Fachhochschul-Diplomstudiengang
SOFTWARE ENGINEERING
A-4232 Hagenberg, Austria

New Media for a New Millennium – Semantic Description of Multimedia Content

DIPLOMARBEIT

zur Erlangung des akademischen Grades
Diplom-Ingenieur (Fachhochschule)

Eingereicht von

Hannes Bauer

Betreuer:	Dipl.-Ing. Michael Hausenblas, JOANNEUM RESEARCH, Graz
Begutachter:	Univ.-Prof. Dipl.-Ing. Dr. Witold Jacak

August 2006

Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und die aus anderen Quellen entnommenen Stellen als solche gekennzeichnet habe.

Hagenberg, am 21. August 2006

Hannes Bauer

Contents

Danksagung

Bedanken möchte ich mich bei der Firma JOANNEUM RESEARCH und all ihren Mitarbeitern, die mir die Umsetzung der Diplomarbeit ermöglicht haben. Besonderer Dank gebührt dabei meinen beiden Kollegen *Dipl.-Ing. Michael Hausenblas* und *Dipl.-Ing.(FH) Martin Umgeher*, die mir während der Erstellung dieser Arbeit jederzeit mit Rat und Tat zur Seite standen.

Weiters möchte ich mich bei den Professoren an der Fachhochschule Hagenberg bedanken, die mich während des Studiums begleitet haben. Im besonderen sei hier mein Diplomarbeitsbetreuer seitens der Fachhochschule Hagenberg *Univ.-Prof. Dipl.-Ing. Dr. Witold Jacak* genannt.

Schließlich möchte ich mich bei meiner Familie bedanken, meinen Eltern *Margarete* und *Johann*, die mir das Studium ermöglicht haben. Dank auch meinen beiden Geschwistern *Andrea* und *Harald*, die mir stets mit moralischer Unterstützung zur Seite standen.

Kurzfassung

Diese Diplomarbeit beschäftigt sich mit der semantischen Beschreibung von multimedialen Inhalten, um die Erstellung von neuartigen, non-linearen und interaktiven Film-Produktionen zu ermöglichen. Diese Produktionen haben als Anforderung, bestimmte Medien-Objekte (Video- oder Audioclips) aus einer Datenbasis herauszufiltern, die gewissen formalen Suchkriterien entsprechen. Zum Beispiel könnte ein Videoclip benötigt werden, der ein Interview mit einer bestimmten Person darstellt, eine eher getrübte Stimmung ausdrückt und aus film-technischen Gründen mit einem Kameraschwenk nach rechts beginnt.

In dieser Arbeit wird ein Ansatz gezeigt, der die semi-automatische Annotierung von Medien-Objekten ermöglicht und eine semantische Suche auf Basis von definierten, kontext-bezogenen logischen Entitäten erlaubt. Diese logischen Entitäten werden in einem formalen Raum unter der Ausnutzung von DL-basierten[1] Ontologien erstellt und können in verschiedene Abstraktionslevel unterteilt werden. Die Annotierung der Medien-Objekte wird durch automatisch generierte Horn-Regeln unterstützt.

Aus technologischer Sicht basiert diese Arbeit auf den beiden ausgereiftesten Standards zur Erstellung von maschinen-interpretierbaren semantischen Beschreibungen: MPEG-7 (Multimedia Content Description Interface) wird zur Beschreibung von automatisch extrahierbaren "Low-level Features" (wie Farbe, Form, etc.) verwendet. Die Modellierung von Domänen in einem formalen Raum erfolgt unter Ausnutzung von Technologien aus dem Semantic Web, wie OWL (Web Ontology Language) und Regeln. Die Arbeit konzentriert sich vor allem auf die Abbildung der Features auf logische Entitäten ("bridging the semantic gap") und die darauf aufbauende Erstellung von höheren Abstraktionsleveln. Die Farbe 'blau' (als Feature) am oberen Rand eines Bildes könnte zum Beispiel abhängig vom Kontext einer Produktion den 'Himmel' darstellen, anzeigen, dass es sich um eine Szene handelt, die sich während des 'Tages' abspielt; in einem anderen Kontext wiederum könnte gefolgert werden, dass es sich um das 'Meer' handelt.

[1] engl. description logics – Beschreibungslogik

Abstract

This thesis deals with the semantic description of multimedia content in order to enable the creation of novel, non-linear and interactive movie productions. Those productions comprise the requirement to retrieve from a given pool of media items (i. e. video clips or audio clips) appropriate ones based on semantic queries. For example, there could be the need for a media item that represents an interview of a certain person, expresses bad mood and starts with a pan right camera motion (to fulfill the editory rules).

This work shows an approach to annotate media items in a semi-automatic way, allowing semantic queries based on the definition of context-based, logical enities. Those logical entities are defined in a formal way using DL-based ontologies and are grouped into different levels of abstraction. The annotation of media items is supported through automatically generated Horn rules.

Technologically seen, this work uses two mature standards for machine-processable and semantic-based content description: On the one hand, MPEG-7, the "Multimedia Content Description Interface", which allows the description of low-level features (such as colour, shape, etc.) and on the other hand, Semantic Web technologies as OWL (the "Web Ontology Language") and rules, which allow the formal modeling of a domain. The focus of this work lies on the mapping of low-level features to logical entities in a specific context ("bridging the semantic gap") and the further mapping to entities of a higher level of abstraction. For example the colour "*blue*" (as low-level feature) on top of an image could be translated to logical entities defined in an ontology like "*sky*", "*daytime*" or in another context "*sea*".

Chapter 1

Introduction

This thesis is mainly motivated by the requirements of the NM2 project (cf. chapter 3). NM2 aims at the creation of interactive, non-linear narrations in the domain of moving images. Audio-visual essence has to be provided to a media player based on a logical description of the desired content. Since the NM2 system computes the actual storyline on-the-fly, the retrieval of the audio-visual content has to be performed in near real-time. For example, in one point of the narration there could be a query for some essence that "*is about soccer*", "*is of type interview*" and "*starts with* a *pan left*" camera motion.

In order to permit such retrieval tasks, it is necessary to mark up the essence in a way to make its content understandable for computers. The semantic of the essence has to become explicit through the annotation with metadata that has to be machine-processable and allows the development of software tools satisfying the needs for retrieval tasks.

This work proposes a solution that allows the annotation of audio-visual content with logical entities automatically. Whereas numerous algorithms to extract low-level features do exist, the difficulty lies in *bridging the semantic gap* in order to heave the information onto a formal level. In this context objects of the real world as well as their relations to other objects can be defined. Such a knowledge base provides the basis for intelligent queries on a semantic level.

The thesis comprises two parts. First, the theoretical foundations are laid out. Second, the implementation of the theoretical ideas are performed. Therefore, the objectives of this work are:

- Discussion of existing and related work.
- Development of theoretical foundations, i.e. the definition of Media Semantics Mapping.
- Reference implementation for the Media Semantics Mapping.

- Evaluation of the results.

This thesis is structured as follows: in chapter 2, metadata in the context of multimedia data are elaborated and the basics for the Media Semantics Mapping are described. Chapter 3 introduces the hosting project (NM2) and discusses its requirements. Chapter 4 describes the theoretical underpinning of the Media Semantics Mapping, presents the ontology layout and the handling of rules. In chapter 5 the reference implementation is shown. Finally, a conclusion is given in chapter 6.

Chapter 2

Multimedial Metadata

This chapter describes the foundations of metadata in the context of multimedia data. It introduces what is meant by data, information and knowledge, explains the core multimedia metadata standard being used throughout this work (MPEG-7) and shows the capabilities of Semantic Web technologies. Inhere, the basics for chapter 4, the genuine work of Media Semantics Mapping, are laid out.

2.1 Introduction

In this section the concepts of data, information, and knowledge are discussed. Following [BCM04] and [AN95], this work uses the below listed definitions:

- **Data** is represented by syntactic entities. It simply exists and has no significance beyond its existence, therefore data provides patterns with no meaning; it is input to an interpretation process. It may exist in any form, usable or not. Data is a representation of the world around us, presented as external signals captured by sensors.
- **Information** is data that has been given meaning – it is data in context, i. e. interpreted data. This "meaning" can be useful, but does not have to be.
- **Knowledge** is the appropriate collection of information, such that its intent is to be useful – it is learned information, hence depends on a subjective interpretation of information. Knowledge is information incorporated in an agent's reasoning resources – may it be a human or a piece of software – and made ready for active use within a decision process.

Following example illustrates the above given definitions:

Data	The numbers 50 and 5, that is the sequence of the numerals 5 and 0
Information	Amount of money: 50 EUR, interest rate: 5%
Knowledge	After a year I get 52,5 EUR back (the concept of growth)

2.2 Metadata

There is a large amount of definitions for the term *metadata*. Literally, it is "data about data". According to [MET02] there are several types of metadata. In the context of this work the focus lies on descriptive metadata, i. e. descriptions of multimedia content. Generally, multimedia data carries information that can be expressed on different levels of abstraction w. r. t. the semantics. The following table 2.1 lists those levels together with possible technologies to provide machine-processable descriptions.

Feature Scope	**Example**	**Description**
'physical'	the resolution of an image, e.g. 200x300 pixel	essence internal
low-level features	color histogram of an image	MPEG-7
logical entities	picture of a frog	RDF(S)/OWL
context information	$hasColour(x, Green) \leftarrow$ $type(x, Frog), state(x, Living)$	SWRL, Prolog

Table 2.1: Scope of Metadata Standards in the Domain of Multimedia.

In the context of this work the term *low-level features* is used as distinctive characteristic of multimedia content. There are two modalities in which low-level features can be recognized: audition and vision. Low-level features can, by and large, be extracted automatically. In NM2 (cf. chapter 3), the Multimedia Content Description Interface (MPEG-7) is used to represent such low-level features.

2.3 MPEG-7

MPEG-7, formally named *Multimedia Content Description Interface*, is an ISO/IEC standard developed by the *Moving Pictures Experts Group* (MPEG). It provides a rich set of tools that enable the description of multimedia content to be processed by computational systems. This section gives an overview of the MPEG-7 standard. Additional information can be found in [MPE01, Mar04, BSHT05].
As shown in 2.1 the main elements of MPEG-7 are:

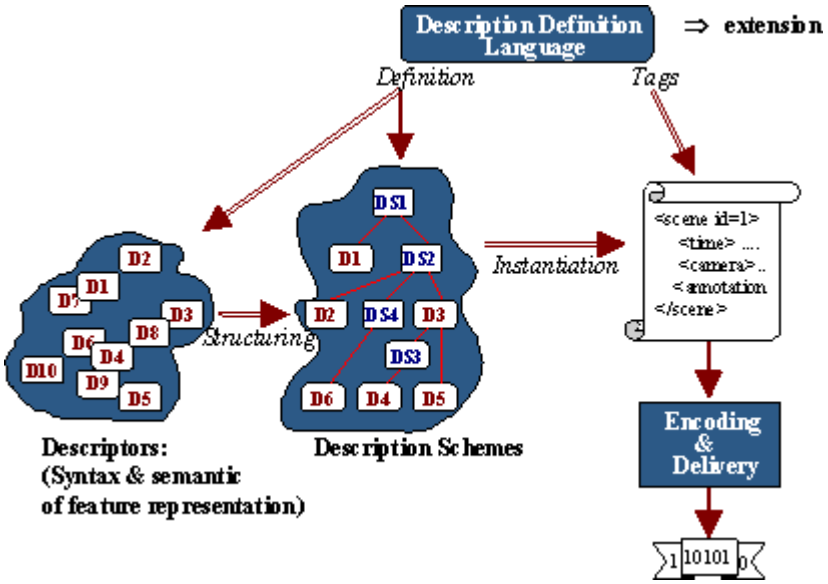


Figure 2.1: Main elements of MPEG-7

- **Descriptors.** They define syntax and semantics of low-level features such as color, shape, texture, camera motion and so forth, but also of high-level features of semantic objects, events and abstract concepts. Low-level features can usually be extracted automatically, whereas high-level descriptors need human interaction.

- **Description Schemes.** They allow the combination of individual descriptors and other description schemes within more complex structures by defining the relationships between those elements. Description schemes specify the structure and semantics of the relationships.

- **Description Definition Language (DDL).** The MPEG-7 DDL is an extension of the XML Schema language and is used for the definition of new descriptors and description schemes. The standardized descriptors and description schemes can be modified or extended.

An MPEG-7 description is created by instantiating description schemes and descriptors. The textual representation in XML can be used for editing, searching and filtering issues, the binary form is suitable for storage, transmission and delivery.

The MPEG-7 standard is organized into the following parts:

1. *MPEG-7 Systems* specifies system tools to prepare MPEG-7 descriptions for efficient transport and storage in binary form.

2. *MPEG-7 Description Definition Language* specifies the DLL.

3. *MPEG-7 Visual* specifies a set of standardized descriptors and description schemes dealing with visual descriptions only.

4. *MPEG-7 Audio* specifies a set of standardized descriptors and description schemes dealing with audio descriptions only.

5. *MPEG-7 Multimedia Description Schemes* specifies a set of standardized descriptors and description schemes for generic features and multimedia descriptions including audio, visual and textual data.

