Martin Höffernig, BSc

Formalizing Semantic Constraints of MPEG-7 Profiles

MASTER'S THESIS

to achieve the university degree of

Diplom-Ingenieur

Master's degree programme: Telematics

submitted to

Graz University of Technology

Supervisor

Univ.-Prof. Dipl.-Inf. Dr. Stefanie Lindstaedt
Knowledge Technologies Institute
Graz University of Technology

Advisor: Dipl.-Ing.(FH) Werner Bailer
JOANNEUM RESEARCH Forschungsgesellschaft mbH
DIGITAL - Institute for Information and Communication Technologies

Graz, March 2015

AFFIDAVIT

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master‘s thesis dissertation.

Date

Signature

Abstract

The amount of multimedia content being created is growing tremendously. In addition, the number of applications for processing, consuming, and sharing multimedia content is growing. Being able to create and process metadata describing this content is an important prerequisite to ensure a correct workflow of applications. The MPEG-7 standard enables the description of different types of multimedia content by creating standardized metadata descriptions.

When using MPEG-7 practically, two major drawbacks are identified, namely complexity and fuzziness. Complexity is mainly based on the comprehensiveness of MPEG-7, while fuzziness is a result of the syntax variability. The notion of MPEG-7 profiles were introduced in order to address and possibly solve these issues. A profile defines the usage and semantics of MPEG-7 tailored to a particular application domain. Thus usage instructions and explanations, denoted as semantic constraints, can be expressed as English prose. However, this textual explanations leave space for potential misinterpretations since they have no formal grounding. While checking the conformance of an MPEG-7 profile description is possible on a syntactical level, the semantic constraints currently cannot be checked in an automated way. Being unable to handle the semantic constraints, inconsistent MPEG-7 profile descriptions can be created or processed leading to potential interoperability issues.

Thus an approach for formalizing the semantic constraints of MPEG-7 profiles using ontologies and logical rules is presented in this thesis. Ontologies are used to model the characteristics of the different profiles with respect to the semantic constraints, while validation rules detect and flag violations of these constraints. In similar manner, profile-independent temporal semantic constraints are also formalized. The presented approach is the basis for a semantic validation service for MPEG-7 profile descriptions, called VAMP. VAMP verifies the conformance of a given MPEG-7 profile description with a selected MPEG-7 profile specification in terms of syntax and semantics. Three different profiles are integrated in VAMP. The temporal semantic constraints are also considered. As a proof of concept, VAMP is implemented as a web application for human users and as a RESTful web service for software agents.

Kurzfassung

Die Anzahl von neu erzeugten Multimedia-Inhalten steigt enorm an. Zusätzlich gibt es immer mehr Anwendungen, die diese Inhalte aufbereiten, verarbeiten und austauschen. Die Unterstützung von Metadaten ist eine wichtige Voraussetzung, um einen korrekten Ablauf der Anwendungen sicherzustellen. Der MPEG-7 Standard ermöglicht die Erstellung von standardisierten Metadaten für die Beschreibung von verschiedensten Multimedia-Inhalten (MPEG-7 Beschreibung). Beim praktischen Einsatz von MPEG-7 können zwei wesentliche Nachteile auftreten, nämlich Komplexität und Mehrdeutigkeiten. Die Komplexität ist hauptsächlich eine Ursache des umfangreichen Beschreibungsumfanges von MPEG-7, während Mehrdeutigkeiten die Resultate von möglichen Syntax Variationen bei der Beschreibung der Metadaten sind.

MPEG-7 wurden Profile spezifiziert, um diese Nachteile zu beseitigen. Ein Profil definiert die genaue Verwendung und Semantik des MPEG-7 Standards, zugeschnitten auf einen speziellen Anwendungsbereich. Dabei können Anweisungen und Erklärungen, die zur richtigen Verwendung des Profiles beitragen, als englischer Text erstellt werden. Diese Instruktionen werden als Semantic Constraints bezeichnet. Leider lassen die Semantic Constraints Fehlinterpretationen zu, da sie nicht auf einer formalen Beschreibungssprache aufbauen.

Zwar ist die Überprüfung des Syntaxes von MPEG-7 Profil-Beschreibungen möglich, die Konformität zu den Semantic Constraints kann aber mit jetzigen Mitteln nicht automatisiert überprüft werden. Werden die Semantic Constraints nicht berücksichtigt, können widersprüchliche Profil-Beschreibungen erstellt und verbreitet werden, die zu Kompatibilitätsproblemen führen können. Diese Arbeit stellt einen neuen Ansatz zur Formalisierung der Semantic Constraints von MPEG-7 Profilen vor, welcher Ontologien und logische Regeln einsetzt. Ontologien werden verwendet um die Eigenschaften der MPEG-7 Profile in Bezug auf die Semantic Constraints zu modellieren, während Regeln Verletzungen dieser Constraints in Profil-Beschreibungen erkennen und kennzeichnen. Auf ähnliche Art und Weise werden auch zeitliche Semantic Constraints, die unabhängig von Profilen definiert sind, formalisiert. Der vorgestellte Ansatz ist die Grundlage für ein Validierungsservice von MPEG-7 Profil-Beschreibungen, abgekürzt als VAMP bezeichnet. VAMP überprüft eine Profil-Beschreibung auf Einhaltung der jeweiligen Profilspezifikation in Bezug auf Syntax und Semantik. Dabei können drei verschiedene Profile und die Profil-unabhängigen zeitlichen Semantic Constraints berücksichtigt werden. Als Machbarkeitsstudie ist VAMP als Web Anwendung und als RESTful Webservice implementiert.

Contents

List of Figures

List of Tables

List of Code Listings

Acknowledgements

This master's thesis was performed at DIGITAL - Institute for Information and Communication Technologies of the JOANNEUM RESEARCH Forschungsgesellschaft mbH. This thesis was supervised by Univ.-Prof. Dipl.-Inf. Dr. Stefanie Lindstaedt, the head of the Knowledge Technologies Institute at TU Graz, while the advisors at JOANNEUM RESEARCH were Dipl.-Ing.(FH) Werner Bailer and Dipl.-Ing. Dr. Michael Hausenblas (now MapR Technologies, Ireland). I hereby wish to express my deepest gratitude to Dr. Lindstaedt for supporting the finalization of this thesis and to my colleague Werner Bailer for his continuing support, constructive criticism, formulation skills, and countless hours of proofreading. Without him, this thesis would not have been successfully completed.

Additionally, I would like to thank Dipl.-Ing. Georg Thallinger for enabling the realization of my thesis at JOANNEUM RESEARCH and all my colleagues for their contributions. Special mention goes to Günter Nagler (JOANNEUM RESEARCH) for formalizing the semantic constraints of the AudioVisual Description Profile and implementing the VAMP RESTful service and to Alia Amin (Centrum Wiskunde & Informatica, Amsterdam) for the initial design of the VAMP web interface.

The research leading to this master's thesis was partially supported by the European Commission under the contracts FP7-231161, "PrestoPRIME", FP6-027026, "Knowledge Space of semantic inference for automatic annotation and retrieval of multimedia content - K-Space", ICT-2007-214793, "TA2 - Together anywhere, together anytime", and IST-2-511316, "IP-RACINE: Integrated Project - Research Area CINE".

Finally, endless thanks goes to my family, especially to my mother Ingeborg and my girlfriend Martina. This thesis is dedicated to my beautiful daughters Paula and Greta and respectfully to my late father Horst.

Martin Höffernig
Graz, March 2015

Chapter 1

Introduction

In this Chapter, the motivation for the work that is presented by this thesis is explained first. Then the objective is stated, followed by a brief summary of the proposed solution and the achieved results. Finally, an overview of the structure of this thesis is given.

1.1 Motivation

The amount of multimedia content being created is growing tremendously. For example, YouTube claims that 100 hours of video are uploaded to its platform every minute[1]. The number of applications that are processing, consuming, and sharing multimedia content, such as videos and images, is also growing. For many use cases, the availability of metadata describing this content is an important prerequisite in order to ensure a correct workflow of these applications. In a broad range of application areas, *MPEG-7* [15], formally named *Multimedia Content Description Interface*, is used for representing the metadata of multimedia content.

MPEG-7 enables the description of different types of multimedia content. For example, an audio file, a video clip, or a still image can be described. Thus the description of different kinds of aspects and features of this content is supported. Among others, information about creation, usage, rights, and technical features can be described. In addition, MPEG-7 provides extensive ways for describing structural information, such as representing the segmentation of the content in time and space.

In order to describe metadata in terms of MPEG-7, so-called *description tools* are defined. These description tools are specified by the Description Definition Language (DDL), which is an extension of XML Schema Language [39]. As a result, an own MPEG-7 XML Schema was defined. By selecting and instantiating particular MPEG-7 description tools an MPEG-7 description about multimedia content is created. Therefore any MPEG-7 description is created with respect to this MPEG-7 XML Schema. Such an MPEG-7 description is either represented in textual form encoded in XML or in

[1] `https://www.youtube.com/yt/press/statistics.html`

a binary format. While the textual representation is suitable for editing and searching, the binary representation is more suitable for storage and transmission tasks.

MPEG-7 allows flexible use of the description tools. Thus these tools enable descriptions about the whole content itself or specific portions only. In addition, the level of detail and granularity of these descriptions can be adapted according to the needs of a specific use case. However, this flexibility leads to increased complexity and fuzziness [69, 21, 78]. The complexity is mainly based on the high number of the available description tools, while fuzziness is a result of the syntax variability and a lack of formal semantics. For example, two syntactically different MPEG-7 descriptions can cover the same intended semantics. Furthermore, a description tool can be used for describing multiple semantically different concepts. The reason is that the semantics and use of the description tools are defined in a rather general way not to exclude possible application scenarios.

In some cases, the usage of the description tools is clearly formulated not leaving any space for misinterpretations. These explanations are expressed as textual guidelines in the MPEG-7 standard. For example, the correct use of the description tools for describing a temporal segmentation in terms of MPEG-7 are stated. In this context, the semantics of attributes addressing some specific characteristics of such a segmentation is explained.

Serious interoperability issues arise when exchanging multimedia content between applications that are interpreting the corresponding MPEG-7 descriptions differently. In order to reduce the complexity, syntax variability, and semantic ambiguities of MPEG-7 descriptions, the notion of MPEG-7 *profiles* were introduced [42]. A profile explicitly defines the usage and semantics of the description tools tailored to a particular application domain. Then neither syntax variability nor space for misinterpretations should be possible for MPEG-7 descriptions conforming to an MPEG-7 profile. As a result, the interoperability of MPEG-7 descriptions and the described multimedia content is getting enhanced.

A profile is a subset of the whole MPEG-7 standard. Thus a profile XML Schema includes the description tools needed for covering the intended application domain only. Additionally, the semantics and usage of the profile in context of the intended application domain can be expressed. For example, a profile expresses how the description tools are used and combined and how the elements and attributes in a resulting MPEG-7 profile description have to be interpreted. These instructions and explanations are denoted as profile semantic constraints. Most of the semantic constraints that are analyzed in this thesis are related to the structural description of multimedia content. For instance, some profile semantic constraints clarify the representation of the results of an automated shot boundary detection for a video. The goal of this detection is to split a video into a set of temporal sequences. A temporal sequence represents a continuous action recorded by a single camera and is denoted as shot. A shot boundary detection is used, among others, for editing, archiving, or quality assessment tasks of videos.

The semantic constraints of MPEG-7 profiles are expressed as prose text written in

English. They are usually part of the profile documentation and complement the related XML Schema (MPEG-7 profile XML Schema). Expressing semantic constraints in prose text leaves space for potential misinterpretation and ambiguities. Moreover, in some cases the semantic constraints are expressed imprecisely or their meaning is hidden between the lines. When checking the conformance of an MPEG-7 description with respect to an MPEG-7 profile, syntactical restrictions are verified against the underlying XML Schema. Due to the rather informal specification of the semantic constraints in prose text, they cannot be validated automatically in a machine-processable way. In case of ignoring or simply being unable to handle these constraints, the description tools can be used in an incompatible way. As a result, inconsistent MPEG-7 profile descriptions are created or processed leading to interoperability issues. The lack of formal semantics of the semantic constraints limits an effective use of MPEG-7 profiles for describing multimedia content [78].

1.2 Objective

The goal of the presented work is to explicitly formalize the semantic constraints of MPEG-7 profiles in a machine-processable way. Formalizing these constraints reduces possible misinterpretations of the profiles. As a consequence, the interoperability of related MPEG-7 profile descriptions is enhanced. This is a desired precondition in several use cases, such as creating, modifying, and exchanging multimedia content that is created on the basis of corresponding MPEG-7 profile descriptions.

By applying the formalized semantic constraints of MPEG-7 profiles, an automated semantic validation service for MPEG-7 profile descriptions is established. Then the conformance of an MPEG-7 profile description to a given MPEG-7 profile specification is verified and possible inconsistencies are reported.

1.3 Proposed Solution and Results

The proposed solution for formalizing the semantic constraints of an MPEG-7 profile is based on Semantic Web technologies, in particular ontologies and logical rules. In contrast to other works [17, 45, 59, 80], the MPEG-7 description tools are not completely mapped onto an ontology. Instead, a profile ontology models the structure and semantics of an MPEG-7 profile After creating such a profile ontology, validation rules are defined based on this ontology and the textual description of the semantic constraints. The purpose of these rules is to detect and flag violations of the semantic constraints in a given MPEG-7 profile description.

For example, when referring to the representation of the results of a shot detection analysis, one related profile semantic constraint is that a video can only decomposed further into segments identified as shots. Therefore among others, the concepts of a video and shot are defined as classes of a profile ontology. In addition, properties are

defined in order to describe relations between these classes, such as part-of relationships. Then property restrictions on classes are stated in order to limit the relations between instances of specific classes. In case of a video and corresponding shots, a property restriction is used to limit the temporal segmentation of a video into shots only. Based on this ontology definition, a validation rule is created. This logical rule detects and flags segments that are not identified as shots, but part of the segmentation of the video.

In addition, temporal semantic constraints are formalized. These constraints are regarded as being profile independent since they are defined by the MPEG-7 standard itself and not by a specific MPEG-7 profile definition. However, in order to ensure full interoperability of MPEG-7 profile descriptions, these constraints have to be considered also. The related formalization approach is similar to the one for strictly profile-related semantic constraints. Additionally, a specific ontology for representing temporal segments is used to classify validation results. For example, applying this approach is used to detect temporal inconsistencies in an MPEG-7 description that is representing a shot detection result. Then a shot having an inconsistent temporal extent with respect to the video being included is detected an flagged.

Based on this solution for formalizing semantic constraints, a semantic validation service for MPEG-7 profile descriptions, VAMP for short, was developed. In Figure 1.1, the interaction of an actor (denoted as validator) and VAMP for validating an MPEG-7 profile description is represented as a UML[2] use case diagram. First, the validator provides the MPEG-7 profile description to be validated and additional validation parameters, such as profile and validation type. Then the validation process is initialized. After this process is passed, a meaningful validation report is returned to the validator.

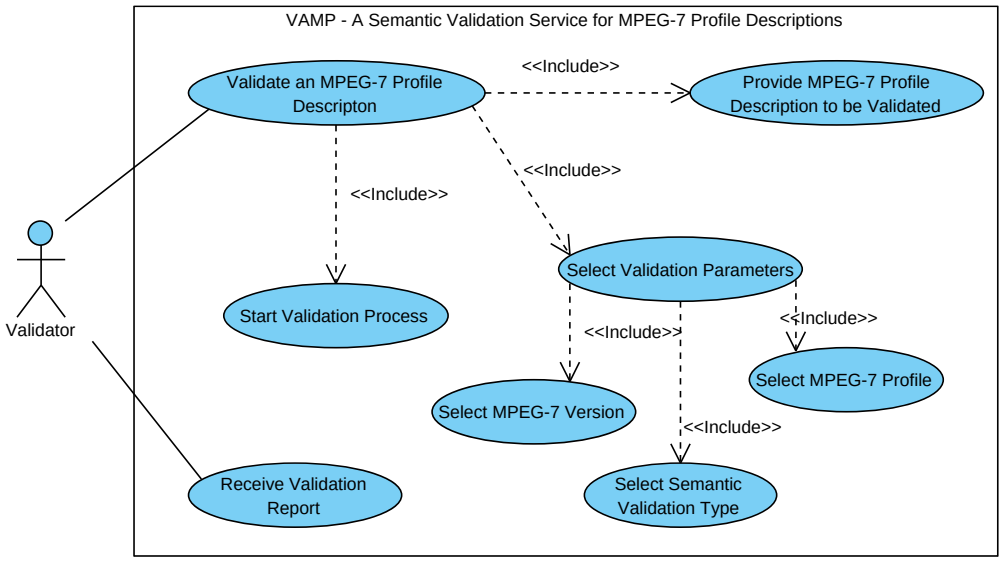

Figure 1.1: UML use case diagram for the validation of an MPEG-7 profile description.

[2]Unified Modeling Language, http://www.uml.org

The validation workflow of VAMP is represented as UML activity diagram, which is depicted in Figure 1.2. First, the MPEG-7 profile description to be validated is syntactically validated against both, the MPEG-7 XML Schema and the selected MPEG-7 profile XML Schema. A syntactically valid and well-formed MPEG-7 profile description is a necessary precondition for validating the semantic constraints. Second, instances of the profile ontology are created with respect to the MPEG-7 profile description to be validated. In this conversion step, an XML transformation defined by the Extensible Stylesheet Language (XSL) [25] and additional conversion rules are applied. In the semantic constraints check, the profile validation rules, which are based on the profile ontology, are applied. The temporal semantic constraints can also be checked. Therefore a mapping from the profile ontology to the temporal segments ontology is needed first. For this purpose, a SPARQL [49] construct query is used to map instances from a profile ontology to the temporal segments ontology. All possible validation violations are flagged by profile validation rules, while temporal violations are classified by the use of an ontology reasoner after performing temporal validation rules [54]. Finally, the flagged violations are summarized and reported using a SPARQL select query. In this validation workflow, a profile ontology is used as an indirect input only, serving as the basis for the ontological representation of the MPEG-7 profile description and the conversion and validation instructions.

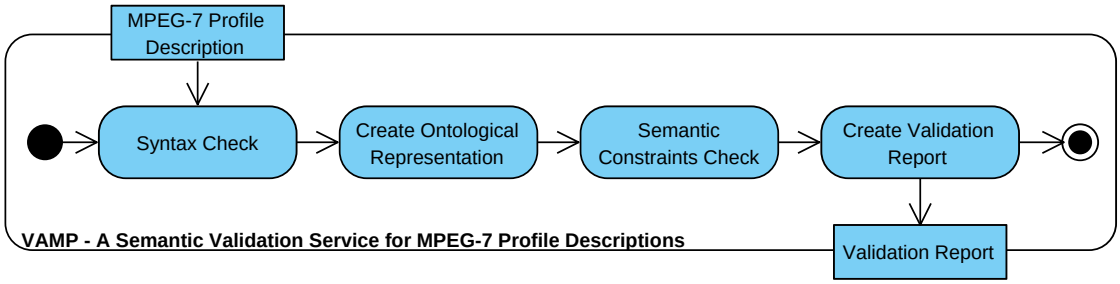

Figure 1.2: UML activity diagram representing the validation workflow of VAMP.

As a proof of concept and for demonstration purpose, VAMP is implemented as a web application for humans and a RESTful web service for agents. When using the VAMP web interface[3], the URI of the MPEG-7 profile description to be validated is provided first. The next step is to select the MPEG-7 profile, to which the input MPEG-7 description document should conform to. Then the semantic validation type is selected. After the validation process is finished, a meaningful report of all possible detected errors is provided. Therefore for each semantic error, the XML elements causing this error are listed. These XML elements are identified by XPath expressions which enable the direct observation of the error locations in the input MPEG-7 profile description.

This thesis is the continuation of the work presented in [78], which describes the initial formalization approach for a subset of the semantic constraints of the Detailed

[3] http://vamp.joanneum.at

Audiovisual Profile (cf. Section 3.1.1). The validation approach is refined and the validation of temporal semantic constraints is also integrated. Three different MPEG-7 profiles were taken into account for the formalization process. These are the Audiovisual Profile (AVDP), the Detailed Audiovisual Profile (DAVP), and the TRECVid Profile. The specification of the TRECVid Profile was done during the work described in this thesis. Furthermore, the approach for formalizing temporal semantic constraints was published separately [54]. In addition, the general validation approach of VAMP was published in a journal paper [79].

1.4 Structure of This Thesis

This thesis is organized as follows: Chapter 2 gives an overview about the MPEG-7 standard and MPEG-7 profiles. Then Chapter 3 discusses the semantic constraints of profiles taken into account, including a usage analysis and a summarization of the most common interoperability issues encountered. Chapter 4 presents relevant technologies and related approaches in context of the described work. Then in Chapter 5, the approach for formalizing semantic constraints of MPEG-7 profiles is presented. The design and implementation of VAMP are explained in Chapter 6. As a proof on concept, two VAMP-based applications are presented in this Chapter. Finally, the conclusion of this work is drawn in Chapter 7.

Chapter 2

MPEG-7

MPEG-7, formally named Multimedia Content Description Interface, is an ISO/IEC standard[1] for representing metadata of multimedia content. This standard has been developed by the Moving Picture Experts Group[2] (MPEG) since 1996. The main focus of this group is the development of international standards for the compression and transmission of video and audio, such as MPEG-1, MPEG-2, and MPEG-4. While these MPEG standards represent the content itself, MPEG-7 represents metadata of multimedia content only.

In this Chapter, the key features of MPEG-7 are described. Afterwards, the MPEG-7 tools, which are the basic building blocks of this metadata standard, are explained. Then the organization of the MPEG-7 standard into several parts is described. Furthermore, the different options of describing multimedia content using MPEG-7 are presented. After a short introduction of applications processing and creating MPEG-7 descriptions, interoperability issues of MPEG-7 are discussed in detail. The idea of MPEG-7 profiles is introduced, and the process for defining a profile is explained and some existing profiles are described.

2.1 Key Features

MPEG-7 supports the description of various types of multimedia content, such as audio, video, images, speech, graphics, and 3D models and collections containing different multimedia items and presentations. Furthermore, different kinds of aspects and features of these content types can be described. For example, representing information about creation, usage, rights, and technical metadata is supported by MPEG-7. In addition, MPEG-7 provides various ways for describing structural and semantic information, summaries, collections, and user preferences. Therefore this standard can be used in a wide range of applications. Potential application domains are multimedia

[1] ISO/IEC 15938

[2] http://mpeg.chiariglione.org/

analysis and editing, digital libraries, multimedia directory services, broadcast media selection, journalism, surveillance, investigation services, and home entertainment [35].

