Metadata Specifications
Metadata 3.0

Asset Management Interface 3.0 Specification

MD-SP-AMIv3.0-I02-121210

ISSUED

DISCLAIMER

This document is published by Cable Television Laboratories, Inc. ("CableLabs®").

CableLabs reserves the right to revise this document for any reason including, but not limited to, changes in laws, regulations, or standards promulgated by various agencies; technological advances; or changes in equipment design, manufacturing techniques, or operating procedures described, or referred to, herein. CableLabs makes no representation or warranty, express or implied, with respect to the completeness, accuracy, or utility of the document or any information or opinion contained in the report. Any use or reliance on the information or opinion is at the risk of the user, and CableLabs shall not be liable for any damage or injury incurred by any person arising out of the completeness, accuracy, or utility of any information or opinion contained in the document.

This document is not to be construed to suggest that any affiliated company modify or change any of its products or procedures, nor does this document represent a commitment by CableLabs or any cable member to purchase any product whether or not it meets the described characteristics. Nothing contained herein shall be construed to confer any license or right to any intellectual property, whether or not the use of any information herein necessarily utilizes such intellectual property. This document is not to be construed as an endorsement of any product or company or as the adoption or promulgation of any guidelines, standards, or recommendations.

Document Status Sheet

Document Control Number:	MD-SP-AMIv3.0-I02-121210			
Document Title:	Asset Management Interface 3.0 Specification			
Revision History:	I01 - Released 4/11/11 I02 - Released 12/10/12			
Date:	December 10, 2012			
Status:	~~WIP~~	~~Draft~~	Issued	~~Closed~~
Distribution Restrictions:	~~Author Only~~	~~CL/Member~~	~~CL/ Member/ Vendor~~	Public

Key to Document Status Codes:

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

Contents

Tables

Figures

This page left blank intentionally.

1 SCOPE

1.1 Introduction and Overview

The Asset Management Interface (AMI) is the means by which assets (content as well as metadata describing the content or offer) are managed on the Asset Management System (AMS), which is a logical entity. It provides the following functions:

- Management of the asset metadata on the AMS, as well as management of the associated content.
- Notification of AMS events, including asset update and deletion and ContentAsset state changes related to content management.

The CableLabs Content 3.0 Specification [CONTENT 3.0] defines the asset metadata, while this document defines the transport and communication for managing asset and content metadata on the AMS.

1.2 Purpose of document

This document specifies standardized interfaces for managing asset metadata and associated content on the AMS. In addition to AMI, the Asset metadata and content files MAY be transmitted by other means mutually agreed upon, such as satellite, terrestrial transmission, or email.

This document does not describe application specific information, content characteristics, or business rules surrounding delivery.

1.3 Requirements

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

"MUST"	This word 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 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 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.

2 REFERENCES

2.1 Normative References

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.

[ADI 1.1] CableLabs Asset Distribution Interface Specification Version 1.1, MD-SP-ADI1.1-C01-120803, August 3, 2012, Cable Television Laboratories, Inc.

[CONTENT 1.1] CableLabs Video-on-Demand Content Specification Version 1.1, MD-SP-VOD-CONTENT1.1-C01-120803, August 3, 2012, Cable Television Laboratories, Inc.

[CONTENT 3.0] CableLabs Content 3.0 Specification, MD-SP-CONTENTv3.0-I02-121210, December 10, 2012, Cable Television Laboratories, Inc.

[XML] W3C REC: eXtensible Markup Language (XML) 1.1, February 4, 2004.

[RFC 2616] IETF RFC 2616. Hypertext Transfer Protocol – HTTP/1.1, June 1999.

[RFC 2617] IETF RFC 2617. HTTP Authentication: Basic and Digest Access Authentication, June 1999.

[RFC 3986] IETF RFC 3986. Uniform Resource Identifier (URI): Generic Syntax, January 2005.

2.2 Informative References

This specification uses the following informative references.

[CEP3.0] Content Encoding Profiles Specification, OC-SP-CEP3.0-I04-121210, December 10, 2012, Cable Television Laboratories, Inc.

2.3 Reference Acquisition

CableLabs Specifications:

- Cable Television Laboratories, Inc. (CableLabs), 858 Coal Creek Circle, Louisville, CO 80027, Phone 303-661-9100; Internet: http://www.cablelabs.com/projects/metadata
- Internet Engineering Task Force (IETF) Secretariat, 48377 Fremont Blvd., Suite 117, Fremont, California 94538, USA, Phone: +1-510-492-4080, Fax: +1-510-492-4001, http://www.ietf.org
- W3C, http://www.w3.org/

3 TERMS AND DEFINITIONS

Asset Management System	A role played by any logical entity functioning as the receiving point of a delivery
Asset	A uniquely identifiable collection of metadata with optionally associated content
Content	The data that is associated with a ContentAsset, typically some form of media such as video, audio, or an image
ContentAsset	An asset with an associated content in addition to metadata
Metadata	Descriptive information associated with an Asset
Provisioning	Configuration of a system component with specific parameters

4 ABBREVIATIONS AND ACRONYMS

AMI Asset Management Interface

AMS Asset Management System

HTTP HyperText Transfer Protocol

REST Representational State Transfer

URL Uniform Resource Locator

XML eXtensible Markup Language

5 ASSET MANAGEMENT INTERFACE

The Asset Management Interface is the means by which assets (content as well as metadata describing the content or offer) are managed on the AMS, which is a logical entity. It provides the following functions:

- Management of the asset metadata on the AMS, as well as management of the associated content.
- Notification of AMS events, including asset update and deletion and ContentAsset state changes related to content management.

The CableLabs Content 3.0 Specification [CONTENT 3.0] defines the asset metadata, while this document defines the transport and communication for managing asset and content metadata on the AMS.

The Asset Metadata Source, Event Notification Listener, and Content Source are also logical entities. In many cases, these logical entities are all performed by a single system operated by the Content Provider or Content Distributor. However, these logical entities may also be three separate systems as well.

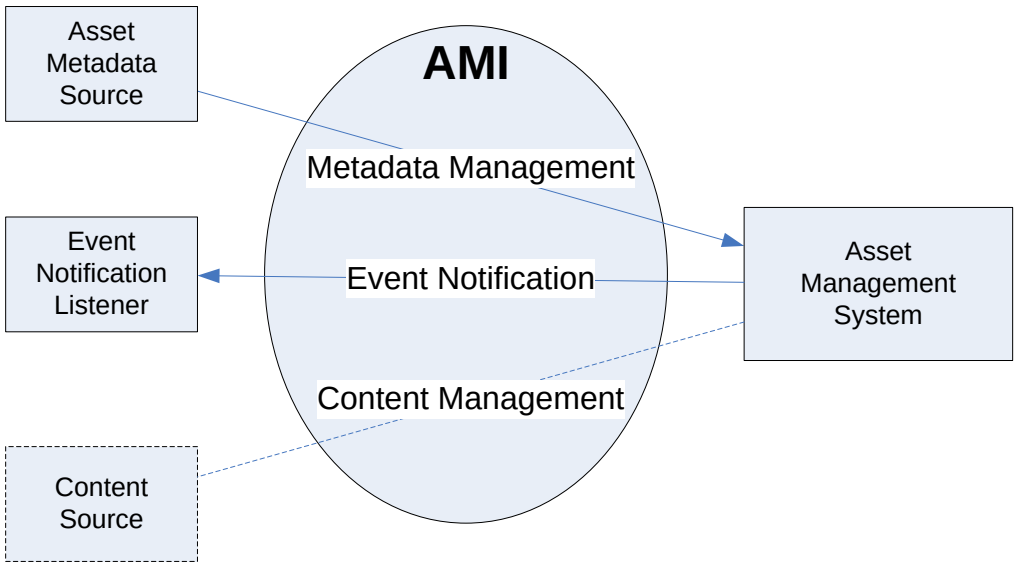

Figure 1 - Asset Management Interface

Only the management interfaces to a single virtual AMS are in scope. The AMI interfaces to a single logical AMS in the Service Provider's asset management network, and although many systems interact with the AMS from within the Service Provider's private network, a single view of assets in the AMS is presented over AMI. An Asset Metadata Source may interact with multiple AMSs, for instance, an AMS at each MSO, and receive notification from each AMS, but the AMI interface describes only the interaction between an Asset Metadata Source and a single AMS. An Event Notification Listener and Content source may likewise interact with one or more AMS. The term "Asset Source" may be used to collectively identify all three logical systems that are upstream of the AMS and use the AMI interface.

5.1 Asset Metadata Management

This interface is used to manage the Asset resource on the AMS, which indirectly manages the actual associated content on the AMS. This information may include the distribution policy (e.g., whether the content needs to be pre-positioned). This interface is also used to manage Bucket resources, which are logical Asset containers associated with the {ProviderId} within the AMS.

The following Asset functions are provided:

- Create an Asset. In the case of a ContentAsset, this announces the existence of the ContentAsset including the metadata (e.g., distribution policy information, SourceUrl) for the associated content to the AMS.
- Update an Asset in the AMS.
- List Assets (filtered by query parameters) known by the AMS.
- Get Asset metadata (including State for ContentAsset) of an Asset in the AMS.
- Delete an Asset from the AMS.
- Bulk process of creation/update of Assets on the AMS.
- Ping the AMS to monitor health of AMS service.

The following Bucket functions are provided:

- Create a Bucket in the AMS.
- Update the Bucket metadata for a Bucket in the AMS.
- List Buckets.
- Get Bucket metadata of a Bucket in the AMS.
- Delete a Bucket from the AMS.

The message body of a PUT to create a ContentAsset contains metadata that describes attributes of the content and a SourceUrl from which the content will be fetched via the Content Management Interface.

The interface is Restful and is based on HTTP 1.1 and HTTPS as specified in [RFC 2616], [RFC 2617], and [RFC 3986]. Table 1 identifies the HTTP methods that are used.

Table 1 - HTTP Methods for Asset Management Interface

Function Name	Method	Request Message Information	Status Code	Response Message Information
Create Asset	PUT	Path: assets/{ProviderId}/{AssetId} Body: Asset metadata	201 Created 409 Conflict 400 Bad Request	Body: Asset metadata Header: ETag
Update Asset	PUT	Path: assets/{ProviderId}/{AssetId} Header: If-Match Body: Asset metadata	200 OK 404 Not Found 421 Precon Fail 400 Bad Request	Body: Asset metadata Header: ETag
List Assets	GET	Path: assets?query	200 OK	Body: AssetList metadata
Get Asset	GET	Path: assets/{ProviderId}/{AssetId} Header: If-None-Match	200 OK 304 Not Mod 404 Not Found	Body: Asset metadata Header: ETag
Delete Asset	DELETE	Path: assets/{ProviderId}/{AssetId} Header: If-Match	204 No Content 404 Not Found 421 Precon Fail	
Bulk Assets	POST	Path: assets Body: AssetList metadata	200 OK	Body: AssetList metadata

Function Name	Method	Request Message Information	Status Code	Response Message Information
Ping	HEAD	Path: assets	200 OK 5XX	Body: ErrorList (5XX response)
Create/ Update Bucket	PUT	Path: assets/{ProviderId} Body: Bucket metadata	200 OK 404 Not Found 4xx	Body: Bucket metadata
List Buckets	GET	Path: assets	200 OK	Body: BucketList metadata
Get Bucket	GET	Path: assets/{ProviderId}	200 OK 404 Not Found	Body: Bucket metadata
Delete Bucket	DELETE	Path: assets/{ProviderId}	204 No Content 404 Not Found 421 Precon Fail	

5.2 Query parameters for List of Assets

A listing of Asset may be requested with an HTTP GET message that contains the ProviderId as a basic filtering on the query. This message may also contain a query-string that identifies (or establishes filter criteria for) the specific items to be returned in the response message body. If the query-string is not present, information for all Asset associated with the ProviderId is returned.

The query string parameters are used to scope the list requests, and follow the HTTP path portion of the request using the following standard form:

assets?*parameter*=*value*&*parameter*=*value*...

Table 2 - Query Parameters for List of Assets

Field	Request-URI	Required
start	Specifies the asset uriId after which the returned list should start.	No
providerId	Limit the results to only assets which have a matching providerId portion of the uriId.	No
offset	Specifies the starting item in the return list (zero-based list). If no offset is specified, the beginning of the list is the first item in the return list.	No, default 0
order	Specifies the order of the returned list by uriId, lastModifiedDateTime, assetType, or state. If not specified, the returned list is ordered by uriId.	No, default uriId
desc	Specifies if the result list order should be descending. If not specified or desc=true, results are in descending order, if desc=false, then results are in ascending order.	No, default to true
max	The maximum number of objects to return in a list object request. The greatest value this can have is 1000, and the maximum value of 1000 is assumed if no max is specified.	No, default to 1000

Field	Request-URI	Required
assetType	Limit the results to assets with of the specified AssetType (e.g., Movie, Preview, Poster, Trickfile, Content). Multiple values may be present.	No
state	Limit the results to assets with the specified state. Multiple values may be present.	No
modifiedAfter	Limit the results to assets which have changed after the specified timestamp.	No
detail	Level of detail: full, summary, list. full includes all asset metadata; summary includes uriId, ETag, state; list includes only uriId.	No, default to summary
EIDR	The results will include assets with an EIDR (Entertainment Identifier Registry) alternateId exactly matching the value specified.	No, support for this parameter is optional for an AMS.
ISAN	The results will include assets with an ISAN (International Standard Audiovisual Number) alternateId exactly matching the value specified.	No, support for this parameter is optional for an AMS.
ADID	The results will include assets with an AD-Id alternateId exactly matching the value specified.	No, support for this parameter is optional for an AMS.
ISRC	The results will include assets with an ISRC (International Standard Recording Code) alternateId exactly matching the value specified.	No, support for this parameter is optional for an AMS.
ISCI	The results will include assets with an ISCI (Industry Standard Commercial Identifier) alternateId exactly matching the value specified.	No, support for this parameter is optional for an AMS.
VOD11	The results will include assets with a VOD11 alternateId exactly matching the value specified.	No, support for this parameter is optional for an AMS.

