Available online at www.sciencedirect.com
ScienceDirect
Web Semantics: Science, Services and Agents on the World Wide Web 6 (2008) 162–169
JOURNAL OF Web Semantics
www.elsevier.com/locate/websem

# ZemPod: A semantic web approach to podcasting

Òscar Celma[a], Yves Raimond[b,*]

[a] *Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain*
[b] *Centre for Digital Music, Queen Mary, University of London, UK*

Received 21 January 2008; accepted 28 January 2008
Available online 26 March 2008

**Abstract**

In this paper we present a semantic web approach to solve some current limitations of *podcasting*. The main shortcomings of podcasts are two. The first one is that there is no formal description of the contents of a podcast session, apart from a textual description only available in HTML. The second problem is that a podcast session consists of a single audio file. Thus, it is very difficult to seek into one of the music tracks that compose a podcast.

Our proposal to cope with these problems uses traditional audio signal processing – such as speech versus music segmentation, and audio identification –, and semantic web techniques to automatically describe and decompose the audio content of a podcast session. Yet, we believe that adding semantics to the podcast to explain its content, and decomposing it into smaller and *meaningful* chunks (that permits seeking into the inner parts of the file) will ease important music information retrieval tasks, such as recommendation, filtering and discovery.

*Keywords:* Podcasting; Semantic web; Audio; Music ontology; Music information retrieval

## 1. Introduction

In recent years the typical music consumption behaviour has changed dramatically. Personal music collections have grown favoured by technological improvements in networks, storage, portability of devices and Internet services. The amount and availability of songs has de-emphasized its value—it is usually the case that users own many music files that they have only listened to once, or even never. In the context of the World Wide Web, the increasing amount of available music makes very difficult, to the user, to find music he/she would like to listen to. To overcome this problem, there are some audio search engines[1] that can fit the user's needs. Some of these search engines are nevertheless not fully exploited because their companies would have to deal with copyright infringing material.

Yet, since 2001, syndication of web content – a section of a website made available for other sites or applications to use – has become a common practice for websites. This originated with news and weblog sites, but nowadays is increasingly used to syndicate any kind of information. In this context, in mid-2004 appeared the *podcasting* phenomenon. A podcast is a media file distributed in Internet using syndication feeds. Usually, a music podcast session consists of a speaker(s) – i.e. the radio presenter(s) – and a list of music titles.

Meanwhile, the explosion in popularity of mobile devices has made the syndication model more attractive, not only for digital music downloads, but also for a wide variety of content that is now being "podcasted" to desktops and handheld devices [7].

Nowadays, thousands of audio podcasts are available on the net. Yet, from a user's point of view, it is frustrating the interaction that one can achieve with a podcast file. The main problem is that a podcast consists of a single audio file (usually in MP3 or OGG format). Thus, it is very difficult to seek into one of the music tracks that compose a podcast. On the other hand, there is no formal description of the contents of a podcast session, apart from a textual description only available in HTML.

In this paper we present a way to overcome the limitations of podcasting. These limitations are: the lack of semantics to explain the content of a podcast session, and the nonexistence of a structural decomposition of the audio file. Thus, our proposal uses traditional signal processing techniques plus semantic web

* Corresponding author.
*E-mail addresses:* oscar.celma@iua.upf.edu (Ò. Celma), yves.raimond@elec.qmul.ac.uk (Y. Raimond).

[1] To mention a few (accessed on January, 7th, 2008): http://audio.search.yahoo.com/, http://www.audiocrawler.com/, http://www.searchsounds.net and http://www.altavista.com/audio/.

1570-8268/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.websem.2008.01.003

related technologies to fully describe and decompose the audio content of a podcast session. We believe that adding semantics to the podcast to explain its content, and decomposing it into smaller and *meaningful* chunks (that allows seeking into the inner parts of the file) allows efficient music information retrieval tasks, such as personalised recommendation [5], filtering and discovery.

This paper is structured as follows. Section 2 presents the background and state of the art of the different elements – web syndication, speech/music segmentation, speech and audio recognition, and formal ontologies – that are needed to achieve our goal. Then, Section 3 presents the conceptual architecture that allows to add formal semantics to a given podcast session. After that, Section 4 presents a concrete example as a possible scenario for the architecture proposed. Finally, Section 5 draws conclusions, and gives some insights for the future and pending work.

## 2. Background

In this section we present the state of the art of the key elements that conform the proposed architecture sketched in Section 3.

