Audio Engineering Society

Convention Paper

Presented at the 123rd Convention
2007 October 5–8 New York, NY

Using the Semantic Web for Enhanced Audio Experiences

Yves Raimond[1], Mark Sandler[1]

[1] *Centre for Digital Music, Queen Mary, University of London*

Correspondence should be addressed to Yves Raimond (`yves.raimond@elec.qmul.ac.uk`)

ABSTRACT

In this paper, we give a quick overview of some key Semantic Web technologies which allow us to overcome the limitations of the current web of documents to create a machine-processable *web of data*, where information is accessible by automated means. We then detail a framework for dealing with audio-related information on the Semantic Web: the Music Ontology. We describe some examples of how this ontology has been used to link together heterogeneous data sets, dealing with editorial, cultural or acoustic data. Finally, we explain a methodology to embed such knowledge into audio applications (from digital jukeboxes and digital archives to audio editors and sequencers), along with concrete examples and implementations.

1. INTRODUCTION

Information management is becoming an important part of multimedia related technologies, ranging from the management of personal collections to the construction of large, distributed, 'semantic' databases. The latter can be addressed, to some extent, using Semantic Web technologies, which allow us to create a *web of data* that is accessible by automated means. This data is mapped onto real-world objects through the use of *ontologies*. The 'Music Ontology' tries to create a formal modular framework for dealing with audio-related information on the Semantic Web: editorial, acoustic, or cultural information. Therefore, it allows us to gather in a distributed knowledge environment a large range of heterogeneous data sets.

On the other hand, audio applications are often completely isolated, and rely only on available *local* storage for keeping track of metadata, outputs of digital signal processing algorithms or other audio-related knowledge. In this paper, we describe how such applications (from digital jukeboxes to audio se-

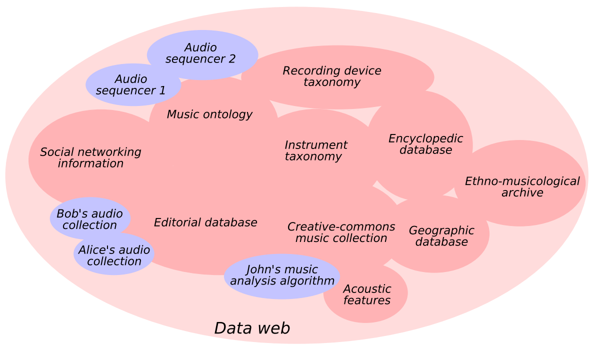

Fig. 1: Interlinking music-related information on the web of data

quencers) can benefit from interacting with a shared and distributed knowledge environment, and therefore cooperate with each other, along with concrete examples and implementations. This relies on the inclusion of content such as personal music collections, a number of audio takes in a studio environment, or computed features in an audio analysis scenario, into this web of data, as represented in fig. 1.

We overview in § 2 the different technologies which let us create such a web of data. They rely on three things: resources, dereferencable identifiers and machine-processable representations including links to other resources. In § 3, we detail some key aspects of the 'Music Ontology', providing a framework to deal with audio-related information on this web of data. Finally, in § 4, we detail how heterogeneous audio applications can benefit from and contribute to this web of data.

2. TOWARDS A WEB OF DATA

2.1. Current limitations

Finding particular information on the current web of documents is a difficult process involving a search engine (an "entry point"), contextualisation of hyperlinks, and skilled manipulation of the resulting data. For example, if we have access to a music library (like a personal collection), and we want to filter out all the works that were not composed in France during a given period of time, we would have to go through a tremendous process—looking for each track on a search engine (using their title, the artist name to disambiguate several performances of a same work, etc.), going through a lot of web pages, and read all of them in order to get back to the composition date and the composition location. Once it is done, we have to enter this information in a playlisting program, which will use it to filter the items according to a given period of time and location corresponding to the location and composition time of the underlying work.

One solution to this problem could be to use a web service accessing a large editorial dataset, such as Musicbrainz[1]. But a problem arises when (especially in such a use case), all the information is not available in this particular dataset. Therefore, we head towards another tremendous task, which is to join the datasets that could answer our particular type of request, and to manually write some glue code between two web services. In our case, we would use Musicbrainz, Wikipedia[2] and a geographic database such as Geonames[3], and the glue code would mainly consist in trying to match similar resources described in all these datasets.

2.2. The Structured Web

It is almost impossible to automate this process, in order to make an application aware of some information available on the web. Therefore, the need for a more structured web which can be directly consumed by automated means is becoming obvious.

Resources available on the web can be far more than just a web page—they can identify a particular work, such as Franz Schubert's Trout quintet, a performer, a release, a particular arrangement, etc. These resources can also, when fetched, provide not only a human-readable representation, but also a machine-processable one, which may include links to other resources ('Franz Schubert **is the composer of** the Trout quintet', linking a resource representing a person to a resource representing a musical work, perhaps located in another dataset).

