AN ONTOLOGY-DRIVEN VIDEO ANNOTATION AND RETRIEVAL SYSTEM

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

BY

GONCAGÜL DEMİRDİZEN

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

SEPTEMBER 2010

Approval of the thesis:

AN ONTOLOGY-DRIVEN VIDEO ANNOTATION AND RETRIEVAL SYSTEM

submitted by **GONCAGÜL DEMİRDİZEN** 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. Adnan Yazıcı 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	_____________________
Asst. Prof. Dr. İlkay Ulusoy Electrical and Electronics Engineering Dept., METU	_____________________

Date: 17.09.2010

ABSTRACT

AN ONTOLOGY-DRIVEN VIDEO ANNOTATION AND RETRIEVAL SYSTEM

Demirdizen, Goncagül
M.Sc., Department of Computer Engineering
Supervisor: Assoc. Prof. Dr. Nihan Kesim Çiçekli

September 2010, 109 pages

In this thesis, a system, called Ontology-Driven Video Annotation and Retrieval System (OntoVARS) is developed in order to provide a video management system which is used for ontology-driven semantic content annotation and querying. The proposed system is based on MPEG-7 ontology which provides interoperability and common communication platform with other MPEG-7 ontology compatible systems. The Rhizomik MPEG-7 ontology is used as the core ontology and domain specific ontologies are integrated to the core ontology in order to provide ontology-based video content annotation and querying capabilities to the user. The proposed system supports content-based annotation and spatio-temporal data modeling in video databases by using the domain ontology concepts. Moreover, the system enables ontology-driven query formulation and processing according to the domain ontology instances and concepts. In the developed system, ontology-driven concept querying, spatio-temporal querying, region-based and time-based querying capabilities are performed as simple querying types. Besides these simple query types, compound queries are also generated by combining simple queries with "(", ")", "AND" and "OR" operators. For all these query types, the system supports both general and video specific query processing. By this means, the user is able to pose queries on all videos in the video databases as well as the details of a specific video of interest.

Keywords: Ontology-Driven Annotation, Semantic Content-Based Querying, Spatio-Temporal Querying, MPEG-7 Ontology, Domain Ontology.

ÖZ

ONTOLOJİ TABANLI VİDEO ETİKETLEME VE ERİŞİM SİSTEMİ

Demirdizen, Goncagül
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi: Doç. Dr. Nihan Kesim Çiçekli

Eylül 2010, 109 sayfa

Bu tez kapsamında, ontoloji tabanlı, anlamsal içeriğe yönelik etiketleme ve sorgulama yapılmasına olanak sağlayan bir video yönetim sistemi sağlamak amacıyla "Ontoloji Tabanlı Video Etiketleme ve Erişim Sistemi" adında bir alt yapı geliştirilmiştir. Sunulan sistem, MPEG-7 ontolojisi tabanlıdır ve bu alt yapı sisteme, diğer MPEG-7 ontolojisi uyumlu sistemlerle ortak bir dil üzerinden haberleşebilme ve birlikte çalışabilirlik yeteneği kazandırmıştır. Rhizomik MPEG-7 ontolojisi sistemde temel ontoloji olarak kullanılır ve kullanıcıya ontoloji tabanlı video içeriği etiketleme ve sorgulama kabiliyetlerini saplayabilmek için alana özel ontolojiler temel ontolojiye entegre edilir. Sunulan sistem, video veritabanlarında, alana özel ontolojilerin kavramları kullanılarak içeriğe yönelik etiketleme ve uzay-zamansal veri modelleme işlevlerini desteklemektedir. Bunların yanı sıra, geliştirilen sistem alana özel ontolojik kavram ve nesnelere yönelik ontoloji tabanlı sorguların oluşturulması ve işlenmesine olanak sağlar. Geliştirilen sistemde, ontoloji tabanlı kavramsal sorgulama, uzay-zamansal sorgulama, bölgesel ve zaman tabanlı sorgulama kabiliyetleri basit sorgu tipleri olarak gerçekleştirilebilmektedir. Bunların yanı sıra, basit sorguların "(", ")", "AND" ve "OR" operatörleri ile birleştirilmesiyle birleşik sorgular oluşturulur. Tüm bu sorgu tipleri için, sistem, hem genel hem de belirli bir videoya özel sorgulama yapabilmeyi desteklemektedir. Bu sayede, kullanıcıya video veritabanındaki tüm videolar üzerinde arama ve veri erişimi kabiliyeti sağlanmasının yanı sıra sadece ilgilenilen belirli bir video üzerinde arama yapma kabiliyeti de sunulmuş olur.

Anahtar kelimeler: Ontoloji Tabanlı Etiketleme, Anlamsal İçeriğe Yönelik Sorgulama, Uzay-Zamansal Sorgulama, MPEG-7 Ontolojisi, Alan Ontolojisi.

To my family…

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude and appreciation to my supervisor Assoc. Prof. Dr. Nihan Kesim Çiçekli for her guidance, understanding and encouragement throughout my masters and thesis study. It was a privilege to study with such a friendly and considerate supervisor.

I would also like to thank to my thesis jury members for their valuable comments and evaluation on this thesis.

I also want to thank to Scientific and Technical Research Council of Turkey (TÜBİTAK) for its support to the project with grant number EEEAG 107E234.

I would like to present my special thanks to TÜBİTAK - BİBED for their support with grant number 2228.

I am indebted to my precious friends Hilal Karaman, Ertay Kaya, Gizem Gürcüoğlu, Enver Veli Atabek, Seda Çakıroğlu, Ali Anıl Sınacı, Duygu Ceylan, Duygu Görgün Yıldırım, Begüm Bulutcu and Bahar Pamuk for their support and encouragement throughout this thesis study. Thanks for standing by me and giving me a shoulder to lean on whenever I need.

I would also like to thank my supervisors and my colleagues from ASELSAN for their understanding and support during my academic studies.

Finally, my deepest thanks are to my family for their never ending love, patience and support throughout my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In latest years, multimedia content has become more important and replaced the traditional media in our daily lives, since it is used in a wide number of domains ranging from entertainment, news, commerce, education, etc. The recent technological advances, as well as the availability of electronic devices for multimedia content consumption, have also increased the amount of multimedia data and brought a demand for the management and access on the data at the semantic content-based level. Thus, the increase in the amount of multimedia data leads difficulty in the content based retrieval of audio-visual data and makes semantic content annotation and querying in multimedia databases an important issue.

Semantic content based retrieval is an important criterion for the evaluation of multimedia content management systems, since the retrieval success has a direct interaction with the system end users. As the user constructs more complex and detailed queries, the multimedia system is expected to present more advanced retrieval performance on multimedia content. In order to provide successful retrieval capabilities, semantic content based data modeling also becomes more of an issue in a multimedia management system.

In multimedia systems, semantic content is defined according to the application and the multimedia data type (i.e. audio, image, video). In this thesis, video data is used as the center of study and other multimedia data types are excluded from the scope of the thesis work. Video data consists of high level features including semantic information and low level features including color, shape, texture etc. In this thesis, only video high level features, namely video semantic content is studied and low level features are remained out of the scope. Video semantic content should be described and stored in a machine understandable way in order to satisfy the required advanced retrieval capabilities on video content and provide semantic content-based search capabilities to the user. Semantic descriptions for the audio-visual content is stored in semantic descriptors and these 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 [10][20][38]. In order to provide interoperability, a standard, namely MPEG-7, is developed by Moving Picture Expert Group for multimedia content description [1][3][6].

MPEG-7 is one of the widespread standards used for describing content of audiovisual information by providing a rich set of standardized tools to describe multimedia content [1]. MPEG-7 provides a set of audio-visual Description Tools in the forms of Descriptors and Description Schemes which are complex data types used to describe audio-visual content in order to provide efficient access to the multimedia content for semantic content based retrieval systems [2]. Although the standard is developed for describing different multimedia contents in a standard representation, its XML-based structure prevents semantic interoperability between similar content based retrieval systems [3][6]. Moreover, MPEG-7 does not support reasoning capability and in MPEG-7 standardization the same semantic content can be represented in different ways. In order to provide semantic interoperability, a standard representation and reasoning capability, MPEG-7 to Semantic Web transition has been attempted [4].

Semantic web technologies are applied to MPEG-7 standard and the standard schema is translated into OWL ontology in order to solve semantic interoperability issues. By this way, four MPEG-7 ontologies having different amount of coverage of the standard and different expressiveness power are proposed [4][5][8]. These

MPEG-7 ontologies are Jane Hunter Ontology [18], Tsinaraki Ontology [7], Rhizomik Ontology [14] and COMM Ontology [19]. In this thesis, MPEG-7 Rhizomik ontology is used as the core ontology due to its complete coverage of the standard and reasonable expressiveness power. Moreover, domain specific ontologies are integrated to the core MPEG-7 ontology in order to improve semantic content-based retrieval capabilities. The usage of domain ontologies presents domain specific concepts to the users for providing legal content annotation and content-based queries.

In this thesis, we aim to propose a solution for semantic content modeling and data retrieval issues in video databases and develop a framework, namely An Ontology-Driven Video Annotation and Retrieval System. Our system provides video content annotation and querying capabilities to the user. In the proposed system, an ontology-driven approach is adopted by constructing MPEG Rhizomik ontology as the core ontology and attaching domain ontologies to this core ontology. Due to the supported ontology infrastructure, domain specific concepts can be used to annotate video contents and formulate content-based user queries. The system supports semantic content modeling and annotation which can be divided into concept annotation and spatio-temporal annotation. Concept annotation covers the annotation of the individuals, objects and events in the video content according to the domain specific concepts. Moreover, the system also supports domain specific common application data integration and this integration process decreases concept annotation effort. Spatio-temporal annotation includes the spatial and temporal relations between individuals, objects and events. The supported spatial relations are composed of directional relations (north, south, east, west), mixed directional relations (northwest, northeast, southwest, southeast), distance relations (near, far), positional relations (left, right, above, below) and topological relations (overlaps, equal, inside, touch, contain, cover, covered-by, disjoint) [37][45]. The supported temporal relations are defined according to Allen’s temporal algebra relations (precedes, preceded-by, meets, met-by, overlaps, overlapped-by, finishes, finished-by, starts, started-by, equals, during) [42].

The proposed system supports ontology-driven concept querying, spatio-temporal querying, region-based and time-based querying capabilities as simple querying types. Besides these simple query types, compound queries are also supported in order to enhance querying capability of the system. Compound queries are generated by combining simple queries with "(", ")", "AND" and "OR" operators. For all these query types, the system supports both general and video specific query processing. By this means, the user will be able to pose queries on all videos in the video databases as well as a specific video of interest.

The main difference between our proposed framework and other video management and retrieval systems is its ontology support. Our system uses both MPEG-7 Rhizomik ontology and domain ontologies for modeling and querying the video semantic content, including spatio-temporal data. The users can formulate ontological queries by using the domain ontology concepts via a form-based user interface. The user queries are converted to SPARQL queries and the query engine executes these SPARQL queries in order to retrieve ontological data and query results. By means of the ontological querying capability, the users can query the members of a domain ontology class. For instance, it is possible to query mammals in an animal documentary. Or, the user may be interested in only the positions of home team players when an offside event occurs in a soccer video.

The system is developed as a web application and provides a form-based user interface for video content annotation and querying. The current version of the framework supports manual annotation of the videos, however the system infrastructure is convenient to integrate automatic or semi-automatic annotator modules in latter studies. On the other hand, the query interface enables the formulation of different types of queries such as concept, spatio-temporal, region based and time based queries. Moreover, by combining these sub-queries, compound queries are also generated and sent to the query engine. After generating corresponding SPARQL queries, the queries are executed and matching query results are retrieved. Finally, the query results are organized and displayed to the end users in the user interface.

This thesis is conducted as a part of a research project partially supported by TUBITAK (TUBITAK-EEEAG 107E234). The main objective of the project is to develop an ontology-driven video management framework for the management of personal videos. The proposed framework provides semantic content based modeling, annotation and querying of multimedia content capabilities to the user. In this thesis work, the main infrastructure of the project is developed providing the mentioned system capabilities.

In the concept of this research project, some academic studies have been already performed. An ontology-based multimedia information management system was proposed by Hilal Tarakçı in 2008 [20]. That was the initial study on the framework and the ontological infrastructure was constructed in the concept of that study. MPEG-7 ontology structure was established and an automatic integration mechanism was developed for the attachment of domain specific ontologies to the MPEG-7 ontology. The initial framework has very limited annotation and querying capabilities. Only concept annotation and querying capabilities are provided. In order to satisfy the project specifications and construct a personal multimedia management system, some extra capabilities are needed to be added.

In order to improve system capabilities, ontology-based spatio-temporal video management system was developed by Atakan Şimşek in 2009. The developed system provided spatio-temporal data modeling and querying capabilities to the system. Thus, a framework that provides ontology-driven concept modeling, spatio-temporal annotation and data retrieval capabilities was achieved [37]. Moreover, a knowledge base structure has been constructed for the management of spatio-temporal data modeling in order to optimize system space usage.

Furthermore, a natural language query interface and SPARQL query generator has been proposed by Filiz Alaca in this year in order to provide a natural language query input to the system. According to the proposed system, natural language queries provided by the users are converted to the SPARQL queries and supplied as query input to the system [44].

In the light of these studies, the infrastructure management, video annotation and querying capabilities are decided to be improved in this thesis work. After conducting a literature survey, domain specific common data integration is decided to be added to the system. Moreover, a time-efficient spatio-temporal data modeling approach has been proposed by replacing the knowledge base structure with a relational database and JENA ontology concepts. On the other hand, spatio-temporal querying capability is enhanced with the region-based and time-based query types. Compound querying mechanism has been formalized and for all query types both general and video specific query processing mechanisms are developed.

The contributions of this thesis can be summarized as follows:

- The proposed solution offers the automatic integration of domain specific common data to the ontology infrastructure and by this means; annotation effort to be expended for domain specific common data addition is decreased.

- In this thesis, all spatio-temporal data and relations for the annotated video is generated in annotation phase and stored in the relational database for latter querying phase. Thus, in query processing time, there is no need to re-generate necessary relations between the queried objects again. Since time efficiency requirements are more important for a query engine than system space requirements, this approach provides a better data retrieval performance.

- The developed system provides region-based and time-based querying capabilities. As a result, individuals, objects, events or domain ontology concepts occurring in a specified region or time interval can also be queried.

- The spatio-temporal querying, region-based and time-based querying capabilities can also be performed for the events in the video content besides objects and individuals.

- The proposed system also supports domain independent ontology-driven compound querying capability. Compound queries are constructed by combining the other simple query types and an optimization approach has been developed for compound query processing in order to provide an efficient query execution.

- In this thesis, time interval based video annotation capability is provided instead of frame by frame annotation.

- The proposed system provides a generic multiple video querying capability. By this means, a user query is executed on a video database and the query can match various video scenes from different video files. As query result, all these matching scenes from multiple videos are returned to the user.

- The system also provides specific video querying capability. According to this feature, the user will be able to select a video for query operations and pose queries on this specific video of interest. As query result, all matching scenes from the selected video file are returned to the user.

- The developed system provides a form based user interface to the users and the formulated form based user queries are converted to the SPARQL queries and send to the system query engine for execution. Since the query engine accepts SPARQL queries, the system can be easily integrated with other systems providing different user interface components and generating SPARQL query outputs.

- The system is developed as an ontology-driven generic system. MPEG-7 Rhizomik ontology is used as the core ontology and domain ontologies are attached to the MPEG-7 ontology. In this way, interoperability and data exchange between other MPEG-7 ontology-driven systems are provided. On the other hand, the system is domain independent and any domain ontology can be integrated to the core ontology automatically.

The rest of the thesis is organized as follows:

- Chapter 2 presents some background information about the concepts and technologies used in this thesis. In addition, some important related work is explained in Chapter 2 and they are compared with this thesis work in terms of their weaknesses and strengths.

- Chapter 3 gives brief description about the system properties and specifications. Moreover, the general system architecture is also mentioned in this chapter.

- In Chapter 4, the details of the video annotation and data modeling concepts are described. Ontology-driven concept annotation and spatio-temporal annotation mechanisms are presented.

- In Chapter 5, the different query types supported by the system are described and the details of the system query processing approach are explained.

- Chapter 6 gives the system implementation details and presents a case study related to the usage of the system. The functionalities of the user interface module are also demonstrated in this chapter.

- Chapter 7 concludes the thesis with a brief summary and discusses some possible future work.

CHAPTER 2

BACKGROUND INFORMATION AND RELATED WORK

This chapter consists of two parts. In the first part, some background information regarding the terminology, standards and methodologies used in this thesis is presented. In addition, definitions and general overview of the multimedia content management concepts are explained in this section.

In the second part, some approaches to multimedia management systems that deal with indexing, storage and content-based retrieval of multimedia content data are described and their comparison with this thesis work is presented.

2.1 Background Information

2.1.1 MPEG-7 Standard

MPEG-7, namely Multimedia Content Description Interface, is developed by Moving Picture Experts Group (MPEG) as an ISO/IEC standard for describing the multimedia contents. The rising goal of MPEG-7 standard is to represent the semantic contents as well as the actual multimedia content by using a comprehensive set of Description Tools [1]. In this way, MPEG-7 standard aims to provide interoperability between different applications dealing with multimedia content data.

The main elements of MPEG-7 standard include the following [2]:

Description Tools:

- *Descriptors (D):* Elements that are used to define syntax and semantics of multimedia content features. Descriptors define low-level features such as color, texture, motion, shape etc., as well as high-level features such as title and author.

- *Description Schemes (DS):* Elements that are used to define the relationship between descriptors and other description schemes.

Description Definition Language (DDL): DDL is used to define the syntax of MPEG-7 descriptors and description schemes. Moreover, DDL also allows the definition of new descriptors and description schemes.

System Tools: System Tools are used to define optimization for effective storage and transmission of the content.

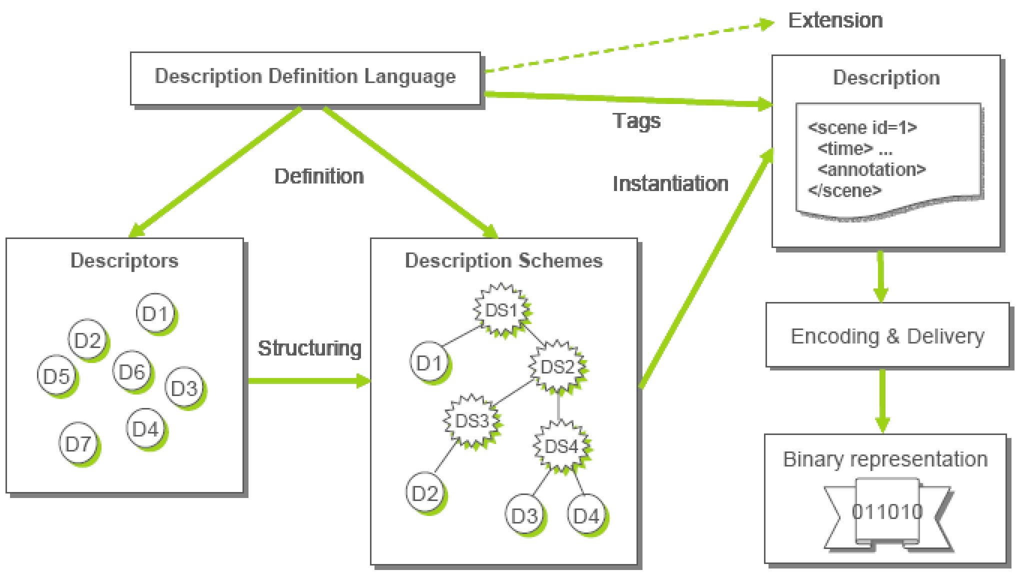


Figure 2.1.1: MPEG-7 Main Elements

Although MPEG-7 standard allows the description of semantic multimedia content and aims to provide interoperability between applications, the data exchange between different systems and interoperability issues are still problematic. Due to the complexity of MPEG-7 standard and its descriptors, the same content can be described in several ways. For that reason, it is difficult to provide interoperability between different systems, since the same multimedia content can be differently described and annotated in each system [3].

As a result, MPEG-7 standard is inadequate to represent semantic content precisely and in order to overcome interoperability issues, there exist several studies to move MPEG-7 standard to Semantic Web [4][5].

2.1.2 MPEG-7 Ontology

MPEG-7 standard is implemented by XML Schemas and therefore most of the semantic content becomes implicit [4]. On the other hand, as mentioned before, semantically identical content can be described in multiple ways by using the MPEG-7 standard, since it is difficult to define precise semantics in XML syntax. For that reason, lack of precise semantic description in XML-based syntax prevents semantic interoperability [6]. In order to enhance semantic interoperability issues, semantic web technologies are applied to MPEG-7 standard and the standard XML schema is translated into an ontology represented by OWL [7][4][18][19]. As a result, four MPEG-7 based multimedia ontologies are proposed [5]. The proposed MPEG-7 ontologies are Jane Hunter's MPEG-7/ABC ontology [18], Tsinaraki's MPEG-7/Tsinaraki (DS-MIRF) ontology [7], Garcia and Celma's Rhizomik model [14] and Arndt's COMM [19]. These ontologies differ from each other in their coverage of the standard and expressiveness power. The proposed MPEG-7 ontologies are summarized in Table 2.1.1 [5]:

Table 2.1.1: Comparison of MPEG-7 Based Ontologies

	Hunter	**Tsinaraki**	**Rhizomik**	**Comm**
Foundations	ABC	-	-	DOLCE
Complexity	OWL-Full	OWL-DL	OWL-DL	OWL-DL

Table 2.1.2 Continue

Coverage	MDS + Visual	MDS + CS	All	MDS + Visual
Applications	Digital Libraries, e-Research	Digital Libraries, e-Learning	Digital Rights Management, e-Business	Multimedia Analysis and Annotations

1. **Jane Hunter Ontology:**

Jane Hunter made the initial efforts towards the semantic representation of MPEG-7 descriptors in a machine understandable way to provide interoperability. RDF Schema and DAML-OIL constructs are used to formalize and describe MPEG-7 MDS and visual metadata structures [18]. Hunter also created an upper ontology and integrated other domain ontologies to the upper ontology in order to provide a basis for data exchange. For that reason, ABC ontology is used as an upper ontology and domain ontologies are integrated to ABC ontology [3][18]. The main deficiency in Jane Hunter's MPEG-7 ontology is its limited coverage of the MPEG-7 standard.

2. **Tsinaraki (DS-MIRF) Ontology:**

Tsinaraki developed a core ontology manually by converting MPEG-7 XML Schema to OWL-DL. Tsinaraki's DS-MIRF ontology has OWL-DL complexity [5][8][14][15][16]. During ontology development, MPEG-7 simple data types are integrated to OWL ontology using rdfs:Datatype [13]. For each complex type in MPEG-7 XML Schema, a corresponding OWL class is defined [12] and relations between entities are constructed. For evaluation and demonstration of the MPEG-7 ontology, soccer and formula domain ontologies are integrated to the core ontology [13].

3. **Rhizomik Ontology:**

In 2005, Garcia and Celma proposed the Rhizomik ontology in order to translate MPEG-7 XML Schema to OWL. Their approach is developed as a generic solution providing an automatic and complete mapping from MPEG-7 Schema to OWL [4]. The Rhizomik ontology is generated as an upper ontology and domain ontologies can be integrated to this upper ontology [4]. The Rhizomik ontology is developed as an MPEG-7 ontology of OWL-Full complexity since both object type and data type properties are needed [5].

In Rhizomik approach, the automatic MPEG-7 Schema to Semantic Web transformation process includes two steps; XSD2OWL Mapping and XML2RDF Mapping. XSD2OWL Mapping captures the implicit MPEG-7 Schema semantics. The mapping is performed by transforming XML Schema constructs into OWL constructs [4]. However, after the automatic transformation, the collision of name domains arises, since OWL uses a unique name domain whereas XML has separate name domains. This problem is fixed by the manual resolution of name collusions [17]. In XML2RDF Mapping, XML metadata that instantiates the OWL ontology obtained in XSD2OWL Mapping, is translated to RDF metadata in a transparent way [4]. The approach uses structure-mapping models for transparent XML metadata to RDF translation.

4. **Comm Ontology:**

COMM is designed as a core multimedia ontology by re-engineering MPEG-7 semantic using DOLCE patterns [19]. COMM has OWL-DL complexity.

In this thesis work, Rhizomik MPEG-7 ontology is chosen as the core multimedia ontology because of its expressiveness power and its flexibility to merge domain ontologies to the core ontology.

2.1.3 Domain Ontology Integration

Integrating domain knowledge to the core MPEG-7 ontology enhances semantic interoperability and audiovisual multimedia content retrieval performance. In addition, integration of domain ontologies to the multimedia metadata model gives the user the ability to generate ontology-based queries on multimedia resources. For this purpose, the domain knowledge is integrated to the upper ontology in the form of domain ontologies [7][11][12]. There are two main approaches to integrate domain ontologies to the MPEG-7 ontology proposed by Hunter et. al. [6][18] and Tsinaraki et. al. [6][7].

In Jane Hunter' approach, ABC ontology is chosen as the upper ontology. MPEG-7 and other domain ontologies are attached to ABC ontology by using the attachment points provided by the ontology for MPEG-7 and domain specific ontology integration. Technically, MPEG-7 classes and other domain ontologies are attached to the core ontology by extending the corresponding ABC classes [18].

In Tsinaraki's approach, MPEG-7/Tsinaraki ontology is chosen as the core ontology and domain specific ontologies are integrated to the core ontology. According to this approach, domain specific ontologies construct the second layer of the multimedia metadata model and extend the existing concepts in the core ontology with domain knowledge. Domain ontology integration includes sub-classing domain ontology concepts from the upper ontology constructs such as SemanticBaseType, EventType, ObjectType, AgentObjectType, ConceptType, SemanticPlaceType and SemanticTimeType [12][13].

