AN ONTOLOGY-BASED MULTIMEDIA INFORMATION MANAGEMENT SYSTEM

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

HİLAL TARAKÇI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

SEPTEMBER 2008

Approval of the thesis:

AN ONTOLOGICAL MULTIMEDIA INFORMATION MANAGEMENT SYSTEM

submitted by **HİLAL TARAKÇI** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Volkan Atalay _____
Head of Department, **Computer Engineering**

Assoc. Prof. Dr. Nihan Kesim Çiçekli _____
Supervisor, **Computer Engineering Dept., METU**

Examining Committee Members:

Prof. Dr. Özgür Ulusoy _____
Computer Engineering Dept., BİLKENT

Assoc. Prof. Dr. Nihan Kesim Çiçekli _____
Computer Engineering Dept., METU

Assoc. Prof. Dr. Ferda Nur Alpaslan _____
Computer Engineering Dept., METU

Assoc. Prof. Dr. Ahmet Coşar _____
Computer Engineering Dept., METU

Assist. Prof. Dr. Pınar Şenkul _____
Computer Engineering Dept., METU

Date: 03.09.2008

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : Hilal Tarakçı

Signature :

ABSTRACT

AN ONTOLOGY-BASED MULTIMEDIA INFORMATION MANAGEMENT SYSTEM

Tarakçı, Hilal

M.Sc., Department of Computer Engineering

Supervisor: Assoc. Prof. Dr. Nihan Kesim Çiçekli

September 2008, 77 pages

In order to manage the content of multimedia data, the content must be annotated. Although any user-defined annotation is acceptable, it is preferable if systems agree on the same annotation format. MPEG-7 is a widely accepted standard for multimedia content annotation. However, in MPEG-7, semantically identical metadata can be represented in multiple ways due to lack of precise semantics in its XML-based syntax. Unfortunately this prevents metadata interoperability. To overcome this problem, MPEG-7 standard is translated into an ontology. In this thesis, MPEG-7 ontology is used on top and the given user-defined ontologies are attached to the MPEG-7 ontology via a user friendly interface, thus building MPEG-7 based ontologies automatically. Our proposed system is an ontology-based multimedia information management framework due to its modular architecture, ease of integrating with domain specific ontologies naturally and

automatic harmonization of MPEG-7 ontology and domain-specific ontologies. Integration with domain specific ontologies is carried out by enabling import of domain ontologies via a user-friendly interface which makes the system independent of application domains.

Keywords: Semantic Querying of Video Content, Multimedia Content Annotation, MPEG-7, MPEG-7 Ontology, MPEG-7 Based Ontology

ÖZ

ONTOLOJİ-TABANLI MULTİMEDYA BİLGİ YÖNETİM SİSTEMİ

Tarakçı, Hilal

Yüksek Lisans, Bilgisayar Mühendisliği Bölümü

Tez Yöneticisi: Doç. Dr. Nihan Kesim Çiçekli

Eylül 2008, 77 sayfa

Çokluortam verisini yönetebilmek için, içerik yardımcı verilerle etiketlenmelidir. Kullanıcı tarafından tanımlanmış etiketler kabul edilebilir olsa da, birçok sistemin aynı etiket formatını kullanması daha iyidir. MPEG-7, çokluortam içerik etiketlenmesi için yaygın kabul görmüş bir standarttır. Bununla beraber, XML-tabanlı dil ile kesin anlambilimsel bilgiyi belirtmek mümkün olmadığından, anlamsal olarak aynı olan yardımcı veri birçok farklı yol ile ifade edilebilir. Bu durum, yardımcı veri değiş-dokuşunu engeller. Buna bir çözüm olarak, MPEG-7 standardı bir ontolojiye dönüştürülmüştür. Bu çalışmada, MPEG-7 ontolojisi en üstte durmakta ve verilen kullanıcı tanımlı ontolojiler bir arayüz ile MPEG-7 ontolojisine entegre edilmektedir. Böylece MPEG-7 tabanlı ontoloji otomatik üretilmiş olmaktadır. Sunulan sistem, gerek modüler mimarisi, gerekse kullanıcı tanımlı ontolojiler ile doğal olarak entegre olması ve MPEG-7 ontolojisi ile alana özel ontolojileri otomatik olarak harmanlaması bakımından, bir ontoloji-tabanlı çokluortam bilgi yönetim çatısıdır. Alan ontolojilerinin entegrasyonu kullanımı

kolay bir arayüz vasıtasıyla Alana özel ontolojilerin import edilmesi ile gerçekleştirilir. Bu durum sistemi uygulama alanlarından bağımsız hale getirir.

Anahtar Sözcükler : Video İçeriğine Göre Anlamsal Arama, Multimedya İçerik Etiketleme, MPEG-7, MPEG-7 Ontolojisi, MPEG-7 Tabanlı Ontoloji

To my family...

ACKNOWLEDGMENTS

I would like to present my special thanks to my supervisor Assoc. Prof. Dr. Nihan Kesim Çiçekli for her guidance, understanding and encouragement throughout the development of this thesis.

I would also like to thank to IS Lab for supplying me the soccer ontology which I used in my case study and for guiding me when I needed .

I would also like to thank Tübitak, since this thesis is a part of the project with reference number 107E234 supported by Tübitak.

Finally, my deepest thanks are to my family who supported me with their never ending love , understanding and patience throughout this study.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

API	: Application Programmer's Interface
GUI	: Graphical User Interface

CHAPTER 1

INTRODUCTION

1.1 Overview

Multimedia content is overtaking the traditional media in our daily life, thus being used in a wide number of domains ranging from commerce, security, education and entertainment. The recent advances in World Wide Web, as well as the availability of cheap electronic devices for consumption of multimedia content, also makes a way for the wide spreading of multimedia content disposal by obtaining a sharing mechanism which provokes the exponential increase in the amount of multimedia data, thus leading to difficulty in the content based retrieval of the audio-visual content. Not only the enormous amount of multimedia content, but also its complexity makes annotation and semantic querying in multimedia databases a crucial problem.

Content based retrieval is a governing factor for a multimedia data management system to be successful, due to its primary importance in the interaction with the end users [7]. For instance, a sports fan may desire to search a specific sports event, such as the matches where a certain player scores, amongst football match videos in a sports videos library. Besides, he may also desire to retrieve video segments where a certain player scores against a certain team and the goalkeeper of the opposing team is another certain player. As the detail and constraints of the query

increases, the management system is expected to possess more advanced retrieval capabilities on the multimedia content.

Semantic content is defined according to the application and the multimedia data type (i.e. audio, image, video). In this thesis, video data is the center of study, since it is the most commonly-used data type and complex enough to show the impact of the thesis. Video data consists of high level features that include semantic information and low level features that include color, shape, texture etc. In this study, only semantic content that is classified as high level features is of concern and low level features are remained out of scope.

In order to satisfy the required advanced retrieval capabilities on video content, semantic descriptions for the audio-visual content should be stored in semantic descriptors and semantic-based search capabilities on the content should be provided. Semantic descriptions include metadata for the content of audio-visual information referring to the whole multimedia content or to segments of the whole content. For interoperability issues between similar audio-visual content based retrieval systems, conforming to widely-accepted international standards for the description of semantic metadata has primary importance [7], [2], [3].

MPEG-7 is one of the widespread standards used for describing content of audio-visual information by providing a rich set of standardized tools to describe multimedia content [30]. MPEG-7 provides a comprehensive set of audio-visual Description Tools that consist of the metadata elements, element structure and relationships between them. The Description Tools are designated by the MPEG-7 standard in the form of Descriptors and Description Schemes which are complex data types used to describe audio-visual content, for the purpose of creating descriptions of audio-visual content, thus forming the basis for semantic content retrieval systems to gain efficient access to audio-visual content [23].

The aim of MPEG-7 is to provide standardized tools for describing audio-visual content from different points of view at different levels of abstraction. XML Schema structure of MPEG-7 enables syntactic interoperability between similar

content based retrieval systems and over the web. However the lack of precise semantics in its XML-based syntax prevents semantic interoperability [31].

In order to overcome the semantic interoperability issues, XML Schema structure should be replaced with a more powerful technology in terms of semantics. At this point, semantic web comes in view as a solution by processing the content of information more effectively in terms of machine understandability. Semantic web which is supported by World Wide Web Consortium (W3C) provides a common framework for the purpose of sharing and reusing data across applications [32]. Semantic Web technologies include Resource Description Framework RDF [33], RDF Schema [34] and Web Ontology Language OWL [35] which is classified as OWL-Lite, OWL-DL and OWL-Full according to its expressiveness. The least expressive one is OWL-Lite, whereas OWL-Full has the greatest expressiveness. Amongst RDF, RDFS and OWL, the strongest technology is OWL in terms of representation of semantics.

In order to provide a solution to semantic interoperability issues, semantic web technologies are applied to MPEG-7 standard by translating the standard schema into an ontology represented by OWL [5], [17][16][19]. As a result of studies in this area, four OWL/RDF proposals of MPEG-7 emerged. Amongst them are Jane Hunter's MPEG-7/ABC ontology [17], Tsinaraki's MPEG-7/Tsinaraki ontology [5], Garcia and Celma's Rhizomik model [16] and Arndt's COMM [19]. All four ontologies have different amount of coverage of the standard and different expressiveness power. In this thesis, Rhizomik model is used as MPEG–7 ontology because of its complete coverage of the standard and reasonable expressiveness power. In our study, an ontology-driven approach is adopted in order to enhance content based retrieval capabilities of the system. To accomplish this, the modeling of metadata for describing audio-visual content is based on domain specific ontologies. Using domain specific ontologies dramatically improves the querying capability of content based retrieval in an open environment by guiding the user to provide legal content descriptions for domain specific audio-visual information [7].

Efforts have been undertaken to enable integration of MPEG-7 ontology with other domain specific ontologies, thus enhancing interoperability [31], [17]. Two main such approaches are proposed by Hunter et. al. and Tsinaraki et. al. The first approach which is developed by Jane Hunter uses ABC core ontology that provides attachment points for integrating MPEG–7 ontology and other domain specific ontologies. The second approach is developed by Tsinaraki and consists of using MPEG-7 ontology as the core ontology and attaching other domain ontologies to MPEG-7. In this thesis, Tsinaraki's approach for MPEG-7 and domain ontology integration is adopted. Rhizomik's MPEG-7 ontology is used as the core MPEG-7 ontology and an intermediate ontology that resides between MPEG-7 and other domain ontologies is defined in order to attach the incoming user-defined ontologies to the upper MPEG-7 ontology as desired, via a graphical user interface.

In this thesis, the aim is to develop an ontological multimedia information management framework that enables efficient annotation and retrieval of semantic content, with a modular architecture, ease of integration with user defined ontologies naturally, and automatic harmonization of MPEG-7 ontology and domain-specific ontologies via a user friendly interface. The system is fed with domain ontologies and annotations and querying is carried out in an ontology-driven way by using generic user friendly interfaces which are independent of the domain of concern. The system uses MPEG-7 ontology on top of domain ontologies in order to be interoperable with other MPEG-7 based multimedia data management systems. However, the system hides the details of the content description standard used from outside by using a layered architecture. Soccer and animal domains are selected as case study for this thesis. The reason for using two domains is to illustrate the domain-independence of the system. The ontologies for the domains are not developed during the thesis work, but provided from existing studies in the literature. The thesis does not include automated or semi-automated annotation but enables integration of any such module.

1.2 Contributions of the Thesis

The main contribution of this thesis is to apply semantic web technologies to a multimedia data management system naturally without any internal standard conversions or translations. The other contributions of the thesis can be summarized as follows:

- In this thesis, the developed multimedia data management system is a web application that uses the most recent client side technologies such as facelets and hot components such as richfaces and myfaces components. One of the benefits of using JSF technology on client side is the fact that it naturally forces the usage of Model View Controller pattern which separates the server and client side (model and view) in implementation, thus making the source code more maintainable and reusable.

- Since Web Ontology Language OWL is used instead of XML Schema structure, the semantic interoperability problem has been overcome to some extent.

- The developed system uses the standards of Semantic Web so that annotation and querying could be done in terms of concepts and attributes of concepts as well.

- The system is designed as a framework that can immediately be fed with domain specific ontologies. In other words, the system is not specific to any special domain.

1.3 Organization

The organization of the rest of the thesis is as follows: In Chapter 2, the existent multimedia data management frameworks in literature that are similar to the

proposed system are introduced as related work. In Chapter 3, MPEG-7 basics are explained in detail. The design and implementation of the ontological multimedia data management system is presented in Chapter 4. A case study including soccer and animal domains is given in Chapter 5. Finally, Chapter 6 concludes the thesis and points out possible future extensions.

CHAPTER 2

RELATED WORK

The metadata of the audio-visual content may be defined on different levels of abstraction. On the lowest syntactical level, there are basic visual features of content like shape, size, texture, color and movement of a camera or an object in a scene. On a higher level, these physical features are interpreted to derive semantic information which includes taxonomies (e.g. genre), organizational information (e.g. scenes for supporting indexing) and basic descriptions (e.g. identification of objects involved in a scene, roles, etc.). Another type of semantic information is the description of the content as annotations in natural language. As the abstraction level of the metadata increases, the management and querying power of the system enhances. There exist a number of projects that have been developed for the management of audio-visual information with respect to its content with different abstraction levels. In this chapter, three multimedia data management systems with different abstraction levels are introduced.

2.1 AceMedia

AceMedia [37] aims to combine advances in knowledge, semantics and multimedia processing technologies in order to form a framework which supports self-analyzing, self-annotating and self-adapting content of audiovisual information. In

order to accomplish these objectives in the aceMedia, the concept of the Autonomous Content Entity abbreviated as ACE was created.

Autonomous Content Entity (ACE) which is the central concept of the aceMedia Integrated Project, consists of three layers: the digital content itself that is scalable allowing personalized content adaptation, the associated metadata which is made of manual and automatically extracted semantic annotations, and an intelligence layer which can adopt autonomous actions on behalf of users.

The ontology infrastructure of aceMedia is illustrated in Figure 1 [37] . Due to the diversity of the application domains, the developed ontology infrastructure is quite complex consisting of numerous separate ontologies that have interactions with each other and each of which modeling a different aspect of the whole system.

Three main components that are shaded as gray in Figure 1, compose the ontology infrastructure: core ontology, multimedia ontologies and domain ontologies. The core ontology is an abstract ontology that models basic concepts such as objects and events. DOLCE which was designed as a core ontology is used as core ontology in the infrastructure. DOLCE is a minimal ontology that only includes reusable and upper level concepts. The multimedia ontologies describe multimedia content from different points of view. The domain ontologies model the content layer of multimedia content for specific domains. All domain ontologies are connected to high level concepts of the DOLCE core ontology in order to support interoperability between different domains.

This ontology infrastructure enables describing the multimedia content in detail since not only the concepts but also the relationships between them are described. Thus, aceMedia can handle more complex search queries.

Inside the aceMedia project an annotation tool named as M-OntoMat-Annotizer (M stands for Multimedia) [29] is developed as an extension of the CREAM (CREAting Metadata for the Semantic Web) framework. M-Onto Mat-Annotizer includes ontologies for describing low-level features of audio-visual content and also links these low level descriptions to domain specific ontology concepts .