2.3. Semantic Web technologies

A family of description language have emerged from

[1]see `http://musicbrainz.org/`
[2]see `http://wikipedia.org/`
[3]see `http://geonames.org/`

these requirements, such as Microformats or RDF (Resource Description Framework [1]). Whereas Microformats provide a way to express structured textual data about particular types of resources, RDF provides a generic framework to describe and link together all types of resources, as long as they are identified by an URI (Uniform Resource Identifier [2]).

By adding a *dereferencing* mechanism to these resources, such as the existing web stack (dereferencable HTTP identifiers), we provide access to representations of the resources (either human-processable (free text, HTML page, etc.) or machine-processable (RDF document, HTML with embedded Microformats or RDF)). Therefore, when linking to a resource, we also implicitly provide a way to access its description, which may hold further links.

Using such a web of data, an automated user agent can jump from resource to resource, which may be distributed across heterogeneous datasets, in order to reach a particular information.

2.4. Examples

A simple RDF document written in the Turtle language[4] [3], linking a person whose name is 'John Doe' to a geographic resource (New York, in this example) could be:

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix ex: <http://example.com/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

:john a foaf:Person;
  foaf:name "John Doe"^^xsd:string;
  foaf:based_near <http://sws.geonames.org/5128638/>;
  .
```

Then, by following the `foaf:based_near` link, we can jump to a resource in the Geonames dataset, describing New York, its population, etc.

```
@prefix geo: <http://www.geonames.org/ontology#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<http://sws.geonames.org/5128638/> a geo:Feature;
  geo:name "New York"^^xsd:string;
  geo:population "19274244"^^xsd:int;
  geo:parentFeature <http://sws.geonames.org/6252001/>;
  [...]
  .
```

[4] this language will be used throughout the paper—the set of `@prefix` statements declare a set of namespaces and each block describes a set of statements about one particular resource

Such representations are specified by *ontologies*, formalising the concepts and relationships that can be used in a particular domain—for example, we may write an ontology which specifies that a person can play an instrument in a performance, therefore leading to statements such as:

```
@prefix mo: <http://purl.org/ontology/mo/> .
@prefix ex: <http://example.com/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix mit: <http://purl.org/ontology/mo/mit#> .

ex:cityboomboom a mo:Performance;
  mo:performer ex:julienlourau;
  mo:instrument mit:Saxophone;
  .
```

Here, our ontology is defining the concept of a performance (`mo:Performance`) and relationships linking this concept to an instrument (`mo:instrument`) and to a performer (`mo:performer`).

Ontologies are also part of the Web of Data—instance data provides references to ontologies structuring it, which are then accessible through the same dereferencing mechanism. In our last example, we can dereference `mo:Performance` in order to get to a formal definition of this concept and of the relationships in which it is involved. Ontologies are linked together, as well. For example, when `mo:Performance` is dereferenced, you get access to the following statement:

```
@prefix mo: <http://purl.org/ontology/mo/> .
@prefix event: <http://purl.org/NET/c4dm/event.owl#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

mo:Performance rdfs:subClassOf event:Event.
```

This statements specifies that every instance of a performance will also be an instance of another concept `event:Event`, defined in another ontology.

Therefore, the mechanism to get from the actual *data* to its *meaning* is exactly the same as the one involved in following a link on the web.

3. THE MUSIC ONTOLOGY

In this section, we briefly overview an ontology aiming at structuring music and audio-related knowledge on the Semantic Web: the Music Ontology. A complete description of this ontology is available in [4].

3.1. Groundings

The Music Ontology is linked with four other ontologies, that act as a foundation for a number of audio specific concepts.

3.1.1. Time

Audio related information involves a lot of temporal information. Indeed, we need to address a large range of temporal description, from 'a recording which happened on the 9th of March, 1984' to 'there is someone speaking between 2 minutes 23 seconds and 3 minutes 12 seconds on this audio track' through 'there is a chorus just after the second verse in this popular song'. We consider using two concepts from OWL-Time [5], respectively dealing with intervals and instants, and the relationships linking them (for example, the Allen's relationships between intervals [6]: during, meets, overlaps, etc.). However, OWL-Time doesn't handle references to multiple *timelines*: we may need to express temporal information that refers to the *physical* timeline (eg. 'the 9th of March, 1984'), but we may also need to express information that have a meaning on another timeline, for example backing an audio track (eg. 'from 2 minutes 23 seconds to 3 minutes 12 seconds'). Therefore, we extended the OWL-Time ontology to handle temporal information referring to multiple timelines, which themselves may have multiple coordinate systems to address time points and intervals on them. Moreover, we also provide a way to relate two timelines together (in order to express, for example, the relationship between the timeline supporting an analog signal and the one supporting the sampled signal). The complete ontology is available at [7].

For example, the following code, referring to this ontology, is defining an instant (at 3 seconds) on the timeline backing an audio signal (in this case, 'City boom boom' on the album of the same name by Julien Lourau).