6. *MPEG-7 Reference Software* provides a reference implementation of relevant parts of the MPEG-7 Standard, known as experimentation software (XM).

7. *MPEG-7 Conformance* provides guidelines and procedures for testing the conformance of MPEG-7 implementations.

8. *MPEG-7 Extraction and Use of Descriptions* provides informative material about the extraction and use of some of the description tools.

Following [Mar05], the MPEG-7 Multimedia Description Schemes (MDS) specifies the different description tools (descriptors and description schemes) that are not visual and audio ones, that is, generic and multimedia ones. The MDS specification comprises therefore the major number of description tools of the MPEG-7 standard from the basic structures allowing to create the structure of the description to the description of collections and user preferences, including also the hooks for adding the audio and visual description tools. MDS Description Tools can be grouped in terms of the functionality they provide. Figure 2.2 depicts the different functional groups. The highlighted tools are used within NM2.

According to [BS06], one of the strengths of MPEG-7 is its flexibility, which is provided by a high level of generality. It makes MPEG-7 usable for a broad range of application areas and does not impose too strict constraints on the metadata models of these applications. However, in the practical use of MPEG-7, two main problems arise from these features: complexity and hampered interoperability. The recently proposed profiling partially solves these problems. Profiles are subsets of MPEG-7 tools which cover a certain functionality, while levels are further restrictions of profiles in order to reduce the complexity of the descriptions. In NM2 the Detailed Audiovisual Profile (DAVP) [BS06] is used, i. e. all MPEG-7 documents are DAVP-compliant.

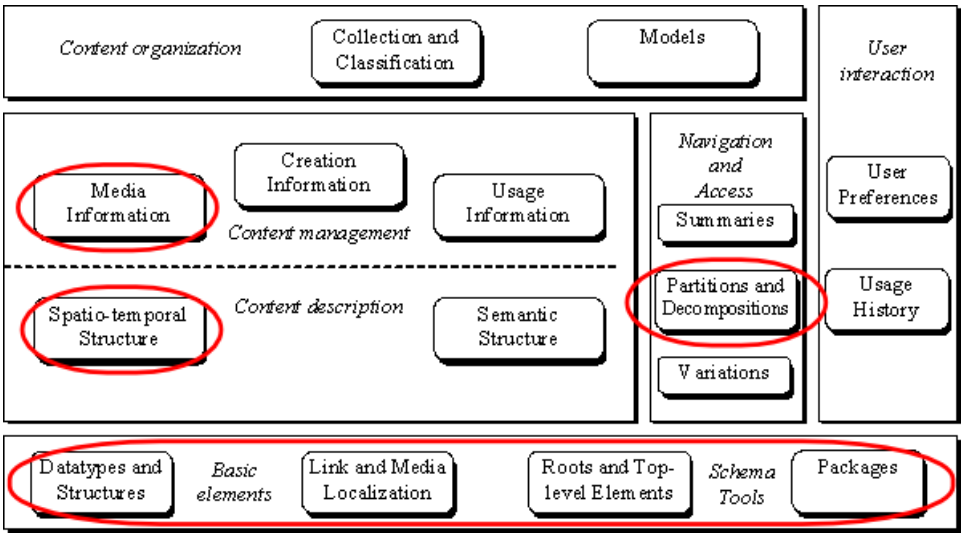


Figure 2.2: MPEG-7 Multimedia Description Schemes (MDS)

2.4 Semantic Web

The World Wide Web Consortium (W3C) describes the Semantic Web in its "Semantic Web Activity Statement" [SWA01] as follows:

> The goal of the Semantic Web initiative is as broad as that of the Web: to create a universal medium for the exchange of data. It is envisaged to smoothly interconnect personal information management, enterprise application integration, and the global sharing of commercial, scientific and cultural data. Facilities to put machine-understandable data on the Web are quickly becoming a high priority for many organizations, individuals and communities.

To fulfill these requirements, the Semantic Web is based on a layered architecture, the so-called *Semantic Web Stack* (cf. figure 2.3). The SW Stack comprises, on the one hand, common standards that allow the interchange of data and, on the other hand, languages that allow to express data in a formal way, so that it can be related to objects of the real world.

2.4.1 Base Technologies

The first and second layer of the Semantic Web Stack deal with the representation of resources, their addressing and their structuring based on following standards:

- **Unicode** [AAB03] is responsible for the encoding of character symbols. It provides a standard set of mappings between machine-processable

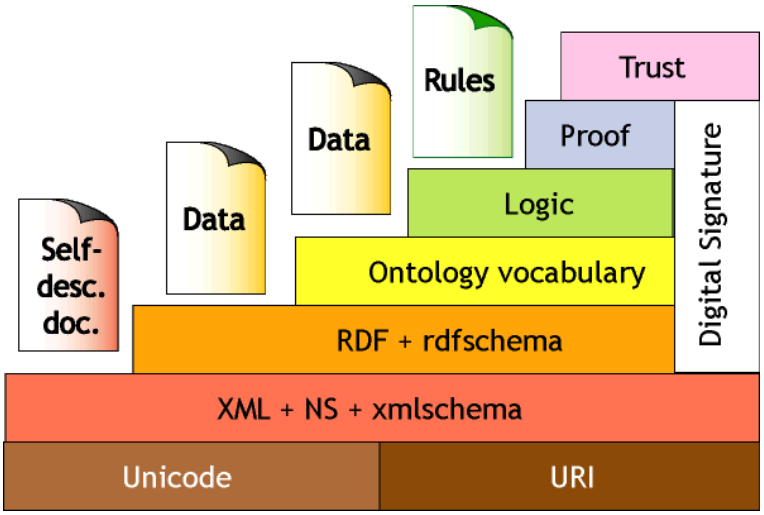


Figure 2.3: The Semantic Web Stack

binary numbers (e.g. 1100001) and symbols (e.g. 'a').

- **Uniform Resource Identifiers (URIs)** [URI] are used to address [ADR] resources, that is naming things uniquely, on the Web.
- The **eXtensible Markup Language (XML)** [XML] together with its schema definition language **XML Schema** [XSD] provide a common model on how to structure data and are used to exchange information as a tree-based, textual representation.

2.4.2 Resource Description Framework

The Resource Description Framework (RDF) [RDF04a] provides an assertion-based logical language – designed for representing machine-processable information of resources identifiable on the Web. An RDF resource is represented by a Uniform Resource Identifier (URI) and provides the natural means to represent and share knowledge. RDF allows to specify triples, each defining a statement that consists of a subject, a predicate and an object (cf. figure 2.4). A set of triples represents an RDF graph (cf. figure 2.5).

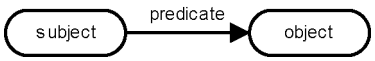


Figure 2.4: RDF Triple

According to [RDF04a], subjects have to be RDF resources or blank nodes (i. e. "anonymous resources"), predicates have to be RDF resources and objects can be RDF resources, blank nodes or RDF literals (i. e. values represented in a lexical form).

2.4.3 RDF Vocabulary Description Language

The RDF Vocabulary Description Language, also called RDF Schema (RDFS) [RDF04b], extends RDF by introducing a basic type system that includes the notions of *class* and *property*. Furthermore, mechanisms to specify class and property hierarchies are provided, as well as global domain and range restrictions for properties. Through these constructs RDFS provides the means to specify the following relationships between classes:

- a class may be a subclass of another one (e. g., *SoccerPlayer* is a subclass of *Person*),
- a class relates to another class through a property (e. g., *Person hasOrigin* in *Country*),
- and the definition of subproperties (e. g., *hasPlayer* is a subproperty of *hasMember*).

This model also provides some reasoning capabilities: provided that the property *hasPlayer* has the class *Team* as its range and the class *SoccerPlayer* as its domain, from the triple X *hasPlayer* Y can be derived that X must be of type *Team* and Y of type *SoccerPlayer*.

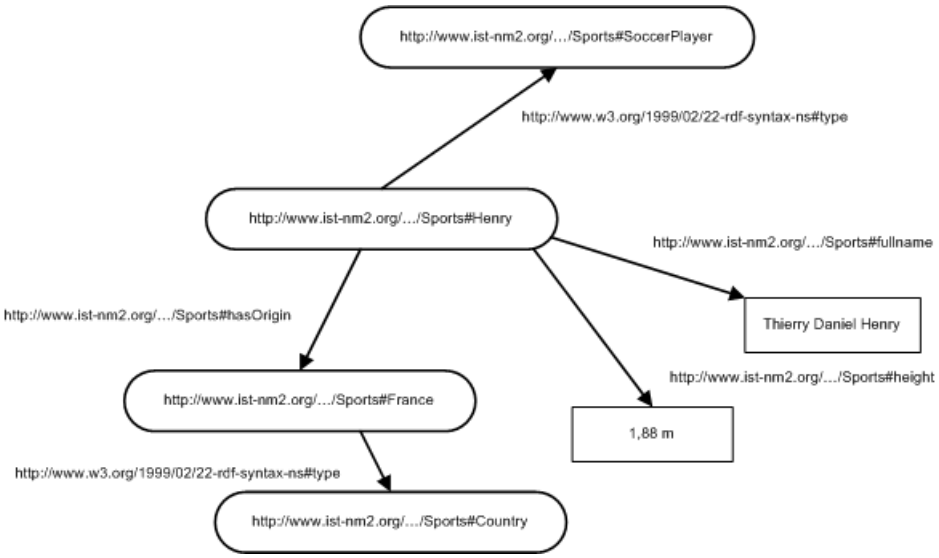


Figure 2.5: RDF Graph

Figure 2.5 illustrates an example of an RDF graph about the resource *Henry*, who is a soccer player, has some literal properties like his full name and height, and originates from France (which in turn is another RDF resource of a certain type).

2.4.4 Ontologies

On top of the RDF(S) layer of the Semantic Web Stack (cf. figure 2.3) lies the ontology layer. An ontology represents a formal conceptualization of a domain to be shared by various applications. RDF(S) itself forms already a simple ontology definition language providing simple ontological primitives, but is not very expressive. Thus, the W3C developed the Web Ontology Language (OWL) [SWM04]: OWL-DL[1] combines the capabilities of RDF(S) with those of Description Logics that constitute a family of knowledge representation formalisms being a decidable subset of first-order predicate logic (FOL). Following [BCM+03], the

> Description Logics [...] represent the knowledge of an application domain by first defining the relevant concepts of the domain (its terminology), and then using these concepts to specify properties of objects and individuals occurring in the domain (the world description).

Furthermore, DLs focus on reasoning tasks, that allow to extend the existing knowledge with intrinsic information.

DL-based ontology languages typically subscribe to the open world assumption, that is a value not present is assumed to be unknown but may reside somewhere else. This contrasts with the closed world assumption found e. g. in relational databases, where null-values are interpreted as being false.

OWL-DL reuses the constructs already defined by RDF(S) (classes, instances and properties) and, in addition, allows to express specific characteristics enhancing expressiveness and reasoning capabilities. For example, classes or properties can be defined to be equivalent or different to another one. Three categories of entities are distinguished within OWL:

Individuals

Individuals, also called instances, represent logical entities of a domain (e. g., Henry's goal in the last match).

Classes

Classes represent groups of individuals that exhibit similar characteristics (e. g., Soccer Players, Countries, etc.). They are organised in hierarchies and can thus be regarded as a taxonomy. Individuals that are member of a class, are also member of all its superclasses.

Following [PvSMK+05], OWL classes can be defined in three different ways:

[1] OWL distinguishes three sub-languages. This thesis concentrates on OWL-DL, which comprises the expressiveness of Description Logics and thus is still decidable.

- axiomatically: by stating that a class exists, e. g., SoccerPlayer.
- intentionally: by defining the membership criteria of a class its instances must fulfill, e. g., person that is member of a soccer team.
- extensionally: by enumerating all individuals that belong to the class, e. g., Country is the class containing Austria, Belgium, ...

Thus, OWL provides much more expressiveness and flexibility to create classes than RDFS does. The ability to express that classes are disjoint provides a formality through which relationships can be checked automatically. This, in turn, provides a powerful way for verifying the consistency of the data model (i. e. the knowledge base).

Properties

There are two categories of properties: object properties and datatype properties. Object properties link individuals to individuals, whereas datatype properties link individuals to simple data values (cf. triples in RDF). When specifying the domain and range of a property, it can only relate instances from the domain class(es) to instances of the range class(es). In case of datatype properties, the range must be a primitive data type, that is a certain XML schema type (e. g. xsd:int, xsd:string, etc.). Like classes, properties are organised in hierarchies. Available characteristics are: FunctionalProperty, InverseFunctionalProperty, TransitiveProperty, and SymmetricProperty.

Reasoning

Again following [PvSMK+05], the formal semantics of OWL allow to perform a range of reasoning tasks that can be categorized into three general types:

1. Class-level reasoning
   - Subsumption: checks if a class is a subclass of another class.
   - Classification: determines the classes that directly subsume or are subsumed by a given class.
   - Satisfiability: checks if a class is satisfiable.
2. Property-level reasoning
   - Computes for all stated relationships the implied ones.
3. Individuals-level reasoning
   - Consistency check: checks if an individual can exist in a given model.