5.3 Content Management

The Asset Management Interface allows for asset metadata management within the AMS, and indirectly the management in the AMS of content related to ContentAsset. Content management is always asynchronous with respect to the associated asset metadata management of ContentAsset.

When asset metadata for a ContentAsset is managed on the AMS, the AMS becomes responsible for the retrieval and distribution of the associated content, but the AMS ultimately determines when the Content is pulled to the AMS. The decision to pull the Content is directly influenced by the ContentAsset SourceUrl as well as the distribution policy (e.g., PropagationPriority, 1 to 10) metadata. When the distribution policy indicates that the Content is to be pre-positioned, the AMS prioritizes the retrieval of the content. Depending on distribution policy, AMS resources, and other configurable factors, the AMS may defer retrieval to a later time or even until a user request for the content. The Asset Source should ensure that the content is available at the SourceUrl as long as it is referenced in any metadata in the AMS.

Once the AMS has determined to perform a management activity on the Content, the manner in which the Content is retrieved depends on the SourceUrl metadata that specifies the transport protocol for the Content. Support for

HTTP/HTTPS URL protocols is mandatory, but the additional URL schemas are readily supported with mutual agreement between the AMS and the content source.

Unicast pull of the Content to the AMS is indicated using a SourceUrl with a protocol of http/https to reference resources stored anywhere in the World Wide Web. If the SourceUrl protocol is http/https, the AMS uses the HTTP 1.1 GET method defined in [RFC 2616] to retrieve (pull) content from the asset source for delivery to the AMS. The AMS must support the content pull mechanism.

In addition to pulling content using the SourceUrl, the AssetSource may push the Content files over AMI at the discretion of the AMS. Content in the AMS is identified by a unique URI just as assets are, and the AMI interface to create, update, get, delete, and list assets can also be applied to the binary content resources. For pull content management, the SourceUrl is provided in the metadata, and the AMS assigns the unique ContentRef (this Content URI is associated with the ContentAsset), while for push content management the client uses a PUT operation to create the Content directly (with the request body containing the content bits) and assigns the created URI directly to the ContentAsset ContentRef metadata. AMS redirection of the content PUT to another endpoint should also be supported by the AMI client. The ContentRef metadata is a unique identifier to the Content in the AMS domain, and systems in the AMS domain can utilize the ContentRef to access and manage the content.

5.4 Event Notification

Each Asset has a state attribute which indicates the current status of the Asset on the AMS, and optionally a stateDetail attribute which provides additional state information. Event notification is based on changes to an asset which may occur as a result of AMI operations or other AMS processing activities. The notifyURI attribute, if present, specifies an URI to which event notification messages should be sent. This notifyURI is implemented on the Event Notification Listener, which is often an endpoint on the Asset Source, but may be any arbitrary URL on any host. Event notification for an asset are only sent if the notifyURI if present.

For simple assets (assets which are not ContentAsset or Offers) the Asset state is trivial. In this case, the AMS assigns an initial state of either "Provisioned" or "Verified" depending on AMS business rules. If set to "Provisioned", then the stateDetail attribute may indicate additional detail indicating the reason that the Asset state cannot be verified immediately. AMS processing at some time after the AMI create/update response will attempt to verify the completeness and readiness of the Asset according to AMS business rules and set the state to "Verified". Deletes of an Asset will trigger an event notification showing a state of "Deleted", even though technically the Asset no longer exists on the AMS and therefore has no real state. The state transition diagram for a simple Asset is given in Figure 2.

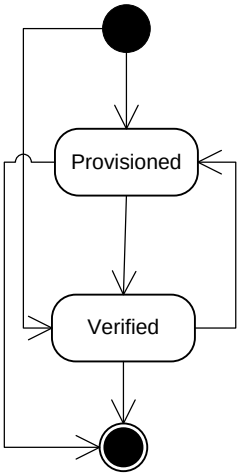

Figure 2 - State Transition Diagram for a Simple Asset

Provisioned - The Asset exists and the metadata has been saved, but the Asset has not been verified.

Verified - The Asset has been verified, and the Asset is complete and ready according to AMS business rules.

Deleted - An Asset has been deleted from the AMS.

For Offers, the simple Asset state is extended to also indicate the availability of the Offer to the end user. In this case, the AMS may assign an initial state of "Provisioned", "Verified", or "Available" depending on AMS business rules. If set to "Provisioned", then the stateDetail attribute may indicate additional detail indicating the reason that the Asset state cannot be verified. AMS processing at some time after the AMI create/update response will attempt to verify the completeness and readiness of the Asset according to AMS business rules and set the state to "Verified". AMS background processing will also attempt to place verified Offers into the "Available" state. In general, the state of "Available" indicates availability of the Offer to end users on all required platforms and applications, but additional information on the availability may be indicated in the stateDetail attribute. Deletes of an Asset will trigger an event notification showing a state of "Deleted", even though technically the Asset no longer exists on the AMS and therefore has no real state. The state transition diagram for an Offer is given in Figure 3.

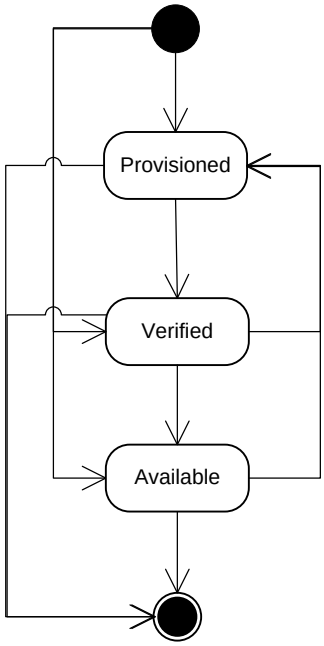

Figure 3 - State Transition Diagram for Offer

Provisioned - The Offer exists and the metadata has been saved, but the Offer has not been verified.

Verified - The Offer has been verified, and the Offer is complete and ready according to AMS business rules.

Available - The Offer is available to the end user according to AMS business rules.

Deleted - An Asset has been deleted from the AMS.

For ContentAsset, the simple Asset state is extended to also represent the state of the Content associated with the ContentAsset. The AMS may perform content management activities of Content associated with ContentAsset independently for the AMI calls to the AMS, and supports Content Pull and Content Push modes as well as other mechanisms based on the ContentAsset metadata. The AMS may assign an initial state of "Provisioned", "Verified", or "Failed" depending on AMS business rules. If the state is set to "Processing" or "Failed", then the stateDetail attribute may indicate additional content processing detail. AMS processing at some time after the AMI create/update response will attempt to move the ContentAsset state from "Provisioned" to "Processing", and when the Content transfer is complete, to verify the completeness and readiness of the ContentAsset according to AMS business rules and set the state to "Verified". The state transition diagram for ContentAsset is given in Figure 4.

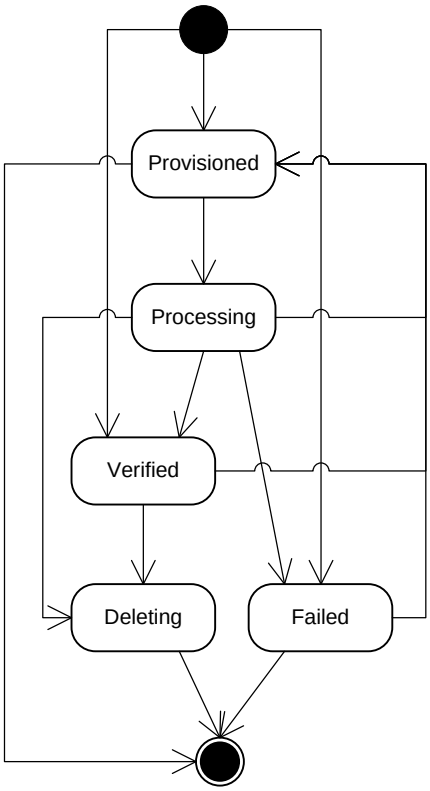

Figure 4 - State transition diagram for ContentAsset State

Provisioned - The ContentAsset exists and the metadata has been saved, but the Asset has not been verified and Content processing has not yet started.

Processing - The Content is being processed.

Verified - The Asset has been verified, and the Asset is complete and ready according to AMS business rules, including any business rules related to the Content associated with the ContentAsset.

Failed - The Content transfer to the AMS failed.

Deleting - The ContentAsset is being removed from the AMS.

Deleted - An Asset has been deleted from the AMS.

Depending on the Content Management approach (push, pull), the state transitions behave differently. For the pull scenario (ContentRef assigned by AMS), the initial State of "Provisioned" is assigned when the ContentAsset is created. While the AMS pulls the Content using the SourceUrl the State will be "Processing", and when successfully completely transferred the State will be "Verified". For the push scenario (ContentRef assign by Asset Source), the AMS is not responsible for initiating Content transfer. The AMS will use the ContentRef to determine if the Content exists in the AMS and the initial State will be either "Verified" if present or "Failed" if not present.

If notifyURI is present on an Asset, the AMS is required to send an HTTP POST to the notifyURI for each state change event. The POST body will consist of a list of metadata for the Asset for each state change event, which may be grouped together by the AMS when reported for efficiency. Because the body may consist of metadata for multiple ContentAsset, the POST URI does not include a ContentAsset URI identifier. Instead, each ContentAsset metadata includes an URI attribute to indicate to which ContentAsset the status change is related. The AMS should provide a retry mechanism with an error reporting mechanism if the retry mechanism is exhausted. The AMS is ultimately responsible for reporting the state change or escalating the failure to do so.

Table 3 - HTTP Methods for State Change Notification

Function Name	Method	Request Message Information	Status Code	Response Message Information
Event notification	POST	Path: notifyURI Body: List of Asset metadata for each event	204 No Content	No Content

It should be noted that the use of event notification and the notifyURI is entirely optional, and that an alternative is for the Asset Source to determine the state of assets and content by periodically calling the Asset Metadata Management "Get" and "List" operations.

5.5 HTTP Conditional Access (ETag)

The AMI requires the use of Entity Tags (ETag), which are values associated with a specific version of an Asset to support "Conditional GET" and "Conditional PUT" as defined in [RFC 2616]. This HTTP 1.1 caching mechanism saves bandwidth, and minimizes client and server process, allowing the client to make a request "give me a resource representation if it has changed", or "update the resource only if it has not changed since I read it". HTTP ETags are to be used to avoid race conditions and stale cache copies of representations. For example, a conditional GET sends the ETag value for the Asset using the If-None-Match header.

```
GET /cp1.com/asset123 HTTP/1.1
If-None-Match: "e180ee84f0671b1"
```

If the Asset has not been modified, the response will be a status code of 304 ("Not Modified"). This allows the client to determine whether it still has the most recent representation at the time of editing.

```
HTTP/1.1 304 Not Modified
Date: Sat, 24 Feb 2007 13:17:11 GMT
```

After editing the representation received by a GET (with corresponding ETag), the client can conditionally PUT the Asset and send the ETag entity value in an If-Match header, informing the server to accept the Asset on the condition that the entity value sent still matches the server's.

```
PUT /cp1.com/asset123 HTTP/1.1
Content-Length: nnn
If-Match: "e180ee84f0671b1"

<?xml version="1.0" ?> …
```

If the server has since received a more recent copy than the client's, it responds with a status code of 412 ("Precondition Failed"). This informs the client that the server has a more recent version of the Entry and will not allow the sent entity to be stored.

```
HTTP/1.1 412 Precondition Failed
Date: Sat, 24 Feb 2007 16:34:11 GMT
```

5.6 Authentication

Authentication according to [RFC 2616] and [RFC 2617] should be supported. HTTPS Digest authentication is recommended. It is recommended that the AMI interface operates over Virtual Private Networks (VPN) or dedicated/protected networks, or that similar network technologies are used to meet network security requirements.

5.7 Authorization

It is recommended that the AMS implementation associate an authenticated user with authorization information, such that access is restricted to only those operations/assets/buckets which are appropriate for the user. An authenticated user may have authorization to multiple buckets, and it is recommended that the buckets in which an asset is contained be a primary determining factor for authorization to assets.

5.8 Metadata Usage by Method

The Asset metadata is based on [CONTENT 3.0], and is used in multiple locations in the Asset Management Interface. The elements required differ based on the usage and the current state of the ContentAsset. Elements descriptions and required vs. optional usage is as follows.

Table 4 - ContentAsset Metadata Requirements by Method

Element	Create/ Modify Request Body	Get (status<>Verified) Response Body	Get (status=Verified) Response Body	Status Notification Request Body	Bulk update/create Request Body
@urlid	n/a[1]	n/a[1]	n/a[1]	Req[2]	Req[2]
ContentRef	Opt.	Opt.	Opt.	Opt.	Opt.
SourceUrl	Opt	Opt.	Opt.	Opt.	Req.
ContentFileSize	Opt.	Opt.	Req.	Opt.	Opt.
ContentCheckSum	Opt.	Opt.	Opt.	Opt.	Opt.
BitRate[3]	Opt.	Opt.	Req.	Opt.	Opt.
Duration[3]	Opt.	Opt.	Req.	Opt.	Opt.
@notifyURI	Opt.	Opt.	Opt.	Opt.	Opt.
@state	n/a	Req.	Req.	Req.	n/a
@stateDetail	n/a	Opt.	Opt.	Opt.	n/a

Note 1: URI is in the request and is not required as an attribute of the Asset metadata.
Note 2: The Request URI does not include an URI identifier. Instead, each Asset metadata includes an @uriId attribute.
Note 3: AudioVideoAssets only.

