate 12,000 AuthBlock 37125  UpstreamPacketClassification</pre>



Flow	Flow Description
	<p>ServiceFlowReference 1                      PacketClassifierReference 1                      ClassifierPriority 150                      ClassifierActivationState Inactive (0)                      IPSourceAddress 128.96.41.1 (MTA)                      IPSourcePort 3456                      IPDestinationAddress ????.????.????.??? (MG)                      IPDestinationPort 6540                      IPProtocol UDP(17)</p> <p>DownstreamPacketClassification                      ServiceFlowReference 2                      PacketClassifierReference 2                      ClassifierPriority 150                      ClassifierActivationState Inactive (0)                      IPSourceAddress ????.????.????.??? (MG)                      IPDestinationAddress 128.96.41.1 (MTA)                      IPDestinationPort 3456                      IPProtocol UDP(17)</p>
<p>10a &lt;DOCSIS&gt;</p>	<p>CMTS sends MTA-o a DSA response indicating that the DSA request has been granted.</p> <p>DSARSP                      TransactionID 1                      ConfirmationCode Success(0)</p> <p>Upstream Service Flow                      ServiceFlowReference 1                      ServiceFlowID 1001                      QoSParameterSetType Admitted(2)                      TimeoutAdmitted 200                      ServiceFlowScheduling UGS(6)                      NominalGrantInterval 10ms                      ToleratedGrantJitter 2ms                      GrantsPerInterval 1                      UnsolicitedGrantSize 111                      AuthBlock 31001</p> <p>DownStreamServiceFlow                      ServiceFlowReference 2                      ServiceFlowID 2001                      QoSParameterSetType Admitted+Active(6)                      TimeoutAdmitted 200                      TrafficePriority 5                      MaximumSustaninedRate 12,000                      AuthBlock 32001</p> <p>UpstreamPacketClassification                      ServiceFlowReference 1                      PacketClassifierReference 1                      ClassifierID 3001                      ClassifierPriority 150                      CalssifierActivationState Inactive (0)</p>

Flow	Flow Description
	IPSourceAddress 128.96.41.1 (MTA) IPSourcePort 3456 IPDestinationAddress 128.96.63.25 (MG) IPDestinationPort 1296 IPProtocol UDP(17)  DownstreamPacketClassification ServiceFlowReference 2 PacketClassifierReference 2 ClassifierID 3002 ClassifierPriority 150 ClassifierActivationState Active (1) IPSourceAddress 128.96.63.25 (MG) IPDestinationAddress 128.96.41.1 (MTA) IPDestinationPort 3456 IPProtocol UDP(17)
10b <DOCSIS>	MTA-o sends CMTS an acknowledgement of the DSARSP.  DSA-ACK TransactionID 1 ConfirmationCode Success(0)
11 <proprietary>	Any backbone provisioning that is required is performed
9b <NCS>	MTA sends CMS a confirmation of transaction complete for MDCX.  200 1203 OK K:
9c <NCS>	CMS sends MTA an acknowledgement of the completion of the MDCX transaction.  000 1203
12 <proprietary>	CMS notifies the MPC to progress the call [proprietary]
13 <D-QoS>	CMS sends a GATE-OPEN message to CMTS GateOpen TransactionID – 81 Gate-ID - 37125
13a <D-QoS>	CMTS responds to GateOpen message  GateOpenAck TransactionID - 81
14 <ASP>	MPC sends MP a RQNT message to play the appropriate announcement and prompt for digit collection.  RQNT 5051 aaln/1@ec-1.mso.net MGCP 1.0 NCS 1.0 N: ca@ca1.mso.net:5678 X: 0123456789AB R: oc, of S: AAU/pc(ip=file://12345<5145551234>,file://34548 dm=x)
14a <ASP>	MP acknowledges receipt of RQNT from MPC  200 5051 OK