- Realisation: given a partial description of an individual, finds the most specific class that describes it.
- Instance retrieval: finds all individuals that belong to a given class.

2.4.5 Rules-based Systems

Rules-based systems form a branch of Knowledge Representation that – according to [RUL] – can be classified into i) deductive languages, comprising logic (LP) and first order logic (FOL), into ii) reactive rule systems, comprising production rules (PR) and event-condition-action rules (ECA), and into iii) normative rules/integrity constraints.

Commonly available constructs in rules-based languages include negation-as-failure (NAF) and procedural attachments, which are not expressible in first order logic. Due to the fact that SWRL (see below) is used in this thesis, the focus here is put on logic programs, in special horn logic.

A logic program is a set of rules, each having the following form:

$$HEAD \leftarrow BODY$$

The left-hand side of the rule is called the rule's head (consequent); the right-hand side is called the rules body (antecedent); no restriction is placed on the arity of the predicates appearing in the atoms of the head or body. Logical variables and logical functions may appear unrestrictedly in these atoms. A clause is said to be Horn when at most one of its literals is positive. A definite Horn clause – having exactly one of its literals positive – is also known as a Horn rule which can be written in the form:

$$H \leftarrow B_1 \wedge \ldots \wedge B_m$$

Horn-logic (HL) was studied in the area of deductive databases and logic programming. A number of efficient evaluation strategies are available for this fragment of predicate logic. HL languages have declarative semantics defined by the minimal Herbrand model [Llo87]. As described in [RH94], Horn clauses form the basis of Prolog, and are powerful enough to encode Turing machines, hence are undecidable. In the presence of function symbols, terms of arbitrary depth can be constructed and thus the Herbrand model may be infinite. Apart from the declarative, model-theoretic semantics, HL-languages have also procedural semantics, which are defined by an inference procedure called SLD-resolution (Linear resolution with Selection function for Definite clauses [Llo87]). SLD-resolution is a sound and complete inference procedure, meaning that declarative and procedural semantics coincide. However, due to undecidability, SLD-resolution might not terminate.

Semantic Web Rules – SWRL

Following [Mal05] integration of rule languages and ontology languages[2] may be classified by the degree of integration into i) the hybrid approach (strict separation between the ordinary predicates and ontology predicates) and ii) the homogeneous approach (both ontologies and rules are embedded in a logical language without making a priori distinction between the rule predicates and the ontology predicates). Following example illustrates an example for the homogeneous approach:

$$\begin{aligned} &isOriginOf(?p, ?c) \leftarrow \\ &\quad isMemberOf(?p, ?t) \wedge hasNationalTeam(?c, ?t) \wedge \\ &\quad SoccerPlayer(?p) \wedge SoccerTeam(?t) \wedge Country(?c) \end{aligned}$$

From the fact that a certain soccer player is member of the national team of a certain country, it can be concluded that he originates from that country. This rule results in an assertion, possibly not present in the knowledge-base and thus provides new information. The terms used in the rule – as *SoccerPlayer* or *isMemberOf* – originate from the ontology defined in the respective domain.

The Semantic Web Rule Language (SWRL) [HPSB+04] extends the expressivity of OWL-DL at the expense of the decidability of query answering operations. In order to cope with this problem several decidable subsets of SWRL have been identified and investigated, including description logic programs (DLP) [GHVD03] and DL-safe rules [MSS05].

[2] here: in the realm of the Semantic Web

Chapter 3

NM2 – New Media for a New Millennium

"NM2 – New Media for a New Millennium" [NM2b] is an Integrated Project under the European Union's 6th Framework Programme[1] in the thematic priority of Information Society Technology.

This chapter introduces NM2, its vision and main objectives, gives an overview of the six media productions developed within the project, describes the main components of the NM2 system and refers to the special production workflow supported. Furthermore, the commercial and artistic aspects of NM2 are discussed.

3.1 Vision

The vision of NM2 is to create a new media genre by taking into consideration the facilities of modern broadband communication and interactive terminals. NM2 aims at developing tools for the media industry that enable the easy production of non-linear, interactive broadband media: in addition to the high production values and aesthetic pleasures of television and cinema, productions based on NM2 technologies can be personalised and influenced directly by interaction of the engagers according to their personal tastes and wishes.

To provide non-linearity NM2 productions don't stand for final edited pieces of media (like conventional video productions), but consist of a pool of small media units to be combined at run-time to a specific narrative. How the narrative might evolve is influenced by the engager via interaction: documentations and news productions for instance, could be reduced to fields interesting for the consumer. Another example are dramas, where engagers are able to spontaneously decide to see a happy end.

[1] http://fp6.cordis.lu/fp6/home.cfm

3.2 Productions

The tools within the NM2 system are evaluated in six audio-visual productions that range from news reporting and documentaries through a quality drama serial to an experimental television production. In the following, the six productions are introduced as published in the NM2 project brochure [NM2a]:

City Symphonies is a new production in a traditional documentary genre. City Symphonies makes use of an old but recently revitalized screen language – montage – which has proved critical to the history of cinema, and is essential to any understanding of the relationship between cinema and the architecture of the city.

Gormenghast is a fantastical, allegorical version of Mervyn Peake's great novel, originally produced by BBC Television. The content from the production will be developed to allow the story to be explored through new narrative paths, enabling flexible narrative structures in drama to be explored.

Runecast is inspired by the time-honoured oral-storytelling, performance-based structures, which contemporary interactive digital media re-enable in new forms, such as the rule-based interactive aleatoric hypermovie. It uses the NM2 system to enable engagers to compose their own coherent story constellations of songs, tales, music and images from mixed audio-visual media.

MyNews & SportMyWay is a digital, interactive archive that, using the NM2 system and a graphical interface, makes it possible for engagers via broadband to discover, select and recombine news & sports items and stories according to their individual tastes.

Accidental Lovers is a participatory black comedy about love for television, mobile phone and Internet. The engager can affect in real-time the unfolding drama of the unlikely romantic couple, Juulia in her sixties and Roope in his thirties.

A Golden Age is an ambitious configurable documentary exploring the arts of the Renaissance in England, concentrating on the final two decades of Elizabeth I's rule. The engager determines the aspects of this subject which are of most interest, and the system produces in real-time a version which responds to these preferences.

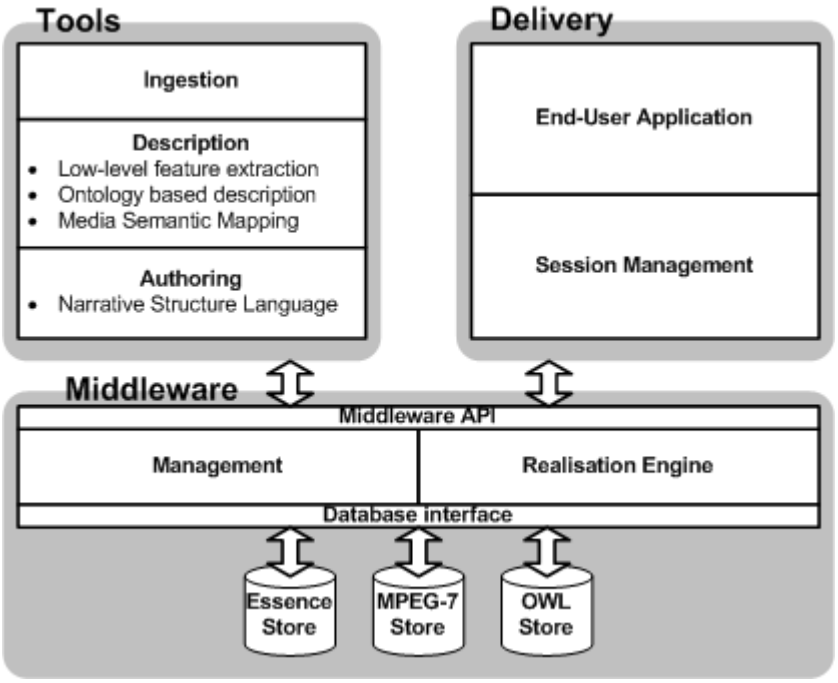


Figure 3.1: NM2 Architecture

3.3 System Architecture

As shown in figure 3.1, the NM2 system consists of the Production Tools, the Delivery System and the Middleware. This section provides an insight into those components and is intended to reveal the requirements for the description part (specified in chapter 4).

3.3.1 Production Tools

The Production Tools are situated in the post-production phase (cf. section 3.4) and support the creators of NM2 based productions to build up stories. They cover the ingestion of essence, the description of media – based on media semantics mapping, and the authoring:

- The *Ingestion Tool* lies at the boundaries of the NM2 system and provides interfaces to non-linear editing (NLE) tools. Importing media from as well as exporting it to those existing production environments (concluding video, image and sound editing systems) is supported.
- The *Media Semantics Mapping Utility* (MSM Utility) enables the definition of logical entities that may occur in the essence. The output is a knowledge base that is used in the Description Tool to (semi)-automatically markup media items.

- The *Description Tool* provides functionality for creating, modifying and deleting media items and is designed for their semi-automatic annotation with metadata. Automatic content analysis of the media items (based on MPEG-7) is supported as well as the (manual) annotation with ontology-based metadata. Results of the automatic description are partially mapped to the logical entities expressed through the ontology metadata (achieved through the MSM Utility).
- The *Authoring Tool* allows producers to create the structure of a non-linear narrative represented by a graph providing several paths from the beginning to the end of a story. So-called "narrative objects" are placed and interconnected onto a workspace and attached with media items (created within the Description Tool). Specific rules and heuristics can be entered to lay out the logic of the respective narrative.

3.3.2 Delivery System

The Delivery System is responsible for presenting the output to the engager and manages interaction. The delivery system is set up in a client-server model that is already supported in many popular domestic devices such as PCs, advanced set-top boxes and game consoles.

3.3.3 Middleware

The Middleware mediates between the Production Tools and the Delivery System by managing and interpreting the metadata and content. It handles all data management tasks and the automatic assembly of media essence. It includes the Realisation Engine, which is responsible for dynamically creating a user-specific story – based on a given story world and the interaction of a particular engager.

3.4 Production Workflow

The conventional media production workflow is a linear process starting with the *pre-production phase* where preliminary tasks like script writing are performed. The shooting itself is done in the *production phase*, followed by the *post-production phase* where the shot material gets edited and prepared.

In opposite, the NM2 workflow, as depicted in figure 3.2, is non-linear and iterative. It can start with the pre-production and production phases (concluding in the ingestion of the shot material), but it is also possible to begin with the designing of a story world in the Authoring Tool and produce the media needed afterwards. Automatic and manual annotation can be performed on the ingested essence via the Description Tool. Changes in the Media Semantics Mapping lead to adaptions in the annotations of the

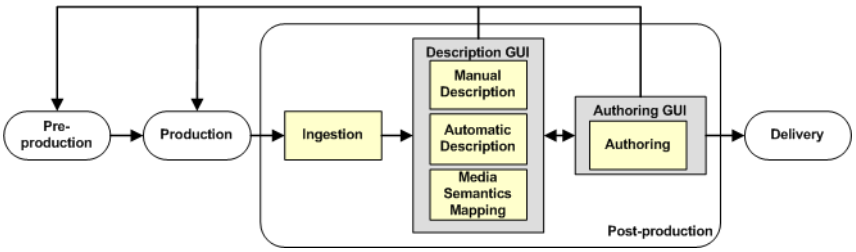


Figure 3.2: NM2 Workflow

media. Anytime new media can be ingested and annotated, as well as the story world can be updated.

3.5 Common Aspects

3.5.1 Impact on Society and Market

Media Experiences

The outcome of NM2 offers new technological capabilities that enable new media experiences based on the ability to provide instant feedback between an audience and a media storyworld. Successful implementation of the experimental productions developed within NM2 enables people to engage with media in new ways, to explore a media space and to enjoy a media experience that is situated somewhere between a computer game and a pure cinematic experience. NM2 may change the way that people spend their leisure time and will introduce them to a new way of exploring and receiving information.

Business

The NM2 project has the potential to stimulate business success in the media industry by allowing creatives to formulate new entertainment formats. Media professionals have to be familiarised with the new capabilities to enable the programme ideas that can only be imagined today. Additionally, the availability of new attractive and engaging entertainment over a broadband link will stimulate broadband take up helping broadband providers to reach an earlier return on the investment they are making in broadband network deployment (cf. partners [NM2c], especially British Telecom).

3.5.2 Artistic Aspects

In computer-enhanced cinematic screen media, new interactive story forms and formats can be achieved both through transforming traditional media content forms and by generating new formats, uniquely enabled by digital media and communications.

In narrative dramas the audience-member is usually invited to become a participator or interactor. In video-based works, interactors or engagers can potentially take part in the composition of a movie, whether in the role of screenwriter, director, editor, or performer. In traditional cinema and television fiction, the audience makes an imaginative identification with characters and stories already carefully designed by the movie-makers.

Interactive production involves a number of new processes, including the redistribution of traditional team roles. Like all productions, interactive movie-making requires skill sets which include an understanding not only of the creative content, but also of the technical possibilities.