The Bucket metadata is also based on [CONTENT 3.0]. Buckets are logical Asset containers maintained within the AMS. Each Bucket is associated with a {ProviderId}, and all Assets are managed within the Bucket indicated by the {ProviderId} portion of the Asset UriId. Buckets are created, updated, listed, retrieved, and deleted separately from Assets, and Bucket metadata from [CONTENT 3.0] is maintained for each.

5.9 Ping Function and Health Monitoring

The light-weight Ping function can be called by clients to determine the current operational health of the AMS. A 200 OK result indicates that the AMS is operational, while an error response will be accompanied by an XML body including information related to the AMS error conditions.

5.10 Relation to CableLabs ADI 1.1 and VOD1.1

CableLabs previously defined the Asset Distribution Interface specification [ADI 1.1] that has been used worldwide to support the distribution of Video-On-Demand content. This AMI specification is intended to replace that specification when Metadata 3.0 xml metadata is used, and provides for more functionality, flexibility, and scalability than ADI1.1.

Metadata 3.0 [CONTENT 3.0] was designed to be able to support lossless translations of VOD 1.1 [CONTENT 1.1] packages into Metadata 3.0 Offers, as well as predictable translation of Metadata 3.0 offers into VOD 1.1 packages for backward compatibility. This ability to translate between VOD 1.1 metadata and Metadata 3.0 metadata supports systems supporting both the old and new standards going forward.

While an AMS implementing this specification MUST accept [CONTENT 3.0] assets, it SHOULD also continue to support the [ADI 1.1] interface to provide backward compatibility with legacy Asset Sources. When metadata is submitted to an AMS using [ADI 1.1], the AMS is expected to immediately transform the metadata to [CONTENT 3.0] asset equivalent and process as for any other request (possibly using the Bulk function). The AMS may likewise provide support for pre-existing update and delete workflows by detecting the [ADI 1.1] Verb attribute. The AMS SHOULD NOT provide status notification messages for Asset Sources provided using [ADI 1.1].

An existing [ADI 1.1] Asset Source may easily enjoy the benefits of status notification by pre-transforming metadata submissions to [CONTENT 3.0] and sending them to the AMS using the interface defined in this specification. The Asset Source will need to provide a notifyURI attribute for each asset as part of the transformation. The Asset Source will also need to properly handle E-tags as defined in Section 5.4 in order to request update and delete operations.

6 USE CASES (INFORMATIVE)

The following sequence diagrams illustrate significant use cases.

6.1 Create an Asset

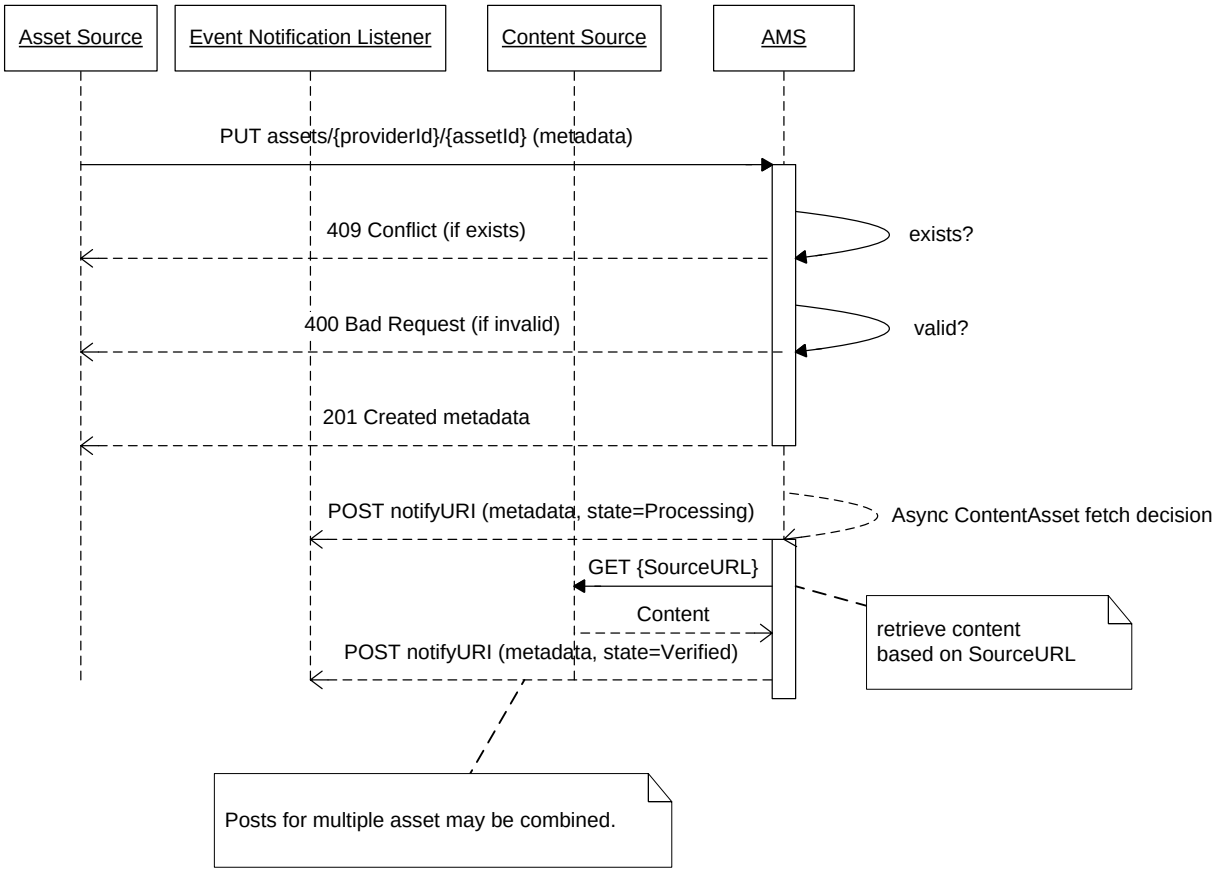

Figure 5 - Create an Asset

Assets are created or modified by issuing a PUT message with the uriId of the asset to the AMS with XML in the request message body that describes the asset metadata. Whether the PUT is a 'create' or an 'update' depends on the existence of the If-Match header - create does not include the If-Match header and update does.

If the If-Match header is not present, then the PUT request is a create operation. If an asset already exists for the uriId, then a 409 Conflict is returned. If the uriId referenced does not yet exist in the AMS, the metadata is validated, and if invalid, then 400 Bad Request is returned. Otherwise, the Asset is persisted in the AMS and the result is returned. When a pull approach is used for Content Management, the Content will not yet be on the AMS and State of created ContentAsset (e.g., Movie, Preview, Poster) has an initial value of "Provisioned". If a push approach to Content Management is used and the Content indicated by the ContentRef metadata is already present on the AMS, then the "Verified" state will be assigned.

The subsequent processing by the AMS depends on the asset type as described in Section 5.4. For instance, a ContentAsset will have further processing related to the management of the associated Content. Depending on the Content Management approach (push, pull), the State transitions behave differently. For the pull scenario (ContentRef assigned by AMS), the initial State of "Provisioned" is assigned when the ContentAsset is created.

While the AMS pulls the Content using the SourceUrl the State will be "Processing", and when successfully completely transferred the State will be "Verified". For the push scenario (ContentRef assign by Asset Source), the AMS is not responsible for initiating Content transfer. The AMS will use the ContentRef to determine if the Content exists in the AMS and the initial State will be either "Verified" if present or "Failed" if not present.

Asynchronously to the PUT call to create the ContentAsset, the AMS makes a decision to acquire the Content associated with the ContentAsset. If the notifyURI is specified on the ContentAsset, the event associated with the state change to "Processing" will trigger a POST to the notifyURI with this state change information, and similarly once the transfer is complete a notification is sent to the notifyURI with the State change of "Verified". Both of these notifications may be delivered in bulk.

6.2 Update an Asset

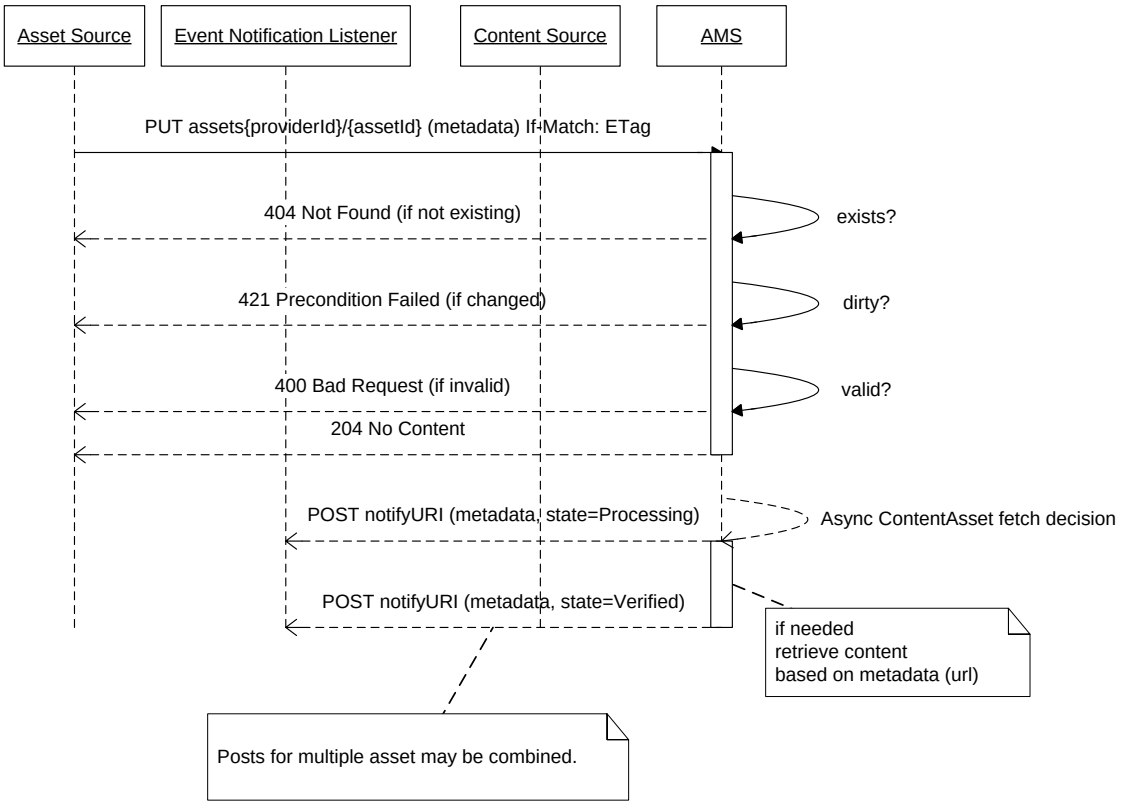

Figure 6 - Update an Asset

Asset are created or modified by issuing a PUT message with the uriId of the asset to the AMS with XML in the request message body that describes the asset metadata. Whether the PUT is a 'create' or an 'update' depends on the existence of the If-Match header - create does not include the If-Match header and update does.

If the If-Match header is present, then the PUT request is an update operation. If an asset does not already exist for the uriId, then 404 Not Found is returned. If the uriId referenced does exist on the AMS, the precondition is checked. This consists of the AMS comparing the ETag specified with the If-Match header with the current ETag on the AMS to determine if the caller has the most recent version of the Asset. If the ETag does not match, then the precondition check fails; otherwise the metadata is validated. If the metadata is invalid, then 400 Bad Request is returned; otherwise the Asset is updated and persisted on the AMS and the result is returned.

For an Asset that is NOT a ContentAsset (just metadata without a directly associated content), no subsequent processing is needed and no event notifications are sent. For a ContentAsset, the AMS will have further processing

related to the management of the associated Content if the State has changed. Asynchronously to the PUT call to create the ContentAsset, the AMS makes a decision to alter the State and perform additional management for the Content associated with the ContentAsset. If the notifyURI is specified on the ContentAsset, the event associated with the state change will trigger a POST to the notifyURI with this state change information, and similarly once the transfer is complete a notification is sent to the notifyURI with the State change of "Verified". Both of these notifications may be delivered in bulk.

This sequence shows notification of a state change to "Processing", due to an updated SourceUrl, re-retrieval of the Content, and notification of a state change to "Verified" for illustration purposes. Other use cases, including no further action or state change to other states such as "Provisioned" or "Failed", are also possible.