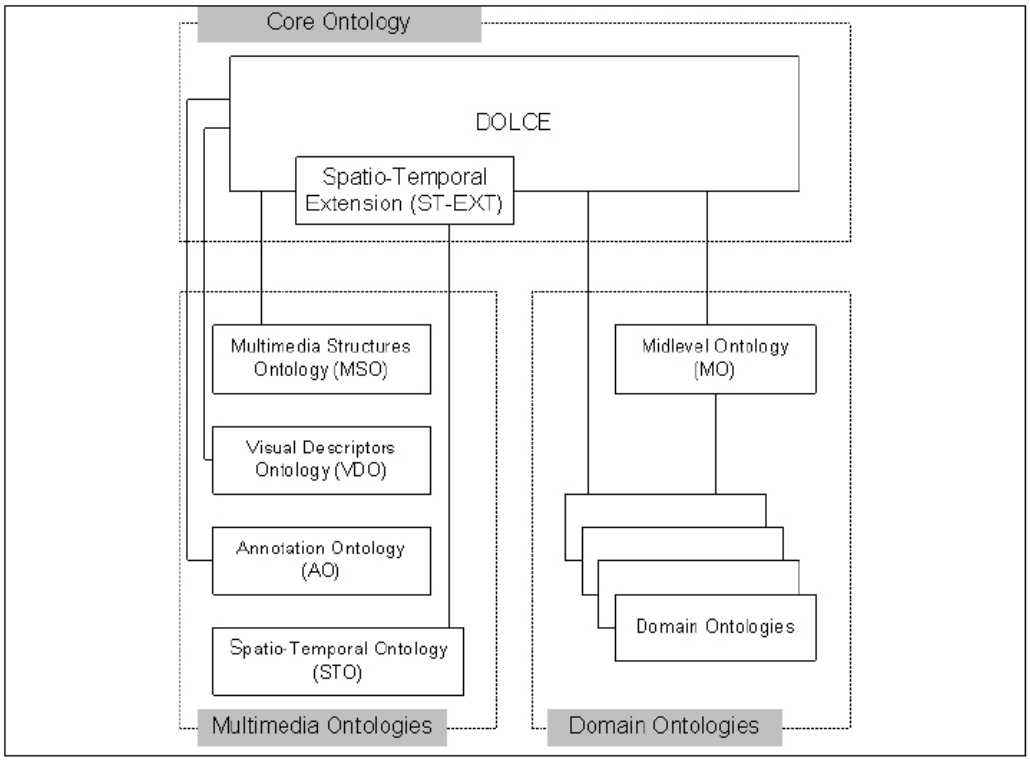

Figure 1 AceMedia Ontology Infrastructure

This thesis differs from aceMedia in terms of the following issues:

- AceMedia encapsulates built in domain ontologies that come bundled with the system. However, in this thesis, the user is allowed and encouraged to import his/her defined domain ontology supporting a more flexible system.

- The AceMedia ontology infrastructure is a triple attaching multimedia and domain ontologies to DOLCE core ontology. Nevertheless, in this study, MPEG-7 ontology itself is used as core/upper ontology and other domain specific ontologies are automatically attached to MPEG-7 via a user friendly interface.

- M-Onto Mat-Annotizer describes low-level features of audio-visual content and links these to domain specific ontology concepts. In this thesis, low level concepts are remained out of scope.

- AceMedia aims to self-analyze, self-annotate and self-adapt content of audiovisual information. However, this thesis does not include automatization or semi automatization of annotation, but enables the integration of any such module.

2.2 BilVideo

BilVideo [36] is a web-based video database management system, providing an integrated support for queries on spatio-temporal, semantic and low-level features (color, shape, and texture) on audio-visual content [28][24][26][27]. The system manages spatio-temporal queries by using a knowledge base consisting of a fact base and set of rules. On the other hand, semantic and low level feature queries are handled by using object relational database. Queries that are combination of spatio-temporal, semantic and low-level features are managed by the query processor by communicating with the knowledge base and object-relational database simultaneously. The system components returns the results for their related part of the query and the intermediate results are combined by the query processor which sends the response to Web clients afterwards [25] .

BilVideo has a mechanism of external predicates which results in easy tailoring of the system according to specific domains making the system application independent.

BilVideo v2.0, which is the following project of the same group that develop BilVideo, is an MPEG-7 compliant, prototype video database management system which supports spatio-temporal queries that contain any combination of spatial,

temporal, object-appearance, trajectory projection, and similarity-based object-trajectory, and low-level feature queries [36].

The architecture of BilVideo v2.0 is illustrated in Figure 2. Visual Query Interface module is available online as a Java applet. The Query Processor module processes the queries by interacting with XML-Native Feature Database with MPEG-7 features. Video Processor component is responsible for the preprocessing videos by performing important object extraction and annotation and sends the output to Feature Extractor component for extraction of MPEG-7 features and construction of MPEG-7 representation for each video.

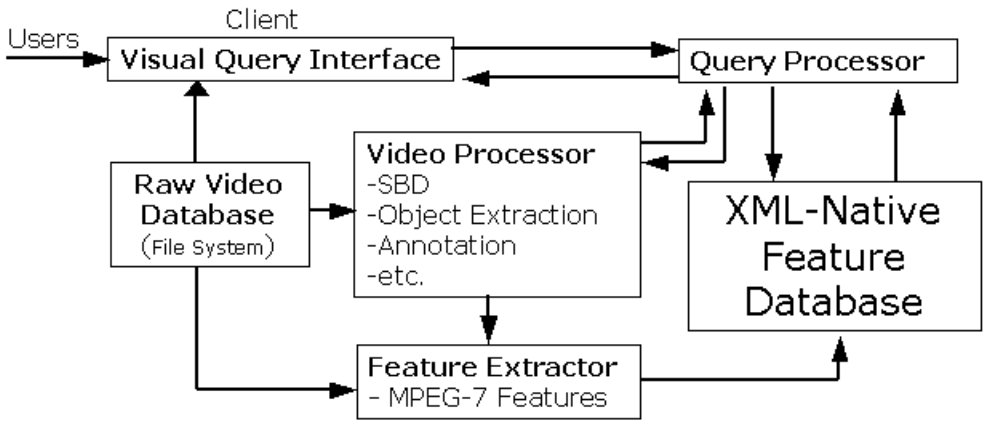

Figure 2 BilVideo v2.0 System Architecture

A comparison between BilVideo and the proposed system in this thesis include the following:

- Both BilVideo and this thesis provide a web based content retrieval system. However, BilVideo accomplishes this by providing an applet where our system uses web technologies.

- BilVideo offers semi automatic annotation of semantic content through Video Annotator and Object Extractor. On the other hand, this thesis only allows integration of such module, since automatization of annotation is remained out of scope.

- The proposed system supports integration of domain ontologies and integrates the incoming domain ontology with the upper MPEG-7 ontology automatically and provides an ontology-driven annotation and querying on semantic content of multimedia data unlike BilVideo.

2.3 Multimedia Ontologies Annotator

Another research activity focuses on creating a framework for the automatic annotation of videos in soccer domain and the semantic retrieval of soccer videos based on high-level concepts. The Multimedia Ontologies Annotator is the framework that enables import of basic ontology schemas, generation of the multimedia ontology and annotation of videos according to the given ontology. Besides, the system performs complex queries in order to retrieve videos containing specific visual concepts and high-level linguistic concepts.

This thesis is similar to the Multimedia Ontologies Annotator in terms of enabling annotation of semantic content according to the given ontology. Nevertheless, this thesis does not depend on any specific domain whereas Multimedia Ontologies Annotator is developed for soccer domain only.

CHAPTER 3

THE MPEG-7 STANDARD AND SEMANTIC WEB

The formalism in representation of video metadata has primary importance in order to support the interoperability between existing multimedia data management systems at least in syntactic level. MPEG-7 which is a widely accepted multimedia content description standard provides the required formalism in syntactical level. In this chapter, the MPEG-7 standard originally consists of XML Schemas and the efforts that are undertaken to move the standard to semantic web is presented.

3.1 The MPEG-7 Standard

MPEG-7, formally named "Multimedia Content Description Interface", is an ISO/IEC standard developed by MPEG (Moving Picture Experts Group) in order to provide a rich set of standardized tools to describe multimedia content for processing of audio-visual information by both human users and automatic systems [30].

The goal of the MPEG-7 standard is to allow interoperable searching, indexing, filtering, and access of audiovisual content by enabling interoperability among devices and applications that deal with audiovisual content description. MPEG-7 specifies the description of features related to the audiovisual content as well as information related to the management of audiovisual content. The scope of the

standard is to define the representation of the description, that is, the syntax and the semantics of the structures used to create MPEG-7 descriptions. The MPEG-7 does this by attaching complex semantics to the content. [23]

The main elements of MPEG-7 include the following:

- *Description Tools:*
    - *Descriptors (D):* Descriptors define the syntax and the semantics of metadata elements, thus is used to describe audiovisual content. A Descriptor deals with low-level features, which represent the signal characteristics, such as color, texture, shape, motion, audio energy or audio spectrum as well as high-level features such as the title or the author.
    - *Description Schemes (DS):* Description Schemes specify the structure and semantics of the relationships between the components which are Descriptors or Description Schemes.

- *Description Definition Language (DDL):* DDL is an extension of the XML Schema language. The DDL is used not only to define the syntax of MPEG-7 Descriptors but also to allow developers to declare the syntax of new Descriptors that are related to specific needs of their application.

    - *System tools:* System tools support binary coded representation for effective storage and transmission.

The main elements and the relationships between components are illustrated in Figure 3 [30].

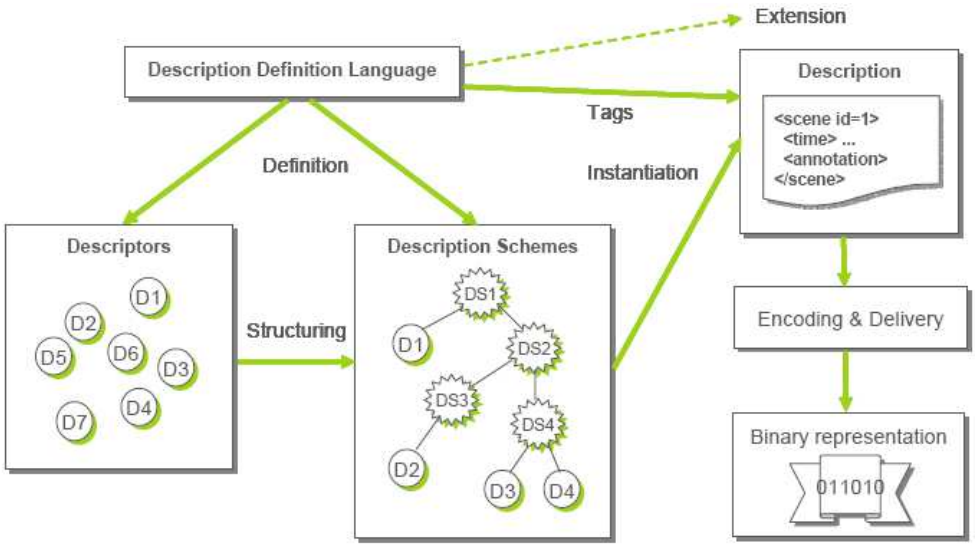

Figure 3 The Main Elements of MPEG-7

An overview of the organization of MPEG-7 Multimedia DSs into the areas of Basic Elements, Content Description, Content Management, Content Description, Content Organization, Navigation and Access, and User Interaction is provided in Figure 4 [23].

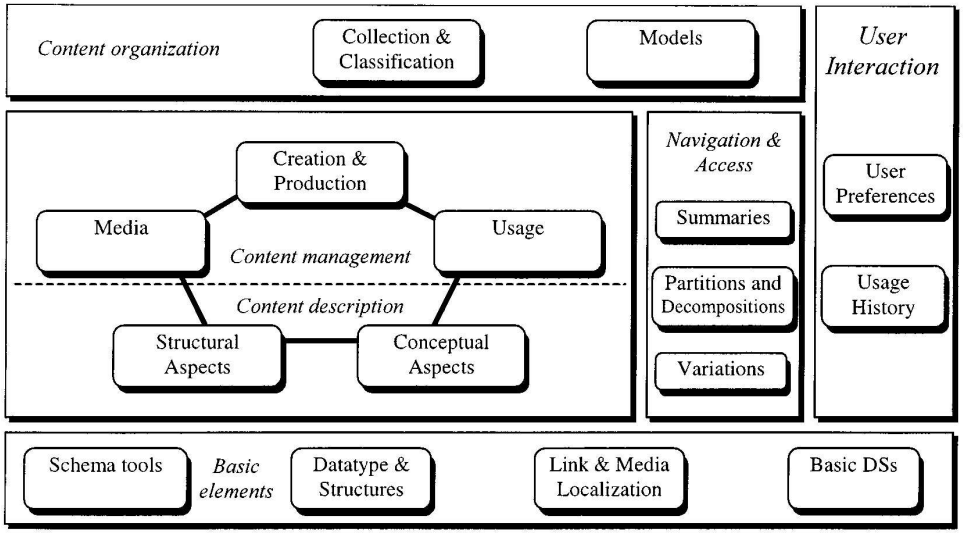

Figure 4 Overview of the MPEG-7 Multimedia DS's.

The MPEG-7 tools that are used in description of semantic content are illustrated in Figure 5.

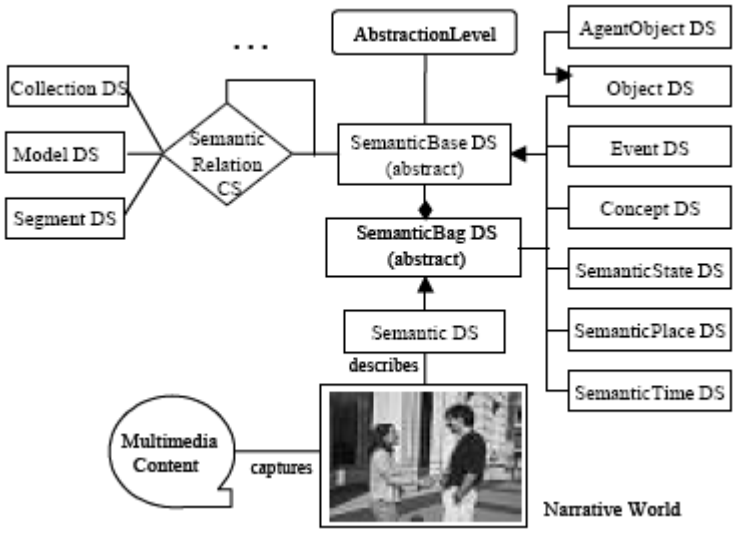

Figure 5 The MPEG-7 Tools for the description of semantic content.

Important description schemes are summarized as follows:

- *SemanticBase DS:* The SemanticBase DS is an abstract type which holds common functionalities such as labels, textual description etc. and relationships to other entities, in describing semantic content.

- *SemanticBag DS and Semantic DS:* The Semantic DS (via the SemanticBag DS) is derived from SemanticBase DS in order to enable the use or embed one narrative world in another.

- *Object DS and Event DS:* The Object DS and Event DS extend SemanticBase DS. They describe objects and events that can exist or take place in time and space and support recursive descriptions.

- *AgentObject DS:* The AgentObject DS extends Object DS and describes the actors that appear in an audiovisual segment [8].

- *SemanticPlace DS and Semantic Time DS:* The SemanticPlace DS and SemanticTime DS are derived from SemanticBase DS and are used in description of location and time in a narrative world.

- *SemanticState DS:* The SemanticState DS extends SemanticBase DS and describes the state or parametric attributes of a semantic entity in an audiovisual content.

- *Concept DS:* The Concept DS is derived from SemanticBase DS and used in description of concepts in an audiovisual content.

3.2 The MPEG-7 Ontologies

In MPEG-7, semantically identical metadata can be represented in multiple ways because of not being able to define precise semantics in XML-based syntax. For instance, four different representations of the same semantic content are illustrated in the following [31]:

The first is by using free text annotation:

```
<FreeTextAnnotation xml:lang="en">
    Zinedine Zidane scoring against England.
</FreeTextAnnotation>
```

The second way is by using keyword annotation:

```
<KeywordAnnotation xml:lang="en">
    <Keyword>Zinedine</Keyword>
    <Keyword>Zidan</Keyword>
    <Keyword>scoring</Keyword>
    <Keyword>England</Keyword>
    <Keyword>goal</Keyword>
</KeywordAnnotation>
```

The third way is by using structured annotation with labels:

```
<StructuredAnnotation>
    <Who>
        <Name xml:lang="en">
            Zinedine Zidane
        </Name>
    </Who>
    <WhatAction>
        <Name xml:lang="en">
            Zinedine Zidane scoring against England.
        </Name>
    </WhatAction>
</StructuredAnnotation>
```

The forth way is by using MPEG-7 semantic descriptor:

```
<Semantic id="FormalAbstractionDescription">
    <SemanticBase xsi:type="AgentObjectType" id="Zidane">
        <Label>
            <Name>
                Zidane
            </Name>
        </Label>
        <Agent xsi: ="PersonType">
            <Name>
                <GivenName>
                    Zinedine
                </GivenName>
                <FamilyName>
                    Zidane
                </FamilyName>
            </Name>
        </Agent>
    </SemanticBase>
    <SemanticBase xsi:type="EventType" id="scoring">
        <Label>
            <Name>
                Zinedine Zidane scoring against England.
            </Name>
        </Label>
    </SemanticBase>
</Semantic>
```

For specifying MPEG-7 Descriptors and Description Schemes, XML Schema language is used as the DDL because of the fact that XML is not only capable of expressing the syntactic, structural, cardinality and data typing constraints necessary for MPEG 7, but also enables extension and refinement of existing

Descriptors and Description Schemes [17]. MPEG-7 standard is implemented by XML Schemas that defines 1182 elements, 417 attributes and 377 complex types hardening the management of the standard and remaining semantics implicit [16]. For this reason, each application using MPEG-7 must redevelop extraction of semantics from the standard. If the application uses XQuery to retrieve MPEG-7 SegmentType, the developer should know the hierarchy of segment types and handle each segment type such as VideoSegmentType, AudioSegmentType and etc [16].

