Content Recognition Metadata 'crmd' namespace

Notices

CONTENTS

1 INTRODUCTION

Studios provide metadata to content recognition (e.g., fingerprint) vendors who then use this information for displays, reports and other processing. Organizations, such as the MPAA, may be an intermediary for passing content recognition data from one party to another when authorized to do so.

MovieLabs has reviewed metadata from several content recognition sources and developed this specification to serve is a format for the delivery of descriptive metadata from studios to content recognition companies. This complements and is compatible with other MovieLabs specifications such as Content Rules and Rights (CRR) (www.movielabs.com/CRR).

MovieLabs has developed a Common Metadata document and XML schema that defines metadata exchange data elements. Content Recognition metadata leverages off the more general Common Metadata (www.movielabs.com/md).

1.1 Document Organization

This document is organized as follows:

1. Introduction—Provides background, scope and conventions
2. Using Content Recognition Metadata – Describes example use cases and how to apply this specification to those use cases.
3. Common Metadata Derived Types – Types that relate directly to Common Metadata encodings. This includes Basic Metadata descriptions of the content, and Digital Metadata, track-by-track information.
4. Content Rules and Rights (CRR) Encoding – Types that relate to Content Rules and Rights. This expresses actions that may be taken upon recognition.
5. Content Recognition Unique Types – Types specifically related to the process of Content Recognition, primarily metadata related to terms, identifiers, hashes, fingerprints and watermarks.

1.2 Document Notation and Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119]. That is:

- “MUST”, “REQUIRED” or “SHALL”, mean that the definition is an absolute requirement of the specification.
- “MUST NOT” or “SHALL NOT” means that the definition is an absolute prohibition of the specification.
- “SHOULD” or “RECOMMENDED” mean that there may be valid reasons to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.

- "SHOULD NOT" or "NOT RECOMMENDED" mean that there may be valid reasons when the particular behavior is acceptable, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
- "MAY" or "OPTIONAL" mean the item is truly optional, however a preferred implementation may be specified for OPTIONAL features to improve interoperability.

Terms defined to have a specific meaning within this specification will be capitalized, e.g. "Track", and should be interpreted with their general meaning if not capitalized.

Normative key words are written in all caps, e.g. "SHALL"

1.2.1 XML Conventions

XML is used extensively in this document to describe data. It does not necessarily imply that actual data exchanged will be in XML. For example, JSON may be used equivalently.

This document uses tables to define XML structure. These tables may combine multiple elements and attributes in a single table. Although this does not align with schema structure, it is much more readable and hence easier to review and to implement.

Although the tables are less exact than XSD, the tables should not conflict with the schema. Such contradictions should be noted as errors and corrected.

1.2.1.1 Naming Conventions

This section describes naming conventions for Common Metadata XML attributes, element and other named entities. The conventions are as follows:

- Names use initial caps, as in InitialCaps.
- Elements begin with a capital letter, as in InitialCapitalElement.
- Attributes begin with a lowercase letter, as in InitiaLowercaseAttribute.
- XML structures are formatted as Courier New, such as `crmd:CRMetadata`
- Names of both simple and complex types are followed with "-type"

1.2.1.2 Structure of Element Table

Each section begins with an information introduction. For example, "The Bin Element describes the unique case information assigned to the notice."

This is followed by a table with the following structure.

The headings are

- Element—the name of the element.
- Attribute—the name of the attribute

- Definition—a descriptive definition. The definition may define conditions of usage or other constraints.
- Value—the format of the attribute or element. Value may be an XML type (e.g., "string") or a reference to another element description (e.g., "See Bar Element"). Annotations for limits or enumerations may be included (e.g.," int [0..100]" to indicate an XML xs:int type with an accepted range from 1 to 100 inclusively)
- Card—cardinality of the element. If blank, then it is 1. Other typical values are 0..1 (optional), 1..n and 0..n.

The 1st header of the table is the element being defined here. This is followed by attributes of this element. Then it is followed by child elements. All child elements (i.e., those that are direct descendents) are included in the table. Simple child elements may be full defined here (e.g., "Title", " ", "Title of work", "string"), or described fully elsewhere ("POC", " ", "Person to contact in case there is a problem", "See POC Element"). In this example, if POC was to be defined by a complex type would be handled defined in place ("POC", " ", "Person to contact in case there is a problem", "POC Complex Type").

Following the table is as much normative explanation as appropriate to fully define the element.

Examples and other informative descriptive text may follow.

1.3 Common Metadata

Common Metadata is defined MovieLabs report TR-META-CM, currently in draft and available upon request. The current structures are pretty well established, but the documenting is rapidly being expanded and some corrections are being made.

Common Metadata is defined to be a collection of commonly used structures that can be incorporated into other specifications. A fingerprint vendor metadata spec will be developed specifically for this application. Common Metadata contains means of communicating descriptive information, data about encoded tracks, packaging information and business rules. Only those portions that apply will be included.