6.3 Delete an Asset

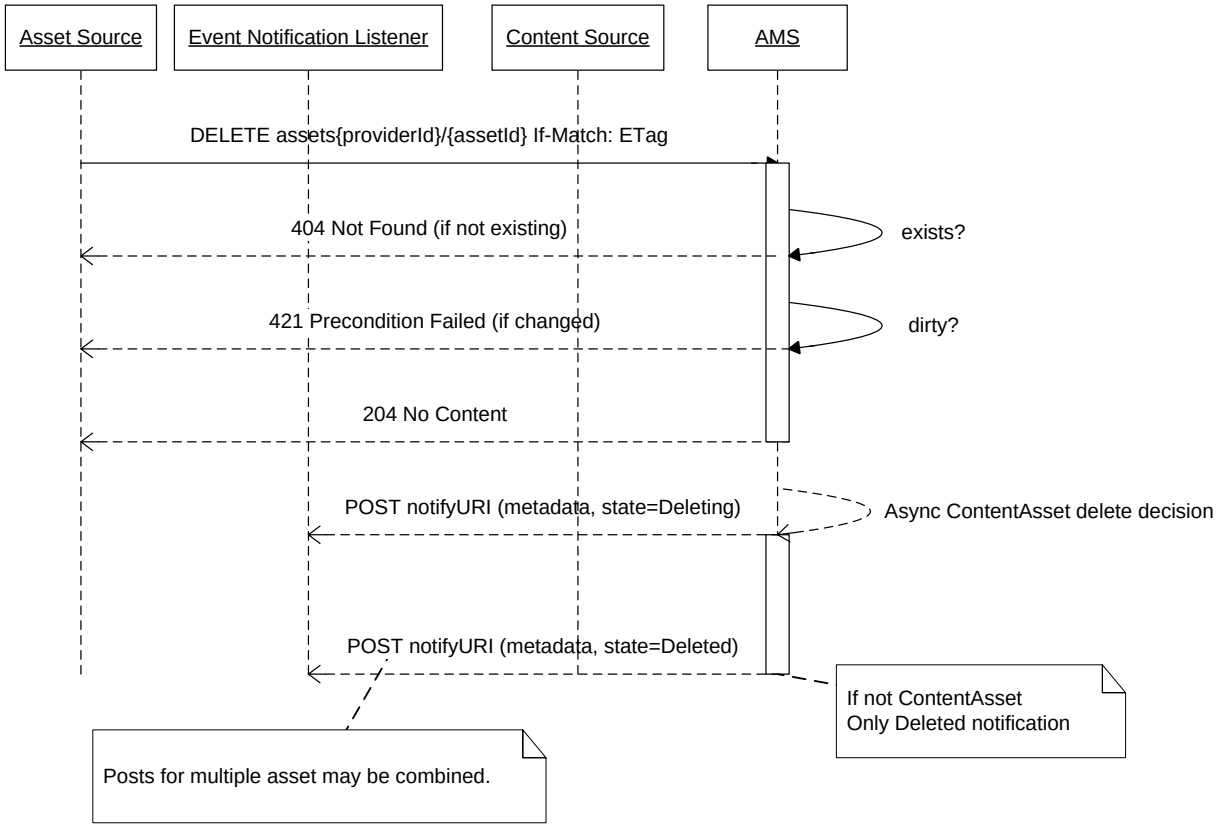

Figure 7 - Delete an Asset

An Asset is deleted by issuing a DELETE message with the uriId of the asset to be deleted to the AMS. The If-Match header is included. If an asset does not already exist for the uriId, then 404 Not Found is returned. If the uriId referenced does exist on the AMS, the precondition is checked. This consists of the AMS comparing the ETag specified with the If-Match header with the current ETag on the AMS to determine if the caller has the most recent version of the Asset. If the ETag does not match, then the precondition check fails; otherwise the AMS proceeds with deleting the Asset. The server results a 204 (No Content) to indicate successful processing of the delete message, which is similar to 200 (OK) but with no entity-body returned in the result.

For an Asset that is NOT a ContentAsset (just metadata without a directly associated content), the Asset is removed immediately, and an asynchronous POST notification is sent. For a ContentAsset, the AMS will have further processing related to the management of the associated Content. Asynchronously to the DELETE call, the AMS

places the Asset in a "Deleting" state. If the notifyURI is specified on the ContentAsset, the event associated with the "Deleting" state change will trigger a POST to the notifyURI. Once the associated Content has been removed from the AMS, the ContentAsset will be deleted and the delete event notification is sent. Both of these notifications may be delivered in bulk.

As a result of a deletion, associated non-content assets may also change state.

In should be noted that the AMI delete message may be treated as a logical delete in an AMS implementation, and that the AMS may retain Assets and Content that have been deleted (both metadata and content files) in accordance with business agreements. However, when the AMI delete message is processed by the AMS, all subsequent interaction with the AMS over AMI will behave as though the Asset and any corresponding Content have been removed from the AMS.

6.4 Retrieve Metadata for an Asset

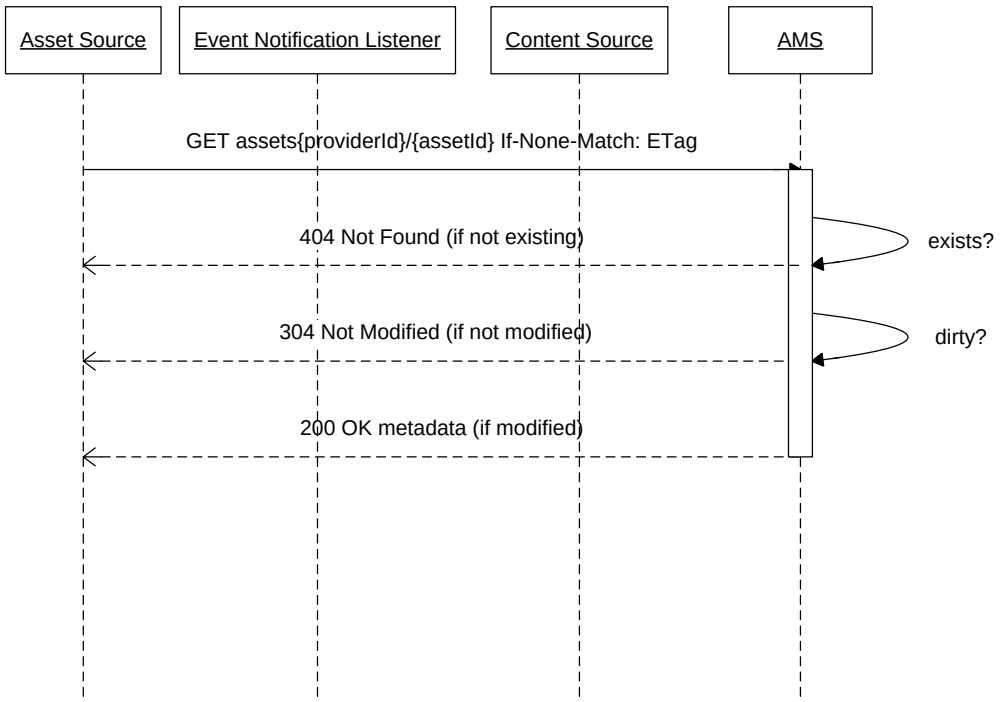

Figure 8 - Retrieve Metadata for an Asset

The Asset Source wishes to obtain details about an Asset. If the client already has obtained the Asset metadata with corresponding ETag, then the ETag is supplied in the If-None-Match header. If it is the first attempt to get the Asset metadata, then no If-None-Match header is included. If the ETag is supplied and the value on the AMS matches the supplied value, then a 304 Not Modified is returned with no body. Otherwise, a 200 OK with the asset metadata in the body is returned.

6.5 Retrieve (filtered) List of Assets

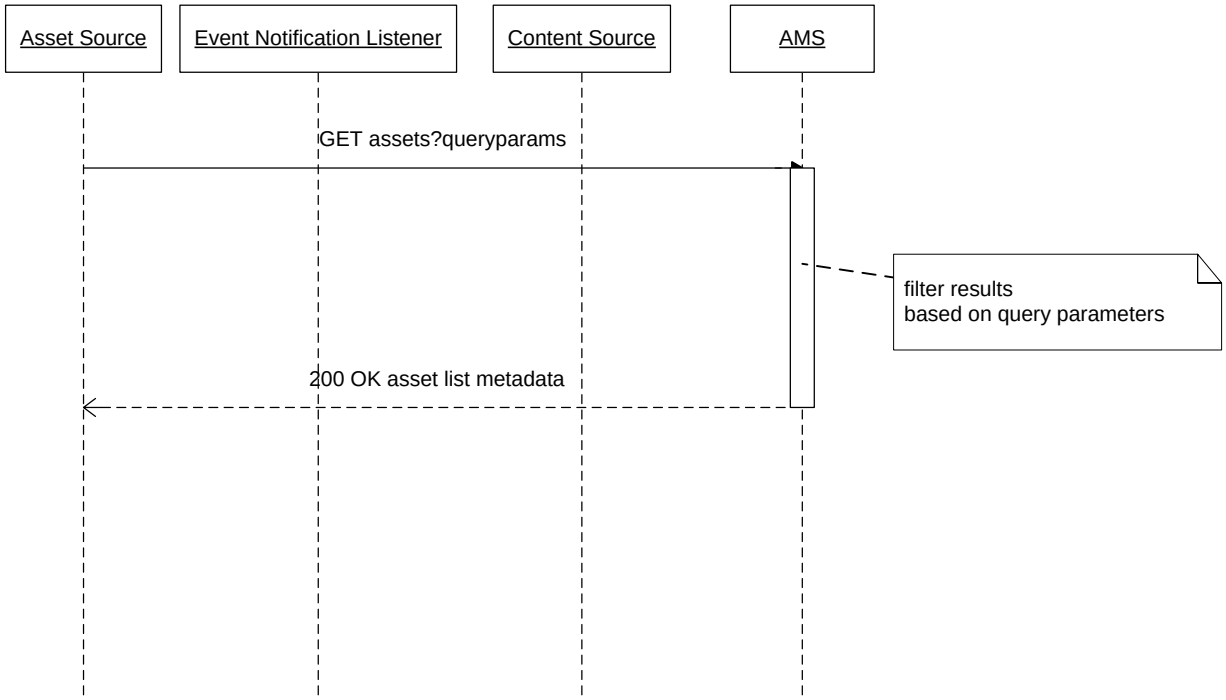

Figure 9 - Retrieve (filtered) List of Assets

The asset source wishes to obtain a list of Assets on the AMS (optionally filtered by the query parameters). If no query parameters are present, then all Assets are returned. If any query parameters are present, then the results include the filtered list of Assets.

6.6 Bulk Creation/Update of Assets

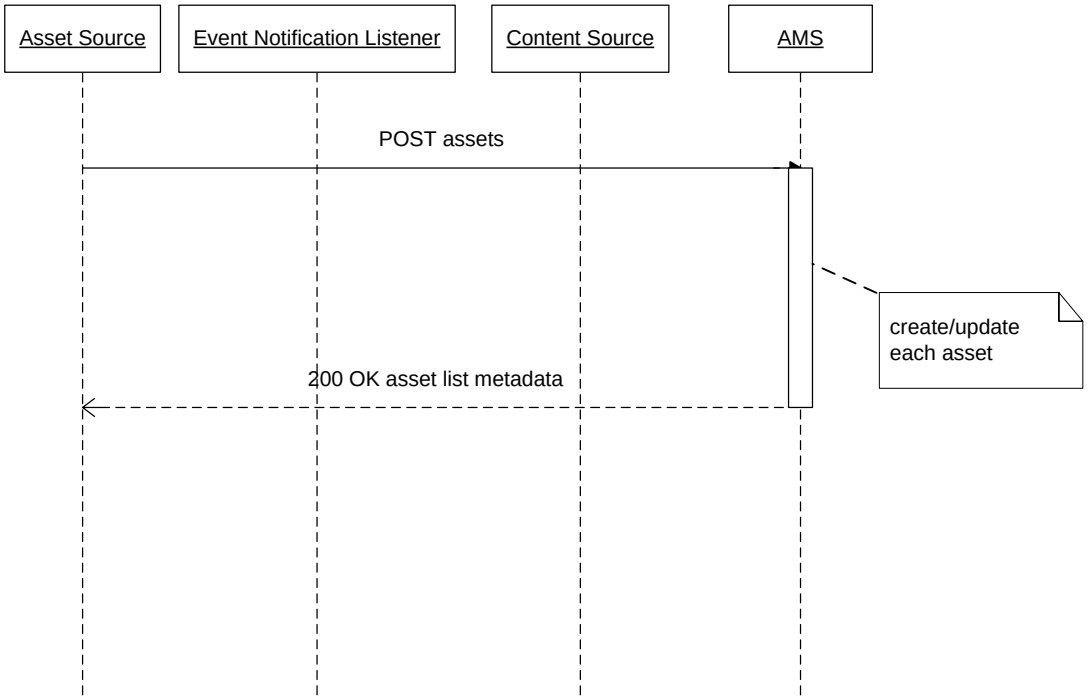

Figure 10 - Bulk Creation/Update of Assets

The asset source submits a collection of creation/update of Assets. For creation, no ETag attribute is specified; for updated Assets the ETag must be included. The behavior of each individual create/update is the same as the atomic operation, but the entire bulk operations occur as a single atomic operation (they all succeed or fail). In the case of success, the AssetList (summary level detail) is returned, and if an error occurs, a 400 Bad Request with a body of the ErrorResponse is returned.

