A Distributed Audio Personalization Framework over Android

Emmanouil Skondras[1], Vasileios Triantafyllou[2]

Department of Informatics, Athens University of Economics and Business
10434 Athens, Greece
e-mail: manolisskondras@yahoo.gr[1], btriantafgr@hotmail.com[2]

ABSTRACT
This paper presents an audio personalization framework for mobile devices. The multimedia models MPEG-21 and MPEG-7 are used to describe metadata information. The metadata which support personalization are stored into each device. The Web Ontology Language (OWL) language is used to produce and manipulate the relative ontological descriptions. The process is distributed according to the MapReduce framework and implemented over the Android platform. It determines a hierarchical system structure consisted of Master and Worker devices. The Master retrieves a list of audio tracks matching specific criteria using SPARQL queries.

Key words: Android, MapReduce, MPEG-7, MPEG-21, OWL, SPARQL, Personalization, Distributed architectures

INTRODUCTION

Nowadays, the volume of multimedia data is increasing rapidly. Also network infrastructures enable information repositories to be accessed from a large variety of devices. The delivery of multimedia services is a common task. However more users tend to require information retrieval services which include high quality features such as semantic description and personalization of information. In this paper a prototype application that uses SPARQL queries to deliver audio information to mobile users is described.

The personalization process is distributed according to the MapReduce framework and implemented over the Android mobile platform. It determines a hierarchical system structure consisted of Master and Worker devices. The Master interacts with the Workers and retrieves a list of audio tracks matching specific criteria using SPARQL queries. The MPEG-7 and MPEG-21 models are used for the description of audio content. The metadata information is managed using the appropriate Web Ontology Language (OWL) ontology. Each Worker contains audio resources and resource adaptation metadata, minimizing thus the central storage requirements. Each device can implement the master-side as well as the worker-side.

The remainder of the paper is organized as follows. First the related research literature as well as an overview of the standards followed in this study are presented. Then the software architecture that supports the prototype application and the relative software elements and modules are described. A case study of our work is also presented. Finally, the proposed framework is concluded and possible future extensions and plans are referred.

MATERIALS AND METHODS

Related Work

The rapid increase in multimedia content has challenged the academic and industrial communities into the development of multimedia tools enhanced with personalization and adaptation capabilities. Mobile devices (such as mobile phones, smart phones and PDAs) have become increasingly powerful. Also distributed approaches provide flexibility and parallelization in the personalization process. Consequently, some related works are presented.

In the work described in [1], the authors propose a personalization process that customizes rich multimedia documents to the needs of an individual reader. Multimedia documents, such as textbooks, reference materials and leisure materials, inherently use techniques that make them accessible for people with disabilities, who are incapable of using printed materials. The authors address issues of establishing user personalization profiles, as well as adapting and customizing content, interaction and navigation. Customization of interaction and navigation leads to different user interfaces, as well as different structural content presentation. Customization of content includes insertion of a summary, synchronization of sign language video with highlighting of text, self-voicing capability, alternative support for screen readers, or reorganization of layout to accommodate large fonts.

The work described in [2] presents the implementation of a personalized learning model in distributed learning environments based on Semantic Web technologies. The authors propose a service based architecture for establishing personalized e-Learning process. The functionality is provided by various web-services. A Personal Learning Assistant integrates personalization services and other supplementary services and provides personalized access to learning resources in an e-Learning network.

The work described in [3] examines a metadata based approach, supporting the personalization process for knowledge workers who interact with distributed information objects. An architecture supporting the personalization process is described, along with a prototype personalization environment. Its metadata are decentralized, in terms that the information is stored locally on each client. The authors discuss the advantages, as well as the challenges of the suggested approach.

Finally, [4] presents how the interest of a user in TV programs can be predicted from his zapping behavior. It also presents how a user's social network can be used to realize a distributed recommendation of TV programs. A user interface for a personalized peer to peer television system that encompasses personalized navigation to available distributed content is demonstrated.