Although NM2 does not aim to replace expert storytelling with machine storytelling, in the new production environment its tools and systems do need to translate between authors, machines, and engagers in the story-making process. The design phase of a traditional production, its visualisation phase and its communication phase can helpfully be mapped to the stages of handling material using digital tools – from software architecture through realisation engine to interactive distribution, exhibition and interactive access via broadband.

Chapter 4

Description of Multimedia Content

Within NM2, the central units when talking about multimedia content description are *media items*. A media item refers to some multimedia essence, which can be a video clip or an audio clip and is to provide a machine-processable description of the essence to make its semantics explicit. Such descriptions form the basis for the retrieval tasks and have to enable semantic queries in narrative objects (cf. section 3.3.1: Authoring Tool).

What are media semantics? The media essence itself – in the author's understanding – does not have intrinsic semantics. Essence can be created, delivered, consumed or otherwise manipulated. Nevertheless, the essence "carries" the semantics and it is up to the consumer of the essence to interpret what she understands from it (based on some contextual knowledge). In NM2, the semantics of media items is made explicit in two ways: first, by generating and attaching an MPEG-7 description (cf. section 2.3) of the essence. Second, through the annotation with logical entities defined in an ontology (cf. section 2.4.4) that represents a specific domain. The major task lies in *bridging the semantic gap*, that is to map media intrinsic information (captured within MPEG-7 descriptions) to logical entities.

According to Harel and Rumpe [HR04], any language definition comprises a syntax, a semantic domain and a semantic mapping from the syntactic elements to the semantic domain. Although the criticism of Harel and Rumpe is originally targeted at the fuzzy semantics of the Unified Modeling Language (UML), the key question can equally be applied to the domain of media. In the realm of NM2, the usage of MPEG-7 descriptions and OWL-DL ontologies to formally capture the metadata yields a simple sort of description language, hence forming the syntactical base. The domain of a production is represented by defined logical entities and their occurrence(s) in the essence (cf. Grosky [Gro94]). Introducing definitions of logical entities (performed by the user) leads to the automatic generation of rules.

NM2 allows both, manual and automatic description of media items, whereas the focus lies on automatizing as much as possible. Based on the related work introduced in section 4.1, this chapter describes an approach, how to extract information out of media essence (section 4.2) and lift this information onto a formal space (section 4.3), which is represented by an ontology (section 4.4). The combination of the ontology with a rule base (section 4.5) provides the powerful means to infer new information and augment the semantics of the description.

4.1 Related Work

Thoughts on how to conceptually bridge the semantic gap w. r. t. media are probably as old as the multimedia content itself [Gro94]. Most of the work in the realm of multimedia understanding in the last decade shows strong efforts to combine multimedia metadata (as MPEG-7) with logic-based languages (as RDF(S)/OWL). Some proposed solutions suffer from weak formalisation, others are only research prototypes, hence do not scale. Often a single modality is taken into account or the solution focuses on a specific domain (e. g. cells, planes, artwork, etc.).

An often referenced work is that by Hunter et.al. [Hun01] – one of the early approaches to "marry" the ISO/MPEG-7 and the W3C/Semantic Web-community. In [Tro03] an attempt is made to capture structural as well as conceptual aspects including reasoning support. Other recent approaches that model parts of the MPEG-7 standard in OWL are described in [GC05, VKSB06]

For the field of ontology-based video retrieval, [TPC04] reports a methodology to support interoperability of OWL with MPEG-7. An Upper Ontology that fully captures the semantics of the MPEG-7 Multimedia Description Schemes (MDS) is defined that enables the integration of domain-specific knowledge in multimedia content applications through the extension of the Upper Ontology with domain ontologies.

Further related work can be found in [ATP+05]. In this paper, a knowledge assisted analysis (KAA) platform is described. The interaction between the analysis algorithms and the knowledge is continuous and tightly integrated, instead of being just a pre- or post-processing step in the overall architecture. A matching process queries the knowledge base and assigns each region with a list of possible concepts along with a degree of relevance.

Other attempts closely related to the work presented herein are [HR03] that use rules to construct semantic descriptions for LOMs and [MB03] which proposes a similar approach regarding abstraction levels. [NMV+05] reports an approach to express behavior with rules in combination with ontologies and MPEG-7.

Audio content analysis and description has been a comparably active re-

search field in the last 20 to 30 years. MPEG-7 Audio, in conjunction with the Multimedia Description Schemes part of the standard, provides structures for describing audio content. These structures are a set of low-level descriptors, for audio features across many applications (spectral, parametric, and temporal features of a signal) as well as high-level description tools that are more specific to specific application domains. Those high-level tools include general sound recognition and indexing, instrumental timbre, spoken content, audio signature, and melodic tools to facilitate query-by-humming. Audio analysis and retrieval applications are usually based on Independent Component Analysis of spectral components (ICA) and classifiers based on continuous Hidden Markov Models. Related work regarding the task of classification in the audio domain can be found in [BNUA04, BK02, KMS04].

Motivated by the promising work reported in [LH04, HLH05] and taking into account the analysis given in [vNH04, vNH05], the approach presented herein is based on the experiences with MPEG-7 annotation and retrieval [BSHT05, CdCC+05].

4.2 Automatic Content Analysis

In NM2, MPEG-7 (cf. section 2.3) is utilized for capturing intrinsic low-level features of multimedia essence. The Multimedia Mining Toolbox [BSHT05] (cf. figure 4.1) allows to produce MPEG-7 descriptions by performing an automatic content analysis of the essence and extracting the below-mentioned features:

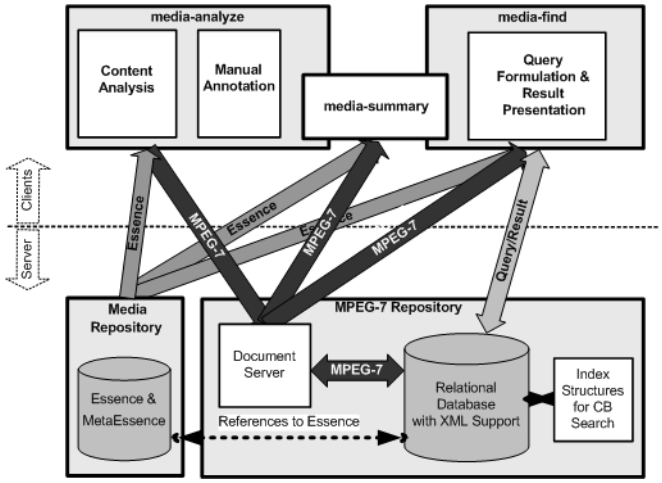


Figure 4.1: Multimedia Mining Toolbox

In the visual domain, the Dominant Color Descriptor and the Color Layout Descriptor are used to capture colour features. To describe textures,

the Edge Histogram Descriptor is used. Shapes can be recognized via the Contour-Based Shape Descriptor. The Camera Motion Descriptor is utilized to describe camera movements (including pans, tilts, zooms, etc.). In the audio modality the focus lies on the classification of audio signals, i. e. the recognition of speech as such, music, studio foleys and other reasonable categories. This is realized by supporting both, MPEG-7 low-level descriptors (as Wave Form, Power) and enhanced classification descriptions.

For the management of MPEG-7 documents within NM2, the MPEG-7 Document Server is used. As described in [BSHT05], this component provides read/write access to MPEG-7 documents for a number of clients and allows the exchange of whole documents or fragments thereof, which are addressed by XPath statements. Access to parts of documents is crucial for the efficiency of the system, as MPEG-7 XML documents of larger media resources tend to have considerable size. The infrastructure used to persistently store the document is abstracted, i. e. both file and RDBMS storage of MPEG-7 documents is available. MPEG-7 documents are compliant with the Detailed Audiovisual Profile (DAVP) [BS06].

The result of the automatic content analysis are sound MPEG-7 descriptions of media essence. They can be created automatically, but do not fulfill the requirements for allowing efficient semantic querying, which is essential within the NM2 system. Therefore, it is necessary to bridge the semantic gap and heave the information onto a formal level – expressed by an ontology. In opposite to the approaches shown in [Hun01, GC05, VKSB06], in NM2 it isn't tried to design an ontology that maps the standardized MPEG-7 descriptors and description schemes. Instead, MPEG-7 fragments (each of them describing a certain feature) are referenced from within logical entities and combinations of those 'primitives' are mapped to more complex entities. This approach is elaborated on it detail in the following section.

4.3 Media Semantics Mapping

On the ontological level, two orthogonal conceptual paradigms are used to model media semantics: spaces and classes/instances. A *space* represents a certain level of abstraction, ranging from media intrinsic low-level features to abstract logical entities. Within the class/instance level, logical entities are defined and grouped. The definition of logical entities takes place on the instance level. Therefore, the "soccer ball" instance in the context of a soccer game is defined to be black, white, and of circular shape but this does not mean that a "ball" in general – referring to the class level – has these properties.

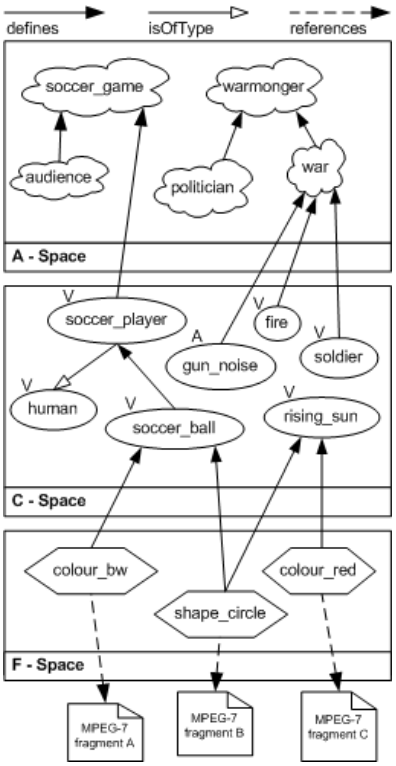


Figure 4.2: The Media Semantics Spaces

As depicted in figure 4.2, the following spaces are available for modeling logical entities, listed by increasing level of abstraction:

1. The **Feature Space (F-Space)** contains logical entities that represent low-level features. A low-level feature (`LLF`) is a single aspect of a certain (spatio-temporal) part of a media item, for example the colour of a spatial region, tone pitch sequence, etc.

2. The **Concrete Entity Space (C-Space)** contains logical entities that can directly be recognized in the essence. A concrete logical entity (`CLE`) is a distinct object being defined by a combination of low-level features and their respective values (simple `CLE`) or using other concrete logical entities (composite `CLE`).
   For example, in the context of a soccer game, the "soccer ball" may be defined by the `LLFs` dominant colour black and white and circular shape. The penalty area may be defined by the `CLEs` goalkeeper and penalty spot.

3. The **Abstract Entity Space (A-Space)** contains logical entities that are not directly observable. An abstract logical entity (`ALE`) can

be defined by a combination of `CLEs` (simple `ALE`) or other `ALEs` (composite `ALE`).
Example: In the context of a news production, the `ALE` "war" may be defined by the simultaneous presence of soldiers, gun noise, and fire.

Subclasses of `LLF` are exhaustively defined by the supported MPEG-7 descriptors (cf. section 4.2). Instances of `LLF` refer to MPEG-7 fragments and form the means for bridging the semantic gap: a media item contains and can therefore be annotated with a certain `LLF` if the (automatically generated) MPEG-7 description of the media item comprises the fragment the `LLF` refers to. The feature values can be defined by plain values (e. g. `RGB: 255, 0, 0` for Dominant Color Descriptor instantiations) or by using reference media documents, that is images or audio clips.

4.4 Ontology Layout

Based on the concepts introduced above, the main classes and properties of the NM2 core ontology have been designed as illustrated in figure 4.3:

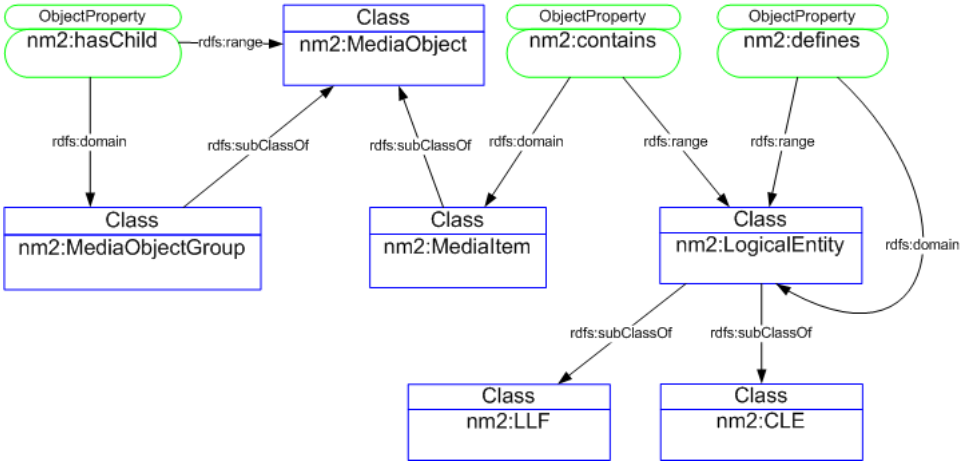


Figure 4.3: Core Ontology (Overview)

Individuals of the class `MediaItem` refer to an essence and the according MPEG-7 description, have `Modality` audio or video, and can be annotated with `LogicalEntity` individuals through the `contains` property. The `defines` property allows to realize the media semantics mapping as described above.