In this thesis work, Tsinaraki's approach is applied by choosing MPEG-7 ontology as the upper ontology and integrating domain specific ontologies to the upper ontology [12][13][20]. The major aim of domain ontology integration is to hide the integration details from the end user. For that reason, domain ontology integration is performed in an automated way in order to provide an efficient usage of domain specific ontologies [20]. According to the applied approach in the thesis work, only domain ontology concepts are visible to the end user. Therefore, the user only deals with the domain knowledge without being familiar with the upper ontology classes and concepts. These domain ontology concepts are used during annotation and querying phases by enabling the user to incorporate domain knowledge to the multimedia metadata model. In addition, ontology-based querying capability also allows formulating queries using the domain ontology concepts and provides an efficient retrieval performance on multimedia resources.

2.1.4 JENA Ontology API

JENA is a Java framework used to create Semantic Web applications. JENA Framework provides classes and interfaces for the creation and manipulation of RDF repositories and OWL ontologies [22]. Through the JENA Ontology API, a consistent programming interface is aimed to be provided in order to develop ontology applications, independent of the ontology language used [23]. JENA Framework provides the following capabilities:

- An RDF API,
- Reading and writing in RDF/XML, N-Triples,
- An OWL API,
- In-memory and persistent storage,
- RDQL - a query language for RDF,
- SPARQL query engine.

In JENA, the Model Interface is used to represent the entire ontology as graphs and models are created, loaded, saved, modified and queried using plain Java [21][22]. Moreover, JENA provides reasoning over ontology models. The main advantage of building an ontology-based structure is to derive additional information about the concepts being modeled [23]. JENA derives extra information and relationships between the concepts that the model does not express directly [21]. For instance, assume building an ontology having a simple hierarchy of the concepts organism, animal and fish. In such a hierarchy, fish is a sub-class of animal and animal is a sub-class of organism. If an individual "Fred" is inserted to the model as a "Fish", the assertion "Fred is a Fish" entails the deduction "Fred is an Animal" and "Fred is an Organism" [23].

JENA provides querying capability by using SPARQL engine and JENA specific ontology queries. ARQ engine is used for SPARQL query processing. By the help of ARQ engine and JENA, SPARQL queries are created, executed and a result set is generated. The result set contains the matching results returned by the executed query and these results can be iterated and displayed to the user.

In this thesis work, JENA framework is used for the domain ontology integration, ontological model creation, modification and querying. JENA ontology API manages all ontological processes. During the domain ontology integration, model creation and ontological data population with annotations, JENA ontology API methods are used. For query formulation and processing phase, SPARQL query engine is used. The user queries, which are formulated via a form based user interface, are converted to SPARQL queries and JENA ARQ engine executes the queries and returns the query results.

2.1.5 SPARQL

SPARQL, namely **S**PARQL **P**rotocol **a**nd **R**DF **Q**uery **L**anguage, is a query language designed by the W3C RDF Data Access Working Group. It is developed as a query language and a protocol for accessing RDF graphs [24]. As a query language, SPARQL is data-oriented without supporting an inference mechanism. However, ARQ and SemWeb implementations of SPARQL are used for providing an inference mechanism. ARQ Engine is a SPARQL query language implementation for JENA and it provides OWL reasoning [25]. SemWeb is an RDF library and it provides RDFS reasoning [26].

1. **A Simple SPARQL Query:**

   A Simple SPARQL Query is composed of two parts; a SELECT clause and a WHERE clause. The SELECT clause includes the variables that will appear in the query result and the WHERE clause includes the matching query patterns that will be matched against the triples in the data graph.

The following is a SPARQL query example that aims to find the title of a book on a given data graph [24].

```
SELECT ?title
WHERE
{
   <http://purl.org/book/Book1> <http://purl.org/book/elements/title> ?title .
}
```

In the example query, the subject and the query predicate have predefined values whereas the query object is a variable. For the evaluation of the query result, the query pattern is matched against the triples having these fixed subject and predicated values and the query result for title variable is generated. There is also a shorthand query formulation mechanism for long URIs using prefixes. The following query is an example of the prefixing mechanism [24]:

```
PREFIX  vcard:   <http://www.w3.org/2001/vcard-rdf/3.0#>

SELECT ?givenName
WHERE
{
   ?y vcard:Family "Smith" .
  ?y vcard:Given   ?givenName .
}
```

2. SPARQL Query Filters:

SPARQL Query Filters are used to restrict the values in a query solution. Most common query filtering mechanisms are string matching filters and testing value filters [24].

For string matching, SPARQL provides a filtering mechanism based on regular expressions to test the string values. The following example query includes the filtering mechanism according to string matching:

```
 PREFIX   user:   <http://purl.org/user/>

 SELECT ?name
WHERE
{
    ?person user:fullName  ?name .
```

```
   FILTER regex(?name, "r", "i")
}
```

In the example above, the usage of the flag "i" denotes that a case-insensitive pattern matching is performed. If the last flag argument is not stated in the query, only case-sensitive pattern matching is performed. In this example, the query returns the full names having "r" or "R" letters in them [24].

For testing value, SPARQL provides a filtering mechanism based on the value of a variable. The following query includes the filtering mechanism according to the testing value approach:

```
PREFIX   user: <http://purl.org/user/>

SELECT  ?person
WHERE
{
   ?person  user:age  ?age .
   FILTER (?age >= 18)
}
```

In the above example, the query results are restricted according to the value of the *age* variable and the query solution includes the people who are older than 18 [24].

3. SPARQL Query Solution Modifiers:

Query patterns generate an unordered solution list. However, the solution list can be converted to a modified sequence by applying solution modifiers. The SPARQL solution modifiers can be one of the following [24]:

Order By: The result set is sorted according to the given criterion.

Projection: The result set is constructed by selecting only certain variables.

Distinct: The solutions in the result set are ensured to be unique.

Reduced: Some of the solutions in the result set are permitted to be eliminated since they are non-unique solutions.

Offset: The result set contains a set of solutions starting from the specified start index.

Limit: The result set size is restricted according to the specified number. The specified number of solutions is returned.

In this thesis work, ARQ implementation of SPARQL is used for query processing and execution. ARQ is a query engine for JENA that supports SPARQL query language. By the usage of ARQ engine, the proposed system also provides OWL reasoning capability.

2.2 Related Work

The approaches dealing with the indexing and retrieval of multimedia data according to its semantic content have become very popular research topics in recent years. There has been a significant amount of work conducted on the storage issues, and semantic content based annotation and querying capabilities for multimedia data especially in video databases. In the following part, we aim to examine the related research conducted in the literature and present the methodologies and approaches for multimedia data management issues. In addition, this thesis work is compared with the proposed approaches in order to present its strengths and weaknesses.

2.2.1 BilVideo

BilVideo is a web-based video database management system that supports spatio-temporal, semantic content based and low-level feature based querying on video data [27][29][31]. The client-server architecture of BilVideo is illustrated in the following figure [27]:

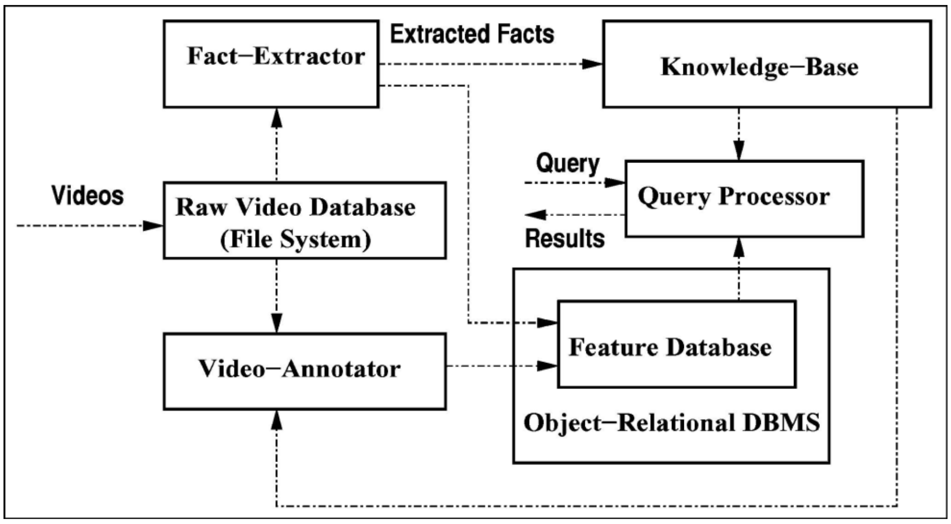


Figure 2.2.1: BilVideo System Architecture

The query processor module is responsible for processing the user queries, retrieving the related audio-visual content and responding to the user. This module interacts with a knowledge base and object relational database. BilVideo handles spatio-temporal queries using a knowledge-base, which consists of a fact-base and a comprehensive set of rules implemented in Prolog [28]. Semantic and low-level feature based queries are handled by using the object-relational database [32]. For compound queries having spatio-temporal, semantic and low-level feature based sub-queries, the query processor directs the sub-queries to the related component and retrieves the intermediate results of these sub-queries by communicating with the knowledge base or the object relational database [29][33]. After collecting the intermediate results, these sub-query results are combined in order to generate the result of the whole query and the final query result is displayed to the users on Web client interfaces [33].

The video annotator tool interacts with the raw video database system and extracts the semantic metadata stored in object relational database for semantic queries [29][31]. On the other hand, fact extractor populates the knowledge base and extracts color and shape histograms of the objects. Extracted color and shape histograms of the objects are stored in the feature database to be used for color and shape queries [34]. Moreover, spatio-temporal relations between objects, object-appearance relations, and object trajectories are extracted semi-automatically and stored in the knowledge base as a set of facts representing the relations and trajectories in order to be used for spatio-temporal querying [28].

In BilVideo, a spatio-temporal query contains any combination of directional, topological, 3D-relation, object-appearance, trajectory-projection and similarity-based object-trajectory conditions [33]. Spatio-temporal query processing is performed according to the extracted facts and query rules stored in the knowledge base. However, not all extracted facts are stored in the knowledge base. After extracting the facts, a fact pruning algorithm is applied and only basic facts are added to the knowledge base. There are extraction rules in the knowledge base and the facts that can be derived using these extraction rules according to the fact extraction algorithm are not stored in the knowledge base. The extraction rules in the knowledge base significantly reduce the number of facts representing the relations that need to be stored for spatio-temporal querying of video data which results in a considerable storage space saving [28][29].

BilVideo also introduces its SQL-like query language and the queries can be sent to the system using its own query language [32].

The latest BilVideo version, BilVideo v2.0 is also developed by the BilVideo research group and it is an MPEG-7 compliant video database management system using MPEG-7 standard for the description of audio-visual content. BilVideo v2.0 uses native XML database for storage and querying capabilities instead of BilVideo’s two different storage environments approach using a relational database and a knowledge base. BilVideo v2.0 has its own XML-based query language. The query processor interprets the queries and generates XQuery statements to query native XML database [33].

The proposed system in this thesis differs from BilVideo and BilVideo v2.0 in the followings:

- BilVideo v2.0 is MPEG-7 compliant and uses MPEG-7 descriptors for the description of the video content [33][34]. However, our system is MPEG-7 ontology-driven. Since MPEG-7 has interoperability problems, using MPEG-7 ontology-driven approach is more powerful in order to overcome the interoperability issues.
- Since our proposed system supports the integration of domain ontologies to the MPEG-7 core ontology, ontology-based annotation and querying is provided. By this way, audio-visual content of the video data is annotated with ontological concepts and queried accordingly.
- BilVideo handles spatio-temporal querying over a knowledge base structure implemented in Prolog [28]. In our proposed system, we keep spatio-temporal relations in relational database with other semantic metadata and annotations. Jena Ontology API features are used to interact with relational database backend and storage and query processing capabilities are performed over the relational database approach.
- In BilVideo for spatio-temporal querying, all extracted facts are not stored in the knowledge base, only a pruned set of necessary facts are stored and the other facts are derived at query time by using the extraction rules and stored facts [28]. This approach provides storage space efficiency, however the inference of all other spatio-temporal relations in the querying phase brings an overhead in query processing time. For this purpose, we store all spatio-temporal relations in the database in the annotation phase and provide a time efficient retrieval for spatio-temporal querying in our system.
- In contrast to BilVideo and BilVideo v2.0, our proposed system does not have its own query language, queries are formulated by using a form based query formulation interface and they are converted to the corresponding SPARQL or JENA specific queries for query processing.
- There are some extra capabilities that BilVideo and BilVideo v2.0 provides which are different from our system. One of them is the low-level feature based querying according to color, shape, texture descriptors [34]. Another

feature is 3D spatio-temporal querying support while our system only deals with 2D spatio-temporal querying. These extra capabilities can be studied and integrated to our system as a future work.

- Semi-automatic fact extractor and video annotator modules of BilVideo handle semantic and spatio-temporal annotation of all objects in video data [29][34]. In our system, manual annotation of objects is supported. A semi-automatic annotation module can be integrated to our system as a future work.

2.2.2 Ontology-Driven Audiovisual Content Management Framework

In [10][11][15], a framework has been developed that handles the management, indexing and retrieval of semantic metadata describing the audio-visual content. The main objective of the developed framework is to enhance the retrieval performance while providing compatibility with the international multimedia standards such as MPEG-7 and TV-Anytime [10][11]. The system has a mechanism for the integration of domain ontologies with the multimedia content description standards. The use of domain specific ontologies guides the users to provide legal content descriptions for the specific content [9][12][13]. The following figure shows a detailed outline of the framework:

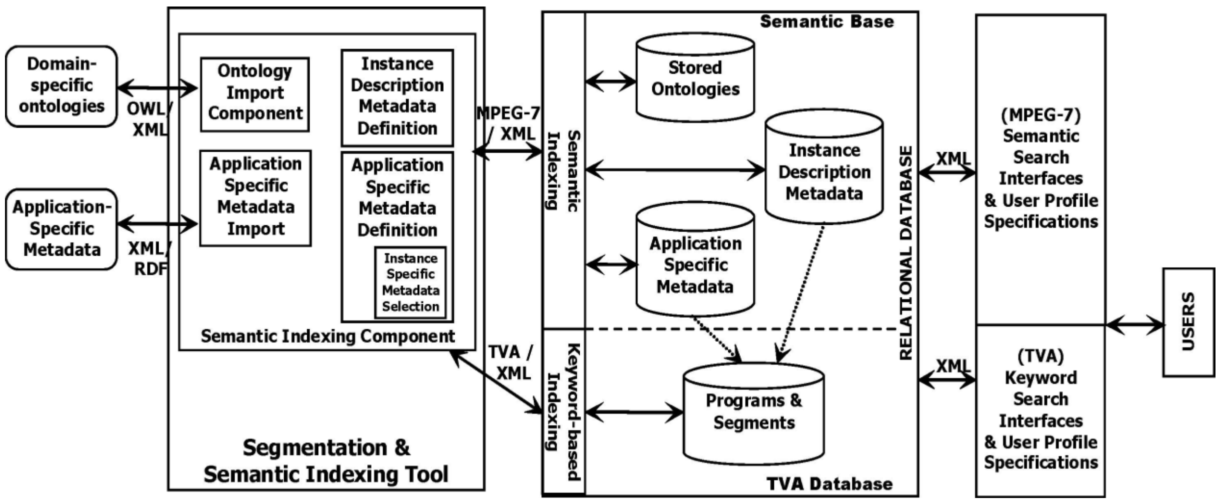

Domain-specific ontologies
OWL/XML
Application-Specific Metadata
XML/RDF
Ontology Import Component
Instance Description Metadata Definition
Application Specific Metadata Import
Application Specific Metadata Definition
Instance Specific Metadata Selection
Semantic Indexing Component
Segmentation & Semantic Indexing Tool
MPEG-7 / XML
TVA / XML
Semantic Indexing
Keyword-based Indexing
Semantic Base
Stored Ontologies
Instance Description Metadata
Application Specific Metadata
Programs & Segments
TVA Database
RELATIONAL DATABASE
XML
XML
(MPEG-7) Semantic Search Interfaces & User Profile Specifications
(TVA) Keyword Search Interfaces & User Profile Specifications
USERS

Figure 2.2.2: Audio-Visual Content Management Framework Architecture

The system includes a TV-Anytime (TVA) compliant database where TV-Anytime metadata and segments are stored and a semantic base where the ontology-driven semantic metadata are stored. The segmentation and semantic indexing component is used for importing the domain ontologies and application specific metadata. After these import operations this component populates the semantic base and TVA database by annotating audio-visual content. During the annotation phase, keywords related to the video content are also stored in the TVA database and used for the keyword-based search [11][13].

The presented system also provides a querying API that supports advanced semantic content based queries in addition to the keyword search capabilities [11][12]. Both databases are aware of the domain specific ontology-based extensions and as a result, querying according to the domain ontology concepts can be performed [16]. By combining the search API methods, more complex semantic queries can be supported.

This system is developed for soccer domain applications. A domain specific ontology for football matches is developed and imported into the system database [12][13]. The soccer videos are annotated and queried according to the domain ontology by using annotation and querying components. However, the framework constructs a base for the definition of other domain ontologies and an adaptation of these ontologies with the current framework. Domain independency is studied as a future work for the system [12][13][15].

Our proposed system differs from the mentioned framework in the following issues:

- Our solution does not depend on any specific domain; it allows the integration of a given domain ontology and supports the annotation and querying according to the domain specific ontology concepts.
- Our proposed system supports spatio-temporal annotation and querying capabilities different from this framework. Moreover, our system also enables region-based and time-based queries to the user.

- Compound querying according to the semantic content and spatio-temporal relations of the video content object is supported in our system.
- Our proposed system supports both general and video specific query processing for all query types different from this framework. By this means, the user is able to pose queries on all videos in the video databases as well as a specific video of interest.
- The mentioned system provides compatibility with TV-Anytime standard different from our system.

2.2.3 Ontology-Based Multimedia Information Management System

The ontology based multimedia information management system in [20] proposes a framework as a solution to the multimedia content management problem. The main objective of the proposed framework is to provide a standardized method for the audiovisual content description and overcome the semantic interoperability problems between similar semantic content based retrieval systems. For this purpose, the system uses Rhizomik MPEG-7 ontology as the core model and integrates this core MPEG-7 ontology with other domain specific ontologies. By this way, a domain independent solution is presented for semantic annotation and querying of multimedia content. The domain specific ontological concepts are used for semantic content based annotation and querying the annotated multimedia content. Ontology-driven approach enhances content based retrieval capabilities [20].

The system is a client-server web application and presents domain independent multimedia content annotation, querying and retrieval capabilities to the user. Figure 2.2.3 shows the system architecture for the proposed ontology based multimedia management framework in [20]:

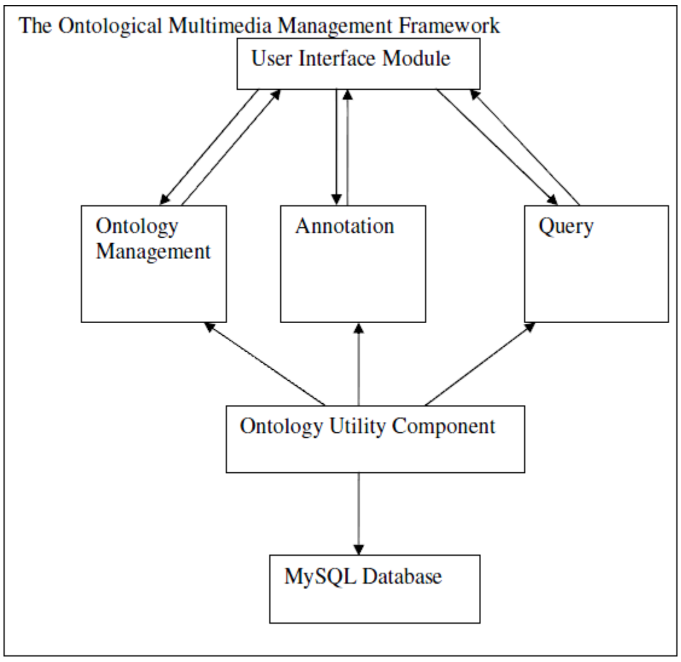

The Ontological Multimedia Management Framework
User Interface Module
Ontology Management
Annotation
Query
Ontology Utility Component
MySQL Database

Figure 2.2.3: Ontology-Based Multimedia Management Framework Architecture

The users interact with ontology management, annotation and querying capabilities of the system via the user interface module. The domain specific ontologies are imported and attached to the MPEG-7 core ontology and after this procedure the users are able to annotate the video contents with domain specific ontological concepts and legal content descriptions. By using the query interface, the user formulates queries on the annotated videos and performs ontology-based querying [20].

The mentioned core MPEG-7 ontology model and framework capabilities construct the basis of this thesis work. In this thesis, we have added extra capabilities for a multimedia management framework and contributed to the mentioned system by supporting the following abilities:

- Common Data Integration to Domain Ontology
- Spatio-Temporal Annotation
- Spatio-Temporal Querying
- Regional and Time-Based Querying

- Compound Querying
- General and Video Specific Querying

2.2.4 Ontology-Based Spatio-Temporal Video Management System (OntoVMS)

OntoVMS [37] is developed as an ontology-based video management system which supports semantic data modeling and querying in video files. The system is ontology-driven and makes use of Rhizomik MPEG-7 ontology as the core ontology and the domain specific ontologies are integrated to the core ontology. By supporting MPEG-7 ontology compatibility, interoperability with the other MPEG-7 ontology compliant systems is achieved. Moreover, the framework provides ontology-based content annotation and querying using domain ontology concepts [37].

The system supports concept modeling, spatio-temporal relations and trajectory modeling. Supported spatial relations are directional, positional, topological and distance relations. Allen’s temporal algebra relations are also supported as temporal relations in video files [37].

The following figure shows the main system architecture:

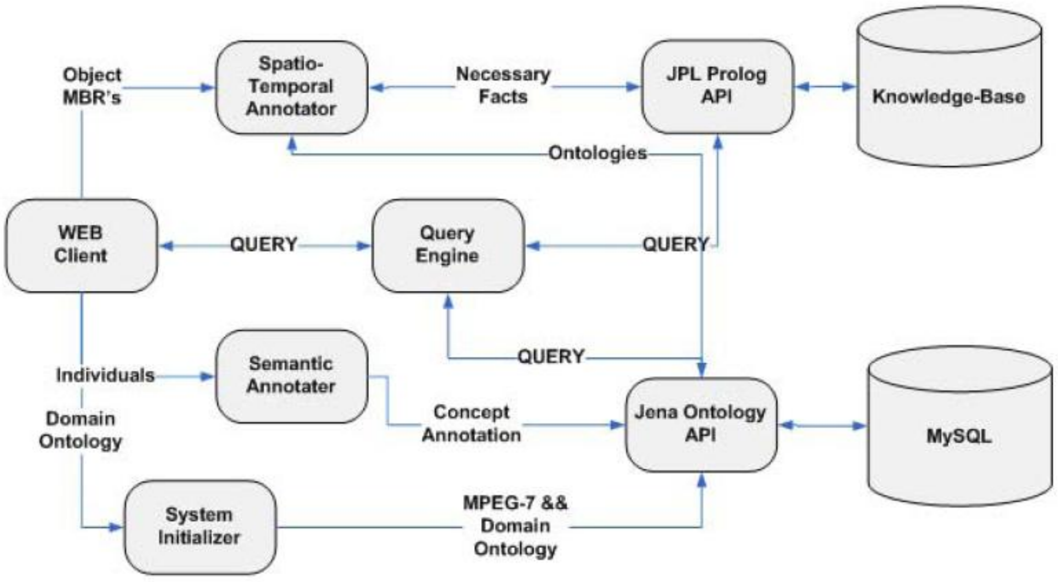


Figure 2.2.4: OntoVMS System Architecture

The query engine supports semantic content based, spatio-temporal and trajectory querying capabilities and interacts with the knowledge base and relational database for query processing. The knowledge base structure of OntoVMS stores spatial and temporal relations between video content objects. The rule based modeling contains basic facts and inference rules. Annotated spatio-temporal relations are pruned according to a fact pruning algorithm and only necessary facts extracted by spatio-temporal annotator are stored. By storing only a necessary subset of all spatio-temporal relations, OntoVMS aims to reduce the required storage space and provide space efficiency. In querying phase, inference rules are used to generate non-stored spatio-temporal relations [37].

OntoVMS query engine makes use of JPL Prolog API and JENA Ontology API in order to interact with the knowledge base and relational database for query execution. The system also supports a basic compound querying capability. Different query types such as concept queries, spatial, temporal, object appearance queries are combined to construct a compound query. For compound query processing, the query is divided into sub-queries, these sub-queries are evaluated separately and finally their results are combined to form the final query result [37].

The mentioned framework, namely OntoVMS, is chosen as the basis for this thesis work. In our proposed work, we have added extra features and extended the framework capabilities with the following issues:

- OntoVMS manages a knowledge base structure in order to store spatial and temporal relations and its query engine interacts with the knowledge base for spatio-temporal querying by using JPL Prolog API. In our proposed system, we keep spatio-temporal relations in the relational database with other semantic metadata and annotations. For spatio-temporal querying, similar to ontological concept querying, the system makes use of Jena ARQ Engine and SPARQL query language in order to communicate with relational database backend. The storage and query processing capabilities are performed over relational database approach.
- Fact pruning approach that OntoVMS applies in order to save storage space brings a cost in the query processing time. In our system, the extracted spatio-temporal facts in the annotation phase are directly stored in the database. The inference of all relations is performed during the annotation phase. By this way, the cost of pruning and inferring again in the querying phase is overcome and a time efficient retrieval for spatio-temporal querying is provided.
- In our system, after attaching the domain ontology to the MPEG-7 core ontology, we provide a mechanism to the user in order to integrate the application specific metadata to the system. By this means, the domain specific common data is integrated to the ontology infrastructure. This extension reduces the annotation effort and also provides common metadata knowledge to the user in querying phase.
- Our proposed system supports both general and video specific query processing for all query types different from OntoVMS. Thus, the user is able to pose queries on all videos in the video database as well as a specific video of interest.
- The developed system in this thesis work supports querying on multiple videos and the matched results retrieved from different videos are returned to

the user. However, in OntoVMS, queries are performed only on a selected video. Multiple video querying capability is not supported in OntoVMS.