Materials

This section makes an overview of the standards and the technologies used for the development of the application prototype. These standards include MPEG-7 [5], MPEG-21 [6], OWL [7], MapReduce [8] and SPARQL [9]. Also, the proposed framework is implemented over the Android [10] platform.

MPEG-7 is a standard developed from Moving Pictures Expert Group (MPEG) for the description of multimedia information. The standard provides a framework for the description of multimedia content encoded in any existing scheme such as MPEG1, MPEG2, and MPEG4. Metadata are stored in XML allowing efficient indexing, searching and filtering of multimedia data. MPEG-7 defines the following elements:

- Description tools, which include Descriptors (D) and Description Schemes (DS). Descriptors define the syntax and the semantics of metadata elements. Description Schemes contain Descriptions, other Description Schemes as well as relationships between them.
- A Description Definition Language (DDL), which is used for defining the syntax of Description Tools and creating new or extending existing Description Schemes.

- System tools which provide mechanisms for multiplexing descriptors and synchronizing descriptions with content.

defining an open framework for multimedia applications

The MPEG-21 standard defines a framework for effectively managing multimedia resources. MPEG-21 uses the architectural concept of the Digital Item. A Digital Item is a combination of resources (such as videos, audio tracks, images), metadata (such as descriptors, identifiers), and structures describing the relationships between resources. Digital Items are declared using the Digital Item Declaration Language (DIDL). MPEG-21 Digital Item Adaptation (DIA) architecture and the MPEG-7 Multimedia Description Schemes (MDS) for content and service personalization provide a Usage Environment which models user preferences. The Usage Environment Description of the DIA framework contains the following attributes:

- The User Characteristics, which specify user features, including:
  - The User Info, where user information is stored.
  - The User Preferences, describing the user browsing, filtering and search preferences.
  - The Usage History, where the history of user interaction with digital items is presented.
  - The Presentation Preferences, which describe user preferences concerning the means of presentation of multimedia information.
  - The Accessibility Characteristics, responsible for content adaptation concerning users with auditory or visual impairments.
- The Terminal Capabilities, which describe the technical characteristics of user devices.
- The Natural Environment Characteristics, providing information about the location and time of a user in a particular environment, as well as audio-visual characteristics which may include noise levels and illumination properties of the natural environment.
- The Network Characteristics, which specify the network characteristics parameters including bandwidth utilization, packet delay and packet loss.

The RDF Schema (RDFS) [11] provides structures for knowledge representation. It deals with the organization of ontological hierarchies such as classes, relationships and properties. However complex structures or restrictions such as the scope of properties or the cardinality of attributes cannot be supported in RDFS. The need of a more powerful ontology language leads us to the Web Ontology Language (OWL). OWL is a family of knowledge representation languages used for composing ontologies. It is considered as an extension of the RDFS and its specifications have been authorized by the World Wide Web Consortium. Ontologies are described in owl documents by defining classes, properties and individuals. Classes are collection of concepts, attributes are properties of classes and individuals represent the objects of a particular class. Three OWL sublanguages are defined (OWL Lite, OWL DL and OWL Full). OWL Lite defines class hierarchies with simple constraints. For example, it allows only 0 and 1 cardinality values. OWL Lite has lower complexity than OWL DL. OWL DL and OWL Full use the same language vocabulary. However OWL Full does not impose any syntactic restrictions. It does not require separation of classes, properties and individuals and allows RDF structures to be mixed with OWL syntax. As a result an RDF document is a valid OWL Full document while it is not a valid OWL Lite or DL document.

SPARQL is an SQL-like language developed for issuing queries to RDF and OWL repositories. Queries are expressed in triple patterns similar to RDF whereas RDF subjects, predicates and objects could be variables. Additional language features include conjunctive or disjunctive patterns as well as value filters. SPARQL components are described in three specifications. The query language specification [12] presents the SPARQL language structures. The query results XML specification [13], defines the format of the results returned from SPARQL queries as XML documents. The SARQL protocol [14] defines the framework for sending queries from clients to remote server using HTTP or SOAP messages.