For each of the six NM2 productions (cf. section 3.2) a domain-specific ontology is defined. These production-specific ontologies define concrete and abstract logical entities in the respective domain. In case of the documentary

production about England's Golden Age (16^{th} century) potential entities are: church, kingdom, paintings etc.; in a news production domain possible entities are: sports, economy, and politics.

4.5 Combining Ontologies with Rules

DL-ontology languages have some well-known limitations with regard to the expressiveness of properties. As described in [HPSBT05], there is no possibility in OWL to capture relationships between a composite property and another property. Figure 4.4 illustrates the standard example for this problem: the composition of parent and brother properties relates to the uncle property – the facts that an individual "*x has a parent y*" and "*y has a brother z*" obviously comprise the information that "*x has an uncle z*", but OWL does not provide the means to accomplish this inference step. In

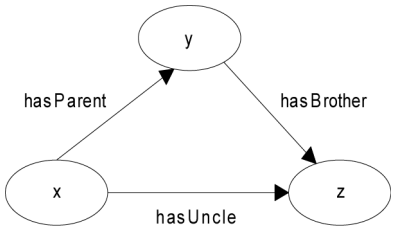


Figure 4.4: The standard "Uncle Rule" example

order to overcome these limitations, rules (cf. section 2.4.5) are used. The rule for the given example would be:

$$hasUncle(?x, ?z) \leftarrow hasParent(?x, ?y) \wedge hasBrother(?y, ?z)$$

The knowledge base within NM2 is defined as follows:

$$\mathcal{KB}_{MSM} = (\mathcal{O}_D, \mathcal{R})$$

$\mathcal{O}_D$ is an ontology that consists of a T-Box (classes, "ontology schema") and an A-Box (instances, "knowledge items"), and
$\mathcal{R}$ is a rule-base including rules.

The rule-base $\mathcal{R}$ is represented using the Semantic Web Rule Language (SWRL) [HPSB$^+$04] to ensure a homogeneous format w. r. t. the ontology $\mathcal{O}_D$. For the purpose of applying $\mathcal{R}$ onto $\mathcal{O}_D$, the SWRL representation encoded in RDF/XML is converted into a number of Prolog rules. The T-Box and the A-Box of $\mathcal{O}_D$ represented in OWL-DL and encoded in RDF/XML are converted into a number of facts in Prolog. SWI-Prolog (http://www.swi-prolog.org/) and its Semantic Web library is used to infer new RDF triples. The outcome of the inference process is reflected in an update of the A-Box of $\mathcal{O}_D$. Two general types are distinguished: Built-in Rules and User-defined Rules.

4.5.1 Built-in Rules

The Built-in Rules build a small set of automatically generated rules and are used to extend the knowledge base with inferred facts.

The concepts introduced so far allow a "static representation" of the knowledge Based on the constructs of the core ontology (cf. 4.4), on the one hand, media items can be annotated with logical entities. On the other hand, it is possible to map logical entities to higher abstraction levels using the `defines` property. A media item annotated with a set of LLFs that define a CLE, should therefore be annotated automatically with that CLE. The following rule is used to create this new fact that can be added to the knowledge base:

$$\begin{aligned} &contains(?mi, ?c) \leftarrow \\ &\quad defines(?l_1, ?c) \wedge \ldots \wedge defines(?l_i, ?c) \wedge \\ &\quad contains(?mi, ?l_1) \wedge \ldots \wedge contains(?mi, ?l_i) \wedge \\ &\quad LLFeature(?l_1) \wedge \ldots \wedge LLFeature(?l_i) \wedge \\ &\quad CLEntity(?c) \wedge MediaItem(?mi) \end{aligned}$$

This rule can be adopted to definitions of logical entities in other abstraction levels respectively.

The second rule states that the modality of a feature co-determines the modality of the media item it is contained in: if a feature is known to have a certain modality, then the media item that contains this feature must also have the same modality:

$$\begin{aligned} &hasModality(?mi, ?md) \leftarrow \\ &\quad hasModality(?l, ?md) \wedge contains(?mi, ?l) \wedge \\ &\quad LLFeature(?l) \wedge Modality(?md) \wedge MediaItem(?mi) \end{aligned}$$

The assembly of the media items to a story is done by the Realisation Engine (cf. section 3.3.3). In order to be able to create smooth transitions, the need for providing some information on how a media item starts or ends is obvious. In the visual domain this kind of information is captured for all camera motions. In the audio domain this is done in the general case because the cut in the audio modality almost always is critical w.r.t. sound intensity and phase. Similar to [PJ01] a set of temporal semantics w.r.t. audio-visual is defined. The example shown below demonstrates the usage of temporal semantics. For each `LLF` (Φ) that is of type camera motion (pan, tilt, zoom, etc.) or any supported audio feature, the following rule is applied:

$$endsWith(?mi, ?\Phi) \leftarrow containsAtEnd(?mi, ?l) \wedge \Phi(?l)$$

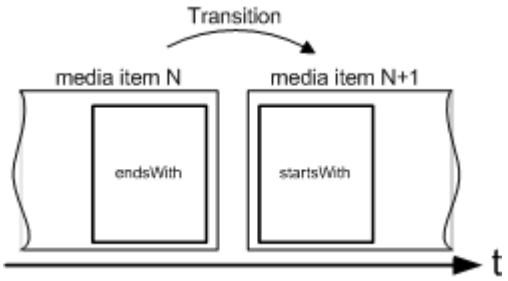


Figure 4.5: Transitions in A/V-essence

As depicted in Fig. 4.5, the information how a media item ends (respectively starts) can be made explicit. This information can then be used to support the creation of a specific narrative embodying the directors style (so-called editory rules), e.g. "never let a pan left directly follow a pan right".

4.5.2 User-defined Rules

To enhance the production-specific ontologies, the definition of user-defined rules – that are entered manually by the user – is granted. In opposite to built-in rules, where the properties are predefined (*defines*, *contains*, etc.), user-defined rules are based on relations. Relations – created by the user by specifying label, domain and range – are subproperties of `relation` (defined in the core ontology) that restricts the domain and range to subclasses of `LogicalEntity`.

Figure 4.6: An inferred property based on a user-defined rule

The following example taken from the news production domain illus-

trates a user-defined rule based on three relations:

$$
\begin{aligned}
nsw : isOriginOf(?player, ?country) \leftarrow \\
\quad nsw : isMemberOf(?player, ?team) \wedge \\
\quad nsw : hasNationalTeam(?country, ?team) \wedge \\
\quad nsw : SoccerPlayer(?player) \wedge \\
\quad nsw : SoccerTeam(?team) \wedge iso : Country(?country)
\end{aligned}
$$

In Figure 4.6 the inferred property (`isOriginOf`) is shown. The figure also illustrates the usage of a subclass (`SoccerPlayer rdfs:subClassOf Human`) and the incorporation of an existing, external ontology (`country rdf:type iso:Country`).

The usage of such rules is rather powerful, but obviously holds the risk of defining a set of rules that are incompatible.

Chapter 5

Implementation

The prototype implementation of the concepts introduced in chapter 4 comprises the definition of object models for accessing the ontology data and tools for i) defining logical entities, ii) performing a mapping of logical entities to a higher level of abstraction and iii) semi-automatically annotating media items with logical entities.

In opposite to the the specification in section 4.3, the implementation comprises only two spaces (i. e. levels of abstraction of logical entities): the low-level feature space (F-Space) and the concrete logical entity space (C-Space). It turned out to be difficult to define the boundaries to more abstract spaces (i. e. the abstract logical entity space – A-Space). Nevertheless, the implementation was kept generic, so that additional spaces can easily be introduced.

Section 5.1 describes the object models used to represent media items and logical entities as C++ objects. Section 5.2 gives an overview of the components that are involved in the mapping from data of an RDF graph to C++ objects and vice versa. In section 5.3, the most important user interface components of the MSM Utility and the Description Tool are presented. Finally, section 5.4 deals with the generation of rules to extend the knowledge base with additional information.

The implementation is C++ based and uses the Qt framework [QT] mainly for the graphical user interface, and the Redland RDF Application Framework [Bec01] for the handling of RDF graphs.

5.1 Object Models

According to chapter 4, *media items* represent the central units in the description part of NM2. They refer to some multimedia essence (video or audio clips) and have to provide the description of the essence to make its semantics explicit. The description comprises, on the one hand, an automatically generated MPEG-7 description, on the other hand, annotations

with *logical entities*. Logical entities are defined on a formal level and can be classified into low-level features (LLFs) and concrete logical entities (CLEs). In the following, the object models for media items and logical entities are introduced.

5.1.1 Media Items

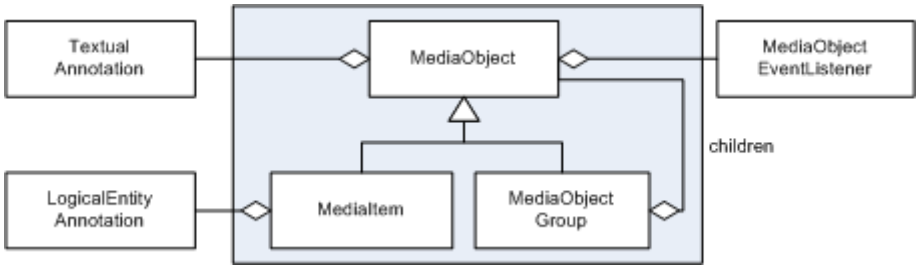


Figure 5.1: Media Object Composite

As depicted in figure 5.1, `MediaObject` (`MO`) is a base class with the two specific subclasses `MediaItem` (`MI`), and `MediaObjectGroup` (`MOG`). According to the composite pattern [GHJV95], `MI` is a primitive class representing a media essence and its annotations; `MOG` represents a collection of `MOs`, which in turn can be `MIs` and/or further `MOGs` – providing the means to build a tree structure.

Two types of annotations can be distinguished:

- *Textual annotations* are implemented as (textual) key-value pairs. They are attachable to both, `MIs` and `MOGs`, and have to be regarded as extendable property system. `MI` provides some predefined properties that are implemented as textual annotations. The most important ones represent the locations of the essence and the MPEG-7 description files, and can be accessed via getting/setting methods.

- *LogicalEntity annotations* are represented by logical entities (cf. section 5.1.2). They are attached to `MIs` and describe the contents of the media essence.

`MediaObject` has an id (to uniquely identify a `MO` instance), a name (to provide a human readable label), a list of listeners that are informed of changes (cf. Gamma et al. [GHJV95], Observer Pattern), and a list containing textual annotations. The interface of `MediaObject` basically declares operations for setting/getting the name and id, adding/removing listeners and methods for checking the dynamic type and casting to it. Furthermore, it allows to add properties in the form of textual annotations.

`MediaItem` inherits from `MO` and has as additional member a list of logical entities representing the annotations. Its interface extends `MO` with methods for adding/removing annotations and for setting/getting predefined properties (such as essence location, MPEG-7 location, in time, out time, etc.).

`MediaObjectGroup` also inherits from `MO` and has as additional member a list of children. It provides methods to add/remove one or more children and to get the children or all descendants (including the children's children). Adding or removing logical entity annotations can also be performed on `MOGs`, which leads to delegations of the call to each of its children.

5.1.2 Logical Entities

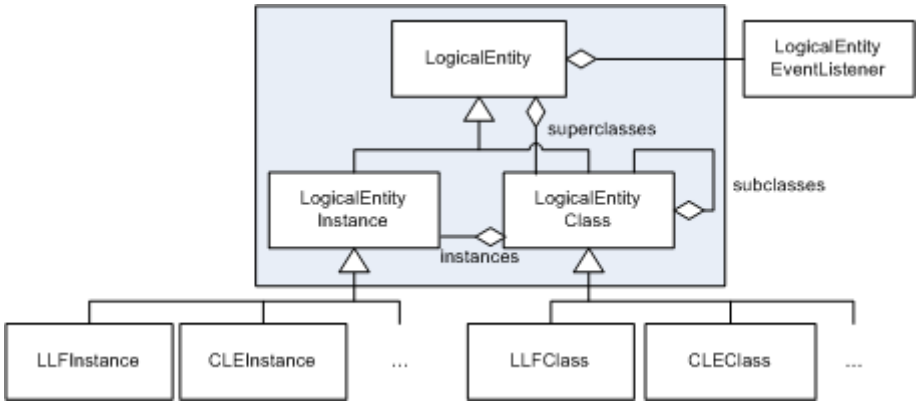


Figure 5.2: Logical Entity Composite

The `LogicalEntity` (`LE`) class hierarchy as illustrated in figure 5.2 is also organized as a composite, which is most suitable for the mapping of ontology data: classes, which form groups in OWL[1], are mapped to instances of `LogicalEntityClass` – being the container class of the composite pattern. Instances[2] are represented through `LogicalEntityInstance`.