### 2.1. Multimedia web syndication

The file format used to syndicate web content is XML. The XML description to use is defined in the RSS family and Atom. The RSS abbreviation is variously to refer to the following standards: Really Simple Syndication (RSS 2.0), Rich Site Summary (RSS 0.91 and 1.0) or RDF Site Summary (1.0).

Of special interest are the feeds that syndicate multimedia content. These feeds publish audiovisual information that is available on the net. An interesting example is the Media RSS (mRSS) specification,[2] lead by Yahoo! mRSS allows to syndicating multimedia files (audio, video, image) in RSS feeds. The following listing shows a partial mRSS feed that describes a podcast session (with a title "Podcast III - Rock and Funky session"). Among other things, we can see the description of the podcast, a few categories (i.e. tags) to describe the session, the author of the podcast, and the URL of the enclosed audio file.

```
<rss version ="2.0"
  (...) namespaces
<channel >
  <title >My rock feed!</title >
  <link >
http://www.ourmedia.org/user/billy2rivers/mrss
  </link >
  <description >A classic rock podcast </description >
  <language >en </language >
  <item >
  <title >Podcast III- Rock and Funky session </title >
  <link >
http://www.ourmedia.org/user/billy2rivers/mrss/item3
  </link >
  <description >Rock music with a funky
  beat and electric lead guitar riffs.
  From the Minneapolis sound to Azucarillo Kings and
  Funkadelic, this style has (...)
  </description >
  <pubDate >Mon, 13 Feb 2007 01:35:49-0500 </pubDate >
<category domain ="urn:ourmedia:term:35">Alternative
  Rock
  </category >
  <category domain ="urn:ourmedia:term:582">funk
  </category >
  <dc:creator >Billy Two Rivers </dc:creator >
  <media:content url =
"http://www.ourmedia.org/user/billy2rivers/files/podcast3.mp3"
  duration ="20:15" />
</item >
(...)
</channel >
</rss >
```

Another syndication format that appeared in 2003, is Atom.[3] The main goal of Atom is to *standardize* feeds notation and autodiscovery, due to some limitations and incompatibility versions of the RSS family. Since 2005, Atom is an IETF RFC standard.[4]

#### 2.1.1. Feeds and the semantic web

**Atom/Owl**[5] aims at capturing the semantics of the Atom syndication format. A simple example[6] of an Atom/Owl feed, based on the previous media RSS example, is:

```
@prefix:
< http://bblfish.net/work/atom-owl/2006-06-06/#>.
[] a:Feed;
:title [a:Content; :type "text /plain";
  :body "My rock feed!";];
:link [a:Link;
  :rel iana:alternate;
  :to [:src
< http://www.ourmedia.org/user/billy2rivers/atom>;]
  ];
:updated "2007-02-13T01:35:49Z"^^xsd:dateTime;
:author [a:Person;
  owl:sameAs _:author1;
  :name "Billy Two Rivers";];
:entry [a:Entry;
  :author _:author1;
  :title [a:Content; :type "text /plain";
  :body "Podcast III- Rock and Funky session";];
  :link [a:Link;
    :rel iana:alternate;
    :to [:src
< http://www.ourmedia.org/user/billy2rivers/atom/atom3>
    ];
  ];
  :updated "2007-02-13T01:35:49Z"^^xsd:dateTime;
  :summary [a:Content; :type "text /plain";
    :body "Rock music with a funky (...)";
  ];].
```

In this example, we describe a resource of type **Feed**, which has attached metadata (an author, a title, etc.). This feed holds a

[2] See http://search.yahoo.com/mrss/.

[3] See http://www.atomenabled.org/.

[4] See http://tools.ietf.org/html/rfc4287.

[5] See http://bblfish.net/work/atom-owl/2006-06-06/.

[6] We are using the Turtle notation (see http://www.dajobe.org/2004/01/turtle/) in all our RDF examples, namespaces are defined at the end of this paper.