MapReduce is a framework for producing and executing parallelized software. It has a lot of applications in a large variety of issues which require distributed computations. It consisted of a “Map” and a “Reduce” function. The “Map” function solves a part of the problem. Then, the “Reduce” function merges the individual results and extracts the entire result. Finally, Android is a Linux based mobile platform, which is used as operating system for cellphones, netbooks and tablets.

Methods

This section presents system’s architecture which is distributed in respect to the MapReduce framework. Audio metadata as well as the OWL ontology are created and stored locally at each device. This distribution of computational load and personalization data improves framework’s scalability.

Each device contains its audio tracks and the respective metadata using MPEG-7 in an MPEG-21 structure. The audio tracks are separated in sixteen music categories (such as pop, classical, dance, electronic etc.). Audio metadata include user defined metadata (such as artist, producer, production year and category) and technical oriented metadata (such as bitrate, sample rate, track duration, audio channels, audio format and file size). Also usage history metadata (track’s popularity in respect to all tracks and track’s popularity in its category and recommended similar tracks) are contained. Table 1 presents a sample of the audio metadata structure.

Table 1. Sample of the audio metadata structure

```
<mpeg21:DIDL xmlns:mpeg21="urn:mpeg:mpeg21:2002:02-mpeg21-NS" xmlns:mpeg7="http://www.mpeg.org/MPEG7/2000">
 <mpeg21:Container>
  <mpeg21:Item>
   <mpeg21:Descriptor>
    <mpeg21:Statement mpeg7:mimeType="text/plain">Metadata about  audio  track.</mpeg21:Statement>
     </mpeg21:Descriptor>
      <mpeg21:Component>
       <mpeg21:Resource mpeg7:mimeType="application/xml">
        <mpeg7:Mpeg7>
         <mpeg7:CreationPreferences>
          <mpeg7:Title mpeg7:preferenceValue="47"  xml:lang=“en”>SoloGuitar.mp3</mpeg7:Title>
         </mpeg7:CreationPreferences>
         <mpeg7:CreationInformation>
          <mpeg7:Creation>
           <mpeg7:Creator>
            <mpeg7:Role
            mpeg7:href=“urn:mpeg:mpeg7:cs:RoleCS:2001:AUTHOR” />
            <mpeg7:Agent xsi:type=“PersonType”>
             <mpeg7:Name>
              <mpeg7:GivenName>Nick</mpeg7:GivenName>
              <mpeg7:FamilyName>Smith</mpeg7:FamilyName>
             </mpeg7:Name>
            </mpeg7:Agent>
           </mpeg7:Creator>
           <mpeg7:Creator>
            <mpeg7:Role
            mpeg7:href=“urn:mpeg:mpeg7:cs:RoleCS:2001:Publisher”/>
            <mpeg7:Agent xsi:type=“PersonType”>
             <mpeg7:Name>
              <mpeg7:GivenName>John</mpeg7:GivenName>