In order to allow interoperability and integration of the standard with application domains, MPEG-7 Descriptors and Description Schemes should be supported with a machine understandable representation of semantics [17]. Thus, the similarity of the aforementioned four statements become machine understandable and the XQuery for segment type (previously stated) retrieves all segment types immediately.

Recently, several efforts have been spent to translate MPEG-7 standard into an ontology and to enable its integration with other ontologies through appropriate frameworks, thus enhancing interoperability. There are four main MPEG-7 ontology proposals which are explained in the following subsections.

3.2.1 The MPEG-7 Ontology Developed by Jane Hunter

The first efforts towards semantic formalization of MPEG-7 were carried out, by Jane Hunter [17][18].

MPEG-7 specification does not have a data model. Due to this fact, the class and property hierarchies and semantic definitions were derived by reverse-engineering the existing XML Schema definitions and interpreting the semantic descriptions and simplifying this by using a core subset of MPEG-7 with a top-down approach [17]. MPEG-7 top level entities are illustrated in Figure 6 [17].

The resulting MPEG-7 used RDF and RDF Schema to formalize a small part of MPEG-7. However, the ontology is incorporated some DAML+OIL constructs to

detail semantics, where the DAML+OIL ontology definition language was used to partially describe the MPEG-7 MDS and visual metadata structures. The ontology has been recently translated into OWL [15]. A major shortcoming of Jane Hunter's MPEG-7 ontology is its limited coverage of the MPEG-7 standard.

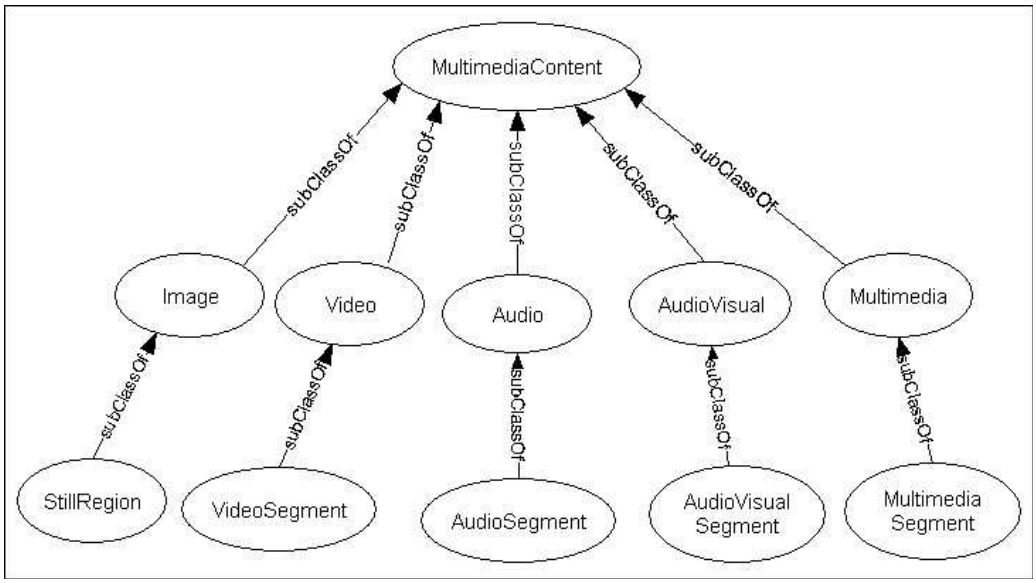

Figure 6 Class Hierarchy of MPEG-7 Top-level Multimedia Content Entities

3.2.2 The MPEG-7 Ontology Developed by Tsinaraki

Tsinaraki developed an upper ontology with complexity OWL-DL fully capturing the MPEG-7 MDS with portions of Visual and Audio needed by MPEG-MDS. A methodology enabling the transformation of the MPEG-7 XML Schema to OWL-DL was used to manually develop the ontology. [5][11][12][13]

The methodology that is used during manual development of semantic part of MPEG-7 in OWL can be summarized as follows:

- *MPEG-7 Simple Data type Representation:* OWL allows integrating simple data types defined in the XML Schema Language to OWL ontologies using the rdfs:Datatype construct. Therefore, simple data types are integrated in the MPEG-7 ontology [10].

  For instance, a datatype "zeroToOneType" representing real numbers between 0 and 1, is defined by rdfs:Datatype in OWL. [6]

  zeroToOneType datatype in the MPEG-7 MDS:

```
<simpleType name="zeroToOneType">
    <restriction base="float">
        <minInclusive value="0.0" />
        <maxInclusive value="1.0" />
    </restriction>
</simpleType>
```

  rdfs:Datatype instance for the zeroToOneType datatype:

```
<rdfs:Datatype rdf:about="&datatypes;zeroToOneType">
    <rdfs:isDefinedBy rdf:resource="&datatypes;"/>
    <rdfs:label>zeroToOneType</rdfs:label>
</rdfs:Datatype>
```

- *MPEG-7 Complex Type Representation:* For every complex type a respective OWL class has been defined. [9].

  - All simple attributes of the complex type have been represented as OWL data type of the appropriate type.

  - Complex attributes of the complex type have been represented by defining the following:

    - an OWL class for the representation of complex attribute instances,

    - an OWL object property for relating the complex attribute instances with the complex type instances.

For example, OWL definitions for PersonType class, citizenship simple attribute and name complex attribute are as follows [9] :

```
<owl:Class rdf:ID="PersonType">
      <rdfs:subClassOf rdf:resource="#AgentType"/>
      <rdfs:label>Person</rdfs:label>
      <rdfs:subClassOf>
           <owl:Restriction>
                <owl:onProperty rdf:resource="#Name"/>
                <owl:minCardinality
                     rdf:datatype="&xsd;nonNegativeInteger">
                     1
                </owl:minCardinality>
           </owl:Restriction>
      </rdfs:subClassOf>
</owl:Class>

<owl:ObjectProperty rdf:ID="Name">
     <rdfs:domain rdf:resource="#PersonType"/>
     <rdfs:range rdf:resource="#PersonNameType"/>
</owl:ObjectProperty>

<owl:DatatypeProperty rdf:ID="Citizenship">
     <rdfs:domain rdf:resource="#PersonType"/>
     <rdfs:range rdf:resource="&datatypes;countryCode"/>
</owl:DatatypeProperty>
```

- *MPEG-7 Complex Type Representation:* Relationships among semantic entities are defined by SemanticRelationType class and its subclasses [9].

As a result of the study, an MPEG-7 ontology of OWL-DL complexity was developed manually by applying the aforementioned rules.

3.2.3 The MPEG-7 Ontology Developed by Rhizomik

The aim of Garcia and Celma is to obtain an MPEG-7 ontology that completely covers the standard which forms a complete semantics-aware solution for MPEG-7 metadata processing. [16] In order to accomplish this, an automatic conversion approach should be adopted rather than a manual approach due to the giant size of the MPEG-7 standard. Therefore, the methodology is mapping MPEG-7 standard XML Schema constructs to OWL constructs by using a generic XML Schema to OWL mapping which is combined with an XML to RDF translation.

The main contribution is enabling to consume the great amount of metadata that has been already produced by previous multimedia data management systems that use XML- based MPEG-7 standard. [15]

There exist many approaches in moving metadata from XML-based MPEG-7 systems to MPEG-7 ontology based systems. However, they produce RDF metadata that is as semantics blind as the XML data due to not using XML semantics or use additional ad-hoc semantic constructs [15].

In Rhizomik approach, the ReDeFer project [39] is used in moving MPEG-7 standard to MPEG-7 ontology via the following steps:

- *XSD2OWL Mapping:* The XML Schema to OWL mapping captures the schema informal semantics which are derived from the combination of XML Schema constructs. The semantic constructs captured are mapped to the OWL constructs that best fits them [16].

  The MPEG-7 ontology which has 2372 classes and 975 properties was generated by applying XSD2OWL mapping on MPEG-7 xml schemas. However, OWL has unique name domain for all constructs in oppose to XML's separate name domains for complex types, leading to name collision between an OWL class and RDF property. This is fixed by resolving the name collisions manually. [15]

  Moreover, an MPEG-7 ontology of OWL-Full complexity is developed.

- *XML2RDF Mapping:* In XML to RDF mapping, the available XML metadata that instantiates the OWL ontology which is obtained in the previous step, is mapped to RDF metadata in a transparent way. [16]

  The approach accomplishes transparency by using structure-mapping models which represents the XML metadata structure using RDF which is based on graph that can easily be used in modeling a tree.

In this thesis, MPEG-7 ontology developed by Rhizomik is used due to its complete coverage of the whole standard and its promise to enable usage of existing XML domain metadata.

3.2.4 The COMM

The aim is to combine the benefits of web based solutions with the experience of MPEG-7. COMM [41] which is a core ontology for multimedia, is the re-engineering of MPEG-7 using DOLCE design patterns [19]. COMM is of OWL-DL complexity.

3.3 The MPEG-7 Format Selection

Since the aim of the thesis is to provide a multimedia data management system with enhanced semantic content retrieval capabilities, the format of MPEG-7 with powerful semantics is the obvious selection. Among the four aforementioned MPEG-7 ontologies, there are significant differences in terms of coverage of the standard and expressiveness power. A brief comparison of existing four MPEG-7 ontologies is given in Table 1.

In this thesis, interoperability with other MPEG-7 based multimedia management systems is crucial. Besides, backward compatibility with the systems that use MPEG-7 in the form of XML Schemas is also important, since there exist many such systems. Therefore, Rhizomik MPEG-7 ontology is chosen to be the upper ontology in the proposed system.

Table 1 Comparison of MPEG-7 Ontologies

	Hunter	DS-MIRF	Rhizomik	COMM
Foundations	ABC	None	None	DOLCE
Complexity	OWL-Full	OWL-DL	OWL-DL	OWL-DL
Coverage	MDS+Visual	MDS+CS	All	MDS+Visual
Applications	Digital Libraries	Digital Libraries	Digital Right	MM Analysis

3.4 Integrating the MPEG-7 Ontology and Domain Ontologies

In order to enhance semantic interoperability, domain knowledge is integrated to MPEG-7 constructs in the form of domain ontologies [4][20][21][22]. Two main approaches to integrate MPEG-7 ontology with domain ontologies include proposals by Hunter et. al.[17], [31] and Tsinaraki et. al. [4], [31].

In the approach proposed by Jane Hunter, ABC ontology is used as the core ontology and it provides attachment points for integrating MPEG-7 and domain specific ontologies. Technically, the mpeg7:MultimediaContent class is extended from the abc:Manifestation class and other domain ontologies are attached to the core ontology via extending from corresponding ABC classes.

Another approach developed by Tsinaraki uses the MPEG-7 ontology as the upper ontology and attaches domain specific ontologies to MPEG-7 ontology basically. The integration process includes sub classing domain ontology classes from the following constructs of the upper ontology:

- SemanticBaseType
    - EventType
    - ObjectType
    - AgentObjectType

- SemanticPlaceType
- SemanticTimeType
- ConceptType.

In this thesis, Tsinaraki's approach which accepts MPEG-7 ontology as the core ontology and attaches domain specific ontologies to MPEG-7 is followed [14]. However, in this approach, the ontology engineer should be aware of the existence of MPEG-7 ontology and some constraints should be imposed on the developed ontology. One of the major goals of this study is to hide the MPEG-7 ontology existence and domain specific ontology integration details from the end user in order to enable easy and efficient usage of "bare" domain ontologies. Therefore, the harmonization of MPEG-7 ontology and domain specific ontologies should be carried out in an automated way. Details for the automatic integration of MPEG-7 ontology and domain ontologies are presented in Chapter 4.

.

CHAPTER 4

DESIGN AND IMPLEMENTATION OF THE ONTOLOGICAL MULTIMEDIA DATA MANAGEMENT FRAMEWORK

The design and implementation details of the ontological multimedia data management framework which includes data model, architectural design, main characteristics of the components, API provided by some important utility components, the technologies used during development etc. are explained in this chapter.

4.1 The Ontological Video Model

In Chapter 3, the rationale for using MPEG-7 standard in the form of MPEG-7 ontology, the selection of the MPEG-7 ontology to use in the system and basic selection criteria and two main approaches for integrating MPEG-7 ontology and domain ontologies are presented. It is stated that the approach developed by Tsinaraki is adopted for harmonization of MPEG-7 ontology and domain specific ontologies.

One of the contributions of this thesis is automating the integration of MPEG-7 ontology and domain specific ontologies. The key that enables the automation of

integration process is the fact that the aforementioned MPEG-7 Semantic DS concepts are conceptually meaningful and complete independent of the standard to describe the narrative world. Examining the schema from a conceptual point of view:

- *Semantic Base:* describes common functionalities such as labels, textual description etc. and relationships to other entities, in describing semantic content.

- *Object :* describes objects that can exist or take place in time and space.

- *Event :* describe events that can exist or take place in time and space.

- *AgentObject:* describes the actors that appear in an audiovisual segment which can be a person, a person group or an organization.

- *SemanticPlace:* describes a location in a narrative world.

- *Semantic Time:* describes time in a narrative world.

- *SemanticState:* describes the state or parametric attributes of a semantic entity in an audiovisual content.

- *Concept:* describes concepts in an audiovisual content, such as “friendship”.

In order to automate the integration of MPEG-7 and domain ontologies, the concepts in the domain ontology are categorized into the above categories which are supplied from MPEG-7 Semantic DS via a user friendly interface and the selected concepts are automatically sub classed from the corresponding super concept selected. However, an intermediate ontology is located between MPEG-7 ontology and domain ontologies. The super concepts that correspond to the categories are actually the concepts in the intermediate ontology which are subclasses of MPEG-7 ontology concepts. The reason for using an intermediate

ontology instead of directly attaching domain ontologies to MPEG-7 is to enable the addition of a property to a concept in ease without requiring the modification of the original MPEG-7 ontology. The intermediate ontology and MPEG-7 relationship is visualized in Figure 7.

4.2 The System Architecture

The ontological multimedia data management framework is a standard web application. The development environment is Eclipse Europa and the technologies that have been used during development can be summarized as follows:

- Server side:
    - Java programming language
    - Jena API
    - SPARQL query language
    - Rhizomik MPEG-7 ontology
    - JMF (for multimedia player applet)
- Client side:
    - Facelets
    - Richfaces and Myfaces components
- Apache Tomcat as web server
- Mozilla Firefox as web browser
- MySQL database

The system framework is designed in a modular way by using abstractions for jena ontology API and MPEG-7 related functionality in order to make the system not depend on them. The architecture of the developed ontological multimedia data management framework is summarized in Figure 8.