- In our system, the annotation is performed according to frame intervals whereas OntoVMS supports frame by frame annotation. For that reason, the same scene and instances are needed to be annotated for each frame of the consequent video frames.

2.2.5 Content-Based Video Query System

In [35][36], Kuo and Chen propose a new mechanism for the content-based retrieval of video data. They present a content based video query language named CVQL and a query processing strategy for the language [35]. The system is a prototype video database system providing a graphical user interface and CVQL processor. CVQL allows users to specify query predicates by using spatial and temporal relations between the video content objects. The user constructs a sketch of the query and provides the corresponding query predicate via GUI and the query processor performs the evaluation of the predicates. As a result, the related videos or frames are displayed to the user [35][36].

A CVQL query is based on the predicate specification of the temporal and spatial relationships of the video objects. For users, it is easier to retrieve videos and frames according to the specified video content. They generally remember snapshots of the videos and query according to the remembered content. By the help of query predicates, CVQL allows the specification of the following query types [35]:

- Existence Querying
- Spatio-Temporal Querying
- Compound Querying

For content-based queries, firstly, video objects and object relations are detected by using video analysis and shot detection techniques. According to the detected video objects and their relations, four types of indices for CVQL are constructed [35].

Symbol Object Hierarchy: Symbol objects are managed in a class hierarchy. When a symbol object class is used in a query, it represents all symbol objects belonging to this class.

Video-Symbol_Object Table (VST): This table keeps an entry for each symbol object detected and lists the videos that this symbol object appears.

Symbol_Object Life-Time Table (SLT): This table is kept for each video. For each symbol object in the video, frame intervals that the object appears are recorded.

Symbol_Object Spatial Information Table (SST): This table is also kept for each video. It records the spatial locations of the symbol objects in each frame. By using SST entries, motion tracks of the objects and spatial relations between the objects can be derived [36].

The query processing is based on three phase elimination. In video elimination phase, VST structure is used to eliminate the videos that do not contain the symbol objects specified in the query predicate. In frame elimination, before predicate evaluation, videos are filtered by the SLT tables and the frames not containing the symbol objects specified in a predicate are eliminated. In video function evaluation phase, SST tables are accessed to evaluate the remaining frames. Query predicates are evaluated and results are generated [35][36].

The differences between this system and our proposed framework are:

- Our proposed system supports the integration of domain ontologies to the MPEG-7 core ontology. Ontology-based annotation and querying is provided. By this way, audio-visual content of the video data is annotated with ontological concepts and queried accordingly.
- Our system also supports annotation and retrieval of semantic events and activities in videos whereas the mentioned system only deals with content objects.
- This system has its own query language, CVQL and queries are formulated in CVQL using query predicates. However, in our system queries are

formulated by using a form based query formulation interface and they are converted to corresponding SPARQL or JENA specific queries for query processing.

- The system makes use of video analysis and detection techniques for the extraction and annotation of video objects semi-automatically. However, our proposed solution provides manual annotation. Semi-automatic annotation and extraction techniques can be added to our system as a future work.

2.2.6 Extended AVIS

Extended-AVIS [38] proposes a video data model that supports fuzzy spatio-temporal modeling and querying capabilities. This system is developed as an extension to AVIS video data model [49]. By providing some extra capabilities, Extended-AVIS supports semantic content modeling with the spatial and temporal properties of objects and allows spatio-temporal querying with the inclusion of fuzziness. Regional and trajectory querying capabilities are also provided in Extended-AVIS. The video data model focuses on video objects, activities and events and semantic metadata related to these concepts are indexed using some special index structures. Association maps, a frame-segment tree and array structures are used as index structures and these index structures are used to retrieve data and derive relations between the video objects in the query processing phase [38].

Our proposed system differs from Extended-AVIS in terms of the following issues:

- Extended-AVIS is not ontology based system whereas our system is MPEG-7 ontology-driven and supports the integration of domain ontologies to the MPEG-7 core ontology. By this way, audio-visual content of the video data is annotated with ontological concepts and queried accordingly.

- Compound querying capability is not supported in Extended-AVIS. However, our proposed system allows compound queries which are combinations of different query types.

2.2.7 Ontology-Supported Video Modeling System

A video database model that provides almost automatic object, event and concept extraction is proposed in [39]. The system is ontology based and ontology of objects, events and concepts is defined and used to extract new events and concepts in videos [39]. N-Cut image segmentation and a genetic algorithm based approach are followed to classify the candidate objects obtained from segments and key-frames [39][40]. Besides the event and concept extraction, spatial relations between the objects, temporal relations between events and object trajectory information are extracted and stored [39][41]. All derived events and concepts are stored with their time intervals. Standardized MPEG-7 descriptors are used in order to be compatible with the other MPEG-7 compliant systems for information exchange. Figure 2.2.5 shows the general system architecture [40]:

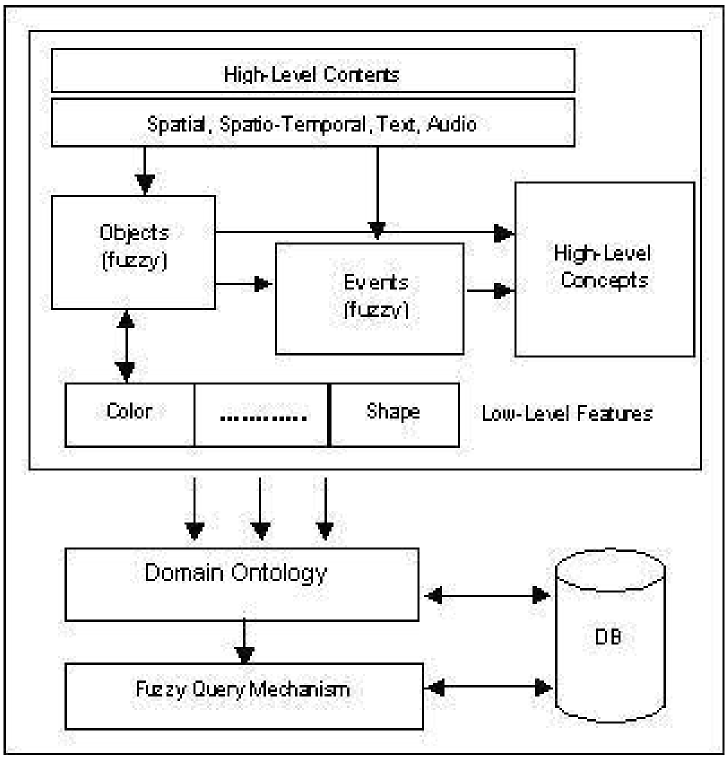


Figure 2.2.5: Ontology-Supported Video Modeling System Architecture

In this video model, there is an object, event, concept ontology which constructs the upper layer of the model and for each domain this generic video ontology is modified with the domain specific data at lower layers. Ontology-supported model provides the ability for ontology-based querying. Moreover, the system also allows fuzziness in object features, relations, ontology and query mechanism in order to make more flexible query formulations [39][40]. Queries related to the objects, events, spatio-temporal relations between the objects and low-level features are supported. Compound queries can also be formulated by combining the sub-queries related to the above mentioned query types [40][41].

The main differences between this system and our proposed framework are:

- Our proposed system makes use of MPEG-7 ontology and integrates domain ontologies to this core ontology in order to resolve semantic interoperability problems. However, the system in [39] uses standardized MPEG-7 descriptors.
- The system in [39] supports semi-automatic object and event extraction whereas our system supports manual annotation of semantic content and events. Moreover, this system uses fuzzy modeling and querying. These features can be studied as a future work in our system.

2.2.8 Spatio-Temporal Query Approach for Multimedia Databases

Wattamwar and Ghosh propose a spatio-temporal formalism in order to specify arbitrary spatio-temporal relations between the multimedia content objects [42]. Their approach includes fuzziness in the formalism and they illustrate Allen’s relations in their uncertain form. Allen’s relations are formulated according to the binary string encoding schema proposed by Papadias [42]. The temporal relations between the two events are defined to be binary 5-tuples (R_{tuvwx}: t, u, v, w, x ϵ {0, 1}) where “0” indicates n empty intersection and “1” indicates a nonempty one. The relationship between a primary interval [a, b] and a secondary interval [c, d] can be uniquely determined by considering the five empty or nonempty intersections of [c, d] with each of the five regions, denoted by five binary variables t, u, v, w, x, respectively.

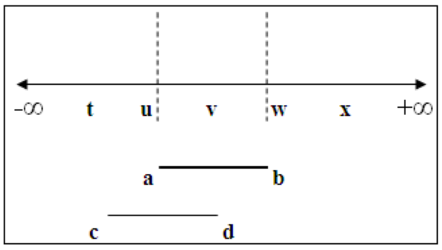


Figure 2.2.6: Five Regions of Interest

According to this approach, for example ***precedes*** relation is denoted by R_{10000} whereas ***overlaps*** relation is denoted by R_{11100}. By defining neighborhood values for the regions and degree of intersection increases the regions of interest, the system describes more detailed relations and also adds fuzziness to the formalism [42].

The main differences between this work and our proposed system are:

- MPEG-7 standard is used for audio-visual content description whereas our system makes use of MPEG-7 ontology and domain ontologies.
- Binary string encoding schema is used in their work for formulating spatio-temporal relations. However, in our work we use a relational approach.
- This approach supports fuzziness and this can be added to our system as a future work.

2.2.9 Multimedia Ontology Framework for Event Annotation and Retrieval

The framework in [43] proposes an ontology-driven approach used for higher-level annotation of the video clips and formulation of complex queries that comprise events, actions and their temporal relations. The multimedia ontology is composed of two main parts: the domain-specific ontology and the video structure ontology. The domain ontology includes all the concepts and relations that define the related application domain whereas the video structure ontology describes the components of a video such as clips, shots, frames, etc. The video structure ontology also defines the descriptors of the visual content. Semantic annotation is performed semi-automatically by using face detection and text recognition algorithms according to the multimedia ontology concepts [43].

Exploitation of the multimedia ontology allows queries that combine visual concepts with high-level domain-specific concepts. The queries formulated via graphical user interface are converted to the SPARQL query syntax and JENA APIs are used to pose queries with SPARQL [43].

The main differences between the mentioned framework and our system are:

- Both approaches are ontology-driven, however, in our proposed system we use MPEG-7 ontology and attach domain ontologies to this core ontology. This approach makes our system compatible with the other MPEG-7 ontology compliant systems and solves interoperability problems.
- Our proposed solution supports spatio-temporal annotation besides semantic content based object and event annotation. By this way, spatio-temporal and trajectory queries are also processed.
- The multimedia ontology framework in [43] is developed for soccer domain; however their ongoing work aims to generalize the framework to other application domains. On the other hand, our solution is domain independent.
- This framework supports semi-automatic object and event extraction and inference mechanisms for the annotation of the videos. Our solution provides manual annotation and semi-automatic annotation can be added as a future work to our solution.

CHAPTER 3

GENERAL SYSTEM ARCHITECTURE

In this thesis, a system called Ontology-Driven Video Annotation and Retrieval System (OntoVARS) is developed in order to provide a video management system which is used for ontology-driven semantic content annotation and querying. The proposed system is MPEG-7 ontology-driven and allows ontological video content annotation and data retrieval.

In this chapter, the architectural design details of OntoVARS which include the general system architecture, the main system characteristics and components are introduced. In section 3.1 general properties of the system are explained. System specifications are mentioned in section 3.2. Finally, in section 3.3, the general system architecture and main component descriptions are detailed.

3.1 System Properties

The proposed system is an ontology driven video management system which provides ontology-based semantic content annotation and querying in video databases. The system uses Rhizomik MPEG-7 ontology as the upper ontology and enables the automatic integration of domain specific ontologies to the core MPEG- 7 ontology. The ontology infrastructure of the system provides interoperability and a common communication platform with other MPEG-7 ontology compatible systems. Moreover, domain ontology integration enables semantic annotation and querying according to domain ontology instances and concepts. The system is domain independent and proposes a generic solution that enables the integration of different domain ontologies to the system at the same time. The system can operate with any type of video for annotation and querying.

The system supports content-based annotation and spatio-temporal data modeling in video databases by using domain ontology concepts. The video annotation is performed manually via a form-based user interface which provides selections for the video to be annotated, annotation concepts and instances. The system also enables new individual definitions and domain specific common data integration. The newly defined and automatically integrated instances are also used during the video annotation and querying processes.

The developed system enables ontology-driven query formulation and processing according to domain ontology instances and concepts. The supported query types in the system are as follows:

- Concept Querying
- Spatio-Temporal Querying
- Region-Based Querying
- Time-Based Querying
- Compound Querying

Concept, spatio-temporal, region-based and time-based querying capabilities are regarded as simple querying types. Compound queries are formulated by combining simple queries with "(", ")", "AND" and "OR" operators. For all these query types, the system supports both general and video specific query processing. Thus, the user can pose queries on all videos in the database as well as a specific video of interest.

3.2 System Specifications

Ontology-Driven Video Annotation and Retrieval System is developed as a standard web application based on client-server architecture. Eclipse Galileo is used as the development IDE during the system development. The other technologies used for the system development can be summarized as follows:

- Server Side:
    - Java programming language
    - JENA Ontology API
    - ARQ Engine
    - SPARQL query language
    - Rhizomik MPEG-7 Ontology
    - JMF (for multimedia player applet)
- Client Side:
    - JavaServer Faces - Facelets
    - Richfaces and Myfaces components
- Apache Tomcat as web server
- Mozilla Firefox as web browser
- MySQL database

3.3 System Architecture and Components

The proposed video management system is designed with a modular architecture and it is composed of ontology management, video content annotation and query processing sub-modules. Due to the modular architecture, the sub-modules and the main components interact with each other through the defined interfaces. Moreover, the modular architecture also enables the integration and replacement of sub-modules and components with different mechanisms if necessary. In other words, a component can be easily separated from the system and replaced with a different mechanism that performs the same functionality in order to provide a more efficient working component. In addition, modularity eases the integration of new modules to the proposed system in order to allow the system gain extra functional capabilities.

The general system architecture is presented in Figure 3.3.1.

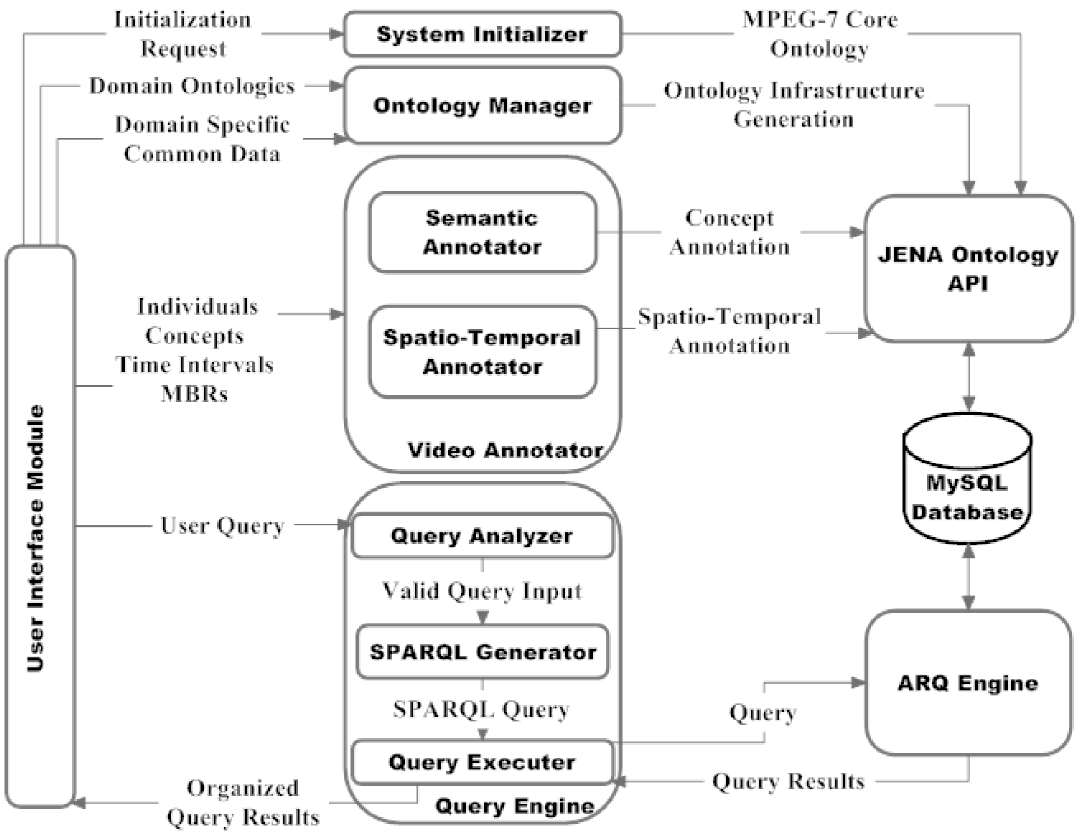


Figure 3.3.1: General System Architecture

The proposed system is mainly composed of the following components:

- MySQL Database: The database is accessed by JENA Ontology API and ARQ Engine in order to store and retrieve all ontological data. MPEG-7 and integrated domain ontologies, semantic annotations, annotated concepts, individuals, spatio-temporal annotations, object/event positions and time intervals are stored in the MySQL database. The storage and data retrieval operations are performed via JENA Ontology API and ARQ Engine interacting with the MySQL database.

- JENA Ontology API: The API is used for ontology management and video content annotation processes. MPEG-7 core ontology construction, domain ontology integration, removal, domain specific common data attachment, semantic and spatio-temporal annotation operations are performed via JENA Ontology API.

- ARQ Engine: ARQ Engine is used for SPARQL query processing and execution. Query Engine module sends the generated SPARQL query to the ARQ Engine and query results are retrieved from the database.

- System Initializer: This component initializes the system in order to provide interfaces for annotation and query operations. During initialization process, MPEG-7 core ontology is loaded to the database and set up for domain ontology integration.

- Ontology Manager: This component is responsible for ontology management operations. Domain ontology insertion, deletion and configuration procedures are handled by the Ontology Manager. Moreover, inserted domain ontologies are integrated with the core ontology and the received domain specific common data are also attached to the ontology infrastructure.

- Semantic Annotator: This component handles semantic annotation procedures. The received semantic annotation concepts, individuals and time intervals are stored in the database by interacting with JENA Ontology API.

- Spatio-Temporal Annotator: This component handles spatio-temporal annotation. Spatial and temporal relations between the annotated objects and/or events are extracted according to the minimum bounding rectangle (MBR) coordinates and time intervals. Interacting with JENA Ontology API, the received spatio-temporal annotation concepts, individuals, region coordinates, extracted spatial and temporal relations are stored in the database.

- Query Analyzer: Query Analyzer component is used for parsing and validating the received query input. If the parsed input is a valid query, its query type is determined and the query input is sent to the related SPARQL Query Generator according to its query type.

- SPARQL Query Generator: This component receives a valid query and its query type in order to convert the query to the corresponding SPARQL query. After the conversion, the generated SPARQL query is sent to the Query Executer module for query execution.

- Query Executer: This module is used to execute the generated SPARQL queries. Query Executer module interacts with ARQ Engine for query processing and execution in order to retrieve the matched ontological data from the database. The query results are then received from the ARQ Engine and organized to be displayed. The organized query results are sent to the user interface module in order to be displayed to the user.

- User Interface Module: The users interact with the User Interface Module and this module provides interfaces for ontology management, video annotation and querying. Domain ontology files and domain specific common data are supplied to the system via the User Interface Module. Video files to be annotated can be browsed and ontological concepts and individuals are selected for annotation from the user interface. Users can formulate different types of queries by using the form-based user interface module. These annotations and formulated queries are sent to the video annotator and query engine modules, respectively.

CHAPTER 4

VIDEO ANNOTATION AND MODELING

Video scenes mainly consist of individuals, objects and events. The individuals can be treated as specific object types representing the human beings. The semantic video content depends on the relationships between these video objects and events. Therefore, semantic content-based querying deals with the video objects, individuals, events and the relationships between them. Moreover, video management systems are mainly interested in video object and event modeling and these systems propose improvements on video data management issues. To generalize, video data management issues can be reduced to modeling and storage of the video objects, events and the relationships between them.

In this chapter, the video annotation and modeling component of the developed framework is introduced. The video annotation and modeling component is composed of common data integration, semantic annotation and spatio-temporal annotation sub-modules. The video annotation process makes use of domain ontology concepts in order to annotate individuals, object and events in video files. Thus, the usage of domain ontology concepts during annotation and modeling process enables ontology-driven content-based querying capability.

The rest of the chapter describes the details of the video annotation and modeling component. In section 4.1 domain specific common data integration process is explained. The semantic annotation details are given in section 4.2. Section 4.3 presents spatio-temporal annotation and data modeling. The computations of the spatial and temporal relations between video objects and events are explained.

4.1 Domain Specific Common Data Integration

The proposed system in this thesis is ontology-driven and enables the import of various domain ontologies independently. After a domain ontology is imported to the system, the imported ontology is attached to the MPEG-7 core ontology and the ontological infrastructure enables the usage of the imported domain ontology concepts during video content annotation and querying processes. The MPEG-7 ontology constructs and domain ontology attachment details are hidden from the end users and only domain ontology concepts are visible to the users in the proposed system. The instances related to these domain ontology concepts are defined and video content is annotated and queried by using the defined instances. The defined instances consist of video objects, individuals and events. The definition of the instances in the system is performed via new instance creation or common data integration processes. The interface for creating new instances provides the user with the ability to select the domain ontology and the related ontology concept. A name tag is assigned to the newly created instance in order to be accessed in later processes. In our framework, domain-specific ontologies also guide the definition of the domain specific common data that is used to describe the contents of audiovisual programs and/or their segments. Domain specific common data consists of the instances that can be commonly used in the video content of domain specific applications. For example, in a soccer tournament application the domain specific common data includes instances of the players in football teams that participate in the tournament, instances of the referees, coaches etc. These instances are reusable both in one football match as well as across several matches in the tournament.

The integration of the common data is performed automatically in the proposed framework and domain ontology entities are used for the description of common data instances. After a domain ontology is imported to the system and attached to the core ontology, the user can provide an OWL file containing the definitions of the domain specific common data entities. The provided OWL file is imported to the system via common data integration interface. During the import process, the related domain ontology is selected and the domain specific common data is integrated to the ontological model by using JENA Ontology API methods. The ontology structure, its contents and integrated common data instances are stored in the MySQL database and accessed by using JENA Ontology API and SPARQL queries in order to be used in video content annotation and retrieval procedures.

Domain specific common data definitions are performed with the use of domain ontologies. In other words, common data integration module is aware of the attached domain ontologies. Therefore, the individuals and objects to be integrated are stored as instances of the related domain ontology classes and concepts. After the integration is completed, the integrated common data instances can be used by ontology-driven concept annotation and spatio-temporal annotation modules. For instance, if the user wants to define the position of a specific football player in a soccer video, the spatio-temporal annotator module looks for all defined individuals by using Jena Ontology API. The user interface module lists the available individuals in the database according to the selected domain ontology concept. By using the form-based user interface, the end user can select the specific individual to be annotated and define the positions and time intervals that the selected individual appears.

Due to ontology awareness, if the user wants to define a new object in the video and this object is not one of the integrated common data instances, the user should define this new object first. New objects and individuals are defined via new instance creation procedure. After defining the new objects and individuals, all annotator modules can use these instances for annotation. Since domain specific common data integration provides the annotation instances directly to the annotator modules, it decreases the annotation effort and eases the annotation process. Otherwise, users would be defining all individuals and objects from new instance creation interface before annotation and querying phases. Defining all instances would be a very time consuming process, especially for large domain specific applications. As a result, domain specific common data integration capability of the system proposes an automatic and efficient instance creation mechanism that enables faster video content annotation.

4.2 Semantic Annotation

In this section we provide an overview of the semantic annotation module that supports ontology-based semantic indexing for the video visual content. The semantic annotation is performed manually by using the instances of domain ontology classes and concepts. Semantic descriptions and metadata related to the video content are stored in the relational database and JENA Ontology API provides methods for semantic metadata insertion and access operations.

The use of domain ontologies guides the users to provide legal semantic metadata for video content and supplies a common annotation platform for annotation concepts and instances. Moreover, the ontology support enhances the retrieval effectiveness and ontology-driven annotation extends the querying capability with ontological reasoning.

The architecture of the semantic annotation module is presented in Figure 4.2.1.