6.7 Ping Health Monitoring

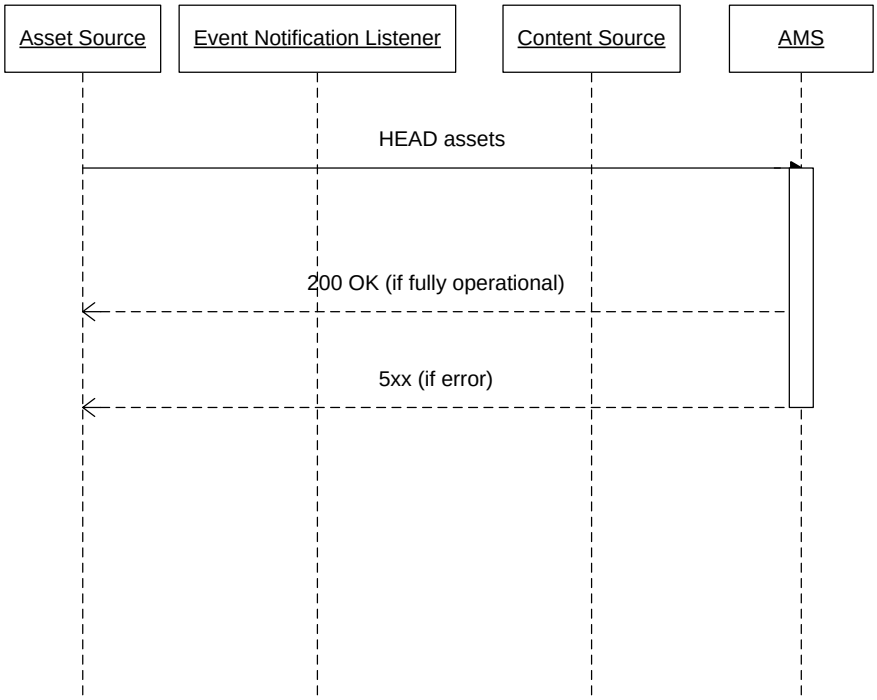

Figure 11 - Ping Health Monitoring

The light-weight Ping function is called by the Asset Source to determine the current operational health of the AMS. A 200 OK result indicates that the AMS is operational, while an error response will be accompanied by an XML body including information related to the AMS error conditions.

6.8 Create/Update a Bucket

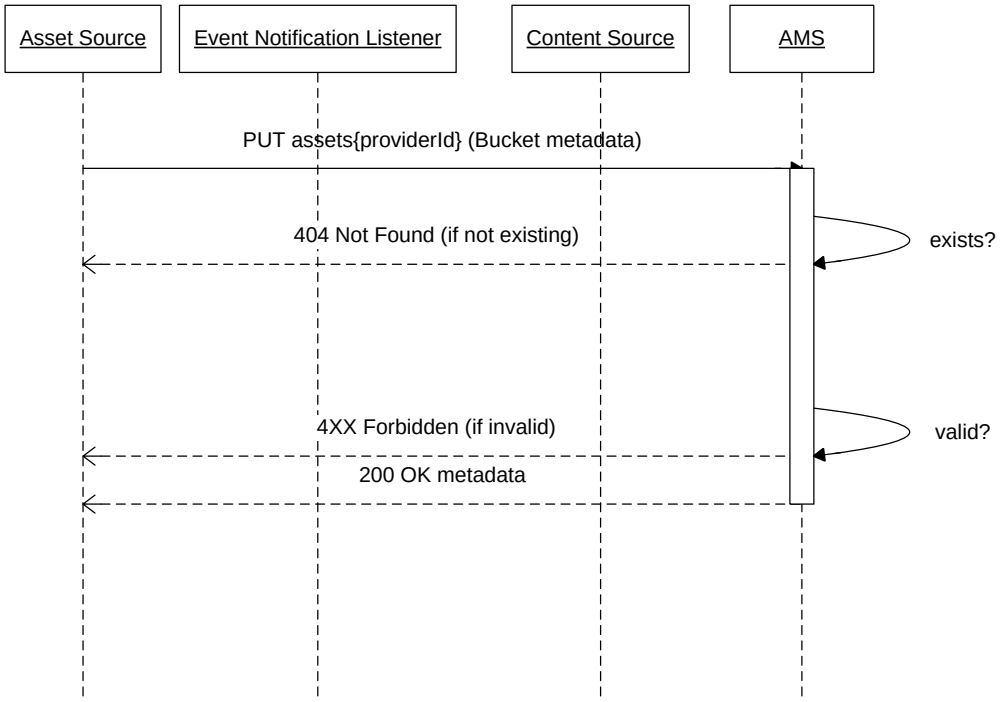

Figure 12 - Create/Update a Bucket

Buckets are created or updated by issuing a PUT message with the {providerId} of the Bucket to the AMS with XML in the request message body that describes the Bucket metadata.

If a Bucket already exists for the {providerId}, then a 409 Conflict is returned. If the providerId referenced does not yet exist in the AMS, the metadata is validated, and if invalid or the request cannot be processed for other reasons, then a 400 or 4XX status code is returned. Otherwise, the Bucket is persisted in the AMS and result is returned.

6.9 Delete a Bucket

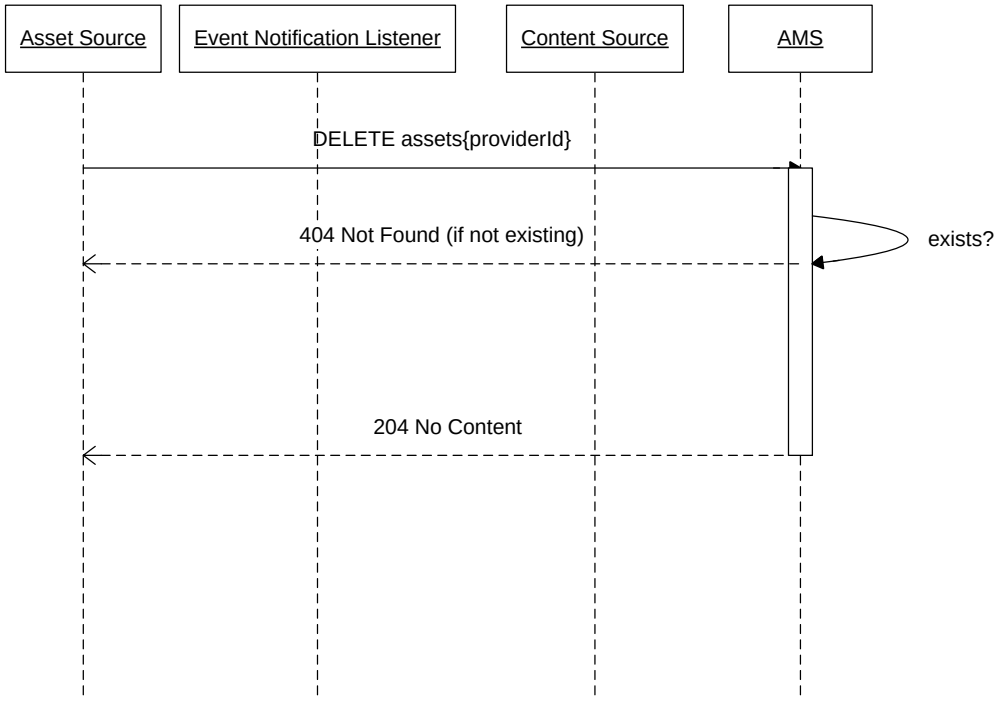

Figure 13 - Delete a Bucket

A Bucket is deleted by issuing a DELETE message with the {providerId} of the Bucket to be deleted to the AMS. If a Bucket does not already exist for the {providerId}, then 404 Not Found is returned. Otherwise, the AMS proceeds with deleting the Bucket. The server results in a 204 (No Content) to indicate successful processing of the delete message, which is similar to 200 (OK) but with no body returned in the response. The AMS rejects deletion request for Bucket which currently contain Assets (4XX).

6.10 Retrieve metadata for a Bucket

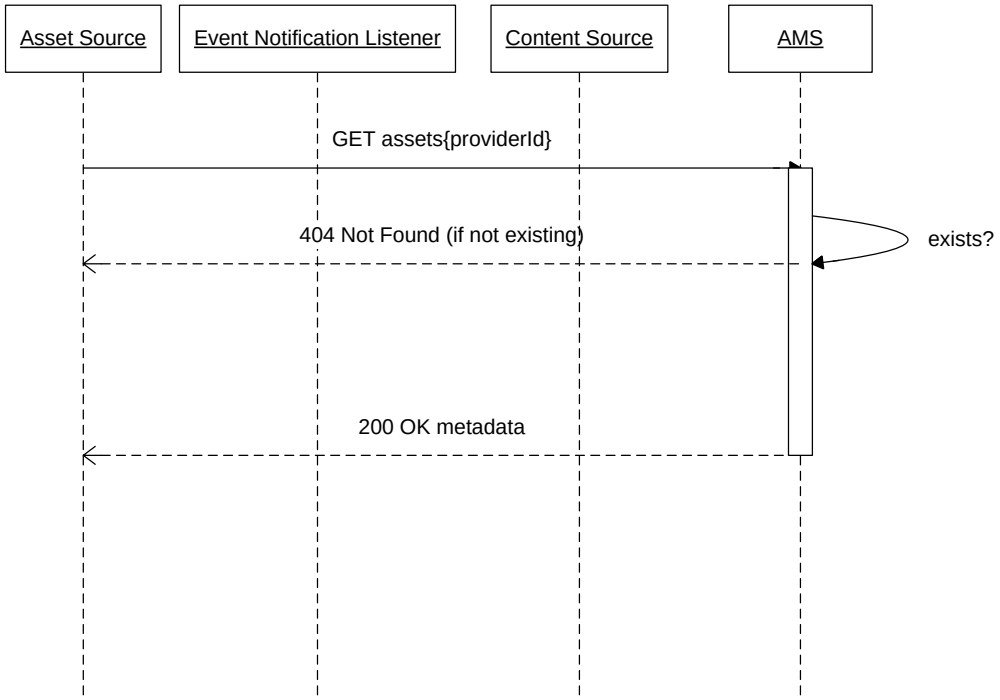

Figure 14 - Retrieve metadata for a Bucket

The Asset Source wishes to obtain details about a Bucket. If the bucket with the {providerId} does not exist on the AMS, then a 404 Not Found is returned, Otherwise, a 200 OK with the Bucket metadata in the body is returned.

6.11 Retrieve list of Buckets

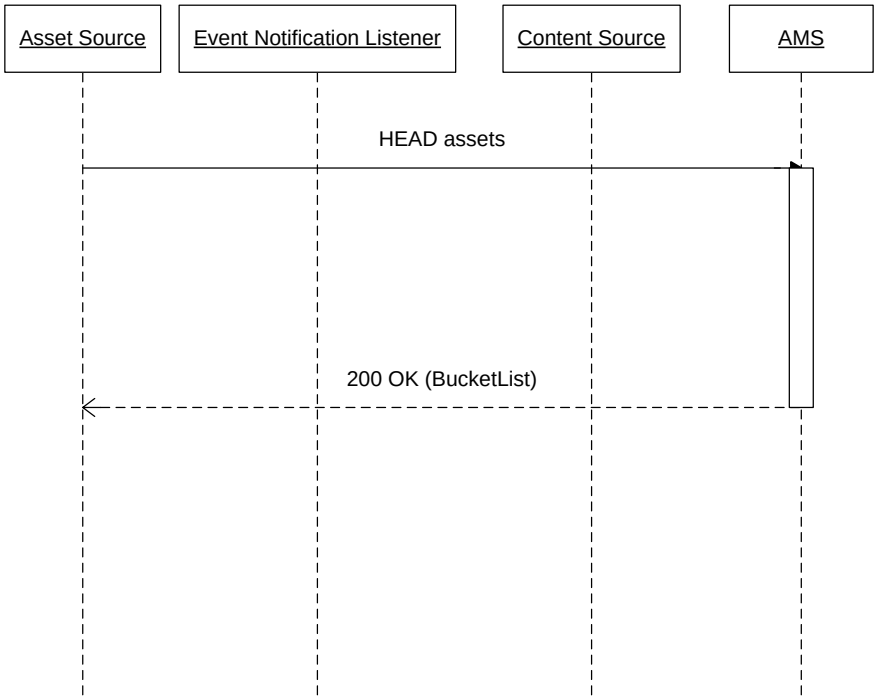

Figure 15 - Retrieve list of Buckets

The asset source wishes to obtain a list of Buckets on the AMS. The results included may be filtered by the AMS based on the identity of the authenticated caller.