Flow	Flow Description
15 <ASP>	MP plays announcement to MTA-o via RTP media stream
16 <ASP>	In response to callers touch-tone, MTA-o sends MP a DTMF "1" via in-band signaling
17 <ASP>	MP sends MPC a Notification message indicating that a DTMF "1" was received.  NTFY 7070 aaln/1@ec-1.mso.net MGCP 1.0 NCS 1.0 N: ca@ca1.mso.net:5678 X: 0123456789AB O: oc(dc=1 na=1)
18 <proprietary>	MPC notifies CMS to re-route the call to the LCR #
19 <ASP>	MPC sends MP an acknowledgement of the NTFY and includes a piggybacked delete connection message.  200 7070 OK  .  DLCX 5052 aaln/1@ec-2.mso.net MGCP 1.0 NCS 1.0 C: A3C47F21456789F0 I: 32F345E2
19a <ASP>	MP sends MPC an acknowledgement of the DLCX and includes the call statistics collected by the MP.  250 5052 OK P: PS=1245, OS=62345, PR=780, OR=45123, PL=10, JI=27, LA=48
20 <NCS>	CMS sends MTA-o an MDCX message de-activating the connection.  MDCX 1204 aaln/1@ec-1.mso.net MGCP 1.0 NCS 1.0 N:ca@ca1.mso.net:5678 C: A3C47F21456789F0 I: FDE234C8 M: inactive X: 0123456789AF R: hu
20a <NCS>	MTA-o sends CMS an acknowledgement of the MDCX message.  200 1204 OK
21 <DQoS>	CMS sends CMTS a Gate-Set message including the local ID for use with gate coordination.  Transaction ID – 3177 Subscriber - MTA  Remote Gate Info – CMS address – 128.96.22.15 CMS Port - 2562 Authentication Algorithm=0x64 Security Key=FourScoreAndSevenYearsAgo Remote Gate ID – 8096  GateSpec Direction upstream Protocol UDP

Flow	Flow Description
	<p>SourceAddress 129.96.41.1 (MTA-o)                      DestinationAddress ????.????.????.??? (MG)                      SourcePort 0                      Destination Port 6540                      b 120                      r 12000                      p 12000                      m 120                      M 120                      R 12000                      S 0</p> <p>GateSpec                      Direction downstream                      Protocol UDP                      SourceAddress ????.????.????.??? (MG)                      DestinationAddress 129.96.41.1 (MTA-o)                      SourcePort 0                      Destination Port 3456                      b 120                      r 12000                      p 12000                      m 120                      M 120                      R 12000                      S 0                      Flag = Auto commit</p> <p>Billing Info –                      Billing Correlation ID – TTTTXXXXXCMSCCCC                      RKS_Primary - 128.96.60.110, 5000                      RKS_Secondary - 128.96.60.210, 5001                      Real_time_Flag - 0 (false)</p>
<p>21a &lt;DQoS&gt;</p>	<p>CMTS sends CMS an acknowledgment of the GateSet</p> <p>Transaction ID – 3177                      Subscriber - MTA                      Gate ID - 37125                      Activity Count - 2</p>
<p>22 &lt;NCS&gt;</p>	<p>CMS sends MTA-t a create connection message asking MTA-t to ring the phone.                      CRCX Includes the SDP profile of MTA-o.</p> <p>CRCX 1301 aaln/1@ec-2.mso.net MGCP 1.0 NCS 1.0                      C: A3C47F21456789F0                      L: p:10, a:PCMU, sc-rtp: 00/51; 62/51, sc-rtcp: 02/03; 01/03 sc-st: base64:</p> <p>pV6BIIHWt+0gDkpgnuxgTfROxYAemhYJTHWgHNt1crTtEUKFatJfSdEFVQueo                      ==                      M: inactive                      N: ca@ca1.mso.net:5678                      X: 0123456789AC                      R: hu                      S: rg</p>