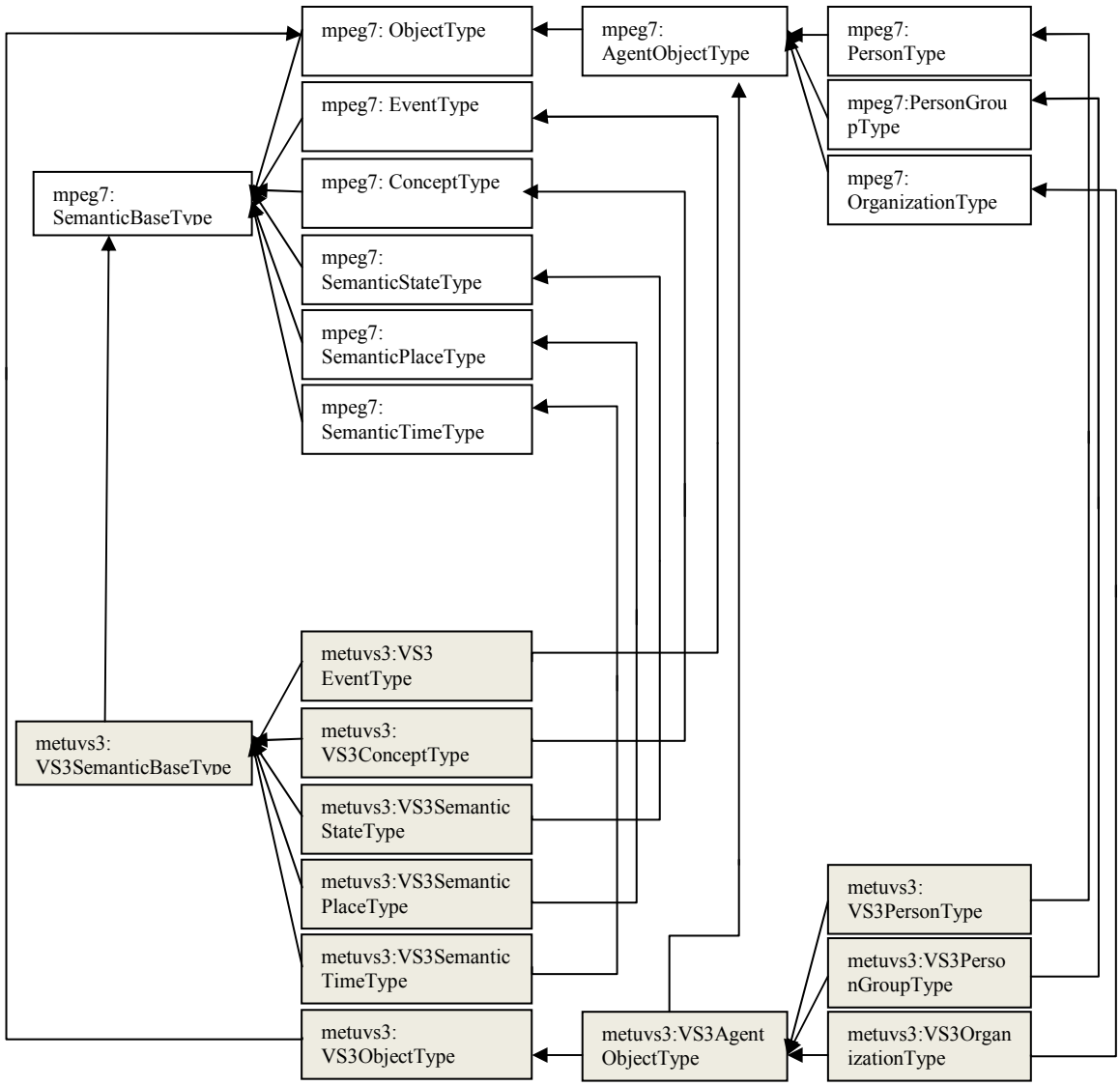

Figure 7 The Relationship between MPEG-7 ontology and Intermediate Ontology

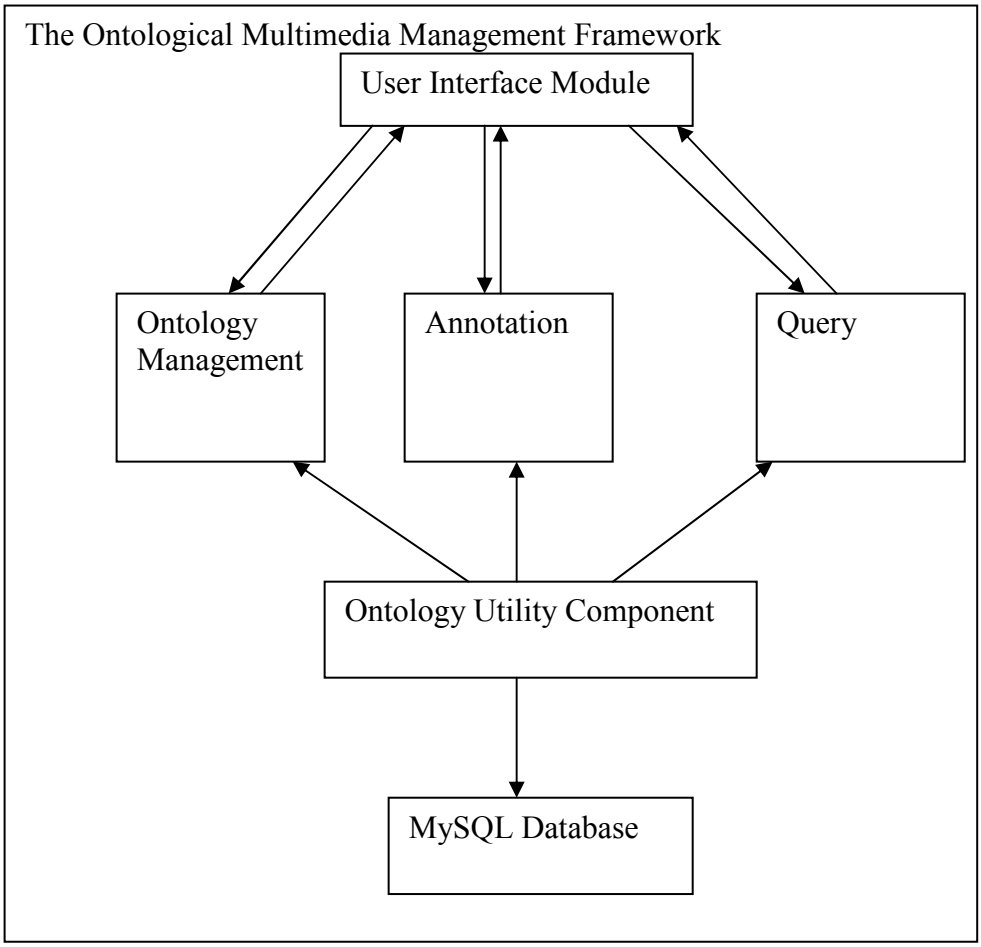

Figure 8 The System Architecture

4.3 The Components

Modularity is one of the main goals of the implementation; therefore the components interact with each other through strictly defined interfaces. In this subsection, the components and the current implementations provided are explained and the important API functionalities are also presented.

The proposed system mainly consists of the following components:

- *Ontology Utility Component:* This component provides the necessary high level functions related to ontology management to other components.

- *Ontology Management Component*: This component is capable of ontology deletion, insertion and configuration.

- *Annotation Component:* This component is capable of video content annotation.

- *Query Component*: This component is capable of video content querying.

- *User Interface Component*: This component provides user-friendly interfaces for ontology management, annotation and querying.

The ontology management, annotation and query components delegate ontology related functionality to ontology utility component and they are mainly responsible for serving the presentation layer. The current implementation of ontology utility component uses MPEG-7 ontology and Jena API. However, the component is actually independent of these constructs.

4.3.1 Ontology Utility Component

The ontology utility component encapsulates the ontology-related functionality details by providing a comprehensive set of functions to other components at upper level. The general architecture of the component is summarized in Figure 9.

The component includes two subcomponents:

- Multimedia Utility Component: provides multimedia related functionality.
- Ontology API Utility Component: provides high level ontology related functionality.

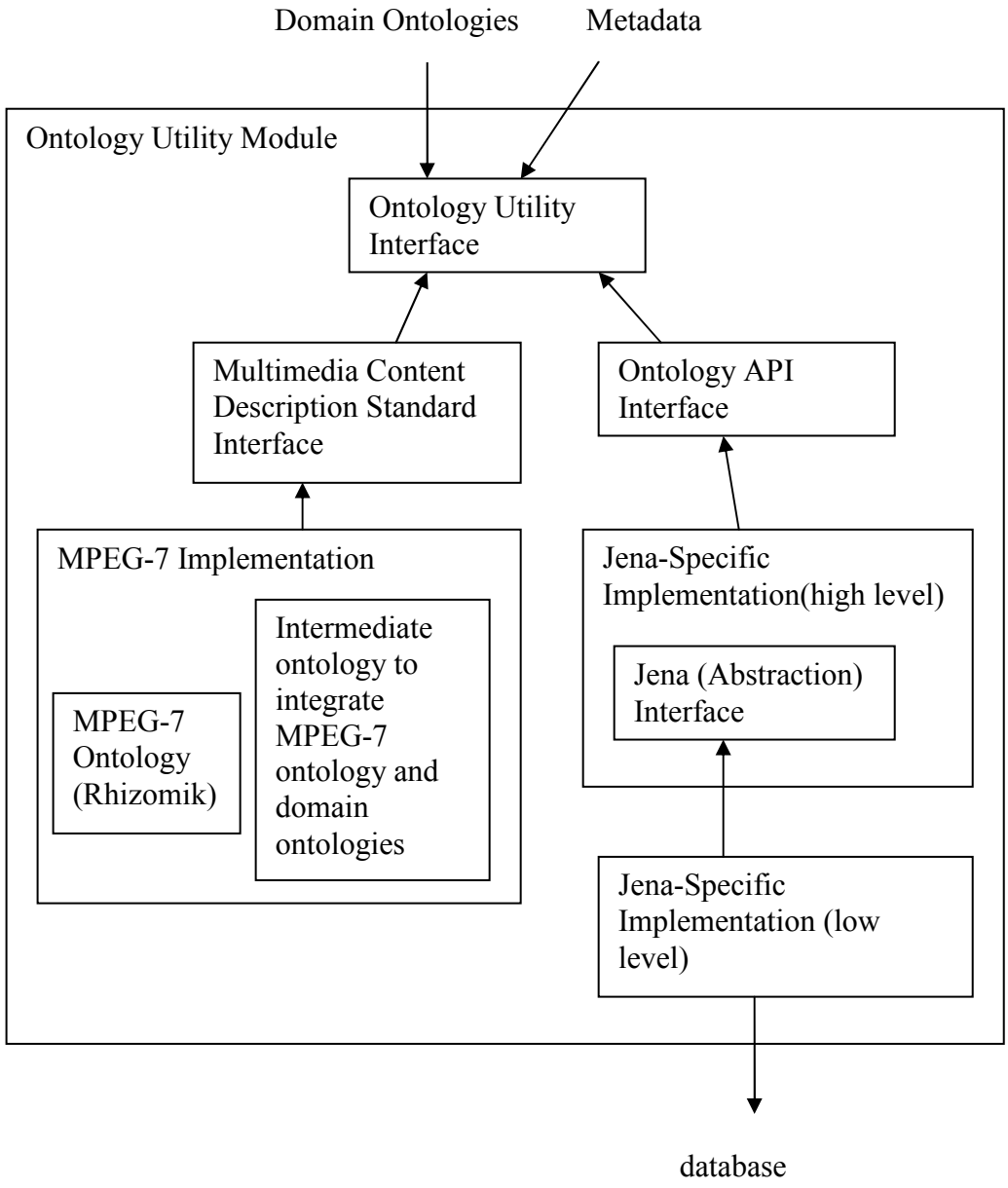

Figure 9 The Ontology Utility Module Architecture

The main functionality that is provided by the ontology utility component can be summarized as follows:

- initSystem() : initializes the system by cleaning the database and loading the upper ontology
- getAllModelsFiltered(): returns nontrivial models in the database. MPEG-7 ontology and intermediate ontology models are considered as trivial and are not retrieved in the list returned by this method. This filtering process is carried out by multimedia utility subcomponent.
- getAllConceptsFiltered(model): returns the nontrivial concepts for the given model that are in nontrivial namespaces. The following namespaces are considered as trivial and concepts with these namespaces are not retrieved by this method:
    - OWL namespace: http://www.w3.org/2002/07/owl#
    - RDF namespace: http://www.w3.org/1999/02/22-rdf-syntax-ns#
    - RDFS namespace: http://www.w3.org/2000/01/rdf-schema#
    - XML Schema namespace: http://www.w3.org/2001/XMLSchema#

  The filtering process is done by multimedia utility subcomponent.

- getAllAnnotationConcepts(model): returns the concepts that satisfy the constraints in the above method besides being a subclass of metuvs3:VS3SemanticBaseType. The subclassing of given domain ontologies are carried out via a user friendly interface by enabling the user to categorize the annotation concepts in corresponding categories.

4.3.1.1 Multimedia Utility Component

The multimedia utility component which is an abstraction on the ontological video model used, is responsible for providing functionality related to the upper ontology, intermediate ontology and the integration of domain ontologies. In this thesis, an MPEG-7 implementation that implements the ontological video model which is mentioned in the previous chapter is used.

The important functions provided by this component include the following:

- loadMultimediaOnto(): loads the underlying upper ontology and intermediate ontology in to the system.
- wrapModel(model, conceptList, superConcept): wraps the conceptList for the given model with the superConcept selected.
  This function enables automatic integration of domain ontologies and the upper ontology through the intermediate ontology.
- filterModel(modelList): removes the trivial models from modelList. The upper ontology and intermediate ontology models are considered as trivial.
- filterConcept(model, conceptList): removes the concepts with trivial namespaces from the conceptList. It is aforementioned that OWL, RDF, RDFS and XML Schema namespaces are considered as trivial.
- filterAnnotationConcept(model, conceptList): removes the concepts that are not subclasses of metuvs3:VS3SemanticBaseType besides concepts in trivial namespaces.

4.3.1.2 Ontology API Utility Component

The ontology api utility component is an abstraction over the ontology api used and provides a comprehensive set of high level functions related to ontology management.

The important functions provided include the following:

- createModel(model): creates a model with the given name.
- createConcept(model, concept): creates a concept with the given name in the given model.
- createProperty(model, property, domainModel, domain, rangeModel, range): creates a property with the given name in the given model with the given domain and range.

- addSuperClass(model, concept, superModel, superConcept): extends the concept in the model from the superConcept in the superModel.
- hasSuperClass(model, concept, superModel, superConcept): checks if the concept in model is subclass of the superConcept in superModel.
- removeModel(model): removes the given model from the database.
- importModel(model, file): imports the given ontology into the system with the given model name.
- getAllModels(): returns a list of the models in the database.
- getAllConcepts(model): returns a list of the concepts in the given model.
- getAllAttributes(model, concept): returns a list of properties of the given concept in the given model.
- getAllAttributeValues(model, concept, attribute): returns a list of the values of the given attribute of the given concept in the given model.
- getAllIndividuals(model, concept): returns a list of the individuals of the given concept in the given model.
- getAllAnnotations(mediaUri): returns all the annotations of the given media.
- findMedia(annotation): returns a list of the media locator which consists of the mediaUri, start frame and end frame, for the given annotation. Annotation consists of the following:
    - model
    - concept
    - individual, *optional*
    - attribute-value pairs, *optional*
- addIndividual(model, concept, individual, attributeHashMap, mediaLocator): adds the given individual for the given concept in the given model. Besides, the attribute-value pairs in the attributeHashMap are inserted in the system. This individual is marked to appear in the interval of the video given in mediaLocator.

In this thesis, Jena API is used as ontology api. A Jena API abstraction is used in order to minimize the dependency on Jena.

4.3.1.2.1 Jena API Abstraction

This component is an abstraction of Jena API providing the following functionality:

- getModelMaker(): returns the Jena specific model maker.
- getModel(model), createModel(model): respectively returns or creates a Jena specific model with the given name.
- getOntModel(model), createOntModel(model): respectively returns or creates a Jena specific ontology model with the given name.
- getResource(model, resource), createResource(model, resource): respectively returns and creates the Jena specific resource with the given name.
- getOntClass(model, concept), createOntModel(model, concept): respectively returns or creates a Jena specific ontology class with the given name in the given model.
- getIndividual(model, concept, individual), createIndividual(model, concept, individual): respectively returns or creates a Jena specific individual with the given name of the given concept in the given model.

4.3.2 Ontology Management Component

The ontology management component provides high level ontology management functionality to the presentation layer. It uses functions provided by the ontology utility component in order to enable the following functionality:

- Ontology import: The component enables import of a given domain ontology.