              <mpeg7:FamilyName>Smith</mpeg7:FamilyName>
             </mpeg7:Name>
            </mpeg7:Agent>
           </mpeg7:Creator>
           <mpeg7:Abstract>
```

```
                <mpeg7:FreeTextAnnotation>Excellent !!!
                </mpeg7:FreeTextAnnotation>
                <mpeg7:StructuredAnnotation>
                  <mpeg7:What><mpeg7:Name>Music Track</mpeg7:Name>
                  </mpeg7:What>
                </mpeg7:StructuredAnnotation>
              </mpeg7:Abstract>
              <mpeg7:CreationCoordinates>
                <mpeg7:CreationDate>
                  <mpeg7:TimePoint>2010-05-11</mpeg7:TimePoint>
                  <mpeg7:Duration>P7D</mpeg7:Duration>
                </mpeg7:CreationDate>
              </mpeg7:CreationCoordinates>
            </mpeg7:Creation>
          </mpeg7:CreationInformation>
          <mpeg7:ClassificationPreferences>
            <mpeg7:Genre mpeg7:preferenceValue="75"
              mpeg7:href="urn:mpeg:ContentCS:1">
              <mpeg7:Name xml:lang="en">Rock</mpeg7:Name>
            </mpeg7:Genre>
          </mpeg7:ClassificationPreferences>
          <mpeg7:MediaLocator>
            <mpeg7:MediaUri>data/tracks/ SoloGuitar.mp3</mpeg7:MediaUri>
          </mpeg7:MediaLocator>
          <mpeg7:MediaTime>
            <mpeg7:MediaTimePoint>T00:00:00F100</mpeg7:MediaTimePoint>
            <mpeg7:MediaDuration>T00:13:07F100</mpeg7:MediaDuration>
          </mpeg7:MediaTime>
          <mpeg7:MediaFormat>
          <mpeg7:Content mpeg7:href="urn:mpeg:mpeg7:cs:ContentCS:2001:2">
            <mpeg7:Name xml:lang="en">audio</mpeg7:Name>
          </mpeg7:Content>
          <mpeg7:Medium
           mpeg7:href="urn:mpeg:mpeg7:cs:MediumCS:2001:2.1.1 ">
            <mpeg7:Name xml:lang="en">HD</mpeg7:Name>
          </mpeg7:Medium>
          <mpeg7:FileFormat
           mpeg7:href="urn:mpeg:mpeg7:cs:FileFormatCS:2001:3">
            <mpeg7:Name xml:lang="en">MP3</mpeg7:Name>
          </mpeg7:FileFormat>
          <mpeg7:FileSize>787082</mpeg7:FileSize>
          <mpeg7:BitRate mpeg7:minimum="N/A" mpeg7:average="8000"
           mpeg7:maximum="N/A"></mpeg7:BitRate>
          <mpeg7:AudioCoding>
            <mpeg7:Format
             mpeg7:href="urn:mpeg:mpeg7:cs:AudioCodingFormatCS:2001:1">
              <mpeg7:Name xml:lang="en">MP3</mpeg7:Name>
            </mpeg7:Format>
            <mpeg7:AudioChannels mpeg7:track="2"></mpeg7:AudioChannels>
            <mpeg7:Sample mpeg7:rate="22050" mpeg7:bitPer="0">
            </mpeg7:Sample>
          </mpeg7:AudioCoding>
         </mpeg7:MediaFormat>
        </mpeg7:Mpeg7>
       </mpeg21:Resource>
      </mpeg21:Component>
     </mpeg21:Item>
   </mpeg21:Container>
</mpeg21:DIDL>
```