Flow	Flow Description
	<p>v=0  o=- 25678 753849 IN IP4 128.96.41.1  s=-  c=IN IP4 128.96.41.1  t=0 0  m=audio 3456 RTP/AVP 0  a=X-pc-csuites-rtp: 62/51  a=X-pc-csuites-rtcp: 02/03  a=X-pc-spi-rtcp: A7843B2  a=X-pc-secret: base64:  pV6BIIHWt+0gDkpgnuxgTfROxYAemhYJTHWgHNt1crTtEUKFatJfSdEFVQueo  ==</p>
<p>22a  &lt;NCS&gt;</p>	<p>MTA-t sends CMS a confirmation of transaction complete for CRCX and it's SDP profile.</p> <p>100 1301 pending</p> <p>v=0  o=- 25678 753849 IN IP4 128.96.41.1  s=-  c=IN IP4 128.96.10.10  t=0 0  m=audio 6789 RTP/AVP 0  a=X-pc-csuites-rtp: 62/51  a=X-pc-csuites-rtcp: 02/03 01/03  a=X-pc-spi-rtcp: A7843B2  a=X-pc-secret: base64:  pV6BIIHWt+0gDkpgnuxgTfROxYAemhYJTHWgHNt1crTtEUKFatJfSdEFVQueo  ==</p>
<p>23  &lt;DOCSIS&gt;</p>	<p>MTA-t sends CMTS a DSA request asking for bandwidth commitment in the access network.</p> <p>DSAREQ  TransactionID 1</p> <p>Upstream Service Flow  Service Flow Reference 1  QoSParameterSetType Admitted(2)  TimeoutAdmitted 200  ServiceFlowScheduling UGS(6)  NominalGrantInterval 10ms  ToleratedGrantJitter 2ms  GrantsPerInterval 1  UnsolicitedGrantSize 111  AuthBlock 37125</p> <p>DownStreamServiceFlow  Service Flow Reference 2  QoSParameterSetType Admitted(2)  TimeoutAdmitted 200  TrafficePriority 5  MaximumSustainedRate 12,000  AuthBlock 37125</p>

Flow	Flow Description
	<p>UpstreamPacketClassification                      ServiceFlowReference 1                      PacketClassifierReference 1                      ClassifierPriority 150                      ClassifierActivationState Inactive (0)                      IPSourceAddress 128.96.41.1 (MTA)                      IPSourcePort 3456                      IPDestinationAddress ????.????.????.??? (MG)                      IPDestinationPort 6540                      IPProtocol UDP(17)</p> <p>DownstreamPacketClassification                      ServiceFlowReference 2                      PacketClassifierReference 2                      ClassifierPriority 150                      ClassifierActivationState Inactive (0)                      IPSourceAddress ????.????.????.??? (MG)                      IPDestinationAddress 128.96.41.1 (MTA)                      IPDestinationPort 3456                      IPProtocol UDP(17)</p>
<p>23a                      &lt;DOCSIS&gt;</p>	<p>CMTS sends MTA-t a DSA response indicating that the DSA request has been granted.</p> <p>DSARSP                      TransactionID 1                      ConfirmationCode Success(0)</p> <p>Upstream Service Flow                      ServiceFlowReference 1                      ServiceFlowID 1001                      QoSParameterSetType Admitted(2)                      TimeoutAdmitted 200                      ServiceFlowScheduling UGS(6)                      NominalGrantInterval 10ms                      ToleratedGrantJitter 2ms                      GrantsPerInterval 1                      UnsolicitedGrantSize 111                      AuthBlock 31001</p> <p>DownStreamServiceFlow                      ServiceFlowReference 2                      ServiceFlowID 2001                      QoSParameterSetType Admitted+Active(6)                      TimeoutAdmitted 200                      TrafficePriority 5                      MaximumSustainedRate 12,000                      AuthBlock 32001</p> <p>UpstreamPacketClassification                      ServiceFlowReference 1                      PacketClassifierReference 1                      ClassifierID 3001</p>