- Ontology configuration: The component enables configuration of an ontology by categorizing the concepts in the domain ontology in to the following categories:
    - metuvs3:VS3SemanticBaseType
    - metuvs3:VS3ObjectType
    - metuvs3:VS3AgentObjectType
    - metuvs3:VS3PersonType
    - metuvs3:VS3PersonGroupType
    - metuvs3:VS3OrganizationType
    - metuvs3:VS3EventType
    - metuvs3:VS3ConceptType
    - metuvs3:VS3SemanticStateType
    - metuvs3:VS3SemanticPlaceType
    - metuvs3:VS3SemanticTimeType
- Ontology deletion: The component enables deletion of selected ontologies.

The general architecture of the component is summarized in Figure 10.

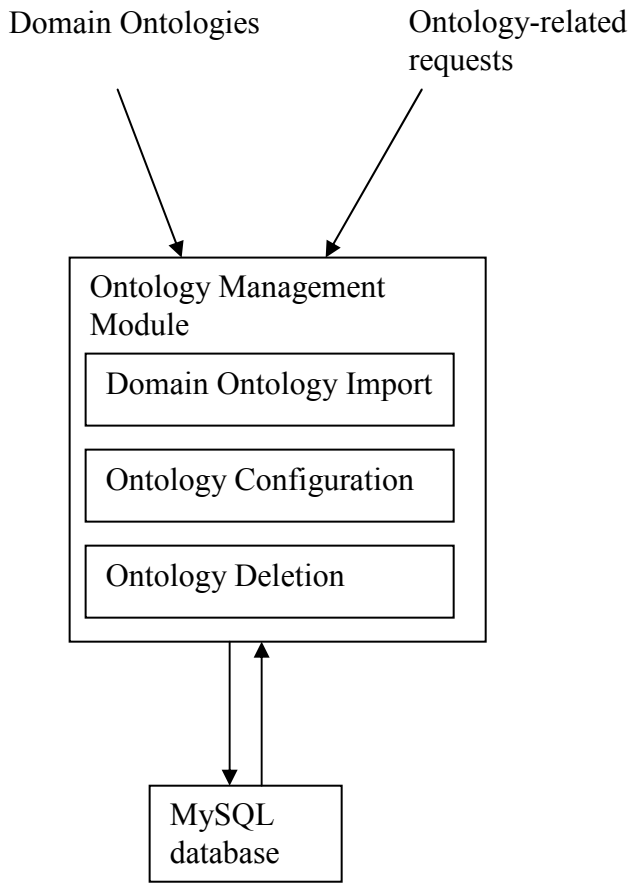

Figure 10 The Ontology Management Module

.

4.3.2 Annotation Component

The annotation component presents the necessary functionality for the ontology-driven content annotation to presentation layer. The component uses the ontology utility component in order to provide the required functionality. The Java Media Framework (JMF) video player applet is included in this module. The component enables ontology-driven annotation by providing the necessary communication between the ontology utility component and the user friendly annotation interface.

The component is summarized in Figure 11.

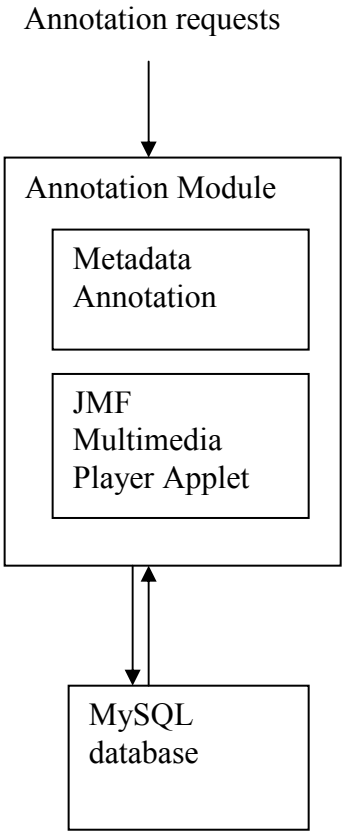

Figure 11 The Annotation Module

4.3.3 Query Component

The query component provides querying functionality to the presentation layer by using the functionality coming from the ontology utility component. This component manages the interaction with the ontology utility component and the user friendly ontology-driven query interface.

The component is given in Figure 12.

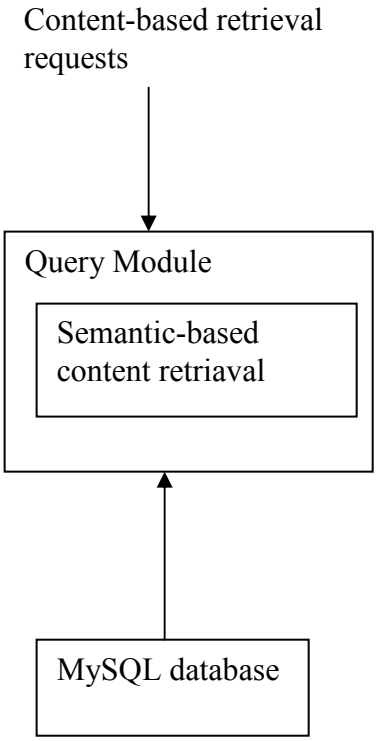

Figure 12 The Query Component

CHAPTER 5

CASE STUDY

During evaluation of the framework, the framework is tested against two application domains which are soccer and animal domains in terms of ease of integration, annotation and querying power. Actually, the application domain does not affect the usage of the program, therefore in this chapter screenshots are captured for soccer domain.

In this chapter, the usage of the proposed system is explained by the help of use cases and screenshots. The reason for the detailed use cases and screenshots is the fact that, since the user interfaces of the system are designed for generic usage independent of any special domain, the usage may seem complicated to an unfamiliar end user.

The main page of the system is shown in Figure 13.

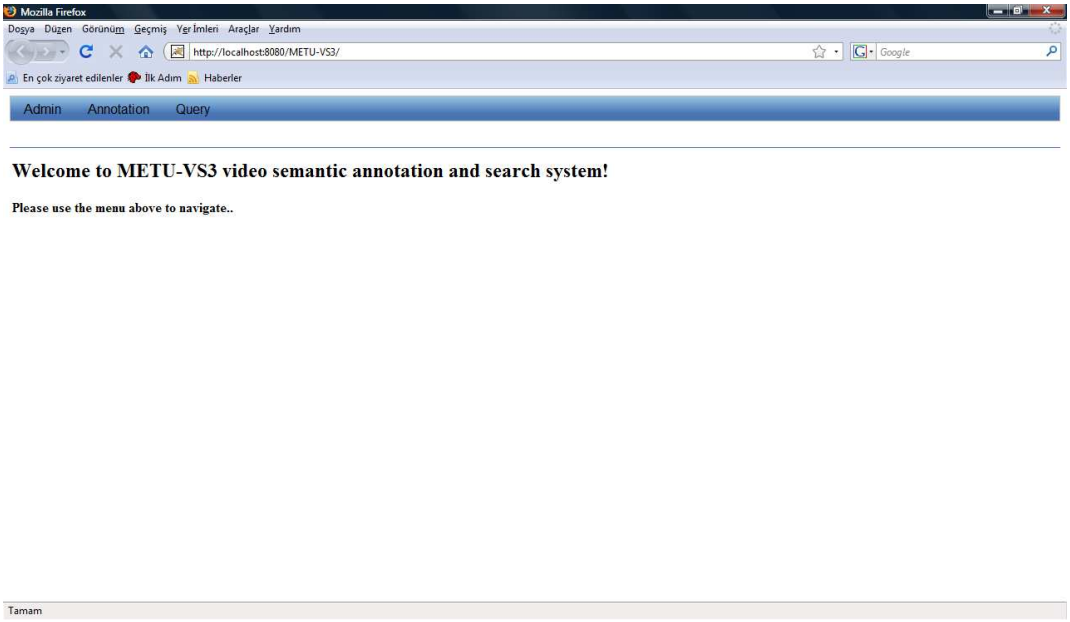

Figure 13 The Main Page

5.1 Ontology Management

The ontology management module is the module which enables interaction with the domain ontologies. The user friendly page that enables automatic harmonization of MPEG-7 upper ontology and domain ontologies is included in this component.

The Ontology Management menu includes the following:

- System initialization
- Ontology import
- Ontology configuration
- Ontology deletion

The ontology management menu is shown in Figure 14.

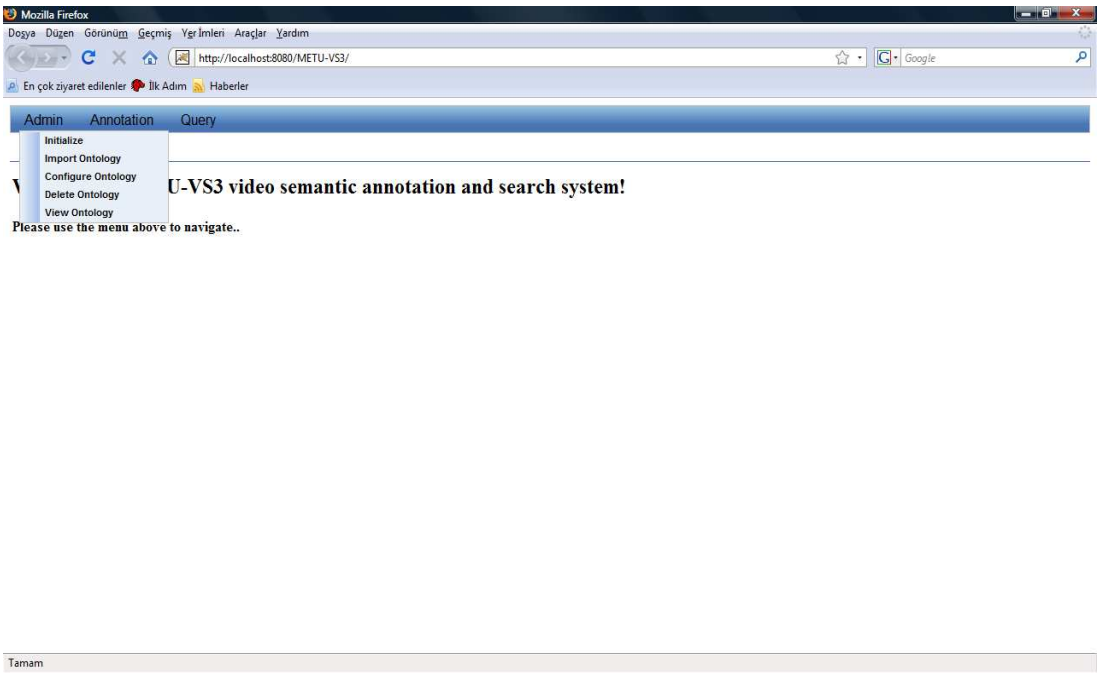

Figure 14 The Ontology Management Menu

5.1.1 System Initialization

The proposed system should be initialized through this page in order to use annotation and querying facilities. During initialization, the system is totally cleaned by deleting the ontologies and data in the database. When the system is clean, the upper ontology which consists of the Rhizomik MPEG-7 ontology and the aforementioned intermediate ontology is loaded to the database.

The use case for the positive path is as follows:

1. The user selects Admin → Initialize from system menu.
2. The initialization page appears.
3. The user clicks on “Initialize” link.
4. “System is successfully initialized” message appears.

The system initialization page is shown in Figure 15.

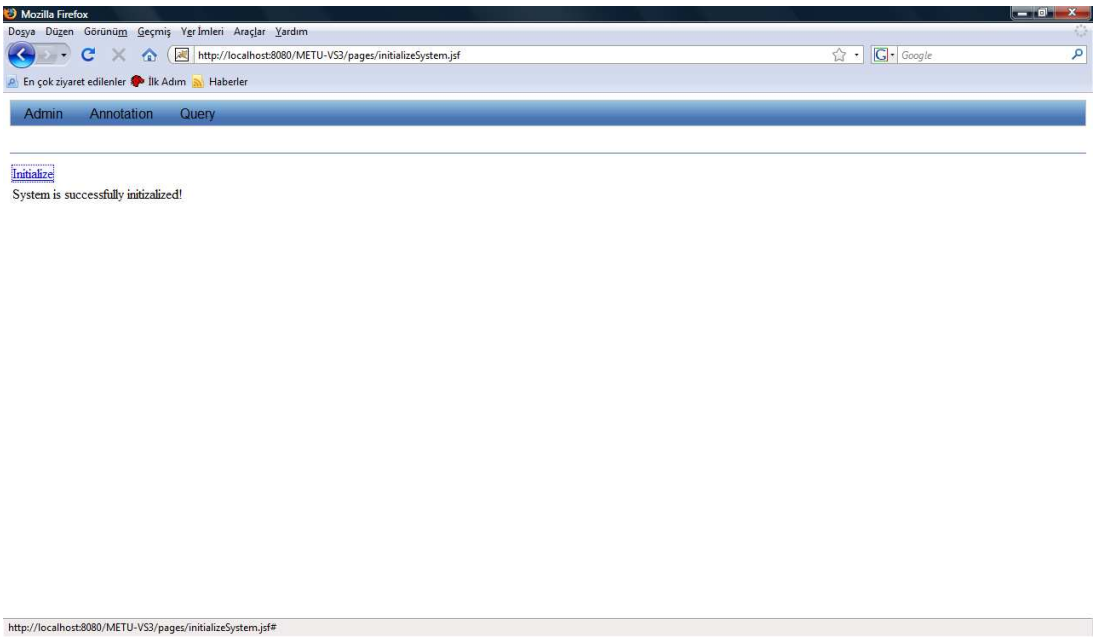

Figure 15 The System Initialization Page

5.1.2 Ontology Import

Import of domain specific ontologies is carried out via this page. The positive path use case for importing an ontology is as follows:

1. The user selects Admin → Import Ontology from system menu.
2. The ontology import page appears.
3. The user enters model name for the ontology that is subject to import.
4. The user selects the ontology file to import from the file system.
5. The user clicks “Import” button.
6. “Ontology is successfully imported”message appears.

The ontology import page is shown in Figure 16.

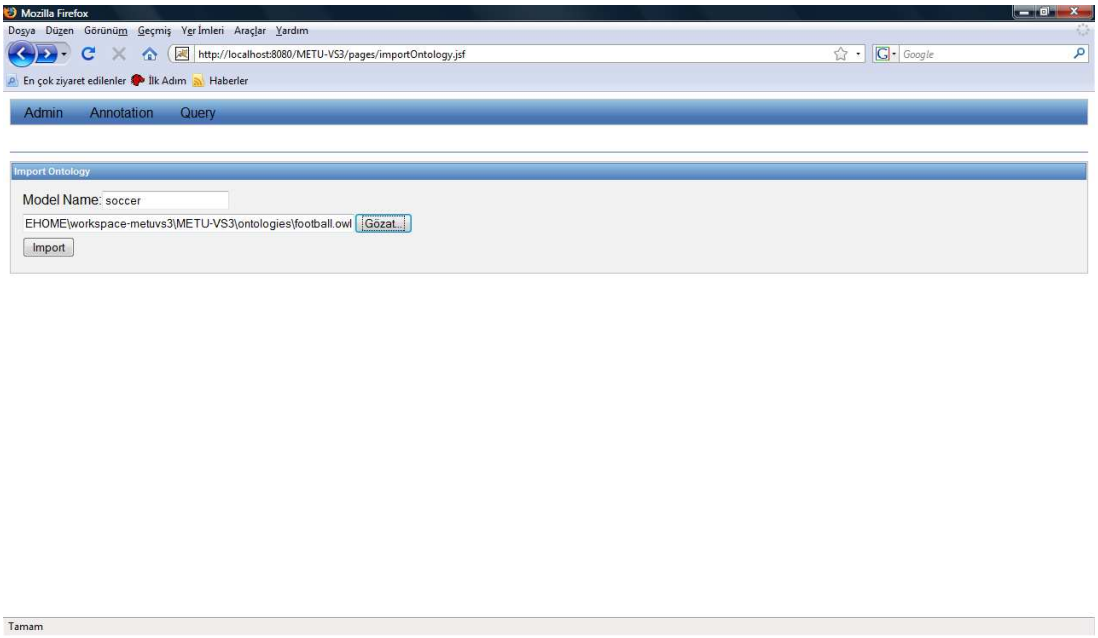

Figure 16 The Ontology Import Page

5.1.3 Ontology Configuration

The system enables the user to categorize the concepts in the imported ontology into the following categories: Semantic Base, Object, Agent Object, Person, Person Group, Organization, Concept, Semantic Place, Semantic Time and Semantic State. Afterwards, the system subclasses the selected concept from the corresponding super concept in the intermediate ontology according to the selected category. The positive path use case is as follows:

1. The user selects Admin → Configure Ontology from system menu.
2. The ontology configuration page appears.
3. The user selects the model name of the ontology that is subject to configuration.
4. The concepts in the selected model are loaded to concept pick up lists.

