Open Data Management & Cloud
exam project

Audio music file archiving

Patrick Indri

May 6, 2020

Structure of the project

Introduction

Aim of the project

Investigation of audio file archiving for music.

In particular:

- UML metadata model;
- XSD implementation and XML sample document;
- discussion of data discovery/access and interoperability;
- discussion of long term archiving and data preservation.

Data resource: not an actual dataset but music files in general.

Metadata standards for audio files

There is no widely used and standardised metadata model for music audio files.

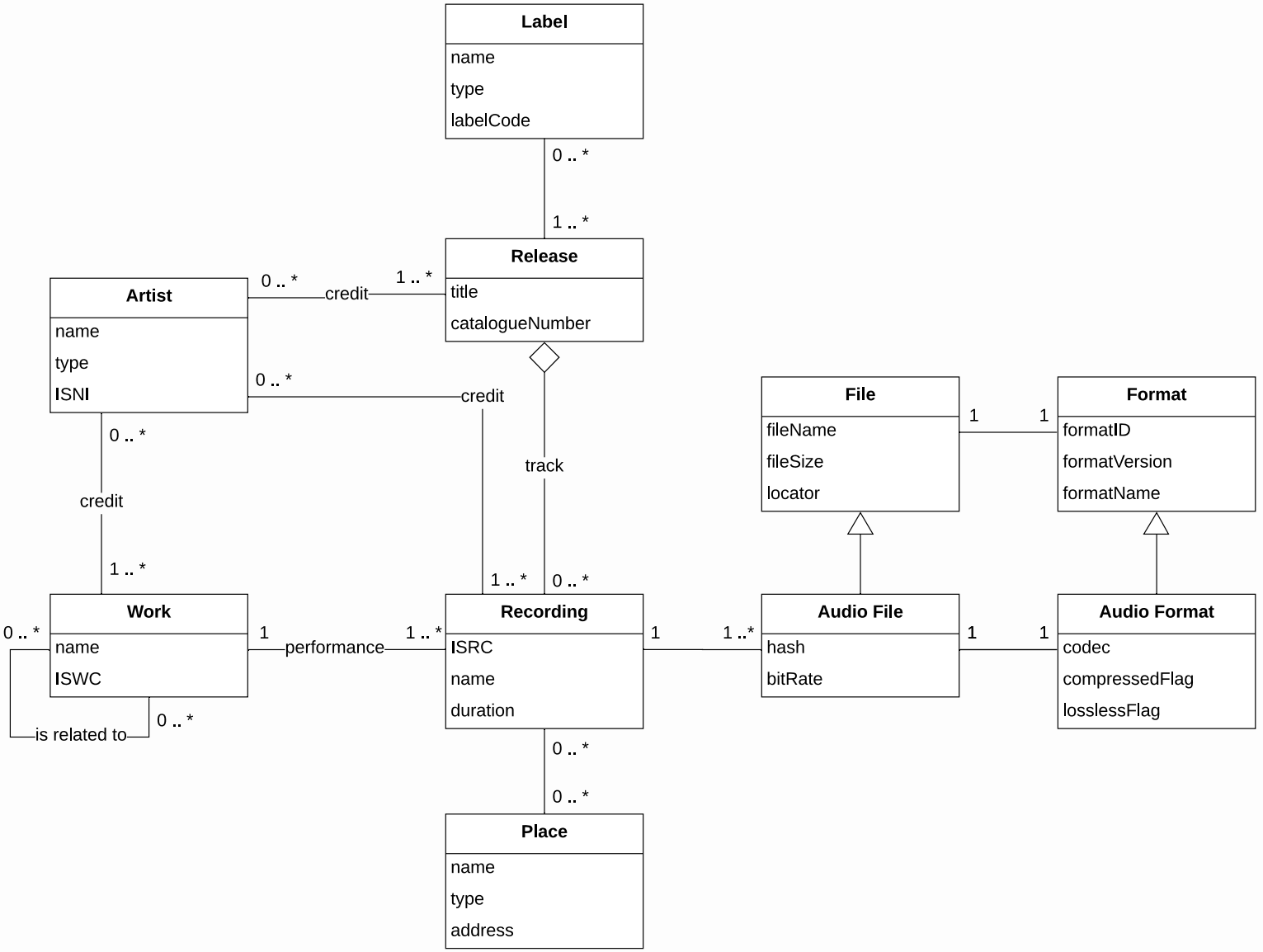
- Dublin Core: simple (15 terms), focus on descriptive metadata;
- EbuCore: detailed DC extension, fine grain technical and administrative metadata for broadcasting;
- METS: handles the structural/hierarchical metadata of a digital library. Open flexibility (no vocabulary).