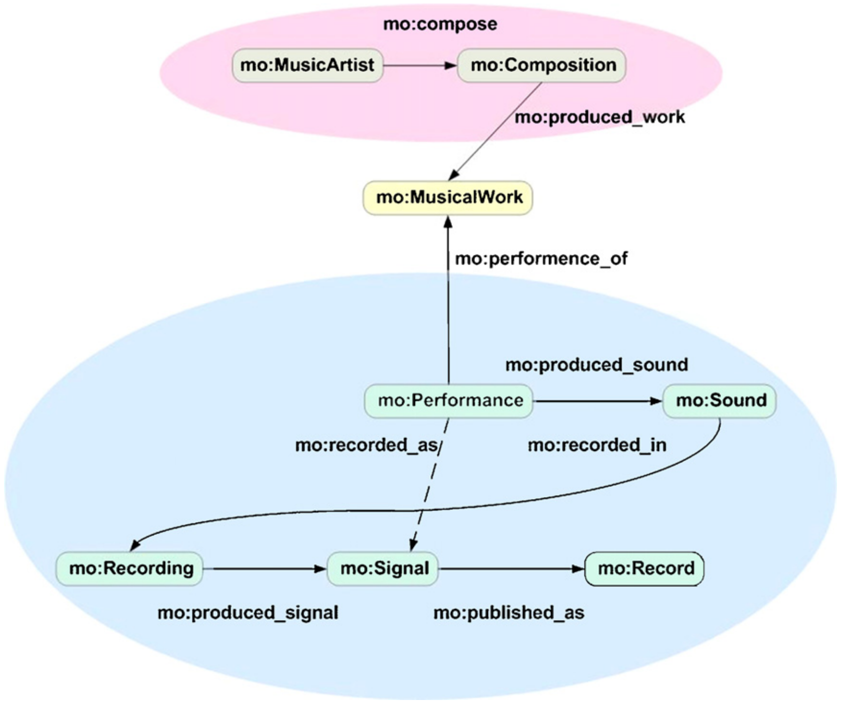

Fig. 1. Describing a music production workflow using the music ontology.

resource of type **Entry**, which itself holds a text content (“Rock music with a funky (. . .)”).

## 2.2. Speech/music segmentation

In this section we overview the existing methods to discriminate between speech (or spoken content) versus music. This part is fundamental in our system, because it is the first step towards achieving an automatically and meaningful segmentation of a podcast session.

In [10], the authors examine 13 audio features to train different multidimensional classifiers, such as Gaussian Mixture Models (GMM) and a nearest neighbour classifiers. [2] presents an evaluation of MFCC using GMM and Support Vector Machines (SVM) classifiers. [12] proposes a classification based on heuristic rules using energy, zero crossing rate, and fundamental frequency audio features. A system based on a combination of standard Hidden Markov Models and Multilayer Perceptrons is used in [1]. Finally, Pikrakis et al. [8] present a method based on the region growing technique from the image segmentation domain. They only use one single feature, a variant of the spectral entropy, computed for short-term frames. The results show up to a 93% on average discrimination, and the system shows a very fast performance.

## 2.3. Audio identification

Audio identification (or audio fingerprinting) is a technology which allows identification of unknown music. A fingerprint is a unique code that is generated for an audio file, and allows very fast retrieval from a metadata repository containing the fingerprint and basic editorial information of the track (e.g. title, artist name, album, etc.).

An audio fingerprint is, then, a unique and compact digest derived from perceptually relevant aspects of a recording [3]. The basic usages of audio fingerprints are: identification, authentication, content-based secret key generation, and content-based audio retrieval and processing.

There are different methods to generate an audio fingerprint. The fingerprint model usually receives a sequence of feature vectors computed on a frame by frame basis, plus some high-level descriptors—such as beats per minute or prominent pitch [11]. Anyway, one of the most used methods is Hidden Markov Models (HMM), because it can precisely model temporal evolution of audio signals [4].

## 2.4. The music ontology

The music ontology (MO [9]) aims to create a formal framework for describing music-related information on the semantic web. It covers complex editorial information (data related to the publication of a record, but also data describing the *workflow* pertaining to the creation of such records, as depicted in Fig. 1). The music ontology also defines a way to address segments of a single audio object. MO considers two different types of time objects: **Interval** and **Instant**, as defined in the OWL-Time ontology.[7] Moreover, these time objects are defined relatively to a *time line*. Temporal information about a particular performance or the release date of an album will refer to the physical time line, on which we can address intervals such as ‘the 1st of July, 1987’, ‘yesterday’, etc. Intervals such as ‘from 2 to 3 min, in

[7] See http://www.w3.org/TR/owl-time/.