5. The user categorizes the concepts that are going to be used in annotation and querying.
6. "Ontology is successfully configured"message appears.
7. The user repeats steps 5 to 8 until ontology is totally configured.

The ontology configuration page is shown in Figure 17.

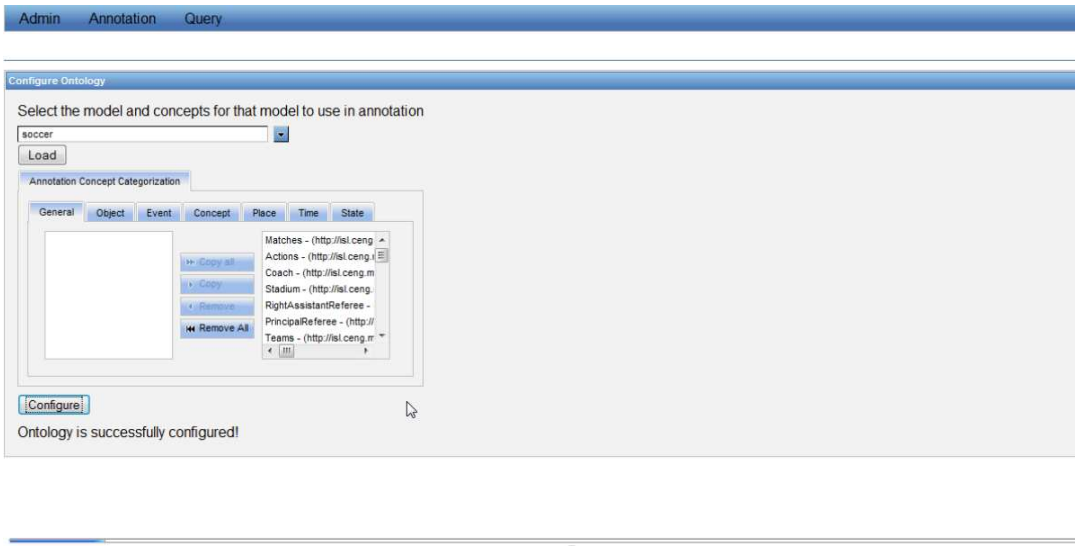

Figure 17 The Ontology Configuration Page

5.1.4 Ontology Deletion

The system enables ontology deletion through this page. The positive path use case is as follows:

1. The user selects Admin → Delete Ontologies from system menu.
2. The ontology deletion page appears.
3. The user selects the model names that are ontologies subject to deletion.
4. The user clicks "Delete" button.

5. “Ontologies are successfully deleted”message appears.

The ontology deletion page is shown in Figure 18.

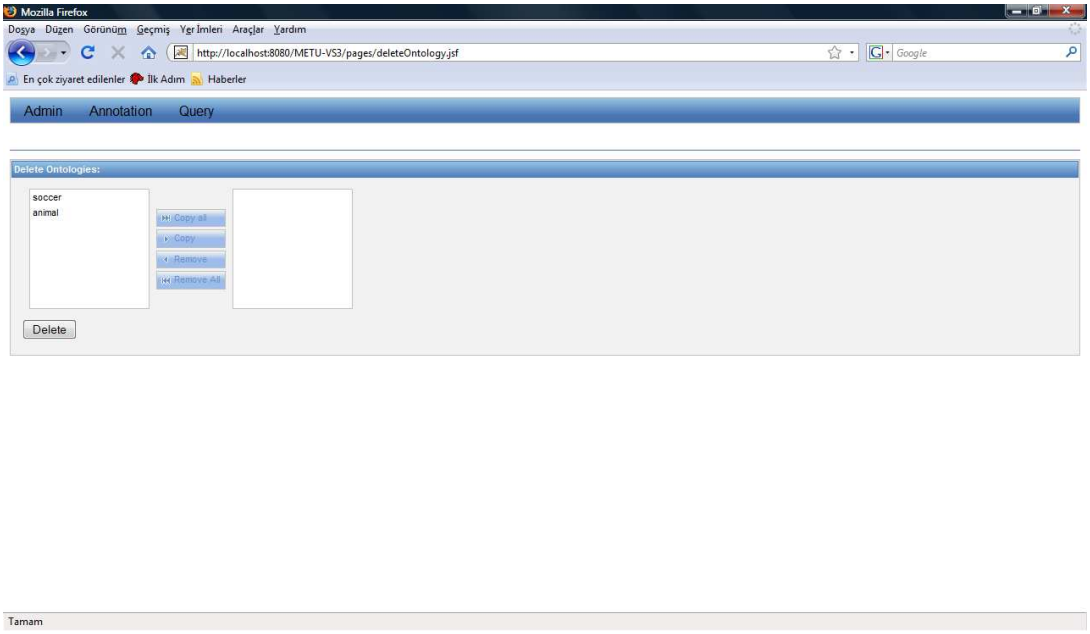

Figure 18 The Ontology Deletion Page

5.2 Annotation

The annotation page provides ontology-driven annotation by enabling the selection of domain ontology where the annotation concepts come from.

1. The user selects Annotation → Annotation from system menu.
2. The annotation page appears.
3. The user clicks “Click for multimedia player” link.
4. A file selection dialog appears.
5. The user selects the multimedia to play.

   File selection for multimedia player is shown on Figure 19.

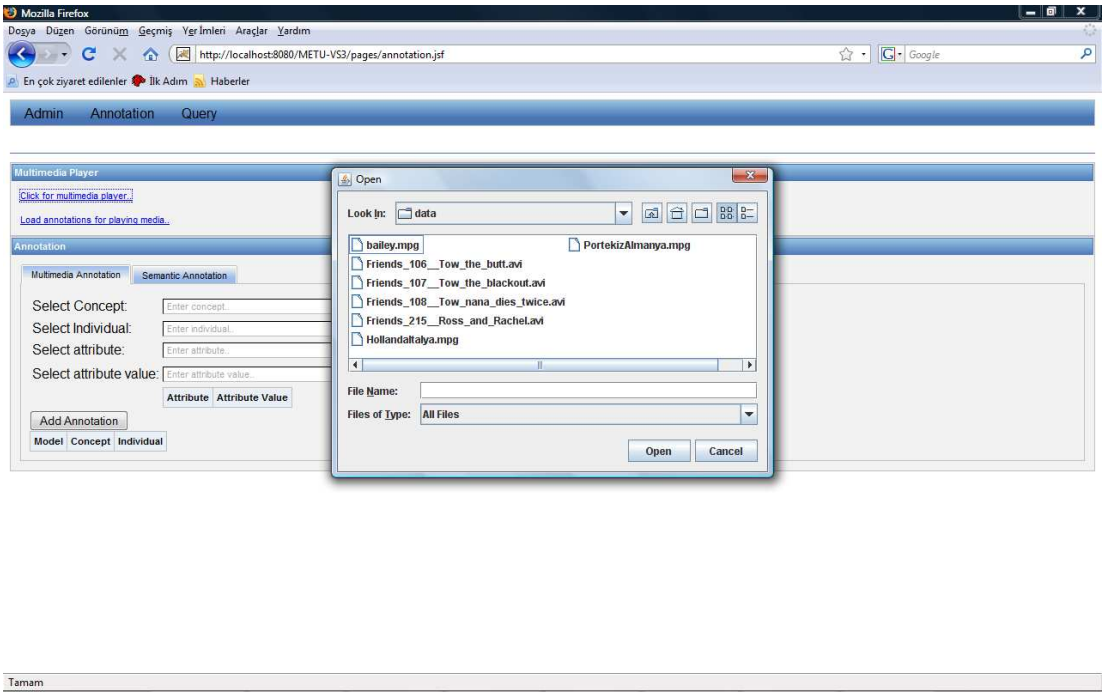

Figure 19 File Selection for Multimedia Player in Annotation Page

6. The JMF Player applet opens and plays the selected video.
7. The user clicks “Set as start frame” button at starting frame and “Set as end frame” button at ending frame to describe the interval for annotation.
   The multimedia player is shown in Figure 20.
8. The user clicks “Load annotations for playing media” link to load previous annotations for the playing video.
9. The user selects semantic annotation tab.
10. The user selects the model from the model combobox and clicks “Load” button.
11. The concept list is loaded with the annotation concepts of the selected model and “Loaded” message appears.
12. The user selects a concept from the concept list and clicks “Load” button.

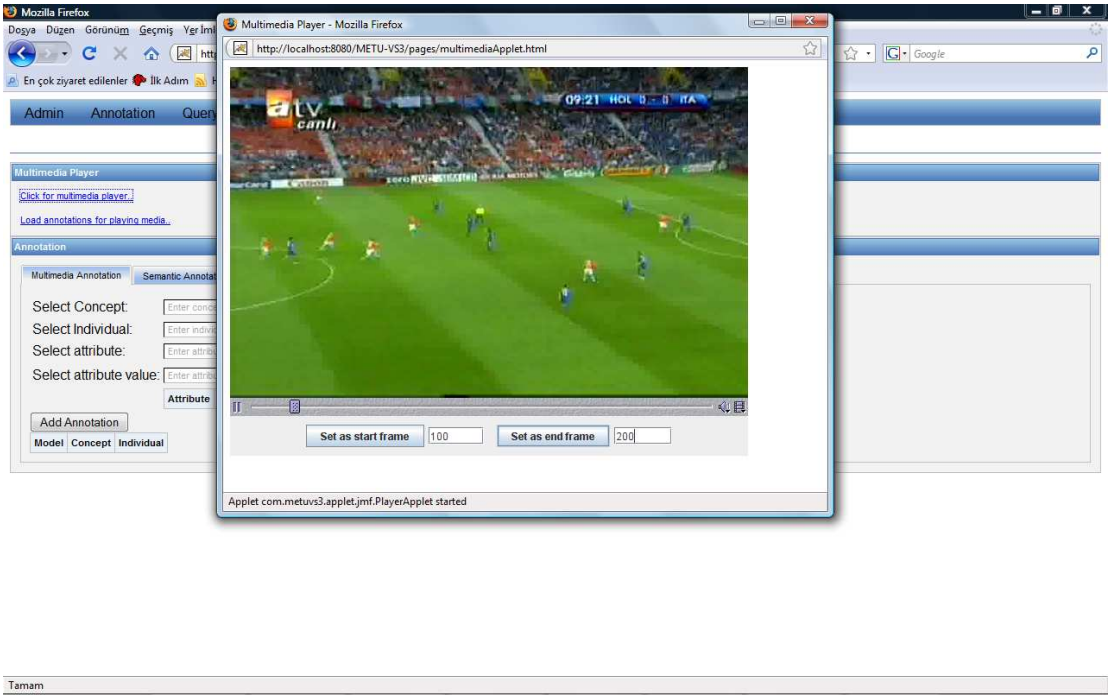

Figure 20 The Multimedia Player in Annotation page

1. The individual and attribute lists are loaded with the corresponding values of the selected concept and “Loaded” message appears.
2. The user enters a new individual or selects an existing individual from individual combobox.
3. The user selects an attribute to set value and clicks “Load” button.
4. The individuals for the selected attribute are loaded to attribute value combobox.
5. The user enters a new individual or selects an existing individual for attribute value and clicks “Add Attribute” button.
6. The attribute-value pair is added to attribute list.
7. The user repeats steps 15-17 as required.
8. The user clicks “Add Annotation” button to add annotation for the selected video segment.
9. The annotation is added to the annotation list.

10. The user repeats the process as required.

The annotation page is shown in Figure 22.

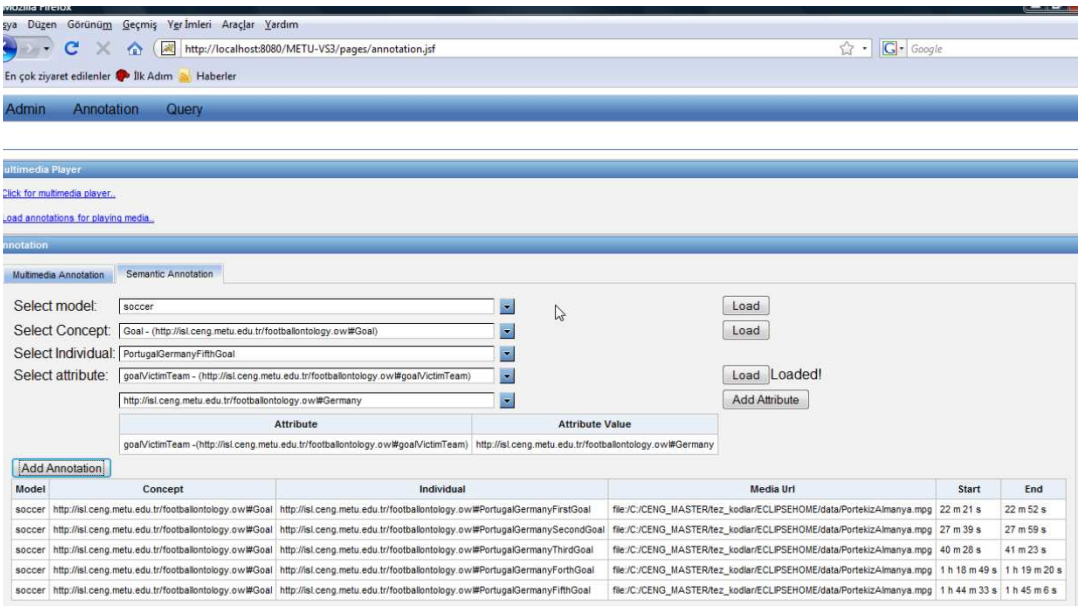

Figure 21 The Annotation Page

5.3 Query

The usage of the query page is similar to the annotation page; both are ontology-driven. Query module converts the query to SPARQL prior to processing.

1. The user selets Query → Query from system menu.
2. The user repeats steps 10-19 in Annotation use case to define a semantic query item.
3. The user uses the query toolkit in order to form a query.

4. The query is displayed in Query Display panel.
5. The user clicks “EXECUTE” button when the query is ready.
6. The query results are listed in Query Results panel.
7. The user clicks the “play” link next to a query result in order to play the video segment.

The query page is shown in Figure 23.