1.4 Normative References

[CM] *Common Metadata, 'md' Namespace, Motion Picture Laboratories*, Technical Report, TR-META-MD, Version 1.0, January 5, 2010, at http://www.movielabs.com/md

[CMXSD] *Common Metadata 'md' namespace schema, v1.0:* *http://www.movielabs.com/md/md/v1.0/md.xsd*

[CRR] *Content Rules and Rights* TR-CRR1, v1.1.1, July 8, 2008 http://www.movielabs.com/CRR/

[CRRXSD] *Content Rules and Rights, 'crr' namespace schema:* *http://www.movielabs.com/CRR/rules.xsd*

1.5 Terms, Definitions and Acronyms

Asset – A unit of content or a piece of media.

Cam – Informal term for an image obtained using a camcorder, typically in a movie theater. Cam's are often moderate to poor quality.

Cryptographic Hash—a *hash* generated by a cryptographic hash algorithm such as SHA-1 or MD5. The hash is for all practical purposes unique to the file, with even single bit changes to the file substantially changing the hash. Applications, including P2P file sharing programs, use hashes to identify files. A hash is unique to the bits that comprise a file, not to the media content.

File – A file containing content to be recognized. Files of interest contain assets.

Fingerprint—A set of bits generated from an asset such that fingerprints generated from an unknown asset can be used to associate the unknown asset with one or more original assets by comparing the fingerprint from the unknown asset to a reference database of fingerprints from original assets. Fingerprints are unique to media content, not to the bits that comprise a file.

Hash—***n.*** data generated by a hash algorithm. ***v.*** to create hash data from digital data. Hash hashes may include *cryptographic hash*, checksums, and cyclic redundancy checks. Generally, in the discussion of media identification, *hash* refers to a cryptographic hash. However, in the context of this document, non-cryptographic hash functions may apply. Note: Fingerprints are sometimes referred to as hashes, but this document does not consider fingerprints hashes.

Identifier – A sequence of bits, usually represented as a number or string, used as reliable shorthand in a particular context for referencing a set of information. Examples include database keys, product SKUs, and email addresses.

ISAN – International Standard Audiovisual Number, a standard identifier for audiovisual information.

P2P – Peer to Peer. Refers to distributed networks where file exchange occurs between user's computers.

Referenced work – Original Asset. A file containing the "referenced work" would, for example, be used to generate a fingerprint.

Scanning – Process of locating P2P clients involved in sharing a particular set of files.

UUID – Universally Unique Identifier; 'universally unique' is often interpreted as 'highly probably unique'. See References section.

Verified Hash – A hash generated by a Hash Verification under the supervision of a Hash Authority, in accordance with this specification.

Viewer – A human watching video or stills, or listening to audio.

Watermark – An identifier that can be added to the audio and/or video of an asset for later extraction. Ideal watermarks in the context of content recognition systems generally are non-removable, imperceptible to human senses, and immune to standard AV transformations. An **Identification Watermark** is a type of watermark that identifies content, typically by carrying data (payload) that is unique to content.

1.6 Encoding

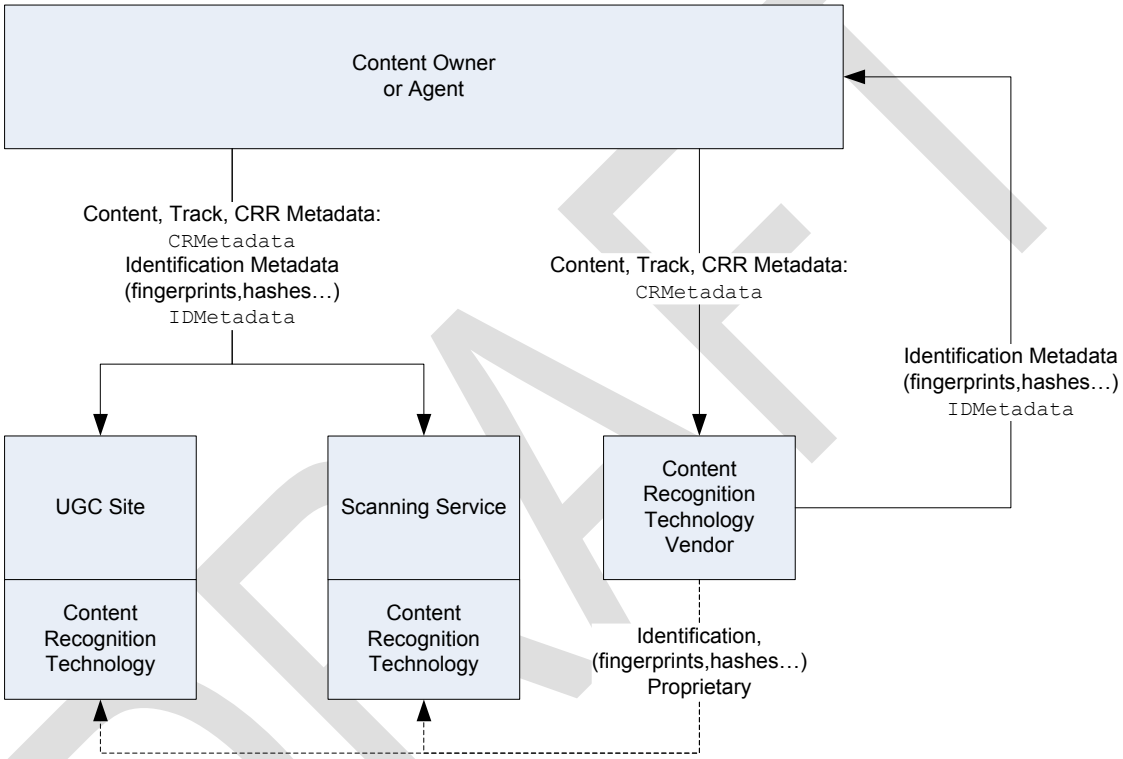
Encoding and enumeration of element and attributes inherited from other specifications SHALL use the definitions from those specifications unless otherwise stated in this specification.