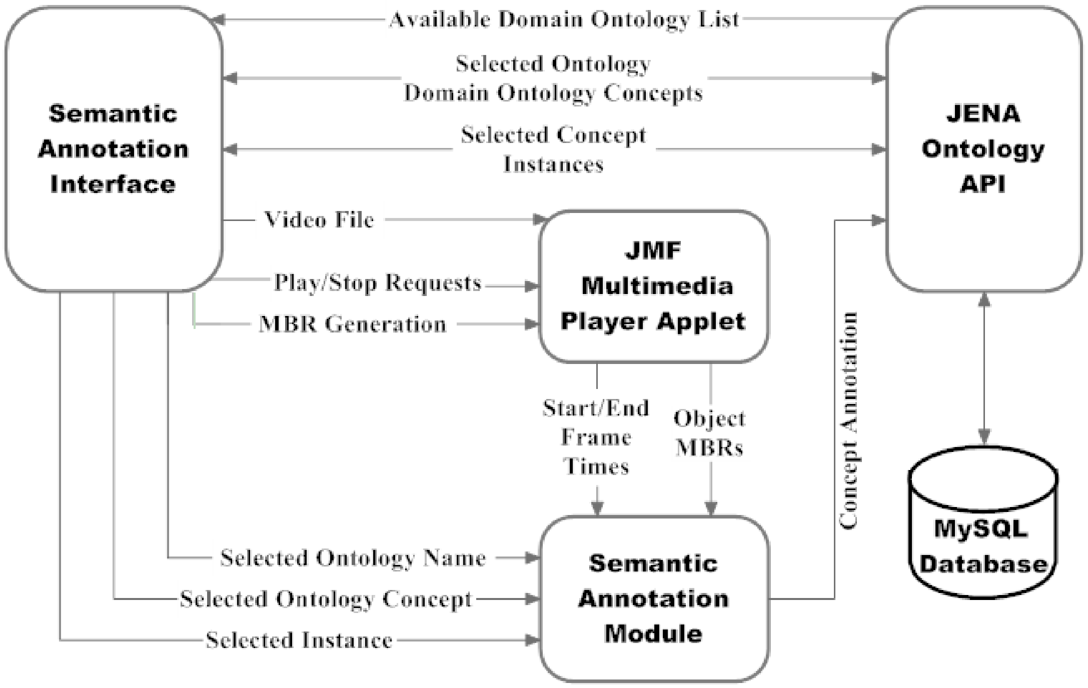


Figure 4.2.1: Semantic Annotation Module Architecture

In semantic annotation process, the instances of the domain ontology concepts and classes are annotated on the video content. The frame intervals that the objects appear on the scene and the object positions are stored for each instance in the video content. The stored instances, their positions and time intervals are used for calculating spatial and temporal relations between video objects and posing semantic content-based queries over video content. In this thesis, minimum bounding rectangle (MBR) formalization is used to annotate the positions of the video objects. The object positions for each video object are defined by drawing a rectangle covering the annotated object and the drawn rectangle is the minimum sized rectangle that contains the object of interest.

The definitions of the individual, object and event positions using MBR formalization in semantic annotation process are shown in Figure 4.2.2.

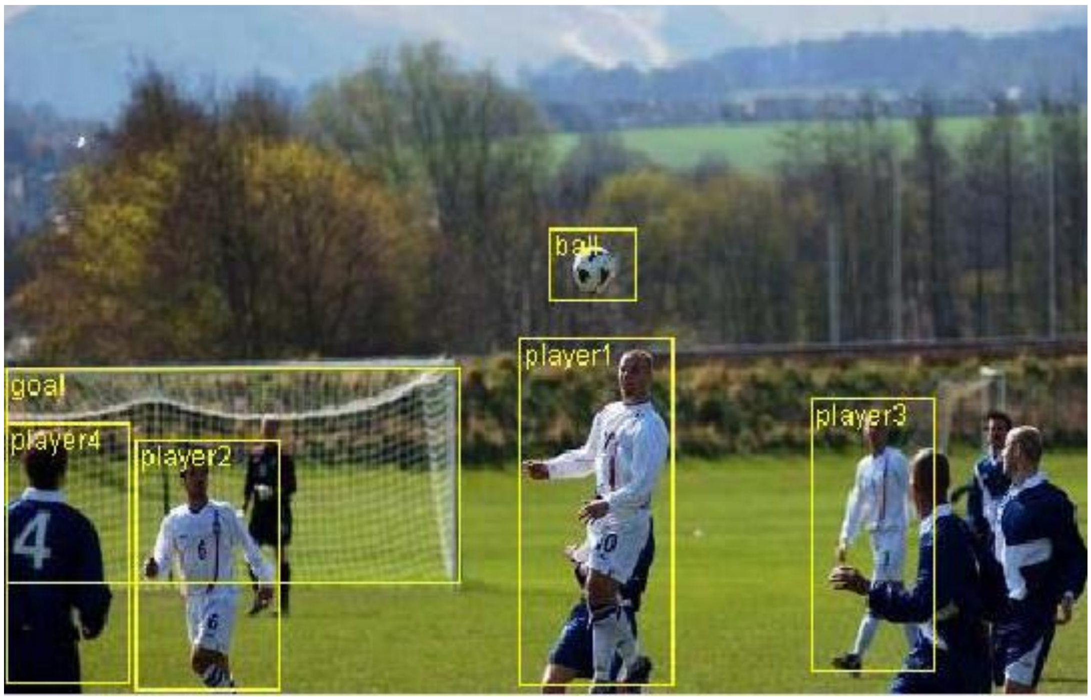


Figure 4.2.2: MBRs for Annotation Instances

The MBR of an object is used to determine the coordinates of the annotated object on the video scene. Each MBR is a rectangle with four corner points and these corner points have some pixel values which are calculated according to their distance to the origin of the coordinate axis. Figure 4.2.3 displays the object MBR representation with its four corner points.

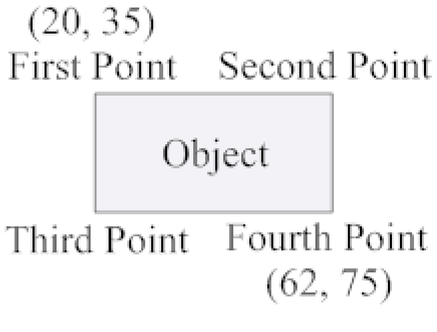


Figure 4.2.3: MBR Corner Points

In the developed system, MBRs of objects are stored with only two corner points. The first and the fourth point’s coordinates are stored and all other corner points can be inferred by using these two corner points. MBRs of objects are stored in the semantic annotation phase and all spatio-temporal relations between objects, individuals and events are calculated in the spatio-temporal annotation phase by using these coordinates. The details of using the MBR coordinates in defining spatio-temporal relations are discussed in Section 4.3.

During semantic annotation, the end users interact with the semantic annotation interface in order to store object appearance time intervals and positions. At the beginning of the annotation session, the user is expected to specify the domain ontology to be used from the available ontologies attached to the system. If the domain ontology that will be used for annotation is not supported by the system, the ontology import operation is carried out via the ontology management module and the imported ontology is attached to the system’s ontology infrastructure. After the domain ontology import operation is finalized, domain specific common data integration process is performed and common data instances are stored in the relational database. After selecting the domain ontology from the list of the available domain ontologies, the semantic annotation interface module interacts with JENA Ontology API methods in order to retrieve the related domain ontology classes and concepts. The user selects one of the ontology concepts and the semantic annotation interface module retrieves the instances of the selected ontology concept e.g. individuals, objects or events. The retrieved instances consist of the defined individuals, objects or events and integrated domain specific common data instances. After selecting the instance of concern from the ontology, the individual, object or event to be annotated is specified. In this way, the first step of the semantic annotation process is finalized.

In the next step, the user browses the video files in the video database and selects one of them for annotation. The selected video is loaded by the JMF player and the player presents playing and stopping options for video files to the users. The user pushes the play button and the video scenes are started to be displayed. When the individual, object or event to be annotated appeared, the user pushes the stop button and the video is stopped. The video time when the player is stopped is taken as the start time of the annotation object frame interval since object of interest appears at this time. The user draws the minimum bounding rectangle for the selected annotation instance and plays the video again. The video is played until the object of interest disappears or leaves the minimum bounding rectangle borders. The user stops the video as soon as the annotated instance leaves the MBR borders or disappears. This stop time is taken as the end time of the annotation object frame interval. This manual annotation procedure is repeated, if necessary, for other appearances of the same instance or the other instances existing in the video content.

The output of the semantic annotation procedure is the metadata describing the appearances of instances. The obtained data for each instance appearance consists of the following attributes:

- Selected domain ontology name,
- Selected domain ontology concept,
- Selected individual, object or event,
- Start frame time,
- End frame time,
- MBR coordinates,
- Video file path.

In the last step of the semantic annotation process, the instance appearance data is stored in the relational database in order to be used for semantic content-based querying and retrieval. For this purpose, the semantic annotation module interacts with JENA Ontology API; and the instance appearance data is stored into the ontological model residing in MySQL database by using the methods provided by JENA. For each domain ontology which is used for annotation, a separate ontological model is created and stored in the database. The instance appearance data constructed for the instances of domain ontology concepts is stored into the related ontological model. In the querying process, user queries are executed on one ontological model since query input formulations contain the selected domain ontology information. All JENA specific retrieval requests and SPARQL queries deal with only one ontological model at a time. This kind of storage strategy provides a more efficient query processing mechanism.

4.3 Spatio-Temporal Annotation

The semantics of media data is derived from the interaction of the media objects in space and time. Complex media events are also characterized by spatio-temporal relations of its media objects. For instance, a “goal scored” event in a soccer game can be characterized by “the ball *inside* goal-box *precedes* cheer”. In this example, assuming that a "ball", a "goal-box" and "cheer" are recognized by audiovisual annotation and recognition techniques, the “goal scored” event can be derived by analyzing their spatial and temporal relations, namely *inside* and *precedes* [42]. This brings the requirement for complete representation of the spatio-temporal ordering and relationships between media objects.

In this section we provide an overview of the spatio-temporal annotation module that supports ontology-based spatio-temporal modeling for the video content. Spatio-temporal annotation deals with the interactions of the annotated video instances in space and time. Spatio-temporal modeling is a crucial step in video retrieval systems for using semantic information on image object relationship to improve the quality of content-based video retrieval. Such information can be used for tagging the video content and thus, performing spatio-temporal querying on video databases [45]. For this purpose, a spatio-temporal model should suggest a practical solution for an effective indexing and retrieval. In addition, combining spatio-temporal information with domain ontology knowledge enhances spatio-temporal querying capability. Therefore, the developed system provides a spatio-temporal modeling mechanism supporting ontology knowledge.

Spatio-temporal relations provided by the spatio-temporal annotation module are composed of spatial and temporal relations according to the relationship definitions.

Spatial relations describe the relative positions of the annotated individuals, objects and events, whereas temporal relations describe the frame appearances of the annotated instances with respect to each other [28, 37]. The details of the supported spatial and temporal relations are explained in the following sub-sections.

4.3.1 Spatial Relations

In a given frame, the spatial relations between two instances are defined using the spatial relationship between the MBRs of each instance. This property gives us the ability to retrieve the spatio-temporal relations between any two objects in a frame sequence [38]. The retrieved spatial properties of annotated instances are used to define the spatio-temporal relations between them in a video stream. Li et al. have proposed a formal definition of these relations for a rule base [46]. They extended Allen's temporal interval algebra [47] into two-dimensional space in order to define spatial relations between rectangular areas. According to the proposed approach, the spatial relations are divided into two groups, namely directional and topological. In this thesis, the developed framework supports positional relations in addition to directional and topological relations. Therefore, spatial relations are categorized as directional, positional and topological relations.

Directional relations include strict directional and mixed directional relations. These relations represent the basic directions of a compass. There are four strict directional relations; South, North, West and East. There are also four mixed directional relations; Northwest, Northeast, Southwest, and Southeast [37,38]. These directional relations are presented in Figure 4.3.1. The directional relations between the instances are calculated according to the center points of the MBRs. If x and y coordinates of the MBR center points are equal, no directional relation can be computed between these objects. However, if the objects have different MBR center points, the spatial relations of the center points relative to each other according to the eight directional coordinate axes represent the directional relation between objects.

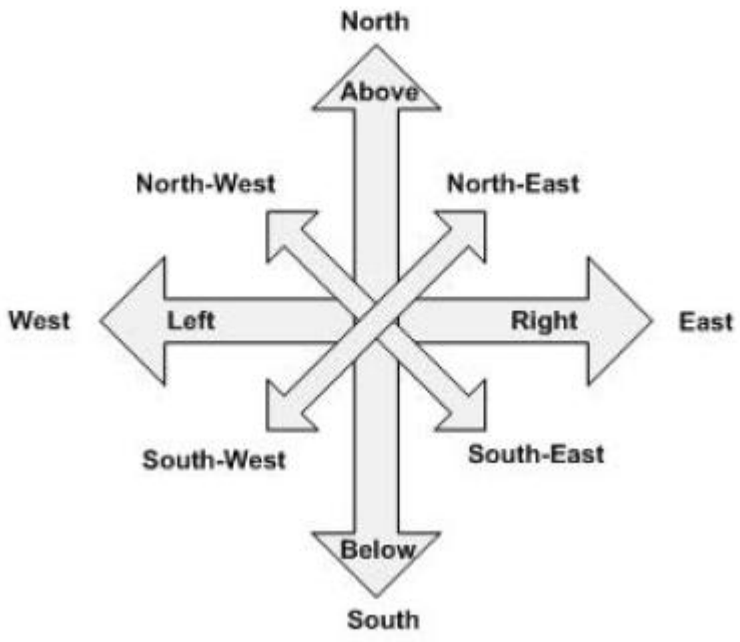


Figure 4.3.1: Directional and Positional Relations

Positional relations are computed with respect to directional relations. There are four positional relations: Left, Right, Above, Below. Figure 4.3.1 also shows positional relations. Despite the equivalent directional relations, positional relations are also supported by the developed framework since they are commonly used in spatio-temporal user queries. These positional relations cover both strict directional and mixed directional relations (Figure 4.3.2).

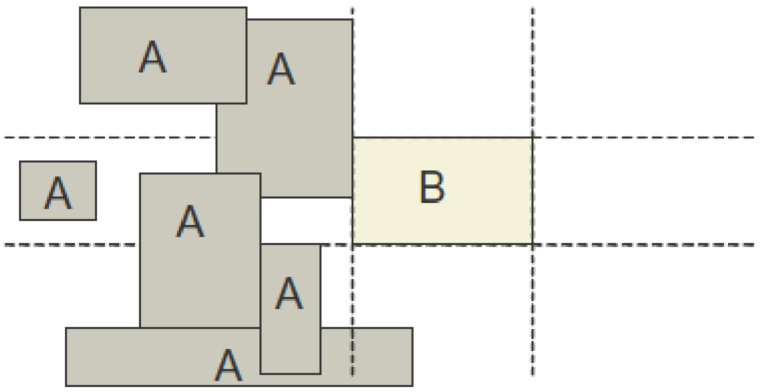


Figure 4.3.2: Left Relation Example

Directional and positional relations address where the annotated individuals and objects are placed relative to each other, express relative orientation of two instances and describe spatial ordering along the x and y directions. Thus, these relations are defined according to point-based representation depending on the comparison of two objects' center points. On the other hand, topological relations address how the boundaries of two annotated instances relate, express topological extension of two object boundaries, and describe neighborhood and incidence [45]. These relations have interval-based representation and they are used to represent the relationships between object boundaries. There are eight different topological relations which include Equal, Contains, Inside, Overlaps, Cover, Covered-By, Touch and Disjoint relations.

The object boundary relationships for these eight topological relations are represented in Figure 4.3.3 [37].

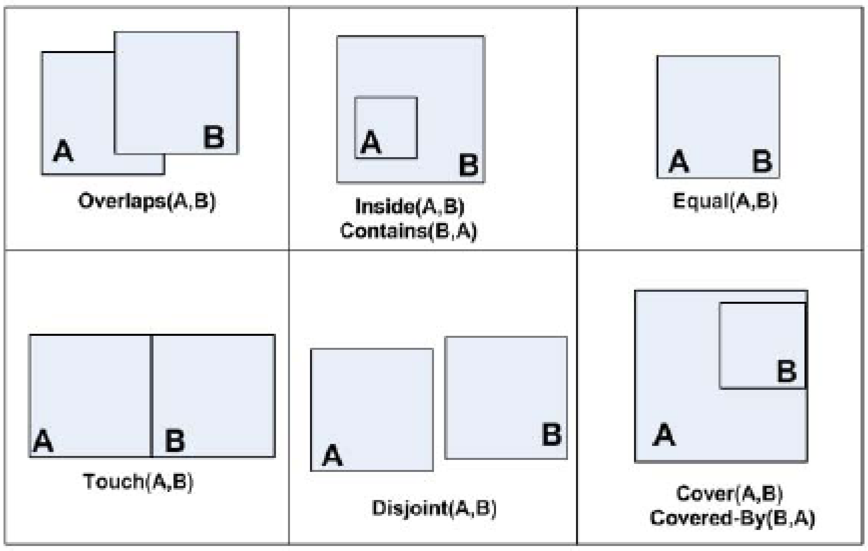


Figure 4.3.3: Topological Relations

The relationships Overlaps, Equal, Touch and Disjoint are symmetric relations which means both (A, B) and (B, A) tuples satisfy the relation. For instance, if the boundary of object A overlaps with the boundary of object B, both Overlaps(A, B) and Overlaps(B, A) are satisfied according to the symmetry property.

4.3.2 Temporal Relations

Video events and object appearances change over time and this brings the requirement for studying temporal relations between object appearances and events. Allen [47] modeled an event as an interval in time and established temporal relations between the events by comparing their start and end times. Allen identified thirteen distinct temporal relations which exhaustively represent all possible combinations of relative start and end times of the events [42,47]. The temporal interval algebra proposed by Allen consists of the following relations: Precedes, Meets, Overlaps, Starts, Finishes, Contains, Equals, During, Finished-By, Started-By, Overlapped-By, Met-By and Preceded-By. Except Equals relation, the first six temporal relations are inverse relations of the last six temporal relations [37]. Although, Allen has proposed these relations with respect to time, they can be generalized to any space dimension.

These temporal relations are summarized in Figure 4.3.4, where A and B represent the object appearance or event time intervals [42].

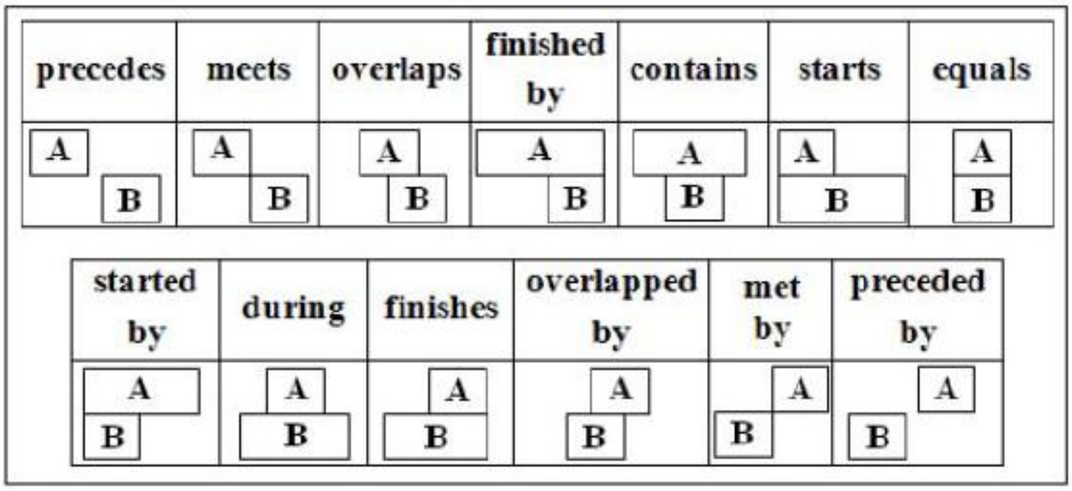


Figure 4.3.4: Temporal Relations

The temporal relationship Equals is a symmetric relation similar to the symmetric spatial relations. Equals(A, B) relation derives Equals(B, A) relation.

4.3.3 Spatio-Temporal Relation Extraction

Spatio-temporal relations between annotated individuals, objects and events are extracted during the spatio-temporal annotation process. Spatial relations are specified according to the object MBRs whereas temporal relations are computed by using frame time intervals in which the objects appear or events occur. As mentioned before, both object MBRs and frame time intervals are stored during semantic annotation. In spatio-temporal annotation, there is no extra annotation effort spent by the users. Thus, the spatio-temporal annotation module uses the semantic annotation input in order to extract spatial and temporal relations between annotated instances.

After the semantic annotation of a video is completed, the spatio-temporal annotation process starts. In most of the cases, the extraction of the spatio-temporal relations between annotated instances are performed only once for each video and used for querying.

During spatio-temporal annotation, first, spatial relations are computed and then the calculation of temporal relations is performed. Spatial relations are computed according to the MBR center points of the annotated instances. The relative positions of the MBR center points with respect to the directional coordinate axes are used to compute the directional relations between two instances. In order to specify the directional relation type, an imaginary line that combines the centers of two instances is constructed and the angles between this imaginary line and x and y coordinate axes are calculated respectively. According to the calculated angle values relative to the horizontal and vertical axes, the angular proximity to the eight directional coordinate axes determines the directional relation type.

The directions and the imaginary line drawn in orange color that combines the MBR center points of two objects A and B are shown in Figure 4.3.5. By comparing the calculated angles and directional angles between axes, it is derived that object A is in south-west direction of object B.

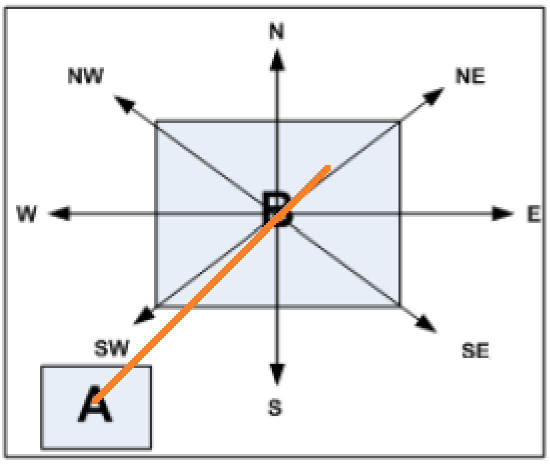


Figure 4.3.5: Directional Relation Extraction

All of these directional relations have their counter relations. In other words, North - South, West - East, Southwest - Southeast and Northwest - Northeast relation couples are inverse relations. For instance, if North(A, B) relation is satisfied, South(B, A) relation is also inferred automatically. For the above example, Southwest(A, B) relation derives Northeast(B, A) relation. Figure 4.3.6 shows examples of West - East and Northwest - Southeast inverse relations.

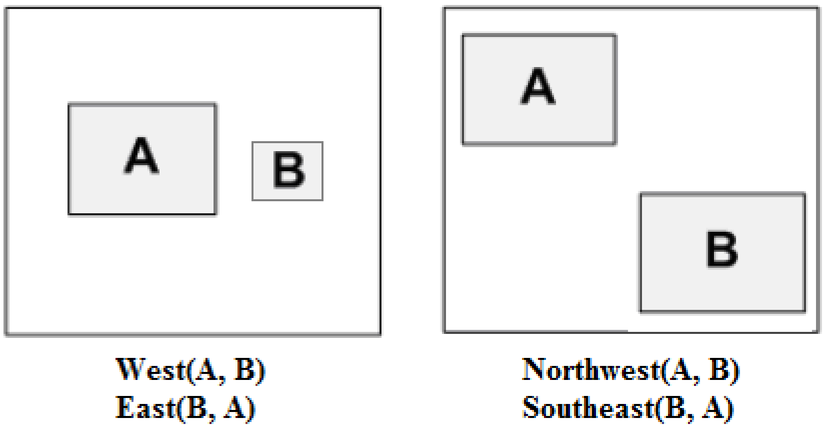


Figure 4.3.6: Inverse Directional Relation Samples

Positional relations are computed by using directional relations. These relations are supported for enhancing the system’s spatial querying capability since positional expressions are more commonly used in user queries. The computations of the positional relations are given as follows:

Above(A, B) ≡ North(A, B) ‖ *Northwest(A, B)* ‖ *Northeast(A, B).*

Below(A, B) ≡ South(A, B) ‖ *Southwest(A, B)* ‖ *Southeast(A, B).*

Left(A, B) ≡ West(A, B).

Right(A, B) ≡ East(A, B).

Topological relations between annotated instances are computed by using MBR boundaries of the instances. The eight topological relationships presented in Figure 4.3.3 are formulated according to the inner areas and border lines of the instances. In the developed system, two operators, namely *Inner* and *Border* are defined for the extraction of topological relations. *Inner* operator represents the inner area of the related object MBR, whereas *Border* operator represents the border lines surrounding the related object MBR.

Table 4.3.1 [48] represents the formalization of topological relations according to *Inner* and *Border* operators.

Table 4.3.1: Topological Relation Formalization

	Border(A) && Border(B)	Border(A) && Inner(B)	Inner(A) && Border(B)	Inner(A) && Inner(B)
Overlaps	≠ Ø	≠ Ø	≠ Ø	≠ Ø
Equal	≠ Ø	Ø	Ø	≠ Ø
Inside	Ø	≠ Ø	Ø	≠ Ø
Contains	Ø	Ø	≠ Ø	≠ Ø

Table 4.3.1 Continue

Touch	$\neq \emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
Disjoint	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
Cover	$\neq \emptyset$	$\emptyset$	$\neq \emptyset$	$\neq \emptyset$
Covered-By	$\neq \emptyset$	$\neq \emptyset$	$\emptyset$	$\neq \emptyset$

As mentioned before, the relations Overlaps, Equal, Touch and Disjoint are symmetric relations. On the other hand, Cover - Covered-By and Inside - Contains relation couples are inverse relations. Therefore, if one side of the following statements is satisfied, the other side is automatically deduced.

Overlaps(A, B) ↔ Overlaps(B, A)

Equal(A, B) ↔ Equal(B, A)

Touch(A, B) ↔ Touch(B, A)

Disjoint(A, B) ↔ Disjoint(B, A)

Cover(A, B) ↔ Covered-By(B, A)

Inside(A, B) ↔ Contains(B, A)

In the developed framework, temporal relationships between annotated objects, individuals and events are extracted according to the frame appearance time intervals of the instances. For each instance couple, their start and end times are compared with respect to each other in order to determine the temporal relationship between the related instances. For instance, according to the definition of *Overlaps* relation, the following condition should be satisfied between the appearance time intervals of objects A and B.