Appendix I Example messages

I.1 Create an Asset (Movie ContentAsset)

Request:

```
PUT /assets/source.cp.com/Asset/MOVO0206000000037955 HTTP/1.1
Content-Type: text/xml
Content-Length: xxx

<?xml version="1.0" encoding="UTF-8"?>
<Movie xmlns="http://www.cablelabs.com/namespaces/metadata/xsd/content/1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cablelabs.com/namespaces/metadata/xsd/content/
1 MD-SP-Content-I01.xsd"
uriId="source.cp.com/Asset/MOVO0206000000037955"
notifyURI="http://source.cp.com/notify">
   <SourceUrl>http://somesource.cp.com/MOVO0206000000037955.mpg</SourceUrl>
   <ContentFileSize>25284375</ContentFileSize>
   <ContentCheckSum>A1258D3269D25852BD26548DC2654C12</ContentCheckSum>
   <PropagationPriority>10</PropagationPriority>
   <BitRate>3750</BitRate>
   <Duration>PT03H14M00S</Duration>
</Movie>
```

Success Response:

```
HTTP/1.1 201 Created
Content-Type: text/xml
Content-Length: xxx
ETag:  "42618110fcf160f4701ed3f704caed7a"

<?xml version="1.0" encoding="UTF-8"?>
<Movie xmlns="http://www.cablelabs.com/namespaces/metadata/xsd/content/1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cablelabs.com/namespaces/metadata/xsd/content/
1 MD-SP-Content-I01.xsd"
uriId="source.cp.com/Asset/MOVO0206000000037955"
notifyURI="http://source.cp.com/notify"
lastModifiedDateTime="2010-03-17T04:13:36.000Z"
eTag="42618110fcf160f4701ed3f704caed7a"
state="Provisioned">
   <SourceUrl>http://somesource.cp.com/MOVO0206000000037955.mpg</SourceUrl>
   <ContentFileSize>25284375</ContentFileSize>
   <ContentCheckSum>A1258D3269D25852BD26548DC2654C12</ContentCheckSum>
   <PropagationPriority>10</PropagationPriority>
   <ContentRef>ams.sp.com/Content/MOVO0206000000037955</ContentRef>
   <BitRate>3750</BitRate>
   <Duration>PT03H14M00S</Duration>
</Movie>
```

Potential Failure Response: (validation error)

```
HTTP/1.1 400 Bad Request
Content-Type: text/xml
Content-Length: xxx

<?xml version="1.0" encoding="UTF-8"?>
<ErrorResponse>
```

```
   <Error code="1000">Illegal SourceUrl format</Error>
</ErrorResponse>
```

I.1.1 Create an Asset (ContentGroupAsset)

Request:

```
PUT /assets/source.cp.com/ContentGroup/UNVA2001081701004001 HTTP/1.1
Content-Type: text/xml
Content-Length: xxx

<?xml version="1.0" encoding="UTF-8"?>
<ContentGroup
xmlns="http://www.cablelabs.com/namespaces/metadata/xsd/offer/1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cablelabs.com/namespaces/metadata/xsd/vod30/1
MD-SP-VODContainer-I01.xsd"
uriId="source.cp.com/ContentGroup/UNVA2001081701004001">
   <TitleRef uriId="source.cp.com/Title/UNVA2001081701004001"/>
   <MovieRef uriId="source.cp.com/Asset/UNVA2001081701004002"/>
   <PreviewRef uriId="source.cp.com/Asset/UNVA2001081701004003"/>
   <TrickRef uriId="source.cp.com/Asset/UNTR2001081701004003"/>
   <MovieRef uriId="source.cp.com/Asset/UNEN2001081701004003"/>
   <BoxCoverRef uriId="source.cp.com/Asset/UNVA2001081701004004"/>
</ContentGroup>
```

Success Response:

```
HTTP/1.1 201 Created
Content-Type: text/xml
Content-Length: xxx
ETag:  "126897696a7c876b7e"

<?xml version="1.0" encoding="UTF-8"?>
<ContentGroup
xmlns="http://www.cablelabs.com/namespaces/metadata/xsd/offer/1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cablelabs.com/namespaces/metadata/xsd/vod30/1
MD-SP-VODContainer-I01.xsd"
uriId="source.cp.com/ContentGroup/UNVA2001081701004001"
lastModifiedDateTime="2010-03-17T04:13:36.000Z"
eTag="126897696a7c876b7e"
state="Provisioned" >
   <TitleRef uriId="source.cp.com/Title/UNVA2001081701004001"/>
   <MovieRef uriId="source.cp.com/Asset/UNVA2001081701004002"/>
   <PreviewRef uriId="source.cp.com/Asset/UNVA2001081701004003"/>
   <TrickRef uriId="source.cp.com/Asset/UNTR2001081701004003"/>
   <MovieRef uriId="source.cp.com/Asset/UNEN2001081701004003"/>
   <BoxCoverRef uriId="source.cp.com/Asset/UNVA2001081701004004"/>
</ContentGroup>
```

Potential Failure Response: (validation error)

```
HTTP/1.1 400 Bad Request
Content-Type: text/xml
Content-Length: xxx

<?xml version="1.0" encoding="UTF-8"?>
<ErrorResponse>
```

```
   <Error code="1000">Malformed uriId</Error>
</ErrorResponse>
```

I.1.2 ContentAsset State Change Event Notification

Asset source is being notified of a change in the ContentAsset State of a ContentAsset on the AMS.

Request (from AMS)

```
POST http://source.cp.com/notify HTTP/1.1
Content-type: text/xml
Content-length: xx

<?xml version="1.0" encoding="UTF-8"?>
<ADI3>
<Movie uriId="source.cp.com/Asset/MOVO02060000000037955"
state="Verified">
</Movie>
</ADI3>
```

Notes:
If appropriate the AMS may group multiple event notifications into a single POST. Multiple Assets are allowed within the POST. No failure responses are expected outside of transmission and xml validation errors, and the event notification listener will accept state change notifications for assets it does not know about.

Success Response (from content source)

```
HTTP/1.1 200 OK
Content-Length: 0
```

I.2 Update an Asset

The asset source wishes to modify an existing ContentAsset (change the SourceUrl).

Request:

```
PUT /assets/source.cp.com/MOVO02060000000037955 HTTP/1.1
Content-Type: text/xml
Content-Length: xx
If-Match:  "686897696a7c876b7e"

<?xml version="1.0" encoding="UTF-8"?>
<Movie notifyURI="http://source.cp.com/notify">
      <SourceUrl>http://somesource.cp.com/somepath/somenewmovie-
MOVO02060000000037955.mpg</SourceUrl>
</Movie>
```

Success Response:

```
HTTP/1.1 20 OK
Content-Type: text/xml
Content-Length: xxx
ETag:  "686897696a7c876eee"

<?xml version="1.0" encoding="UTF-8"?>
<Movie notifyURI="http://source.cp.com/notify"
state="Provisioned">
      <SourceUrl>http://somesource.cp.com/somepath/somenewmovie-
MOVO02060000000037955.mpg</SourceUrl>
</Movie>
```

Potential Failure Response:
(if provided entity tag does not match current for uriId indicated)

```
HTTP/1.1 421 Precondition Failed
Content-Length: 0
```

Failure Response: (validation error)

```
HTTP/1.1 400 Bad Request
Content-Type: text/xml
Content-Length: xxx

<?xml version="1.0" encoding="UTF-8"?>
<ErrorResponse>
   <Error code="1000">Illegal SourceUrl format</Error>
</ErrorResponse>
```

I.3 Delete an Asset

The asset source wishes to have the AMS delete an asset. If the If-Match header is present then a conditional delete is performed only if the current ETag to be deleted agrees with the If-match ETag.

Request:

```
DELETE /assets/source.cp.com/MOVO0206000000037955 HTTP/1.1
If-Match:  "686897696a7c876b7e"
```

Success Response:

```
HTTP/1.1 204 No Content
Content-Length: 0
```

Potential Failure Response:

```
HTTP/1.1 421 Precondition Failed
Content-Length: 0
```

Potential Error Response:

```
HTTP/1.1 404 Not Found
Content-Length: 0
```

I.4 Retrieve Metadata for an Asset

I.4.1 Retrieve Metadata for an Asset (first time)

The asset source wishes to obtain details about an Asset.

Request:

```
GET /assets/source.cp.com/MOVO0206000000037955 HTTP/1.1
Content-Length: 0
```

Success Response:

```
HTTP/1.1 200 OK
Content-type: text/xml
Content-length: xx
ETag:  "686897696a7c876eee"

<?xml version="1.0" encoding="UTF-8"?>
<Movie state="Processing" notifyURI="http://source.cp.com/notify">
   <ContentRef>ams.sp.com/Content/MOV0020600000000037955</ContentRef>
```

```
   <BitRate>37500</BitRate>
   <SourceUrl>http://somesource.cp.com/somepath/
MOVO0206000000037955.mpg</SourceUrl>
   <Duration>1823</Duration>
   <ContentFileSize>25284375</ContentFileSize>
   <ContentCheckSum>A1258D3269D25852BD26548DC2654C12</ContentCheckSum>
</Movie>
```

Notes:
ContentRef is an example and does not indicate any requirement on the path of the value.
The absence of the ETag on a request indicates an open request for the resource representation.

Potential Error Response:

```
HTTP/1.1 404 Not Found
Content-Length: 0
```

I.4.2 Retrieve Metadata for an Asset (refresh)

The asset source wishes to obtain details about an asset if it has changed since the last time details were obtained.

Request:

```
GET /assets/source.cp.com/MOVO0206000000037955 HTTP/1.1
If-None-Match:  "686897696a7c876eee"
Content-Length: 0
```

Success Response (if modified from ETag value):

```
HTTP/1.1 200 OK
Content-type: text/xml
Content-length: xx
ETag:  "686897696a7c832eee"

<?xml version="1.0" encoding="UTF-8"?>
<Movie state="Processing" notifyURI="http://source.cp.com/notify">
   <BitRate>3750</BitRate>
   <SourceUrl>http://somesource.cp.com/somepath/
MOVO0206000000037955.mpg</SourceUrl>
   <Duration>1858</Duration>
   <ContentFileSize>25284375</ContentFileSize>
   <ContentCheckSum>A1258D3269D25852BD26548DC2654C12</ContentCheckSum>
</Movie>
```

Success Response (if not modified from ETag value):

```
HTTP/1.1 304 Not Modified
Content-Length: 0
```

Potential Error Response:

```
HTTP/1.1 404 Not Found
Content-Length: 0
```

I.5 Retrieve (filtered) list of Assets

The asset source wishes to obtain a list of Assets on the AMS (optionally filtered by the query parameters). If no query parameters are present then all Assets are returned. If any query parameters are present then the results include the filtered list of Assets.

This example shows the results for a filter query for only ContentAsset with a state of Verified and Failed.

Request:

```
GET /assets?providerId=source.cp.com&state=Verified&state=Failed HTTP/1.1
Content-Length: 0
```

Success Response:

```
HTTP/1.1 200 OK
Content-type: text/xml
Content-length: xx

<?xml version="1.0" encoding="UTF-8"?>
<ADI3>
   <Movie uriId="source.cp.com/MOVO0206000000037956" state="Verified" />
   <Movie uriId="source.cp.com/MOVO0206000000037957" state="Failed"/>
</ADI3>
```

I.6 Bulk creation/update of Assets

The asset source submits a collection of creation/update of Assets. For creation, no ETag attribute is specified, for updated Assets the ETag must be included. The behavior of each individual create/update is the same as the atomic operation, but the entire bulk operations occur as a single atomic operation (they all succeed or fail). In the case of success the AssetList (summary level detail) is returned, and if an error occurs a 400 Bad Request with a body of the ErrorResponse is returned.

This example shows a bulk request to create one Movie Asset and update another Movie Asset.