```
@prefix tl: <http://purl.org/NET/c4dm/timeline.owl#> .
@prefix time: <http://www.w3.org/2006/time#> .
@prefix zgtl: <http://zitgist.com/music/timeline/> .
@prefix ex: <http://example.org/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

ex:instant a time:Instant;
  tl:onTimeLine zgtl:87044794-e286-47f4-aa97-87afe1c721cb;
  tl:atDuration "PT3S"^^xsd:duration;
```

.

3.1.2. Event

The workflow leading to the production of a particular audio item involves many physical events, that occur at certain places and times, and that can involve the participation of a number of objects, both animate or inanimate. For example, we may want to describe musical performances, involving a number of instruments, musicians, etc., or recording events, transducing a physical sound field. Moreover, we want to express events that are occurring on other timelines. For example, we may want to express that during a particular interval of an audio signal, someone is speaking. Therefore, we adopt a broad definition of an event: they are the way by which cognitive agents classify arbitrary regions of space-time. An event may have a place, a time, some factors (a musical instrument or a particular type of microphone, for example), some agents (a performer, a sound engineer, etc.) and some products (a physical sound, a signal, etc.). Moreover, an event can also be split into several sub-events, in order to break complex events into simpler events. For example, we might use this mechanism to describe a group performance, by splitting it into a number of parallel sub-events, each of which representing the participation of one performer using one particular instrument. This leads us to the Event ontology [8], which is linked with the Timeline ontology.

3.1.3. Functional Requirements for Bibliographic Records

The FRBR ontology [9] provides a framework for dealing with several useful concepts in the music realm, such as musical works (eg. 'Franz Schubert's Trout Quintet'), or the conceptual difference between a manifestation (such as a particular record) and an item (such as a particular physical disc). The Music Ontology therefore links towards these three concepts. However, it does not reuse the expression concept, as this concept is supposed to cover the whole workflow from a particular work to a particular record. In the scope of the Music Ontology we want to be able to describe the different events constituting this workflow as separate entities: arrangements, performances, etc., as we explain in § 3.2.

3.1.4. Friend of a Friend

The FOAF [10] vocabulary provides a way to describe people, groups of people and organisations. The Music Ontology links to this vocabulary in order to describe performers, composers, sound engineers, conductors, orchestra, bands, record labels, etc. The following small example shows how a person can be described using this vocabulary:

```
@prefix ex: <http://example.com/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

ex:julienlourau a foaf:Person;
   foaf:name "Julien Lourau"^^xsd:string;
  foaf:homepage
   <http://www.label-bleu.com/artist.php?lng=e&artist_id=75>;
   foaf:img
    <http://www.label-bleu.com/Publish/artist/75/lourau_220px.jpg>;
   .
```

3.2. Music production workflow

The Music Ontology builds on top of these four ontologies in order to specify the workflow leading from a particular work to a particular record, as a set of interconnected events, using a similar model as the one used in the ABC ontology [11]. Therefore, the ontology provides a way to specify a composition event (which may be linked to a place, a time, and a person—the composer) which leads to the creation of a musical work, followed by an arrangement event, a performance of the resulting arrangement (involving a conductor, some performers, etc.) and a recording of this performance (involving a particular type of microphone, a sound engineer, etc.). Such a workflow is depicted in fig. 2. At the record level, the ontology supports standard editorial metadata—track title, release title, track number, cover art, liner notes, etc.

The Music Ontology divides itself in three *levels of expressiveness* in order to cover a wide range of possible use-cases. The first level just provides a mechanism to express information at the record level—simple editorial metadata. The second level provides a way to express the above mentioned workflow information, by linking the record to a set of events describing the music production process. The third level provides a mechanism to express time-dependent annotations, such as temporal event decomposition ('this performer was playing that particular instrument at that particular time'), speech/music classification, keywords, onsets, structural segments, etc.

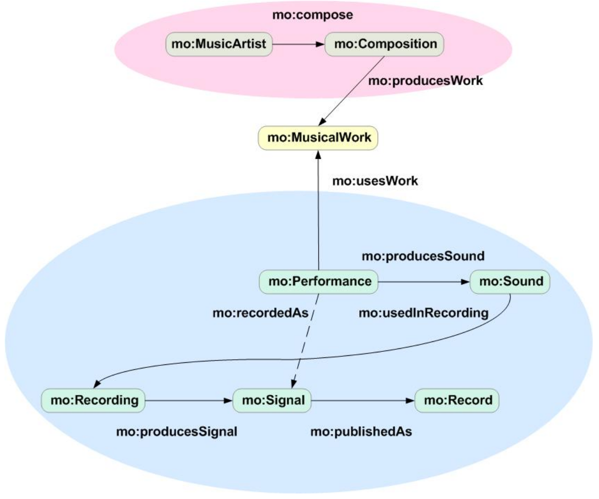

Fig. 2: A music production workflow as defined by the Music Ontology

3.3. Data publishing and interlinking efforts

There are numerous efforts of data publishing and interlinking using this ontology and other related ones. Most of them are part of the 'Linking Open Data on the Semantic Web' community project [12] of the W3C Semantic Web Education and Outreach group, working on open data sets, such as Wikipedia, Musicbrainz, Creative Commons music repositories, Geonames, etc.

So far, in the music realm, the following datasets have been published as Music Ontology instance data and interlinked with other relevant datasets: Musicbrainz, Jamendo, Magnatune, the RSAMD HotBed database and the BBC John Peel sessions[5].

For example, the links between Jamendo and Musicbrainz provide a way to access detailed editorial information from Musicbrainz, as well as the content itself from the Jamendo database. The links from Jamendo to Geonames provide detailed geographic information about the location of artists. Accessing the resource `http://dbtune.org/jamendo/artist/5` indeed gives, among others, the following statement:

[5] see `http://zitgist.com/music/` and `http://purl.org/dbtune/`

```
@prefix foaf: <http://xmlns.com/foaf/0.1/>.