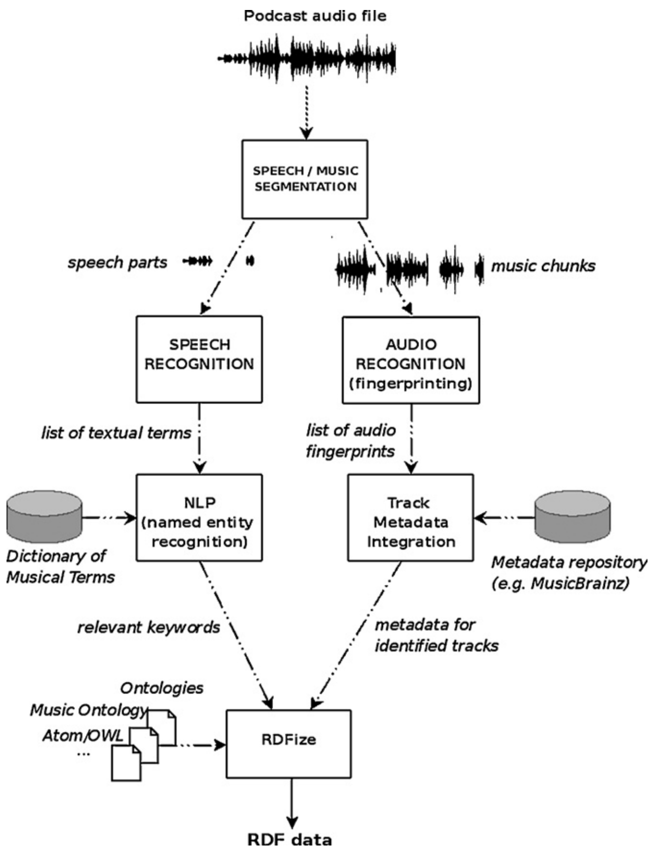
Podcast audio file
SPEECH / MUSIC SEGMENTATION
speech parts
music chunks
SPEECH RECOGNITION
AUDIO RECOGNITION (fingerprinting)
list of textual terms
list of audio fingerprints
NLP (named entity recognition)
Track Metadata Integration
Dictionary of Musical Terms
Metadata repository (e.g. MusicBrainz)
relevant keywords
metadata for identified tracks
Ontologies
Music Ontology
Atom/OWL
...
RDFize
RDF data

Fig. 2. System architecture for semantic podcasting.

this particular podcast entry', will refer to the time line backing the actual content—the audio signal. MO then defines an **Event** concept, considered as being the way by which cognitive agents classify arbitrary regions of space–time. This definition is broad enough to include descriptions such as 'this is a recording of a performance that happened at that time and at that place', 'this podcast entry starts with this track, then someone is speaking' or even 'in this segment corresponding to a track, this part holds a structural segment'.

## 3. System architecture

In this section we describe the proposed system architecture, that allows to analyse and decompose a given podcast audio file (see Fig. 2). The system segments the audio file into speech and music sections (see Section 2.2 for more details). For each detected speech part it applies speech recognition to extract a list of textual terms. Then, for each recognised term, the system weights its relevance according to a dictionary of musical terms. After this process, the system outputs a list of relevant, music related, keywords that can be used to describe that part of the audio file (however, this feature is not yet included in the system). For all the music chunks detected, the system tries to recognise them using an audio recognition algorithm, like fingerprinting[8] (see Section 2.3). This module outputs a list of unique identifiers for all the audio identified. Using these audios identifiers, the system queries a metadata repository to get basic information related with the track (e.g. song title, artist name, album, etc.). This process is depicted in Fig. 2.

In Section 3.1, we explain how to *RDFize* a podcast, as well as how to describe the parts that have been automatically segmented. In Section 3.2, we explain the *workflow* involved in our system (registering new feeds, analysing new entries and expanding their description).

### 3.1. RDFizing a podcast session

In order to get to a knowledge representation framework allowing us to describe the inner semantics of a podcast, we consider using both Atom-OWL, described in Section 2.1.1, and the music ontology, described in Section 2.4. For this purpose, we consider describing a **Content** object in Atom-OWL (the content of an entry) as a **Podcast** resource in the music ontology, which subsumes **Manifestation** in FRBR:

```
mo:Podcast rdfs:subClassOf mo:MusicalManifestation.
mo:Podcast rdfs:subClassOf awol:Content.
```

Therefore, we can describe a particular audio content in a feed by using the music ontology. By classifying regions of the corresponding **TimeLine** object as described in Section 2.4, we can describe the different parts of the corresponding signal. For example, we may express 'from 0 to 2 min and 30 s, there is someone speaking about Rock'n'Roll in the mid sixties, then there is a recording of a Sonics' gig in Tacoma in 1964'. In order to do so, we first define two sub concept of the **Event** one: **MusicSegment** and **SpeechSegment**. It allows us to classify the podcast signal into temporal regions holding speech, and into regions holding music. For the speech regions (**SpeechSegment**), we express the output of a speech-to-text component by defining **TextEvent** concept. Individuals of this concept associate a particular region of the signal to a text transcript of this region. For the music regions (**MusicSegment**), the system computes a fingerprint of the signal, and then query an identification service in order to identify that track, and link it with relevant metadata.