A suitable OWL ontology for metadata manipulation has been created (Figure 1) and stored in each device. It describes the structure and the semantics of the audio metadata providing thus flexibility to the proposed framework.

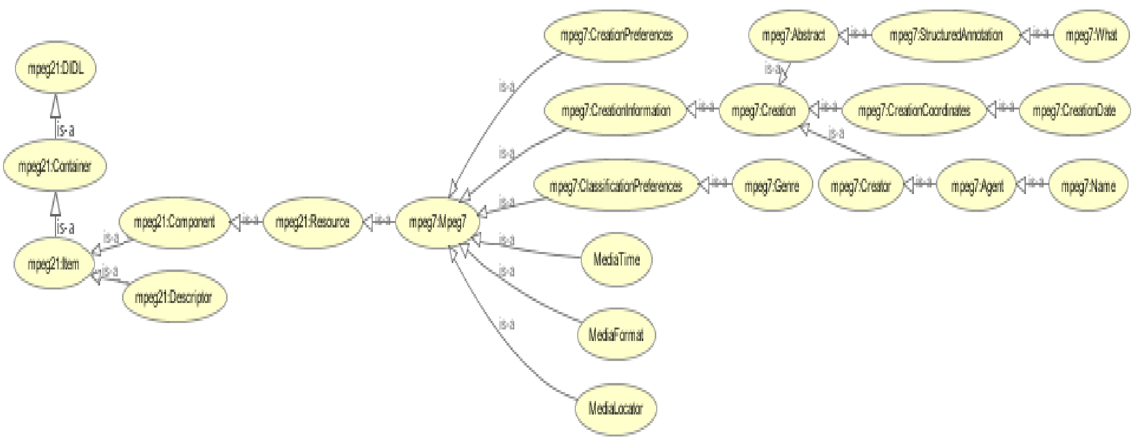

Figure 1. The OWL ontology about audio metadata

When a device requests a specific list of audio tracks using a SPARQL query, it becomes the Master and the other devices become the Workers. Each Worker receives the request, executes the Map function and extracts its relative local audio list. Then the Master receives the individual Workers' audio lists, executes the Reduce function and extracts the entire audio list. Figure 2 presents the distribution of our system.

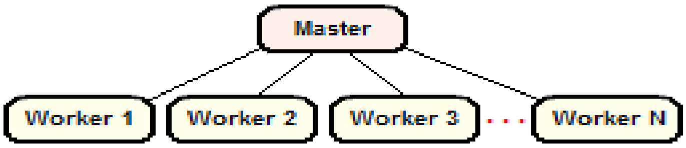

Figure 2. System's distribution

Table 2 presents the MapReduce functions' models. Map's arguments are the Master's SPARQL query and the Worker's audio metadata. Additionally it returns the relative Worker's audio list. On the other hand, Reduce's arguments are the number of the Workers and the relative audio lists. Consequently it returns the entire audio list according to the SPARQL query.

Table 2. The MapReduce functions

workerAudioList[] Map(SPARQLquery, localAudioMetadata)
Map(k1, v1) → list(k2, v2) k1 = *SPARQLquery* v1 = *localAudioMetadata* k2 = workerIdentifier*[]* v2 = *workerTop10[]*
entireAudioList[] Reduce(numberOfWorkers, workersTop10s[])
Reduce(k2, list(v2)) → list(v3) k2 = numberOfWorkers*[]* list(v2) = *workersTop10s[]* list(v3) = *summarizedTop10[]*

Figure 3 illustrates the design of Map function. The Master can also listen to an audio track that contained in a Worker. It results to an update of the Worker's usage history metadata about the relative audio track. System's modules are developed using the Java Android API [15].