Overlaps relation condition: $\mathbf{(A_{start} < B_{start}) \cap (A_{end} < B_{end}) \cap (A_{end} > B_{start})}$

On the other hand, for *Precedes* relation, the condition to be satisfied is $\mathbf{A_{end} < B_{start}}$ whereas *Starts* relation criterion is $\mathbf{(A_{start} = B_{start}) \cap (A_{end} < B_{end})}$.

Figure 4.3.7 displays the temporal orderings for objects A and B satisfying *Overlaps*, *Precedes* and *Starts* relations respectively.

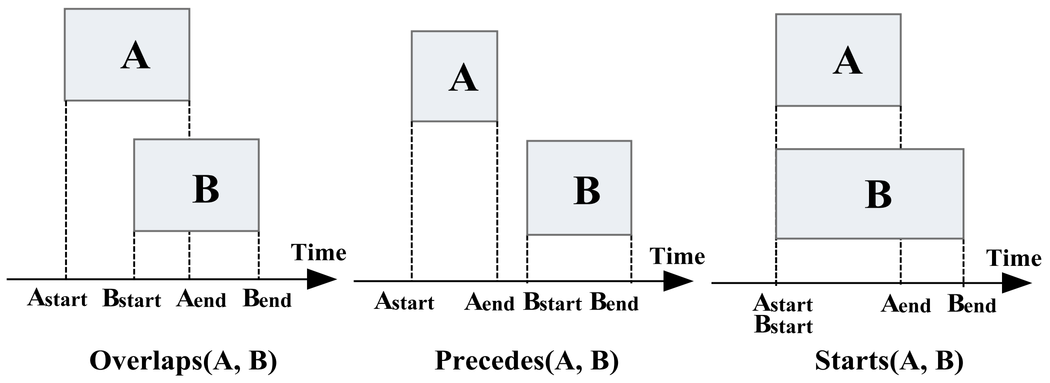


Figure 4.3.7: Temporal Relation Examples

In the set of temporal relations, Equals relation is a symmetric relation whereas Precedes - Preceded-By, Meets - Met-By, Overlaps - Overlapped-By, Starts - Started-By, Finishes - Finished-By, Contains - During relation couples are inverse relations. Therefore, if one side of the following statements is satisfied, the other side is automatically deduced.

Equals(A, B) ↔ Equals(B, A)

Precedes(A, B) ↔ Preceded-By(B, A)

Meets(A, B) ↔ Met-By(B, A)

Overlaps(A, B) ↔ Overlapped-By(B, A)

Starts(A, B) ↔ Started-By(B, A)

Finishes(A, B) ↔ Finished-By(B, A)

During(A, B) ↔ Contains(B, A)

4.3.4 Spatio-Temporal Data Modeling

The spatio-temporal annotation process includes the extraction and storage of spatial and temporal relations between annotated video instances. Moreover, the annotation module should be aware of the domain ontology knowledge in order to support ontology-driven querying capability. Therefore, the extracted spatial and temporal relations needed to be stored with the ontological information.

The previous studies conducted on spatio-temporal modeling and query processing proposed a rule base approach [28,29,37]. According to the rule base approach, the annotated information is stored in the knowledge base as basic facts and inference rules. Basic facts represent only necessary facts which are annotated by the semantic annotator module and inference rules represent reasoning rules among spatio-temporal relations. These inference rules are formalized by Allen's temporal algebra and spatial logic [28,37]. In this rule base approach, only a pruned set of facts are stored in the knowledge base. Other non-stored relations are extracted by using the inference rules during query processing [28,37]. This approach provides space efficiency by optimizing the system space. The usage of inference rules enables to prune spatio-temporal relations in annotation phase since most of the relations have inverse and symmetry properties and these relations can be derived later in querying phase. However, in a video retrieval system, a time efficient query processing and result generation mechanism is much more important than space efficiency requirements. In order to decrease the query response time, the computations and spatio-temporal relation extractions in query processing phase should be reduced. Moreover, in modern computer systems space specifications are more adequate and proposing a time efficient solution is encouraged.

In the developed system, the spatio-temporal annotation module stores all extracted relations instead of a pruned set of relations into the relational database by using JENA Ontology API methods. According to the proposed approach, extra effort spent for the computation of the spatio-temporal relations in query processing time is disposed. By this means, the query processing and result retrieval process is performed in a time efficient manner. On the other hand, interacting with JENA Ontology API, the extracted relations are stored into the ontological model existing in the relational database. Thus, the spatio-temporal annotations are stored with ontological data being aware of the domain ontology knowledge which enhances ontology-based spatio-temporal querying capability. The architecture of the spatio-temporal annotation module is presented in Figure 4.3.8.

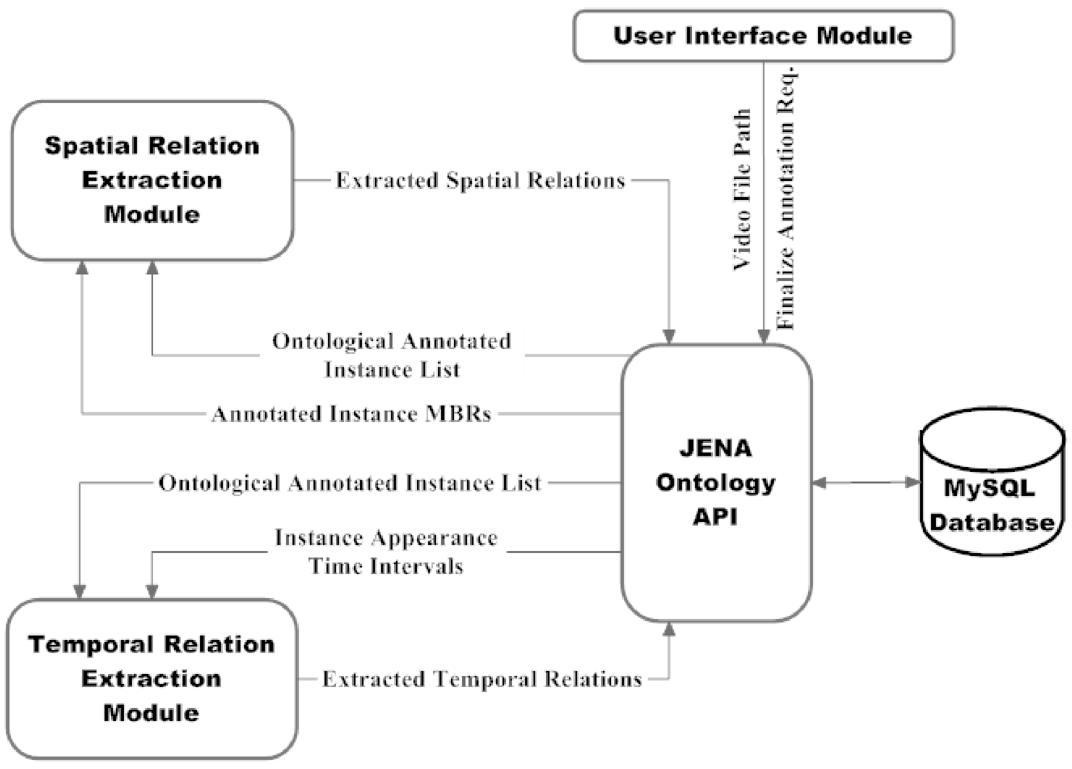

User Interface Module
Spatial Relation Extraction Module
Extracted Spatial Relations
Video File Path
Finalize Annotation Req.
Ontological Annotated Instance List
Annotated Instance MBRs
JENA Ontology API
MySQL Database
Ontological Annotated Instance List
Instance Appearance Time Intervals
Temporal Relation Extraction Module
Extracted Temporal Relations

Figure 4.3.8: Spatio-Temporal Annotation Module Architecture

For each annotated video, the spatio-temporal annotation process is started by Finalize Annotation Request from the user interface. The spatio-temporal annotation module consists of spatial and temporal relation extraction sub-modules. These sub-modules communicate with JENA Ontology API and retrieve the annotated instances in the specified video file, MBR coordinates and appearance time intervals. By using MBR coordinates, the spatial relation extraction module computes spatial relations between annotated instances. On the other hand, temporal relation extraction module computes temporal relations by using instance appearance time intervals. The calculated spatial and temporal relations are stored with the ontological data of the instances via JENA Ontology API. Thus, spatio-temporal annotation is performed and ontology-driven spatio-temporal querying capability is provided.

CHAPTER 5

QUERY PROCESSING

In OntoVARS, an ontology-driven video content querying capability is provided to the end users. The integrated domain ontology concepts are used to formulate queries on the semantic video content. The usage of domain specific ontologies improves querying capability and content-based retrieval performance by providing legal content descriptions for domain specific audio-visual content to the users. In addition, ontological querying enables content-based queries on all members of a domain ontology class as well as each annotated instance. For instance, it is possible to query the scenes where any home team player appears in the left of the goalkeeper when a goal event is occurred. Similarly, the same query can also be performed for a specific player or goalkeeper.

In the developed framework, it is possible to query the content of either a specific video or multiple videos. In multiple video querying, the formulated queries via a form-based query interface are executed on the whole video database and the formulated query can match various video scenes from different video files. All these matching scenes from multiple videos are returned to the users as the query result. In video specific querying, the formulated user queries are executed on the selected video. All matching scenes from the selected video file are returned to the user as the query result. Multiple video querying provides a comprehensive video content search on the whole video database whereas the video specific querying capability enables a more efficient video content retrieval for a specific video.

This chapter is organized as follows: In Section 5.1, the supported query types in the developed system are explained. In Section 5.2, the general architecture of OntoVARS query engine is mentioned. The details of the query processing mechanism are discussed in section 5.3 and finally, some illustrative examples are shown in Section 5.4.

5.1 OntoVARS Query Types

In OntoVARS, simple query types provide ontology-driven concept querying, spatio-temporal querying, region-based and time-based querying capabilities. In addition, compound queries are also supported in order to enhance the querying capability of the system.

5.1.1 Concept Querying

Concept queries are used to retrieve the scenes related to the individuals, objects or events of interest in video files. According to the ontology support, concept querying enables the search of selected domain ontology concepts in order to retrieve the instances of the selected concepts. For instance, in the proposed system the following queries can be formulated as concept query type:

"Return all scenes where Meg Ryan appears."

"Return frame intervals where goal event occurs."

"Return frame intervals where a carnivore animal is seen."

5.1.2 Spatial Querying

Spatial queries are formulated in order to query the object positions and spatial orientations of the instances relative to each other. Spatial querying capability allows the users to pose queries about directional, positional and topological relations between instances. In OntoVARS, the following queries can be formulated as spatial query type:

"Return all scenes where a road is in the west of a house." (Directional Relation)

"Return frame intervals where a male appears to left of a female." (Positional Relation)

"Return frame intervals where the ball is inside goal area." (Topological Relation)

5.1.3 Temporal Querying

Temporal queries are formulated in order to query the object appearances and temporal relations between individuals, objects and events. Using temporal query type, the users can pose the following queries:

"Return all scenes where a goal event is preceded by a penalty event."

"Return cars which appear overlapping with a tree."

5.1.4 Time-Based Querying

Time-based queries are used to query object appearances and event occurrences with respect to specific time intervals. There are four time specific relations: Before, After, Contain and Between which are used to formulate time-based queries. The user enters a frame or time interval and the object appearances and event occurrences according to the specified frame times and selected relation are queried. Time-based queries are used to restrict the query result set and enhance retrieval efficiency if the time intervals of the queried objects or events are known approximately. The following queries can be given as examples for the time-based query type:

"Return frames where a goal event occurs between time interval [15, 25]."

"Return mammals where a mammal instance appears before frame time 12.4."

5.1.5 Region-Based Querying

Region-based queries are used to query individual, object and event locations according to a specified region of the video files. Before formulating the region-based queries, the users specify a query region by using the mouse motion coordinates over the video player applet. The positions of the queried instances and events that are located in the specified region are retrieved as query results. Region-based queries also restrict the result set and are used efficiently when the user deals with the objects or events in a specific region of the video files.

5.1.6 Compound Querying

In OntoVARS, besides simple query types, compound queries are also supported to enhance the querying capability of the system. Compound queries are formulated by combining concept, spatial, temporal, time-based and region-based queries. These simple queries are combined with "(", ")", "AND" and "OR" operators in order to generate compound queries. By the help of compound query support, the users can query more complex and comprehensive situations on video databases. For instance, "Return all football players who appear in the right of the goalkeeper when the penalty event occurs and they precede a goal event and the goal event occurs between time interval [25.0, 45.0]." is a legal compound query that can be formulated in the developed framework. This sample query is composed of four different simple queries. The first query is a spatial query; "Return all football players who appear in the right of the goalkeeper." In this query, ontological football player class instances are examined according to their spatial positions with respect to the goalkeeper class instances. The second query is a concept query; "When penalty event occurs" and retrieves the scenes in which a penalty event occurs. The third query; "Return all football players who precede a goal event." is a temporal query and retrieves the football players that appear before a goal event. Finally the fourth simple query; "Return the goal events occurred between time interval [25.0, 45.0]." is a time-based query used to examine the goals scored in the given time period. After executing these four different types of simple queries, Union or

Intersection operations are applied to the query results in order to combine the results retrieved from different query types.

5.2 OntoVARS Query Engine Architecture

In OntoVARS, the formulated user queries are processed and the query results are generated by OntoVARS Query Engine. The query engine interacts with a form-based user interface and processes the queries that are sent through the query interface. The general architecture of OntoVARS query engine is shown in Figure 5.2.1.

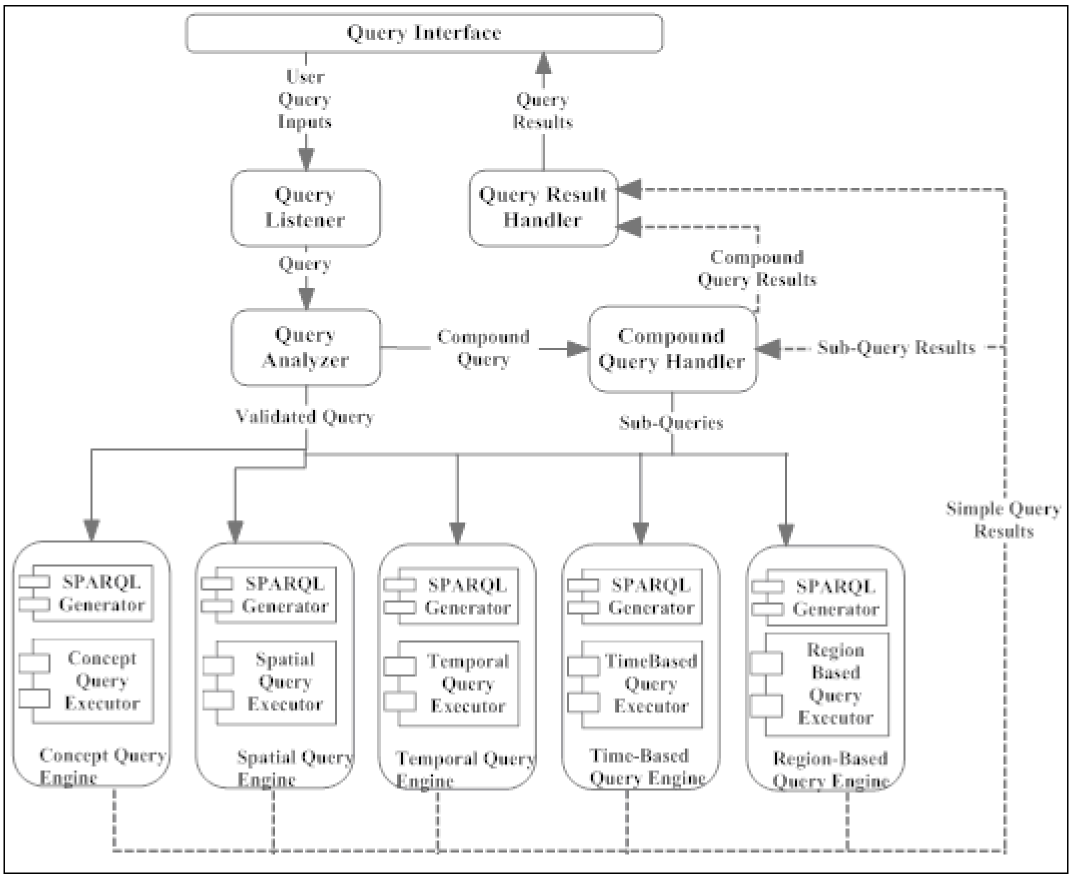


Figure 5.2.1: OntoVARS Query Engine Architecture

The query engine consists of the following components:

- Query Listener: This component is responsible for retrieving the user query from the query interface. When a query is formulated by using OntoVARS GUI components, Query Listener retrieves the values of the query parts from GUI components and sends the formulated query to the Query Analyzer module.

- Query Analyzer: Query Analyzer component is used for parsing and validating the query inputs received from the query interface. During the query parsing process, this module decomposes the query into sub-queries in case it may contain multiple queries (i.e., the query is a compound query). If the parsed query input is a valid simple query, its query type is determined and the query input is sent to the related SPARQL Query Generator component according to its query type. In case of a valid compound query input, a compound query structure is generated consisting of its sub-queries and sub-query types and sent to Compound Query Handler module.

- SPARQL Query Generator: This component receives a valid query input and its query type in order to convert the query input to the corresponding SPARQL query. After the conversion, the generated SPARQL query is sent to the related Query Executor module for query execution.

- Compound Query Handler: This component retrieves the compound query structure from Query Analyzer component and sends the sub-queries of the received compound query to the related query executor modules. After the executions of the sub-queries are finalized, the query results of the sub-queries are returned to the Compound Query Handler. Then, this component evaluates the final query result by applying union and intersection operations on the sub-query results and the evaluated final query result is sent to the Query Result Handler component.

- Concept Query Executor: This component executes the received SPARQL query that is generated according to the concept query input. After the query execution is completed, the concept query result is sent to Query Result Handler component.

- Spatial Query Executor: This component executes the received SPARQL query that is generated according to the spatial query input. After the query execution is completed, the spatial query result is sent to Query Result Handler component.

- Temporal Query Executor: This component executes the received SPARQL query that is generated according to the temporal query input. After the query execution is completed, the temporal query result is sent to Query Result Handler component.

- Time-Based Query Executor: This component executes the received SPARQL query that is generated according to the time-based query input. After the query execution is completed, the time-based query result is sent to Query Result Handler component.

- Region-Based Query Executor: This component executes the received SPARQL query that is generated according to the region-based query input. After the query execution is completed, the region-based query result is sent to Query Result Handler component.

- Query Result Handler: The query results are retrieved from the related query executor modules and displayed to the end users by Query Result Handler component. The collected query results are transformed to the query display format in order to be displayed in a user understandable way.

5.3 OntoVARS Query Processing

OntoVARS query processor supports ontology-driven querying capabilities for different query types. The query processing mechanism includes three main phases: query analysis, SPARQL generation and query execution. During the query analysis phase, the user query input is parsed and validated according to its query type. If the query input is valid, a query structure is constructed and transferred to the SPARQL generation phase. In SPARQL generation phase, the received query structure is used to generate the corresponding SPARQL query for execution. After SPARQL query construction is finalized, the generated SPARQL query is sent to the related query execution module. During the execution of the query, the query processor communicates with the relational database via ARQ Engine in order to retrieve the query results. At the end of the query execution phase, these retrieved query results are sent to the Query Result Handler in case the executed query is a simple query. If the executed query is part of a compound query, the query results are transferred to the Compound Query Handler where the final query result is computed by using the sub-query results. The details of the query processing mechanism are explained in the following sections.

5.3.1 Query Analysis

Query analysis process parses the received query input and constructs a query structure for valid queries. Query analysis mechanism for simple queries differs from the analysis process for compound queries. A formulated simple query is retrieved with its query type and data. The query data is provided through the query interfaces and the system query analyzer validates the provided data according to the query type. If one of the required fields is invalid or not supported in the query data, the simple query is found to be invalid and eliminated. For instance, domain ontology model name should be valid for all simple query types. Or, the time interval values are required to be supported during query formulation for valid time-based queries. If the query input satisfies the parser checks, the query analyzer validates the query input and generates a query structure for the execution. The generated query structure consists of the query type and the provided query data such as the selected domain ontology name, domain ontology concept, ontological instance, video file name for a specific video query, selected relation for spatio-temporal or time-based queries, etc. The query structure constructed in the query analysis phase is used to generate a corresponding SPARQL query for the execution.

In the compound query analysis process, the query analyzer is responsible for parsing the retrieved query, dividing it into its sub-queries and constructing a query parse tree for execution. Before dividing into sub-queries, the compound query string is checked for structural formatting errors. If the format of the compound query is valid, the query analyzer divides the compound query into sub-queries. At the end of the compound query analysis, a query tree is constructed by using the generated sub-queries and the query operators. The sub-queries form the leaf nodes of the generated query tree where as the operators are the parent nodes. The constructed query tree is transferred to the Compound Query Handler module for execution.

5.3.2 SPARQL Generation

SPARQL generation process receives a validated query structure constructed in the query analysis phase in order to convert the query input to the corresponding SPARQL query. By using the query type in the query structure, SPARQL generator accesses the related query data fields and generates a SPARQL query according to these query data fields and query type. After the conversion operation is finalized, the generated SPARQL query is sent to the related Query Executor module for query execution.

5.3.3 Query Execution

Query execution process takes place in five different sub-modules that are specific to the query type. These five sub-modules share the same infrastructure for performing common tasks. These common tasks include ARQ Engine connectivity, which is used for connecting to the relational database and executing the generated SPARQL query, and the query result generation, which is used for parsing the result set structure retrieved from the relational database when SPARQL query is executed. The details of the query execution for each query type are explained in the following sections.

5.3.3.1 Concept Query Execution

The concept query executor module deals with the execution of the concept queries. In the query analysis phase, the retrieved concept query input is translated into a validated concept query structure in the following form:

Concept Query Structure:

Query Type

Domain Ontology Name

Domain Ontology Concept

Instance Name

Video File Path

This concept query structure is used to generate the corresponding SPARQL query in SPARQL generation phase. For instance, the details of the concept query execution phases are explained in the following query examples.

Example: Show all frame intervals where a male John appears.

Concept Query Structure:

Query Type = Concept

Domain Ontology Name = Person

Domain Ontology Concept = Male

Instance Name = John

Video File Path

By using the validated query structure, SPARQL generator outputs the following SPARQL query for execution.

```
SELECT  ?startTime  ?duration  ?mediaUri
WHERE
{
<http://www.sw-app.org/Person#John>
        <http://rhizomik.net/ontologies/2005/03/Mpeg7-2001.owl#MediaLocator>
        ?mediaLocator .
?mediaLocator
        <http://rhizomik.net/ontologies/2005/03/Mpeg7-2001.owl#MediaIncrDuration>
       ?duration .
?mediaLocator
     <http://rhizomik.net/ontologies/2005/03/Mpeg7-2001.owl#MediaRelIncrTimePoint>
     ?startTime .
?mediaLocator
     <http://rhizomik.net/ontologies/2005/03/Mpeg7-2001.owl#MediaUri>
     ?mediaUri .
 }
```

Example: Show all scenes where a carnivore animal is seen in video file AnimalDocumentary.mpg.

Concept Query Structure:

Query Type = Concept

Domain Ontology Name = Animal

Domain Ontology Concept = Carnivore

Instance Name

Video File Path = AnimalDocumentary.mpg

The following SPARQL query is constructed according to the validated concept query structure.

```
SELECT  ?animal  ?startTime  ?duration
WHERE
{
?animal  a  <http://www.sw-app.org/Animal#Carnivore> .
?animal
        <http://rhizomik.net/ontologies/2005/03/Mpeg7-2001.owl#MediaLocator>
        ?mediaLocator .
?mediaLocator
        <http://rhizomik.net/ontologies/2005/03/Mpeg7-2001.owl#MediaIncrDuration>
        ?duration .
?mediaLocator
     <http://rhizomik.net/ontologies/2005/03/Mpeg7-2001.owl#MediaRelIncrTimePoint>
     ?startTime .
?mediaLocator
     <http://rhizomik.net/ontologies/2005/03/Mpeg7-2001.owl#MediaUri>
     ?mediaUri .
FILTER regex(?mediaUri, "AnimalDocumentary.mpg","i") .
 }
```

The concept query executor receives the generated SPARQL query as an input for execution. In order to execute the query, this module interacts with ARQ Engine that provides relational database connectivity for the execution of SPARQL query. After SPARQL query is executed, a result set is obtained. The result set contains the query results in a table-like manner and each row corresponds to a set of bindings that satisfies the conditions of the query. The obtained result set is parsed and a query result list is constructed by using each query solution in the result table. At the end of this process, the constructed query result list is passed to Query Result Handler module which converts the query results to the query display format and displays them to the users in the user interface.

Figure 5.3.1 represents OntoVARS concept query execution process in details.