### 3.2. Access and workflow

To access and retrieve podcast sessions, we define a simple REST [6] interface, that allows us to access the podcast analysis service, and the associated semantic representations. From now on, we consider that our service is available at http://zempod.net/.

#### 3.2.1. Awareness of feeds

In order to make our system aware of feeds, we handle POST requests on the resource located by http://zempod.net/feed. This will send back a `201` (**Created**) HTTP code, with a **Location** header field pointing towards the identifier of the resource holding this newly submitted feed. For example, such an identifier

[8] MusicDNS, from MusicIP, offers this service, and it links with the MusicBrainz metadata repository. See: http://www.musicip.com/dns/ for more details.

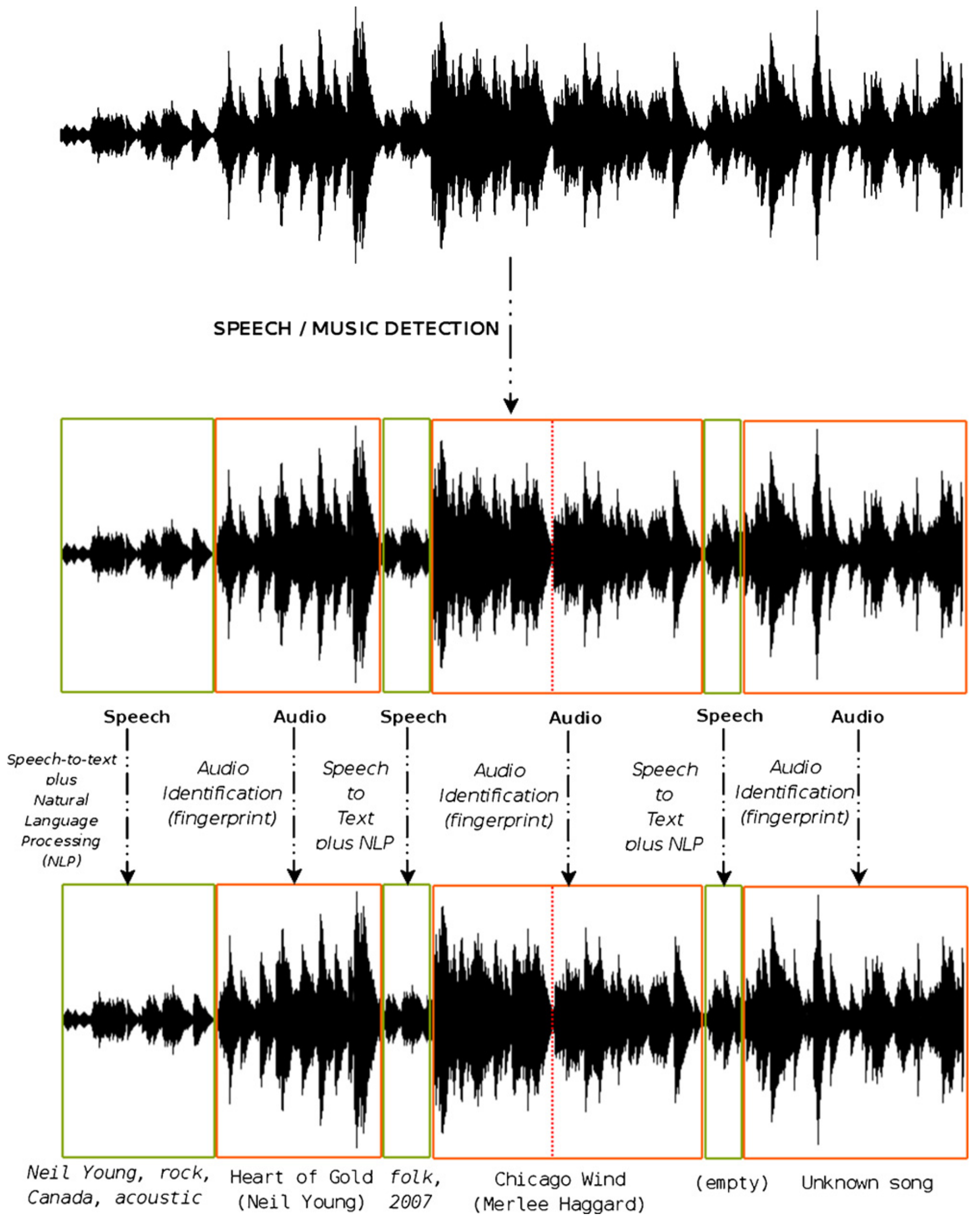
SPEECH / MUSIC DETECTION
Speech
Audio
Speech
Audio
Speech
Audio
Speech-to-text plus Natural Language Processing (NLP)
Audio Identification (fingerprint)
Speech to Text plus NLP
Audio Identification (fingerprint)
Speech to Text plus NLP
Audio Identification (fingerprint)
Neil Young, rock, Canada, acoustic
Heart of Gold (Neil Young)
folk, 2007
Chicago Wind (Merlee Haggard)
(empty)
Unknown song

Fig. 3. Example of the temporal decomposition of an audio content.

could be http://zempod.net/feed/4567. The system handles feeds in the most popular syndication formats: RSS 2.0, Atom, RSS 1.0, and mRSS.