<http://dbtune.org/jamendo/artist/5>
  foaf:based_near
    <http://sws.geonames.org/2991627/>.
```

Getting the Geonames resource can provide the user agent with detailed information about this particular location, as illustrated in § 2.4.

The link from DBPedia (structured information extracted from Wikipedia) to Musicbrainz allows to access encyclopedic information about artists or records held in the Musicbrainz dataset. Accessing the resource `http://dbpedia.org/resource/Metallica` indeed gives, among others, the following statement:

```
@prefix owl: <http://www.w3.org/2002/07/owl#>.
@prefix zg:<http://zitgist.com/music/band/>.

<http://dbpedia.org/resource/Metallica>
  owl:sameAs
    zg:65f4f0c5-ef9e-490c-aee3-909e7ae6b2ab.
```

Getting this last resource can provide the user agent with detailed editorial information coming from the Musicbrainz dataset.

4. AUDIO APPLICATIONS AND THE SEMANTIC WEB

Heterogeneous audio applications can really benefit from accessing and contributing to this knowledge environment. In this section, we first detail some interaction examples between different types of audio applications and the Semantic Web. Then, we detail a general architecture to make audio applications consume and contribute to this web of data.

4.1. Personal audio collection management

A digital jukebox application may hold links from audio items in the local collection to related resources available on the web, using for example the Musicbrainz dataset, providing such entry points to the Semantic Web for about 5 millions tracks. The Music Ontology indeed makes the same distinction as FRBR between manifestations (eg. a particular record or a particular track on a particular record) and items (eg. a particular audio file, but also a particular LP or any *physical embodiment* of a manifestation). The relationship between a manifestation and an item is captured within the `mo:availableAs` predicate.

Therefore, given an audio file accessible on the local file system, the link towards a manifestation held by the Musicbrainz dataset is done through a statement such as:

```
@prefix mo: <http://purl.org/ontology/mo/>.
@prefix local: <file:///home/yves/audio/>.
@prefix mbz: <http://zitgist.com/music/track/>.