`LogicalEntity` has as members an id (to uniquely identify an LE instance), a name (to provide a human readable label), a list of superclasses (of type `LogicalEntityClass`) and a list of `LogicalEntityEventListener` elements that are informed of state changes. The interface of `LogicalEntity` basically declares operations for setting/getting the name and id, adding/removing superclasses, registering listeners, and methods for checking the dynamic type and casting to it.

`LogicalEntityInstance` inherits from `LE` and provides additional methods to dynamically cast to its concrete sublcasses. These represent low-level

[1] In OWL, different classes may contain the same instances, which differs from the common use of the terms *classes* and *instances* in object-oriented programming.

[2] Instances form the A-Box of an ontology and represent the "knowledge items".

features of the F-Space (`LLFInstance`) and concrete logical entities of the C-Space (`CLEInstance`). Elements of both types are used to annotate media items with.

`LogicalEntityClass` also inherits from `LE` and provides additional methods to dynamically cast to its concrete sublcasses – representing the classes of low-level features of the F-Space (`LLFClass`) and the classes of concrete logical entities of the C-Space (`CLEClass`). Furthermore, it provides factory methods (cf. Factory Method Pattern [GHJV95]) for creating subclasses and instances: the instantiation is deferred to the concrete subclasses, each of them creating new elements that belong to their own space.

5.2 Data Handling

Figure 5.3 shows the components that are responsible for generating the object models (as described in the previous section) from the data stored in an OWL file and for serializing the object models back to OWL.

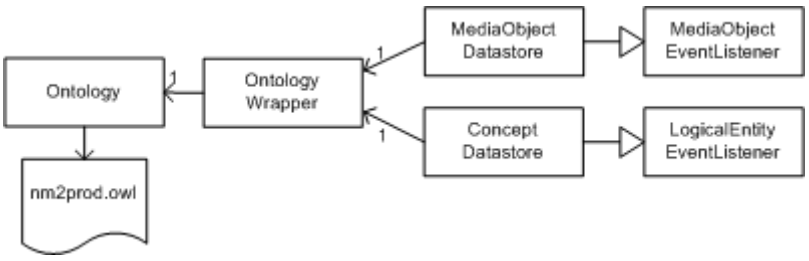


Figure 5.3: Data Handling

5.2.1 Ontology

For RDF/OWL processing, the mature and high-performance Redland RDF Application Framework [Bec01] is used, which comprises several C libraries to parse and serialize RDF graphs (in various notations), an API to manipulate RDF graphs, and a storage API providing in-memory, file-based and RDBMS-based storage.

Within NM2, the class `Ontology` wraps the Redland API and provides an object-oriented C++ interface. Therefore, `Ontology` can be regarded as abstraction layer, as it allows to switch the backend implementation for handling RDF graphs from the Redland framework to any other.

`Ontology` declares operations to load an RDF graph stored in an RDF/OWL file into memory, to add new resources, manipulate and delete existing ones, and to write back the manipulated graph to a file. The prototype implementation does not provide an RDBMS backend, but an extension can easily be added. As Redland is based on RDF, the functionality

to create OWL classes and OWL properties was added to the API in order create OWL compliant graphs. A major functionality of `Ontology` are the querying capabilities based on SPARQL [SPA04]. The interface provides a method to perform common SPARQL (select) queries on the graph loaded into memory, and some convenience methods to get the subclasses or instances of a specific class, the subproperties of a given one, or all values of a given property.

5.2.2 OntologyWrapper

`OntologyWrapper` provides convenience methods for the `Ontology` interface. The creation of a class, for example, results in several method calls in `Ontology` to update the RDF graph: creating a new resource (of type `owl:Class`), creating the subclass relation to an existing resource (`owl:SubclassOf`) and creating a label (`rdf:label`) for the resource. Thus, `OntologyWrapper` can be regarded as additional abstraction layer, allowing to change the backend implementation responsible for the RDF/OWL handling. Furthermore, `OntologyWrapper` is responsible for creating unique identifiers (URIs) for resources to be added to the RDF graph.

5.2.3 MediaObjectDataStore

`MediaObjectDataStore` (cf. listing 5.1) is responsible for the management of media objects and is part of the data business model. It provides a dummy element of type `MediaObjectGroup` that comprises no information but serves as root group of all media objects – forming a tree of group and item elements[3]. The interface declares operations for getting the root group (as parent of the whole hierarchy), and for getting any media object with a given id.

```
class MediaObjectDataStore :
  public MediaObjectFactory, public MediaObjectEventListener {
public:
  virtual MediaObjectGroup* getRoot() = 0;

  virtual MediaObject* getMediaObject(const QString& id) = 0;
};
```

Listing 5.1: MediaObjectDataStore Interface

Moreover, `MediaObjectDataStore` implements the `MediaObjectFactory` interface (cf. listing 5.2) providing factory methods for the creation of new media objects (either groups or items), and the `MediaObjectEventListener` interface (cf. listing 5.3) to get notified whenever the state of a media object

[3]Strictly speaking, the media object hierarchy is a graph but no tree, because the same media object can be containee of different groups. Nevertheless, the term *tree* is used, as it is granted that a group cannot be a (direct or indirect) child of itself (avoiding any cycles in the graph).

changes. The changes are delegated to the `OntologyWrapper` component and lead to an update of the RDF graph.

```
class MediaObjectFactory {
public:
  virtual MediaItem* createItem(MediaObjectGroup* parentGroup=0) = 0;

  virtual MediaObjectGroup* createGroup(MediaObjectGroup* parentGroup
      =0) = 0;
};
```

Listing 5.2: MediaObjectFactory Interface

```
class MediaObjectEventListener {
public:
  virtual void mediaObjectChanged(const MediaObjectEvent* event) = 0;
};
```

Listing 5.3: MediaObjectEventListener Interface

5.2.4 LogicalEntityDataStore

`LogicalEntityDataStore` (cf. listing 5.4) is responsible for the management of logical entities and forms the second major part of the data business model. It provides a dummy element of type `LLFClass` that serves as root for all logical entities in the F-Space, as well as a dummy element of type `CLEClass` that serves as root for all logical entities in the C-Space. These two class/instance hierarchies are independent from each other. Logical entities (classes as well as instances) may have more than one superclass – the two class/instance hierarchies (for the F-Space and the C-Space) therefore represent directed graphs. The `LogicalEntityDataStore` interface declares operations for getting the root classes of the two hierarchies and for getting any logical entity with a given id. Moreover, it implements the `LogicalEntityEventListener` interface (cf. listing 5.5) to get notified whenever the state of a logical entity changes – leading to an update of the underlying RDF graph.

```
class LogicalEntityDataStore : public LogicalEntityEventListener {
public:
  virtual LogicalEntityClass* getLLFRoot() const = 0;
  virtual LogicalEntityClass* getCLERoot() const = 0;

  virtual LogicalEntity* getLogicalEntity(const QString& id) const =
      0;
};
```

Listing 5.4: LogicalEntityDataStore Interface

```
class LogicalEntityEventListener {
public:
  virtual void logicalEntityChanged(LogicalEntityEvent* event) = 0;
};
```

Listing 5.5: LogicalEntityEventListener Interface

5.3 User Interface

For implementing the graphical user interface the Qt framework [QT] is used, which provides in addition to common GUI widgets a set of model/view classes and interfaces [QTM] to realize the Model-View-Controller (MVC) design pattern. According to Gamma et al. [GHJV95],

> MVC decouples views and models by establishing a subscribe/notify protocol between them. A view must ensure that its appearance reflects the state of the model. Whenever the model's data changes, the model notifies views that depend on it. In response, each view gets an opportunity to update itself. This approach lets you attach multiple views to a model to provide different presentations. You can also create new views for a model without rewriting it.

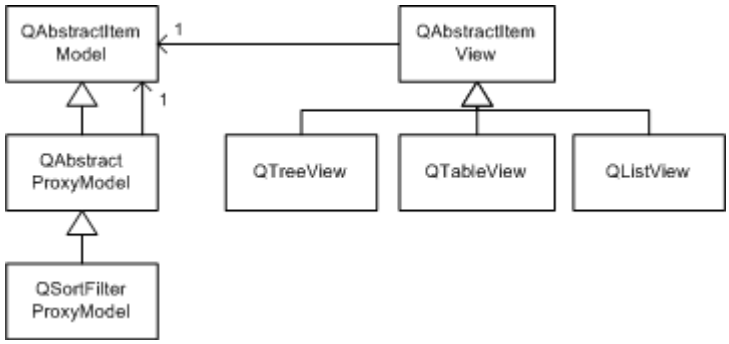


Figure 5.4: Qt Model View Framework

As depicted in figure 5.4, Qt models have to implement the `QAbstractItemModel` interface allowing views (implementing the `QAbstractItemView` interface) to access the data and presenting it in the form of tables, lists or trees. The view can communicate directly with the model, or with a proxy model (of type `QAbstractProxyModel`) that grants access to the data of the original model and can be extended to provide another view on the data: a standard implementation of a proxy model is provided with `QSortFilterProxyModel`, which allows to sort the data elements and passing only elements according to some filter. The data elements are identified via model indexes (of type `QModelIndex`) and have to be structured in a tree hierarchy where each element has exactly one parent. To use the `MediaObject` and `LogicalEntity` models, which allow their elements to have more than one parent, with this framework, an additional model layer is used – the "View model".

The application model responsible for the management of logical entities, for example, is split up into a *business model* and a *view model* as

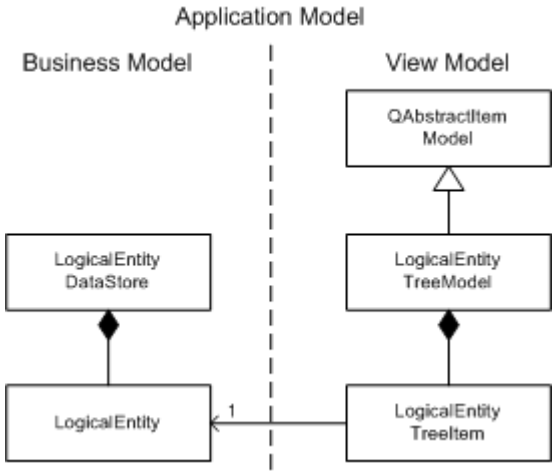


Figure 5.5: Application Model

illustrated in figure 5.5. The business model is represented by an instance of `LogicalEntityDataStore` (cf. section 5.2.4). The view model is realized through the class `LogicalEntityTreeModel`, which performs a transformation of the data hierarchy of logical entities (each having a list of superclasses) into a tree hierarchy of `LogicalEntityTreeItem` elements – each referencing a logical entity and having exactly one parent item. The transformation (of a minimal hierarchy) is illustrated in figure 5.6.

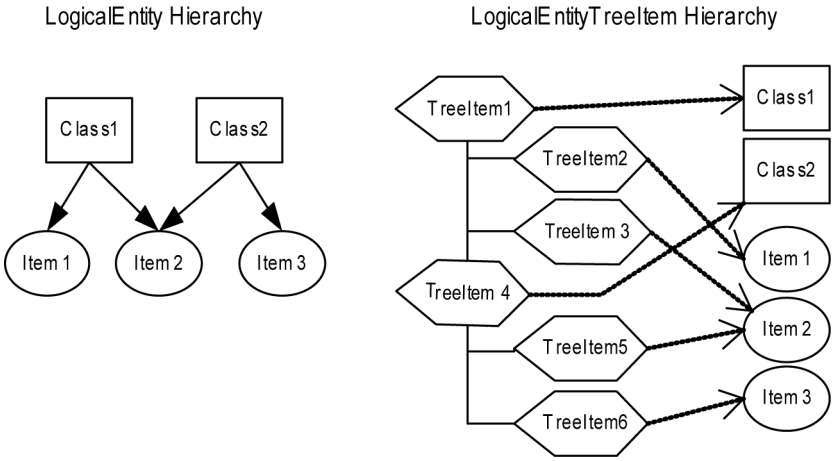


Figure 5.6: Data Transformation

5.3.1 Media Semantics Mapping Utility

The Media Semantics Mapping Utility (MSM Utility) provides the functionality to create logical entities of the F-Space and the C-Space, as well as to define mappings between logical entities.

Figure 5.7 shows a screenshot of the `LogicalEntityEditor` widget, which comprises a class editor (`LEClassEditor`) and an instance editor (`LEInstanceEditor`). The views allow creating, removing and renaming logical entities and thus represent a simple ontology editor. The class editor presents the tree of all logical entity classes (of a certain space). In addition to the names, the number of individuals are shown. When selecting a certain class, the instance editor gets updated and shows a list of the individuals of the selected class.

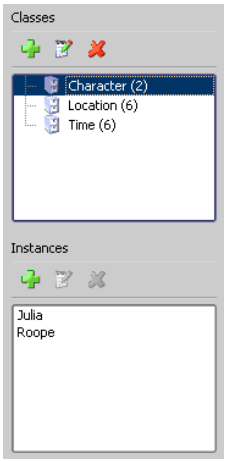


Figure 5.7: Logical Entity Editor

The selection of an individual leads to an update of the definition widget (figure 5.8), which shows on the left side a list of all individuals that define the selected one and, on the right side, a tree of all logical entities. The add and remove buttons provide the functionality to extend the definition list with a logical entity and to take away one definition. As cyclic definitions are not allowed, a message box pops up if the user tries to do so. The simplest cyclic definition would be A defines B and B defines A, but also more complex cycles (if for instance B defines C and C defines A) are avoided. The definition widget thus provides the means to map logical entities to composite ones, the output of which leads to the generation of rules (cf. section 5.4).