Flow	Flow Description
	ClassifierPriority 150 ClassifierActivationState Inactive (0) IPSourceAddress 128.96.41.1 (MTA) IPSourcePort 3456 IPDestinationAddress 128.96.63.25 (MG) IPDestinationPort 1296 IPProtocol UDP(17)  DownstreamPacketClassification ServiceFlowReference 2 PacketClassifierReference 2 ClassifierID 3002 ClassifierPriority 150 ClassifierActivationState Active (1) IPSourceAddress 128.96.63.25 (MG) IPDestinationAddress 128.96.41.1 (MTA) IPDestinationPort 3456 IPProtocol UDP(17)
23b <DOCSIS>	MTA-t sends CMTS an acknowledgement of the DSARSP.  DSA-ACK TransactionID 1 ConfirmationCode Success(0)
22c <NCS>	MTA-t sends CMS a 200 OK and its SDP profile.  200 1301 OK  v=0 c=IN IP4 128.96.63.25 m=audio 1296 RTP/AVP 0 a=X-pc-csuites-rtp: 62/51 a=X-pc-csuites-rtcp: 02/03

Flow	Flow Description
24 <NCS>	<p>CMS sends MTA-o an MDCX message indicating ringback and the SDP profile of MTA-t.</p> <pre> MDCX 1205 aaln/1@ec-1.mso.net MGCP 1.0 NCS 1.0 N:ca@ca1.mso.net:5678 C: A3C47F21456789F0 I: FDE234C8 M: sendrecv X: 0123456789AF R: hu S: rb  v=0 o=- 25678 753849 IN IP4 128.96.41.1 s=- c=IN IP4 128.96.10.10 t=0 0 m=audio 6789 RTP/AVP 0 a=X-pc-csuites-rtp: 62/51 a=X-pc-csuites-rtcp: 02/03 01/03 a=X-pc-spi-rtcp: A7843B2 a=X-pc-secret: base64: pV6BIIHWt+0gDkpgnuxgTfROxYAemhYJTHWgHNt1crTtEUKFatJfSdEFVQueo == </pre>
24a <NCS>	<p>MTA-o sends CMS an acknowledgement of the MDCX transaction.</p> <pre> 200 1205 OK </pre>
<p>Call State is a ringing on-net to on-net call between MTA-o and MTA-t. The call proceeds as a standard On-net to On-net PacketCable call.</p>	



## Appendix B Call Flow for MTA-stored Announcement

This section provides an example call flow where a User-1 attempts to call User-2. Due to facility problems on the terminating side the call can not be completed. The MTA associated with User-1 is instructed to play a local announcement. It should be noted, that this call flow, although a valid one, is merely an example that may or may not be used in practice.