Request:

```
POST /assets HTTP/1.1
Content-Length: xx

<?xml version="1.0" encoding="UTF-8"?>
<ADI3 xmlns="http://www.cablelabs.com/namespaces/metadata/xsd/vod30/1"
xmlns:content="http://www.cablelabs.com/namespaces/metadata/xsd/content/1"
xmlns:core="http://www.cablelabs.com/namespaces/metadata/xsd/core/1"
xmlns:offer="http://www.cablelabs.com/namespaces/metadata/xsd/offer/1"
xmlns:terms="http://www.cablelabs.com/namespaces/metadata/xsd/terms/1"
xmlns:title="http://www.cablelabs.com/namespaces/metadata/xsd/title/1"
xmlns:vod30="http://www.cablelabs.com/namespaces/metadata/xsd/vod30/1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cablelabs.com/namespaces/metadata/xsd/vod30/1
MD-SP-VODContainer-I01.xsd">
   <Offer uriId="source.cp.com/Offer/UNVA2001081701004000"
providerVersionNum="1" internalVersionNum="0" creationDateTime="2002-01-
11T00:00:00Z" startDateTime="2002-02-01T00:00:00Z" endDateTime="2002-03-
31T11:59:59Z">
      <core:AlternateId
identifierSystem="VOD1.1">vod://source.cp.com/UNVA2001081701004000</core:Alte
rnateId>
      <core:AssetName deprecated="true">The_Titanic</core:AssetName>
      <core:Product deprecated="true">MOD</core:Product>
      <core:Provider deprecated="true">InDemand</core:Provider>
      <core:Description deprecated="true">The Titanic asset
package</core:Description>
      <core:Ext/>
      <offer:Presentation>
         <offer:CategoryRef uriId="source.cp.com/Category/InDemand/Movies A-
Z"/>
         <offer:DisplayAsNew>P7D</offer:DisplayAsNew>
         <offer:DisplayAsLastChance>P7D</offer:DisplayAsLastChance>
```

```
      </offer:Presentation>
      <offer:PromotionalContentGroupRef
uriId="source.cp.com/ContentGroup/UNVA2001081701004001"/>
      <offer:ProviderContentTier>InDemand1</offer:ProviderContentTier>
      <offer:ProviderContentTier>InDemand2</offer:ProviderContentTier>
      <offer:BillingId>56789</offer:BillingId>
      <offer:TermsRef uriId="source.cp.com/Terms/UNVA2001081701004001"/>
      <offer:ContentGroupRef
uriId="source.cp.com/ContentGroup/UNVA2001081701004001"/>
   </Offer>
   <Title uriId="source.cp.com/Title/UNVA2001081701004001"
providerVersionNum="1" internalVersionNum="0" creationDateTime="2002-01-
11T00:00:00Z" startDateTime="2002-02-01T00:00:00Z" endDateTime="2002-03-
31T11:59:59Z">
      <core:AlternateId
identifierSystem="VOD1.1">vod://source.cp.com/UNVA2001081701004001</core:Alte
rnateId>
      <core:AlternateId identifierSystem="ISAN">1881-66C7-3420-000-7-9F3A-
02450- U</core:AlternateId>
      <core:ProviderQAContact>John Doe,
JDoe@InDemand.com</core:ProviderQAContact>
      <core:AssetName deprecated="true">The_Titanic_Title</core:AssetName>
      <core:Product deprecated="true">MOD</core:Product>
      <core:Provider deprecated="true">InDemand</core:Provider>
      <core:Description deprecated="true">The Titanic title
metadata</core:Description>
      <core:Ext/>
      <title:LocalizableTitle xml:lang="en">
         <title:TitleSortName>Titanic, The</title:TitleSortName>
         <title:TitleBrief>The Titanic</title:TitleBrief>
         <title:TitleMedium>The Titanic</title:TitleMedium>
         <title:TitleLong>The Titanic</title:TitleLong>
         <title:SummaryShort>Fictional romantic tale of a rich girl and poor
boy who meet on the ill-fated voyage of the 'unsinkable'
ship</title:SummaryShort>
         <title:SummaryMedium>Fictional romantic tale of rich girl and poor
boy who meet on the ill-fated voyage of the 'unsinkable'
ship</title:SummaryMedium>
         <title:SummaryLong>Fictional romantic tale of a rich girl and poor
boy who meet on the ill-fated voyage of the 'unsinkable'
ship</title:SummaryLong>
         <title:ActorDisplay>Kate Winslet,Leonardo DiCaprio, Billy
Zane</title:ActorDisplay>
         <title:Actor xmlns="" fullName="Kate Winslet" firstName="Kate"
lastName="Winslet" sortableName="Winslet,Kate"/>
         <title:Actor xmlns="" fullName="Leonardo DiCaprio"
firstName="Leonardo" lastName="DiCaprio" sortableName="DiCaprio,Leonardo"/>
         <title:Actor xmlns="" fullName="Billy Zane" firstName="Billy"
lastName="Zane" sortableName="Zane,Billy"/>
         <title:WriterDisplay>James Cameron</title:WriterDisplay>
         <title:Director xmlns="" fullName="James Cameron" firstName="James"
lastName="Cameron" sortableName="Cameron,James"/>
         <title:Producer xmlns="" fullName="James Cameron" firstName="James"
lastName="Cameron" sortableName="Cameron,James"/>
         <title:StudioDisplay>Paramount</title:StudioDisplay>
         <title:RecordingArtist>Celine Dion</title:RecordingArtist>
         <title:RecordingArtist>Bob Dylan</title:RecordingArtist>
```

```
         <title:SongTitle>My Heart Will Go On</title:SongTitle>
         <title:SongTitle>Blowin' in the Wind</title:SongTitle>
         <title:EpisodeName deprecated="true">Collision With
destiny</title:EpisodeName>
         <title:EpisodeID deprecated="true">The only one</title:EpisodeID>
         <title:Chapter heading="Opening Scene" timeCode="00:00:02:00"/>
         <title:Chapter heading="They Meet" timeCode="00:15:12:00"/>
         <title:Chapter heading="The Collision" timeCode="02:03:06:00"/>
         <title:Chapter heading="She Goes Under" timeCode="02:55:15:00"/>
         <title:Chapter heading="Final Scene" timeCode="03:02:09:00"/>
      </title:LocalizableTitle>
      <title:Rating ratingSystem="MPAA">R</title:Rating>
      <title:Rating ratingSystem="MSO">Age-14</title:Rating>
      <title:Audience>Adult</title:Audience>
      <title:Advisory>AL</title:Advisory>
      <title:Advisory>N</title:Advisory>
      <title:IsClosedCaptioning>true</title:IsClosedCaptioning>
      <title:DisplayRunTime>03:14</title:DisplayRunTime>
      <title:Year>1997</title:Year>
      <title:CountryOfOrigin>US</title:CountryOfOrigin>
      <title:Genre>Drama</title:Genre>
      <title:Genre>Romance</title:Genre>
      <title:ShowType>Movie</title:ShowType>
      <title:IsSeasonPremier deprecated="true">true</title:IsSeasonPremier>
      <title:IsSeasonFinale deprecated="true">true</title:IsSeasonFinale>
      <title:IsEncryptionRequired>true</title:IsEncryptionRequired>
      <title:BoxOffice>1835000000</title:BoxOffice>
      <title:ProgrammerCallLetters>IND</title:ProgrammerCallLetters>
   </Title>
   <ContentGroup uriId="source.cp.com/ContentGroup/UNVA2001081701004001"
providerVersionNum="1" internalVersionNum="0" creationDateTime="2002-01-
11T00:00:00Z" startDateTime="2002-02-01T00:00:00Z" endDateTime="2002-03-
31T11:59:59Z">
      <core:AlternateId
identifierSystem="VOD1.1">vod://source.cp.com/UNVA2001081701004001</core:Alte
rnateId>
      <offer:TitleRef uriId="source.cp.com/Title/UNVA2001081701004001"/>
      <offer:MovieRef uriId="source.cp.com/Asset/UNVA2001081701004002"/>
      <offer:PreviewRef uriId="source.cp.com/Asset/UNVA2001081701004003"/>
      <offer:TrickRef uriId="source.cp.com/Asset/UNTR2001081701004003"/>
      <offer:MovieRef uriId="source.cp.com/Asset/UNEN2001081701004003"/>
      <offer:BoxCoverRef uriId="source.cp.com/Asset/UNVA2001081701004004"/>
   </ContentGroup>
   <Terms uriId="source.cp.com/Terms/UNVA2001081701004001"
providerVersionNum="1" internalVersionNum="0" creationDateTime="2002-01-
11T00:00:00Z" startDateTime="2002-02-01T00:00:00Z" endDateTime="2002-03-
31T11:59:59Z">
      <core:AlternateId
identifierSystem="VOD1.1">vod://source.cp.com/UNVA2001081701004001</core:Alte
rnateId>
      <terms:ContractName>InDemand</terms:ContractName>
      <terms:BillingGracePeriod>PT300S</terms:BillingGracePeriod>
      <terms:RentalPeriod>P00DT24H00M</terms:RentalPeriod>
      <terms:HomeVideoWindow>P1000D</terms:HomeVideoWindow>
      <terms:SubscriberViewLimit startDateTime="2002-02-01T00:00:00Z"
endDateTime="2002-02-28T23:59:59Z" maximumViews="5"/>
```

```
      <terms:SubscriberViewLimit startDateTime="2002-03-01T00:00:00Z"
endDateTime="2002-03-31T23:59:59Z" maximumViews="5"/>
      <terms:SuggestedPrice>5.95</terms:SuggestedPrice>
      <terms:DistributorRoyaltyInfo>
         <terms:OrganizationName>InDemand</terms:OrganizationName>
         <terms:RoyaltyPercent>52.5</terms:RoyaltyPercent>
         <terms:RoyaltyMinimum>3.124</terms:RoyaltyMinimum>
         <terms:RoyaltyFlatRate>3.155</terms:RoyaltyFlatRate>
      </terms:DistributorRoyaltyInfo>
      <terms:StudioRoyaltyInfo>
         <terms:OrganizationName>Paramount</terms:OrganizationName>
         <terms:OrganizationCode>PAR</terms:OrganizationCode>
         <terms:RoyaltyPercent>47.5</terms:RoyaltyPercent>
         <terms:RoyaltyMinimum>2.826</terms:RoyaltyMinimum>
         <terms:RoyaltyFlatRate>2.795</terms:RoyaltyFlatRate>
      </terms:StudioRoyaltyInfo>
   </Terms>
   <Category uriId="source.cp.com/Category/InDemand/Movies A-Z"
providerVersionNum="1" internalVersionNum="0" creationDateTime="2002-01-
11T00:00:00Z" startDateTime="2002-02-01T00:00:00Z" endDateTime="2002-03-
31T11:59:59Z">
      <core:AlternateId
identifierSystem="VOD1.1">vod://source.cp.com/UNVA2001081701004001</core:Alte
rnateId>
      <offer:CategoryPath>InDemand/Movies A-Z</offer:CategoryPath>
   </Category>
   <Movie uriId="source.cp.com/Asset/UNVA2001081701004002"
providerVersionNum="1" internalVersionNum="0" creationDateTime="2002-01-
11T00:00:00Z" startDateTime="2002-02-01T00:00:00Z" endDateTime="2002-03-
31T11:59:59Z">
      <core:AlternateId
identifierSystem="VOD1.1">vod://source.cp.com/UNVA2001081701004002</core:Alte
rnateId>
      <core:AssetName deprecated="true">The_Titanic.mpg</core:AssetName>
      <core:Product deprecated="true">MOD</core:Product>
      <core:Provider deprecated="true">InDemand</core:Provider>
      <core:Description deprecated="true">The Titanic
Movie</core:Description>
      <core:Ext/>
      <content:SourceUrl>The_Titanic.mpg</content:SourceUrl>
      <content:ContentFileSize>3907840625</content:ContentFileSize>

   <content:ContentCheckSum>12558D3269D25852BD26548DC2654CA2</content:Content
CheckSum>
      <content:PropagationPriority>1</content:PropagationPriority>
      <content:AudioType>Dolby Digital</content:AudioType>
      <content:AudioType>Mono</content:AudioType>
      <content:ScreenFormat>Widescreen</content:ScreenFormat>
      <content:Resolution>480i</content:Resolution>
      <content:FrameRate>30</content:FrameRate>
      <content:Codec>AVC MP@L42</content:Codec>
      <content:BitRate>3750</content:BitRate>
      <content:Duration>PT03H14M00S</content:Duration>
      <content:Language>en</content:Language>
      <content:Language>sv</content:Language>
      <content:SubtitleLanguage>es</content:SubtitleLanguage>
      <content:SubtitleLanguage>sv</content:SubtitleLanguage>
```

```
      <content:DubbedLanguage>es</content:DubbedLanguage>
      <content:DubbedLanguage>sv</content:DubbedLanguage>
      <content:Rating ratingSystem="MPAA">R</content:Rating>
      <content:Rating ratingSystem="MSO">Age-14</content:Rating>
      <content:Audience>Adult</content:Audience>
      <content:CopyControlInfo>
         <content:IsCopyProtection>true</content:IsCopyProtection>

   <content:IsCopyProtectionVerbose>true</content:IsCopyProtectionVerbose>
         <content:AnalogProtectionSystem>1</content:AnalogProtectionSystem>
         <content:EncryptionModeIndicator>1</content:EncryptionModeIndicator>
         <content:ConstrainedImageTrigger>1</content:ConstrainedImageTrigger>
         <content:CGMS_A>1</content:CGMS_A>
      </content:CopyControlInfo>
      <content:IsResumeEnabled>true</content:IsResumeEnabled>
      <content:TrickModesRestricted>RW</content:TrickModesRestricted>
      <content:TrickModesRestricted>Pause</content:TrickModesRestricted>
      <content:TrickModesRestricted>FF</content:TrickModesRestricted>
   </Movie>
   <Preview uriId="source.cp.com/Asset/UNVA2001081701004003"
providerVersionNum="1" internalVersionNum="0" creationDateTime="2002-01-
11T00:00:00Z" startDateTime="2002-02-01T00:00:00Z" endDateTime="2002-03-
31T11:59:59Z">
      <core:AlternateId
identifierSystem="VOD1.1">vod://source.cp.com/UNVA2001081701004003</core:Alte
rnateId>
      <core:AssetName
deprecated="true">The_Titanic_Preview.mpg</core:AssetName>
      <core:Product deprecated="true">MOD</core:Product>
      <core:Provider deprecated="true">InDemand</core:Provider>
      <core:Description deprecated="true">The Titanic
Preview</core:Description>
      <core:Ext/>
      <content:SourceUrl>The_Titanic_Preview.mpg</content:SourceUrl>
      <content:ContentFileSize>25284375</content:ContentFileSize>

   <content:ContentCheckSum>A1258D3269D25852BD26548DC2654C12</content:Content
CheckSum>
      <content:PropagationPriority>1</content:PropagationPriority>
      <content:AudioType>Dolby Digital</content:AudioType>
      <content:ScreenFormat>Widescreen</content:ScreenFormat>
      <content:Resolution>480i</content:Resolution>
      <content:FrameRate>30</content:FrameRate>
      <content:Codec>AVC MP@L42</content:Codec>
      <content:BitRate>3750</content:BitRate>
      <content:Duration>PT00H00M45S</content:Duration>
      <content:Language>en</content:Language>
      <content:SubtitleLanguage>es</content:SubtitleLanguage>
      <content:DubbedLanguage>es</content:DubbedLanguage>
      <content:Rating ratingSystem="MPAA">G</content:Rating>
      <content:Rating ratingSystem="MSO">All-Ages</content:Rating>
      <content:Audience>Adult</content:Audience>
      <content:TrickModesRestricted>RW</content:TrickModesRestricted>
      <content:TrickModesRestricted>Pause</content:TrickModesRestricted>
      <content:TrickModesRestricted>FF</content:TrickModesRestricted>
   </Preview>
```

```
   <Trick uriId="source.cp.com/Asset/UNTR2001081701004003"
providerVersionNum="1" internalVersionNum="0" creationDateTime="2002-01-
11T00:00:00Z" startDateTime="2002-02-01T00:00:00Z" endDateTime="2002-03-
31T11:59:59Z">
      <core:AlternateId
identifierSystem="VOD1.1">vod://source.cp.com/UNTR2001081701004003</core:Alte
rnateId>
      <core:AssetName
deprecated="true">The_Titanic_Trick.mpg</core:AssetName>
      <core:Product deprecated="true">MOD</core:Product>
      <core:Provider deprecated="true">InDemand</core:Provider>
      <core:Description deprecated="true">The Titanic
Trick</core:Description>
      <core:Ext/>
      <content:MasterSourceRef uriId="$TitleUriId"/>
      <content:SourceUrl>The_Titanic_Mediahawk.mpg</content:SourceUrl>
      <content:ContentFileSize>25284375</content:ContentFileSize>

   <content:ContentCheckSum>A1258D3269D25852BD26548DC2654C12</content:Content
CheckSum>
      <content:PropagationPriority>1</content:PropagationPriority>
      <content:BitRate>3750</content:BitRate>
      <content:VendorName>Concurrent</content:VendorName>
      <content:VendorProduct>Mediahawk</content:VendorProduct>
      <content:ForVersion>8.30</content:ForVersion>
      <content:TrickMode>FFWD</content:TrickMode>
   </Trick>
   <Movie uriId="source.cp.com/Asset/UNEN2001081701004003"
providerVersionNum="1" internalVersionNum="0" creationDateTime="2002-01-
11T00:00:00Z" startDateTime="2002-02-01T00:00:00Z" endDateTime="2002-03-
31T11:59:59Z">
      <core:AlternateId
identifierSystem="VOD1.1">vod://source.cp.com/UNEN2001081701004003</core:Alte
rnateId>
      <core:AssetName
deprecated="true">The_Titanic_Encrypted.mpg</core:AssetName>
      <core:Product deprecated="true">MOD</core:Product>
      <core:Provider deprecated="true">InDemand</core:Provider>
      <core:Description deprecated="true">The Titanic
Encrypted</core:Description>
      <core:Ext/>
      <content:MasterSourceRef
uriId="source.cp.com/Asset/UNVA2001081701004002"/>
      <content:SourceUrl>The_Titanic_Verimatrix.mpg</content:SourceUrl>
      <content:ContentFileSize>25284375</content:ContentFileSize>

   <content:ContentCheckSum>A1258D3269D25852BD26548DC2654C12</content:Content
CheckSum>
      <content:PropagationPriority>1</content:PropagationPriority>
      <content:AudioType>Dolby Digital</content:AudioType>
      <content:AudioType>Mono</content:AudioType>
      <content:ScreenFormat>Widescreen</content:ScreenFormat>
      <content:Resolution>480i</content:Resolution>
      <content:FrameRate>30</content:FrameRate>
      <content:Codec>AVC MP@L42</content:Codec>
      <content:BitRate>3750</content:BitRate>
      <content:Duration>PT03H14M00S</content:Duration>
```

```
      <content:Language>en</content:Language>
      <content:Language>sv</content:Language>
      <content:SubtitleLanguage>es</content:SubtitleLanguage>
      <content:SubtitleLanguage>sv</content:SubtitleLanguage>
      <content:DubbedLanguage>es</content:DubbedLanguage>
      <content:DubbedLanguage>sv</content:DubbedLanguage>
      <content:Rating ratingSystem="MPAA">R</content:Rating>
      <content:Rating ratingSystem="MSO">Age-14</content:Rating>
      <content:Audience>Adult</content:Audience>
      <content:EncryptionInfo>
         <content:VendorName>Verimatrix</content:VendorName>
         <content:ReceiverType>system1</content:ReceiverType>
         <content:ReceiverVersion>2</content:ReceiverVersion>
         <content:Encryption>symmetric</content:Encryption>
         <content:EncryptionAlgorithm>DES</content:EncryptionAlgorithm>
         <content:EncryptionDateTime>2002-01-
11T12:23:23Z</content:EncryptionDateTime>

   <content:EncryptionSystemInfo>system123</content:EncryptionSystemInfo>

   <content:EncryptionKeyBlock>A1258D3269D25852BD26548DC2654C12</content:Encr
yptionKeyBlock>
      </content:EncryptionInfo>
      <content:IsResumeEnabled>true</content:IsResumeEnabled>
      <content:TrickModesRestricted>RW</content:TrickModesRestricted>
      <content:TrickModesRestricted>Pause</content:TrickModesRestricted>
      <content:TrickModesRestricted>FF</content:TrickModesRestricted>
   </Movie>
   <BoxCover uriId="source.cp.com/Asset/UNVA2001081701004004"
providerVersionNum="1" internalVersionNum="0" creationDateTime="2002-01-
11T00:00:00Z" startDateTime="2002-02-01T00:00:00Z" endDateTime="2002-03-
31T11:59:59Z">
      <core:AlternateId
identifierSystem="VOD1.1">vod://source.cp.com/UNVA2001081701004004</core:Alte
rnateId>
      <core:AssetName
deprecated="true">The_Titanic_Box_Cover.bmp</core:AssetName>
      <core:Product deprecated="true">MOD</core:Product>
      <core:Provider deprecated="true">InDemand</core:Provider>
      <core:Description deprecated="true">The Titanic Box
Cover</core:Description>
      <core:Ext/>
      <content:SourceUrl>The_Titanic_Box_Cover.bmp</content:SourceUrl>
      <content:ContentFileSize>15235</content:ContentFileSize>

   <content:ContentCheckSum>B3258D3269D25852BD26548DC2654CD2</content:Content
CheckSum>
      <content:PropagationPriority>1</content:PropagationPriority>
      <content:X_Resolution>320</content:X_Resolution>
      <content:Y_Resolution>240</content:Y_Resolution>
      <content:Language>en</content:Language>
   </BoxCover>
</ADI3>
```