Once registered, the internal representation of a feed within the system is using MO/AtomOWL, as described in Section 3.1, and every such RDF data is located within a single RDF store, that can be queried through SPARQL.[9]

### 3.2.2. *Resource identifiers*

The URIs of the different resources described in this MO/AtomOWL representation (**Feed**, **Entry** and **Content**) are designed as a hierarchical URI space. Feeds are identified by http://zempod.net/feed/{FEEDID}, entries are identified by http://zempod.net/feed/{FEEDID}/entry{ENTRYID}, and actual contents are identified by http://zempod.net/feed/{FEEDID}/entry{ENTRYID}/item{ITEMID}.

### 3.2.3. *Resource representations*

A GET request on http://zempod.net/feed (the resource representing the whole system) sends back a *feed of feeds*: a feed holding the newly added feeds to the system. It has the following representations depending on what the user agent is asking for (through the **Accept** field in the request header):

- **Default**: XHTML representation with embedded RDFa[10] representation (AtomOWL)

[9] See http://www.w3.org/TR/rdf-sparql-query/.

[10] See http://www.w3.org/TR/xhtml-rdfa-primer/.

- **application/rss+xml**: `303` (**see other**) HTTP redirection towards the feed in RSS 2.0
- **application/rdf+xml**: `303` towards the feed in AtomOWL (RDF/XML serialization)
- **application/x-turtle or text/rdf+n3**: 303 towards the feed in AtomOWL (Turtle serialization)
- **application/atom+xml**: `303` towards the feed in Atom

A GET request on a registered feed identifier, such as http://zempod.net/feed/4567, is accessing a representation of this feed. The same representations as mentioned above are available. The RDF representation is using MO/AtomOWL, as described in Section 3.1. The other identifiers used in the MO/AtomOWL representation (the ones corresponding to the concepts **Entry** and **Content/Stream**) also support a similar content negotiation mechanism. So far, the system just provides format conversion abilities between various syndication formats. We now consider automatically *expanding* the music ontology/AtomOWL representation using speech/music segmentation, speech-to-text techniques and track identification on new audio entries.

### *3.2.4. Processing new entries*

From now on, the system tries to analyse new entries in the registered feeds, using the processing mechanism depicted in Fig. 2. Resulting statements are linked to the MO/AtomOWL representation of the feed and put into the main RDF store. Therefore, a user agent accessing the MO/AtomOWL representation of a newly registered feed first accesses the RDF representation of the feed, just involving a conversion process between the original format and AtomOWL. When a new entry is processed, along with the RDF description of this feed, a representation of the speech/music segmentation of the podcast is available at first. The timeline of the new content is classified using **SpeechSegment** and **MusicSegment** individuals, as described in Section 3.1. The identifiers of these individuals are designed according to the pattern: http://zempod.net/feed/{FEEDID}/entry{ENTRYID}/item{ITEMID}/segment{SegmentID}. A GET request on such identifiers with a **Accept** field set to `audio/mpeg`accesses the corresponding audio segment. When the entry is fully processed, the transcribed speech and information about the embedded music tracks becomes available.

## 4. Usage scenario

In this section, we present a concrete user scenario. We assume that a feed is referenced in our system using its REST interface. Then, new entries (podcast sessions) are processed in order to extract semantic information. All this information is appended to the MO/AtomOWL representation of the feed.