2 USING CONTENT RECOGNITION METADATA

This document is a collection of types and element definitions that can be incorporated into other specifications.

It is designed with specific use cases in mind, but is general enough to be used in other contexts as well. This section describes a few anticipated use cases and describes recommend document application.

The following illustrates how these definitions can be used:



2.1 Use Case 1: Metadata to content recognition vendors

A content owner or its agent (e.g., distributor or producer) wishes to add their content to a content recognition vendor's database. Vendors use metadata for various purposes, including user interfaces, reports and actions to be taken upon discovery.

A typical example is a studio working with a fingerprint vendor. The fingerprint may be generated by a tool at the studio and the fingerprint is transferred with metadata. Or the studio transfer media data and metadata to the vendor who then generates the fingerprint themselves. Although there may be additional metadata required for the process of fingerprinting, the metadata describing the content is the same.

MovieLabs evaluated four fingerprint technology vendors' metadata requirements and concluded the metadata in the 'Content Description Types' section below meets their requirements. The evaluation document is available with conditions upon request.

This specification defines a minimal subset that should accommodate most parties, however, the metadata definition is much broader and other elements may be used as appropriate.

Metadata is in three groups:

- Basic Metadata – Descriptions of the content, including information such as work type, runtime, title, and episode/season information. Basic Metadata is required.
- Digital Asset Metadata – Descriptions of each audio, video and subtitle track, including, for example, language. Digital Asset Metadata is optional.
- Content Rules and Rights – Descriptions of actions to be taken upon recognition. Some content recognition vendors accept these data and take actions accordingly. CRR types are described 'Content Rules and Rights (CRR) Types' below.

The `CRMetadata` element in the schema can be used for transferring metadata:

Element	Attribute	Definition	Value	Card.
CRMetadata				
BasicMetadata		Basic Metadata	crmd:BasicMetadata-type	
DigitalAssetList		List of Digital Asset on a track-by-track basis	crmd:DigitalAssetList-type	0..1
RuleList		List of Content Rules and Rights	crmd:CRRRuleList-type	0..1

2.2 Use Case 2: Metadata to services using content recognition

In this case, both the metadata and the content recognition information (fingerprints, hashes, names, etc.) are transferred from a repository, for example a trade organization, to a service performing services incorporating content recognition technology. For example, a content owner might transfer fingerprints and metadata to an online scanning service (P2P, site scanning, etc.).

The schema, as defined in 'Identification Metadata Types' below contains definitions for transferring individual types of recognition information:

- `NamePhraseList-type` – Isnformation used in online searches such as titles and commonly used phrases.
- `AltIdentifierList-type` – Identifiers use to refer to content. This includes both standard identifiers (e.g., ISAN) as well as relevant non-standard identifiers (e.g., studio internal).
- `HashList-type` – File hashes that are known to be associated with the content. General hashes, such as SHA-1 as well as protocol-specific hashes (e.g., eDonkey and Bittorrent).
- `FingerprintList-type` – Fingerprints corresponding with content audio, video or both. If a content recognition service supports multiple fingerprint technologies, the structure supports delivery of multiple sets of fingerprints.

- `IDWatermarkList-type` – Payload information associated with one or more identification watermarks

The `Identification-type` complex type, defined below, and the `IDMetadata` element consolidates these into a single complex type and a single element.

Element	Attribute	Definition	Value	Card.
IDMetadata		Identification Metadata, possibly including Names and Phrases, Identifiers, Hashes, Fingerprints and Identification Watermark payloads	crmd:Identifcation-type	

2.3 Use Case 3: Metadata to UGC sites

This use case addresses the transfer of recognition information, metadata and rules to a User Generated Content (UGC) site that is performing content recognition. The UGC site uses content recognition to detect content upon upload and then takes some action.