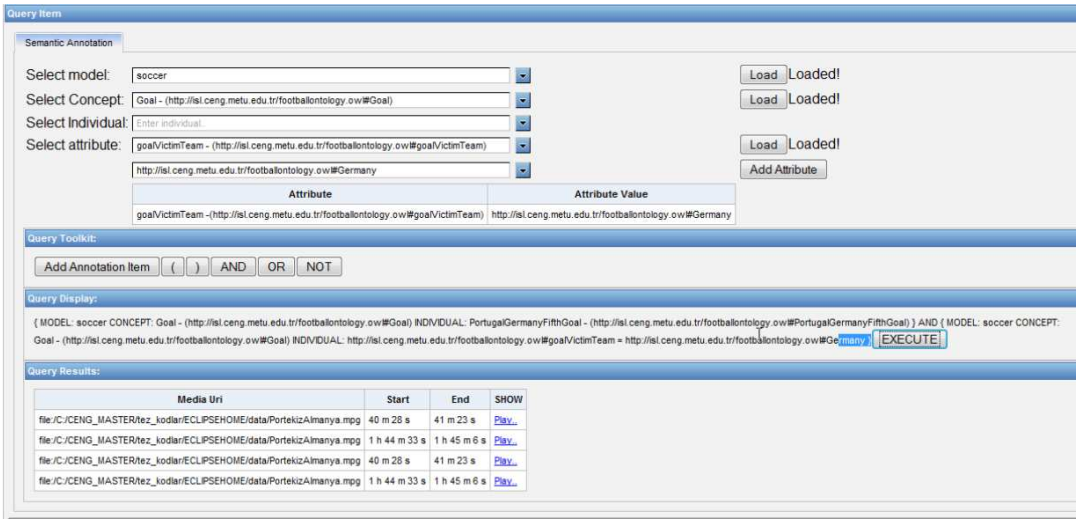

Figure 22 The Query Page

CHAPTER 6

CONCLUSION

The ontological multimedia information management framework can be easily used in some specific application domain naturally. For example, a user who wants to use the system in soccer domain can achieve this just by feeding the program with soccer-domain ontology. Moreover, the system can easily be modified according to domain-specific requirements due to its modular architecture. For example, if the user wants to specialize the interface according to a specific domain, it is possible since the implementation is as modular as possible. [1]

In this thesis, we focused on binding MPEG-7 ontology and domain-specific ontologies in a standard and automated way, the general annotation and querying features. We do not consider automating the annotation process itself. As a future work, automatic or semi-automatic annotation feature may be added to the system. The architecture of the system is designed to allow this improvement. Furthermore, a domain-specific multimedia content management system can be put on top of the proposed system. [1]

Our proposed system is an ontological multimedia information management framework with a modular architecture, ease of integrating with user-defined ontologies naturally and automatic harmonization of MPEG-7 ontology and domain-specific ontologies. The system makes use of existing user-defined ontologies in multimedia content annotation. Therefore, the system delegates the

querying power of the multimedia management system from the system to the user-defined ontology. [1]

The system opens a door for many future works due to its general structure. One of the aims of this work is to give a starting point to developers who want to develop a multimedia information management system in a specific domain. With this system in their hand, they do not have to worry about MPEG-7 details, ontology API details and interface details. [1]

REFERENCES

[1] H. Tarakçı, N. K. Çiçekli, “Ontological Multimedia Information Management System”, In the Preceedings of e-Challenges 2008, Sweden, October 2008.

[2] Çigdem Çelik, “An MPEG-7 Database System for Content-Based Management and Retrieval”, MSc Thesis, Graduate School of Natural and Applied Sciences of Middle East Technical University, September 2005.

[3] M. Köprülü, N. K. Çiçekli, A. Yazıcı, "Spatio-temporal Querying in Video Databases", Information Sciences 160, 2004, Elsevier Science, pp. 131-152, 2004.

[4] C. Tsinaraki, “Ontology-Driven Interoperability for MPEG-7”, In the Procedings of DELOS Conference, 2007.

[5] C. Tsinaraki, P. Polydoros, S. Christodoulakis S, "Interoperability support between MPEG-7/21 and OWL in DS-MIRF", In Transactions on Knowledge and Data Engineering (IEEE-TKDE), Special Issue on the Semantic Web Era, 2007.

[6] C. Tsinaraki , P. Polydoros , S. Christodoulakis , "Interoperability support for Ontology-based Video Retrieval Applications", In the Proceedings of CIVR 2004, Dublin/Ireland, July 2004

[7] C. Tsinaraki, E. Fatourou, S. Christodoulakis, “An ontology-driven framework for the management of semantic metadata describing audiovisual information,” In the Proceedings of CAiSE 2003, Velden, Austria, pp. 340–356, 2003.

[8] C. Tsinaraki, P. Polydoros, F. Kazasis, S. Christodoulakis, “Ontology-Based Semantic Indexing for MPEG-7 and TV-Anytime Audiovisual Content”, Multimedia Tools Appl. 26, 3 , 299-325, Aug.2005

[9] C. Tsinaraki, P. Polydoros, N. Moumoutzis, S. Christodoulakis, “Coupling Owl with MPEG-7 and TVAnytime for Domain-specific Multimedia Information Integration and Retrieval”, In the Proceedings of RIAO 2004, Avignon/ France, April 2004.

[10] C. Tsinaraki, P. Polydoros, S. Christodoulakis, “Integration of OWL Ontologies in MPEG-7 and TVAnytime Compliant Semantic Indexing”, In the Proceedings of CaiSE, 2004.

[11] C. Tsinaraki, S. Christodoulakis, "XS2OWL: A Formal Model and a System for enabling XML Schema Applications to interoperate with OWL-DL Domain Knowledge and Semantic Web Tools", DELOS Conference, Tirrenia, Italy, March 2007.

[12] P. Polydoros, C. Tsinaraki, S. Christodoulakis, "GraphOnto: OWL-Based Ontology Management and Multimedia Annotation in the DS-MIRF Framework", In the proceedings of the Workshop on Multimedia Semantics 2006 (WMS 2006), pp. 14-24, 19-21 June 2006, Chania, Crete, 2006.

[13] C. Tsinaraki, P. Polydoros, S. Christodoulakis, "GraphOnto: A Component and a User Interface for the Definition and Use of Ontologies in Multimedia Information Systems", In the Proceedings of AVIVDiLib 2005, Cortona, Italy, April 2005

[14] C. Tsinaraki, S. Papadomanolakis, S. Christodoulakis S, "Towards a two - layered Video Metadata Model", DLib Workshop 2001 (in conjunction with DEXA '01), September 2001.

[15] R. García, C. Tsinaraki, O. Celma and S. Christodoulakis, “Multimedia Content Description using Semantic Web Languages”, In Y. Kompatsiaris, P. Hobson (Eds) “Semantic Multimedia and Ontologies: Theory and Application”, pp. 17-34, Springer, 2008.

[16] R. García, O. Celma, “Semantic Integration and Retrieval of Multimedia Metadata”, In the Proceedings of the Knowledge Markup and Semantic Annotation Workshop, Semannot'05, 2005.

[17] J. Hunter , “Adding Multimedia to the Semantic Web - Building an MPEG-7 Ontology”. In International Semantic Web Working Symposium (SWWS 2001), Stanford University, California, USA, July 30 - August 1, 2001.

[18] J. Hunter, S. Little, “A framework to enable the semantic inferencing and querying of multimedia content”. Int. J. Web Eng. Technol. 2(2/3): 264-286 , 2005.

[19] R. Arndt, R. Troncy, S. Staab, L. Hardman, M. Vacura, “COMM: designing a well-founded multimedia ontology for the web”, In the Proceedings of the 6th International Semantic Web Conference (ISWC'2007), 2007.

[20] T. Athanasiadis, V. Tzouvaras, K. Petridis, F. Precioso, Y. Avrithis, Y. Kompatsiaris, “Using a Multimedia Ontology Infrastructure for Semantic

Annotation of Multimedia Content“, In the Proceedings of 5th International Workshop on Knowledge Markup and Semantic Annotation (SemAnnot '05) at the 4th International Semantic Web Conference (ISWC 2005), 2005.

[21] Md. Anisur Rahman, Anwar Hossain, Iluju Kiringa, Abdulmotaleb El Saddik , "Towards an ontology for MPEG-7 Semantic Descriptions", Proc. Intelligent Interactive Learning Object Repositories (I2LOR) Conference, Montreal, QC, Canada, November 2006.

[22] N. Guarino, “Formal Ontology in Information Systems”, In the Proceedings of FOIS’98, Trento, Italy, 6-8 June 1998. Amsterdam, IOS Press, pp. 3-15, 1998.

[23] P. Salembier and J. R. Smith, “MPEG-7 Multimedia Description Schemes”, IEEE Trans. Circuits Syst. for Video Technol., August 2001.

[24] M.E. Donderler, E. Saykol, U. Arslan, O. Ulusoy, U. Gudukbay, BilVideo: Design and Implementation of a Video Database Management System, Multimedia Tools and Applications, Vol. 27, pp. 79-104, September 2005.

[25] U. Arslan, M.E. Donderler, E. Saykol, O Ulusoy, U. Gudukbay, A Semi-Automatic Semantic Annotation Tool for Video Databases, In SOFSEM 2002, Workshop on Multimedia Semantics, Milovy, Czech Republic, November 2002.

[26] M.E. Donderler, E. Saykol, O. Ulusoy, U. Gudukbay, A Video Database Management System, In Proceedings of Hermes Project Workshop, Middle East Technical University, Ankara, Turkey, June 2002.

[27] M. E. Donderler, O. Ulusoy, U. Gudukbay, A Rule-based Approach to Represent Spatio-Temporal Relations in Video Data, International Conference on Advances in Information Systems (ADVIS'2000), Izmir, Turkey, Lecture Notes in Computer Science (Springer-Verlag), Vol.1909, pp. 409-418, October 2000.

[28] E. Saykol, “Web-based user interface for query specification in a video database system,” M.S. thesis, Department of Computer Engineering, Bilkent University, Ankara, Turkey, September 2001.

[29] S. Bloehdorn, K. Petridis, C. Saathoff, N. Simou, V. Tzouvaras, Y. Avrithis, S. Handschuh, I. Kompatsiaris, S. Staab, M.G. Strintzis. “Semantic Annotation of Images and Videos for Multimedia Analysis”. In Proceedings of the 2nd European Semantic Web Conference (ESWC 2005), May 2005.

[30] J. M. Martinez, “MPEG-7 Overview Version 10”, http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm, 2008.

[31] W3C Incubator Group, MPEG-7 and the Semantic Web, http://www.w3.org/2005/Incubator/mmsem/XGR-mpeg7/, August 2007

[32] W3C Incubator Group ,W3C Semantic Web Activity, http//www.w3.org/2001/sw/, 2008.

[33] W3C Semantic Web Activity, Resource Description Framework(RDF), http//www.w3.org/RDF/, 2008.

[34] W3C Candidate Recommendation ,RDF Schema Specification 1.0, http://www.w3.org/TR/rdf-schema/, February 2004.

[35] D.McGuiness, F. Van Harmelen, "OWL web ontology language overview," W3C Candidate Recommendation, http://www.w3.org/TR/owl-features/, 2003.

[36] Bilkent University, BilVideo project, http://pcvideo.cs.bilkent.edu.tr, 2008.

[37] Motorola , AceMedia project, http://www.acemedia.org/, 2008.

[38] Rhizomik, Rhizomik home page, http://rhizomik.net/, 2008.

[39] Rhizomik, DeReFer project, http://rhizomik.net/ content/ redefer/, 2008

[40] Rhizomik, MPEG-7 OWL Ontologies, Mpeg-7Ontos project, http://rhizomik.net/ content/ mpeg7ontos, 2008.

[41] COMM - A Core Ontology for Multimedia web page, http://comm.semanticweb.org/, 2008

APPENDIX A

THE INTERMEDIATE ONTOLOGY

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#" >

  <rdf:Description
rdf:about="http://www.metuvs3.com/video/ontology#VS3MediaLocatorType">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://rhizomik.net/ontologies/2005/03/Mpeg7-
2001.owl#TemporalSegmentLocatorType"/>
  </rdf:Description>

  <rdf:Description
rdf:about="http://www.metuvs3.com/video/ontology#VS3SemanticTimeType">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf
rdf:resource="http://www.metuvs3.com/video/ontology#VS3SemanticTimeType"/>
    <rdfs:subClassOf rdf:resource="http://rhizomik.net/ontologies/2005/03/Mpeg7-
2001.owl#SemanticTimeType"/>
  </rdf:Description>

  <rdf:Description rdf:about="http://www.metuvs3.com/video/ontology#VS3ConceptType">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf
rdf:resource="http://www.metuvs3.com/video/ontology#VS3SemanticBaseType"/>
    <rdfs:subClassOf rdf:resource="http://rhizomik.net/ontologies/2005/03/Mpeg7-
2001.owl#ConceptType"/>
  </rdf:Description>

  <rdf:Description
rdf:about="http://www.metuvs3.com/video/ontology#VS3SemanticStateType">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf
rdf:resource="http://www.metuvs3.com/video/ontology#VS3SemanticBaseType"/>
    <rdfs:subClassOf rdf:resource="http://rhizomik.net/ontologies/2005/03/Mpeg7-
2001.owl#SemanticStateType"/>
  </rdf:Description>
```

```
  <rdf:Description
rdf:about="http://www.metuvs3.com/video/ontology#VS3AgentObjectType">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf
rdf:resource="http://www.metuvs3.com/video/ontology#VS3ObjectType"/>
    <rdfs:subClassOf rdf:resource="http://rhizomik.net/ontologies/2005/03/Mpeg7-
2001.owl#AgentObjectType"/>
  </rdf:Description>

  <rdf:Description
rdf:about="http://www.metuvs3.com/video/ontology#VS3SemanticPlaceType">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf
rdf:resource="http://www.metuvs3.com/video/ontology#VS3SemanticBaseType"/>
    <rdfs:subClassOf rdf:resource="http://rhizomik.net/ontologies/2005/03/Mpeg7-
2001.owl#SemanticPlaceType"/>
  </rdf:Description>

  <rdf:Description
rdf:about="http://www.metuvs3.com/video/ontology#VS3SemanticBaseType">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf rdf:resource="http://rhizomik.net/ontologies/2005/03/Mpeg7-
2001.owl#SemanticBaseType"/>
  </rdf:Description>

  <rdf:Description
rdf:about="http://www.metuvs3.com/video/ontology#VS3OrganizationType">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf
rdf:resource="http://www.metuvs3.com/video/ontology#VS3AgentObjectType"/>
    <rdfs:subClassOf rdf:resource="http://rhizomik.net/ontologies/2005/03/Mpeg7-
2001.owl#OrganizationType"/>
  </rdf:Description>

  <rdf:Description
rdf:about="http://www.metuvs3.com/video/ontology#VS3PersonGroupType">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf
rdf:resource="http://www.metuvs3.com/video/ontology#VS3AgentObjectType"/>
    <rdfs:subClassOf rdf:resource="http://rhizomik.net/ontologies/2005/03/Mpeg7-
2001.owl#PersonGroupType"/>
  </rdf:Description>

  <rdf:Description rdf:about="http://www.metuvs3.com/video/ontology#VS3PersonType">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf
rdf:resource="http://www.metuvs3.com/video/ontology#VS3AgentObjectType"/>
    <rdfs:subClassOf rdf:resource="http://rhizomik.net/ontologies/2005/03/Mpeg7-
2001.owl#PersonType"/>
  </rdf:Description>

  <rdf:Description rdf:about="http://www.metuvs3.com/video/ontology#VS3ObjectType">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf
rdf:resource="http://www.metuvs3.com/video/ontology#VS3SemanticBaseType"/>
```

```
    <rdfs:subClassOf rdf:resource="http://rhizomik.net/ontologies/2005/03/Mpeg7-
2001.owl#ObjectType"/>
  </rdf:Description>