Since MPEG-7 is a metadata description language only, it neither includes nor normatively defines techniques for metadata extraction from the multimedia content to be described. In addition, a metadata description using MPEG-7, also denoted as MPEG-7 description, is either stored separately from the content, or it is multiplexed along with the content. Furthermore, an MPEG-7 description is either represented in textual form encoded by the Extensible Markup Language (XML) [28] or in a proprietary binary format (Binary Format for XML (BiM) [1]). While the textual representation is suitable for editing and searching, the binary representation is more suitable for storage, transmission, and delivery.

MPEG-7 descriptions are independent of the storage and format of the described multimedia content. Thus all kinds of storage types, such as paper, tape, or disk, and streaming techniques are supported. Furthermore, the multimedia content can be encoded in any analog or digital format. For example, one MPEG-7 description represents a picture that is printed on a sheet of paper, while another one describes a song that is encoded as MP3 and stored on a solid-state drive (SSD), which is formatted with HFS Plus[3].

2.2 Tools

The specification of the MPEG-7 standard is based on a comprehensive set of standardized *tools*. Various requirement scenarios [71] were considered for the specification of these tools. Furthermore, these tools are divided into three groups: the *description tools*, the *Description Definition Language (DDL)*, and the *systems tools*.

Any MPEG-7 description about multimedia content is created by instantiating particular description tools. Therefore these description tools enable the description of various kinds of aspects and features of the content. The description tools consist of *descriptors (D)* and *description schemes (DS)*. A descriptor represents a single feature, attribute, or group of attributes of multimedia content. In this context, a feature refers to the distinctive characteristic of multimedia content [8]. In addition, different descriptors related to the description of audiovisual, visual, and audio characteristics are available. For example, the color or texture of an image, shapes of objects, motion in a video, and waveform of a song are represented by using appropriate descriptors.

A description scheme is a set of descriptors and data types. Therefore the relationships of the included descriptors and data types in terms of structure and semantics are defined. Relations to other description schemes are also defined. While a descriptor corresponds to the description of a specific single feature, a description scheme allows the representation of a more complex and structured description, such as a scene description or the detailed segmentation of the content in time and space.

[3] a file system developed by Apple Inc.

In order to define the syntax and semantics of the description tools and their relations, an appropriate language is required. For this purpose, the Description Definition Language (DDL) was developed. The DDL is based on XML Schema [39] and is the formal basis for the creation of MPEG-7 descriptions. By using XML Schema, elements, attributes, simple and complex types, and a set of syntactic, structural, and value constraints are specified. In combination with some extensions, such as array and matrix data types, all descriptors and description schemes are defined. As a result, an own MPEG-7 XML Schema was defined. By selecting and instantiating particular MPEG-7 description tools an MPEG-7 description about multimedia content is created.

In Figure 2.1, the composition of the MPEG-7 tools is depicted. XML Schema is the basis for the DDL, which is used to define the descriptors and description tools. Another part of these tools are the systems tools. The system tools are responsible for the implementation of requirements related to the binary representation, storage, transmission, multiplexing, and synchronization of MPEG-7 descriptions [73].

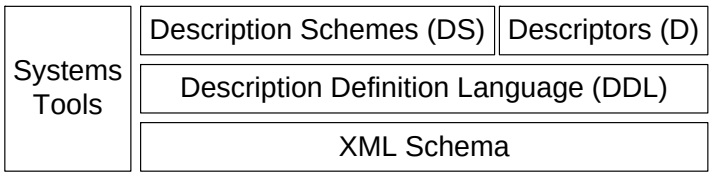

Figure 2.1: Organization of the MPEG-7 tools (adapted from [35]).

2.3 Parts

The specification of MPEG-7 is spread over 12 parts. In Table 2.1, the MPEG-7 parts and their corresponding ISO/IEC numbers are listed. Additionally, in Figure 2.2, the organization of the MPEG-7 tools is depicted in order to illustrate the relations between these parts.

Part 1 includes the systems tools. The Description Definition Language is defined in part 2. Part 3 and 4 define the description tools for the description of visual and audio features respectively. For example, part 3 (visual) includes color, texture, shape, and motion descriptors, while descriptors of part 4 (audio) describe waveforms or spoken content. In part 5, the description tools for describing multimedia content are defined. Additionally, basic elements and the structure of an MPEG-7 description are defined. Hence this part is considered as the most important one of the whole standard.

Part 6 provides a reference software implementation of the standard, while part 7 deals with guidelines and instructions for testing the conformance of implementations in context of the standard. Part 8 provides examples for the extraction and use of several descriptors. The usage of profiles and levels is defined in part 9. Part 10 comprises the MPEG-7 Schema for the parts 1, 3, 4, and 5, while part 11 includes the schema

Part	Title	Number
1	Systems	ISO/IEC 15938-1 [9]
2	Description definition language	ISO/IEC 15938-2 [5]
3	Visual	ISO/IEC 15938-3 [6]
4	Audio	ISO/IEC 15938-4 [7]
5	Multimedia Description Schemes	ISO/IEC 15938-5 [8]
6	Reference software	ISO/IEC 15938-6 [10]
7	Conformance testing	ISO/IEC 15938-7 [11]
8	Extraction and use of MPEG-7 descriptions	ISO/IEC TR 15938-8 [12]
9	Profiles and levels	ISO/IEC 15938-9 [13]
10	Schema definition	ISO/IEC 15938-10 [2]
11	MPEG-7 profile schemas	ISO/IEC TR 15938-11 [3]
12	Query format	ISO/IEC 15938-12 [4]

Table 2.1: Organization of the MPEG-7 parts.

definitions of adopted profiles. The latest part of the standard is part 12, which defines a query format.

The reference software, which is described in part 6, can be implemented in many different ways, as long as the conformance tests are passed. Thus this part is regarded as being non-normative. The same is true for part 8, which provides guidelines for the extraction of MPEG-7 descriptions. Except for these two parts, all other parts of the standard are normative.

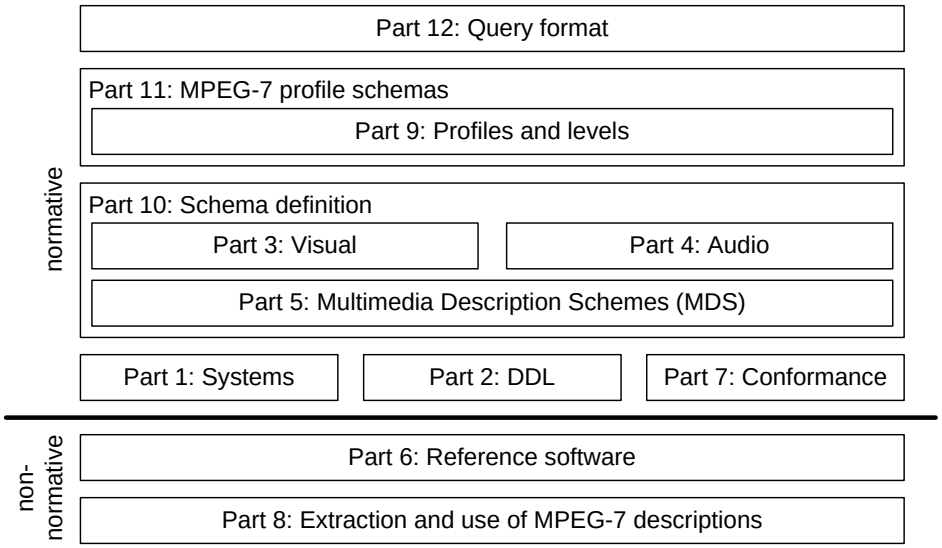

Figure 2.2: Structure of the MPEG-7 standard (adapted from [74]).

2.4 Multimedia Content Description

Any MPEG-7 description of multimedia content uses the Multimedia Description Schemes (MDS), which are defined in part 5 of the standard (cf. Section 2.3). The defined description tools describe various aspects and characteristics of the multimedia content. The domain-specific audio tools (part 4) and video tools (part 3) appear in combination with the description tools of the Multimedia Description Schemes only. Based on their functionality, the MDS are split into *basic elements*, *content management*, *content description*, *navigation and access*, *content organization*, and *user interaction*. An overview of the MPEG-7 Multimedia Description Schemes is depicted in Figure 2.3.

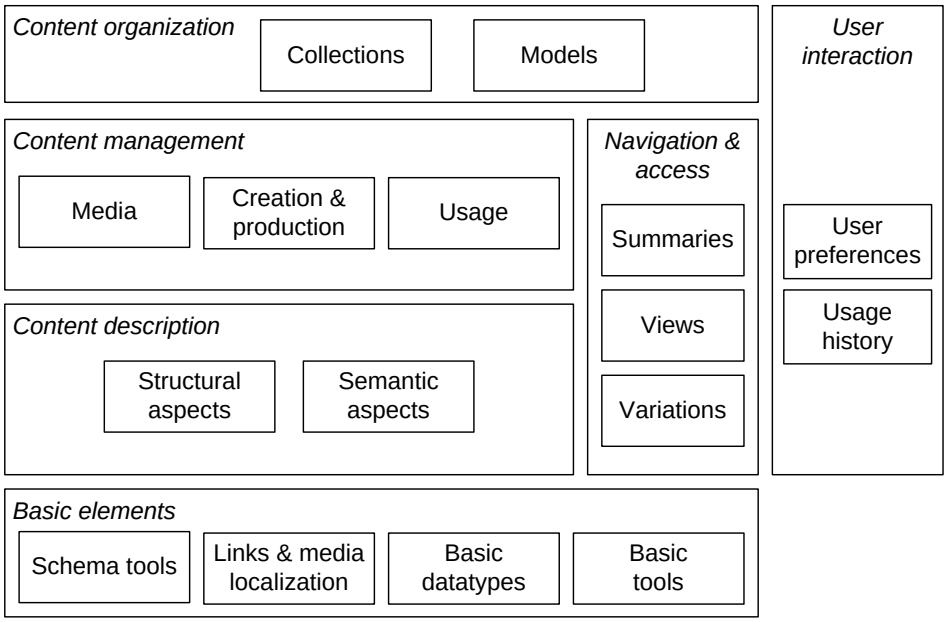

Figure 2.3: Overview of the MPEG-7 Multimedia Description Schemes (adapted from [35]).

The basic elements provide the generic data types and description tools for creating an MPEG-7 description. The root elements of an MPEG-7 description and related top-level description tools are defined. Moreover, the multimedia content entity tools, which are used for describing different kinds of multimedia content, such as images, video, and audio, are defined. In addition, the base types for defining the hierarchy of all other description tools are specified. Furthermore, descriptors for links and media locators and for describing time, places, persons, organizations, individuals, and classification schemes are provided.

In MPEG-7, content management is related to the description of creation, media, and usage information of the content. Therefore several description tools are available. The creation information tools provide information about the creation and production process of the content. For example, information about the creator, date, location, title, author, director is described. Classification information, such as rating, genre, or target

audience is also addressed. The media description tools describe information about the encoding, file format, storage location, and quality parameter, such as resolution and sampling rate. Rights information, information about the availability, and financial aspects are addressed by the usage information tools.

Supporting the description of structural and semantic aspects of multimedia content is an important feature, which is enabled by the content description tools. In this context, a structural description refers to the segmentation of the content into its subparts in terms of time and space, or based on different media sources. These segmentation variants are described by using different description schemes. In addition, the description schemes are used in combination with corresponding segment types. Furthermore, the segments involved in a decomposition can be annotated further to describe various related features, such as visual, audio, or creation information. In order to describe a decomposition hierarchy, decompositions are applied recursively on the sub-segments. Moreover, multiple decompositions on the same level are allowed to enable complex content descriptions. Then criteria, such as faces, objects, and events, are assigned to distinguish between different types of segmentations.

Describing the segmentation of multimedia content in terms of time and space or based on media sources is a necessary requirement for several use cases. For example, an MPEG-7 description that is representing the results of a shot boundary detection is based on the description of a temporal segmentation. The goal of this task is to split a video into appropriate temporal portions, which are denoted as shots. A shot represents a temporal sequence of continuous action recorded by a single camera. Such a shot boundary detection is used, among others, for editing, archiving, or quality assessment tasks for videos. In addition, a key frame extraction process can be applied to shots in order to detect single frames that are representing entire shots. Then these key frames are also described by a temporal segmentation structure in MPEG-7.

Other use cases require the detection or recording of the positions of objects in an image. Similar use cases are related to the representation of the movement of objects in a video. For example, in a video surveillance environment, an object tracker is used to track the movement of a person in space and time. Therefore MPEG-7 provides spatial (image-related) and spatio-temporal (video-related) segmentation structures. Furthermore, when shooting a scene from different camera views or recording audio from multiple channels, a media source decomposition structure allows to distinguish between these different modalities.

Besides the description of structural aspects, the ability of describing semantic aspects of the content is also important. In MPEG-7, a semantic description is related to a scene description based on objects, events, concepts and their relations. For instance, when describing the handshake between two persons, which is depicted in a picture, the persons are represented as objects and the handshake as event. The whole scene can be tagged by a descriptive concept, for example “friendship”. Furthermore, structural and semantic descriptions can be linked together. Then the combination of these descriptions leads to additional knowledge expressed by an MPEG-7 description.

An MPEG-7 description can also include a summary for supporting browsing, navigation, and retrieval tasks. Therefore a summary is represented in either hierarchical or sequential order. A hierarchical order is based on different levels of detail, while a sequential order refers to a slide-show-fashion rather. Moreover, the creation of collections based on different media items and user preferences is possible. Such a description is related to the consumption of multimedia content.

2.5 Applications Using MPEG-7 Descriptions

The W3C Multimedia Semantics Incubator Group[4] has published a comprehensive list of applications and tools that are creating and processing MPEG-7 descriptions[5]. In the following, some of these tools are introduced. The selected tools are grouped by their predominant media types:

Video-related tools: IBM VideoAnnex[6] is a semi-automatic annotation tool for video sequences. It supports the creation of shot segmentations and spatial decompositions of key frames. In addition, controlled vocabularies, which are based on a classification scheme (cf. [8]), are employed.

Muvino[7] is a tool for manually annotating videos. Therefore the segmentation of the video content in time is represented by MPEG-7. The content creation descriptors, for example for describing place, date, and creator, are used. Values of these creation descriptors are expressed either as free text annotations or using predefined keywords.

The Metadata Editor[8], which was developed by NHK[9], is an application for producing and storing metadata in terms of the Metadata Production Framework (MPF)[10]. The MPEG-7 standard was adapted and an own MPEG-7 profile (cf. Section 2.7) was created. This profile represents the metadata model of the MPF [16]. Besides the support of this profile, the AudioVisual Description Profile (cf. Section 2.7.2) is supported. In addition, profile-related semantic constraints(cf. Section 2.7.1) are implemented by the Metadata Editor directly.

Image-related tools: Caliph & Emir[11] is a semi-automatic annotation tool for images. This tools supports free text and graph-based annotation methods. In addition, a large number of MPEG-7 visual feature descriptors are employed.

[4] http://www.w3.org/2005/Incubator/mmsem/
[5] http://www.w3.org/2005/Incubator/mmsem/wiki/Tools_and_Resources
[6] http://www.research.ibm.com/VideoAnnEx
[7] http://vitooki.sourceforge.net/components/muvino/code/index.html
[8] http://www.nhk.or.jp/strl/mpf/english/editor.htm
[9] Nippon Hoso Kyokai, Japanese public broadcasting corporation
[10] http://www.nhk.or.jp/strl/mpf/english/index.htm
[11] http://www.semanticmetadata.net/features/

Moreover, pre-existing metadata, such as Exif[12] or IPTC[13] tags inside images, is converted into MPEG-7 descriptions in an automated fashion by this tool. The underlying conversion approach is based on mapping rules.

The M-OntoMat-Annotizer[14] supports the manual annotation of regions of a still image. Therefore domain specific ontologies were created. In addition, mappings from low-level MPEG-7 visual descriptors to these ontologies were established.

Audio-related tools: The MPEG-7 Audio Analyzer[15] is used to measure several characteristics of a given audio sample. The results of such an audio analysis are represented by an MPEG-7 description. Therefore all low-level descriptors, which are defined in part 4 of the standard, are implemented by this tool.

In addition, the MPEG-7 Audio Encoder[16] is a Java-based tool, which is used to describe the content of an audio file. Therefore some of the audio-related descriptors of MPEG-7 are used.

The MPEG-7 Spoken Content Demonstrator[17] generates the output of an Automatic Speech Recognition (ASR) system. Again, this output is represented by MPEG-7. More precisely, the `SpokenContent DS` is used for this purpose.

2.6 Interoperability Issues

MPEG-7 is designed for the description of different types of multimedia content, covering a wide range of application areas. Therefore various general and widely applicable description tools are available for describing different features of the multimedia content. As a result, MPEG-7 is very comprehensive. In fact, the definition of the description tools comprises 1182 elements, 417 attributes, and 377 complex types [51]. Furthermore, MPEG-7 allows flexible use of the description tools. Thus these tools enable descriptions that are associated with the whole content itself or specific portions only. In addition, the level of detail and granularity of these descriptions can be adapted according to the needs of a specific use case. For instance, arbitrary segments or regions of the content can be addressed and described by various features. Additional flexibility is achieved by extending MPEG-7. If the existing description tools are not sufficient for a certain application, the standard can be extended according to the conformance guidelines, which are defined in part 7 of the standard specification.

When using MPEG-7, the flexibility of MPEG-7 leads to increased complexity and fuzziness. As a consequence, the use of MPEG-7 as an efficient multimedia description

[12] Exchangeable Image File Format
[13] Information Interchange Model developed by the International Press Telecommunications Council
[14] `http://www.acemedia.org/aceMedia/results/software/m-ontomat-annotizer.html`
[15] `http://mpeg7lld.nue.tu-berlin.de/`
[16] `http://mpeg7audioenc.sourceforge.net/`
[17] `http://mpeg7spkc.nue.tu-berlin.de/`

language is limited [69, 21, 78]. The complexity is a result of the comprehensiveness of MPEG-7, since a high number of the description tools is available. In addition, these description tools are based on generic concepts and refinements and are organized using deep hierarchical structures. As a result, learning and understanding the specification of MPEG-7 is difficult and time-consuming, ending in a certain hesitance for using MPEG-7 in applications. Fuzziness is a result of the syntax variability and a certain lack of formal semantics. The syntax and structure of the MPEG-7 description tools are defined by the DDL. Explanations in textual form provide extra information about the usage and the semantics of these tools. However, the DDL contains flexible definitions and the textual description is done on a general level only, not to exclude possible application scenarios. As a result, ambiguities in the interpretation of the standard exist. Syntactically different MPEG-7 descriptions are possible that are covering the same semantics.

For example, structural description tools can be arranged and combined in different ways for describing a temporal segmentation of a video. The resulting structures of two different MPEG-7 descriptions are sketched in Listing 2.1 and Listing 2.2. These descriptions are based on different `MultimediaContent` types. The type in the first description is the `VideoType`, while the second description is based on the `AudioVisualType`. The `VideoType` is refers to the content description of a video only, while the `AudioVisualType` is used to describe video, audio, and audiovisual content. In Listing 2.1, the video itself is denoted by a `Video` element, while temporal portions of this video are denoted by `VideoSegment` elements, which are enclosed by a `TemporalDecomposition` element. In Listing 2.2, the video part of an audiovisual description is separated in the first place. Therefore the `MediaSourceDecomposition DS` is used, and this video part is denoted by a `VideoSegment` element. Then the temporal segmentation is described using the same descriptors as in Listing 2.1.

```
<Mpeg7>
  <Description xsi:type="ContentEntityType">
    <MultimediaContent xsi:type="VideoType">
      <Video id="video1">
        ...
        <TemporalDecomposition>
          <VideoSegment id="segment1"> ... </VideoSegment>
            ...
          <VideoSegment id="segmentM"> ... </VideoSegment>
          <VideoSegment id="segmentN"> ... </VideoSegment>
        </TemporalDecomposition>
      </Video>
    </MultimediaContent>
  </Description>
</Mpeg7>
```

Listing 2.1: Outline of a temporal segmentation in MPEG-7 based on the `VideoType`.

```
<Mpeg7>
  <Description xsi:type="ContentEntityType">
    <MultimediaContent xsi:type="AudioVisualType">
      <AudioVisual id="av1">
        ...
        <MediaSourceDecomposition>
          <VideoSegment id="video1">
            ...
            <TemporalDecomposition>
              <VideoSegment id="segment1"> ... </VideoSegment>
              ...
              <VideoSegment id="segmentM"> ... </VideoSegment>
              <VideoSegment id="segmentN"> ... </VideoSegment>
            </TemporalDecomposition>
          </VideoSegment>
        </MediaSourceDecomposition>
      </AudioVisual>
    </MultimediaContent>
  </Description>
</Mpeg7>
```

Listing 2.2: Outline of a temporal segmentation in MPEG-7 based on the `AudioVisualType`.

Although the outcome of these two Listings are different MPEG-7 descriptions, the underlying intention, which is describing a temporal segmentation of a video, is the same. This syntax variability can cause interoperability problems, if applications are using different variations. For example, when exchanging MPEG-7 descriptions between applications or systems that are creating syntactically different MPEG-7 descriptions, while covering the same intended semantics. Such syntax-based interoperability issues are avoided, if all possible syntactic variations are supported by any application. However, this is approach is not feasible in practice.

Besides the syntax variability, additional interoperability issues are caused by limitations of the semantic expressiveness of MPEG-7. These issues are triggered when one description tool describes multiple semantically different features or concepts of the content. For example, when describing the segmentation of a video by a shot list including key frames, a temporal segmentation structure similar as shown in Listing 2.1 or Listing 2.2 can be used. In this context, the `VideoSegment DS` is used for representing the video itself and for included shots and key frames also (cf. [78]). However, when considering the XML Schema definition of the `VideoSegment DS` only, it is incapable to distinguish between the concepts of a video, shot, and key frame. Neither is it possible to formulate different restrictions on these concepts. For example, to express that a video is having exactly one shot list, a shot list contains shots only, and a key frame cannot be decomposed further at all.

In some cases, the usage of the description tools is clearly formulated not leaving any space for misinterpretations. These explanations are expressed as textual guidelines

in the MPEG-7 standard. For example, the correct use of the description tools for describing a temporal segmentation in terms of MPEG-7 are stated. In this context, the semantics of attributes addressing some specific characteristics of such a segmentation is explained. However, most of the description tools are rather defined in a general way not to exclude possible application scenarios. Therefore the description tools are defined mostly general not providing a way to address semantic limitations. Due to this lack of formal semantics of MPEG-7 descriptions, the resulting interoperability problems prevent an effective use of MPEG-7 for describing multimedia content in some application areas [78].

2.7 Profiles

As described in the previous Section, the modeling foundations of MPEG-7 cause complexity, syntax variability, and semantic ambiguities of MPEG-7 descriptions. In order to overcome these issues, the notion of *profiles* were introduced by the MPEG working group [42]. In contrast to the whole standard, an MPEG-7 profile clearly defines the usage and semantics of the description tools tailored to a particular application domain. Then neither syntax variability nor misinterpretation should be possible when creating an MPEG-7 descriptions with respect to a profile. Such a description is denoted as MPEG-7 profile description. As a result, the interoperability of such MPEG-7 descriptions is improved, which is a precondition for exchanging descriptions between different applications.