mbz:c527cc1a-98b2-47a7-b57b-7a0c5d41a8ae
  mo:availableAs
    local:city_boom_boom.mp3.
```

By keeping track of such statements within a local RDF *cache*, we keep track of entry points to the Semantic Web, which can be dereferenced in order to find more useful information about our audio items. For example, by dereferencing such URIs, we get editorial metadata (release date, track title, etc.) and a link towards an artist identifier (through a `foaf:maker` statement), which can itself be dereferenced. This artist URI gives access to some information (birth date, name, etc.) but also a `owl:sameAs` link towards the corresponding resource in the DBPedia dataset, therefore allowing us to access encyclopedic information about the artist.

Automatically generating such `mo:availableAs` statements is also possible, using either a plain metadata lookup using the Musicbrainz web service or the computation of a fingerprint, matched against the MusicDNS [13] database, and concatenating the resulting identifier to the URI space holding RDF representation of such resources (here, `http://zitgist.com/music/artist/`). An implementation of such a mechanism is the **GNAT** software[6].

Once we have links set up from the items in our collection to the web of data, a large number of applications is possible. For example, we can ask our collection management software to display the geographic location of our artist on a map (fig. 3 gives an example of such a display). Such an interaction actually triggers the dereferencing of the artist resources in our RDF cache, looks for `foaf:based_near` links, and dereferences the corresponding geographic resources.

[6]see `http://moustaki.org/gnat/`

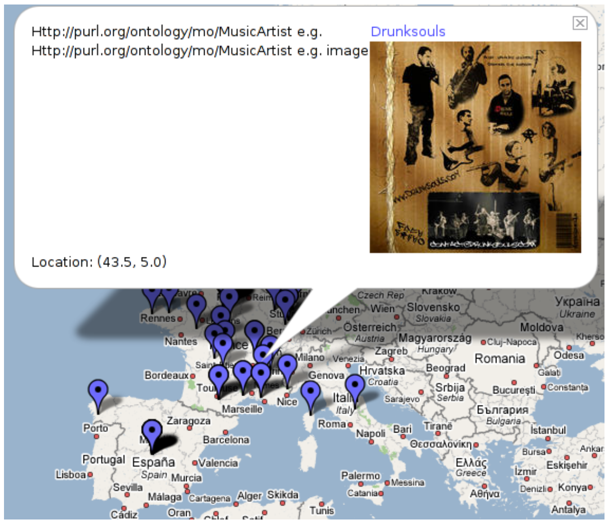
Http://purl.org/ontology/mo/MusicArtist e.g.
Http://purl.org/ontology/mo/MusicArtist e.g. image
Drunksouls
Location: (43.5, 5.0)

Fig. 3: Displaying artists involved in a personal audio collection on a map, using geographical information captured by links from the artist resource to a resource in the Geonames database—a pin denotes the location of an artist

We can also ask our collection management software to display our items on a timeline, according to their release date (fig. 4 gives an example of such a display). This involves a similar underlying mechanism.

Then, by involving a simple SPARQL [14] query over our RDF data, we can generate playlists according to a number of criteria, such as 'create me a playlist involving bands whose members are located in a city which has less than 1000 inhabitants', or 'create me a playlist of artists that are married to each others'.

4.2. Digital libraries

Digital libraries management software can also benefit from such technologies. In most cases, even within a similar framework, two software instances are not aware of each other. Therefore, if the same composer is involved in each dataset, his description will have to be entered twice. There is no way for several archives to automatically collaborate by providing metadata to each other.

This limitation is often linked to another one, due to an asset-centric data model which is often involved in Digital library softwares such as Greenstone [15]: the digital assets are considered as primary objects, on which we attach secondary ones—'meta'-data. Therefore, the expressiveness of the data model is fixed: even if new metadata fields are added, there will always be an emphasis on the digital assets, which makes it impossible to state that two secondary resources (metadata fields) are the same across two datasets. The interlinking, if there is one, is restricted to range over the assets themselves.

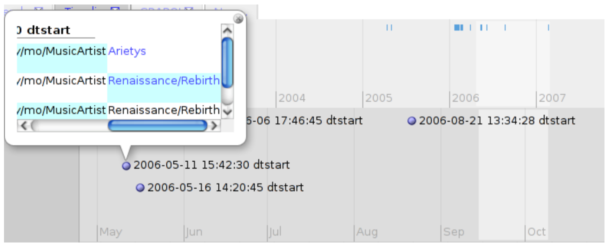
dtstart
/mo/MusicArtist Arietys
/mo/MusicArtist Renaissance/Rebirth
/mo/MusicArtist Renaissance/Rebirth
2004
2005
2006
2007
-06 17:46:45 dtstart
2006-08-21 13:34:28 dtstart
2006-05-11 15:42:30 dtstart
2006-05-16 14:20:45 dtstart
May
Jun
Jul
Aug
Sep
Oct

Fig. 4: Displaying audio items on a timeline according to their release date, using links from the items to the corresponding resources in the Musicbrainz database—a dot denotes the release time of an item

As mentioned in § 2.4, the ontologies used on the web of data evolve in an organic and interlinked way. They are referring to each other in order to provide a vocabulary covering as much ground as possible. For example, the Music Ontology is linked to FOAF, which is itself linked to the Relationship vocabulary [16]. Therefore it is possible to express, for example, that two performers are engaged to each others.

Using Semantic Web technologies as a grounding for information management in digital library management software allows us to decouple our data model from the assets themselves (the digital assets are just resources, as important as musical works, performances, conductors, geographic locations, etc.), and to allow it to constantly evolve and adapt to new needs. Several digital library projects are moving towards such data models: Fedora [17] or EASAIER [18].

Then, interlinking of digital archives is possible by making resource identifiers dereferencable and linking them together. For example, the following statement may map a performance described in one archive and a performer described in another archive:

```
@prefix ex: <http://myarchive.org/>.
@prefix mo: <http://purl.org/ontology/mo>.

ex:gg_preludes a mo:Performance;
  mo:performer <http://dbpedia.org/resource/Glenn_Gould>.
```

The archive management software can then follow this link to grab more information about the performer involved in this performance event.

Such a distributed data model for digital archives allows a lot of further possibilities. For example, in the case of an ethno-musicological digital archive, we may want to link an audio asset in one archive to a geographic location in another dataset. Therefore, we focus our work on the information which is specific to our particular dataset, and leave the burden of creating, managing and maintaining other types of information to other datasets.

4.3. Audio processing and feature extraction

Feature extraction is the process which, from an audio signal, derives lower-dimensional vectors which model some perceptually relevant characteristics of the signal. A large number of feature extraction libraries are available, such as **jMIR** [19], **MARSYAS** [20], **VAMP** [21] plugins, etc.

These toolkits could link the knowledge they derive to relevant resource identifiers available in the web, in order to make their results available for other applications. Other instances of such libraries may use these results in order to save themselves from evaluating them, or just to access features when the content itself is not available, in order to derive higher-order knowledge from them.

In order to address these issues, we are developing an ontology of audio features linked to the Event ontology [22]. We consider a feature as an attribute of an event defined on the timeline of the signal. Therefore, the concept of event here captures the way such algorithms arbitrarily classify a time region. This ontology does not aim to cover every possible feature type, but just to provide a guideline for publishing new features by subsuming the event concept.

Expressing a structural segment within this ontology can be done as follows:

```
@prefix af: <http://purl.org/ontology/af/>.
@prefix tl: <http://purl.org/NET/c4dm/timeline.owl#'>.
@prefix zgtl: <http://zitgist.com/music/timeline/>.
@prefix ex: <http://example.org/>.
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

ex:segment a af:MusicStructuralSegment;
  event:time [
    tl:startsAtDuration "PT12S"^^xsd:duration;
    tl:durationXSD "PT30S"^^xsd:duration;
    tl:onTimeLine
      zgtl:87044794-e286-47f4-aa97-87afe1c721cb;
    ];
  rdfs:label "first verse";
  .
```

For example, a library may be able to extract mixture model parameters from Mel-Frequency Cepstrum Coefficients (MFCCs). It can publish the result of such an analysis by providing an access to the following statements:

```
@prefix af: <http://purl.org/ontology/af/>.
@prefix ex: <http://example.org/>.
@prefix zgtl: <http://zitgist.com/music/timeline/>.

ex:mfcc_mv a af:MFCCMeanVariance;
  event:time [
    tl:onTimeLine
      zgtl:87044794-e286-47f4-aa97-87afe1c721cb;
    tl:startsAtDuration "PT0S";
    tl:durationXSD "PT3M20S";
    ];
  af:mean [...] ;
  af:variance [...] ;
  .
```

Then, this resource and another similar one may be used by another toolkit (which then doesn't need access to the actual content) providing a distance measure (such as a Kullback-Leibler divergence, in our example), and publishing `mo:similar_to` statements in order to relate two signals that *sound similar*.

```
@prefix mo: <http://purl.org/ontology/mo/>.
@prefix zgtl: <http://zitgist.com/music/timeline/>.
@prefix ex: <http://example.org/>.

ex:signal1
  owl:sameAs
    zgtl:69862e67-16dd-400e-9d61-f4325495bb9e.

ex:signal2
  owl:sameAs
    zgtl:87044794-e286-47f4-aa97-87afe1c721cb.

ex:signal1 mo:similar_to ex:signal2.
```

Here, we might argue about the relevance of such automatically derived statements. However, using a Named Graph approach as defined by Carroll et al. [23] to keep track of the provenance of such statements (here, http://example.org/signal1 or http://example.org/signal2), we can filter out graphs that we do not consider as relevant enough. We can then choose to trust more the similarity statements coming from a rhythm similarity model than the ones coming from such timbral features.

Such knowledge can directly be beneficial to the applications mentioned in § 4.1, in order to generate playlists of tracks that are similar, according to some criteria filtered through the origin of the similar_to statements.

Further details about how such descriptions can be dynamically computed when accessed on the Semantic Web are available in [24].

4.4. Audio editors, visualisers and sequencers

Another type of audio application that could benefit from the knowledge provided by feature extraction application are audio editors / visualisers. Indeed, the available features can provide other relevant visualisation layers than the traditional waveform one. For example, we might visualise the output of a structural segmentation on top of the audio, dividing the timeline in several segments corresponding to speech, music, chorus, verses, etc. This knowledge can also be used to enhance the editing process by enabling a structure-based navigation, as described in [25].

Some visualisers, such as the **Sonic Visualiser** [26], provide both a way to analyse the signal (through the use of **VAMP** plugins) and a way to graphically represent the output of such analysis. Therefore, such a program can both publish knowledge output by its feature extractors, and import knowledge exported by other feature extraction components.

Audio sequencing programs can also keep track of some audio production workflow, in a semi-automated way, expressed using the Music Ontology. This allows to access meaningful information about multiple takes and the small differences between them. For example, in the case of a single performance recorded by several microphones, we can create one performance resource, which is recorded through several recording resources, holding relevant information about the position of the microphone, the type of microphone being used, etc. This would be expressed as in the following code:

```
@prefix mo: <http://purl.org/ontology/mo/>.
@prefix ex: <http://example.org/>.
@prefix tl: <http://purl.org/NET/c4dm/timeline.owl#>.
@prefix event: <http://purl.org/NET/c4dm/event.owl#>.
@prefix foaf: <http://xmlns.com/foaf/0.1/>.

ex:johndoe a foaf:Person;
  foaf:name "John Doe";
  .

ex:perf a mo:Performance;
  rdfs:label "John Doe playing the drums";
  mo:performer ex:johndoe;
  event:time [
    tl:durationXSD "PT25.4S";
  ]
  mo:producesSound ex:sound;
  .

ex:overhead a mo:Recording;
  rdfs:label "Overhead microphone";
  event:place ex:place1;
  mo:usesSound ex:sound;
  .

ex:kickdrum a mo:Recording;
  rdfs:label "Kick drum microphone";
  event:place ex:place2;
  mo:usesSound ex:sound;
  .

ex:place1 rdfs:label """
  insert exact location of the
  overhead microphone here
  """.