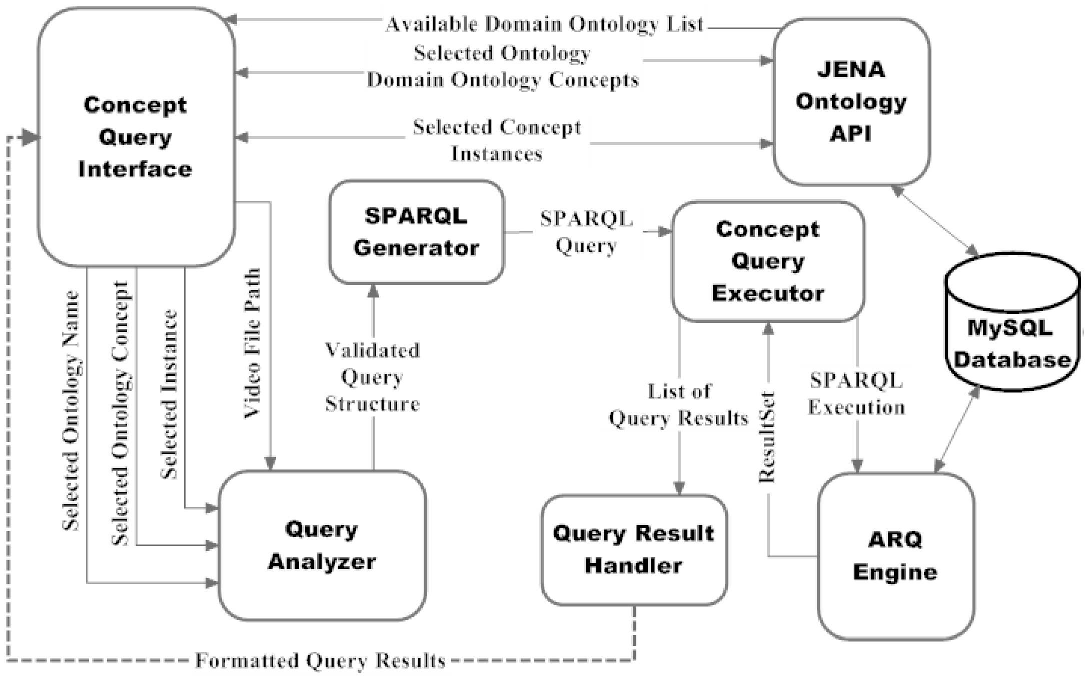


Figure 5.3.1: Concept Query Execution Mechanism

5.3.3.2 Spatial Query Execution

The execution of the spatial queries is handled in the spatial query executor module. The major difference between the execution of a spatial query and a concept query is that in a spatial query there are two ontological instances and a spatial relation to be queried. During the spatial query execution process, SPARQL query generated for a spatial query type is retrieved and the spatial query executor module interacts with ARQ Engine in order to execute the query over the ontological video models constructed in spatio-temporal annotation phase in the relational database. In annotation phase, for each annotated video file, an ontological model that stores the spatio-temporal relations between annotated instances is constructed. In spatial query processing phase, these ontological models in the database are accessed via ARQ Engine and the query is executed to retrieve the matching results from the related models. For video specific querying, the user selects a video file via query interface and the query executor module executes the query only on the ontological model of the related video. However, for general video querying, the user requests to retrieve the matching results from different video files. Therefore, the query is executed via ARQ Engine on the ontological models of all annotated video files in the relational database and all matching results are retrieved. The following example displays a sample spatial query and presents the corresponding SPARQL query generated for execution.

Example: Show all frame intervals where a male appears to left of a female.

Spatial Query Structure:

Query Type = Spatial

First Domain Ontology = Person

First Domain Ontology Concept = Male

First Instance Name

Relation = Left

Second Domain Ontology = Person

Second Domain Ontology Concept = Female

Second Instance Name

Video File Path

By using the spatial query structure, SPARQL generator outputs the following SPARQL query to be executed on ontological video models.

```
SELECT  ?stLoc
WHERE
{
        ?x  a  <http://www.moass.com/SpatioTemporal#Left> .
        ?x  <http://www.moass.com/SpatioTemporal#SpatioTemporalLocator>  ?stLoc .
        ?stLoc <http://www.moass.com/SpatioTemporal#FirstDomain> "Person" .
        ?stLoc <http://www.moass.com/SpatioTemporal#FirstConcept> "Male" .
        ?stLoc <http://www.moass.com/SpatioTemporal#SecondDomain> "Person" .
```

?stLoc <http://www.moass.com/SpatioTemporal#SecondConcept> "Female" .

}

After executing the related SPARQL query, the query result set is obtained and parsed to construct a query result list similar to the concept query execution process. Finally, the query result list is sent to Query Result Handler module in order to be displayed in user interface. The details of OntoVARS spatial query execution process is shown in Figure 5.3.2.

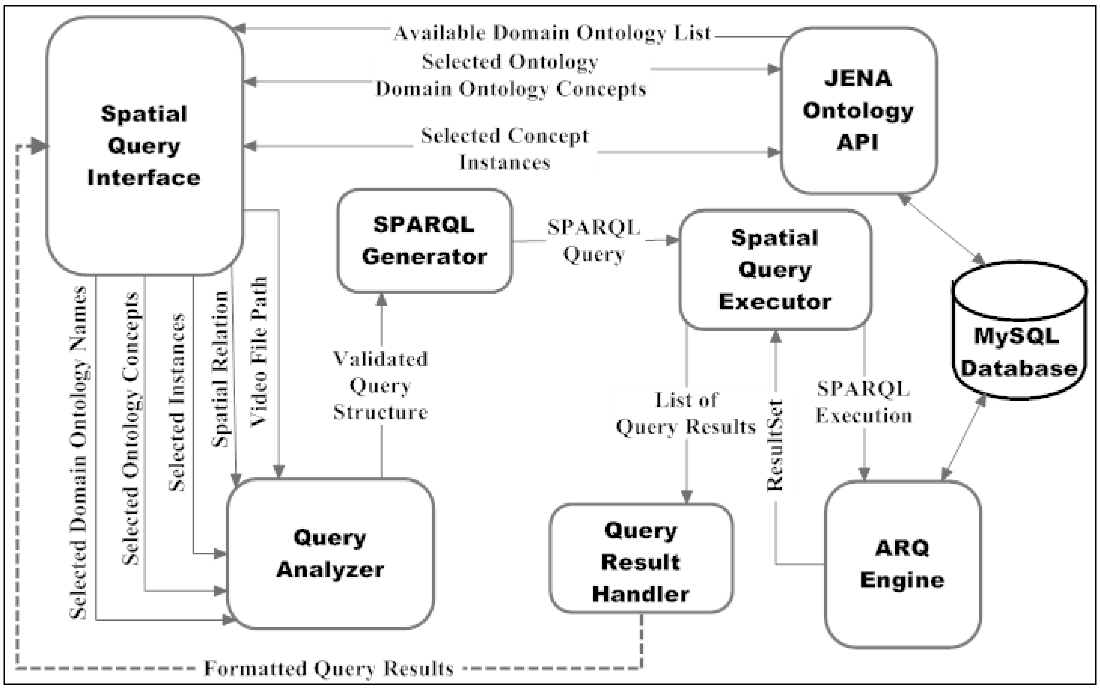


Figure 5.3.2: Spatial Query Execution Mechanism

5.3.3.3 Temporal Query Execution

The execution of the temporal queries is performed in the temporal query executor module. By using the validated query structure, SPARQL query to be executed is generated according to the temporal query data that consists of the ontological information of both instances and the selected temporal relation to be queried. After the generation of SPARQL query, the execution of the temporal queries is performed similar to the execution of spatial queries. For instance, the following example displays a sample temporal query and shows the corresponding SPARQL query generated for execution.

Example: Show frame intervals where the player Hakan Şükür precedes an Offside event.

Temporal Query Structure:

Query Type = Temporal

First Domain Ontology = Soccer

First Domain Ontology Concept = Player

First Instance Name = Hakan Şükür

Relation = Precedes

Second Domain Ontology = Soccer

Second Domain Ontology Concept = Offside

Second Instance Name

Video File Path

By using the temporal query structure, SPARQL generator outputs the following SPARQL query to be executed on ontological video models.

```
SELECT  ?stLoc WHERE
{
        ?x  a  <http://www.moass.com/SpatioTemporal#Precedes> .
        ?x  <http://www.moass.com/SpatioTemporal#SpatioTemporalLocator>  ?stLoc .
        ?stLoc <http://www.moass.com/SpatioTemporal#FirstDomain> "Soccer" .
        ?stLoc <http://www.moass.com/SpatioTemporal#FirstConcept> "Player" .
        ?stLoc <http://www.moass.com/SpatioTemporal#FirstInstance> "Hakan Şükür" .
        ?stLoc <http://www.moass.com/SpatioTemporal#SecondDomain> "Soccer" .
        ?stLoc <http://www.moass.com/SpatioTemporal#SecondConcept> "Offside" .
}
```

In Figure 5.3.3, OntoVARS temporal query execution mechanism is presented.

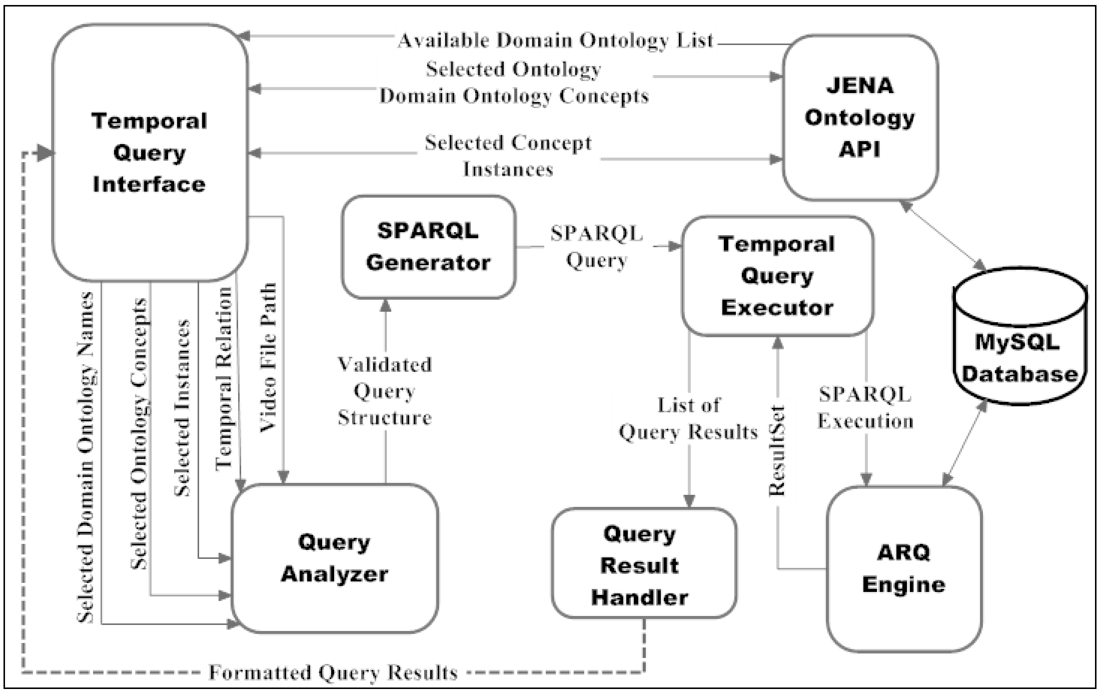


Figure 5.3.3: Temporal Query Execution Mechanism

5.3.3.4 Time-Based Query Execution

The time-based query executor module deals with the execution of the time-based queries. SPARQL generator interacts with the query analyzer to retrieve the valid time-based query structure which is in the following form:

Time-Based Query Structure:

Query Type

Domain Ontology Name

Domain Ontology Concept

Instance Name

Time-Based Relation

Time Interval Start Value

Time Interval End Value

Video File Path

This time-based query structure is used to generate the corresponding SPARQL query in SPARQL generation phase. For instance, the following query structure and SPARQL query are generated for the time-based query given below.

Example: Show all frame intervals where a female Mary appears before frame time 43.5.

Time-Based Query Structure:

Query Type = Time-Based

Domain Ontology Name = Person

Domain Ontology Concept = Female

Instance Name = Mary

Time-Based Relation = Before

Time Interval Start Value = 43.5

Time Interval End Value

Video File Path

The corresponding SPARQL query:

```
SELECT  ?startTime  ?duration  ?mediaUri
WHERE
{
<http://www.sw-app.org/Person#Mary>
        <http://rhizomik.net/ontologies/2005/03/Mpeg7-2001.owl#MediaLocator>
        ?mediaLocator .
?mediaLocator
        <http://rhizomik.net/ontologies/2005/03/Mpeg7-2001.owl#MediaIncrDuration>
        ?duration .
?mediaLocator
```

```
    <http://rhizomik.net/ontologies/2005/03/Mpeg7-2001.owl#MediaRelIncrTimePoint>
    ?startTime .
?mediaLocator
    <http://rhizomik.net/ontologies/2005/03/Mpeg7-2001.owl#MediaUri>
    ?mediaUri .
FILTER (<http://www.w3.org/2001/XMLSchema#double>(?startTime) < 43.5) .
}
```

The time-based query executor receives the generated SPARQL query as an input for execution. In order to execute the query, this module interacts with ARQ Engine that provides relational database connectivity for the execution of SPARQL query. After SPARQL query is executed, the obtained result set is parsed and a query result list is constructed. At the end of this process, Query Result Handler module receives the constructed query result list and displays the query results to the users in the user interface. In Figure 5.3.4, the general architecture of the time-based query execution mechanism is shown in details.

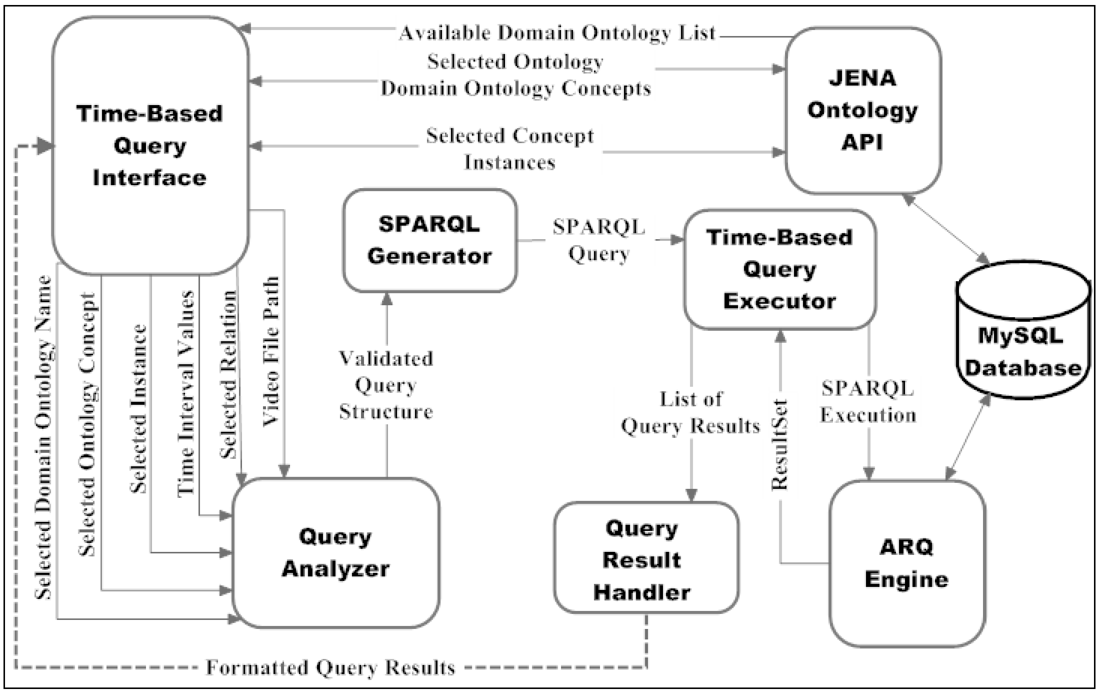


Figure 5.3.4: Time-Based Query Execution Mechanism

5.3.3.5 Region-Based Query Execution

The region-based query executor module is responsible for the execution of the region-based queries. The region-based query execution process resembles the time-based query execution process. In region-based query processing, SPARQL generator interacts with the query analyzer to retrieve the valid region-based query structure which is in the following form:

Region-Based Query Structure:

Query Type

Domain Ontology Name

Domain Ontology Concept

Instance Name

Query Region Data

Video File Path

By using these region-based query data fields, a related SPARQL query is formulated. However, the execution of region-based queries is performed in two steps. In the first step, the query executor module receives the generated SPARQL query as an input for execution and retrieves the matching scenes with the instance MBR coordinates. In the next step, the retrieved instances and their object coordinates are filtered according to the specified query region coordinates. For instance, the following region-based query is analyzed as an example for the generation of the query structure and SPARQL query.

Example: Show all frame intervals where a Penalty event occurs in the specified region.

Region-Based Query Structure:

Query Type = Region-Based

Domain Ontology Name = Soccer

Domain Ontology Concept = Penalty

Instance Name

Query Region Data = Query Region Coordinates

Video File Path

The corresponding SPARQL query:

```
SELECT ?startTime ?duration ?mediaUri ?mbrPoints
WHERE
{
?x a <http://www.sw-app.org/Soccer#Penalty> .
?x <http://rhizomik.net/ontologies/2005/03/Mpeg7-2001.owl#MediaLocator>
        ?mediaLocator .
?mediaLocator
        <http://rhizomik.net/ontologies/2005/03/Mpeg7-2001.owl#MediaIncrDuration>
       ?duration .
?mediaLocator
     <http://rhizomik.net/ontologies/2005/03/Mpeg7-2001.owl#MediaRelIncrTimePoint>
     ?startTime .
?mediaLocator
     <http://rhizomik.net/ontologies/2005/03/Mpeg7-2001.owl#MediaUri>
     ?mediaUri .
?mediaLocator
     <http://rhizomik.net/ontologies/2005/03/Mpeg7-2001.owl#Polygon>
     ?mbrPoints .
}
```

After executing the generated SPARQL query, the MBR corner points of the retrieved results are compared with the specified query region coordinates. If the retrieved individual, object or event resides in the query region, the retrieved result is added to the constructed query result list. Otherwise, the instances that are out of the query region are eliminated. Finally, the query results are sent to Query Result Handler module for display in the user interface. Figure 5.3.5 displays the general architecture of the region-based query execution process.

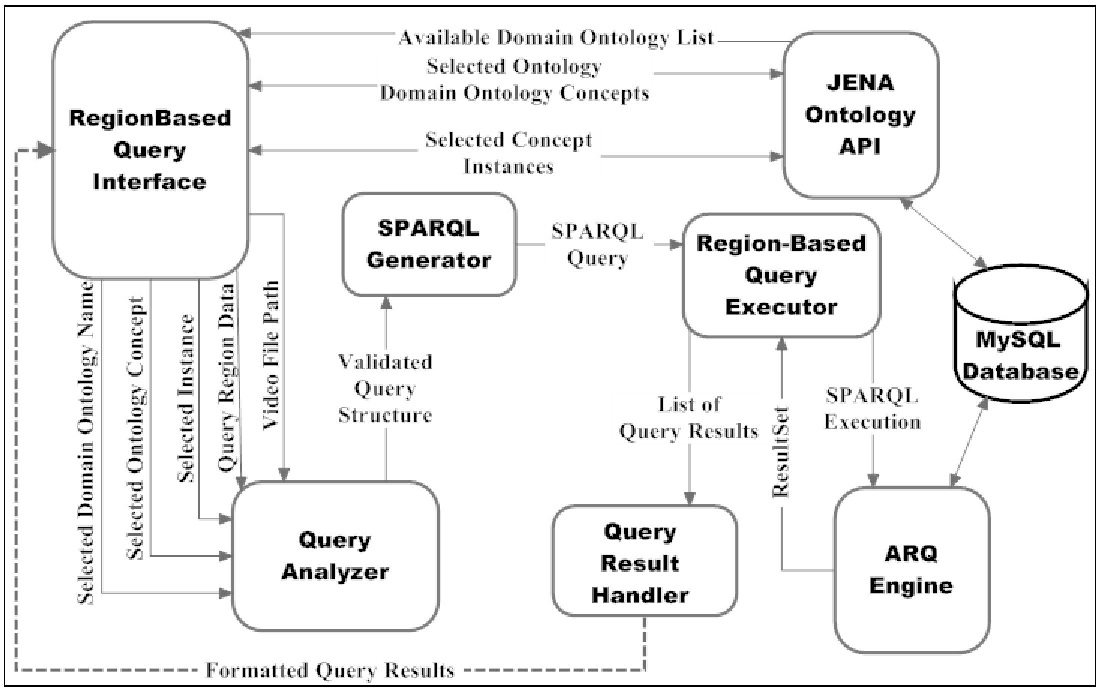


Figure 5.3.5: Region-Based Query Execution Mechanism

5.3.3.6 Compound Query Execution

OntoVARS supports ontology-driven compound queries which are formulated by combining concept, spatial, temporal, time-based and region-based simple queries using "AND", "OR", "(" and ")" operators. In query analysis phase, the query analyzer retrieves the compound query input, parses the query and divides it into its sub-queries. Afterwards, a compound query tree structure is constructed regarding the operator precedence and expressions in parenthesis. In OntoVARS video database model, a compound query is represented by a query tree in which the sub-queries of the compound query form the leaf nodes of the query tree whereas the operators form the internal nodes. At the end of the query analysis process, the constructed query tree is sent to Compound Query Handler module for execution.

Compound Query Handler module applies an optimization algorithm on the initial query tree in order to process more selective sub-queries before the others. The optimization algorithm restructures the initial query tree and generates an optimal tree for execution [30]. By using the optimal query tree, the query result size is kept as small as possible and the final result generation process is performed more efficiently. The query optimization process implemented in [30] proposes two optimization mechanisms, which are internal node reordering and leaf node reordering. In the concept of this thesis study, only internal node reordering mechanism is applied for the compound query processing. During the query tree reconstruction process according to the internal node reordering algorithm, the children of "AND" typed nodes are reordered in order to place more selective nodes as the left child of the "AND" parent node since the left child is processed first [30]. The internal node reordering algorithm restructures the query tree using the following rules:

- The "AND", "Concept", "Time-Based", "Region-Based", "Spatial" and "Temporal" type child nodes are placed as the left child if the other child is "OR" type. Since "OR" type nodes combine the query results of two different result sets, they are less selective nodes compared to the other types.

- The "Concept", "Time-Based", "Spatial" and "Temporal" type child nodes are placed on the left node if the other child node contains "Region-Based" type queries, since these queries are processed faster than "Region-Based" queries.

Some compound query formulations are given in the following examples. In each example, the initial query tree and the query tree after internal node reordering process are shown.

Query 1:

((East(A, B) AND Disjoint(A, B)) OR (Above(A, B) AND Starts(A, B))) AND (Appear(A) AND Between(B, 45.0, 60.0)).

The above compound query is formulated by using the spatial, temporal, concept and time-based sub-queries. According to the internal node reordering algorithm, the children of the root "AND" type node are exchanged since the type of the left node is "OR" and the type of the right node is "AND" in the initial query tree. In Figure 5.3.6, the initial and the final query tree structures are displayed.

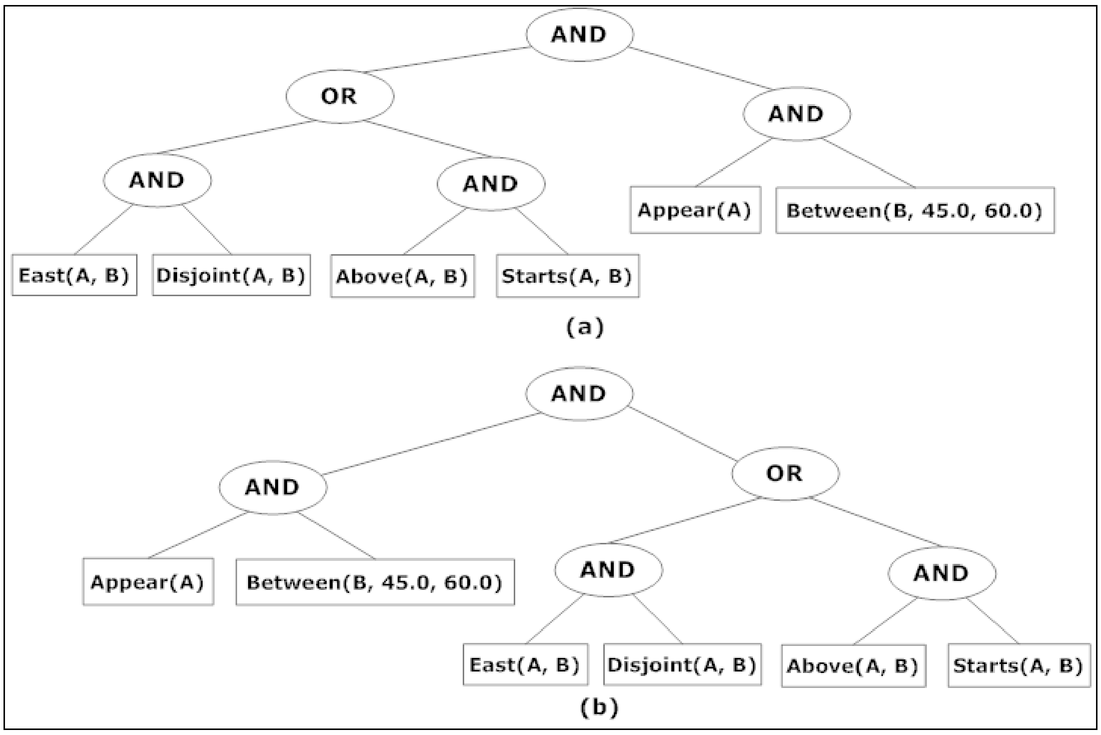


Figure 5.3.6: (a) Initial Query Tree for Query 1 and (b) Final Query Tree for Query 1 After Internal Node Reordering

Query 2:

((InRegion(X, region1) AND During(X, Y)) OR Appear(Z)) AND ((Touch(X, Y) OR Left(X, Z)) AND (InRegion(Y, region2) AND After(X, 50.0)))

The above compound query is composed of the region-based, time-based, concept, temporal and spatial sub-queries. Both rules of the internal node reordering mechanism are applied on the initial query tree in order to construct the final compound query tree for execution. Figure 5.3.7 shows the initial query tree and the reconstructed final query tree after the internal node reordering is applied.