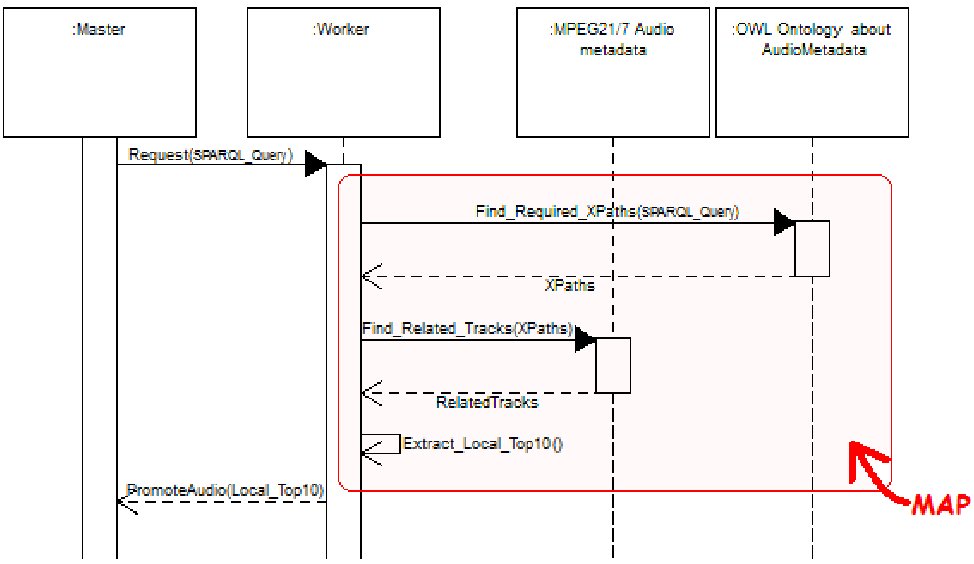

Figure 3. The Map function

\RESULT AND DISCUSSION

In this section an example of framework's functionality is presented. Three Android virtual devices are emulated. One emulator instance runs as Master and the other two as Workers. The Master composes a SPARQL query and sends the relative request. Table 3 presents an example SPARQL query. It retrieves a catalog consisted of the audio files contained to the Workers according to the arguments of the "FILTER" statement, in respect of the methods presented in the previous section. The results are ordered in descending sequence according to their popularity.

Table 3. An example SPARQL query

```
PREFIX mpeg7: < http://www.mpeg.org/MPEG7/2000>
SELECT ?Title
WHERE { ?x mpeg7:title ?title .
 FILTER (?Genre=Pop || ?Genre=Rock
          && ?Publisher=PublisherName
          && ?CreationDate>=2004-01-01
          && ?MediaDuration>30
          && ?Format=MP3  || ?Format=WAV
          && ?FileSize<20000
          && ?BitRate>=44000)
}ORDER BY DESC(?preferenceValue)
```

Each Worker receives the Master's request, runs the Map function and returns its relative local audio top-10. The metadata blocks that contribute in this step are presented in Figure 4.