CRMetadata defines the structure for transferring Basic and Digital Asset Metadata. It also provides the structure for passing Content Rules and Rights (CRR), which defines action to be taken upon detect.

If content recognition information is passed directly to the UGC, the `Identification-type` type or subtypes should be used. Typically a UGC gets fingerprints directly from a fingerprint technology vendor who got fingerprints via Use Case 1 mechanisms. However, other forms of identification (names/phrases, identifiers, etc.) should still be passed through identification structures defined in this specification.

3 CONTENT DESCRIPTION TYPES

This section defines types and elements that describe content.

MovieLabs' Common Metadata includes elements that cover typical definitions of media, particularly movies and television. Basic Metadata includes descriptions such as title and artists. It describes information about the work independent of encoding. Physical metadata describes information about individual encoded audio, video and subtitle streams, and other media included. Package and File Metadata describes one possible packaging scenario and ties in other metadata types. Ratings and Parental Control information is described.

Common Metadata is designed to provide definitions to be inserted into other metadata systems. The following type is derived directly from Common Metadata:

Content Recognition (crmd) Type	Common Metadata (md) Type
BasicMetadata-type	md:BasicMetadata-type

Element	Attribute	Definition	Value	Card.
DigitalAssetList-type				
DigitalAsset		Single digital asset (audio, video, etc.) metadata	md:DigitalAssetMetadata-type	1..n

3.1 Fingerprint Metadata-specific Usage Rules

The following defines whether metadata MAY or SHOULD be included (noted as optional) or SHALL be included (not noted as optional), or otherwise included as noted. Any metadata not listed here SHALL not be included in the Basic or Digital Content Metadata.

- BasicMetadata-type
  - ContentID attribute
  - UpdateNum—SHALL be included if the record is an update (i.e., not the first record distributed)
  - LocalizedInfo
    - TitleDisplay19
    - TitleSort
    - OriginalTitle
    - Summary400
    - ArtReference – At least one instance is mandatory, additional instances are optional
    - CopyrightLine
  - RunLength

- ReleaseYear, ReleaseDate and ReleaseDateTime SHOULD include the highest date/time resolution available
- WorkType
- PictureColorFormat—optional, but it SHOULD be included
- PictureFormat—optional, but it SHOULD be included
- AltIdentifier—optional, but it SHOULD be included for all commonly used identifiers. For example, if ISAN is available, it should be included.
- People—SHALL include director and primary cast
- SequenceInfo and Parent—SHALL be included for the following work types: Season, Episode, Promotion, Excerpt, Supplemental
- Parent –SHALL be included for work type of Non-episodic Show if that show is part of a season or series.

- DigitalAssetMetadata-type—SHALL be included for each track included in the Container. Only Language elements are required. All others are optional.
  - Audio
    - Encoding
      - Codec
      - CodecType—The IANA namespace SHALL be used
      - BitrateMax
      - SampleRate
      - SampleBitDepth
    - Language
    - Channels
  - Video:
    - Encoding
      - Codec
      - CodecType
      - BitrateMax
    - Picture:
      - Aspect Ratio
    - SubtitleLanguage—SHALL be included if the video contains visible subtitles.
- Subtitle (if applicable)
  - Format
  - FormatType
  - Language

4 CONTENT RULES AND RIGHTS (CRR) TYPES

MovieLabs' Content Rules and Rights (CRR), found at www.movielabs.com/CRR, describes actions to be taken upon detection. This is an optional component of fingerprint metadata.

The following type is identical to the CRR `RuleList` element, except that it is defined as a type.

Element	Attribute	Definition	Value	Card.
CRRRuleList-type				
	version	Current version is 1	xs:integer	
	revision	Current revision is 1	xs:integer	
RuleListName		Name for this set of rules; intended for incorporation into human-readable logs and statistical analysis	xs:string	0..1
	version	Optional version number for this list of rules; no default	xs:integer	0..1
	revision	Optional revision number for this list of rules; no default	xs:integer	0..1
RuleListCreationTime		Creation time for this RuleList	xs:dateTime	0..1
RuleListID		Identifier for this RuleList internal to the supplier of the list. It is intended to be something that is easier to use in automated handling of notifications and ingestion status than the RuleListName	xs:string	0..1
RuleListValidDuration		Period for which this RuleList applies. If not present, the RuleList is always valid.	crr:TimeInterval	0..1
SiteConcerned		Informational field describing the site for which the rules are intended (if known.)	xs:anyURI	0..1
Owner		Information about the content owner.	crr:Owner	
AssetList		Contains one or more assets to which the Rules apply. The file is valid if this is not present. See the Templates section.	crr:Asset	0..1
Rule		One or more individual Rule elements. If no rules are specified, no actions are taken on detection. This makes it possible to accept original assets that don't require detection rules without requiring a different ingestion path – everything has a rules file.	crr:Rule	0..1

5 CONTENT IDENTIFICATION TYPES

This section describes new types specific to content recognition metadata. There are multiple means of identifying content. These include:

- Names and Phrases – Titles, called 'names' here to distinguish from other uses of 'title', and phrases associated with a work. These are common search terms.
- Identifiers – Multiple identifiers may refer to a work.
- Hash – Cryptographic and other hashes identify files that contain a work. Variations may exist for protocols, particularly P2P protocols.
- Fingerprint – Fingerprints associated with vendors' identification technologies.
- Identification Watermark – Watermark technologies can embed a unique code (payload) in audio or video that identifies the work. Note that this is distinguished from watermarks that do not identify the content, and are not addressed here.