2.7.1 Defining a Profile

A profile represents a subset of the MPEG-7 standard in terms of the applicable description tools. The starting point of a profile definition is a profile specific XML Schema, which is defined by the MPEG-7 DDL. Thus this profile XML Schema contains the description tools needed for covering the intended application domain only. In combination with additional tool constraints, such as exclusions and cardinality restrictions, the syntax variability and complexity of MPEG-7 descriptions that are created with respect to a profile are limited. Any MPEG-7 description according to a profile must conform to the MPEG-7 XML Schema. Additionally, the semantics and usage of the profile in context of the intended application domain can be expressed as additional usage instructions. The following three steps were proposed for defining an MPEG-7 profile [42]:

1. **Selection of included description tools:** First, the description tools needed for describing the intended functionality of the profile are selected. Since these description tools are a subset of MPEG-7, this step reduces the number of descriptors and description schemes compared to the whole standard.

2. **Definition of description tool constraints:** The second step is to define constraints related to the structure of the selected description tools. Therefore exclusions or cardinality restrictions on the description tools are expressed by the DDL. For example, to tighten cardinality restrictions of XML elements or change the occurrence of XML attributes from optional to mandatory.

3. **Definition of semantic constraints:** In this optional step, the semantics and usage of the profile in context of the intended application domain is expressed. For example, expressing how the description tools should be used and combined or how the elements and attributes in a resulting MPEG-7 profile description have to be interpreted. These usage instructions and explanations are denoted as semantic constraints, which are phrased as additional guidelines written in English. These guidelines, usually being part of the profile documentation, extend the corresponding profile XML Schema. Expressing the semantic constraints in this way is a result of the limited semantic expressiveness of the MPEG-7 DDL (cf. Chapter 3).

The first two steps of a profile definition address the complexity issue by limiting the number of descriptors and description schemes and by excluding elements or constraining their cardinality. The selection of the description tools and most of the tool constraints are specified by the DDL, which results in a more specific and constrained profile XML Schema. In addition, expressing semantic constraints clarifies possible ambiguities associated with the use of a profile and the included description tools. In summary, understanding and following the semantic constraints of a profile are key requirements in order to ensure the interoperability of MPEG-7 profile descriptions.

2.7.2 Existing Profiles

Currently four profiles are standardized by the MPEG working group [13, 14]. These are the *Simple Metadata Profile (SMP)*, *User Description Profile (UDP)*, *Core Description Profile (CDP)*, and *AudioVisual Description Profile (AVDP)*. These profiles are described in part 9 and their related XML schemas are defined in part 11 of the standard. Besides these adopted profiles, other non-standardized profiles exist. For example, the *Detailed AudioVisual Profile (DAVP)*, *TRECVid profile*, and *NHK Metadata Production Framework (MPF)*.

In the following, these standardized and non-standardized profiles are introduced.

Simple Metadata Profile (SMP): The Simple Metadata Profile describes single instances or collections of multimedia content. This profile contains description tools for describing textual metadata only. Thus structural descriptions for the segmentation of the content are not supported. The motivation of this profile is to support simple metadata tagging similar to ID3[18] for music and Exif[19] for

[18]Iterative Dichotomiser 3: `http://www.id3.org/`
[19]Exchangeable Image File Format: `http://www.exif.org/`

images. Another use case is the support of mobile applications such as 3GPP[20].

User Description Profile (UDP): The purpose of the User Description Profile is to describe the personal preferences and usage patterns of users that are consuming multimedia content. Therefore the goal of the UDP is to enable an automatic discovery, selection, personalization, and recommendation of multimedia content for users. This profile contains all MPEG-7 description tools that were adopted by the TV-Anytime Forum[21] and referenced by the TV-Anytime Metadata specification [72].

Core Description Profile (CDP): The Core Description Profile consists of tools for describing a multimedia content in general. The described content can either be an image, video, or audio. For describing this content, the top-level types defined in part 5 of the standard are employed. A typical use case of this profile is the description of structural aspects of video content of a TV program and its corresponding materials. This includes the description of media management, distribution, and archiving information.

Detailed Audiovisual Profile (DAVP): The Detailed Audiovisual Profile [22], which was developed by JOANNEUM RESEARCH[22], provides a clear definition for content description tailored to audiovisual productions, search and retrieval tasks, and media monitoring. This profile supports the description of audio, video, audiovisual content, and still images. By using the DAVP, a comprehensive structural description of these content types is possible. Thus different structural segmentations can be applied to the whole content itself and to arbitrary portions on different levels of detail. Therefore this profile includes many of the multimedia description tools that are defined in part 5 of the standard. Besides all structuring tools for describing segmentations, tools for describing media, creation, usage information, and summaries are included in this profile. Furthermore, the DAVP supports audio and visual feature descriptions that are required for comparison, filtering, and browsing of the content. The required visual and audio tools are defined in part 3 and 4 of the standard. In none of the former profiles, these parts were included. In addition, many semantic constraints are defined to clarify the use of the profile and included description tools.

TRECVid profile: The aim of the TRECVid profile is to represent the master shot boundary reference data of the TREC Video Retrieval Evaluation[23]. Therefore the shot structure of a video and the representation of associated key frames for each shot are defined. For this purpose, including a small set of the multimedia description tools defined in part 5 is sufficient. Thus the audio and video tools defined in part 3 and 4 are not part of this profile. Semantic constraints address

[20] 3rd Generation Partnership Project: `http://www.3gpp.org/`
[21] `http://www.tv-anytime.org/`
[22] `http://mpeg7.joanneum.at/`
[23] `http://www-nlpir.nist.gov/projects/trecvid/`

restrictions related to valid structural descriptions. This profile was developed along with the work described in this thesis. The specification consists of an XML Schema[24] and the text-based semantic constraints (cf. Section 3.1.2).

The profile XML Schema of the TRECVid profile is online available[25], while the relevant semantic constraints are defined in Section 3.1.2.

NHK Metadata Production Framework (MPF): The NHK Metadata Production Framework data model [16] is an industrial application of the Core Description Profile (CDP). This framework addresses complexity and ambiguity problems of MPEG-7 by defining a metadata model. This model further restricts the CDP by additional exclusions and cardinality restrictions. The use of the visual and audio descriptors defined in parts 3 and 4 is permitted. The definition of the data model contains a number of semantic constraints related to the structural description as well as several syntactic and semantic constraints on different elements of the description. In terms of the MPF these constraints are denoted as "operational rules".

AudioVisual Description Profile (AVDP): The goal of the AudioVisual Description Profile [14] is to provide a uniform way for describing the results of media analysis tasks. Therefore different kinds of analysis tasks, such as shot or scene detection, face recognition and tracking, speech recognition, and summarizations, are supported by this profile. One major aspect of the AVDP is to ensure interoperability when exchanging MPEG-7 descriptions that are representing the analysis results. The AVDP supports the description of audio, video, and audiovisual content. Thus the AVDP includes the description tools that are defined in the parts 3, 4, and 5 of the standard. In addition, a AVDP-based description supports multiple modalities, which can be coexisting side by side. In this case, a clear separation between the different modalities and their corresponding metadata and metadata sources is possible. The correct use of this profile including restrictions related to the segmentation options on different context levels are expressed by semantic constraints.

The AVDP was developed by the EBU MIM/SCAIE group[26], based on the Detailed Audiovisual Profile and the NHK Metadata Production Framework. Thus this profile was co-edited by members of the Audiovisual Media Group at JOANNEUM RESEARCH[27]. The AVDP has become an standardized MPEG-7 profile at the 100$^{\text{th}}$ MPEG meeting in May 2012[28].

[24] http://vamp.joanneum.at/data/xsd/trecvid_xsd/trecvid-2001.xsd
[25] http://vamp.joanneum.at/data/xsd/trecvid_xsd/
[26] http://tech.ebu.ch/groups/pscaie
[27] http://www.joanneum.at/en/digital/avm.html
[28] http://mpeg.chiariglione.org/meetings/100

2.8 Summary

MPEG-7 is a metadata description standard for representing the metadata of various types of multimedia content, such as video and image, or audio and speech. While other MPEG standards represent the content itself, MPEG-7 represents metadata of the content only. Since the description of different kinds of aspects and features of these content types is supported by MPEG-7, it is used in a broad range of application areas. In addition, a large set of description tools is provided by MPEG-7. All these tools are defined by the Description Definition Language (DDL), which is based on XML Schema. In order to create an MPEG-7 description, appropriate tools are selected and instantiated. The resulting MPEG-7 description is either represented in textual form encoded in XML or in a binary format (BiM).

When using MPEG-7 practically, two major drawbacks were identified, namely complexity and fuzziness. Complexity is mainly based on the comprehensiveness of MPEG-7, while fuzziness is a result of the syntax variability and the lack of formal semantics. Serious interoperability issues can occur when processing and exchanging multimedia content between applications that are interpreting the standard differently. The resulting interoperability problems limit an effective use of MPEG-7 as a language for describing multimedia content.

MPEG-7 profiles were introduced in order to address and possibly solve these problems. A profile defines the usage and semantics of the description tools tailored to a particular application domain. Neither syntax variability nor space for misinterpretations should be possible when creating MPEG-7 descriptions being conform to a profile. Similar to the MPEG-7 standard itself, a profile definition is based on the MPEG-7 DDL resulting in a profile related XML Schema. Optionally, the semantics and usage of a profile in context of the tailored application domain can be stated. These usage instructions and explanations are denoted as semantic constraints and are expressed as textual guidelines written in English.

Chapter 3

Interoperability Issues of MPEG-7 Profile Descriptions

The modeling foundations of the MPEG-7 standard cause syntax variability, complexity, and semantic ambiguities of MPEG-7 descriptions (cf. Section 2.6). In order to overcome these issues, MPEG-7 profiles (cf. Section 2.7) were introduced. A profile is a subset of the whole MPEG-7 standard that is tailored to a particular application domain. When defining a profile, usage instructions and explanations can be stated in order to eliminate ambiguities in interpreting a profile and resulting MPEG-7 profile descriptions. These instructions are denoted as semantic constraints, which are provided as prose text extending a profile XML Schema.

In this Chapter, the semantic constraints of three selected MPEG-7 profiles used to describe audiovisual content are summarized. Then a usage analysis of these semantic constraints is presented. Furthermore, possible inconsistencies of MPEG-7 profile descriptions are discussed and classified. These inconsistencies are the result of the rather informal definition of the semantic constraints in terms of machine-processability and understandability.

3.1 Semantic Constraints of Selected MPEG-7 Profiles

In this Section, the semantic constraints of three selected MPEG-7 profiles, presented in Section 2.7.2, are summarized. These profiles are the Audiovisual Profile (AVDP), the Detailed Audiovisual Profile (DAVP), and the TRECVid Profile. The function of these profiles is to ensure an unambiguous description of metadata of audiovisual content.

3.1.1 Detailed Audiovisual Profile

The semantic constraints of the DAVP are expressed as prose text in the corresponding profile documentation (cf. [22, Section 4]). These constraints are mainly addressing the resulting structure of MPEG-7 profile descriptions. Thus the various options for

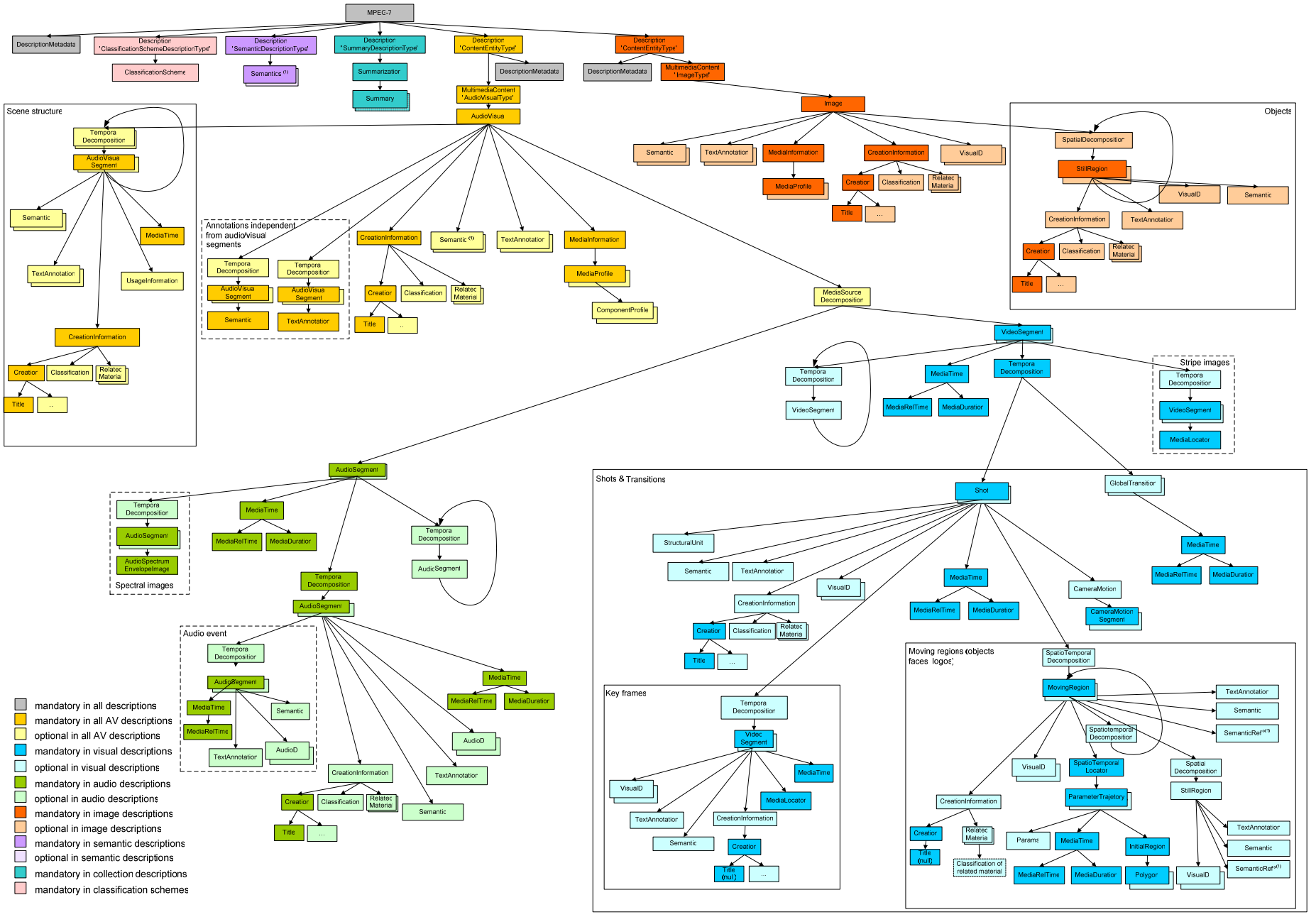

Figure 3.1: Structure of an MPEG-7 description according to the DAVP (taken from [22]).

creating an MPEG-7 description being conformant to the DAVP are depicted in Figure 3.1. Most of the structural semantic constraints can be derived from this graphical setup. One goal of these constraints is to establish a structure as modular as possible. Hence metadata portions coming from different sources or related to different modalities are stored in separate parts of the description. For instance, the description of audiovisual content is split into separate visual and audio parts. As a result, dependencies between different structural descriptions are reduced to a minimum, allowing to change and access these different descriptions without influencing other parts at the same time. Descriptions based on different levels of abstraction and common decomposition structures are also kept separately. Therefore the use of the decomposition tools is regulated in order to define appropriate spatio, temporal, and spatio-temporal decomposition structures related to different context levels. For instance, to describe a scene, a shot and key frame structure, or an object segmentation.

The structural semantic constraints of the DAVP and their consequences to the representation of related MPEG-7 profile descriptions are summarized as follows:

- Only one single multimedia content is described per MPEG-7 description. Therefore the root `Mpeg7` element has exactly one `Description` element of type `ContentEntityType`.
- The described content is either an audiovisual content or an image. Thus the `MultimediaContent` element is either of type `AudioVisualType` or `ImageType` containing exactly one `AudioVisual` or `Image` element respectively.
- An audiovisual content can be split into its video and audio portions. Therefore the top `AudioVisual` element has at most one `MediaSourceDecomposition` element denoted by setting the value of the `criteria` attribute to *modalities*. This decomposition contains at least one `AudioSegment` or `VideoSegment` element having the same start time and duration as the associated `AudioVisual` element.
- All `AudioVisual`, `Video`, and `Audio` segments are denoted by a unique `ID` in order to enable references.
- Metadata based on both visual and audio information is placed to the root `AudioVisual` element using a decomposition structure based on a `TemporalDecomposition` or `SpatioTemporalDecomposition` element. `AudioVisual` elements are part of such a decomposition. Additionally, applying a recursive structure is allowed.
- A scene description is based on both visual and audio information. Such a description is attached to the root `AudioVisual` element represented as `TemporalDecomposition` containing `AudioVisual` elements. These elements can be recursively applied in order to describe sub scenes. The value of the `criteria` attribute of the decomposition is assigned to *scene* and neither temporal gaps nor overlaps are allowed.

- For describing the shot structure of a video, a `TemporalDecomposition` element is attached to the top `Video` element. This temporal decomposition may contain `Video` elements only, which are representing shots. In the same manner, key frames are described as `Video` elements within a temporal decomposition of a shot. One shot structure per content containing one list of key frames per shot is allowed only. A shot may be decomposed into key frames only, while key frames must not be decomposed further at all. The appearance of shots is mandatory, while the appearance of key frames is optional.

- The temporal decompositions included in a shot list are qualified by the `criteria` attribute. For decompositions of a video into shots this value is set to *visual shots*, while the value is set to *key frames* for temporal decomposition of shots into key frames.

- Furthermore, a temporal decomposition containing shots allows neither gaps nor overlaps, while a temporal decomposition containing key frames has gaps but no overlaps.

- For the video itself and all included shots time point and duration information must be present, whereas key frames have time point information only.

Another group of semantic constraints regulates the use of the media information tools on different positions in the content description. The description of the media information is based on media profiles describing different media features such as content type, file format, file size, signal quality, and content location. These constraints ensure a correct link between the description of the technical metadata and the content. Examples of semantic constraints related to media information are:

- A `MediaInformation` element must be part of the MPEG-7 description. These information can be only attached to the root `AudioVisual` or `Image` element respectively. In contrast, the presence of a stand-alone `MediaLocator` element is not permitted in combination with these root elements.

- A media profile may include multiple component profiles only when it is attached to an `AudioVisual` segment. Then the number of component profiles can be the same as the number of modalities described in the MPEG-7 description.

- For all audiovisual, video, and audio segments expect the root ones, a stand-alone `MediaLocator` element may be present. However, this element must not be enclosed by a `MediaInformation` element.

Finally, semantic constraints related to the creation and production process and summaries are included in the profile. For example:

- A `CreationInformation` element must be present for the root `AudioVisual` segment. In contrast, describing the creation information is optional for all other video, audio, and audiovisual elements.

- One summary per content description is allowed only. However, a stand-alone summary is also permitted. In any case, a summary refers to one certain multimedia content only.

3.1.2 TRECVid Profile

The TRECVid profile defines the description of a shot list with respect to the reference data of the TREC Video Retrieval Evaluation. Therefore an MPEG-7 description according to this profile describes the temporal decomposition of video into shots including associated key frames for each shot. All involved video, shot, and key frame segments are defined by the `VideoSegmentType`.

The related semantic constraints define the decomposition structure and the of use of the MPEG-7 tools on different context levels:

- Exactly one shot structure of a video is described per MPEG-7 profile description. Therefore the root `Video` element contains exactly one `TemporalDecomposition` element, which is representing the shot list. Other decomposition types are not allowed.
- The temporal decomposition of a video may include shots only. In addition, one shot must be present at least.
- Corresponding key frames are described for each shot. Therefore exactly one `TemporalDecomposition` element must be present for each shot segment. Other decomposition types are not allowed.
- A temporal decomposition of a shot into key frames may only include key frames. At least one key frame must be present.
- A key frame cannot be decomposed further at all.
- A temporal decomposition into shots allows neither gaps nor overlaps, while a temporal decomposition into key frames has gaps but no overlaps.

In addition, constraints on segments related to references and time-related information are stated:

- All segments are denoted by a unique ID in order to enable references.
- The root `VideoSegment` element contains a Uniform Resource Locator (URL)[1] serving as locator. In contrast, all other segments must not contain any locator information.
- For the video itself and all included shots, time point and duration information must be present, whereas key frames have time point information only (no duration information).

[1] `http://tools.ietf.org/html/rfc1738`

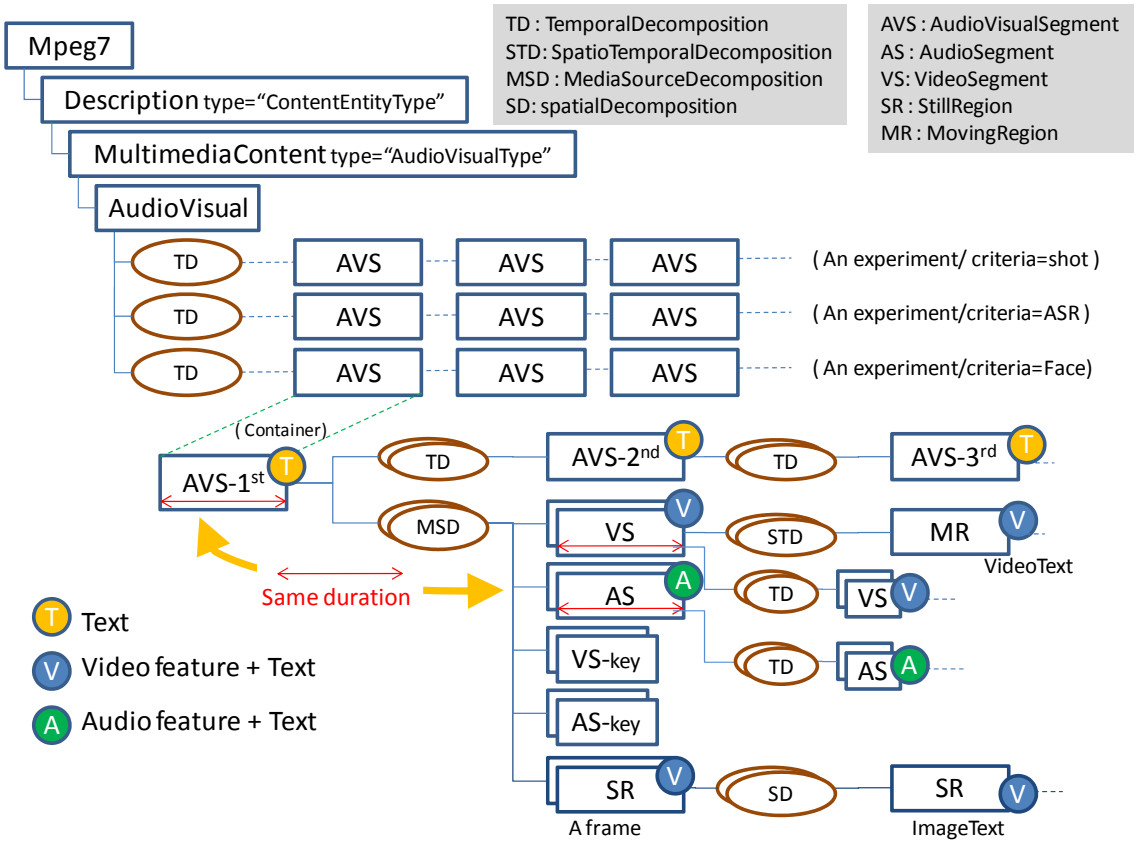

Figure 3.2: Structural composition of an MPEG-7 description according to the AVDP (taken from [14]).