Model design

What should the data model represent?

- Songs and their different versions;
- Groups of songs (releases);
- Artists;
- Basic technical metadata;
- Relations between songs, releases and artists.

UML


Label
name
type
labelCode
0 .. *
1 .. *
Release
title
catalogueNumber
Artist
name
type
ISNI
0 .. *
credit
1 .. *
0 .. *
credit
0 .. *
credit
track
1 .. *
0 .. *
File
fileName
fileSize
locator
1
1
Format
formatID
formatVersion
formatName
Work
name
ISWC
0 .. *
1
performance
1 .. *
Recording
ISRC
name
duration
1
1 ..*
Audio File
hash
bitRate
1
1
Audio Format
codec
compressedFlag
losslessFlag
0 .. *
is related to
0 .. *
0 .. *
0 .. *
Place
name
type
address

XSD

Choice of implementation:

- RDB: easy to enforce constraints (primary/foreign keys), widely used, easy to model relationships, rigid structure;
- **XSD**: flexible, easily handle partial data, harder relationship handling.

The proposed XSD implementation should:

- Refine Dublin Core;
- Balance integrity constraints and partial data;
- Model relationships with detail.

The resulting XSD can be retrieved here.

XML example

Example of an XML document, valid against the proposed XSD.

```
<work>
   <ISWC id="ISWC_T−000.000.000−A"></ISWC>
   <title lang="en">
      <dc:title>Test Work</dc:title>
   </title>
   <hasArtist label="Will Wilson" description="Singer">
   </hasArtist>
   <hasPerformance label="Test Rec." description="Studio Ver.">
      <relationIdentifier>
         <ISRC idref="ISRC_AAAAA0000000"></ISRC>
      </relationIdentifier>
   </hasPerformance>
</work>
```

The KEY/KEYREF approach was used to model the relationships.

Difficulties and possible expansions

Difficulties:

- flexibility;

Possible expansions:

- Include sort names;
- Include pictures for artists and releases;
- KEY/KEYREF cross domain.

Data discovery: search/filter service

The most fundamental service would be a search/filter service.

- Free text queries on the various categories (artist, release, work, recording);
- Queries can be implemented using XQuery;
- Information retrieval techniques to improve results: edit distance and k-gram distance for spelling correction;
- Popularity rank for the results.

Once the resource has been identified: download (lossy and lossless file) and online preview.

Crucial point: XML indexing, trade-off between memory usage and performance.

Data annotation

Metadata can be embed in audio files. **Is it advisable?** A minimal amount, the *catastrophic* metadata is necessary. Most metadata should be saved in external files.

Viable metadata containers:

- ID3: designed for Mp3, highly structured;
- XMP: ISO standard for JPEG images, defined for other formats including Mp3 and WAW;
- BWF `<bext>` chunk: XML administrative metadata for BWF;
- Vorbis comments: unstructured metadata for Vorbis and FLAC.

Conclusion: data annotation heavily depends on file format choice.

Storage and Cloud Solutions

Storing **XML** files:

- XML-native databases are not scalable;
- XML-enabled databases are well established, scalable and can be queried with SQL;

Crucial point: the proposed XSD itself is not scalable.

Audio files can be stored in BLOBs in the database: not flexible.
Alternatively:

- Up to *few* TB: remote filesystem;
- Large archives: distributed DB;
- Cloud solutions: Amazon S3 offers versioning, orphan files handling, redundancy and access control.

Data preservation

Interoperability

Audio file formats

Final considerations

References