### *4.1. Submission of the original feed*

We consider that a mRSS feed is available at http://www.ourmedia.org/user/billy2rivers/mrss. We POST this URL to http://zempod.net/feed, and we get back a `201` (**Created**) HTTP code, with a **Location**: header field set to http://zempod.net/feed/1234. Our system then knows about this feed, and will process new entries when they are available. We can also use the content negotiation mechanism provided by our framework in order to access either the original feed, an Atom representation of it, an AtomOWL one, or a HTML human-readable one. Now, we consider that there is a new audio content enclosed in this feed. The following RSS example shows this new feed entry (named "*Podcast IV - A Country/Folk session*"):

```
(...)
<channel >
  <title >My rock feed!</title >
<link > http://www.ourmedia.org/user/billy2rivers</link >
  <description >A classic rock podcast </description >
  <language >en </language >
  <item >
    <title >Podcast IV- A Country /Folk session </title >
    <link >http://example.com/user/me/podcast4//link >
    <description >Some more country from Canada (...)
    </description >
    <pubDate >Fri, 30 Mar 2007 01:35:49-0500 </pubDate >
    <dc:creator >Billy Two Rivers </dc:creator >
    <media:content url =
"http://www.ourmedia.org/user/billy2rivers/files/podcast4.mp3"
type ="application /octet-stream"/>
</item >
(...)
```

### *4.2. Analysis of the new entries*

The system, processes this new entry (i.e. a new podcast session), as depicted in Fig. 3. The system segments the audio content in speech/music parts. Then, it processes the speech parts by performing natural language processing on it. Meanwhile, another thread processes the music parts, by computing a fingerprint of the corresponding signals. With the identified tracks, the system tries to retrieve metadata information using the Musicbrainz web-service. All this knowledge is appended, as soon as it is computed, to the MO/AtomOWL representation of the feed (see Section 3.1). As we can see in Fig. 3 some errors may occur when trying to identify audio chunks. For instance, the second detected audio chunk contains two concatenated songs (separated by a dotted line) without any speech interruption between them. Our system should be aware of this effect, else it would identify that audio part with only one song (in this case, *"Chicago Wind"*, by *Merlee Haggard*). However, there are audio fingerprint algorithms that can track continuous playlist, that is to detect the boundaries between two concatenated songs.

### *4.3. Semantic description of the new entries*

Finally, if we do a GET request on http://zempod.net/feed/1234 with a **Accept** header field set to `application/x-turtle`, we get the following description:

```
@prefix: < http://zempod.net/feed/1234/>.
[...]
< http://zempod.net/feed/1234>
```

```
  a awol:Feed;
  awol:author < http://moustaki.org/foaf.rdf#me>;
  awol:updated "2007-02-13T01:35:49Z"^^xsd:dateTime;
  awol:entry:entry1.
:entry1
  a awol:Entry;
  dc:title "My rock feed";
  awol:author
    < http://www.ourmedia.org/user/billy2rivers/>;
  awol:content
    < http://zempod.net/feed/1234/entry1/item1>.
< http://zempod.net/feed/1234/entry1/item1>
  a mo:Podcast;
  mo:available_as
< http://www.ourmedia.org/user/billy2rivers/files/podcast4.mp3>.
# First, we describe all the anchor points we need
:signal
  a mo:Signal;
  mo:published_as
    < http://zempod.net/feed/1234/entry1/item1>;
  mo:time mo:overallti.
:overallti
  a tl:Interval;
  tl:duration "PT23M2S"; # 23 min and 2 s
  tl:onTimeLine:timeline;
.
:timeline a tl:RelativeTimeLine.
# Now, we split the timeline
# FIRST SECTION: Speech part
:part1 a tl:Interval;
  tl:beginsAt "PT0S"; # starts at 0
  tl:duration "PT15S"; # during 15 s
  tl:onTimeLine:timeline.
< http://zempod.net/feed/1234/entry1/item1/segment1>
  a zp:SpeechSegment;
  event:time:part1;
  zp:text:txt1.
:txt1
  a zp:Text;
  zp:text "Neil Young, Rock, Canada, acoustic".
# SECOND SECTION: Audio part
:part2
  a tl:Interval;
  tl:beginsAt "PT15S"; # from 15s
  tl:duration "PT3M30S"; # and lasts 3 min 30 s
  tl:onTimeLine:timeline.
< http://zempod.net/feed/1234/entry1/item1/segment2>
  a zp:MusicSegment;
  event:time:part2.
:part2signal
  a mo:Signal;
  mo:time:part2;
  mo:puid "77f5bed3-eaa3-4efc-a916-99b182488f29";
  mo:published_as [
    a mo:Track;
  owl:sameAs zgt:e832b734-62e9-4eb7-84f8-0951f39e719e;
    mo:available_as
< http://zempod.net/feed/32874/item1/part2/>;
    ];].
[. . .]
```

The previous example shows another benefit of our approach: now it is straightforward to play a podcast, and skipping sections that a user might not be interested in. Moreover, the aggregated data of all the feeds and the results of the processing of their audio entries is available through the SPARQL endpoint of our service. Therefore, we can issue queries such as 'give me all the feeds that hold the Heart of Gold song by Neil Young'. The service can also crawl the remote identifiers, such as the Musicbrainz ones for music tracks, in order to get further information about them. By using the links from the Musicbrainz dataset to the Geonames or the DBPedia one, we can gather encyclopedic information about the artists, or geographical information about their general location. Then, this information can be queried as well, through queries such as 'Give me all podcast entries issued yesterday which involves an artist based in New York, and who died in 2002'.