5.1 Identification-type

The Identification-type complex type packages each of the various types of identification.

Elements are all optional, although elements of this type should not be included if none are present. Each element within is structured as a list with at least one element, so the inclusion of any element ensures at least one entry.

Element	Attribute	Definition	Value	Card.
Identification-type				
NamePhraseList		Name and Phrase list	crmd:NamePhraseList-type	0..1
AltIdentifierList		Identifiers.	crmd:AltIdentifierList-type	0..1
HashList		List of hashes identifying files.	crmd:HashList-type	0..1
FingerprintList		Information about fingerprints, and optionally the fingerprint itself.	crmd:FingerprintList-type	0..1
IDWatermarkList		Information about watermarks that identify the work,	crmd:IDWatermarkList-type	0..1

5.1.1 Name (Title) and Phase

Media can be described by titles and phrases. The NamePhrase element describes terms used to refer to a work. Note that although 'Name' will typically be a title, the term 'title' is not used to avoid confusion with other uses of the term 'title'.

5.1.1.1 NamePhraseList-type

This complex type allows numerous name-phrase collections to be expressed. Expected usage is that one `NamePhraseList` will be included without a `Language` element indicating that the `NamePhraseList` covers all languages, or a `NamePhraseList` will be included for each language.

Element	Attribute	Definition	Value	Card.
NamePhraseList-type				
NamePhraseList		List name-phrase sets	crmd:NamePhraseSet	1..n

5.1.1.2 NamePhraseSet-type

This complex type allows related collections of names and phrases to be included.

Element	Attribute	Definition	Value	Card.
NamePhraseSet-type				
Language		Language to which names and phrases apply	xs:language	0..1
Locale		Locales to which names and locations apply.	md:Region-type	0..n
Name		Name of work, typically a title	xs:string	1..n
	PrimaryName	If primary is 'true' the Name refers to the published title (e.g., "Terminator 2", not "T2").	xs:string	0..1
Phrase		Any combination of terms that may identify the work.	xs:string	0..n

Applicability of names and phrases depends on the combination of language and locale as follows:

- If neither is present, names and phrases apply to all languages world-wide
- If language is present, but not locale, names and phrases apply to that language world-wide
- If locale is present, but not language, names and phrases apply to the locale, regardless of language
- If both language and locale are present, names and phrases apply to the language, but only in the stated locale(s)

Encoding of episodic material's season and episode, of movie series or other ordered media types are typically an implicit phrase for a title. For example, of a show 'My Show', season 3, episode 4 might appear as "S3E4", "s03e04" or any number of other variants. As there are numerous variations, these should be inferred from season and episode information in the metadata.

5.1.2 Identifiers

There are various identifiers that may refer to a particular work. The AltIdentifiersList-type and its subtypes allow are arbitrary number of identifiers to be listed along with their namespace (e.g., "ISAN").

Identifiers are expressed as defined in Common Metadata [CM][CMXSD]. This accommodates standard and well-known identifiers as well as organization-specific internal identifiers.

When this definition is used as part of a broader schema, a primary identifier typically exists elsewhere in the structure, hence the naming 'Alt' for alternate. However, there is no reason this list cannot be inclusive. Those using this type should be clear about whether this list is exhaustive.

5.1.2.1 AltIdentifierList-type

Each entry in this complex type is an identifier. It is based on *Common Metadata* `ContentIdentifier-type`.

Element	Attribute	Definition	Value	Card.
AltIdentifierList-type				
Identifier		Identifier	md:ContentIdentifier-type	1..n

5.1.3 Fingerprint

Fingerprint data allows content to be recognized, particularly in environments where the content may be transformed from its original form. Although standards exist, generally fingerprints are vendor-proprietary.

5.1.3.1 FingerprintList-type

This complex type allows multiple fingerprint sets to be included.

Element	Attribute	Definition	Value	Card.
FingerprintList-type				
Fingerprint		Fingerprint entry	crmd:Fingerprint-type	1..n

5.1.3.2 Fingerprint-type

The `Fingerprint-type` complex type identifies the fingerprint technology used and optionally contains fingerprint data.