3.1.3 AudioVisual Description Profile

The AudioVisual Description Profile (AVDP) were developed in order to record the results of different kinds of automatic media analysis tasks of audiovisual content. Thus the structural composition of MPEG-7 descriptions according tho the AVDP is addressed by most of the semantic constraints. Such a composition is depicted in Figure 3.2, These constraints are defined in prose text in [14, Section 4.5.5] and are summarized as follows:

- The most common use case of the AVDP is the description of one single audiovisual content. Therefore exactly one `Description` element is used. In addition, the `type` attribute of this element is set to `ContentEntityType` in order to designate a content description. Alternatively, relations between more than two audiovisual contents can be described, for example copy detection results. Then multiple `Description` elements are part of the MPEG-7 profile description.

- A summary is described either exclusively or in combination with the description of the related audiovisual content (one or multiple). The description of a summary starts with a `Description` element designated as `SummaryDescriptionType`.

- Different media analysis results are represented by separate temporal decomposition structures. The node for attaching such decompositions is the root `AudioVisual` segment, which is included in the `MultimediaContent` element. Each analysis result is represented as `TemporalDecomposition` element containing `AudioVisualSegment` elements. In order to distinguish between different analysis tasks, appropriate `criteria` values must be set for the `TemporalDecomposition` elements.

- In Figure 3.2, the temporal segmentation of the root `AudioVisual` segment into further `AudioVisual` segments is depicted. In this context, n in the term *AVS-n*, which represents an `AudioVisual` segment, refers to the level in the decomposition structure. Textual descriptions may be attached directly. An `AVS-1`st segment may serve as a container for additional descriptions. `TemporalDecomposition` and `MediaSourceDecomposition` structures are also permitted. However, other decomposition types are not allowed at this level.

- The description of a temporal decomposition hierarchy is acceptable for `AVS-1`st segments. Therefore multiple `TemporalDecomposition` elements are permitted at an unlimited number of decomposition levels. The included `AudioVisual-Segments` may be described by textual annotations, as described for `AVS-2`nd and `AVS-3`rd in Figure 3.2. However, at levels deeper than two, criteria and structural unit values already used at previous levels must not be used again.

- In addition, the `MediaSourceDecomposition` type is used to describe the decomposition of `AVS-1`st segments based on modalities into its video and audio parts respectively (cf. `VS` and `AS` in Figure 3.2). Furthermore, the description of key frames (`VS-key` and `AS-key`) and the results of a visual feature extraction analysis represented as still regions (`SR`) may be included in such decomposition structure. Thus multiple `MediaSourceDecomposition` elements are permitted. For example, to separate the decompositions based on modalities and key frames. Then different `criteria` values are required. In case the `criteria` is set to `genericmsd`, as recommended by using the `DecompositionCS`[2], all possible segment types may be combined into one decomposition. However, this criteria value must not be used if more than one decomposition is described for a `AVS-1`st segment.

- When describing the decomposition of an `AVS-1`st segment into its video and audio segments, the number of these segments is limited by the number of available audio and video channels. Additionally, all audio and video segments must have the same duration information as the parent `AVS-1`st segment. Additionally, appropriate structural unit values must be set for these segments.

- The video (`VS`) and audio (`AS`) segments of an `AVS-1`st segment can be decomposed further. Therefore the description of a temporal decomposition hierarchy (`TD`) is

[2] An MPEG-7 classification schema: `http://www.ebu.ch/metadata/cs/mpeg/avdp/DecompositionCS.xml`

permitted for these segments. In this hierarchy, multiple `TemporalDecomposition` elements at the same level and unrestricted number of decomposition levels are acceptable. In the same manner, moving regions (`MR`) of video segments can be described by using `SpatioTemporalDecomposition` elements (`STD`).

- The description of key clips (`VS-key` and `AS-key`) is also supported. A key clip, which is an extension of a key frame, may be represented for audio and video segments of the content. The duration of a key clip is less than the duration of the parent `AVS-1`st segment. Furthermore, a key clip must not be decomposed further nor annotated by feature descriptors.

3.2 Usage Analysis of Semantic Constraints

The presented semantic constraints of the selected MPEG-7 profiles are textual guidelines that are clarifying the correct use of these profiles in their intended application domains. Therefore the semantic constraints affect the use and combination of the applicable description tools of the MPEG-7 profiles, which is reflected in the resulting MPEG-7 profile descriptions. For instance, the presence, position, and interpretation of XML constructs (elements and attributes) in these descriptions are discussed.

Most of the semantic constraints being reviewed are related to the structural description of multimedia content. On the one hand, they influence the general setup of MPEG-7 profile descriptions. Therefore the use of the applicable description tools on the top-level nodes of the resulting MPEG-7 profile descriptions is explained. On the other hand, segmentation options of the content are specified by semantic constraints. For example, to describe the semantics of a decomposition hierarchy to describe the decomposition of a video into shots and key frames. Therefore semantic constraints address the function of the decomposition structure in general and on different decomposition levels in order to distinguish between different decomposition types. For instance, AVDP enables the description of multiple shot lists that are based on different analysis tools and parameters. Here, semantic constraints are used to distinguish between semantically different concepts and their valid occurrences on different decomposition levels. Moreover, the presence and position of media and creation information in MPEG-7 profile descriptions is defined by semantic constraints of some profiles. In addition, the use of cross references to external classification schemes is specified. Finally, the correct setup of summaries of the content is defined.

When defining an MPEG-7 profile, the boundaries between semantic constraints and tool constraints can be fuzzy in some cases. This happens when the intention of a tool constraint cannot be expressed by the related profile XML Schema fully. For example, expressing that certain segments must be denoted by unique identifiers in order to enable references cannot be defined. As a consequence, a supplementary textual description supports the profile XML Schema. In this thesis, tool constraints that are addressed by additional textual descriptions are treated as semantic constraints.

The semantic constraints of MPEG-7 profiles are comparable to the textual descriptions of the description tools of the standard, which are denoted as tool semantics [21]. Profile semantic constraints are restrictive and related to a specific application domain only, whereas the tool semantics is defined in a general way, not limited to specific application scenarios. For example, the semantics of temporal or spatial decompositions are described by the tool semantics of the standard. Then the semantics of specific decomposition structures are defined in an MPEG-7 profile specification.

Expressing semantic constraints as prose text leaves space for potential misinterpretations and ambiguities. Some semantic constraints are expressed rather fuzzily or their intention is hidden between the lines. For example, when explaining the correct use of decomposition hierarchies, semantic constraints deal with all acceptable decomposition and segment types on different levels. However, when ignoring or misunderstanding the meaning of these constraints, unintended decomposition structures are possible.

3.3 Inconsistent MPEG-7 Profile Descriptions

All MPEG-7 description tools are defined by the DDL, which is based on XML Schema. In addition, the semantic constraints of MPEG-7 profiles explain the correct use of these description tools for a specific application domain. Since XML Schema enables only limited possibilities for addressing semantics, the modeling capabilities of the DDL for considering semantic constraints are limited also. As a consequence, most semantic constraints cannot be reflected by the DDL. Thus, as described in the profile definition process (cf. Section 2.7.1), the semantic constraints are defined rather informal as prose text written in English. However, in contrast to a knowledge representation language based on a defined set of applicable terms and relationships, these textual constraints have no formal grounding.

By ignoring the semantic constraints, regardless of which reasons, the description tools can be used in an improper way. As a result, inconsistent MPEG-7 profile descriptions are created or processed, which are possibly leading to interoperability issues. In order to prevent this problem, a validation step is required. While checking the conformance of an MPEG-7 profile description is possible on a syntactical level against the corresponding XML Schema, the semantic constraints cannot be checked in an automated. This is the consequence of the informal textual representation of the semantic constraints, which is not machine-processable. Thus the proper application of the relevant semantic constraints has to be checked manually. However, this validation approach is time-consuming, error-prone, and inappropriate for some use cases. For example, in a metadata production or exchange chain, a fully automated validation step for MPEG-7 profile descriptions would be more desirable than a manual approach.

In the following, different types of semantic inconsistencies of MPEG-7 profile descriptions based on the rather informal textual representation of the related semantic constraints are analyzed. All these inconsistencies are caused by the limited machine-processability in terms of capturing and understanding the intended semantics. In fact,

the presented inconsistencies yield to syntactically valid MPEG-7 profile descriptions created with respect to the related profile XML Schemas. However, these descriptions interfere with the corresponding semantic constraints. The outcome of this analysis is to make aware the need for a formalization strategy of semantic constraints. Formalized semantic constraints that are processable in an automated fashion are a key factor to ensure interoperability of MPEG-7 profile descriptions.

3.3.1 Inconsistent Structural Description

Most of the semantic constraints are dealing with explanations how to describe the structure of multimedia content. Thus being able to process structural semantic constraints is a major requirement to ensure interoperability of MPEG-7 profile descriptions.

3.3.1.1 Inconsistent General Structure

This group of inconsistencies is caused by violating structural semantic constraints, which restrict the general setup of MPEG-7 profile descriptions. The structural setup is mostly influenced by addressing the allowable top-level types in an MPEG-7 profile description. These types are defined by an related profile XML Schema in terms of applicable XML elements, attributes, and valid arrangements. Thus a profile XML Schema includes several complex types that are derived from the same generic type or a chain of generic types. In case a cardinality restriction is set on a generic type, types that are based on this generic type are restricted the same way. As a result, defining different restrictions for complex types having the same base type is infeasible. Although the MPEG-7 standard enables extensions of the description tools, a profile represents a subset of these tools only. Thus defining additional complex types in order to address different restrictions in a separate manner is not legal in profiles.

For example, the DAVP allows the description of one single content per MPEG-7 profile description, which is either a video or an image. An optional summary can also be described along with the content description or separated in an own MPEG-7 profile description. These semantic restrictions are only partially reflected by the DAVP XML Schema. The relevant parts of this schema are shown in Listing 3.1. The applicable multimedia content types are restricted to `ImageType` and `AudioVisualType`, which are derived from the `MultimediaContentType`. All other multimedia content types, defined by the MPEG-7 standard, are excluded in the DAVP. In order to limit the description to one single content only, the cardinality of the element `MultimediaContent` is restricted to one, which is done in the definition of the `ContentEntityType`. This complex type is derived from the `CompleteDescriptionType` via `ContentDescriptionType`. Besides the `ContentDescriptionType`, additional complex types are derived from the `CompleteDescriptionType`, for instance to represent a summary of the content. The `CompleteDescriptionType` is used in the specification of the `Mpeg-7` element, which represents the root node of any MPEG-7 description, as part of the `Description` ele-

ment. The crucial point is that the cardinality of the `Description` element is set to two in order to permit the description of one multimedia content along with a summary. However, this cardinality restriction does not differentiate between the various derivations of the `CompleteDescriptionType`. As a result, the description of two content descriptions or two summaries is also possible, which contradicts the intention of the profile.

3.3.1.2 Inconsistent Decomposition Hierarchy

Another common use case of semantic constraints of MPEG-7 profiles is to provide explanations for describing decomposition hierarchies of the multimedia content. Such constraints are part of the DAVP, AVDP, and TRECVid profile. All possible decomposition options are addressed by semantic constraints, for instance allowing the description of several decomposition variants on different levels, which are based on different criteria or methods. The interpretation of a described decomposition hierarchy is ambiguous without the context information provided by semantic constraints. Thus leaving these semantic constraints unprocessed may cause inconsistent interpretations. Not knowing the semantics of a decomposition hierarchy may lead to misplaced decomposition and segment types. For instance, when specific types are missing, while they are required and vice versa. In addition, misplaced description tools used to describe additional information of involved segments, such as specific features, lead to inconsistencies.

An excerpt of an MPEG-7 description including an inconsistent decomposition hierarchy in terms of the DAVP is presented in Listing 3.2. The intention of this hierarchy is to describe the temporal segmentation of a video in order to represent a shot list including key frames. Therefore the DAVP contains several semantic constraints addressing the structure and semantics of the included decomposition and segment types. According to these constraints, the valid shot structure is based on the temporal decomposition of the related video into shots in the first place. Thus only segments denoted as shots can be part of this decomposition. Afterwards, the temporal decomposition of shots into key frames is described. Key frames must not be decomposed further at all. Furthermore, the temporal extent of shots are described by time point and duration information. Since a key frame cannot have a duration, time point information is provided only.

In the presented decomposition hierarchy, the particular decompositions into shots and key frames are denoted by special criteria values (`visual shots` and `key frames`). Shots and key frames are marked by `StructuralUnit` elements. The related values represent terms of a particular classification scheme (cf. [22, Section 5.1]). Several violations of the semantic constraints lead to an inconsistent hierarchy in terms of the DAVP. For example, the segment `shot1_1` is part of the temporal decomposition into shots. According to the related `StructuralUnit` element, this segment is denoted as key frame. However, a key frame must not be part of a temporal decomposition into shots, which is designated by the criteria value `visual shots`. The classification of segment `shot1_1` is also contradictory to the media information. For this segment

```
1   <element name="Mpeg7">
2     <complexType>
3       <complexContent>
4         <extension base="mpeg7:Mpeg7Type">
5           <choice>
6             <!-- allow 2 descriptions only in the case of multimedia
                  content + summary -->
7             <element name="Description" type="mpeg7:
                  CompleteDescriptionType" maxOccurs="2"/>
8           </choice>
9         </extension>
10      </complexContent>
11    </complexType>
12  </element>
13
14  <complexType name="ContentDescriptionType" abstract="true">
15    <complexContent>
16      <extension base="mpeg7:CompleteDescriptionType">
17        ...
18      </extension>
19    </complexContent>
20  </complexType>
21
22  <complexType name="ContentEntityType">
23    <complexContent>
24      <extension base="mpeg7:ContentDescriptionType">
25        <sequence>
26          <!-- allow just 1 Multimedia Content -->
27          <element name="MultimediaContent" type="mpeg7:
                MultimediaContentType"/>
28        </sequence>
29      </extension>
30    </complexContent>
31  </complexType>
32
33  <complexType name="ImageType">
34    <complexContent>
35      <extension base="mpeg7:MultimediaContentType">
36        ...
37      </extension>
38    </complexContent>
39  </complexType>
40
41  <complexType name="AudioVisualType">
42    <complexContent>
43      <extension base="mpeg7:MultimediaContentType">
44        ...
45      </extension>
46    </complexContent>
47  </complexType>
48 </schema>
```

Listing 3.1: DAVP XML Schema (partially shown).

a duration information is provided, which is not permitted for key frames. Another inconsistency is caused by segment `shot1_2_KF`. This segment is denoted as key frame and contains a `TemporalDecomposition` element. However, key frames must not be decomposed further. Furthermore, segment `shot1_3`, which is denoted as shot, leads also to an inconsistency. This segment contains a decomposition into shots, which is designated by the criteria `visual shots`, but a shot can not be decomposed into shots further. At this point, a temporal decomposition into key frames would be the only option. Additionally, another violation is caused by the missing media duration information of segment `shot1_2`.

These inconsistencies become obvious only when consulting the relevant semantic constraints, as these constraints cannot be reflected by the profile XML Schema of the DAVP. For example, in context of the presented decomposition hierarchy, the concepts of a video, shot, and key frame are based by the `VideoSegmentType`. An illustration of the XML Schema definition of this type is depicted in Figure 3.3. In this definition, several elements such as `MediaLocator`, `MediaDuration`, `TemporalDecomposition`, and `id` are defined as being optional. Depending on the semantic concept expressed by the `VideoSegmentType`, the presence or absence of these optional elements lead to inconsistencies. For example, a shot without a duration information would violate the semantics of the DAVP, while a key frame without a duration information is semantically valid.

3.3.2 Inconsistent Temporal Description

Providing appropriate temporal data types and structures for time representation is a crucial point when describing multimedia content having a temporal dimension. MPEG-7 provides description tools for representing time in general (e.g. a time stamp including time zone information) and audiovisual content in particular (e.g. time points and intervals of a video). The specification of these temporal description tools is based on syntactic patterns defined by the MPEG-7 XML Schema. When considering this schema only, semantically inconsistent time declarations in MPEG-7 descriptions are possible. In addition, a temporal decomposition is described by attributes, which summarize temporal aspects of the involved segments. The semantics of such a temporal description is expressed by textual guidelines in the MPEG-7 standard and not by a specific MPEG-7 profile definition. In order to ensure full interoperability of MPEG-7 profile descriptions, these constraints have to be considered also. In the following, inconsistent time representations and inconsistent summarization attributes of a temporal decomposition are explained.

3.3.2.1 Inconsistent Time Representation

The time representation in MPEG-7 is based on describing time points and time intervals (cf. Figure 3.4). Therefore different `TimePoint` and `Duration` data types are

[3] `http://www.altova.com/xml-editor/`

```
<VideoSegment id="TRECVID2005_1">
  ...
  <TemporalDecomposition criteria="visual shots">
    <VideoSegment id="shot1_1">
      <StructuralUnit href="...:StructuralUnitCS:vis.keyframe"/>
      <MediaTime>
        <MediaTimePoint>...</MediaTimePoint>
        <MediaDuration>...</MediaDuration>
      </MediaTime>
    </VideoSegment>

    <VideoSegment id="shot1_2">
      <StructuralUnit href="...:StructuralUnitCS:2005:vis.shot"/>
      <MediaTime>
        <MediaTimePoint>...</MediaTimePoint>
      </MediaTime>
      <TemporalDecomposition criteria="key frames">
        <VideoSegment id="shot1_2_KF">
          <StructuralUnit href="...:StructuralUnitCS:2005:vis.keyframe
              "/>
          <MediaTime>
            <MediaTimePoint>T00:00:03:26116F30000</MediaTimePoint>
          </MediaTime>
          <TemporalDecomposition criteria="key frames">
            ...
          </TemporalDecomposition>
        </VideoSegment>
      </TemporalDecomposition>
    </VideoSegment>

    <VideoSegment id="shot1_3">
      <StructuralUnit href="...:StructuralUnitCS:2005:vis.shot"/>
      <MediaTime>...</MediaTime>
      <TemporalDecomposition criteria="visual shots">
        <VideoSegment id="shot1_3_shot">
          <StructuralUnit href="...:StructuralUnitCS:2005:vis.shot"/>
          <MediaTime>
            <MediaTimePoint>...</MediaTimePoint>
            <MediaDuration>...</MediaDuration>
          </MediaTime>
        </VideoSegment>
        <TemporalDecomposition criteria="visual shots">
          ...
      </TemporalDecomposition>
      </TemporalDecomposition>
    </VideoSegment>
    ...
  </TemporalDecomposition>
</VideoSegment>
```

Listing 3.2: An excerpt of an inconsistent decomposition hierarchy violating the semantic constraints of the DAVP.

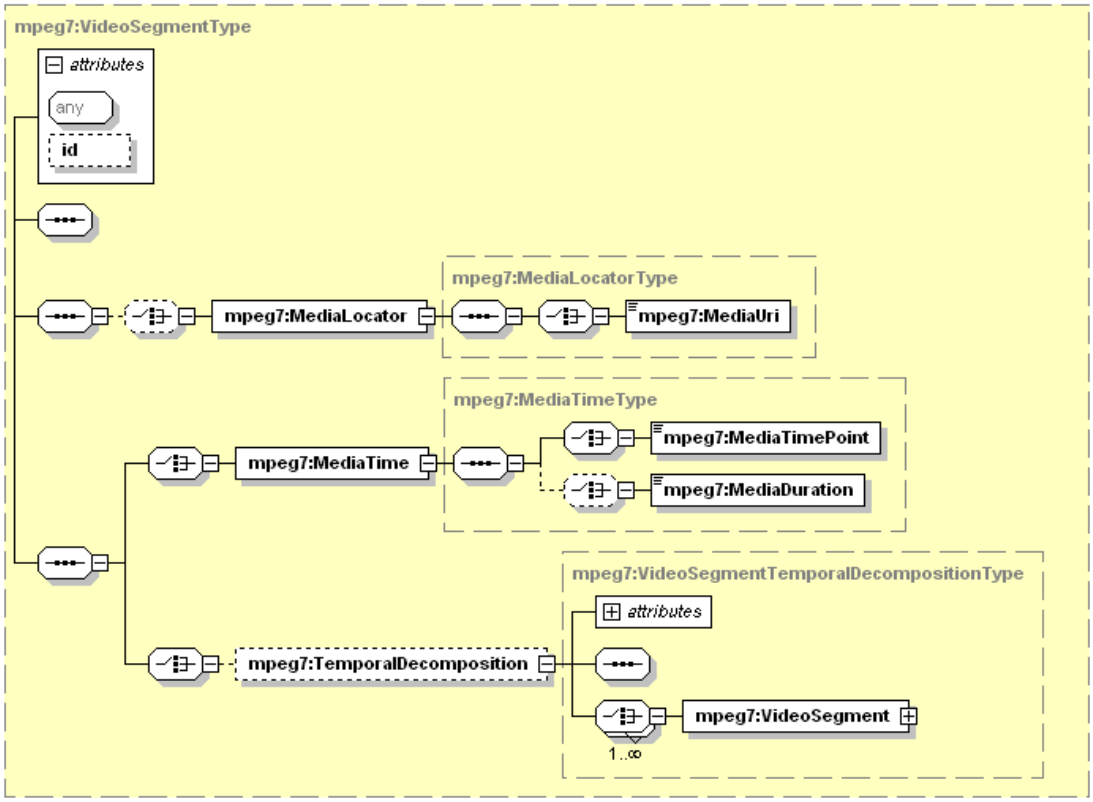

Figure 3.3: Illustration of the XML Schema definition of the `VideoSegmentType` (rendered by XMLSpy[3]).

provided. In Figure 3.4A, the simplest way for defining a time interval is shown. Here, the time point `t1` is the starting point of a time interval specified as half open interval (`[t1, t2[`). The length of this interval is defined by a duration information. In addition, a time point can be specified relatively with respect to a time base `t0` by using `RelTimePoint` and `timeBase` data types (cf. Figure 3.4B). Another option for representing a relative time point is depicted in Figure 3.4C. In this case, a time point, defined as `RelIncrTimePoint`, is represented by counting a reference time unit with respect to a time base. Finally, a time interval can be defined in the same manner by counting such a reference time unit with respect to a starting time point `t1`.