ex:place2 rdfs:label """
  insert exact location of the
  kick drum microphone here
  """.
```

Such data can then be published by making the identifiers dereferencable, and shared across studio applications, which can themselves keep track of the processing they do on the corresponding signal resources. This can be done by linking together the resulting signal resource, the old one, and the corresponding **LADSPA** [27] plugin URI. We can also keep track of the mixing, by using a Mixing concept (subsuming Event) dealing with signals, pan and gain values, and a downmix signal. Therefore, we can keep track of the whole workflow happening in a studio environment and share it across applications, which all contribute to this knowledge, and consume it.

4.5. General architecture for Semantic Web-enabled audio applications

Here, we try to specify a general framework for enabling audio applications to consume data available on the Semantic Web and to publish the resource they create, in order to let other datasets (perhaps held by other audio applications) benefit from them. Such a framework can address a wide range of use cases, including those mentioned in the previous sections.

The first thing we want to address is RDF storage. There are many implementations of RDF stores. Among them, we can cite **Jena**, **Sesame**, **OpenLink Virtuoso**, **YARS** or the **semweb** module of **SWI-Prolog**. This component will be our RDF *cache*—it will keep track of all the information either aggregated from remote sources or produced by the application.

Then, on top of this cache, we have a second component, able to query the Semantic Web as a single graph of interconnected resources. For example, asking this component for 'give me the latitude and the longitude associated to the location of this particular band' will dereference the URI of the band, get the URI of the geographical resource, and dereference it to get access to the latitude and the longitude.

In SPARQL, such a query would look like:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX mo: <http://purl.org/ontology/mo/>
PREFIX pos: <http://www.w3.org/2003/01/geo/wgs84_pos#>
SELECT ?lat ?long
WHERE
  {
    <http://dbtune.org/jamendo/artist/5>
      foaf:based_near ?geo.
    ?geo pos:lat ?lat.
    ?geo pos:long ?long.
  }
```

Such a client, given a query, programmatically tries to answer it by dereferencing all the resources it may need to know more about. Implementations of such a client include the **Semantic Web Client Library** [28] or **SWIC** [29].

We also need a third component able to handle updates guided by the application on the RDF cache. This can be done using a language such as SPARQL-Update [30].

Then, we need a final component which will provide dereferencable URIs for all resources created locally by our audio applications. Implementations of such a mechanism include **Pubby** [31] or **UriSpace** [32].

Using these two last components, the application is able to publish new resources and their RDF representation, therefore allowing other datasets (perhaps held by other applications) to refer to them.

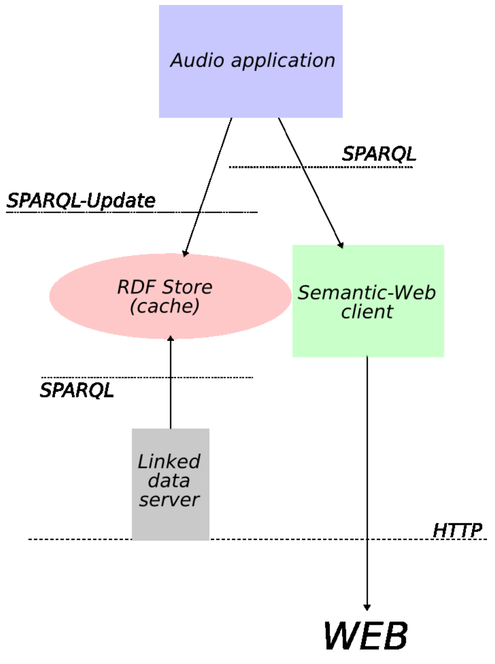

Fig. 5: A general architecture for making an audio application aware of the Semantic Web

This therefore leads us to the architecture depicted in fig. 5.

5. CONCLUSION AND FURTHER WORK

In this paper, we provided overviews of several key technologies which allow us to create a machine-processable *web of data*, holding resources coming from a wide range of datasets, links between these resources expressed within their representation, and also links towards a formal specification of the relevant concepts. We also gave an overview of the 'Music Ontology', which provides a framework for describing audio-related data in this web. Then, we explained concrete examples of how using this web can be beneficial for a wide range of applications: from digital jukeboxes to audio sequencers, along with a general software architecture built around several already available building blocks.

In such a framework, audio applications are using knowledge coming from other datasets (such as an editorial database) and are also publishing new resources (such as recorded audio, processed audio, extracted features or metadata captured during the production chain), therefore allowing other applications or datasets to benefit from them—they can build on top of this knowledge by linking new resources to it, or they can just consume it for their own purpose.

Further work includes the creation of a portable software library, embedding all the components mentioned in § 4.5, and the provision of a high-level programming interface to the Music Ontology constructs, as well as interfaces to Semantic Web search engines (to allow them to easily discover relevant resource identifiers).

6. ACKNOWLEDGEMENTS

The authors acknowledge the support of both the Centre For Digital Music and the Department of Computer Science at Queen Mary University of London for the studentship for Yves Raimond. This work has been partially supported by the EPSRC-funded ICT project OMRAS-2 (EP/E017614/1), and by the European Community 6th Framework Programme project EASAIER (IST-033902).