Element	Attribute	Definition	Value	Card.
Fingerprint-type				
Vendor		Vendor or other provider of technology on which fingerprint is	crmd:FingerprintVendor-stringtype	

		based.		
ProductVersionID		Product and version identification.	xs:string	
MediaType		Indicates whether fingerprint covers audio, video or both	crmd:MediaType-stringtype	
Source		Source of file used to generate fingerprint if known.	xs:string	0..1
FingerprintData		Technology-specific fingerprint	xs:base64Binary	1..n (choice)
FingerprintLocation		Location of fingerprint (e.g., file name or URL)	xs:anyURI	(choice)

The triple `Vendor:ProductVersionID:MediaType` unambiguously identifies a fingerprint technology.

`Vendor` is a representation of a fingerprint vendor or relevant technology. This is not a strict enumeration to allow new vendors to be added. However, it is important that vendor names are used consistently. As general guidance, use initial caps (except for acronyms) and no spaces or punctuation. Following are a few examples:

- MPEG7 – for MPEG-7 fingerprint
- INA – Institut National de l'Audiovisuel
- Vobile – Vobile, Inc.
- AudibleMagic – for Audible Magic

`ProductVersionID` identifies the precise version of the technology. In particular, it is used to differentiate between incompatible fingerprints from the same `Vendor`.

`MediaType` indicates whether the fingerprint applies to audio, video or both. Acceptable values are:

- 'audio'
- 'video'
- 'audiovideo'

`Source` indicates the type of material from which the fingerprint was generated. Acceptable values include:

- 'Camcord'
- 'TheatricalMaster'
- 'DVDMaster'
- 'DVDRip'
- 'Other'

`Fingerprint` is the fingerprint encoded in a vendor-specific format using `base64Binary`.

5.1.4 Hashes

The Hash-type complex type allows the encoding of hashes. A given file may have multiple hashes in different forms. There may be a hash for a complete file and there might be hashes for specific portions.

5.1.4.1 HashList-type

This complex type includes as many hashes as are available for the file. A variety of hash types and formats are allowed in subtypes.

Element	Attribute	Definition	Value	Card.
FileHashList-type				
FileHash		A collection of all hashes	crmd:FileHash-type	1..n

5.1.4.2 FileHash-type

This complex type allows for a variety of hash types to be included. In this structure, any known hashes can be included. If an additional format of interest is not included, please contact this document's author.

Element	Attribute	Definition	Value	Card.
FileHash-type				
Filename		Name of the file being hashed. Because hash refers to the contents, not the name, a given file may have multiple names.	xs:string	0..n
FullFile		Hashes of the entire file	crmd:FullFileHash-type	0..n
ProtocolSpecific		Hashes that are specific protocol, such as 'torrent' files for Bittorrent	crmd:ProtocolSpecificHash-type	0..n
Piece		Hashes for subsets of a file (pieces)	crmd:PieceHash-type	0..n

5.1.4.3 FullFileHash-type

This type contains hashes of the complete file. If included, it is recommend that at least a SHA-1 hash is included.

Element	Attribute	Definition	Value	Card.
FullFileHash-type				
HashType		The type of hash used. See HashType enumeration	crmd:HashType-stringtype	
FileHashValue		Hash value of entire file according to HashType encoding	xs:base64Binary	

5.1.4.4 ProtocolSpecificHash-type

Some protocols, particularly Bittorrent, have a specific means of specifying hashes. Although other information may exist in the torrent file, these are to be ignored.

Element	Attribute	Definition	Value	Card.
FullFileHash-type				
Bittorrent		Bittorrent torrent file	crmd:TorrentHash-type	(choice)
ED2K		ED2K hashes	crmd:ED2KHash-type	(choice)

5.1.4.4.1 Bittorrent Types

Element	Attribute	Definition	Value	Card.
TorrentHash-type				
TorrentName		A file name for the torrent file. Note that a torrent file may be posted with multiple names.	xs:string	0..n
Torrent		Torrent file itself	xs:base64Binary	

5.1.4.4.2 ED2K types

ED2K (eDonkey and Kademlia) hashes use the

Element	Attribute	Definition	Value	Card.
ED2KHash-type				
Ed2kURI		An ed2k: scheme URI.	xs:anyURI	1..n
ChunkHash		Chunk Hash.	crmd:ChunkHash-type	0..n

The Ed2kURI is of type “file”. Non-hash fields are ignored. MD4-HASH is required. AICH Root hash (“h=”) is optional. Part Hashes (chunk hashes) (“p=”) are optional.

This ChunkHash-type type contains the chunk has and optionally the 53 subhashes that are used to construct the complete AICH tree.

Element	Attribute	Definition	Value	Card.
ChunkHash-type				
	chunkno	Which chunk, starting from 0	xs:integer	
ChunkHash		The MD4 hash of the whole chunk. If present, then all 53 entries must be included.	xs:base64Binary	
ChunkHash		Chunk Hash.	xs:base64Binary	0..53
	subno	Subhash number (0-52)	xs:integer	

5.1.4.4.3 NZB types

This type is an NZB file relating. This is not strictly a hash, but it does define a file, so it included here. This is conformant with NZB documentation here: http://docs.newzbin.com/index.php/Newzbin:NZB_Specs. This document does not replicate NZB documentation.