All time-related data types in MPEG-7 are based on ISO 8601 [60]. This standard is generally considered as the reference "specification of the representation of dates in the proleptic Gregorian calendar[4] and times and representations of periods of time" [60]. However, some derivations were made. In contrast to the ISO 8601 standard, the representation of decimal fraction of seconds is different in MPEG-7. While in ISO 8601 an arbitrary number of decimals followed by a comma (`","`) or full stop (`"."`) is used, for instance `0,125 s`, in MPEG-7, fractions of seconds are represented by counting a predefined interval. Therefore two decimal values are required. One for counting the

[4]The proleptic Gregorian calendar includes dates prior to 1582 (the year it came into use as an ecclesiastical calendar).

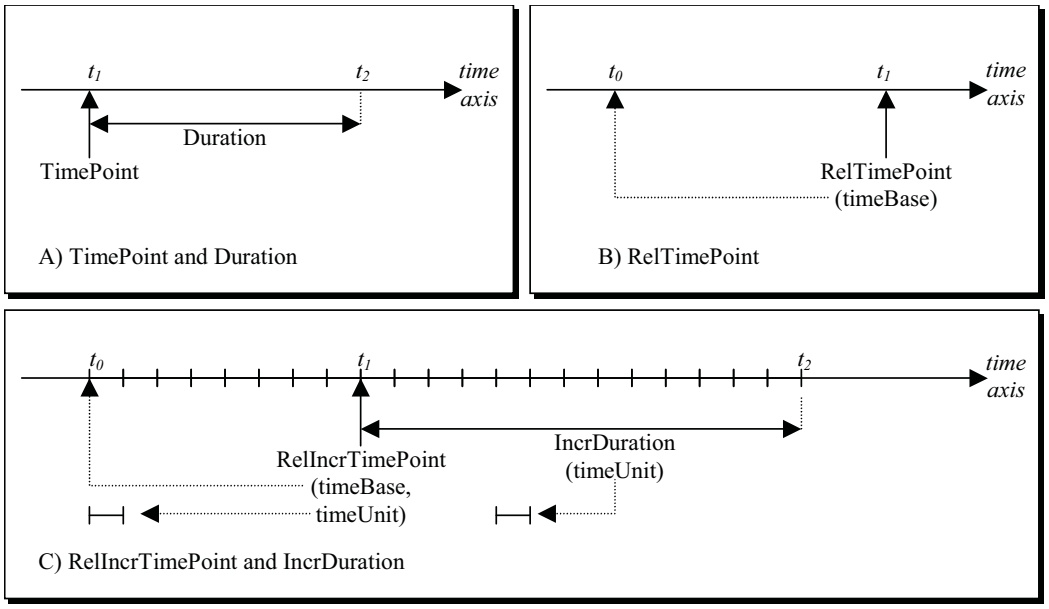

Figure 3.4: Describing time instants and time intervals in MPEG-7 (taken from [15]).

number of fractions `n` and one for counting the number of fractions of one second `N`. For example, `0.125 s` can be represented, among others, as `n=1, N=8` or `n=375, N=3000`. Representing fractions of seconds in this way is widely used in the audiovisual domain.

These time-related data types are defined by the MPEG-7 XML Schema as simple types. For example, the simple type defining the format of absolute media time points is shown in Listing 3.3. The main part of such a data type definition is a syntactic pattern. This pattern defines the structure and options of temporal values according to the presented ISO 8601-derived format. However, a pattern cannot grasp the semantics of time points or intervals fully. As a consequence, invalid time statements in temporal descriptions are possible. While a pattern specifies the number of digits for each time part mainly, restrictions of value ranges cannot be expressed. For example, `2014-14-32T44:66:01` denotes a completely valid media time point according to the definition of the underlying simple type (`mediaTimePointType`). However, according to ISO 8601, `14` is an invalid value for the month part, `32` is invalid for days, `44` is an invalid value for hours, and `66` is an invalid minute assertion. Since the specification of such basic valid value ranges is not supported, more complex restrictions of temporal values are completely out of scope. For example, excluding impractical dates such as the 31^{st} of November or 29^{th} of February 2014 in not possible using the MPEG-7 XML Schema only.

The representation of improper fractions of seconds is also possible by the current specification of the temporal data types in MPEG-7. An improper fraction is designated by a numerator that is higher than the denominator. Improper fractions should be avoided since they complicate the representation of time-related data. For example, `T00:00:00:27F25` represents a time point having an improper fraction, while a proper

```
<simpleType name="mediaTimePointType">
  <restriction base="mpeg7:basicTimePointType">
    <pattern value="(\-?\d+(\-\d{2}(\-\d{2})?)?)?
      (T\d{2}(:\d{2}(:\d{2}(:\d+)?)?)?)?(F\d+)?"/>
  </restriction>
</simpleType>
```

Listing 3.3: Definition of the `mediaTimePointType` (taken from MPEG-7 Multimedia Description Schemes [15]).

```
<simpleType name="mediaDurationType">
  <restriction base="mpeg7:basicDurationType">
    <pattern value="\-?P(\d+D)?(T(\d+H)?(\d+M)?(\d+S)?(\d+N)?)?(\d+F)?
      "/>
  </restriction>
</simpleType>
```

Listing 3.4: Definition of the `mediaDurationType` (taken from MPEG-7 Multimedia Description Schemes [15]).

time point is `T00:00:01:2F25`. Additionally, fractions having a denominator that is set to zero can be represented when consulting the MPEG-7 XML Schema only. However, a division by zero is not defined. Thus the denominator part of a fraction must not be `0` either.

The representation of duration data types is also based on patterns. For example, the pattern for defining the `mediaDurationType` is shown in Listing 3.4. The pattern of this data type has a similar setup as the one presented in Listing 3.3. Thus analog inconsistencies in the value representation are observed. Moreover, a negative duration value can be set by using the optional minus sign of this pattern. For example, the value `-PT1H` represents a negative time interval. Then the end time point of such interval appears to be earlier in time as the corresponding starting time point. As a result, negative time intervals would rather cause confusion and provide space for misinterpretations. Therefore negative values should be avoided when describing time intervals of multimedia content.

3.3.2.2 Inconsistent Description of a Temporal Decomposition

Besides possible inconsistencies related to values representing time points and intervals, inconsistencies related to the description of a temporal decomposition in MPEG-7 are also identified. In fact, the semantics of a temporal decomposition is clearly expressed in the textual guidelines of the standard. For any segment having a temporal dimension its temporal decomposition into further sub-segments can be described. For example, the temporal decomposition of a video into shots is described in this way. Additionally, the occurrence of gaps and overlaps within a temporal decomposition are addressed.

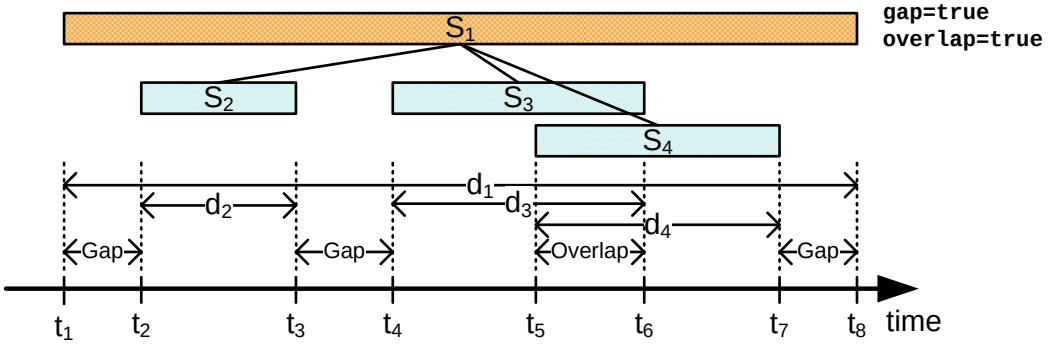

Figure 3.5: Temporal decomposition of segment S_1 into three segments (S_2, S_3, S_4) with gaps and overlap.

Therefore boolean values are used to represent these two properties of a temporal decomposition. According to MPEG-7, the property gap is marked as `true` when the time interval of the segment being decomposed is not fully covered by its sub-segments, and the property overlap is set to `true`, in case at least two sub-segments are overlapping in time.

An example of a temporal decomposition is visualized in Figure 3.5. Here, segment S_1 is decomposed into 3 sub-segments, namely S_2, S_3 and S_4. Each segment is denoted by a start point and a duration. For example, S_1 has start point t_1 and a duration d_1. Since gaps exist between t_1 and t_2, t_3 and t_4, and t_7 and t_8, the 3 sub-segments do not cover the time interval of segment S_1 fully. As a result, property `gap` of this temporal decomposition is set to `true`. In addition, property `overlap` is set to `true` since the sub-segments S_3 and S_4 are overlapping between t_5 and t_6. An example of a corresponding MPEG-7 description is partly shown in Listing 3.5. The segment being decomposed (S_1) is represented by the `Video` type, while the 3 sub-segments are represented by the `VideoSegment` type. For each segment, start time and duration information is provided. The `TemporalDecomposition` element denotes the actual segmentation. Therefore all sub-segments are enclosed by this `TemporalDecomposition` element. In order to describe the gap and overlap assertions by boolean values, the attributes `gap` and `overlap` are attached to the `TemporalDecomposition` element.

It is obvious, that a temporal decomposition of a segment into sub-segments is only meaningful when the time range filled by each of the sub-segments is at most the time range of the segment being decomposed. In other words, a sub-segment cannot start before or end after its corresponding parent segment. Otherwise an inconsistent description violating the semantics of a temporal decomposition is triggered. An example of a temporal decomposition containing invalid time intervals of sub-segments with respect to the segment being decomposed is depicted in Figure 3.6. Two segments, S_2 and S_3, are not within the expected time range defined by segment S_1. Segment S_2 is completely out of this time range, while segment S_4 is only partially within. Another inconsistent description of a temporal decomposition is caused by invalid gap and overlap assertions. Here, the calculation of actual gap and overlap values based

```
<Video id="S1">
  [...]
  <MediaTime>
   <MediaTimePoint>T00:00:00:0F1000</MediaTimePoint>
   <MediaDuration>PT00H0M30S0N30000F</MediaDuration>
  </MediaTime>
  <TemporalDecomposition gap="true" overlap="true">
    <VideoSegment id="S2">
      <MediaTime>
        <MediaTimePoint>T00:00:03:0F30000</MediaTimePoint>
        <MediaDuration>PT00H00M03S1000N30000F</MediaDuration>
      </MediaTime>
    </VideoSegment>
    <VideoSegment id="S3">
      <MediaTime>
        <MediaTimePoint>T00:00:10:0F30000</MediaTimePoint>
        <MediaDuration>PT00H00M06S0N30000F</MediaDuration>
      </MediaTime>
    </VideoSegment>
    <VideoSegment id="S4">
      <MediaTime>
        <MediaTimePoint>T00:00:13:0F30000</MediaTimePoint>
        <MediaDuration>PT00H00M06S0N30000F</MediaDuration>
      </MediaTime>
    </VideoSegment>
  </TemporalDecomposition>
 </Video>
```

Listing 3.5: Example MPEG-7 description representing the temporal decomposition shown in Figure 3.5.

on the time intervals of the sub-segments is contradictory to the attached values. s is contradictory to the attached values. In Figure 3.7, an example of inconsistent gap and overlap assertions is presented. Since segment S_3 overlaps with segment S_4, the actual `overlap` value should be set to `true`. However, the asserted `overlap` value is set to `false`, which does not reflect the actual decomposition. Similarly, the asserted `gap` value (`false`) is incorrect, since a gap between segments S_2 and S_3 does exist in the current decomposition.

The MPEG-7 XML Schema provides the structure, applicable descriptors, and data types for describing a temporal decomposition. However, detecting the presented inconsistencies of temporal decomposition cannot be accomplished by XML Schema. There is no mechanism to link the temporal information of all segments involved together in order to verify time ranges and gap and overlap assertions. As a result, the presented semantic inconsistencies of temporal decompositions may occur, while they remain syntactically valid.

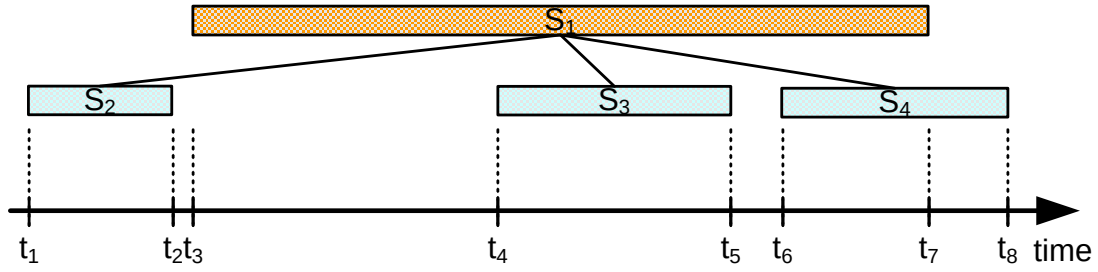

Figure 3.6: Invalid time intervals of sub-segments with respect to the segment being decomposed.

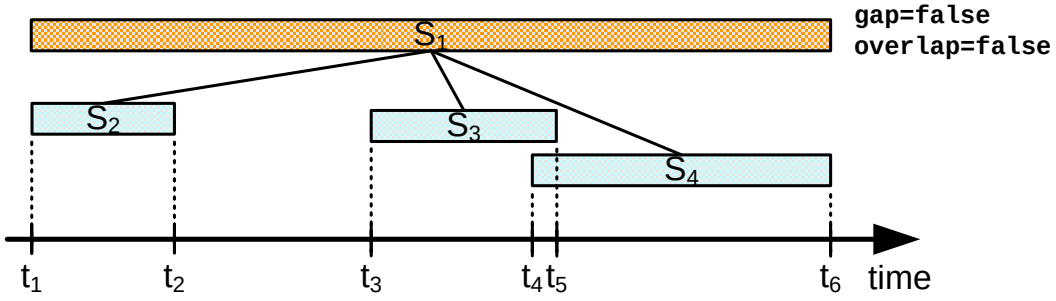

Figure 3.7: Invalid gap and overlap assertions.

3.3.3 Other Inconsistencies

Other inconsistencies are related to the description of the media information about the described multimedia content and the integration of terms referenced from classification schemes. The presence and cardinality of relevant elements in the resulting MPEG-7 descriptions can be controlled using XML Schema. However, additional semantic restrictions cannot be checked by consulting XML Schema only.

The description of media information can be specified at multiple places in an MPEG-7 description. In this way, information about the content type is contained in the `MultimediaContent` element and `MediaFormat` element. However, the declaration of the content type in these elements can mismatch. For example, it is possible to set the content type in the `MultimediaContent` element to an image`xsi:type="ImageType"`, while the content type in the `MediaFormat` element is set to a video. The consequence is an MPEG-7 description containing inconsistent media content types. Here, XML Schema cannot specify the required semantic restrictions between the involved elements.

The description of inconsistent modality information remains also unrecognized when considering XML Schema only. The `MediaProfile` describes the visual and audio encoding of the content. Therefore the encoding of multiple streams, for example, the master quality and low resolution preview, are described. This information must match the defined content type. Again, there is no way to check that the values are consistent. Different modalities can be included in a structural description, while the related media

information contains contradicting information about these modalities. For example, an MPEG-7 description contains the structural description of a video having two audio channels, while the related media information states that the described content is mono audio only.

Another group of inconsistencies is based on the misuse of classification schemes. In MPEG-7, a classification scheme is a generic mechanism for defining multilingual and controlled vocabularies. The set of terms and definitions belonging to a scheme is organized in a taxonomy. A term is identified by a specific URI, which is referenced by a respective element in the MPEG-7 description. MPEG-7 already defines some basic classification schemes, amongst others, for media types, file formats, genres, ratings, and roles. Some semantic constraints refer to the use of classification schemes. The appropriateness of a classification scheme in a certain context is a source of possible inconsistencies. For example, in case a classification scheme does not contain appropriate terms for a certain application context. Again, checking the correct application of these terms is out of scope of XML Schema.

3.4 Summary

The semantic constraints of the discussed MPEG-7 profiles are textual guidelines clarifying the use of these profiles in context of their intended application domains. Therefore the semantic constraints affect the use and combination of the applicable description tools. In addition, the semantics of these tools and the impact to resulting MPEG-7 profile descriptions are addressed.

Most of the semantic constraints being reviewed are related to the structural description of multimedia content. They influence the general setup of MPEG-7 profile descriptions as well as the segmentation of the content describing decomposition hierarchies, for example into time and space. Moreover, the presence and position of media and creation information in an MPEG-7 profile description is addressed. In addition, the use of cross references to external classification schemes and summaries is specified.

Expressing the semantic constraints as English prose leaves space for potential misinterpretations and differences of opinion. The textual semantic constraints have no formal grounding. Since XML Schema enables only limited possibilities for addressing semantics, the semantic constraints cannot be modeled either. When ignoring or being unable to handle the semantic constraints, for whatever reason, the description tools may be used in an improper way. As a result, inconsistent MPEG-7 profile descriptions being created or processed lead to potential interoperability issues. While checking the conformance of an MPEG-7 profile description is possible on a syntactical level by consulting the related XML Schema, the semantic constraints cannot be checked in an automated way currently. However, in a metadata production or exchange chain, a fully automated validation step for MPEG-7 profile descriptions is desirable in order to ensure interoperability. Currently a human intervention for checking the proper use of the semantic constraints in MPEG-7 profile descriptions is required.

Chapter 4

Relevant Technologies and Related Approaches

In this Chapter, relevant technologies in context of this thesis are introduced. These are the Resource Description Framework (RDF), the Web Ontology Language (OWL), and logical rules. In addition, the reasoning capabilities of OWL and logical rules are discussed. Finally, related approaches in context of this thesis are presented and compared.

4.1 Resource Description Framework

The Resource Description Framework (RDF)[1] is a framework for modeling and exchanging data on the web in a formal way. This framework consists of a number of standards[2] defined by the World Wide Web Consortium (W3C)[3]. The initial version of this framework denoted as RDF 1.0 [62] was released in 2004. In February 2014, RDF 1.1 [34] containing minor changes was published.

For short, using RDF enables the formal description of data about different kinds of things. In terms of RDF, the things to be described are denoted as resources. Furthermore, any resource following a specific identification mechanism is addressable by RDF. While Uniform Resource Identifiers (URIs) [26] are used for this purpose in RDF 1.0, they were replaced by Internationalized Resource Identifiers (IRIs) [38] in RDF 1.1. An IRI is a generalization of an URI having an increased language support. Expressible data for resources comprises application-specific metadata and semantic relationships between resources. For example, metadata dealing with title, creator, and copyright information about a web page can be modeled. In this case the web page is a describable resource by RDF. However, the usage of RDF is not at all limited to web resources like web pages. More precisely, none of the resources being described by RDF

[1] http://www.w3.org/RDF/
[2] http://www.w3.org/standards/techs/rdf#stds
[3] http://www.w3.org/

have to be directly accessible on the web. Even a book in a library or a real person can be described.

The syntax and semantics of RDF are defined formally based on a graph model and a data-modeling vocabulary. RDF-based data is machine-processable and exchangeable between different applications and systems without any loss of meaning [62]. Furthermore, RDF-based data originating from different sources can be linked together in order to gather additional information about resources.

4.1.1 RDF Data Model

The RDF data model is based on expressing statements about resources. Each RDF statement, also denoted as RDF triple, represents one fact about a given resource. An RDF statement consists of the resource being described along with a property and an associated property value. The property represents the kind of information, like the type of metadata or a specific relationship, to be expressed about a resource, while the associated property value represents the actual value of this information. Formally, any RDF statement consists of one subject, one predicate, and one object. Here, the subject is the resource being described, the predicate denotes the property, and the object is the associated property value. Subjects and predicates must always be resources. A resource is usually identified by an IRI. An alternative for representing resources are blank nodes. A blank node corresponds to a variable in terms of algebra having local scope only. The object is either a resource or a literal. A literal is a simple data type, such as a string or a integer. Since literals are different from resources, they cannot act as subjects. As a consequence, no statements about literals can be made.

The RDF data model defines the graphical representation of RDF statements. Therefore a directed graph of two nodes and one labeled arc between them is used to specify one RDF statement. The two nodes represent the subject and the object, while the arc between represents the predicate. The arrowhead of this arc points from the subject node to the object node. A ellipse is used to designate a node based on a resource, while a rectangle is used for a literal. An example of a graphical representation of an RDF statement is depicted in Figure 4.1. In this example, the resource `http://musiclub.web.cern.ch/MusiClub/bands/cernettes/pictures/LHC5.jpg` is described by the property `dc:format` and the associated property value `mm:typeJPEG`. The namespace prefixes `dc` and `mm` are used in order simplify IRIs in RDF statements.

An RDF statement represents one binary relation between a subject and object. Statements can be merged together forming a linking structure. In order to describe multiple facts about a resource, several RDF statements about the same resource are made . A resource can be the subject of a statement, while being the object of another one. Therefore expressing structured information, like a person's address, is possible. When required, a defined set of statements can be grouped and identified by an distinct IRI. This feature was introduced in RDF 1.1 and is called named graph [34].

The graphical representation of RDF statements may be good for humans but it

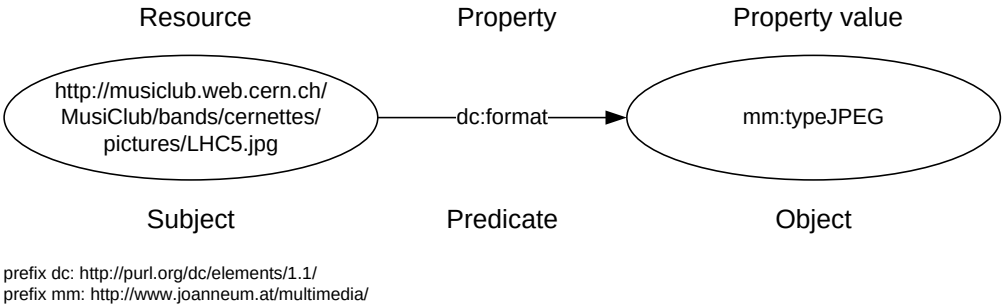

Figure 4.1: Graphical representation of an RDF statement.

```
1 @prefix dc:       <http://purl.org/dc/elements/1.1/> .
2 @prefix mm:       <http://www.joanneum.at/multimedia/> .
3 @prefix musiccern: <http://musiclub.web.cern.ch/MusiClub/bands/
     cernettes/pictures/> .
4
5 musiccern:LHC5.jpg dc:format mm:typeJPEG .
```

Listing 4.1: Turtle serialization of the RDF graph depicted in Figure 4.1.

is not suitable in terms of computer processing. The RDF model defines no concrete format for encoding RDF data per se. However, different serialization formats for RDF data are available, in order to ensure machine readability and processability. The following serialization formats are defined by the RDF working group[4]: RDF/XML [23], Turtle [24], N-Triples [33], TriG [27], and N-Quads [32]. In addition, a special syntax for embedding RDF data into HTML and XML documents (RDFa [52]) and a JSON[5]-based syntax (JSON-LD [77]) are available. For example, two different serialization options of the RDF statement presented in Figure 4.1 are shown in Listing 4.1 (Turtle syntax) and Listing 4.2 (RDF/XML syntax). Turtle is more readable for human users, while the RDF/XML syntax is more common for machine-processability since it is based on XML.