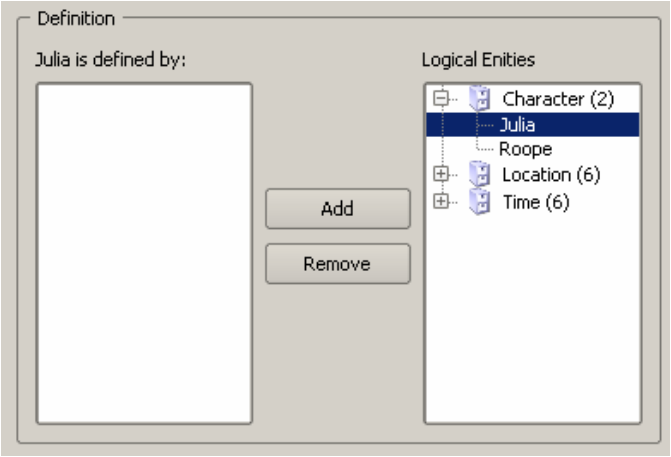


Figure 5.8: Definition Widget

5.3.2 Description Tool

The prototype implementation of the Description Tool provides the functionality to manage and annotate media items. As shown in figure 5.9, it presents a view containing a tree of all media object groups and operations to create new groups, delete existing ones (including the subtree items) and rename them. Media objects (along with some of their properties) that are children of a selected group are shown in a table view that also provides the functionality to add, remove and rename items, as well as to change their properties.

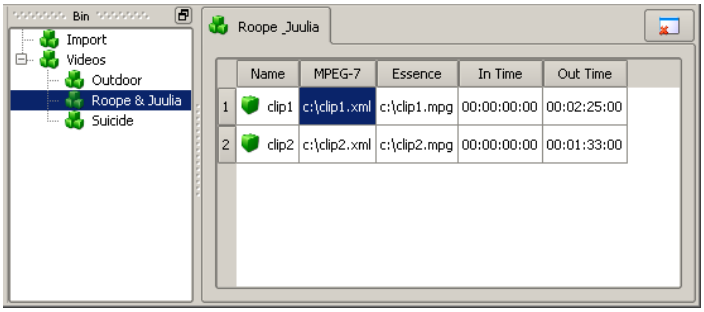


Figure 5.9: Description Tool

The selection of media objects updates the Annotation Widget (figure 5.10), which comprises a list view that contains the logical enitities the selected media objects are annotated with, a view containing all logical entities available and buttons to add and remove annotations. The annotation of a media object group leads to an annotation of all its child elements.

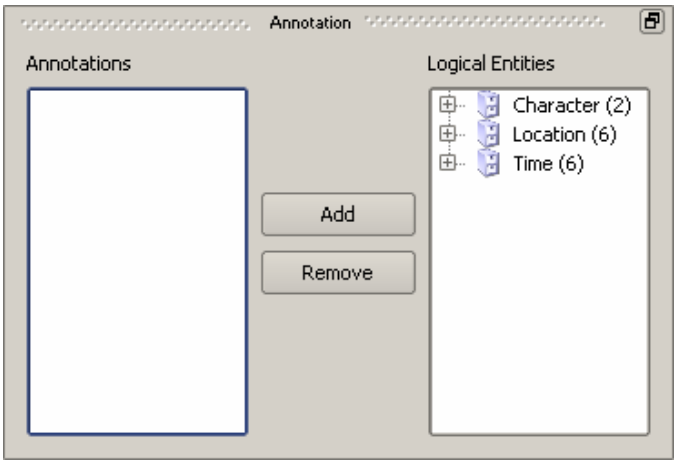


Figure 5.10: Annotation Widget

5.4 Rule Generation

In the prototype implementation, the generation of rules is limited to built-in rules responsible for the automatic annotation of media items with logical entities that have been defined through mappings of other ones. In opposite to the specification given in section 4.5, no SWRL rules are used – as currently no SWRL engine does exist – but executable Prolog code is generated directly: for each (concrete) logical entity that has a non-empty list of mapping definitions a Prolog rule is generated. An example of such a rule in an abstract syntax looks like follows:

$$
\begin{aligned}
&contains(?mi, war) \leftarrow \\
&\quad defines(soldier, war) \wedge defines(gun\_noise, war) \wedge \\
&\quad contains(?mi, soldier) \wedge contains(?mi, gun\_noise) \wedge \\
&\quad CLEntity(soldier) \wedge CLEntity(gun\_noise) \wedge \\
&\quad CLEntity(war) \wedge MediaItem(?mi)
\end{aligned}
$$

Each media item `mi` that is annotated with ("contains") the logical entities *soldier* and *gun_noise* is annotated automatically with the logical entity *war*. A test example of an executable prolog file containing several rules can be found in appendix B.

Chapter 6

Conclusion

The concepts and partial implementation shown in this thesis allow the semi-automatic annotation of media essence with logical entities. It has been illustrated how to use Semantic Web technologies in an application to generate a semantic domain and define logical entities in this domain. The annotation is based on the automatic extraction of low-level features of the essence via MPEG-7. This approach allows an effective and efficient retrieval of the essence based on semantic queries and therefore basically fulfills the requirements of the NM2 project.

The main results of the prototype implementation comprise the RDF/OWL data store, which is based on the Redland API and supports the manipulation of ontology data. Furthermore, components for mapping the ontology data to C++ objects and providing interfaces to access this data have been implemented. The GUI components comprise the Media Semantics Mapping Utility (MSMU), which on the one hand allows the definition of logical entities and therefore represents a simple ontology editor. On the other hand, it provides the functionality to map logical entities to others of the same or a higher level of abstraction. The rules API (based on SWI:Prolog) is responsible for applying rules on the ontology data in order to gain new explicit information in the knowledge base. This has not been fully integrated in the prototype implementation, but is fundamental for the automatic annotation with logical entities of a higher level of abstraction.

A useful and important extension to the implementation (to be done in the future) is a user-friendly way to create low-level features that reference MPEG-7 fragments (e. g. a certain shape description). Currently, they have to be defined manually. Future work has to concentrate on the implementation of a query-by-example (QBE) mechanism to define low-level features[1], so that users of the tools can work on the logical level and don't have to

[1]QBE supports the definition of MPEG-7 features through reference images or audio clips. By supplying, for example, the image of a soccer ball it should be possible to generate MPEG-7 descriptions of the colour of the ball, its shape, texture, etc.

have any knowledge about MPEG-7.

Another important issue that hasn't been mentioned in this work so far but demands some additional functionality, is about the spatio-temporal aspect of annotations: In the current implementation, the automatic annotation of media items with a certain concrete logical entity (CLE) is based solely on the existence of low-level features in the essence that define that CLE. It is not taken into consideration, if the low-level features occur at the same position and at the same time. This problem could be solved by extracting complex objects via MPEG-7, which is generally possible but would i) demand a non-trivial mechanism to define those objects via the MPEG-7 DDL and ii) reduce the probability to find those (complex) objects in the essence. The temporal aspect could be handled via an extension of the core ontology, so that each annotation also comprises a begin and an end time – leading to a "partial annotation" of media items.

The annotation of media items takes place on the instance level of the ontology, which is a reasonable solution for the recognition of objects that occur in more than one video clip and always have the same appearance. As example serves once again the *soccer ball* instance already mentioned before in this thesis. However, the annotation with a class of a logical entity could sometimes be preferred: for instance, to annotate interview clips, it obviously does not make sense to create an instance for each interview. Rather, the presence of a certain person (i. e. a reporter) and two voices could be recognized as features that characterize an interview. Through the usage of rules, such annotations (on the class level) could be attached to media items. The present design and implementation of the NM2 tools do not support that functionality, future work could enclose this feature.

At last, the general handling of rules has to be discussed. The design of the system plans to generate a SWRL representation of the rules, which forms part of the knowledge base. The current implementation only supports the direct creation of executable Prolog code whose execution leads to an update of the knowledge base and supports the automatic annotation of media items with concrete logical entities. This workaround over Prolog is necessary, due to the absence of an existing SWRL compliant rule engine. As soon as such an engine is available the system should be changed to use this rule engine instead of Prolog.

Reusable parts of the implementation are represented by the RDF/OWL data store for the creation and manipulation of ontology data, the ontology editor to define classes and instances of an ontology, and the rules API to infer new assertions to a knowledge base. These components are not restricted to NM2, but can be used in any application that deals with ontology data.

Appendix A

NM2 Core Ontology

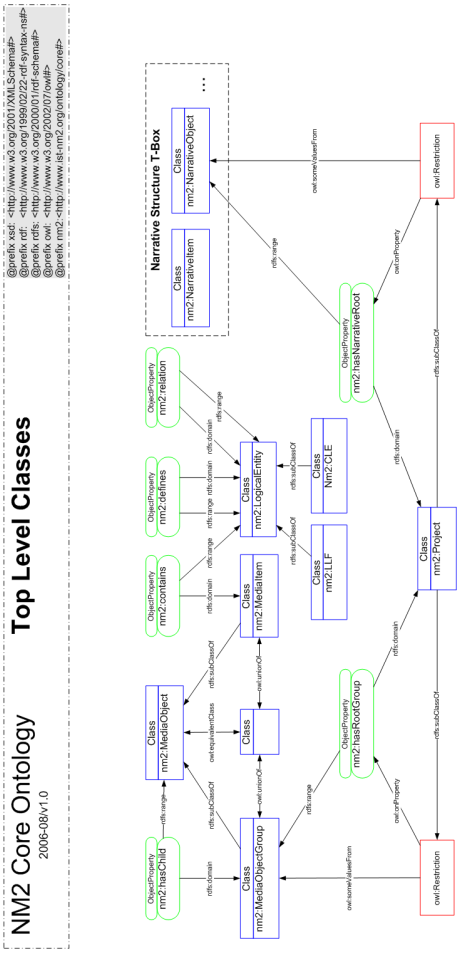


Figure A.1: NM2 Core

Appendix B

SWI-Prolog Code Example

```
classify :-
  assert(library_directory('c://programs//pl//library//semweb//')),
  use_module(library(rdf_db)),
  use_module(library(rdfs)),
  rdf_load('nm2_dummy_prod.owl'),
  rdf_register_ns(dmy, 'http://www.ist-nm2.org/productions/dummy#'),
  contains_proxy12, contains_proxy23 .

contains_proxy12 :- contains12(MI, LE), rdf_assert(MI,
  'http://www.ist-nm2.org/productions/dummy#contains', LE).

contains_proxy23 :- contains23(MI, LE), rdf_assert(MI,
  'http://www.ist-nm2.org/productions/dummy#contains', LE).

contains12(MI, 'http://www.ist-nm2.org/productions/dummy#L2_0') :-
  rdf(MI, 'http://www.ist-nm2.org/productions/dummy#contains',
    'http://www.ist-nm2.org/productions/dummy#L1_0'),
  rdf(MI, 'http://www.ist-nm2.org/productions/dummy#contains',
    'http://www.ist-nm2.org/productions/dummy#L1_1').

contains12(MI, 'http://www.ist-nm2.org/productions/dummy#L2_1') :-
  rdf(MI, 'http://www.ist-nm2.org/productions/dummy#contains',
    'http://www.ist-nm2.org/productions/dummy#L1_2'),
  rdf(MI, 'http://www.ist-nm2.org/productions/dummy#contains',
    'http://www.ist-nm2.org/productions/dummy#L1_3').

contains12(MI, 'http://www.ist-nm2.org/productions/dummy#L2_2') :-
  rdf(MI, 'http://www.ist-nm2.org/productions/dummy#contains',
    'http://www.ist-nm2.org/productions/dummy#L1_4'),
  rdf(MI, 'http://www.ist-nm2.org/productions/dummy#contains',
    'http://www.ist-nm2.org/productions/dummy#L1_5'),
  rdf(MI, 'http://www.ist-nm2.org/productions/dummy#contains',
    'http://www.ist-nm2.org/productions/dummy#L1_6').
```

```
contains12(MI, 'http://www.ist-nm2.org/productions/dummy#L2_3') :-
  rdf(MI, 'http://www.ist-nm2.org/productions/dummy#contains',
    'http://www.ist-nm2.org/productions/dummy#L1_7'),
  rdf(MI, 'http://www.ist-nm2.org/productions/dummy#contains',
    'http://www.ist-nm2.org/productions/dummy#L1_8').

contains12(MI, 'http://www.ist-nm2.org/productions/dummy#L2_4') :-
  rdf(MI, 'http://www.ist-nm2.org/productions/dummy#contains',
    'http://www.ist-nm2.org/productions/dummy#L1_9').

contains23(MI, 'http://www.ist-nm2.org/productions/dummy#L3_0') :-
  rdf(MI, 'http://www.ist-nm2.org/productions/dummy#contains',
    'http://www.ist-nm2.org/productions/dummy#L2_0'),
  rdf(MI, 'http://www.ist-nm2.org/productions/dummy#contains',
    'http://www.ist-nm2.org/productions/dummy#L2_1'),
  rdf(MI, 'http://www.ist-nm2.org/productions/dummy#contains',
    'http://www.ist-nm2.org/productions/dummy#L2_2').

contains23(MI, 'http://www.ist-nm2.org/productions/dummy#L3_1') :-
  rdf(MI, 'http://www.ist-nm2.org/productions/dummy#contains',
    'http://www.ist-nm2.org/productions/dummy#L2_3'),
  rdf(MI, 'http://www.ist-nm2.org/productions/dummy#contains',
    'http://www.ist-nm2.org/productions/dummy#L2_4').
```

Bibliography

[AAB03] Joan Aliprand, Julie Allen, and Joe Becker. *The Unicode Standard, Version 4.0: The Unicode Consortium with CDROM.* Addison Wesley, 2003.