  <rdf:Description rdf:about="http://www.metuvs3.com/video/ontology#VS3EventType">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf
rdf:resource="http://www.metuvs3.com/video/ontology#VS3SemanticBaseType"/>
    <rdfs:subClassOf rdf:resource="http://rhizomik.net/ontologies/2005/03/Mpeg7-
2001.owl#EventType"/>
  </rdf:Description>
</rdf:RDF>
```

APPENDIX B

THE SOCCER ONTOLOGY

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns="http://isl.ceng.metu.edu.tr/footballontology.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
  xml:base="http://isl.ceng.metu.edu.tr/footballontology.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="FirstHalfEnd">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Whistle"/>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty>
          <owl:ObjectProperty rdf:ID="firstHalfEndWhistle"/>
        </owl:onProperty>
        <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >1</owl:cardinality>
      </owl:Restriction>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Illegal">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="GameActions"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="RefereeActions">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Actions"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="GoalKeeperSave">
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty>
          <owl:ObjectProperty rdf:ID="saverPlayer"/>
```

```
      </owl:onProperty>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Class rdf:ID="Deffensive_Actions"/>
  </rdfs:subClassOf>
</owl:Class>
.....
<owl:Class rdf:ID="Teams">
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="hasCoach"/>
      </owl:onProperty>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="PassAndRun">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Pass"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Kick_Off">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Set_Play"/>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="kickOffTeam"/>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="VisitingTeam">
  <rdfs:subClassOf rdf:resource="#Teams"/>
</owl:Class>
<owl:Class rdf:ID="Injury">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:maxCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:maxCardinality>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="injuryCommitter"/>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
```

```
        <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >1</owl:cardinality>
        <owl:onProperty>
          <owl:ObjectProperty rdf:ID="injuredPerson"/>
        </owl:onProperty>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Class rdf:about="#Legal"/>
    </rdfs:subClassOf>
  </owl:Class>
.....
  <owl:Class rdf:about="#Shot">
    <rdfs:subClassOf rdf:resource="#BallUsage"/>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty>
          <owl:ObjectProperty rdf:ID="shotPerson"/>
        </owl:onProperty>
        <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >1</owl:cardinality>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf rdf:resource="#Offensive_Actions"/>
  </owl:Class>
  <owl:Class rdf:ID="Out_Shot">
    <rdfs:subClassOf rdf:resource="#Shot"/>
  </owl:Class>
  <owl:Class rdf:ID="Clear">
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >1</owl:cardinality>
        <owl:onProperty>
          <owl:ObjectProperty rdf:ID="clearerPlayer"/>
        </owl:onProperty>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Class rdf:about="#Deffensive_Actions"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:about="#FourthOfficialAction">
    <rdfs:subClassOf rdf:resource="#RefereeActions"/>
  </owl:Class>
  <owl:Class rdf:ID="Foul">
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty>
          <owl:ObjectProperty rdf:ID="foulCommitter"/>
        </owl:onProperty>
        <owl:maxCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >1</owl:maxCardinality>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
```

```
    <owl:Restriction>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="foulVictim"/>
      </owl:onProperty>
      <owl:maxCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:maxCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="#Illegal"/>
</owl:Class>
<owl:Class rdf:ID="Corner_Kick">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="cornerKickPerson"/>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Set_Play"/>
  </rdfs:subClassOf>
</owl:Class>
.....
<owl:Class rdf:ID="Stadium">
  <rdfs:subClassOf rdf:resource="#Places"/>
</owl:Class>
<owl:Class rdf:ID="PassToGoalArea">
  <rdfs:subClassOf rdf:resource="#Pass"/>
</owl:Class>
<owl:Class rdf:ID="BlockedShot">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:ObjectProperty rdf:about="#blockingTeam"/>
      </owl:onProperty>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="#Shot"/>
</owl:Class>
<owl:Class rdf:about="#Set_Play">
  <rdfs:subClassOf rdf:resource="#Legal"/>
</owl:Class>
<owl:Class rdf:ID="Assist">
  <rdfs:subClassOf rdf:resource="#Pass"/>
</owl:Class>
<owl:Class rdf:ID="Yellow_Card">
  <rdfs:subClassOf>
    <owl:Class rdf:ID="Card"/>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
```

```
      >1</owl:cardinality>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="yellowCardPlayer"/>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="LeftAssistantReferee">
  <rdfs:subClassOf rdf:resource="#Referees"/>
</owl:Class>
<owl:Class rdf:ID="MatchEnd">
  <rdfs:subClassOf rdf:resource="#Whistle"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="gameEndWhistle"/>
      </owl:onProperty>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Red_Card">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Card"/>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="redCardPlayer"/>
      </owl:onProperty>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Free_Kick">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="freeKickUser"/>
      </owl:onProperty>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="#Set_Play"/>
</owl:Class>
<owl:Class rdf:about="#Actions">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:maxCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:maxCardinality>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="hasPitchPlace"/>
      </owl:onProperty>
```

```
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="hasMatch"/>
      </owl:onProperty>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:FunctionalProperty rdf:ID="minute"/>
      </owl:onProperty>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#Card">
  <rdfs:subClassOf rdf:resource="#PrincipalRefereeAction"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="cardReferee"/>
      </owl:onProperty>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Off_Side">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="offSidePerson"/>
      </owl:onProperty>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="#Illegal"/>
</owl:Class>
<owl:Class rdf:ID="Counter_Attack">
  <rdfs:subClassOf rdf:resource="#Offensive_Actions"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="counterAttackerTeam"/>
      </owl:onProperty>
    </owl:Restriction>
```

```
  </rdfs:subClassOf>
 </owl:Class>
 <owl:Class rdf:ID="RightAssistantReferee">
  <rdfs:subClassOf rdf:resource="#Referees"/>
 </owl:Class>
 <owl:Class rdf:about="#TeamMembers">
  <rdfs:subClassOf rdf:resource="#People"/>
 </owl:Class>
 <owl:Class rdf:ID="Kick_to_Throw_In">
  <rdfs:subClassOf rdf:resource="#Out"/>
 </owl:Class>
 <owl:Class rdf:ID="Shot_On_Goal">
  <rdfs:subClassOf>
   <owl:Restriction>
    <owl:onProperty>
     <owl:ObjectProperty rdf:about="#shotPerson"/>
    </owl:onProperty>
    <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >1</owl:cardinality>
   </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="#Shot"/>
 </owl:Class>
 <owl:Class rdf:about="#GameActions">
  <rdfs:subClassOf rdf:resource="#Actions"/>
 </owl:Class>
 <owl:Class rdf:ID="Goal">
  <rdfs:subClassOf>
   <owl:Restriction>
    <owl:onProperty>
     <owl:ObjectProperty rdf:ID="goalVictimTeam"/>
    </owl:onProperty>
    <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >1</owl:cardinality>
   </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="#Shot"/>
 </owl:Class>
.....
</rdf:RDF>

<!-- Created with Protege (with OWL Plugin 3.3.1, Build 430)  http://protege.stanford.edu --
>
```

APPENDIX C

THE ANIMAL ONTOLOGY

```
<?xml version="1.0"?>
<rdf:RDF xmlns="http://nlp.shef.ac.uk/abraxas/ontologies/animals.owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:daml="http://www.daml.org/2001/03/daml+oil#" xmlns:p1="http://www.owl-
ontologies.com/assert.owl#" xmlns:dc="http://purl.org/dc/elements/1.1/"
xml:base="http://nlp.shef.ac.uk/abraxas/ontologies/animals.owl">
  <owl:Ontology rdf:about="">
    <dc:creator rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Christopher
Brewster (C.Brewster@dcs.shef.ac.uk)</dc:creator>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string">A simple
ontology (more properly a taxonomy or folksonomy) of animals encountered in English
children's books</rdfs:comment>
    <dc:date rdf:datatype="http://www.w3.org/2001/XMLSchema#string">April
2006</dc:date>
    <owl:imports rdf:resource="http://purl.org/dc/elements/1.1/"/>
  </owl:Ontology>
  <owl:Class rdf:ID="Bee">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Insect"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Blackbird">
    <owl:disjointWith>
      <owl:Class rdf:ID="Owl"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Crow"/>
    </owl:disjointWith>
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Bird"/>
    </rdfs:subClassOf>
    <owl:disjointWith>
      <owl:Class rdf:ID="Ostrich"/>
    </owl:disjointWith>
```

```
    <owl:disjointWith>
      <owl:Class rdf:ID="Woodpecker"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Nightingale"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Robin"/>
    </owl:disjointWith>
  </owl:Class>
  <owl:Class rdf:about="#Woodpecker">
    <owl:disjointWith>
      <owl:Class rdf:about="#Robin"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:about="#Nightingale"/>
    </owl:disjointWith>
    <owl:disjointWith rdf:resource="#Blackbird"/>
    <owl:disjointWith>
      <owl:Class rdf:about="#Owl"/>
    </owl:disjointWith>
    <rdfs:subClassOf>
      <owl:Class rdf:about="#Bird"/>
    </rdfs:subClassOf>
    <owl:disjointWith>
      <owl:Class rdf:about="#Ostrich"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:about="#Crow"/>
    </owl:disjointWith>
  </owl:Class>
  <owl:Class rdf:ID="Fox">
    <owl:disjointWith>
      <owl:Class rdf:ID="Antelope"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Hedgehog"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Hare"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Mouse"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Camel"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Badger"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Otter"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Dog"/>
    </owl:disjointWith>
```

```
    <owl:disjointWith>
      <owl:Class rdf:ID="Squirrel"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Cow"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Tiger"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Hippopotamus"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Cheetah"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Weasel"/>
    </owl:disjointWith>
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Mammals"/>
    </rdfs:subClassOf>
    <owl:disjointWith>
      <owl:Class rdf:ID="Mole"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Rhinoceros"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Bear"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Rat"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Cat"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Lion"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Monkey"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Giraffe"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Sloth"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="Rabbit"/>
    </owl:disjointWith>
  </owl:Class>
.....
  <owl:Class rdf:about="#Mammals">
    <owl:disjointWith rdf:resource="#Bird"/>
    <owl:disjointWith>
```

```
    <owl:Class rdf:about="#Marsupial"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Dragon"/>
  </owl:disjointWith>
  <rdfs:subClassOf rdf:resource="#Animals"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Reptile"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Unicorn"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Amphibian"/>
  </owl:disjointWith>
</owl:Class>
<owl:Class rdf:ID="Frog">
  <owl:disjointWith>
    <owl:Class rdf:ID="Toad"/>
  </owl:disjointWith>
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Amphibian"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#Otter">
  <owl:disjointWith>
    <owl:Class rdf:about="#Dog"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Hare"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Bear"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Monkey"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Weasel"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Lion"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Sloth"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#Fox"/>
  <owl:disjointWith rdf:resource="#Camel"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Cheetah"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#Giraffe"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Rat"/>
  </owl:disjointWith>
  <owl:disjointWith>
```

```
    <owl:Class rdf:about="#Hedgehog"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Antelope"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#Hippopotamus"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Mouse"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#Rhinoceros"/>
  <owl:disjointWith rdf:resource="#Mole"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Cat"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Cow"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Badger"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Squirrel"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Rabbit"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Tiger"/>
  </owl:disjointWith>
  <rdfs:subClassOf rdf:resource="#Mammals"/>
</owl:Class>
<owl:Class rdf:about="#Ostrich">
  <owl:disjointWith rdf:resource="#Woodpecker"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Owl"/>
  </owl:disjointWith>
  <rdfs:subClassOf rdf:resource="#Bird"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Robin"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#Nightingale"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Crow"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#Blackbird"/>
</owl:Class>
<owl:Class rdf:ID="Kangaroo">
  <owl:disjointWith>
    <owl:Class rdf:ID="Koala"/>
  </owl:disjointWith>
  <rdfs:subClassOf>
    <owl:Class rdf:about="#Marsupial"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#Robin">
  <owl:disjointWith rdf:resource="#Blackbird"/>
```

```
  <owl:disjointWith rdf:resource="#Ostrich"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Crow"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#Nightingale"/>
  <owl:disjointWith rdf:resource="#Woodpecker"/>
  <rdfs:subClassOf rdf:resource="#Bird"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Owl"/>
  </owl:disjointWith>
</owl:Class>
<owl:Class rdf:about="#Badger">
  <owl:disjointWith>
    <owl:Class rdf:about="#Bear"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Weasel"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Rat"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#Rhinoceros"/>
  <rdfs:subClassOf rdf:resource="#Mammals"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Cow"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Monkey"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Cheetah"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Rabbit"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Squirrel"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#Mole"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Sloth"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Dog"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Hedgehog"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#Otter"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Cat"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Tiger"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#Hippopotamus"/>
```

```
  <owl:disjointWith rdf:resource="#Camel"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Hare"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#Fox"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Mouse"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Lion"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Antelope"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#Giraffe"/>
</owl:Class>
<owl:Class rdf:about="#Tiger">
  <owl:disjointWith>
    <owl:Class rdf:about="#Rat"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Mouse"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Sloth"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Cat"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Cheetah"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Monkey"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Hedgehog"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Bear"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Rabbit"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Dog"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#Badger"/>
  <owl:disjointWith rdf:resource="#Fox"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Lion"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#Rhinoceros"/>
  <owl:disjointWith rdf:resource="#Hippopotamus"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Hare"/>
```

```
    </owl:disjointWith>
    <owl:disjointWith rdf:resource="#Giraffe"/>
    <owl:disjointWith rdf:resource="#Mole"/>
    <owl:disjointWith>
      <owl:Class rdf:about="#Antelope"/>
    </owl:disjointWith>
    <rdfs:subClassOf rdf:resource="#Mammals"/>
    <owl:disjointWith rdf:resource="#Otter"/>
    <owl:disjointWith>
      <owl:Class rdf:about="#Squirrel"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:about="#Weasel"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:about="#Cow"/>
    </owl:disjointWith>
    <owl:disjointWith rdf:resource="#Camel"/>
  </owl:Class>
.....
  <owl:Class rdf:about="#Toad">
    <owl:disjointWith rdf:resource="#Frog"/>
    <rdfs:subClassOf>
      <owl:Class rdf:about="#Amphibian"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:about="#Hare">
    <owl:disjointWith rdf:resource="#Sloth"/>
    <owl:disjointWith rdf:resource="#Squirrel"/>
    <owl:disjointWith rdf:resource="#Cat"/>
    <owl:disjointWith rdf:resource="#Lion"/>
    <owl:disjointWith rdf:resource="#Hippopotamus"/>
    <owl:disjointWith rdf:resource="#Mouse"/>
    <owl:disjointWith rdf:resource="#Rhinoceros"/>
    <owl:disjointWith rdf:resource="#Tiger"/>
    <owl:disjointWith rdf:resource="#Otter"/>
    <owl:disjointWith rdf:resource="#Cow"/>
    <owl:disjointWith rdf:resource="#Hedgehog"/>
    <owl:disjointWith rdf:resource="#Cheetah"/>
    <owl:disjointWith rdf:resource="#Dog"/>
    <owl:disjointWith rdf:resource="#Badger"/>
    <owl:disjointWith rdf:resource="#Weasel"/>
    <owl:disjointWith rdf:resource="#Rabbit"/>
    <owl:disjointWith rdf:resource="#Bear"/>
    <rdfs:subClassOf rdf:resource="#Mammals"/>
    <owl:disjointWith rdf:resource="#Rat"/>
    <owl:disjointWith rdf:resource="#Antelope"/>
    <owl:disjointWith rdf:resource="#Giraffe"/>
    <owl:disjointWith rdf:resource="#Monkey"/>
    <owl:disjointWith rdf:resource="#Mole"/>
    <owl:disjointWith rdf:resource="#Camel"/>
    <owl:disjointWith rdf:resource="#Fox"/>
  </owl:Class>
  <owl:Class rdf:about="#Marsupial">
    <owl:disjointWith>
      <owl:Class rdf:about="#Dragon"/>
```

```
    </owl:disjointWith>
    <owl:disjointWith rdf:resource="#Unicorn"/>
    <rdfs:subClassOf rdf:resource="#Animals"/>
    <owl:disjointWith rdf:resource="#Mammals"/>
    <owl:disjointWith rdf:resource="#Reptile"/>
    <owl:disjointWith rdf:resource="#Bird"/>
    <owl:disjointWith>
      <owl:Class rdf:about="#Amphibian"/>
    </owl:disjointWith>
  </owl:Class>
  <owl:Class rdf:about="#Dragon">
    <owl:disjointWith rdf:resource="#Mammals"/>
    <owl:disjointWith rdf:resource="#Bird"/>
    <rdfs:subClassOf rdf:resource="#Animals"/>
    <owl:disjointWith rdf:resource="#Marsupial"/>
    <owl:disjointWith rdf:resource="#Unicorn"/>
    <owl:disjointWith rdf:resource="#Reptile"/>
    <owl:disjointWith>
      <owl:Class rdf:about="#Amphibian"/>
    </owl:disjointWith>
  </owl:Class>
  <owl:Class rdf:about="#Insect">
    <rdfs:subClassOf rdf:resource="#Animals"/>
  </owl:Class>
  <owl:Class rdf:ID="Spider">
    <rdfs:subClassOf rdf:resource="#Animals"/>
  </owl:Class>
  <owl:Class rdf:about="#Amphibian">
    <owl:disjointWith rdf:resource="#Mammals"/>
    <owl:disjointWith rdf:resource="#Reptile"/>
    <owl:disjointWith rdf:resource="#Marsupial"/>
    <owl:disjointWith rdf:resource="#Bird"/>
    <owl:disjointWith rdf:resource="#Unicorn"/>
    <owl:disjointWith rdf:resource="#Dragon"/>
    <rdfs:subClassOf rdf:resource="#Animals"/>
  </owl:Class>
</rdf:RDF>
<!-- Created with Protege (with OWL Plugin 2.2, Build 311)  http://protege.stanford.edu -->
```