Element	Attribute	Definition	Value	Card.
NZB-type				
head		head	crmd:NZBHead-type	
file		file	crmd:NZBFile-type	

Element	Attribute	Definition	Value	Card.
NZBHead-type				
meta		meta	xs:string	1..n
	type	type	xs:string	

Element	Attribute	Definition	Value	Card.
NZBFile-type				
	subject	subject	xs:string	
	poster	poster	xs:string	0..1
	date	date	xs:int	0..1
groups		groups	crmd:NZBGroups-type	
segments		segments	crmd:NZBSegments-type	

Element	Attribute	Definition	Value	Card.
NZBGroups-type				
group		group	xs:string	1..n

Element	Attribute	Definition	Value	Card.
NZBSegments-type				
segment		segment	xs:string	1..n
	bytes	bytes	xs:int	
	number	number	xs:int	

5.1.4.4.4 RAR types

RAR (Roshal ARchive) files do not have a specific hash associated with the archive itself. RAR may be a single file or a collection of files. The following types allow identification of RAR files.

There is not a specific hash associated with the RAR file itself, so a conventional hash (e.g., SHA-1) is used instead.

Element	Attribute	Definition	Value	Card.
RARHash-type				
FilenameContained		The name(s) of the file(s) contained in the archive.	xs:string	1..n
FilenameBase		Either the name of a single-file archive or the name of the first file in a multi-part archive.	xs:string	1..n
Piece		Hash information for each piece. If the archive is one file, then this is the hash of the entire file.	crmd:RARPieceHash-type	1..n

FilenameBase is the RAR file's name if the archive is a single file (e.g., filename.rar). If the archive is divided into parts, this is the filename of the the first part (e.g., filename.part01.rar or filename.r01). Two set of file may be binary identical but have different file names. In this case, the base filename for each archive may be included (e.g., filename.part01.rar and anotherfilename.part01.rar).

Element	Attribute	Definition	Value	Card.
RARPieceHash-type				
PartNo		The number of the part, starting with 1.	xs:integer	
Type		Type of hash used	xs:HashType-stringtype	
Hash		Hash for piece using the hash associated with element 'Type'	base64Binary	

5.1.4.5 PieceHash-type

This is an alternate mechanism for encoding piece hashes beyond what is defined under protocol-specific hashes (`ProtocolSpecificHash-type`). For protocols whose definitions are in `ProtocolSpecificHash-type`, it is preferred to use that structure.

Element	Attribute	Definition	Value	Card.
PieceHash-type				
PieceType		Type of hash used. See HashType enumeration.	crmd:HashType-stringtype	
Piece		Information about the piece	crmd:PieceHashDescr-type	
Protocol		Protocol associated with pieced	crmd:HashProtocol-string	0..1

As this structure is defined to support protocols not specifically enumerated in protocol-specific structures, this PieceType is not a controlled enumeration.

5.1.4.6 PieceHashDescr-type

Element	Attribute	Definition	Value	Card.
PieceHashDescr-type				
ByteOffset		Offset from beginning from file, in bytes. First byte of file is offset 0.	xs:integer	
SizeInBytes		Size of piece hashed, in bytes.	xs:integer	
PieceHashValue		Value of piece hash	xs:base64Binary	

5.1.4.7 HashType enumeration

HashTypes may refer to any type of hash (see definitions above).

- Cryptographic Hash
  - 'SHA1'
  - 'SHA2-224'
  - 'SHA2-256'
  - 'SHA2-384'
  - 'SHA2-512'
  - 'MD2'
  - 'MD4'
  - 'MD5'
- CRC
  - 'CRC16'
  - 'CRC32'
  - 'CRC64'
- Checksum
  - 'BSD'
  - 'SYSV'
  - 'SUM8'
  - 'SUM16'
  - 'SUM32'
- Protocol Specific
  - 'BTInfoHash' – Bittorrent info_hash
  - 'ED2KAICHTop' – eDonkey AICH Top Hash

- ‘ED2KMD4Hash’ – eDonkey MD4 Hash. Note that this is an ED2K-specific hash, that uses MD4 but is not the same as a MD4 hash of the entire file.

Hashes of types other than mentioned above may be included. By convention, use the name of the hash in all caps without spaces or unnecessary punctuation.

5.1.5 ID Watermark

Identification watermarks contain information that identifies content.

5.1.5.1 IDWatermarklist-type

This complex type allows the inclusion of multiple watermarks.

Element	Attribute	Definition	Value	Card.
IDWatermarkList-type				
IDWatermark		Watermark description and payload	crmd:IDWatermark-type	1..n

5.1.5.2 IDWatermark-type

This complex type describes which watermark is used and also includes payload information used for recognition.