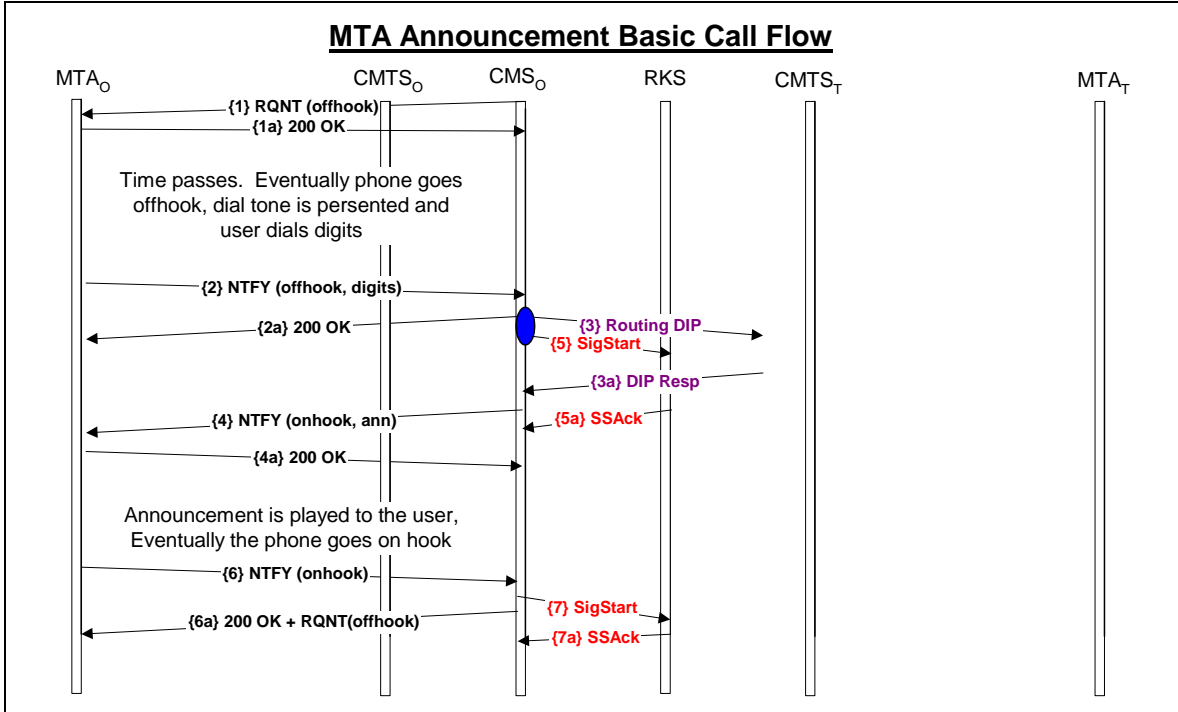


Figure 7. Call Flow for MTA-Stored Announcement

### B.1 Call Flow Details

Flow	Flow Description	Depends upon these completion of these flows:	Triggers start of these flows:
<b>Initialization</b>			
1 <NCS>	CMS sends MTAo a NotificationRequest instructing MTAo to look for an off-hook event, and to report it.  RQNT 1201 aaln/1@ec-1.whatever.net MGCP 1.0 NCS 1.X N: ca@ca1.whatever.net:5678 X: 0123456789AB R: hd(E (R([0-9#*T](D), hu(N)), S(dl), ;)) D: (0T   00T   [2-9]xxxxxxx   1[2-9]xxxxxxxxxx   011xx.T)		1a

Flow	Flow Description	Depends upon these completion of these flows:	Triggers start of these flows:
1a <NCS>	MTAo sends CMS an ACK in response to the command, repeating in the response the transaction id that the Call Agent attached to the query and providing a return code indicating success:  200 1201 OK	1	
<b>Service Request</b>			
2 <NCS>	MTAo sends CMS a Notification message indicating that an off-hook was observed.  NTFY 2001 aaln/1@ec-1.whatever.net MGCP 1.0 <b>NCS 1.X</b> N: ca@ca1.whatever.net:5678 X: 0123456789AB O: hd, 3, 0, 3, 5, 5, 5, 1, 2, 1, 2	1, user stimulus	2a, 3, 4, 5
2a <NCS>	CMS sends MTAo an acknowledgement of the notification.  200 2001 OK	2	
3 <??>	CMS contacts the routing database requesting a mapping of the dialed number to a routable destination in the network.	2	3a
3a <??>	The routing database server responds to the CMS with the routing information.	3	4, 8
4 <NCS>	CMS sends MTAo a notification request message. The connection is created in inactive mode. Packetization parameters are passed in the CRCX message.  RQNT 1202 aaln/1@ec-1.whatever.net MGCP 1.0 <b>NCS 1.X</b> N: ca@ca1.whatever.net:5678 X: 0123456789AC R: hu, oc, of S: A/ann(file://audio/23945)	2, 3a	4a, 5
4a <NCS>	MTAo sends CMS an acknowledgement of the RQNT, adding its own SDP profile.  200 1202 OK	4	6, 8