Success Response:

```
HTTP/1.1 200 OK
Content-Length: xx
```

```
<?xml version="1.0" encoding="UTF-8"?>
<ADI3>
   <Offer uriId="source.cp.com/Offer/UNVA2001081701004000"
eTag="aa6897696a7c876b13"/>
   <Title uriId="source.cp.com/Title/UNVA2001081701004001"
eTag="996897696a7c876b13"/>
   <ContentGroup uriId="source.cp.com/ContentGroup/UNVA2001081701004001"
eTag="886897696a7c876b13"/>
   <Terms uriId="source.cp.com/Terms/UNVA2001081701004001"
eTag="776897696a7c876b13"/>
   <Category uriId="source.cp.com/Category/InDemand/Movies A-Z"
eTag="666897696a7c876b13"/>
   <Movie uriId="source.cp.com/Asset/UNVA2001081701004002"
eTag="556897696a7c876b13" state="Provisioned"/>
   <Preview uriId="source.cp.com/Asset/UNVA2001081701004003"
eTag="446897696a7c876b13" state="Provisioned"/>
   <Trick uriId="source.cp.com/Asset/UNTR2001081701004003"
eTag="336897696a7c876b13" state="Provisioned"/>
   <Movie uriId=" source.cp.com/Asset/UNEN2001081701004003"
eTag="226897696a7c876b13" state="Provisioned"/>
   <BoxCover uriId="source.cp.com/Asset/UNVA2001081701004004"
eTag="116897696a7c876b13" state="Provisioned"/>
</ADI3>
```

Potential Error Response:

```
HTTP/1.1 400 Bad Request
Content-Length: 0
<?xml version="1.0" encoding="UTF-8"?>
<ErrorResponse>
   <Error code="1000" xml:lang="en">Start time is in the past</Error>
   <Error code="1001">Missing required element SourceUrl</Error>
</ErrorResponse>
```

I.7 Ping Health Monitoring

The asset source wishes to determine the current operational health of the AMS.

Request:

```
HEAD /assets HTTP/1.1
```

Success Response:

```
HTTP/1.1 200 OK
```

Potential Error Response:

```
HTTP/1.1 503 Service Unavailable
Content-Length: 0
<?xml version="1.0" encoding="UTF-8"?>
<ErrorResponse>
   <Error code="5000" xml:lang="en">System is being maintained and will be
restored to operation by 2012-12-05 11:13. Please contact AMS administration
for more details</Error>
</ErrorResponse>
```

I.8 Create a Bucket

Request:

```
PUT /assets/source.cp.com HTTP/1.1
Content-Type: text/xml
Content-Length: xxx

<?xml version="1.0" encoding="UTF-8"?>
<Bucket xmlns="http://www.cablelabs.com/namespaces/metadata/xsd/core/1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cablelabs.com/namespaces/metadata/xsd/core/1
MD-SP-CORE-I01.xsd" providerId="source.cp.com">
  <TotalStorage>1000000000000</TotalStorage>
  <InputProtocols>http,https</InputProtocols>
  <TotalInputBandwidth>1000000000000</TotalInputBandwidth>
  <OutputProtocols>http,https</OutputProtocols>
  <TotalOutputBandwidth>1000000000000</TotalOutputBandwidth>
</Bucket>
```

Success Response:

```
HTTP/1.1 201 Created
Content-Type: text/xml
Content-Length: xxx

<?xml version="1.0" encoding="UTF-8"?>
<Bucket xmlns="http://www.cablelabs.com/namespaces/metadata/xsd/core/1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cablelabs.com/namespaces/metadata/xsd/core/1
MD-SP-CORE-I01.xsd" providerId="source.cp.com">
  <TotalStorage>1000000000000</TotalStorage>
  <AvailableStorage>1000000000000</AvailableStorage>
  <InputProtocols>http,https</InputProtocols>
  <TotalInputBandwidth>1000000000000</TotalInputBandwidth>
  <AvailableInputBandwidth>1000000000000</AvailableInputBandwidth>
  <OutputProtocols>http,https</OutputProtocols>
  <TotalOutputBandwidth>1000000000000</TotalOutputBandwidth>
  <AvailableOutputBandwidth>1000000000000</AvailableOutputBandwidth>
  <NumAssets>0</NumAssets>
</Bucket>
```

Potential Failure Response: (validation error)

```
HTTP/1.1 400 Bad Request
Content-Type: text/xml
Content-Length: xxx

<?xml version="1.0" encoding="UTF-8"?>
<ErrorResponse>
  <Error code="2000">Illegal Bucket request</Error>
</ErrorResponse>
```

I.8.1 Update a Bucket

Request:

```
PUT /assets/source.cp.com HTTP/1.1
Content-Type: text/xml
Content-Length: xxx
```

```
<?xml version="1.0" encoding="UTF-8"?>
<Bucket xmlns="http://www.cablelabs.com/namespaces/metadata/xsd/core/1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cablelabs.com/namespaces/metadata/xsd/core/1
MD-SP-CORE-I01.xsd" providerId="source.cp.com">
  <TotalStorage>9999999999999</TotalStorage>
  <InputProtocols>http,https</InputProtocols>
  <TotalInputBandwidth>1000000000000</TotalInputBandwidth>
  <OutputProtocols>http,https</OutputProtocols>
  <TotalOutputBandwidth>1000000000000</TotalOutputBandwidth>
</Bucket>
```

Success Response:

```
HTTP/1.1 200 Ok
Content-Type: text/xml
Content-Length: xxx

<?xml version="1.0" encoding="UTF-8"?>
<Bucket xmlns="http://www.cablelabs.com/namespaces/metadata/xsd/core/1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cablelabs.com/namespaces/metadata/xsd/core/1
MD-SP-CORE-I01.xsd" providerId="source.cp.com">
  <TotalStorage>9999999999999</TotalStorage>
  <AvailableStorage>9000000000000</AvailableStorage>
  <InputProtocols>http,https</InputProtocols>
  <TotalInputBandwidth>1000000000000</TotalInputBandwidth>
  <AvailableInputBandwidth>1000000000000</AvailableInputBandwidth>
  <OutputProtocols>http,https</OutputProtocols>
  <TotalOutputBandwidth>1000000000000</TotalOutputBandwidth>
  <AvailableOutputBandwidth>1000000000000</AvailableOutputBandwidth>
  <NumAssets>1234</NumAssets>
</Bucket>
```

Potential Failure Response: (validation error)

```
HTTP/1.1 400 Bad Request
Content-Type: text/xml
Content-Length: xxx

<?xml version="1.0" encoding="UTF-8"?>
<ErrorResponse>
  <Error code="2001">Storage quota exceeded</Error>
</ErrorResponse>
```

I.9 Delete a Bucket

The asset source wishes to have the AMS delete an empty bucket.

Request:

```
DELETE /assets/source.cp.com HTTP/1.1
```

Success Response:

```
HTTP/1.1 204 No Content
Content-Length: 0
```

Potential Error Response:

```
HTTP/1.1 404 Not Found
```

```
Content-Length: 0
```

I.10 Retrieve metadata for a Bucket

The asset source wishes to obtain details about a bucket.

Request:

```
GET /assets/source.cp.com HTTP/1.1
Content-Length: 0
```

Success Response:

```
HTTP/1.1 200 OK
Content-type: text/xml
Content-length: xx

<?xml version="1.0" encoding="UTF-8"?>
<Bucket xmlns="http://www.cablelabs.com/namespaces/metadata/xsd/core/1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cablelabs.com/namespaces/metadata/xsd/core/1
MD-SP-CORE-I01.xsd" providerId="source.cp.com">
  <TotalStorage>9999999999999</TotalStorage>
  <AvailableStorage>34560000</AvailableStorage>
  <InputProtocols>http,https</InputProtocols>
  <TotalInputBandwidth>1000000000000</TotalInputBandwidth>
  <AvailableInputBandwidth>34560000323</AvailableInputBandwidth>
  <OutputProtocols>http,https</OutputProtocols>
  <TotalOutputBandwidth>1000000000000</TotalOutputBandwidth>
  <AvailableOutputBandwidth>45600000323</AvailableOutputBandwidth>
  <NumAssets>2345</NumAssets>
</Bucket>
```

Potential Error Response:

```
HTTP/1.1 404 Not Found
Content-Length: 0
```

I.11 Retrieve list of Buckets

The asset source wishes to obtain a list of Buckets on the AMS (which the calling authenticated user has access to).

Request:

```
GET /assets HTTP/1.1
Content-Length: 0
```

Success Response:

```
HTTP/1.1 200 OK
Content-type: text/xml
Content-length: xx

<?xml version="1.0" encoding="UTF-8"?>
<BucketList xmlns="http://www.cablelabs.com/namespaces/metadata/xsd/core/1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.cablelabs.com/namespaces/metadata/xsd/core/1
MD-SP-CORE-I01.xsd" >
  <Bucket providerId="source.cp.com" />
  <Bucket providerId="east.source.cp.com" />
  <Bucket providerId="west.source.cp.com" />
</BucketList>
```

Appendix II Revision History

The following ECN was incorporated into version I02 of this specification:

ECN	Date Accepted	Title	Author
AMIv3.0-N-11.0081-2	11/8/11	AMI buckets	Bill Hood