[ADR] Naming and Adressing – W3C. http://www.w3.org/Addressing/.

[AN95] Agnar Aamodt and Mads Nygard. Different Roles and Mutual Dependencies of Data, Information, and Knowledge - An AI Perspective on their Integration. *Data Knowledge Engineering*, 16(3):191–222, 1995.

[ATP+05] Th. Athanasiadis, V. Tzouvaras, K. Petridis, F. Precioso, Y. Avrithis, and Y. Kompatsiaris. Using a Multimedia Ontology Infrastructure for Semantic Annotation of Multimedia Content. In *5th International Workshop on Knowledge Markup and Semantic Annotation (SemAnnot'05)*, Galway, Ireland, 2005.

[BCM+03] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[BCM04] Gene Bellinger, Durval Castro, and Anthony Mills. Data, Information, Knowledge, and Wisdom. http://www.systems-thinking.org/dikw/dikw.htm, 2004.

[Bec01] D. Beckett. The design and implementation of the Redland RDF application framework. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 449–456, New York, NY, USA, 2001.

[BK02] S. Baumann and A. Kluter. Super-convenience for non-musicians: Querying MP3 and the semantic web. In *Proceedings of the Third International Conference on Music*

Information Retrieval: ISMIR 2002, pages 297–298, Paris, France, 2002. IRCAM - Centre Pompidou.

[BNUA04] E. Batlle, H. Neuschmied, P. Uray, and G. Ackermann. Recognition and analysis of audio for copyright protection: The RAA project. *JASIST*, 55(12):1084–1091, 2004.

[BS06] W. Bailer and P. Schallauer. The Detailed Audiovisual Profile: Enabling Interoperability between MPEG-7 based Systems. In *12th International MultiMedia Modelling Conference (MMM'06)*, pages 217–224, Beijing, China, 2006.

[BSHT05] W. Bailer, P. Schallauer, M. Hausenblas, and G. Thallinger. MPEG-7 Based Description Infrastructure for an Audiovisual Content Analysis and Retrieval System. In *Proceedings of SPIE - Storage and Retrieval Methods and Applications for Multimedia*, volume 5682, pages 284–295, 2005.

[CdCC+05] B. Cardoso, F. de Carvalho, L. Carvalho, G. Fernàndez, P. Gouveia, B. Huet, J. Jiten, A.López, B. Merialdo, A. Navarro, H. Neuschmied, M. Noé, R. Salgado, and G. Thallinger. Hyperlinked video with moving objects in digital television. In *ICME 2005, IEEE International Conference on Multimedia & Expo*, Amsterdam, The Netherlands, 2005.

[GC05] R. Garcia and O. Celma. Semantic Integration and Retrieval of Multimedia Metadata. In *5th International Workshop on Knowledge Markup and Semantic Annotation (SemAnnot'05)*, Galway, Ireland, 2005.

[GHJV95] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional, 1995.

[GHVD03] Benjamin Grosof, Ian Horrocks, Raphael Volz, and Stefan Decker. Description Logic Programs: Combining Logic Programs with Description Logics. In *Proc. of WWW-2003*, Budapest, Hungary, 2003.

[Gro94] W. I. Grosky. Multimedia Information Systems. *IEEE MultiMedia*, 1(1):12–24, 1994.

[HLH05] L. Hollink, S. Little, and J. Hunter. Evaluating the application of semantic inferencing rules to image

annotation. In *3rd International Conference on Knowledge Capture (K-CAP 2005)*, pages 91–98, Banff, Alberta, Canada, 2005. ACM.

[HPSB+04] Ian Horrocks, Peter F. Patel-Schneider, Harold Boley, Said Tabet, Benjamin Grosof, and Mike Dean. SWRL: A Semantic Web Rule Language Combining OWL and RuleML. Member Submission, World Wide Web Consortium, May 2004. Available at http://www.w3.org/Submission/SWRL/.

[HPSBT05] I. Horrocks, P. F. Patel-Schneider, S. Bechhofer, and D. Tsarkov. OWL Rules: A Proposal and Prototype Implementation. *Journal of Web Semantics*, 3(1):23–40, 2005.

[HR03] M. Hatala and G. Richards. Value-Added Metatagging: Ontology and Rule Based Methods for Smarter Metadata. In *Rules and Rule Markup Languages for the Semantic Web, 2nd International Workshop (RuleML 2003)*, volume 2876 of *Lecture Notes in Computer Science*, pages 65–80, Sanibel Island, Florida, USA, 2003.

[HR04] D. Harel and B. Rumpe. Meaningful Modeling: What's the Semantics of "Semantics"? *Computer*, 37(10):64–72, 2004.

[Hun01] J. Hunter. Adding Multimedia to the Semantic Web: Building an MPEG-7 ontology. In *First International Semantic Web Working Symposium (SWWS'01)*, Stanford, California, USA, 2001.

[KMS04] H. Kim, N. Moreau, and T. Sikora. Audio classification based on MPEG-7 spectral basis representations. *IEEE Transactions on Circuits and Systems for Video Technology 7, Special Issue on Audio and Video Analysis for Multimedia Interactive Services*, 14(5):716–725, 2004.

[LH04] S. Little and J. Hunter. Rules-By-Example - A Novel Approach to Semantic Indexing and Querying of Images. In *3rd International Semantic Web Conference (ISWC'04)*, volume 3298 of *Lecture Notes in Computer Science*, pages 534–548, Hiroshima, Japan, 2004.

[Llo87] J. W. Lloyd. *Foundations of logic programming; (2nd extended ed.)*. Springer-Verlag New York, Inc., New York, NY, USA, 1987.

[Mal05] Jan MaluszynskiLittle:ISWC2004. Combining rules and ontologies. a survey. Research report IST506779/Linköping/I3-D3/D/PU/b3, Linköping University, 2005. REWERSE Deliverable.

[Mar04] J. M. Martinez. MPEG-7 Overview (version 10); ISO/IEC JTC1/SC29/WG11N6828. http: //www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm, 2004.

[Mar05] J. M. Martinez. Introducing MPEG-7 MDS – an Overview (version 1) ISO/IEC JTC1/SC29/WG11N7418. http://www. chiariglione.org/mpeg/technologies/mp07-mds/index.htm, 2005.

[MB03] O. Marques and N. Barman. Semi-automatic Semantic Annotation of Images Using Machine Learning Techniques. In *2nd International Semantic Web Conference (ISWC'03)*, volume 2870 of *Lecture Notes in Computer Science*, pages 550–565, Sanibel Island, Florida, USA, 2003.

[MET02] Definition of types of metadata. digital library committee. http://staffweb.library.northwestern.edu/dl/metadata/ standardsinventory/definition.html, 2002.

[MPE01] MPEG-7. Multimedia Content Description Interface. Standard No. ISO/IEC n°15938, 2001.

[MSS05] Boris Motik, Ulrike Sattler, and Rudi Studer. Query answering for OWL-DL with rules. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, 3(1):41–60, 2005.

[NM2a] Nm2 Brochure. http://www.ist-nm2.org/publications/NM2_Brochure-2005.pdf.

[NM2b] Nm2 - New Media for a New Millennium. http://www.ist-nm2.org.

[NM2c] Nm2 Partners. http://www.ist-nm2.org/about/partners.html.

[NMV+05] Y. Naudet, D. Mathevon, A. Vagner, S. Renault, and J.-S. Brunner. MPEG-7 and the Semantic Web: An Hybrid Approach for Semantic and Behavioral Descriptions in Interactive Object-Oriented Multimedia Applications. *axmedis*, 0:213–216, 2005.

[PJ01] M. Petkovic and W. Jonker. Content-Based Video Retrieval by Integrating Spatio-Temporal and Stochastic Recognition of Events. In *IEEE Workshop on Detection and Recognition of Events in Video*, pages 75–82, 2001.

[PvSMK+05] Luís Ferreira Pires, Marten van Sinderen, Ellen Munthe-Kaas, Stanislav Pokraev, Mathieu Hutschemaekers, and Dirk-Jaap Plas. Techniques for describing and manipulating context information. Technical report, Lucent Technologies, October 2005. Available at https://doc.freeband.nl/dscgi/ds.py/Get/File-62334.

[QT] Qt - Cross-Platform C++ Library. http://www.trolltech.com/products/qt.

[QTM] Qt - Model/View Programming. http://doc.trolltech.com/4.1/model-view-programming.html.

[RDF04a] Resource Description Framework (RDF) Concepts and Syntax Specification. http://www.w3.org/TR/rdf-concepts, 2004.

[RDF04b] Resource Description Framework (RDF) Schema Specification 1.0. http://www.w3.org/TR/rdf-schema, 2004.

[RH94] Kotagiri Ramamohanarao and James Harland. An introduction to deductive database languages and systems. *The VLDB Journal*, 3(2):107–122, 1994.

[RUL] Rulesystem Arrangement Framework – W3C. http://www.w3.org/2005/rules/wg/wiki/Rulesystem_Arrangement_Framework.

[SPA04] SPARQL Query Language for RDF – W3C. http://www.w3.org/TR/rdf-sparql-query/, 2004.

[SWA01] Semantic Web Activity Statement – W3C. http://www.w3.org/2001/sw/Activity, 2001.

[SWM04] Michael K. Smith, Chris Welty, and Deborah L. McGuinness. OWL Web Ontology Language Guide. W3C recommendation, World Wide Web Consortium, February 2004. Available at http://www.w3.org/TR/owl-guide.

[TPC04] C. Tsinaraki, P. Polydoros, and S. Christodoulakis. Interoperability support for Ontology-based Video Retrieval Applications. In *3rd International Conference on Image and Video Retrieval (CIVR'04)*, Dublin, Ireland, 2004.

[Tro03] R. Troncy. Integrating Structure and Semantics into Audio-visual Documents. In *2nd International Semantic Web Conference (ISWC'03)*, volume 2870 of *Lecture Notes in Computer Science*, pages 566–581, Sanibel Island, Florida, USA, 2003.

[URI] Uniform Resource Identifiers (URI): Generic Syntax – RFC. http://www.ietf.org/rfc/rfc2396.txt.

[VKSB06] S. Vembu, M. Kiesel, M. Sintek, and S. Baumann. Towards bridging the semantic gap in multimedia annotation and retrieval. Proc. of the 1st International Workshop on Semantic Web Annotations for Multimedia, SWAMM 2006 at the 15th International World Wide Web Conference, WWW 2006, 2006.

[vNH04] J. van Ossenbruggen, F. Nack, and L. Hardman. That Obscure Object of Desire: Multimedia Metadata on the Web (Part I). *IEEE Multimedia*, 11(4), 2004.

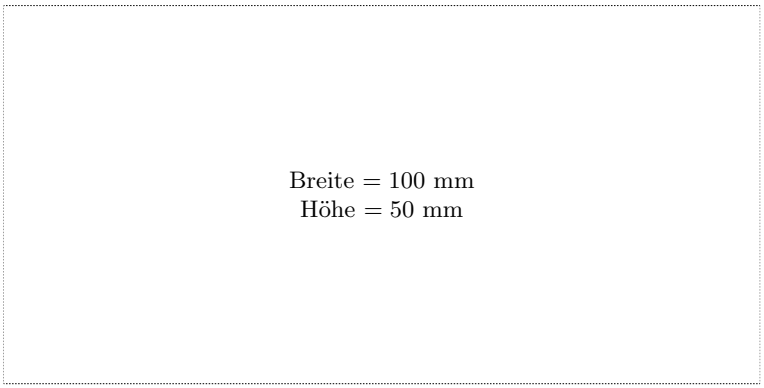
[vNH05] J. van Ossenbruggen, F. Nack, and L. Hardman. That Obscure Object of Desire: Multimedia Metadata on the Web (Part II). *IEEE Multimedia*, 12(1), 2005.

[XML] XML Specification – W3C. http://www.w3.org/TR/REC-xml.

[XSD] XML Schema Specification – W3C. http://www.w3.org/XML/Schema#dev.

Messbox zur Druckkontrolle

— Druckgröße kontrollieren! —



— Diese Seite nach dem Druck entfernen! —