Flow	Flow Description	Depends upon these completion of these flows:	Triggers start of these flows:
<b>Announcement is being played</b>			
5 <EM>	<p>CMS creates the BillingCorrelationID for this transaction.</p> <p>CMS sends RKS a Call_Signaling_Start Event Message.</p> <p>The message contents include:                      Event_Message_Header(Version_ID, BillingCorrelationID, "Call_Signaling_Start Event Message", Element_Type, Element_ID, Element_Seq_Num, Message_Timestamp, Message_Status, Message_Priority, Attribute_Count, Event_Object ),                      Event_Time, MTA_Port_ID, Calling_Party_Number, Called_Party_Number</p> <p>The message format is:                      &lt;insert example coded message&gt;</p>	2	5a
5a <EM>	<p>RKS sends CMS a RADIUS ACK in response to Call_Signaling_Start</p> <p><b>ACK</b></p> <p>The message format is:                      &lt;insert example coded message&gt;</p>	5	
<b>User is listening to the announcement and hangs up</b>			
11 <NCS>	<p>MTAo sends CMS a notification that the attached device has gone on-hook.</p> <p>NTFY 2002 aaln/1@ec-2.whatever.net MGCP 1.0  <b>NCS 1.X</b>                      X: 0123456789AF                      O: hu</p>		12, 13, 14

Flow	Flow Description	Depends upon these completion of these flows:	Triggers start of these flows:
12 <NCS>	<p>CMS sends MTAo an acknowledgement of the NTFY and includes a piggybacked delete connection message.</p> <p>200 2002 OK</p> <p>.</p> <p>RQNT 1207 aaln/1@ec-2.whatever.net MGCP 1.0  <b>NCS 1.X</b>                      X: 0123456789B2                      N: ca@ca1.whatever.net:5678                      R: hd (E (dl:hu, D/[0-9# *T] (D) ;))                      D: (0T   00T   [2-9]xxxxxx   1[2-9]xxxxxxxxxx   011xx.T)</p>	11	12a, 15
12a <NCS>	<p>MTAo sends CMS an acknowledgement of the DLCX and includes the call statistics collected by the MTA.</p> <p>250 12?? OK</p>	12	22, 25
14 <EM>	<p>CMS sends RKS a Media_Connection_Stop Event Message.</p> <p>The message contents include:                      Event_Message_Header(Version_ID,                      BillingCorrelationID, "Media_Connection_Stop                      Event Message", Element_Type, Element_ID,                      Element_Seq_Num, Message_Timestamp,                      Message_Status, Message_Priority, Attribute_Count,                      Event_Object                      ),                      Event_Time, Call_Termination_Cause</p> <p>The message format is:                      &lt;insert example coded message&gt;</p>	11	14a
14a <EM>	<p>RKS sends CMS a RADIUS ACK in response to Media_Connection_Stop</p> <p><b>ACK</b></p> <p>The message format is:                      &lt;insert example coded message&gt;</p>	14	

## Appendix C Future Considerations

Future phases of PacketCable will consider the following enhancements:

- The CMS to MPC Interface will be specified. This interface is currently undefined. It is expected that this signaling interface will be based upon the PacketCable CMS-CMS signaling protocol specified in PacketCable 1.2.
- Further definition of MTA stored announcements. It is currently an optional requirement that the MTA store some simple announcements locally. It is expected that this will be REQUIRED MTA feature for future releases of PacketCable.
- Support for advanced IVR features, conferencing, and other capabilities and services that are available on network based Feature Servers may be added.

## APPENDIX D Acknowledgements

This specification was developed and influenced by numerous individuals representing many different vendors and organizations. PacketCable hereby wishes to thank everybody who participated directly or indirectly in this effort. In particular, PacketCable wants to recognize the following individuals for their significant involvement and contributions to this specification:

David Cromwell	Nortel Networks Cable Solutions
Rodney Green	Nortel Networks Cable Solutions
Don Stanwyck	IP Unity
Barry Hoffner	Telcordia
Bill Foster	Cisco
Roy Spitzer	Telogy/TI
Anthony Toubassi	Ericsson
David McIntosh	CableLabs
Glenn Russell	CableLabs
Keith Kelly	Netspeak
Stephane Proulx	Broadsoft
Tung-Hai Hsiao	AT&T
Orit Levin	Radvision
Michael Lynes	Radvision

## Appendix E Revision History

The following ECN has been incorporated into PKT-SP-ASP1.5-I02-070412.

ECN	ECN Date	Summary
ASP1.5-N-07.0395-2	3/12/07	Clarification of eCM DOCSIS versions