## 5. Conclusions

In this paper we have presented a semantic web approach to solve current limitations of *podcasting*. As we have shown, the main shortcomings of podcasts are two. The first one is that there is no formal description of the contents of a podcast session, apart from a textual description only available in HTML. The second problem is that a podcast session consists of a single audio file. Thus, it is very difficult to seek into one of the music tracks that compose a podcast.

Our proposal uses traditional audio signal processing – such as speech versus music segmentation, and audio identification –, and semantic web techniques to automatically describe and decompose the audio content of a podcast session. More in detail, we propose a method to transform the current RSS and Atom feeds to the Atom/OWL version. Based on the Atom/OWL, we are able to describe the audio content by means of a music ontology that includes temporal timeline decomposition.

Current achievements in audio signal processing show that it is possible to discriminate between speech and music with a very high precision. Moreover, audio fingerprint is already available in the market, with a few companies offering this service. Thus, it is clear that describing and decomposing a podcast session into smaller and *meaningful* chunks (that permits seeking into the inner parts of the file) will ease some important music information retrieval tasks.

Moreover, it is worth noting that nowadays there is no system that allows to a user to manually segment and describe a newly created podcast session. Such a tool it could be very easy to implement upon our system. Yet, there are hundreds of thousands of podcasts available on the net that still can benefit from our approach.

## Acknowledgements

The authors acknowledge the support of the Music Technology Group from Universitat Pompeu Fabra, and both the Centre For Digital Music and the Department of Computer Science at Queen Mary University of London for the Yves Raimond's studentship.

## References

[1] J. Ajmera, I. McCowan, H. Bourlard, Speech/music segmentation using entropy and dynamism features in a hmm classification framework, Speech Communication 40 (2003) 351–363.

[2] J.A. Arias, J. Pinquier, R. Andre-Obrecht, Evaluation of classification techniques for audio indexing, in: H. Dutagacy, B. Sankur, T. Akgul (Eds.), 13th European Conference on Signal Processing (EUSIPCO'2005), Antalya, Suvisoft Oy Ltd, September 2005.

[3] P. Cano, E. Batlle, E. Gomez, L. Gomes, M. Bonnet, Audio Fingerprinting: Concepts and Applications, Springer-Verlag, 2005, pp. 233–245.

[4] P. Cano, E. Batlle, T. Kalker, J. Haitsma, A review of audio fingerprinting, J. VLSI Signal Process. 41 (3) (2005) 271–284.

[5] O. Celma, Foafing the music: bridging the semantic gap in music recommendation, in: Proceedings of 5th International Semantic Web Conference, Athens, GA, USA, 2006, p. 40.

[6] R.T. Fielding, Architectural Styles and the Design of Network-based Software Architectures, PhD thesis, University of California, Irvine, 2000.

[7] P. Rogers, Podcasting and its role in semantic social networks, the web 2.0, and the semantic web, AIS Semantic Web Inform. Syst. (SIGSEMIS) Bull. 2 (2005) 57–61.

[8] A. Pikrakis, T. Giannakopoulos, S. Theodoridis, A computationally efficient speech/music discriminator for radio recordings, in: Proceedings of the International Conference on Music Information, October, 2006, pp. 107–110.

[9] Y. Raimond, S. Abdallah, M. Sandler, F. Giasson, The music ontology, Specification available at http://musicontology.com/, in: Proceedings of the International Conference on Music Information Retrieval, September, 2007.

[10] E. Scheirer, M. Slaney, Construction and evaluation of a robust multifeature speech/music discriminator, in: Proceedings of International Conference on Acoustics, Speech, and Signal Processing, Munich, Germany, 1997, pp. 1331–1334.

[11] J. Wheaton, T. Blum, D. Keislar, E. Wold, Method and article of manufacture for content-based analysis, storage, retrieval and segmentation of audio information, U.S. Patent 5,918,223 (June 1999).

[12] T. Zhang, C.-C. Jay Kuo, Audio content analysis for online audiovisual data segmentation and classification, in: IEEE Transactions on Speech and Audio Processing, vol. 9, No. 4, Suvisoft Oy Ltd., May 2001, pp. 441–457.