The RDF data model defines how to make statements about resources. Therefore the composition and representation of a statement is defined. In this context, the role of a subject, object, and predicate is specified. However, nothing is said about the semantics of the resources, properties, and property values acting as subjects, objects, and predicates in RDF statements. Without additional information it is impossible to grasp the meaning of a statement. For example, the information provided by RDF data model about the statement depicted in Figure 4.1 is, that `dc:format` is a property describing the resource `http://musiclub.web.cern.ch/MusiClub/bands/-cernettes/pictures/LHC5.jpg` by value `mm:typeJPEG`. Additional information about the semantics of property `dc:format` or the kind of the described resource cannot be

[4] `http://www.w3.org/2011/rdf-wg/wiki/Main_Page`
[5] JavaScript Object Notation, `http://json.org/`

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
  <rdf:Description rdf:about="http://musiclub.web.cern.ch/MusiClub/
     bands/cernettes/pictures/LHC5.jpg">
    <dc:format rdf:resource="http://www.joanneum.at/multimedia/
       typeJPEG"/>
  </rdf:Description>
</rdf:RDF>
```

Listing 4.2: RDF/XML serialization of the RDF graph depicted in Figure 4.1.

provided by the data model.

4.1.2 RDF Schema

The RDF data model defines the role of the subject, object, and predicate in RDF statements. However, expressing the semantics of applied resources and properties is out of scope. RDF Schema (RDFS) [29] is a Vocabulary Description Language for formally defining application-specific properties and relations. RDFS is regarded as the semantic extension of RDF. The term RDF Schema is used to designate the modeling language itself and RDFS-based vocabularies also.

RDFS enables a type system based on the definition of classes and properties. A class refers to the concept of a type or category. Members of a class are also denoted as instances. Furthermore, a property represents an attribute which is applied to instances of classes. In addition, the use of properties in connection to certain classes can be restricted. In contrast to an object-oriented programming language, properties are not defined as being members of classes. Instead, applicable classes are listed for each property using domain and range restrictions. Furthermore, classes and properties can be organized in a hierarchical fashion. All classes, properties, and instances are modeled as resources.

RDFS itself and resulting RDFS vocabularies are modeled as RDF Data. In order to define an RDFS vocabulary, special resources and properties are provided by RDFS. They are grouped together and referenced by two namespaces. The corresponding namespace prefixes are `rdfs`[6] and `rdf`[7]. For example, a class is defined by resource `rdfs:Class` and instances of classes are created using property `rdf:type`. In addition, a property is defined by resource `rdf:Property`. Domain and range restriction for properties can be set using the properties `rdfs:domain` and `rdfs:range`. Furthermore, specialization relations between classes are expressed by property `rdfs:subClassOf` respectively `rdfs:subPropertyOf` for properties.

A new RDFS vocabulary is defined by creating new statements using these re-

[6] `http://www.w3.org/2000/01/rdf-schema#`
[7] `http://www.w3.org/1999/02/22-rdf-syntax-ns#`

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix mm: <http://www.joanneum.at/multimedia/> .

mm:StillImage rdf:type rdfs:Class .

mm:StillImageJPEG rdf:type rdfs:Class ;
                  rdfs:subClassOf mm:StillImage .

mm:MimeType rdf:type rdfs:Class .

dc:format rdf:type rdf:Property ;
          rdfs:domain mm:StillImage ;
          rdfs:range mm:MimeType .
```

Listing 4.3: RDFS example encoded in Turtle.

sources and properties. In Listing 4.3, the statements specifying the relations between still pictures and corresponding MIME types [43], especially for pictures encoded in the JPEG[8] format is shown. Therefore three classes are defined. One for representing all types of images (`mm:StillImage`), all JPEG pictures (`mm:StillImageJPEG` is a sub-class of `mm:StillImage`), and all MIME types (`mm:MimeType`). The existing property `dc:format` is also included. This property is a Dublin Core [36] metadata element defined in the corresponding RDF Schema[9]. In the presented example, the usage of `dc:format` is restricted. This property should be used in statements where the subject is an instance of class `mm:StillImage` and the object is an instance of class `mm:MimeType`. Therefore the properties `rdfs:domain` and `rdfs:range` are used. To define a vocabulary, special RDFS resources, such as `rdf:type`, `rdfs:Class`, and `rdfs:subClassOf`, are used.

The presented vocabulary provides the semantic background for the RDF statement shown in Figure 4.1. Assuming that the subject of this statement refers to a JPEG image. Then, this subject is also denoted as an instance of class `mm:StillImageJPEG`. The resulting RDF statements shown in Figure 4.2 reveal additional information. Since `mm:StillImageJPEG` is a sub-class of `mm:StillImage`, the described JPEG image is also an instance of this class. Additionally, based on the range restriction of `dc:format`, the resource `mm:typeJPEG` is an instance of class `mm:MimeType`.

RDFS has the ability to describe class or property hierarchies and to restrict the usage of properties globally. However, expressing the union, intersection, or disjointness of classes is not supported by RDFS. For example, suppose that the RDFS vocabulary shown in Listing 4.3 defines the concept of GIF[10] encoded images. Then it makes sense

[8] `http://www.jpeg.org/jpeg/`
[9] `http://dublincore.org/schemas/rdfs/`
[10] Graphics Interchange Format: `http://www.w3.org/Graphics/GIF/spec-gif89a.txt`

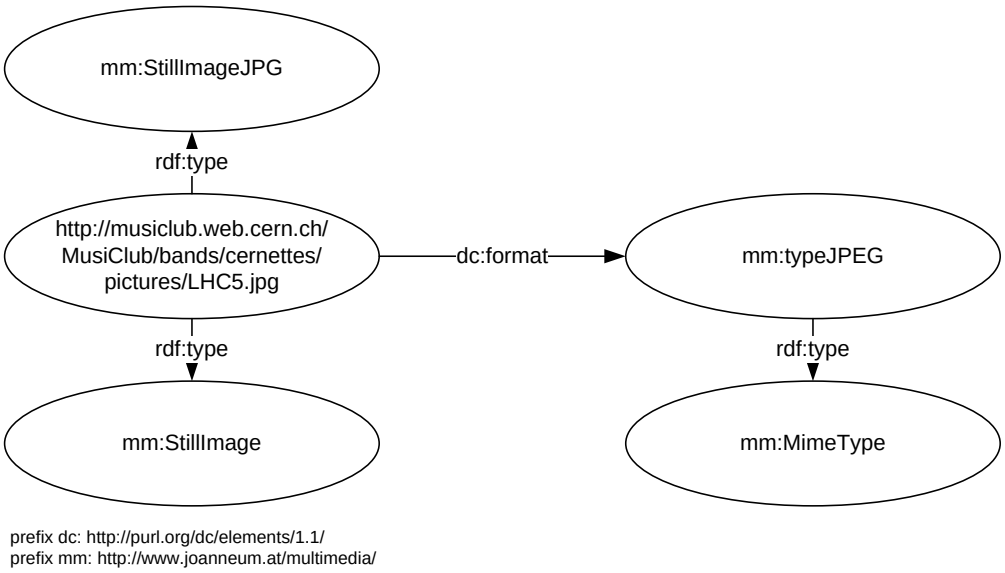

Figure 4.2: Example of applying an RDFS vocabulary.

to define that a JPEG image is not a GIF image and vice versa. In summary, expressing the required disjointness is not supported by RDFS. Moreover, there is no mechanism to define restrictions for properties on a local basis or set cardinality restrictions. Therefore it is impossible to describe that a sill image is described by exactly one related MIME type (via property `dc:format`) only.

4.1.3 Querying RDF Data

In order to perform queries on RDF data, several query languages are defined, like RQL [61], SeRQL [30], and SPARQL [49]. However, this Section deals with SPARQL only, since it is the most widely adopted one. An extensive overview about query languages is presented in [44].

The term SPARQL is an recursive acronym and stands for SPARQL Protocol And RDF Query Language. SPARQL 1.0, the initial version, was published by the W3C in 2008, while SPARQL 1.1 became a W3C Recommendation in March 2013. Besides being a query and update language for RDF data, SPARQL is also a protocol [40]. This protocol defines how to transfer SPARQL queries between a client and a query processor.

However, SPARQL is a query language primarily. The source of a query is an RDF dataset which is a collection of RDF graphs. Since SPARQL only enables to query for plain RDF data, additional vocabulary information must be included in an RDF dataset for taking into consideration. Very much like SQL[11] for relational databases, SPARQL allows to express queries to find matching RDF triples, using variables and even resembling SQL syntax. A SPARQL query is based on graph pattern matching

[11] Structured Query Language, ISO/IEC 9075

```
1 PREFIX mm: <http://www.joanneum.at/multimedia/>
2
3 SELECT ?instance
4 WHERE {
5   ?instance a mm:StillImageJPEG . }
```

Listing 4.4: Example SPARQL select query.

being applied to a given RDF dataset for finding query solutions. A graph pattern consists of one or more triple patterns. A triple pattern is like an RDF statement containing placeholders. The syntax for defining triple patterns is based on Turtle. Any RDF statement matching the triple pattern is a query solution.

Four different query forms are defined by SPARQL, denoted by the keywords `SELECT`, `CONSTRUCT`, `DESCRIBE`, and `ASK`. A `SELECT` query returns resources and literals matching the variables of the graph pattern in a table format. Different query result formats are available, such as plain text, XML, or JSON. The result of the `CONSTRUCT` query is an RDF graph. Therefore the triple patterns act as a template for the resulting RDF graph. Using an `ASK` query returns whether a query solution exists for a stated pattern. The `DESCRIBE` form returns descriptive RDF data about resources matching the graph pattern. The resulting content is based on decisions made by the SPARQL query processor what is descriptive in this context than returning resources matched by the graph pattern itself. Besides these four query forms, SPARQL Update [46] provides the functionality to insert and delete statements in existing RDF data sets.

An example of a `SELECT` query is shown in Listing 4.4. Variables in triple patterns are designated by a question mark (`?`). This query returns all instances of class `mm:StillImageJPEG`. In the `SELECT` clause all variables included in the result are listed. Then the graph pattern is defined within the `WHERE` clause. In the example, the graph pattern consists of one triple pattern only. The subject of this pattern is represented as variable, namely `?instance`. The triple pattern describes a `rdf:type` relation, between variable `?instance` and the class `mm:StillImageJPEG`. Therefore any resource being an instance of class `mm:StillImageJPEG` substitutes `?instance` and is a query solution. For demonstration, this query is applied to the RDF data presented in Figure 4.2. The corresponding query result encoded in XML is shown in Listing 4.5. Since there is one instance of class `mm:StillImageJPEG` in the RDF dataset only, one substitution for variable `?instance` was found only.

The presented query example is based on one triple pattern only. However, more complex queries can be defined by SPARQL. A graph pattern may include various triple patterns. Triple patterns can be grouped and defined as being optional or alternative. SPARQL supports constraints, such as numerical restrictions, and references to named graphs. An RDF dataset may contain a reference to an other graph. Such a graph is denoted as named graph. Finally, a query solution can be modified in various ways, using the keywords `OFFSET`, `LIMIT`, and `ORDER BY`.

```
<?xml version="1.0"?>
<sparql
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xs="http://www.w3.org/2001/XMLSchema#"
    xmlns="http://www.w3.org/2005/sparql-results#" >
  <head>
    <variable name="instance"/>
  </head>
  <results ordered="false" distinct="false">
    <result>
      <binding name="instance">
        <uri>http://musiclub.web.cern.ch/MusiClub/bands/cernettes/
           pictures/LHC5.jpg</uri>
      </binding>
    </result>
  </results>
</sparql>
```

Listing 4.5: Example SPARQL query result expressed in the SPARQL Query Results XML Format.

4.2 Web Ontology Language

Due to the semantic limitations of RDFS, a more powerful language was developed by the W3C, namely the Web Ontology Language (OWL) [56]. OWL is a semantic markup language for defining ontologies in order to enhance the machine interpretability of web-based content. According to [50], "an ontology defines (specifies) the concepts, relationships, and other distinctions that are relevant for modeling a domain." In the broadest sense, OWL can be regarded as an semantic extension of RDFS.

4.2.1 Variants

The first version of OWL, OWL 1 [37], became a W3C Recommendation in 2004. In 2006, an extended version, OWL 1.1 [70], was published as a W3C Member Submission. This submission was the initial point for the specification of the current version, OWL 2 [53], in 2008. The evolution of OWL, including the functionality introduced by OWL 2 such as extended data types and annotations, additional properties, enhanced restrictions, improved metamodeling, and property chains, is described in [47]. OWL 2 is fully backwards compatible to OWL 1.

Discriminated by their expressive power, three different sub-languages of OWL are defined by OWL 1 that are also supported respectively updated by OWL 2. These sub-languages are OWL Full, OWL DL, and OWL Lite. OWL Full, the most expressive one, is extending the semantics of RDFS with additional OWL constructs [75]. Thus the semantics of OWL Full is expressed as RDF statements, which is denoted as RDF-based

semantics. In contrast to OWL Full, OWL DL is based on description logics [19]. More precisely, OWL 1 DL corresponds to the description logic variant $\mathcal{SHOIN}(\mathcal{D})$ [58] and OWL 2 DL corresponds to $\mathcal{SROIQ}$ [57]. The semantics of OWL DL [66], denoted as direct semantics, is natively defined by an abstract syntax defined by the OWL 2 Structural Specification [67]. A mapping between this abstract syntax and the RDF graphs is defined. OWL Lite is a subset of OWL DL by restricting or excluding the use of specific OWL constructs. Reducing this formal complexity should enable an easier understanding and use of this sub-language.

Beside these three sub-languages, three profiles of OWL are defined by OWL 2 [47]. These profiles are OWL 2 EL, OWL 2 QL, and OWL 2 RL. OWL 2 EL is based the based on description logic $\mathcal{EL}$++ [18] and OWL 2 QL is based on the DL-Lite family [31]. OWL 2 RL can be implemented as a set of rules. A profile aims to a specific use case or application scenario. For example, OWL 2 EL is suitable for describing large-scale ontologies, while OWL 2 QL and OWL 2 RL are appropriate to be used with data already stored in relational databases respectively expressed as RDF triples.

4.2.2 Ontology Modeling

In this Section, the essential ontology modeling capabilities of OWL are summarized. Furthermore, serialization formats of OWL ontologies are presented and the RDFS example introduced in the previous Section is rewritten using OWL.

OWL describes a domain of discourse by defining different constructs or axioms. Thus defining classes, instances of classes (individuals), properties, and restrictions is supported by OWL. Expressing sub-class and sub-property relations is possible in order to gain a more refined ontology definition. In addition, operations of the set theory can be applied to classes. For instance, describing the union or intersection of classes or the complement of a class is possible. The disjointness of classes can also be described. For example, if two classes are stated to be disjoint, an individual cannot be an instance of both classes at the same time. Furthermore, the characteristics of a property can be described more precisely. Among others, a property may be transitive, symmetric, reflexive, or functional. In addition, a property can be defined as the inverse of another property or be disjoint with another property. Besides domain and range instructions of properties, additional property restrictions can be stated in order to express a more detailed and restricted class definition. Thus property restrictions are used to define restrictions on properties of classes in order to model certain characteristics of the individuals belonging to these classes. Therefore universal, existential, cardinality, and value restrictions are applied. For example, an existential property restriction of a class defines that any instance of this class must be described by the involved property and property value. Furthermore, describing the equivalence between classes, properties, and instances is also possible. It is important to note that OWL is based on the open world assumption, meaning that something which is not explicitly stated to be true is unknown and thus cannot be assumed to be false per se. In contrast to the open world assumption, following the closed world assumption means that something that is not

```
Class: mm:StillImage
    SubClassOf:
        owl:Thing,
        dc:format some mm:MimeType,
        dc:format exactly 1 owl:Thing
    DisjointWith: mm:MimeType

Class: mm:StillImageJPG
    SubClassOf:
        mm:StillImage,
        dc:format value mm:imageJPEG
    DisjointWith: mm:StillImageGIF

Class: mm:StillImageGIF
    SubClassOf:
        mm:StillImage,
        dc:format value mm:imageGIF
    DisjointWith: mm:StillImageJPG

Class: mm:MimeType
    DisjointWith: mm:StillImage

ObjectProperty: dc:format
    Domain:  mm:StillImage
    Range:   mm:MimeType

Individual: mm:typeJPEG
    Types: mm:MimeType, owl:Thing

Individual: mm:typeGIF
    Types: mm:MimeType, owl:Thing
```

Listing 4.6: Excerpt of the OWL ontology refining the RDFS vocabulary shown in Listing 4.3 (encoded in Manchester OWL syntax).

defined due to missing information is regarded as being false.

An OWL ontology is stored and shared using different formats. OWL 1 ontologies are mainly represented by RDF graphs. In addition, a functional-style syntax [67] and an XML syntax [65] are defined by OWL 2. Any OWL 1 expressed as RDF graphs is a valid OWL 2 ontology. Besides these variants, the Manchester Syntax [55] is available. This frame-based syntax is widely used and supported by ontology editors, such as protégé[12] and TopBraid Composer[13].

The RDFS vocabulary shown in Listing 4.3 is refined as OWL ontology, which is depicted in Listing 4.6. This ontology includes the same classes and properties as defined in the RDFS vocabulary. Furthermore, two instances of the class mm:MimeType are de-

[12] http://protege.stanford.edu
[13] http://www.topquadrant.com/tools/ide-topbraid-composer-maestro-edition/

fined, namely mm:typeJPEG and mm:typeGIF. In addition, the classes mm:StillImage and mm:MimeType are defined as being disjoint. An individual cannot represent an MIME type and an image at the same. Similar, a JPEG image is not a GIF image. Thus the related sub-classes of class mm:StillImage are also defined as being disjoint. Furthermore, property restrictions are stated in classes representing images. The existential restriction in combination with the cardinality restriction of class mm:StillImage define that any instance of this class is described by exactly one MIME type (via property dc:format). The value restriction of property dc:format in class mm:StillImageJPG assigns the MIME type mm:typeJPEG to any individual of this class. In the same manner, the value restriction in class mm:StillImageGIF assigns the MIME type mm:typeGIF to related individuals. As shown in this example, using the disjointness construct and property restrictions, a more refined domain description is possible by OWL compared to RDFS.

4.3 Reasoning

In terms of knowledge representation, reasoning means deriving new facts from a knowledge base of asserted axioms or facts in an automated way. Thus reasoning makes implicit information, which is already available in the knowledge base, explicit. In this Section, the reasoning capabilities of OWL and logical rules are discussed.

4.3.1 OWL-Based Reasoning

Applying reasoning to OWL ontologies derives new facts, which are not explicitly expressed in the ontology. The open world assumption is still valid when performing reasoning on OWL ontologies. In the following, useful reasoning tasks related to OWL ontologies are presented [76]. For each presented task, an example is provided, which is related to the OWL ontology described in Listing 4.6.

Consistency task: This task detects contradictions in an ontology. Therefore the consistency of instances with respect to the defined ontology constructs is validated. The result of such consistency check is represented by a boolean value. For example, assigning an individual as an instance of the classes mm:StillImageJPG and mm:StillImageGIF at the same time causes an inconsistency since these two classes are defined as being disjoint.

Satisfiability task: This task checks the satisfiability of an ontology. An ontology is regarded as satisfiable if instances can be created of all defined classes. An unsatisfiable class leads to an inconsistent ontology. For example, the class mm:StillImageJPGIF defined as the intersection of the classes mm:StillImageJPG and mm:StillImageGIF is unsatisfiable. Any instance of this class must also be an instance of the classes mm:StillImageJPG and mm:StillImageGIF. However, this is not possible since they are defined as being disjoint.

Classification task: Classification infers the class hierarchy of an ontology. Therefore sub-class relations that are not explicitly stated are computed by determining all general classes of a given class. For example, when defining the new class `mm:StillImageAnimatedGIF`, which represents animated GIF files, as a sub-class of class `mm:StillImageGIF`, the classification task infers that this new class is also a sub-class of the class `mm:StillImage`.

Realization task: The memberships of individuals to classes are computed by the realization task. Thus it can be checked if a given individual belongs to a specific class or to infer all individuals of a given class. For example, realization infers that the individual `mm:typeJPEG`, which is explicitly assigned to class `mm:StillImage-JPG`, is also an instance of class `mm:StillImage` due to the stated sub-class relation between the classes `mm:StillImage` and `mm:StillImageJPG`. As a consequence, the ontologies developed as part of the described work are modeled by OWL DL.

When applying reasoning, computational completeness and decidability are important aspects to be considered. Completeness means that all logical consequences of an ontology must be guaranteed to be computed, while decidability ensures that all required computations of the reasoning process are finished in finite time. As stated in [47], there exists currently no reasoner that is considering the complete implementation of OWL Full. Moreover, OWL Full constructs can be used without any restrictions. As a result, OWL Full is undecidable [64]. In contrast to OWL Full, OWL DL is decidable since it is based on Description Logics, which is a decidable fragment of first order logic (FOL) [19]. In addition, complete OWL DL reasoners are available, for example Pellet[14] and RacerPro[15].

4.3.2 Rule-Based Reasoning

Rule-based reasoning enables the derivation of new facts by applying a set of rules denoted as rule base to a data base. In scope of this thesis, a data base is represented as ontology. Newly derived facts are stored in the data base and are considered for further derivations.

The rules to be applied are defined by logic programs (LP) [48], which is a knowledge representation formalism. The relation between logic programs and first order logic is depicted in Figure 4.3. Description logic is a decidable subset of first order logic. Logic programs do partially overlap with first order logic and also description logic, which is denoted as description logic programs (DLP). Features of logic programs that are not covered by first order logic are negation as failure and procedural attachments. Negation as failure is a non-monotonic feature to infer that an assertion is false when failing to compute that this assertion is true. Thus rules are a way to perform reasoning of OWL DL ontologies under the closed world assumption.