Element	Attribute	Definition	Value	Card.
IDWatermark-type				
Vendor		Organization associated with watermark.	crmd:WatermarkVendor-stringtype	
ProductVersionID		Identification of specific watermark method. Must be globally unique (i.e., no just unique to the vendor).	xs:string	
MediaType		Indicates whether watermark is carried on audio, video or both	crmd:MediaType-stringtype	
Payload		ID Payload. If absent, payload is assumed to be known by both parties.	xs:base64Binary	0..n

The triple `Vendor:ProductVersionID:MediaType` unambiguously identifies a watermark technology.

`Vendor` is a representation of a watermark vendor or relevant technology. This is not a strict enumeration to allow new vendors to be added. However, it is important that vendor names are used consistently. As general guidance, use initial caps (except for acronyms) and no spaces or punctuation. Following are a few examples:

- ‘Philips’
- ‘Civolution’

- 'Verance'
- 'Nielsen'
- 'AACS'

`ProductVersionID` identifies the precise version of the technology. In particular, it is used to differentiate between incompatible watermarks from the same `Vendor`.

`MediaType` indicates whether the watermark is embedded in audio, video or both. Acceptable values are:

- 'audio'
- 'video'
- 'audiovideo'

`Payload` is the watermark data in a vendor-specific format using `base64Binary`.

MediaType SHALL be one of the following values:

- 'audio' – Watermark is only on audio.
- 'video' – Watermark is only on video.
- 'audiovideo' – Watermark is on audio and video

6 REDEFINABLE STRINGS

The following strings are designed to be redefined using xs:redefine in schemas using the crmd namespace and wish to force enumeration.

- MediaType-stringtype
- FingerprintVendor-stringtype
- FingerprintSourceContent-stringtype
- WatermarkVendor-stringtype
- HashType-stringtype

Following is an example of redefine:

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:crmd="http://www.movielabs.com/crmd" xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.movielabs.com/crmd" elementFormDefault="qualified"
attributeFormDefault="unqualified">
   <!--*** Redefine to enumerate strings in crmd.xsd ***-->
   <xs:redefine schemaLocation="crmd.xsd">
      <xs:simpleType name="MediaType-stringtype">
         <xs:restriction base="crmd:MediaType-stringtype">
            <xs:enumeration value="audio"/>
            <xs:enumeration value="video"/>
            <xs:enumeration value="audio-video"/>
         </xs:restriction>
      </xs:simpleType>
      <xs:simpleType name="FingerprintVendor-stringtype">
         <xs:restriction base="crmd:FingerprintVendor-stringtype">
            <xs:enumeration value="MPEG7"/>
            <xs:enumeration value="INA"/>
            <xs:enumeration value="Vobile"/>
            <xs:enumeration value="AudibleMagic"/>
         </xs:restriction>
      </xs:simpleType>
      <xs:simpleType name="FingerprintSourceContent-stringtype">
         <xs:restriction base="crmd:FingerprintSourceContent-stringtype">
            <xs:enumeration value="Camcord"/>
            <xs:enumeration value="TheatricalMaster"/>
            <xs:enumeration value="DVDMaster"/>
            <xs:enumeration value="DVDRip"/>
            <xs:enumeration value="Other"/>
         </xs:restriction>
      </xs:simpleType>
      <xs:simpleType name="WatermarkVendor-stringtype">
         <xs:restriction base="crmd:WatermarkVendor-stringtype">
            <xs:enumeration value="Philips"/>
            <xs:enumeration value="Civolution"/>
            <xs:enumeration value="Verance"/>
            <xs:enumeration value="Nielsen"/>
            <xs:enumeration value="AACS"/>
         </xs:restriction>
      </xs:simpleType>
      <xs:simpleType name="HashType-stringtype">
         <xs:restriction base="crmd:HashType-stringtype">
            <xs:enumeration value="SHA1"/>
            <xs:enumeration value="SHA2-224"/>
            <xs:enumeration value="SHA2-256"/>
            <xs:enumeration value="SHA2-384"/>
            <xs:enumeration value="SHA2-512"/>
            <xs:enumeration value="MD2"/>
            <xs:enumeration value="MD4"/>
```

```
            <xs:enumeration value="MD5"/>
            <xs:enumeration value="CRC16"/>
            <xs:enumeration value="CRC32"/>
            <xs:enumeration value="CRC64"/>
            <xs:enumeration value="BTInfoHash"/>
            <xs:enumeration value="ED2KAICHTop"/>
            <xs:enumeration value="ED2KMD4Hash"/>
            <xs:enumeration value="BSD"/>
            <xs:enumeration value="SYSV"/>
            <xs:enumeration value="SUM8"/>
            <xs:enumeration value="SUM16"/>
            <xs:enumeration value="SUM32"/>
         </xs:restriction>
      </xs:simpleType>
   </xs:redefine>
</xs:schema>
```