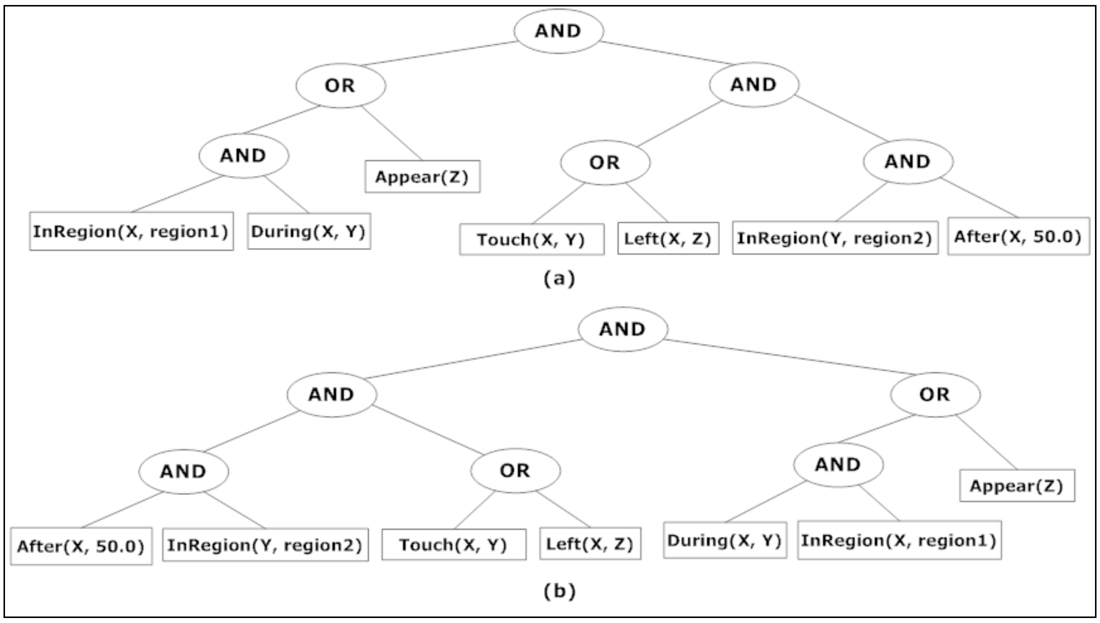


Figure 5.3.7: (a) Initial Query Tree for Query 2 and (b) Final Query Tree for Query 2 After Internal Node Reordering

By applying the internal node reordering process on the initial compound query tree, an optimal query tree is constructed for execution. After the optimal query tree is generated, Compound Query Handler module sends each sub-query to the related query executor modules according to the sub-query type. These sub-queries are executed separately in the related query engines and their results are returned back to Compound Query Handler module for the final query result generation. By applying union and intersection operations, the sub-query results are combined and the final query result list is transferred to Query Result Handler module in order to be displayed in the user interface. In Figure 5.3.8, the execution of the compound queries by combining the query result sets of each sub-query is shown. If the connector between two sub-queries is "AND" operator, the intersection of two result sets are evaluated. On the other hand, if the sub-queries are connected with "OR" operator, the union of two result sets is computed. In order to find the intersection of two result sets, each frame interval in the first result set is compared with every frame interval in the second result set. If the first time interval overlaps with the second time interval, then the overlapping interval is computed and added to the intersection result. For union operation, similar to the intersection process, each frame interval in the first result set is compared with every frame interval in the second result set and the union of two time intervals is obtained and added to the union result. This evaluation process continues until all sub-query results are merged and the final compound query result is generated.

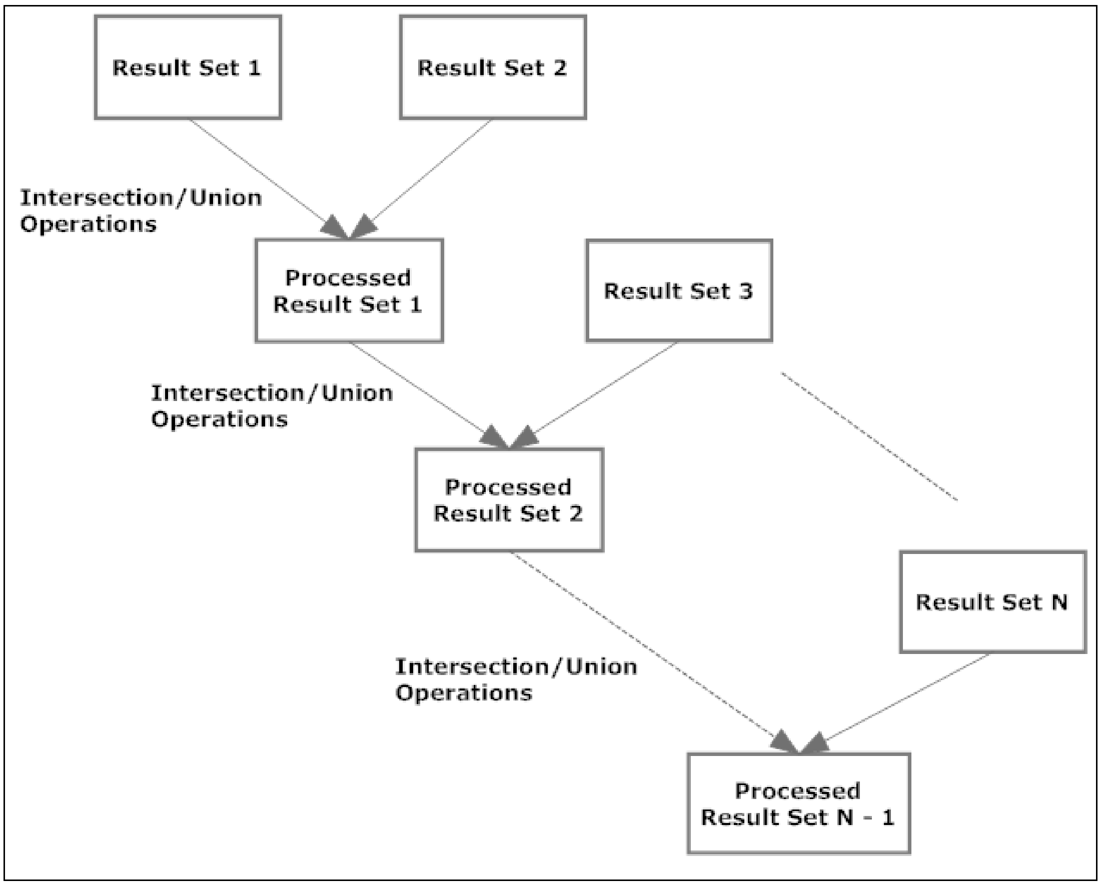

Result Set 1
Result Set 2
Intersection/Union Operations
Processed Result Set 1
Result Set 3
Intersection/Union Operations
Processed Result Set 2
Result Set N
Intersection/Union Operations
Processed Result Set N - 1

Figure 5.3.8: Compound Query Evaluation

CHAPTER 6

IMPLEMENTATION

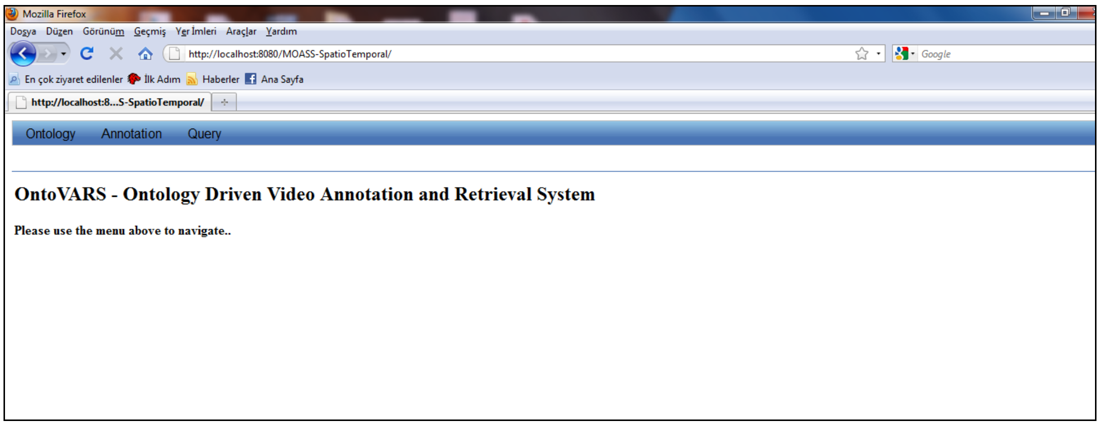
OntoVARS is a domain independent ontology-driven video annotation and retrieval system that provides semantic content-based annotation and querying capabilities in video databases. During the evaluation of the proposed framework, the framework is tested and evaluated with different domain ontologies in terms of ontological integration, video content annotation and querying power. Essentially, the system capabilities and usage do not depend on the selected domain ontology. Therefore in this chapter person ontology is selected for the demonstration of the system capabilities and its usage.

In the following sections, the usage and the features of the proposed system are explained by the help of user interface screenshots. OntoVARS consists of three main components, namely ontology management, annotation, querying and the system end users interact with these components through the system main page. The main page of OntoVARS is shown in Figure 6.1.1.

This chapter is organized as follows: In Section 6.1, ontology management menu and its features are explained. OntoVARS annotation panel is discussed in Section 6.2 and finally, Section 6.3 presents the usage of the query panel and system querying capabilities.

6.1 Ontology Management Panel

The ontology management component handles MPEG-7 and domain ontology operations in order to construct the ontological infrastructure. The automatic harmonization of MPEG-7 core ontology and imported domain ontologies is performed through the sub-menus of the ontology management panel. Moreover, the domain specific common data integration capability is also enabled in the concept of this module. The ontology management menu is presented in Figure 6.1.2.

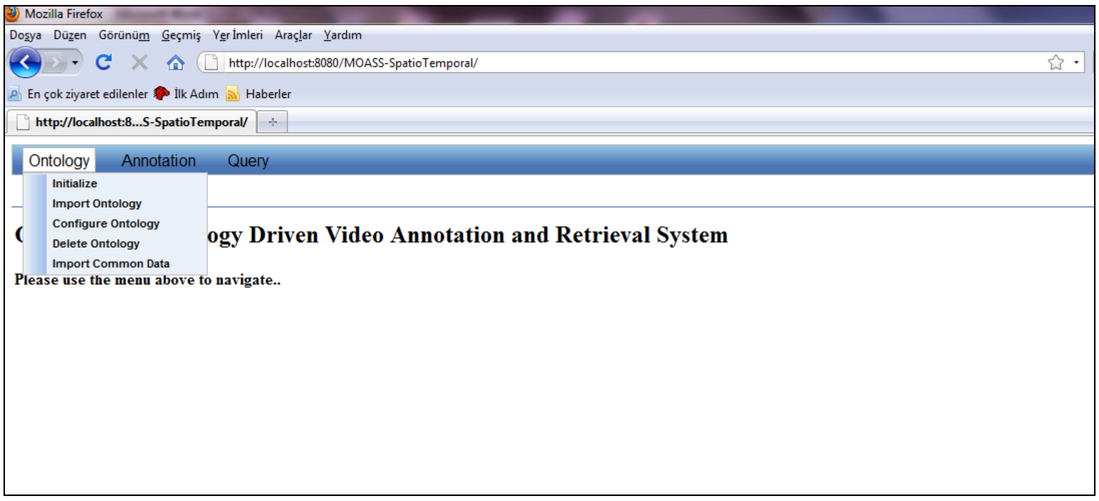


Mozilla Firefox
Dosya Düzen Görünüm Geçmiş Yer İmleri Araçlar Yardım
http://localhost:8080/MOASS-SpatioTemporal/
Google
En çok ziyaret edilenler İlk Adım Haberler Ana Sayfa
http://localhost:8...S-SpatioTemporal/
Ontology Annotation Query

OntoVARS - Ontology Driven Video Annotation and Retrieval System

Please use the menu above to navigate..

Figure 6.1.1: OntoVARS Main Page



Mozilla Firefox
Dosya Düzen Görünüm Geçmiş Yer İmleri Araçlar Yardım
http://localhost:8080/MOASS-SpatioTemporal/
En çok ziyaret edilenler İlk Adım Haberler
http://localhost:8...S-SpatioTemporal/
Ontology Annotation Query
Initialize
Import Ontology
Configure Ontology
Delete Ontology
Import Common Data

O[illegible]ogy Driven Video Annotation and Retrieval System

Please use the menu above to navigate..

Figure 6.1.2: Ontology Management Panel

The ontology management panel provides the following operations through its sub-menus:

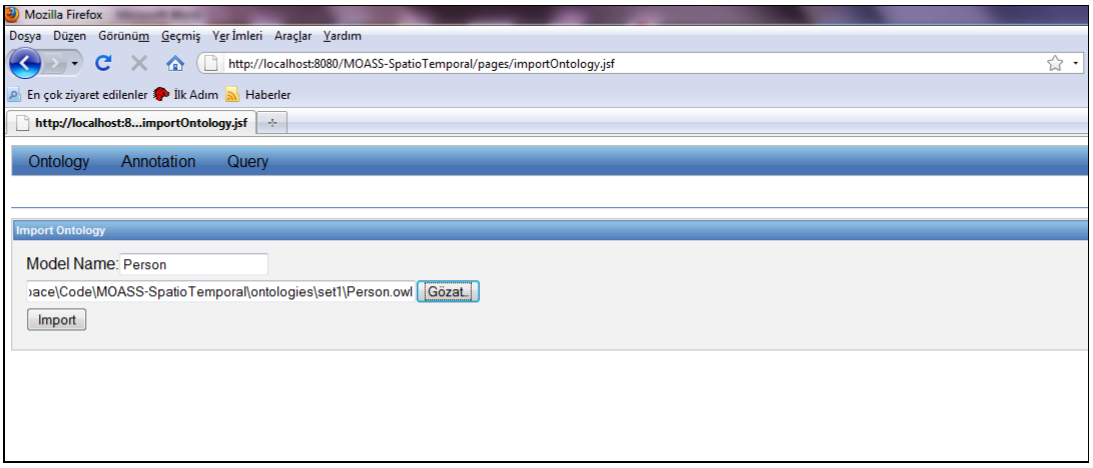
- System Initialization: This operation is used to clean up and initialize the whole system resources. All ontological data, imported domain ontologies and annotations are removed from the relational database. After cleaning operation is finalized, system initialization process loads MPEG-7 upper ontology automatically to the system.

- Ontology Import: The integration of the domain specific ontologies is carried out via import ontology user interface. The user enters a model name for the domain ontology and selects the ontology file to be imported. Figure 6.1.3 represents the domain ontology import operation.

- Ontology Configuration: The ontology configuration interface enables the user to categorize the imported domain ontology concepts into the following categories: Semantic Base, Object, Agent Object, Person, PersonGroup, Organization, Concept, Semantic Place, Semantic Time and Semantic State. According to this categorization, the system subclasses the selected domain ontology concepts from the corresponding super concepts in the upper ontology [20].

- Ontology Deletion: The system enables the deletion of the imported domain ontologies through ontology deletion interface. In the concept of the deletion operation, the selected ontological model and the annotations related to this domain ontology concepts are removed from the system relational database.

- Common Data Integration: This sub-menu enables the integration of the domain specific common data into the ontological structure. Similar to ontology import operation, the user enters a model name in order to specify the ontological model in which the common data is integrated and selects the data file for domain specific common data import operation. Figure 6.1.4 demonstrates the domain specific common data import operation.

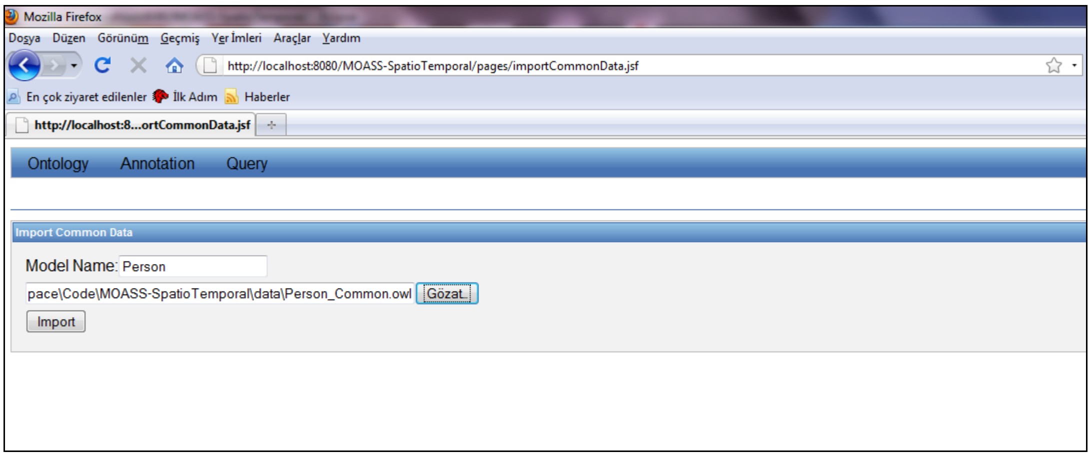

Mozilla Firefox
Dosya Düzen Görünüm Geçmiş Yer İmleri Araçlar Yardım
http://localhost:8080/MOASS-SpatioTemporal/pages/importOntology.jsf
En çok ziyaret edilenler İlk Adım Haberler
http://localhost:8...importOntology.jsf
Ontology Annotation Query
Import Ontology

Model Name: Person

›ace\Code\MOASS-SpatioTemporal\ontologies\set1\Person.owl Gözat...

Import

Figure 6.1.3: Ontology Import Interface


Mozilla Firefox
Dosya Düzen Görünüm Geçmiş Yer İmleri Araçlar Yardım
http://localhost:8080/MOASS-SpatioTemporal/pages/importCommonData.jsf
En çok ziyaret edilenler İlk Adım Haberler
http://localhost:8...ortCommonData.jsf
Ontology Annotation Query
Import Common Data

Model Name: Person

pace\Code\MOASS-SpatioTemporal\data\Person_Common.owl Gözat...

Import

Figure 6.1.4: Common Data Integration Interface

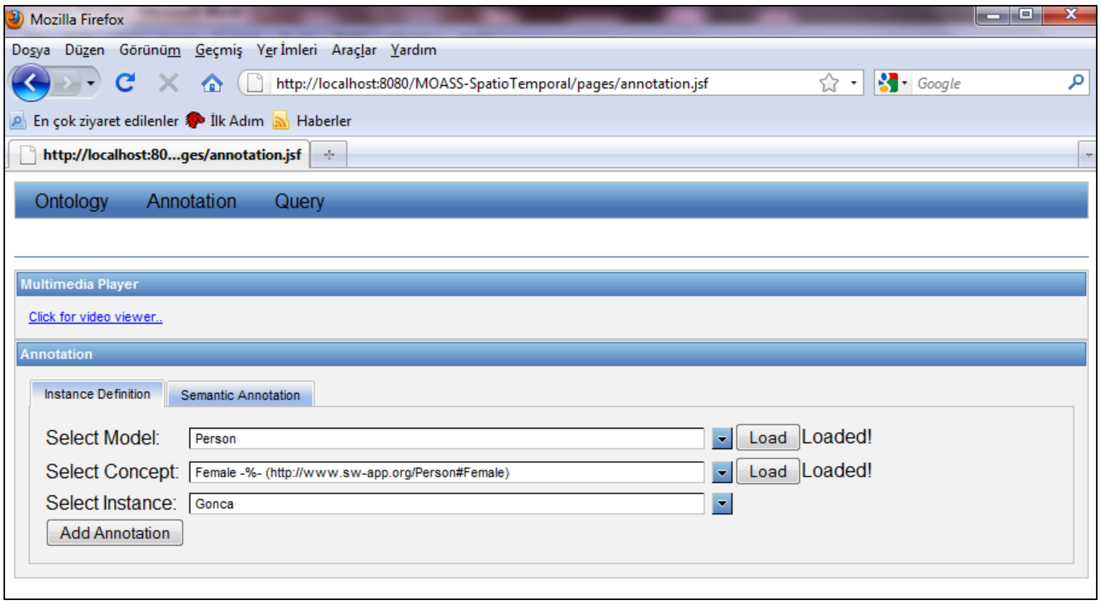
6.2 Video Annotation Panel

The video annotation panel is used to manage ontology-driven video content annotation operations. In OntoVARS, video content annotation procedure is performed manually. However, the modular architecture of the proposed system enables the integration of semi-automatic and automatic annotation modules to the system. In addition, the annotation procedure is performed using domain ontology concepts. By this means, the annotation interface provides legal content descriptions and the ontological relation between the annotations and domain ontologies is established. In this panel, the users have the following opportunities for ontology-driven annotation:

- Instance Definition: OntoVARS supports ontological annotation in order to provide ontology-driven querying capability to the end users. For this purpose, the instances should be defined according to the ontology classes before the annotation process starts. The ontological instances can be supplied to the system by domain specific common data integration process or defined through instance definition interface. For instance in Figure 6.2.1, *Gonca* instance is defined as an individual of *Female* ontology class in *Person* domain ontology.

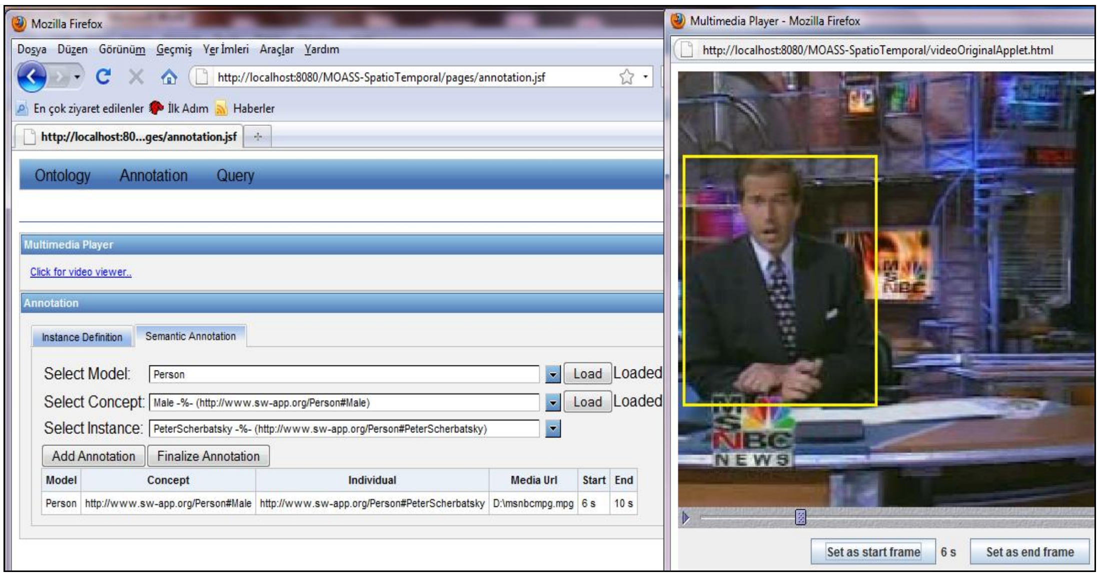
- Semantic Annotation: In semantic annotation process, the instances of the domain ontology concepts and classes are annotated on video content. The object appearance frame intervals and the object positions are stored for each instance in the video content during semantic annotation process. These annotated instances, their positions and time intervals are used for calculating spatial and temporal relations between video objects and posing semantic content-based queries over video content. For specifying the instance positions, minimum bounding rectangle (MBR) formalization is used. During annotation process, the user selects the domain ontology, ontology concept and the instance to be annotated by using the provided user interface menu options. Afterwards, the video file on which the annotation operation will be performed is played and the selected instance appearance time intervals and positions are tagged through the semantic annotation interface. Figure 6.2.2 presents the semantic annotation of the individual *Peter Scherbatsky*.

- Spatio-Temporal Annotation: Spatio-temporal annotation process is started after the semantic annotation of the instances for a video file is completed and *Finalize Annotation* button is pressed in semantic annotation user interface. For each annotated video file, the spatio-temporal relations between the annotated instances are computed implicitly and stored into the relational database for later usage in ontology-driven spatio-temporal querying. *Finalize Annotation* button is also displayed in Figure 6.2.2 which is used to start the computations of spatial and temporal relations during spatio-temporal annotation process.