[14] `http://clarkparsia.com/pellet/`
[15] `http://franz.com/agraph/racer/`

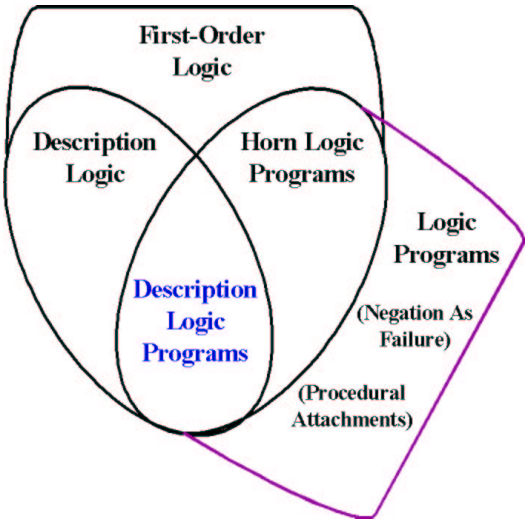

Figure 4.3: Relation between logic programs and first order logic (extracted from [48]).

```
[UncleRule:
(?x :isBrotherOf ?y),
(?y :isParentOf ?z),
->
(?x :isUncleOf ?z)]
```

Listing 4.7: Example rule to infer the family relationship uncle expressed as Jena rule. For simplicity, namespace specifications are omitted.

An ordinary logic program is a set of rules each having the form:

$$H \leftarrow B_1 \wedge \ldots \wedge B_m \wedge \sim B_{m+1} \wedge \ldots \wedge \sim B_n$$

where

- H and B_i are *atomic formulae (atoms)*,
- $\sim$ is a logical connector called *negation as failure*,
- $\leftarrow$ is to be read as *if*, so that the overall rule should be read as "[head] if [body]",
- and $n \geq m \geq 0$.

The left-hand side of the rule is called head, while the right-hand side is called body of the rule. Synonyms for the head of a rule are conclusion and consequent, the body is also denoted as premise or antecedent. No restrictions are placed on the arity of the predicates appearing in the included atomic formulae. In addition, variables and functions may appear unrestrictedly in these formulae.

For example, the logical rule to infer someone's uncle based on family relations is shown in Listing 4.3. This rule is expressed by the Jena rules syntax[16] and is applied to RDF-based data. The meaning of this rule reads as follows: "If x is the brother of y who is the father of z, then x is the uncle of y." Due the limited expressivity in OWL 1 DL, this rule cannot be expressed. In contrast, in OWL 2 DL it is possible to express the concept of uncle using property chaining. Related property chain axioms allow to infer the existence of a property from a chain of properties. However, inference based on the closed world assumption cannot be realized by property chaining.

4.4 Related Approaches

A validation service for MPEG-7 descriptions[17] was created by the National Institute of Standards and Technology (NIST)[18]. This service validates the conformance of a given MPEG-7 description syntactically to the MPEG-7 standard by performing an XML Schema validation. Possible syntax violations are reported to the user. The MPEG-7 standard is considered by this validation service only, specific MPEG-7 profiles are not supported. According to the validation service website, the integration of semantic checks is planned. However, the latest available version of this validation service does not include semantic validation options so far.

Several attempts were made to represent the MPEG-7 description tools as ontologies. In this context, automatic mappings from the MPEG-7 XML Schema to OWL covering the whole standard are proposed [45, 80]. However, the resulting ontologies are unable to formalize semantic constraints not represented by XML Schema without extensive re-engineering work. In addition, other attempts were made to manually model an MPEG-7 ontology. While one ontology is either restricted to the upper level elements and types of MPEG-7 [59], the other one is tailored to a very specific use of the standard for a particular application [17]. All these ontologies could be used as an alternative for modeling the semantic constraints. However, neither specific MPEG-7 profiles are considered by these ontologies nor transformations of MPEG-7 profile descriptions to ontology-based representations are provided currently.

In contrast to these approaches, the approach described in this thesis does not completely express the MPEG-7 description tools by an OWL ontology. Instead, ontologies in combination with logical rules are used to represent the semantic constraints of MPEG-7 profiles defined in natural language that cannot be expressed and validated using XML Schema only.

This thesis is the continuation of the work presented in [78], which describes the initial formalization approach for a subset of the semantic constraints of the Detailed Audiovisual Profile. In this thesis the validation approach is refined and additional MPEG-7 profiles are also considered. Furthermore, the validation of temporal semantic

[16] http://jena.apache.org/documentation/inference/index.html#rules
[17] http://m7itb.nist.gov/M7Validation.html
[18] http://www.nist.gov/index.html

constraints is integrated also.

4.5 Summary

In this Chapter, the technologies needed for the realization of the proposed approach were introduced. The Resource Description Framework (RDF) is a framework for modeling and exchanging data in a formal way. The RDF Schema (RDFS) is a Vocabulary Description Language based on RDF. Due to the semantic limitations of RDFS, the Web Ontology Language (OWL) was developed.

Reasoning is applied to ontologies to infer new facts, which are not explicitly expressed, in an automated way. In similar manner, logical rules are applied to a rule base in order make implicit knowledge explicit. Rules are described by an "if-then" pattern. OWL ontologies are based on the open world assumption, while rules follow the closed world assumption.

Several attempts were made to represent the MPEG-7 description tools as ontologies. However, these ontologies are unable to formalize semantic constraints not represented by XML Schema without extensive re-engineering work. Neither specific MPEG-7 Profiles are considered nor transformations of MPEG-7 profile descriptions to ontology-based representations are provided.

Chapter 5

Formalizing Semantic Constraints

The presented semantic constraints of MPEG-7 profiles raise interoperability issues of MPEG-7 profile descriptions (cf. Chapter 3). These issues are based on the informal specification of the semantic constraints in prose text. In order to ensure the interoperability of MPEG-7 descriptions, an automated validation process verifying the conformance to the semantics of a given MPEG-7 profile is desirable. Therefore the textual semantic constraints have to be represented formally in a machine-processable way.

This Chapter presents the proposed approach for formalizing the semantic constraints of MPEG-7 profiles. For simplicity, if not essentially required, namespace specifications are omitted in the explanations and related Figures and Listings.

5.1 General Approach

The proposed approach for formalizing semantic constraints of MPEG-7 profiles is based on the application of Semantic Web technologies, in particular, ontologies and logical rules. Both the text-based profile semantic constraints and the temporal semantic constraints are considered by this formalization approach. The semantic constraints to be formalized were discussed in Section 3.1. The temporal semantic constraints have to be formalized as well in order to address potential temporal inconsistencies. These potential inconsistencies were presented in Section 3.3.2. The temporal semantic constraints are profile independent since these constraints are defined by the MPEG-7 standard without referring further to any MPEG-7 profile specification.

In Figure 5.1, an overview of the general approach is depicted. First, the characteristics of the different MPEG-7 profiles are identified by considering the MPEG-7 profile XML Schemas and related textual semantic constraints. Based on this identification process, profile ontologies are modeled. One ontology is defined for each MPEG-7 profile. Additionally, the temporal characteristics are modeled by a separate temporal ontology. All these ontologies are created by hand.

Then validation rules are created with respect to these ontologies and the textual

semantic constraints. The purpose of these rules is to detect and flag violations of the semantic constraints of a given MPEG-7 profile description. The validation rules are represented by logical rules. In detail, the Jena rules syntax[1] including predefined builtin primitives is used for defining the validation rules. In addition, customized builtin primitives are created. These primitives are written in Java. One set of validation rules is created for each MPEG-7 profile. Beside these profile validation rules, separate rules for the validation of temporal semantic constraints are also created.

This approach for formalizing the semantic constraints is the basis for VAMP, which is a semantic validation service for MPEG-7 profile descriptions. The design and the implementation of VAMP are discussed in Chapter 6.

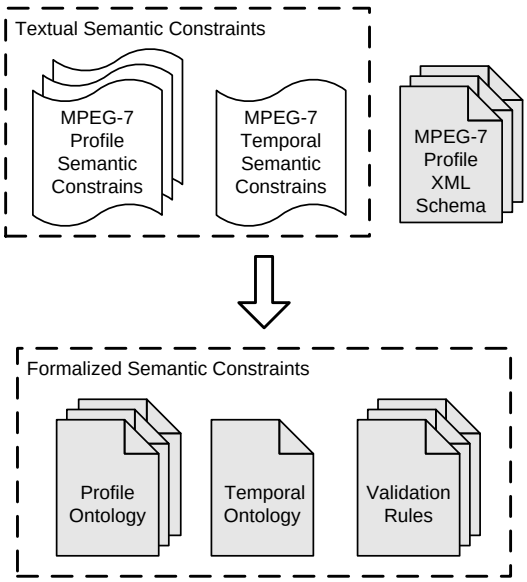

Figure 5.1: General approach for formalizing semantic constraints.

In the following, the presented general approach is discussed in more detail. The formalization of profile-specific semantic constraints is explained in Section 5.2, while the formalization of temporal semantic constraints is described in Section 5.3.

5.2 Formalizing Profile-Specific Semantic Constraints

Profile-specific semantic constraints are formalized by a profile ontology and validation rules. A profile ontology reflects the structural and semantic characteristics of a profile. Based on the profile ontology and the textual semantic constraints validation rules are created. This approach is used to validate the conformance of a given MPEG-7 profile description against the corresponding semantic constraints (cf. Figure 5.2). First, the profile ontology is instantiating with respect to the MPEG-7 profile description. An automated way for creating this ontological representation of an MPEG-7 profile

[1] http://jena.apache.org/documentation/inference/index.html#rules

description with respect to the corresponding profile ontology is discussed in Section 6.3. However, for describing the formalization approach it is not relevant how this ontological representation is created. Afterwards, validation rules are applied to this representation in order to detect and flag violations of the semantic constraints in the selected MPEG-7 profile description. These validation rules are executed by a rule reasoner as part of the rule reasoning process.

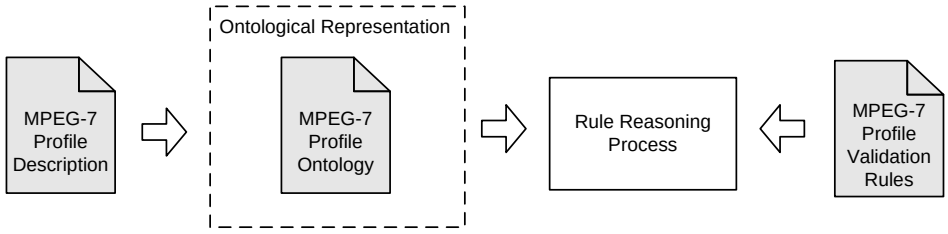

Figure 5.2: Validation of the conformance of an MPEG-7 profile description against profile-specific semantic constraints.

In the next two Sections, the modeling of the profile ontology and the creation of the profile validation rules are discussed in detail.

5.2.1 Modeling a Profile Ontology

A profile ontology models the structure and semantics of an MPEG-7 profile. The profile ontology is expressed by OWL DL in order to provide maximum expressiveness, while ensuring computational completeness and decidability at the same time. In contrast to other work [17, 45, 59, 80], the profile ontology does not completely map the MPEG-7 description tools to an ontology, but represents the structure and semantics of an MPEG-7 profile with respect to the included semantic constraints. First, the description tools that are addressed by profile semantic constraints are identified. Then, these tools are modeled as classes and relevant relations between these tools are modeled as properties. The use of properties can be constrained to specific classes according to the semantic constraints by using universal, existential, cardinality, and value restrictions.

Most of the profile semantic constraints of the MPEG-7 profiles taken into account are related to the structural description of the content. Hence most of the classes and properties of a profile ontology are reflecting structural description tools. The general setup of an MPEG-7 profile description is modeled by the ontology. Therefore acceptable occurrences and combinations of description tools are expressed. Furthermore, the segmentation of the content is addressed in a detailed way. For this purpose, the profile ontology reflects the allowable decomposition structure including applicable decomposition types and segments. Thus attributes of the decomposition types, criteria values, and references to structural units are also modeled. In addition, the correct presence of time and media information in the structural description is defined.

For example, one structural semantic constraint of the DAVP defines the decomposition of a shot into key frames. Such a decomposition is denoted by the crite-

ria value *key frames* and may include key frames only. This constraint, described in Section 3.3.1.2, cannot be checked by the MPEG-7 profile XML Schema only since shots and key frames are expressed by the same type (`VideoSegmentType`). In Listing 5.1, the exemplary classes and properties for formalizing this constraint are shown. First the concepts of a shot and key frame are defined as disjoint sub-classes of class `Segment`. Hence an instance of class `Shot` can not be an instance of class `Keyframe` at the same time and vice versa. Class `TemporalDecompositionIntoKeyframes` represents the concept of a temporal decomposition including key frames only. This class is a sub-class of class `TemporalDecomposition`. In order to express relations between a shot, a decomposition, and included key frames, two object properties are defined. Property `hasSegment` expresses the relation between a temporal decomposition and a segment being part of this decomposition. Therefore the domain of this property is set to class `TemporalDecomposition`, while the range is set to class `Segment`. Property `hasTemporalDecompositionIntoKeyframes` links any segment with a related temporal decomposition including key frames. The domain of this property is class `Segment` and the range is class `TemporalDecompositionIntoKeyframes`. Additionally, property restrictions based on these properties are added to some class definitions. While property `hasTemporalDecompositionIntoKeyframes` is used for defining a universal restriction in class `Shot`, property `hasSegment` is used as part of a universal restriction in class `TemporalDecompositionIntoKeyframes`. The universal restriction in class `Shot` defines that a shot can only have `hasTemporalDecompositionIntoKeyFrames` relations to a decomposition including key frames, while the universal restriction in class `TemporalDecompositionIntoKeyframes` denotes that only key frames may be included in such a decomposition (via property `hasSegment`). Moreover, as defined by a value restriction in class `TemporalDecompositionIntoKeyframes`, any instance of this class is characterized by the data property `hasCriteria` and the corresponding value `keyframes`.

5.2.2 Creating Profile Validation Rules

After creating the ontological representation of the MPEG-7 description to be validated, validation rules are applied. The idea of the profile validation rules is to detect and flag violations of the semantic constraints in the ontological representation of the MPEG-7 profile description. In case a violation is detected by a rule, an error notification is flagged. In case no violation is found, a success notification cannot be propagated by the same rule instantly since an "if-then-else" construct is not supported by rules. However, rules are based on the closed world assumption. Thus the non-existence of an error notification leads to the non-existence of the violation being validated by the related rule.

The body of a profile validation rule contains statements for detecting a violation of a semantic constraint. Therefore these statements form a pattern for defining a violation with respect to the classes and properties defined by the related profile ontology. For this purpose, variables and possible builtin primitives are used by the pattern. The

```
Class: Shot
    SubClassOf:
        Segment,
        hasTemporalDecompositionIntoKeyframes only
           TemporalDecompositionIntoKeyframes
    DisjointWith: Keyframe

Class: Keyframe
    SubClassOf: Segment
    DisjointWith: Shot

Class: TemporalDecompositionIntoKeyframes
    SubClassOf:
        TemporalDecomposition,
        hasSegment only Keyframe,
    hasCriteria value "keyframes"

ObjectProperty: hasSegment
    Domain: TemporalDecomposition
    Range: Segment

ObjectProperty: hasTemporalDecompositionIntoKeyframes
    Domain: Segment
    Range: TemporalDecompositionIntoKeyframes

DataProperty: hasCriteria
    Range: xsd:string
```

Listing 5.1: Exemplary classes and properties for formalizing a structural semantic constraint of the DAVP (encoded in Manchester OWL syntax).

validation rules detect wrongly placed or missing relations in this ontological representation. For example, the violation of a universal restriction defined for a specific property in a class definition can be detected. Such a restriction limits the acceptable values of a property in context of a class. By using the builtin primitive `noValue`, related instances having invalid property values are detected. Moreover, using the builtin primitive `noValue` in the body of a rule enables the execution of closed world checks. Then a missing property relation defined by an existential restriction of a class can be detected.

In case a violation is detected by a rule, an error notification is directly attached to the resource that is causing the violation. Such a resource is always an instance of a class of the ontological representation of the MPEG-7 profile description to be validated. For designating an error notification, the class `Error` and the corresponding property `hasError` are introduced. The subject of the property `hasError` is the resource causing the violation, while the object is an instance of class `Error`. In order to denote different types of validation errors, several instances of class `Error` are defined.

An example of a profile validation rule is depicted in Listing 5.2. This rule detects and flags any segment that is not a key frame, but still part of a temporal decomposition into key frames. The formal grounding of this rule is the ontology depicted in Listing 5.1. In particular, the rule considers the universal restriction of property `hasSegment` in the definition of class `TemporalDecompositionIntoKeyframes`. The body of this rule is based on three statements including variables and the predefined builtin primitive `noValue`. In the first statement, the variable `?decomposition` is a placeholder for an instance of class `TemporalDecompositionIntoKeyframes`. A segment included in this decomposition is represented by variable `?segment`. This relationship is expressed by the property `hasSegment` in the second statement of the exemplary rule. The third statement, which is based on the builtin primitive `noValue`, is used to check if a certain segment is not an instance of class `Keyframe`. In case all three statements apply, a segment violating the semantic constraint that a key frame can be part of a temporal decomposition into key frames only is detected. As a consequence, the head of this rule this triggered. Here, the segment being detected in the body of the rule (`?segment`) is flagged by an error notification. This notification is expressed by adding an additional statement to the ontological representation for designating the type of error. Therefore the segment to be flagged is the subject of this statement, while the object is property `hasError`. The related object value is `MisplacedSegmentInTemporalDecomposition-IntoKeyframes`, which is an instance of class `Error`.

```
1 [MisplacedSegmentInTemporalDecompositionIntoKeyframes:
2   (?decomposition rdf:type TemporalDecompositionIntoKeyFrames),
3   (?decomposition hasSegment ?segment),
4    noValue(?segment rdf:type Keyframe)
5    ->
6   (?segment hasError
      MisplacedSegmentInTemporalDecompositionIntoKeyframes)
7 ]
```

Listing 5.2: Validation rule for designating improper segments in a temporal decomposition into key frames.

As described, the approach for formalizing profile semantic constraints is based on the definition of a profile ontology and profile validation rules for each profile. In this approach the profile ontology is not directly used by a ontology reasoner as part of the validation task. This ontology is rather serving as the basis, beside the textual semantic constraints, for creating the profile validation rules only. This approach is chosen for multiple reasons. First, the ontology reasoning process stops at the first inconsistency being encountered. Then possible further inconsistencies would not be detected unless this first inconsistency is corrected. Second, not all violations of the semantic constraints can be expressed and detected by the use of an ontology only. For example, by using an ontology reasoner it is impossible to check existential or minimal cardinality constraints on relations that are actual not existing in a ontological

representation. The reason therefore is that an OWL ontology uses the open world assumption. It cannot be assumed that all information is known about all instances of the domain being described. In this context a missing statement does not lead to an inconsistency since it could be stated somewhere else, for example in another document. Therefore using logical rules is more suitable for the validation task. They are based on the closed world assumption assuming that all information needed is available in order to draw consequences. Then a missing statement can be detected by using the `noValue` builtin primitive. In this way, existential or minimal cardinality constraints can be validated fully for both existent and non-existent relations.

5.3 Formalizing Temporal Semantic Constraints

Preventing inconsistent temporal descriptions is another requirement in order to establish interoperability of MPEG-7 profile descriptions. Inconsistent temporal decompositions of segments are caused by time range violations and invalid gap and overlap assertions. Therefore temporal semantic constraints have to be formalized and considered as well by an automated validation service for MPEG-7 profile descriptions.

5.3.1 Validating Time Data

As described in Section 3.3.2.2, the acceptable formats for representing time points and intervals are defined by simple patterns, which are employed in the data type definitions of the MPEG-7 XML Schema. Due to limitations of the underlying pattern syntax, the specification of meaningless time data is possible. However, correct time data is a prerequisite when checking the semantics of temporal decompositions. Otherwise the temporal decomposition check might be based on faulty data and may produce unreliably outcome.

In order to check the values of time points and durations in an MPEG-7 profile description, the same validation approach as described for profile-specific constraints is used. Therefore validation rules are created to detect and flag inconsistent time information. The corresponding rules contain custom builtin primitives for detecting inconsistent time representations. In case such a builtin primitive detects a violation, an appropriate error notification is created. Alternatively, such a value check can be integrated in another rule as a preprocessing step. For example, when performing a value conversion, the value check is executed first. Then an invalid time value is detected by finding the nonexistent properties that would be describing the conversion result in the normal case.

For example, a validation rule for detecting and designating an invalid media time point is shown in Listing 5.3. Assuming that appropriate classes and properties for describing a media time point exist in an ontological representation, the first two statements of the rule are used to identify the value of a media time point to be checked. Such a value is represented by the variable `?mediaTimePointString`. This variable is the

argument of the custom builtin primitive `detectInvalidMediaTimePoint`. This builtin primitive contains different value checks. In case an inconsistent value representation is detected, an error notification for the media time point is created.

```
[MediaTimePointViolation:
  (?mediaTimePoint rdf:type MediaTimePoint),
  (?mediaTimePoint hasString ?mediaTimePointString)
  detectInvalidMediaTimePoint(?mediaTimePointString)
  ->
  (?mediaTimePoint hasError InvalidMediaTimePointFormat)
]
```

Listing 5.3: Example rule for validating the data format of media time points.

5.3.2 Formalizing the Semantics of a Temporal Decomposition

For formalizing the semantics of a temporal decomposition, an ontology models the characteristics of temporal segments and decompositions in terms of MPEG-7. This ontology is called temporal ontology[2]. Furthermore, an appropriate combination of rule and ontology reasoning steps is applied to detect inconsistencies of temporal decompositions presented in Section 3.3.2.2. Therefore validation rules are applied in order to verify the gap and overlap attributes and invalid parent-child relations of a temporal decomposition. The results of this rule reasoning process are classified using a validation hierarchy, which is also part of the temporal ontology.

In contrast to the formalization approach of profile semantic constraints, the approach for formalizing the semantics of a temporal decomposition is slightly different. While the profile ontology provides the classes, properties and restrictions for describing the characteristics of an MPEG-7 profile but it is not directly used for detecting violations of semantic constraints, the temporal ontology is explicitly used for the classification task by an ontology reasoner. Therefore the reasoning process consists of the application of both rule reasoning and ontology reasoning steps.

In the following, the different parts of this approach are discussed in more detail. Afterwards, the temporal validation workflow is presented.

5.3.2.1 Modeling Temporal Segments

The temporal ontology models the characteristics of temporal segments and decompositions. Therefore the extent of a temporal segment is modeled. For segments being decomposed further, also the relations to their sub-segments are described. Finally, gap and overlap attributes of a temporal decomposition are modeled.