7. REFERENCES

[1] O. Lassila and R. Swick, "Resource description framework (RDF) model and syntax specification," 1998. [Online]. Available: citeseer.ist.psu.edu/article/lassila98resource.html

[2] T. Berners-Lee, R. Fielding, U. Irvine, and L.Masinter, "Uniform resource identifier," RFC. [Online]. Available: http://www.ietf.org/rfc/rfc2396.txt

[3] D. Beckett, "Turtle - terse rdf triple language," Working draft, 2006. [Online]. Available: http://www.dajobe.org/2004/01/turtle/

[4] Y. Raimond, S. Abdallah, M. Sandler, and F. Giasson, "The music ontology," in *Proceedings of the International Conference on Music Information Retrieval*, September 2007.

[5] J. R. Hobbs and F. Pan, "Time ontology in owl," Working draft, September 2006. [Online]. Available: http://www.w3.org/TR/owl-time/

[6] J. F. Allen, "Maintaining knowledge about temporal intervals," *Artificial Intelligence and Language Processing*, vol. 26, pp. 832–843, 1983.

[7] Y. Raimond and S. A. Abdallah, "The timeline ontology," OWL-DL ontology, 2006. [Online]. Available: http://purl.org/NET/c4dm/timeline.owl

[8] ——, "The event ontology," OWL-DL ontology, 2006. [Online]. Available: http://purl.org/NET/c4dm/event.owl

[9] I. Davis and R. Newman, "Expression of core frbr concepts in rdf," Working draft, 2005. [Online]. Available: http://vocab.org/frbr/core

[10] D. Brickley and L. Miller, "Foaf vocabulary specification," Working draft, 2005. [Online]. Available: http://xmlns.com/foaf/0.1/

[11] C. Lagoze and J. Hunter, "The ABC ontology and model," *Journal of Digital Information*, vol. 2, no. 2, 2001. [Online]. Available: http://jodi.ecs.soton.ac.uk/Articles/v02/i02/Lagoze/

[12] C. Bizer, T. Heath, D. Ayers, and Y. Raimond, "Interlinking open data on the web," in *Demonstrations Track, 4th European Semantic Web Conference, Innsbruck, Austria*, 2007. [Online]. Available: http://www.eswc2007.org/pdf/demo-pdf/LinkingOpenData.pdf

[13] "Musicdns." [Online]. Available: http://www.musicip.com/dns/index.jsp

[14] E. Prud'hommeaux and A. Seaborne, "SPARQL query language for RDF," 2005. [Online]. Available: http://www.w3.org/TR/rdf-sparql-query/

[15] "Greenstone digital library software." [Online]. Available: http://www.greenstone.org/

[16] I. Davis and E. V. Jr, "Relationship: A vocabulary for describing relationships between people," Working draft, 2005. [Online]. Available: http://vocab.org/relationship/

[17] "Fedora digital object repository system." [Online]. Available: http://www.fedora.info/

[18] "Enabling access to sound archives through enrichment and retrieval." [Online]. Available: http://www.easaier.org/

[19] "jmir." [Online]. Available: http://jmir.sourceforge.net/

[20] "Marsyas." [Online]. Available: http://marsyas.sness.net/

[21] "Vamp plugins." [Online]. Available: http://www.vamp-plugins.org/

[22] Y. Raimond, "The audio features ontology," Working draft, February 2007. [Online]. Available: http://purl.org/ontology/af/

[23] J. J. Carroll, C. Bizer, P. Hayes, and P. Stickler, "Named graphs," *Journal of Web Semantics*, 2005.

[24] Y. Raimond, C. Sutton, and M. Sandler, "A distributed data space for music-related information," in *Proceedings of the International Workshop on the Many Faces of Multimedia Semantics, colocated with ACM-Multimedia*, 2007.

[25] G. Fazekas and M. Sandler, "Intelligent editing of studio recordings with the help of automatic music structure extraction," in *Proceedings of the 122nd AES Convention*, 2007.

[26] C. Cannam, C. Landone, M. Sandler, and J. P. Bello, "The sonic visualiser: A visualisation platform for semantic descriptors from musical signals," in *Proceedings of the International Conference on Music Information Retrieval*, 2006.

[27] "Linux audio developer's simple plugin api." [Online]. Available: http://www.ladspa.org/

[28] "Semantic web client library." [Online]. Available: http://sites.wiwiss.fu-berlin.de/suhl/bizer/ng4j/semwebclient/

[29] "Swi-prolog semantic web client." [Online]. Available: http://moustaki.org/swic/

[30] A. Seaborne and G. Manjunath, "Sparql/update - a language for updating rdf graphs," Working draft, 2007. [Online]. Available: http://jena.hpl.hp.com/~afs/SPARQL-Update.html

[31] "Pubby - a linked data frontend for sparql endpoints." [Online]. Available: http://www4.wiwiss.fu-berlin.de/pubby/

[32] "Urispace." [Online]. Available: http://moustaki.org/urispace/