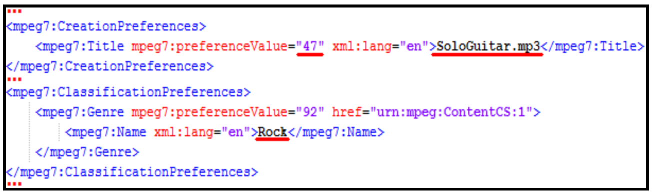

Figure 4. The relative metadata blocks

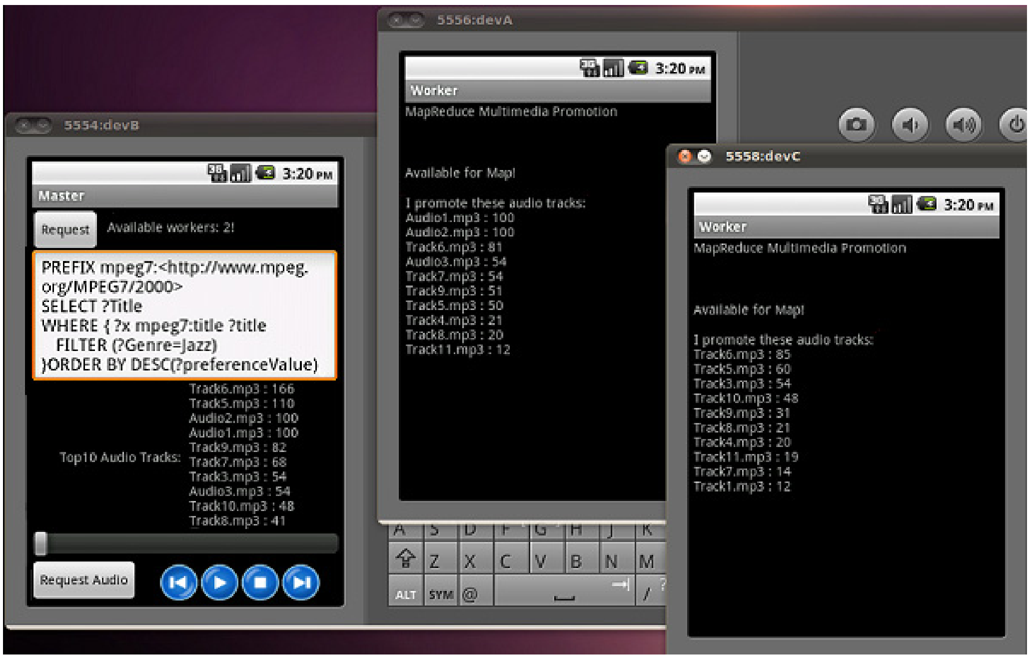

Figure 5. SPARQL query execution and entire top-10 extraction

Consequently, the Master receives the Workers' top-10s, runs the Reduce function and extracts the entire top-10. Master's user interface appears a list with the entire top-10 as well as the appropriate components for requesting and listening to audio files. Then the user selects an audio track from the entire top-10 and retrieves it from the relative Worker. Figure 5 presents a SPARQL execution example as well as the relative top-10's extraction.

CONCLUSION

The proposed framework is implemented over the Android platform. It relies on MPEG-21 and MPEG-7 standards to achieve personalization. MPEG-21 DIDL and DIA are used handling Digital Items declaration and user preferences, respectively. Moreover, the appropriate OWL ontology is used for managing the metadata. MapReduce is applied, enhancing the distributed architecture. Each device organizes its own metadata locally, decreasing thus the network's load.

Future work contains the implementation of an energy aware protocol for query routing. Also the proposed framework will be extended with the MPEG Query Format (MPQF), enabling the capability of more advanced queries' execution.

References

[1] H.L. Petrie, G. Weber, W. Fisher, "Personalization, interaction, and navigation in rich multimedia documents for print-disabled users", IBM Systems Journal, Vol 44, No 3, pp. 629-635, IBM, 2005

[2] Peter Dolog, Nicola Henze, Wolfgang Nejdl, Michael Sintek, "Personalization in Distributed eLearning Environments", Thirteen International World Wide Web Conference, pp.170-179, ACM Press, New York, 2004

[3] David L. Hicks, Klaus Tochtermann, "Personalizing Information Spaces: A Metadata Based Approach", International Conference on Dublin Core and Metadata Applications, pp.213-220, Tokyo, Japan, October 24-26, 2001

[4] Jun Wang, Johan Pouwelse, Jenneke Fokker, Arjen P. de Vries and Marcel J. T. Reinders, "Personalization on a peer-to-peer television system", Special Issue on Personalized and Mobile Digital TV Applications, Vol. 36, Num. 1-2, pp.89-113, Springer, January, 2008

[5] MPEG-7, ISO/IEC JTC1/SC29/WG11 Coding of Moving Pictures and Audio, http://mpeg.chiariglione.org/standards/mpeg-7/mpeg-7.htm

[6] MPEG-21, ISO/IEC JTC1/SC29/WG11 Coding of Moving Pictures and Audio, http://mpeg.chiariglione.org/standards/mpeg-21/mpeg-21.htm

[7] Web Ontology Language, World Wide Web Consortium, http:// www.w3.org/TR/owl-features/

[8] Jeffrey Dean, Sanjay Chemawat, "MapReduce: Simplified Data Processing on Large Clusters", OSDI'04: Sixth Symposium on Operating System Design and Implementation, San Francisco, CA, December, 2004

[9] Jun Zhai, Kaitao Zhou, "Semantic Retrieval for Sports Information Based on Ontology and SPARQL", ISME 2010: International Conference of Information Science and Management Engineering, Vol. 2, pp. 395-398, Beijing, China, August 7-8, 2010

[10] Android, http://www.android.com/

[11] RDF Vocabulary Description Language 1.0: RDF Schema, W3C Recommendation, 10 February 2004, http://www.w3.org/TR/rdf-schema/

[12] SPARQL Query Language for RDF, W3C Recommendation, 15 January 2008, http://www.w3.org/TR/rdf-sparql-query/

[13] SPARQL Query Results XML Format, W3C Recommendation, 15 January 2008, http://www.w3.org/TR/rdf-sparql-XMLres/

[14] SPARQL Protocol for RDF, W3C Working Draft, 14 January 2005, ftp://www.w3.org/TR/2005/WD-rdf-sparql-protocol-20050114/

[15] Java Android API: http://developer.android.com/reference/packages.html