Mozilla Firefox
Dosya Düzen Görünüm Geçmiş Yer İmleri Araçlar Yardım
http://localhost:8080/MOASS-SpatioTemporal/pages/annotation.jsf
Google
En çok ziyaret edilenler İlk Adım Haberler
http://localhost:80...ges/annotation.jsf
Ontology Annotation Query
Multimedia Player
Click for video viewer..
Annotation
Instance Definition Semantic Annotation

Select Model:	Person	Load	Loaded!
Select Concept:	Female -%- (http://www.sw-app.org/Person#Female)	Load	Loaded!
Select Instance:	Gonca		

Figure 6.2.1: Instance Definition Interface


Mozilla Firefox
Dosya Düzen Görünüm Geçmiş Yer İmleri Araçlar Yardım
http://localhost:8080/MOASS-SpatioTemporal/pages/annotation.jsf
En çok ziyaret edilenler İlk Adım Haberler
http://localhost:80...ges/annotation.jsf
Ontology Annotation Query
Multimedia Player
Click for video viewer..
Annotation
Instance Definition Semantic Annotation
Multimedia Player - Mozilla Firefox
http://localhost:8080/MOASS-SpatioTemporal/videoOriginalApplet.html
MSNBC
NBC NEWS
Set as start frame
6 s
Set as end frame

Select Model:	Person	Load	Loaded
Select Concept:	Male -%- (http://www.sw-app.org/Person#Male)	Load	Loaded
Select Instance:	PeterScherbatsky -%- (http://www.sw-app.org/Person#PeterScherbatsky)		

Add Annotation
Finalize Annotation

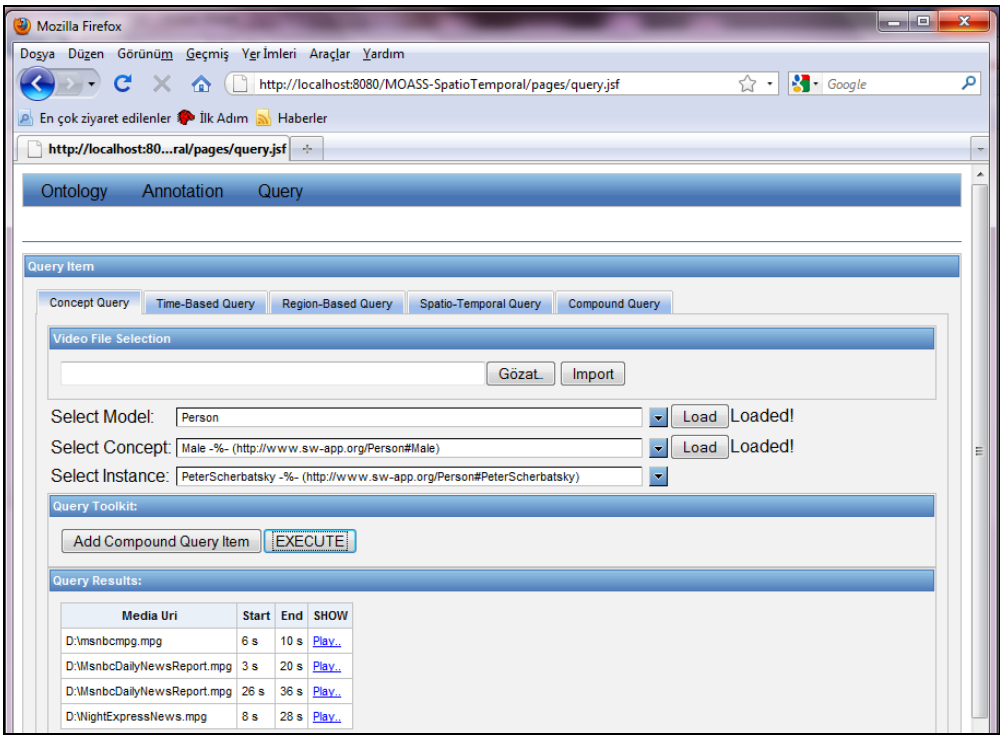
Model	Concept	Individual	Media Url	Start	End
Person	http://www.sw-app.org/Person#Male	http://www.sw-app.org/Person#PeterScherbatsky	D:\msnbcmpg.mpg	6 s	10 s

Figure 6.2.2: Semantic Annotation Interface

6.3 Video Query Panel

The query panel provides user interfaces for the system supported query types. The form-based query interfaces enable the formulation of ontology-driven semantic content-based queries and display the retrieved query results according to the result display format to the end users. In the query panel, the system provides the following query interfaces for ontology-driven query processing:

- Concept Query: The users can pose simple concept queries by using this interface in order to query the appearances of the ontological instances and domain ontology classes. For instance, Figure 6.3.1 displays the query results containing the video scenes from different video files in which *Peter Scherbatsky* appears.

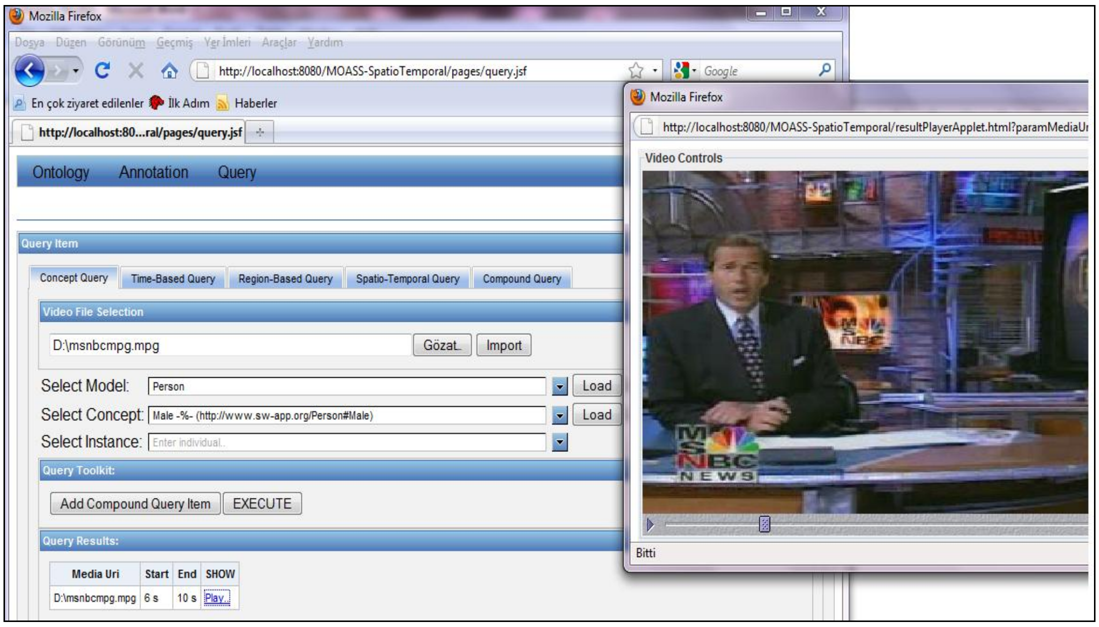

Mozilla Firefox
Dosya Düzen Görünüm Geçmiş Yer İmleri Araçlar Yardım
http://localhost:8080/MOASS-SpatioTemporal/pages/query.jsf
Google
En çok ziyaret edilenler İlk Adım Haberler
http://localhost:80...ral/pages/query.jsf
Ontology Annotation Query
Query Item
Concept Query Time-Based Query Region-Based Query Spatio-Temporal Query Compound Query
Video File Selection
Gözat.. Import
Query Toolkit:
Add Compound Query Item EXECUTE
Query Results:

Select Model: Person Load Loaded!
Select Concept: Male -%- (http://www.sw-app.org/Person#Male) Load Loaded!
Select Instance: PeterScherbatsky -%- (http://www.sw-app.org/Person#PeterScherbatsky)

Media Uri	Start	End	SHOW
D:\msnbcmpg.mpg	6 s	10 s	Play..
D:\MsnbcDailyNewsReport.mpg	3 s	20 s	Play..
D:\MsnbcDailyNewsReport.mpg	26 s	36 s	Play..
D:\NightExpressNews.mpg	8 s	28 s	Play..

Figure 6.3.1: Multiple Video Concept Query

In order to generate concept queries for a specific video, the users select a video file from the video file selection menu. After a video file is selected, the formulated query is executed to retrieve the matching scenes for the specified video. Figure 6.3.2 displays the query results containing the video scenes from the specified video file in which any *Male* class individual appears. By pressing *Play* button that is provided at the end of each query result row, the related video intervals are played automatically in the video player applet.



http://localhost:8080/MOASS-SpatioTemporal/pages/query.jsf

http://localhost:8080/MOASS-SpatioTemporal/resultPlayerApplet.html?paramMediaUr

D:\msnbcmpg.mpg

Select Model: Person

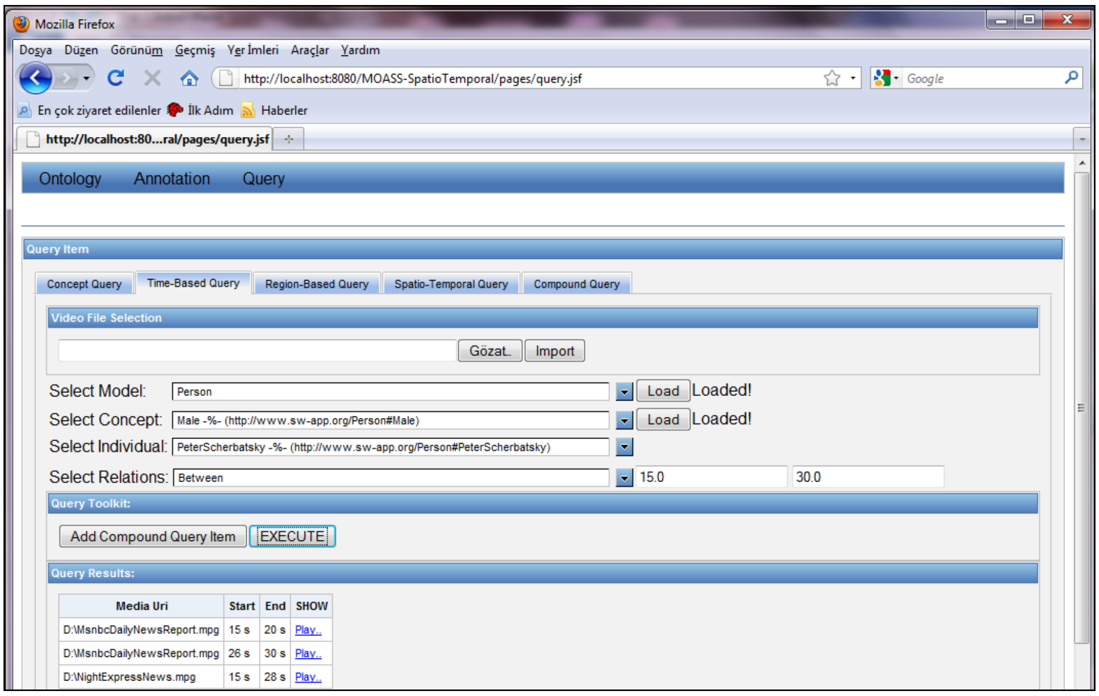
Select Concept: Male -%- (http://www.sw-app.org/Person#Male)

Select Instance: Enter individual

Media Uri	Start	End	SHOW
D:\msnbcmpg.mpg	6 s	10 s	Play..

Figure 6.3.2: Specific Video Concept Query

- Time-Based Query: The users formulated time-based queries through time-based query interface. Similar to all other query types, both multiple and specific video querying features are supported for time-based queries. In order to formulate the queries, the users select the related domain ontology model name, ontology concept and/or ontological instance. In addition, the time-based relation; Before, After, Between and Contain is specified and time interval values are provided from this query interface. For instance, Figure 6.3.3 displays the query results containing the video scenes from different video files in which *Peter Scherbatsky* appears between time interval 15.0 and 30.0.

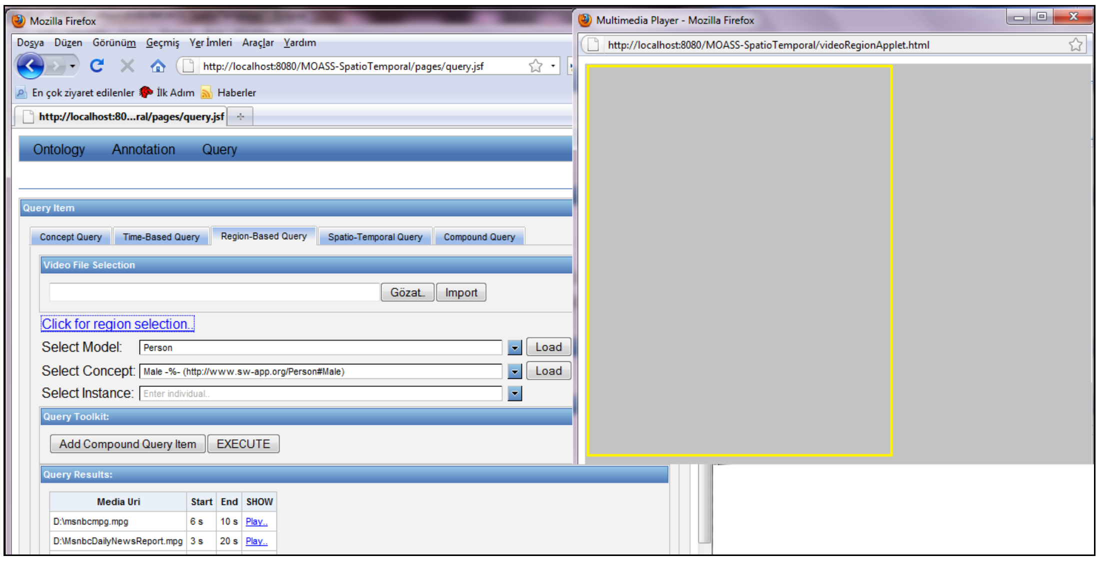

Mozilla Firefox
Dosya Düzen Görünüm Geçmiş Yer İmleri Araçlar Yardım
http://localhost:8080/MOASS-SpatioTemporal/pages/query.jsf
Google
En çok ziyaret edilenler İlk Adım Haberler
http://localhost:80...ral/pages/query.jsf
Ontology Annotation Query
Query Item
Concept Query
Time-Based Query
Region-Based Query
Spatio-Temporal Query
Compound Query
Video File Selection
Gözat..
Import
Query Toolkit:
Add Compound Query Item
EXECUTE
Query Results:

Select Model: Person — Load — Loaded!
Select Concept: Male -%- (http://www.sw-app.org/Person#Male) — Load — Loaded!
Select Individual: PeterScherbatsky -%- (http://www.sw-app.org/Person#PeterScherbatsky)
Select Relations: Between — 15.0 — 30.0

Media Uri	Start	End	SHOW
D:\MsnbcDailyNewsReport.mpg	15 s	20 s	Play..
D:\MsnbcDailyNewsReport.mpg	26 s	30 s	Play..
D:\NightExpressNews.mpg	15 s	28 s	Play..

Figure 6.3.3: Time-Based Query Interface

- Region-Based Query: The users formulate region-based queries through region-based query interface. Before query formulation and execution, the users select the related domain ontology model name, ontology concept and/or ontological instance from the query interface. Moreover, in order to specify the query region coordinates, the users click for region selection and draw a rectangular region by dragging the mouse on region selection area. After the query execution is completed, the video scenes that contain the queried instance or concept in the specified query region are displayed to the user. For instance, Figure 6.3.4 displays the query formulation and region selection process for region-based queries through this interface. In addition, the query results containing the video scenes in which any *Male* class individual appears in the specified query region are also listed in Figure 6.3.4.


Mozilla Firefox
Dosya Düzen Görünüm Geçmiş Yer İmleri Araçlar Yardım
http://localhost:8080/MOASS-SpatioTemporal/pages/query.jsf
En çok ziyaret edilenler İlk Adım Haberler
http://localhost:80...ral/pages/query.jsf
Ontology Annotation Query
Query Item
Concept Query Time-Based Query Region-Based Query Spatio-Temporal Query Compound Query
Video File Selection
Gözat.. Import
Click for region selection..
Query Toolkit:
Add Compound Query Item EXECUTE
Multimedia Player - Mozilla Firefox
http://localhost:8080/MOASS-SpatioTemporal/videoRegionApplet.html

Select Model: Person
Select Concept: Male -%- (http://www.sw-app.org/Person#Male)
Select Instance: Enter individual..

Query Results:

Media Uri	Start	End	SHOW
D:\msnbcmpg.mpg	6 s	10 s	Play..
D:\MsnbcDailyNewsReport.mpg	3 s	20 s	Play..

Figure 6.3.4: Region-Based Query Interface

- Spatio-Temporal Query: The spatial and temporal queries are generated by using this interface in order to query the spatio-temporal relations between video instances. The users can specify the domain ontology, ontological concept and instances for the queried individuals, objects or events by selecting from the user interface. Moreover, the spatial or temporal relation to be queried can also be selected during the query formulation process. After executing the formulated query, the matching query results are displayed to the users in spatio-temporal query display format.

- Compound Query: OntoVARS provides ontology-driven compound querying capability to the end users. The supported simple queries are combined by using "AND" and "OR" connectors in order to formulate compound queries. The simple queries are constructed in their own query tabs and *Add Compound Query Item* button is used to add the formulated simple query to the compound query. After adding the generated simple query to the compound query, the user moves to compound query tab in order to specify the connector between the lastly added simple query and the existing compound query. This operation is repeated for each sub-query insertion.

Finally, the compound query is executed by clicking *Execute* button in the compound query tab and the retrieved query results are displayed.

CHAPTER 7

CONCLUSION AND FUTURE WORK

In this thesis, an ontology-driven video annotation and retrieval system is proposed. The developed framework is a generic solution for ontological video content modeling and content-based querying. The main difference of the developed system from other video management and retrieval systems is its ontological infrastructure. The system is MPEG-7 ontology-based and MPEG-7 ontology infrastructure provides interoperability with other MPEG-7 ontology compatible systems. The Rhizomik MPEG-7 ontology is used as the core ontology and domain specific ontologies are integrated to the core ontology in order to provide ontology-based video content annotation and querying capabilities to the user. The system is domain independent, thus any domain ontology can be attached to the system.

The system is developed in a modular architecture and consists of ontology management, video annotation and query processing sub-modules. The ontology management module enables the construction of MPEG-7 core ontology and provides domain ontology insertion, deletion and integration capabilities. Video annotation module enables concept and spatio-temporal annotation on video content by using domain ontology concepts. Moreover, domain specific common data integration is also provided by the annotation module and the annotation effort is relieved by the automatic integration of common data. The query processing module provides ontology-driven conceptual, spatio-temporal, region-based, time-based and compound querying capabilities. For all query types, both multiple video querying and specific video querying capabilities are supported. In multiple video querying, the user query input is executed on video database and matching scenes from different video files are returned to the users as query results. In specific video querying, the user selects a specific video and the query is executed on the selected video content. The matching scenes of this specific video are returned to the users as query results. In query execution process, SPARQL queries are generated according to the form-based user query inputs and query types. The generated SPARQL queries are executed in the query engine and the results are retrieved. Compound query processing mechanism differs from other query types. An optimization mechanism proposed in [30] is adapted to our framework for compound query execution. The compound query input is decomposed into its sub-queries and corresponding SPARQL queries are generated. On the other hand, the compound query tree is generated and the parent nodes containing "AND", "OR" operators are re-organized on the query tree structure. After new query tree is constructed, the sub-queries are executed and combined according to the operators in the parent nodes. By means of the optimization process, the intermediate query result sizes are kept small and an efficient query execution mechanism is provided.

The current version of the system supports manual annotation of the video content; however, the modular architecture of the system enables the integration of any automatic or semi-automatic annotation module. As a future work, video annotation mechanism can be improved by developing or integrating semi-automatic or automatic annotators.

The developed system does not support 3D relations. Spatio-temporal annotation and querying capability for 3D relations can also be added as a future extension. Moreover, the framework is developed for video management capabilities and does not support the annotation and querying of other multimedia content types such as images and audio. These extensions can also be added to the current framework in the future. Thus, by enabling image, audio and text data description and querying features, the system capabilities are enhanced and the framework is transformed to a full-functional multimedia content management and retrieval system.

The query processing mechanism converts the form-based user input to SPARQL queries for execution. The query engine accepts SPARQL query input and retrieves the query results by executing the SPARQL queries. Due to the modular architectural design of the system, SPARQL query generator and query processor modules do not depend on each other. Thus, the query processor module can be separated from the form-based query interface and the SPARQL generator module in order to be integrated with different mechanisms. For instance, a system having a natural language query interface and generating SPARQL query outputs from the given natural language query inputs can be integrated with the query processor component of the proposed framework. As a result, ontology-driven natural language query processing and retrieval capability can also be provided. Similarly, different user interface modules can be integrated to the current system as a future extension.

To summarize, in this thesis, an ontology-driven video annotation and retrieval framework has been developed in order to provide semantic content-based modeling and querying capabilities. The system enhances its video content modeling and querying capabilities with its ontology support. In the light of the future studies, new features can be added to the developed framework since it provides a modular infrastructure that allows extensions.

REFERENCES

[1] J. M. Martinez, "MPEG-7 Overview Version 10", http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm, last visited on 20 Aug. 2010

[2] B. S. Manjunath, P. Salembier, and T. Sikora, editors. "Introduction to MPEG-7 Multimedia Content Description Interface". Wiley, England, 2002.

[3] "Use Case: MPEG-7 metadata interoperability", http://www.w3.org/2005/Incubator/mmsem/wiki/MPEG-7, last visited on 15 Aug. 2010.

[4] 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.

[5] R. Troncy, Celma O., Little S., Garcia R., and Tsinaraki C. "MPEG-7 based multimedia ontologies: Interoperability support or interoperability issue". In Proceedings of the 1st International Workshop on Multimedia Annotation and Retrieval enabled by Shared Ontologies (MAReSO), 2007.

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

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

[8] 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.

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

[10] 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.

[11] 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

[12] 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.

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

[14] 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.

[15] 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.

[16] 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

[17] 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.

[18] 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.

[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] Hilal Tarakçı, "An Ontology-Based Multimedia Information Management System", Master's thesis, METU, 2008.

[21] H. Rajigopal, "JENA: A Java API for Ontology Management", October, 2005.

[22] K. Tzonas, "JENA RDF Framework", May, 2008.

[23] "Jena Ontology API", http://jena.sourceforge.net/ontology, last visited on 15 Aug. 2010.

[24] SPARQL, http://www.w3.org/TR/rdf-sparql-query, last visited on 2 Agu. 2010.

[25] SPARQL Tutorial, http://openjena.org/ARQ/Tutorial/index.html, last visited on 2 Agu 2010.

[26] "Semantic Web/RDF Library for C#/.NET", http://razor.occams.info/code/semweb/, last visited on 3 Agu. 2010.

[27] Mehmet Emin Dönderler, Ediz Saykol, Özgür Ulusoy, and Uğur Güdübay., Ediz Saykol, Cemil Alper. "BilVideo Video Database Management System", VLDB Journal, pp: 1371–1376, 2004.

[28] Mehmet Emin Dönderler, Özgür Ulusoy, Uğur Güdükbay. "Rule-Based Spatio-Temporal Query Processing for Video Databases", VLDB Journal, 13:86–103, 2004.

[29] Mehmet Emin Dönderler, Ediz Saykol, Umut Arslan, Özgür Ulusoy, Uğur Güdükbay. "BilVideo: Design and Implementation of a Video Database Management System", Multimedia Tools and Applications, 27:79–104, 2005.

[30] Gülay Ünel, Mehmet Emin Dönderler, Özgür Ulusoy, Uğur Güdükbay. "An Efficient Query Optimization Strategy for Spatio-Temporal Queries in Video Databases", pp. 113-131, September, 2004.

[31] Mehmet Emin Dönderler. "Data Modeling and Querying for Video Databases", PhD thesis, Bilkent University, 2002.

[32] Cemil Alper. "Semantic Query Execution In A Video Database System", Master's thesis, Bilkent University, 2004.

[33] Hayati Çam. "Query Processing for An MPEG-7 Compliant Video Database", Master’s thesis, Bilkent University, 2008.

[34] Muhammet Baştan, Hayati Çam, Uğur Güdükbay, Özgür Ulusoy. "An MPEG-7 Compatible Video Retrieval System with Integrated Support for Complex Multimodel Queries", IEEE Multimedia, August, 2009.

[35] Tony C. T. Kuo, Arbee, L. P. Chen, "A Content-Based Query Language for Video Databases", pp: 209-214, 2000.

[36] Tony C. T. Kuo, Arbee, L. P. Chen, "Content-Based Query Processing for Video Databases", IEEE Multimedia, March, 2000.

[37] A. Şimşek. "Ontology-Based Spatio-Temporal Video Management System", Master’s thesis, METU, 2009.

[38] M. Koprulu, N. K. Cicekli, and A. Yazici. "Spatio-Temporal Querying In Video Databases", In Information Sciences, pages 131–152, 2004.

[39] Y. Yildirim, T. Yilmaz, A. Yazici. "Ontology-Supported Object and Event Extraction with a Genetic Algorithms Approach for Object Classification", In CIVR’07: Proceedings of the 6th ACM international conference on Image and video retrieval, pages 202–209, New York, NY, USA, 2007. ACM.

[40] Y. Yildirim, A. Yazici. "Ontology-Supported Video Modeling and Retrieval", pp:28-41, June, 2007.

[41] Y. Yildirim. "Automatic Semantic Content Extraction In Videos Using a Spatio-Temporal Ontology Model", PhD thesis, METU, 2009.

[42] S. Subhash Wattamwar, H. Ghosh. "Spatio-Temporal Query for Multimedia Databases", In MS ’08: Proceeding of the 2nd ACM workshop on Multimedia semantics, pages 48–55, New York, NY, USA, 2008. ACM.

[43] A. D. Bagdanov, M. Bertini, A.D. Bimbo, G. Serra, C. Torniai. "Semantic Annotation and Retrieval of Video Events Using Multimedia Ontologies", pp:713-720, IEEE Multimedia, 2007.

[44] F. A. Aygül. "Natural Language Query Processing in Ontology Based Multimedia Databases", Master’s thesis, METU, 2010.

[45] W. Ren, S. Singh, M. Singh, and Y. S. Zhu. "State-of-the-art on Spatio-Temporal Information-Based Video Retrieval", Pattern Recognition, 42(2):267–282, 2009.

[46] J. Z. Li, M. T. Özsu, D. Szafron, "Modeling of Moving Objects In a Video Database", Proceedings of IEEE International Conference on Multimedia Computing and Systems, Ottawa, Canada, June 1997, pp. 336–343.

[47] J. F. Allen, "Maintaining Knowledge About Temporal Intervals", Communications of ACM 26 (1983) 832–843.

[48] R. H. Güting, "An Introduction to Spatial Database Systems". The VLDB Journal, 3(4):357–399, 1994.

[49] S. Adali, K. S. Candan, S. S. Chen, K. Erol, and V. S. Subrahmanian. "Advanced Video Information System: Data Structures and Query Processing", Multimedia Systems, 4:172–186, 1996.