[2]namespace: `http://mpeg-7.joanneum.at/semantics/temporal#`

```
Class: Segment
    SubClassOf:
        owl:Thing,
        hasStartPointEnumerator exactly 1 xsd:long,
        hasStartPointDenominator exactly 1 xsd:long,
        hasDurationEnumerator exactly 1 xsd:long,
        hasDurationDenominator exactly 1 xsd:long.

Class: ParentSegment
    EquivalentTo:
        Segment and (hasChild some Segment)
    SubClassOf:
        hasAssertedGap exactly 1 xsd:boolean,
        hasAssertedOverlap exactly 1 xsd:boolean,
        hasCalculatedGap exactly 1 xsd:boolean,
        hasCalculatedOverlap exactly 1 xsd:boolean,
        hasInvalidChild exactly 1 xsd:boolean.
```

Listing 5.4: Excerpt of the definition of Class `Segment` and `ParentSegment` of the temporal ontology (encoded in Manchester OWL Syntax).

The classes and properties needed for modeling temporal segments and their decompositions are partly shown in Listing 5.4. Class `Segment` denotes the basis for describing temporal segments. Thus every temporal segment is represented as an instance of this class. The temporal extent for instances of class `Segment` is described by start time point and duration value. Time-related data types in MPEG-7 are based on ISO 8601. However, fractions of seconds are represented as a fraction number instead of a floating number (cf. Section 3.3.2.1). In contrast to MPEG-7, in the temporal ontology, a time value is always expressed as one fraction number of seconds. This approach is chosen in order to simplify time-based comparison and validation procedures. In order to implement this approach, four properties for expressing the enumerator and denominator parts of time points and durations are available. These properties are `hasStartPointEnumerator`, `hasStartPointDenominator`, `hasDurationDenominator`, and `hasDurationEnumerator`. Exactly one temporal extent for every instance of class `Segment` is described by using these properties.

Additionally, class `ParentSegment` represents all segments being decomposed. This class is as sub-class of class `Segment`. In order to express the relationship between segment and its sub-segments based on a temporal decomposition, a parent-child relation is needed. The required relation is expressed by property `hasChild`. This property is applicable in statements having an instance of class `ParentSegment` as subject and an instance of class `Segment` as object. Class `ParentSegment` is modeled as defined class. As a result, any instance of class `Segment` having at least one `hasChild` relation can also be classified as `ParentSegment`.

Finally, the properties `hasAssertedGap` and `hasAssertedOverlap` are available to represent gap and overlap assertions of a temporal decomposition. Therefore boolean

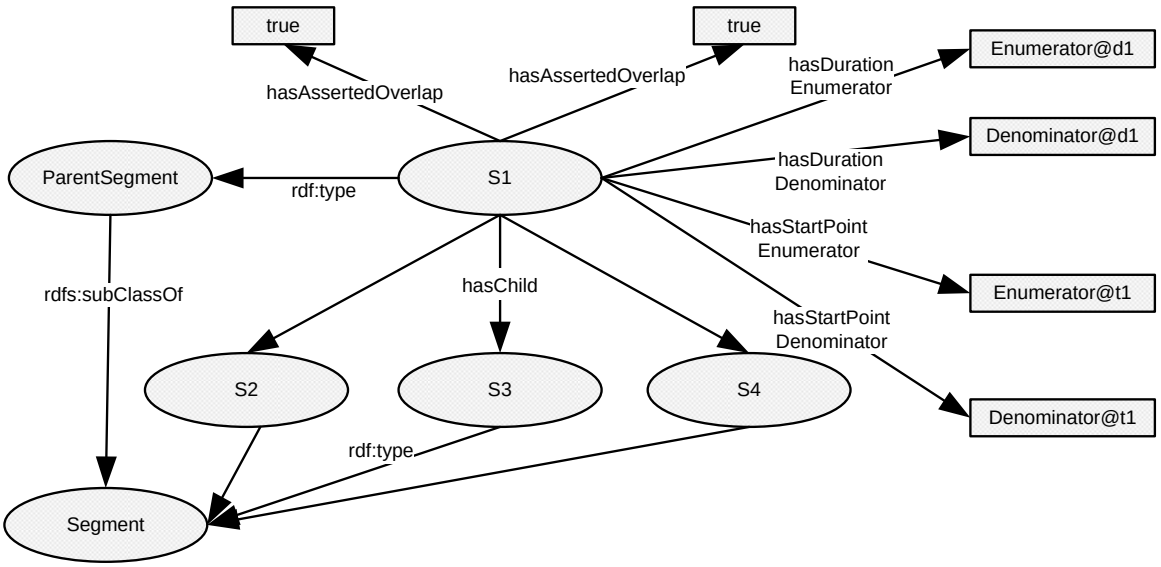

Figure 5.3: Representing the temporal decomposition shown in Figure 3.5 by the temporal ontology (partly shown).

values are used as property values in order to denote whether a temporal decomposition has a gap or overlap respectively. The subject to be used with these properties is any instance of class `ParentSegment`. The occurrence of these properties in context of class `ParentSegment` is restricted to one.

Applying the presented classes and properties enable the ontological representation of temporal segments and their decompositions. In this context, only one decomposition variant per segment can be described. For example, the ontology representation of the temporal decomposition depicted in Figure 3.5 is partly shown in Figure 5.3. In the original example, the segment S_1 is decomposed into segments S_2, S_3, S_4 having gaps and a overlap. Using the temporal ontology, segment S_1 is expressed as instance of class `ParentSegment` (`S1`), while the remaining segments are instances of class `Segment` (`S2`, `S3`, and `S4`). In order to describe the sub-segment relation between `S1` and the segments `S2`, `S3`, and `S4`, property `hasChild` is used. In addition, the related temporal properties for describing the fraction number of a temporal value are used to reflect the start time point and the duration of the segments. For simplicity this is shown for `S1` only in Figure 3.5. Finally, the gap and overlap information is represented for `S1`.

For the validation process, additional properties of class `ParentSegment` are required. Thus the properties `hasCalculatedOverlap` and `hasCalculatedGap` are defined in order to validate the correctness of the gap and overlap assertions. These properties represent the actual values of the gap and overlap attributes. The values are calculated based on the ontological representation of the temporal segments. In addition, the boolean property `hasInvalidChild` is introduced. During the validation process, this property indicates whether a parent segment contains invalid sub-segments or not. An invalid sub-segment is based on an improper temporal extent related in context of its parent segment.

5.3.2.2 Calculating Actual Gap and Overlap Attributes

In order to compare the asserted and actual values of the gap and overlap attributes, the actual values have to be computed first. However, the algorithm for calculating these values cannot be expressed by the temporal ontology. The boolean value of `hasInvalidChild` cannot be computed either. In order to calculate the boolean values of the properties `hasCalculatedOverlap`, `hasCalculatedGap`, and `hasInvalidChild` logical rules are used. These rules are based on the temporal ontology and the results provided by the rule reasoning process are directly reflected in the ontology.

Different rules are responsible to flag whether a gap or overlap relation exists. Therefore the temporal extents of the parent segment and the corresponding sub-segments are considered. For calculating the actual gap and overlap values the temporal extents of the sub-segments are compared pairwise in order to detect gaps and overlaps. In the related rules, the temporal comparisons are processed by builtin primitives. In addition, a trailing or leading gap between the parent segment and its sub-segments is observed in a similar manner. In case at least one gap or overlap is found by these rules, the associated property (`hasCalculatedOverlap` or `hasCalculatedGap`) is set to `true`. Otherwise another set of rules assign the values `false` to these properties. However, these rules are applied in a second rule processing step after the rules for finding gaps and overlaps are processed.

The rules for detecting invalid parent-child relations are implemented in a similar way. Apart from the fact that sub-segments are compared with the corresponding parent segment only. The relevant rules are shown in Listing 5.5. First, the rule `parent_has_invalid_child_true` is applied. Here each sub-segment is compared with its parent segment on a temporal basis. Therefore the fraction values of the start time points and end time points of these segments are compared by the custom builtin primitive `parentHasInvalidChildMPEG7`. The required end time point is computed by a separate rule before. In case a violation is detected, the statements in the head of the rule are added to the ontology. Here, one statement denotes that at least one invalid parent-child relation exists in the observed decomposition (via `hasInvalidChild`), while the other designates the segment responsible for such a violation (via `isInvalidChild`). In case no violations are found, rule `parent_has_invalid_child_false` returns the value `false` for property `hasInvalidChild`. This rule is performed in a separate rule processing step to prevent a potential race condition with rule `parent_has_invalid_child_true`.

5.3.2.3 Modeling Validation Hierarchy

The idea of the validation hierarchy is to provide additional classes representing the results of the gap, overlap, and invalid child validation process. Since the relevant properties needed for the validation process are defined as restrictions of class `ParentSegment`, the validation hierarchy is also defined in context of this class. In detail, classes of the validation hierarchy are modeled as sub-classes of class `ParentSegment`.

```
[parent_has_invalid_child_true:
  (?parent rdf:type ParentSegment),
  (?parent hasChild ?child),
  (?parent hasStartPointEnumerator ?parent_sp_enumerator),
  (?parent hasStartPointDenominator ?parent_sp_denominator),
  (?parent hasEndPointEnumerator ?parent_ep_enumerator),
  (?parent hasEndPointDenominator ?parent_ep_denominator),
  (?child hasStartPointEnumerator ?child_sp_enumerator),
  (?child hasStartPointDenominator ?child_sp_denominator),
  (?child hasEndPointEnumerator ?child_ep_enumerator),
  (?child hasEndPointDenominator ?child_ep_denominator)
  parentHasInvalidChildMPEG7(?parent_sp_enumerator, ?
     parent_sp_denominator, ?parent_ep_enumerator, ?
     parent_ep_denominator,  ?child_sp_enumerator,  ?
     child_sp_denominator,  ?child_ep_enumerator, ?
     child_ep_denominator)
  ->
  (?parent hasInvalidChild "true"^^xsd:boolean),
  (?child isInvalidChild "true"^^xsd:boolean)
]

[parent_has_invalid_child_false:
  (?parent rdf:type ParentSegment),
  noValue(?parent hasInvalidChild "true"^^xsd:boolean)
  ->
 (?parent hasInvalidChild "false"^^xsd:boolean)
]
```

Listing 5.5: Rules for computing the value of property `hasInvalidChild`.

Therefore additional classes are added in the temporal ontology.

An excerpt of the validation hierarchy is depicted in Figure 5.4. For simplicity, the classes needed for overlap verification are not depicted. However, they are modeled the same way as the validation classes for the gap verification. The bottom classes in this hierarchy represents all possible variants of a validation result. For example, when comparing the boolean values of the gap validation, four different results are possible. The asserted and calculated values can be the same (either both `true` or `false`). The corresponding validation classes of this two variants are `PSAssertedGapTrueCalculatedGapTrue` and `PSAssertedGapFalseCalculatedGapFalse`. Different boolean values are represented by the classes `PSAssertedGapTrueCalculatedGapFalse` and `PSAssertedGapTrueCalculatedGapFalse`. Additionally, the classes representing a negative respectively positive validation result are grouped together. Thus class `PSGapValidationPassed` represents all positive validation results while `PSGapValidationFailed` comprise all negative validation results. These classes are sub-classes of class `PSWithoutInvalidChild` expressing all instances of class `ParentSegment` without any invalid parent-child relations. In case a parent segment containing at least one invalid

child relation is detected, this parent segment is classified as instance of class `PSWith-InvalidChild`.

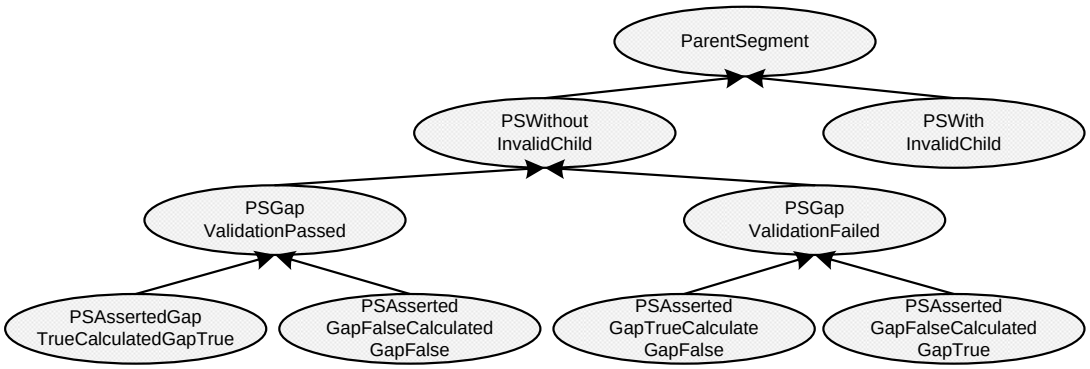

Figure 5.4: Example validation hierarchy including classes for parent-child and gap verification.

In Listing 5.6, the definitions of the relevant classes for the gap validation are listed. Class `PSAssertedGapFalseCalculatedGapTrue` represents any parent segment without an invalid child relation having an asserted gap value set to `true` while its calculated gap value is `false`. In similar manner, class `PSAssertedGapTrueCalculatedGapFalse` is defined. However, the boolean values of asserted and calculated gap are swapped. These two classes are defined as being disjoint since a parent segment cannot be an instance of both classes the same time. Finally, the union of these classes is represented by class `PSGapValidationFailed`. Since these classes are specified as defined classes, denoted by `EquivalentTo` instruction, an automated instance classification by an ontology reasoner is supported.

5.3.2.4 Validation Workflow

All information needed for the validation of the temporal decompositions is represented by the temporal ontology and the validation rules. First, all segments involved in a temporal decomposition to be validated are expressed in terms of the temporal ontology (cf. Section 5.3.2.1). Therefore instances of class `Segment` are created including start time and interval information. In addition, the relation between a segment and its corresponding sub-segments is denoted by property `hasChild`. Furthermore, gap and overlap information is modeled. Class `ParentSegment` is specified as defined class. When required, instances of class `ParentSegment` can be classified based on `hasChild` relations using an ontology reasoner.

After modeling all temporal segments and their relations, logical rules are applied (cf. Section 5.3.2.2). The result of this rule reasoning step are the actual gap and overlap values of a temporal decomposition based on its representation by the temporal ontology. These values are added to the related instance of class `ParentSegment` using the properties `hasCalculatedGap` and `hasCalculatedOverlap`. Invalid parent-child

```
Class: PSAssertedGapFalseCalculatedGapTrue
    EquivalentTo:
        PSWithoutInvalidChildSegment
         and (hasAssertedGap value false)
         and (hasCalculatedGap value true)
    DisjointWith:
        PSAssertedGapTrueCalculatedGapFalse

Class: PSAssertedGapTrueCalculatedGapFalse
    EquivalentTo:
        PSWithoutInvalidChildSegment
         and (hasAssertedGap value true)
         and (hasCalculatedGap value false)
    DisjointWith:
        PSAssertedGapFalseCalculatedGapTrue

Class: PSGapValidationFailed
    EquivalentTo:
        PSWithoutInvalidChildSegment
         and ((PSAssertedGapFalseCalculatedGapTrue or
            PSAssertedGapTrueCalculatedGapFalse))
    DisjointWith:
        PSGapValidationPassed
```

Listing 5.6: Definition of the classes for gap validation (encoded in Manchester OWL Syntax).

relations are also detected and reflected in the temporal ontology using the properties `hasInvalidChild` and `isInvalidChild`.

Finally the validation results are classified using an ontology reasoner again (cf. Section 5.3.2.3). Therefore a validation hierarchy consisting of sub-classes of class `ParentSegment` is modeled. Since these classes are specified as defined classes, an ontology reasoner is able to assign instances of class `ParentSegment` to classes in this hierarchy. The gap and overlap classifications consider the asserted and calculated gap respectively overlap values of an instance of class `ParentSegment` that has no invalid parent-child relations.

In Figure 5.5, the validation workflow for verifying the gap and parent-child relations of the temporal decomposition shown in Figure 3.5 is depicted. First, the instances of class `Segment` and their relations are created. Based on the `hasChild` relations, `S1` is classified as an instance of class `ParentSegment`. Then validation rules are applied in order to calculate the values of the actual gap and invalid parent-child relation. Since gaps exist in the relevant temporal decomposition, the value of property `hasCalculatedGap` is `true`. Additionally, no invalid parent-child relation of `S1` is found, denoted by setting the value of `hasInvalidChild` to `false`. In the last step, the validation result is classified. Therefore `S1` is classified as an instance of `PSWithoutInvalidChild` since the property value of `hasInvalidChild` is `false`. Finally, based on

the gap values, `S1` is classified as an instance of class `PSAssertedGapTrueCalculated-GapTrue` and `PSGapValidationPassed`.

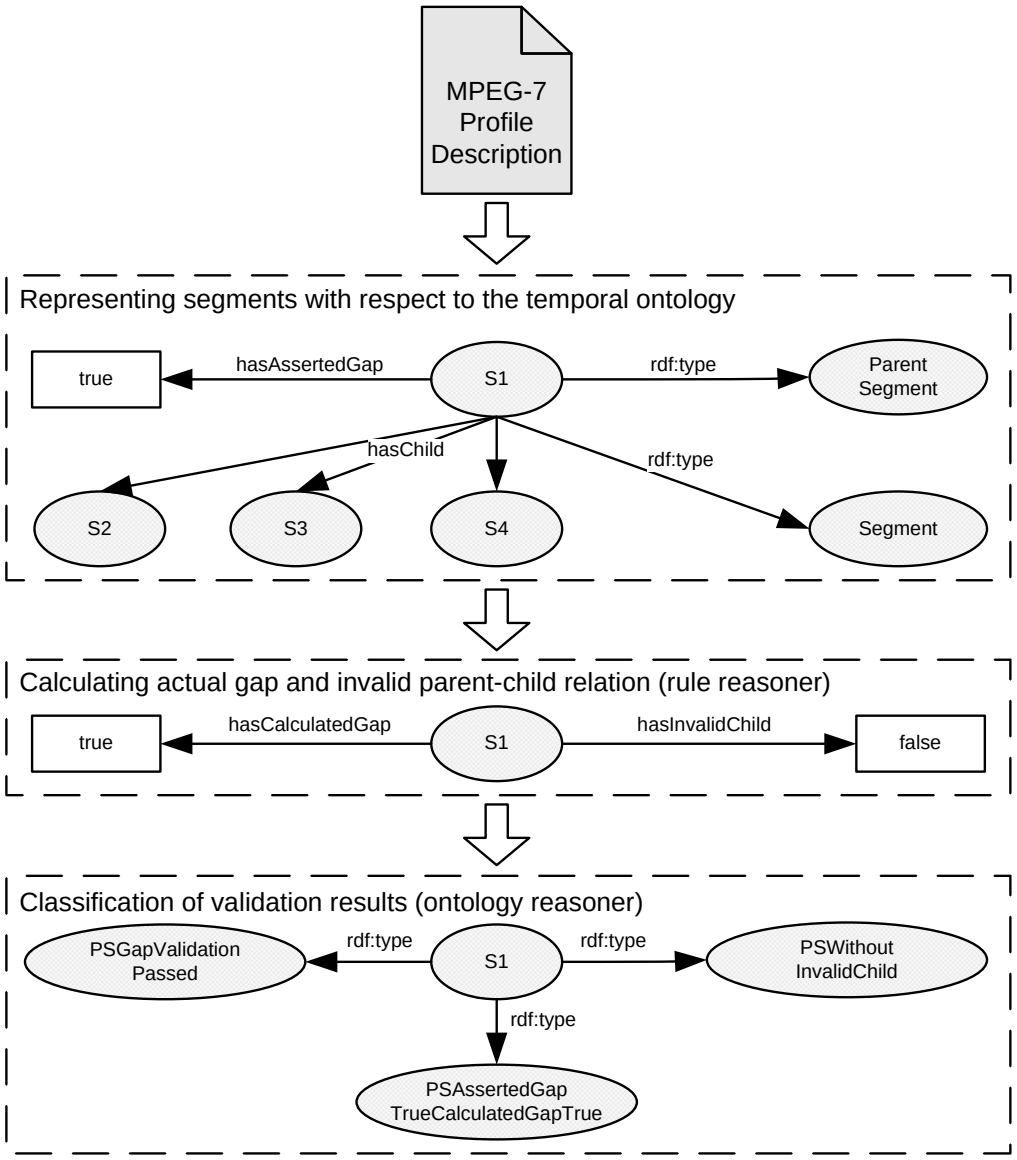

Figure 5.5: Validation steps for parent-child and gap verification.

In contrast to the approach for formalizing profile semantic constraints, the formalization approach for temporal semantic constraints is based on the combination of ontology and rule reasoning steps. Therefore the temporal ontology is used to classify the validation results, while these results are calculated by logical rules. Neither an ontology nor an ontology reasoner can perform the required gap overlap calculations of the segments. Thus logical rules are applied. The required ontology and rule reasoning steps are performed in a successive and not parallel order. This approach is chosen in order to ensure decidability while not producing infinite loops. Indeed, there would be the possibility of using DL-safe rules [68]. However, the present approach works with OWL-DL and rules in an independent manner.

5.4 Summary

In this Chapter, an approach for formalizing the semantic constraints of MPEG-7 profile descriptions was presented. Two different groups of semantic constraints are addressed by this approach. One group consists of profile-specific semantic constraints, while the other addresses temporal semantic constraints, which are profile-independent.

The proposed approach for formalizing the semantic constraints is based on the application of Semantic Web technologies, in particular, ontologies and logical rules. Ontologies are used to model the characteristics of the different profiles and temporal concepts. In addition, validation rules are defined. These rules hinge on the concepts and relations defined by the ontologies. The validation rules are represented by logical rules including predefined and customized extensions. The purpose of the validation rules is to detect and flag violations of the semantic constraints in the ontological representation of an MPEG-7 profile description.

Chapter 6

VAMP: A Semantic Validation Service for MPEG-7 Profile Descriptions

The approach for formalizing semantic constraints was presented in the previous Chapter. This approach is the basis for a semantic validation service for MPEG-7 profile descriptions, shortened by the acronym VAMP. VAMP verifies the conformance of a given MPEG-7 profile description with a selected MPEG-7 profile specification in terms of syntax and semantics. The temporal semantic constraints are also considered. In this Chapter, the validation workflow of VAMP is described first. Then the main classes of VAMP are explained. Important design and implementation details are discussed in Section. Finally, two VAMP-based applications are presented. One application realizes the VAMP approach as a web application for human users, while the other implements a RESTful web service for software agents. For simplicity, if not essentially required, namespace specifications are omitted in the explanations and related Figures and Listings.

6.1 Validation Workflow

The validation workflow describes the different actions needed for validating a single MPEG-7 profile description. This workflow is modeled by a UML activity diagram, which is depicted in Figure 6.1. In this UML diagram, an action is denoted as round-cornered rectangle. Input and result parameters of actions are defined as object nodes and are depicted as normal rectangles. The connection between object nodes and actions is denoted as object flow.

The first actions encountered in the validation workflow are part of the syntax validation process of the MPEG-7 profile description to be validated. A syntactically valid MPEG-7 profile description is a necessary precondition to validate the conformance of this description with the semantic constraints. An MPEG-7 profile description is syntactically valid if it is XML well-formed, and does conform to both the MPEG-7 XML Schema and the corresponding MPEG-7 profile XML Schema. In the activity

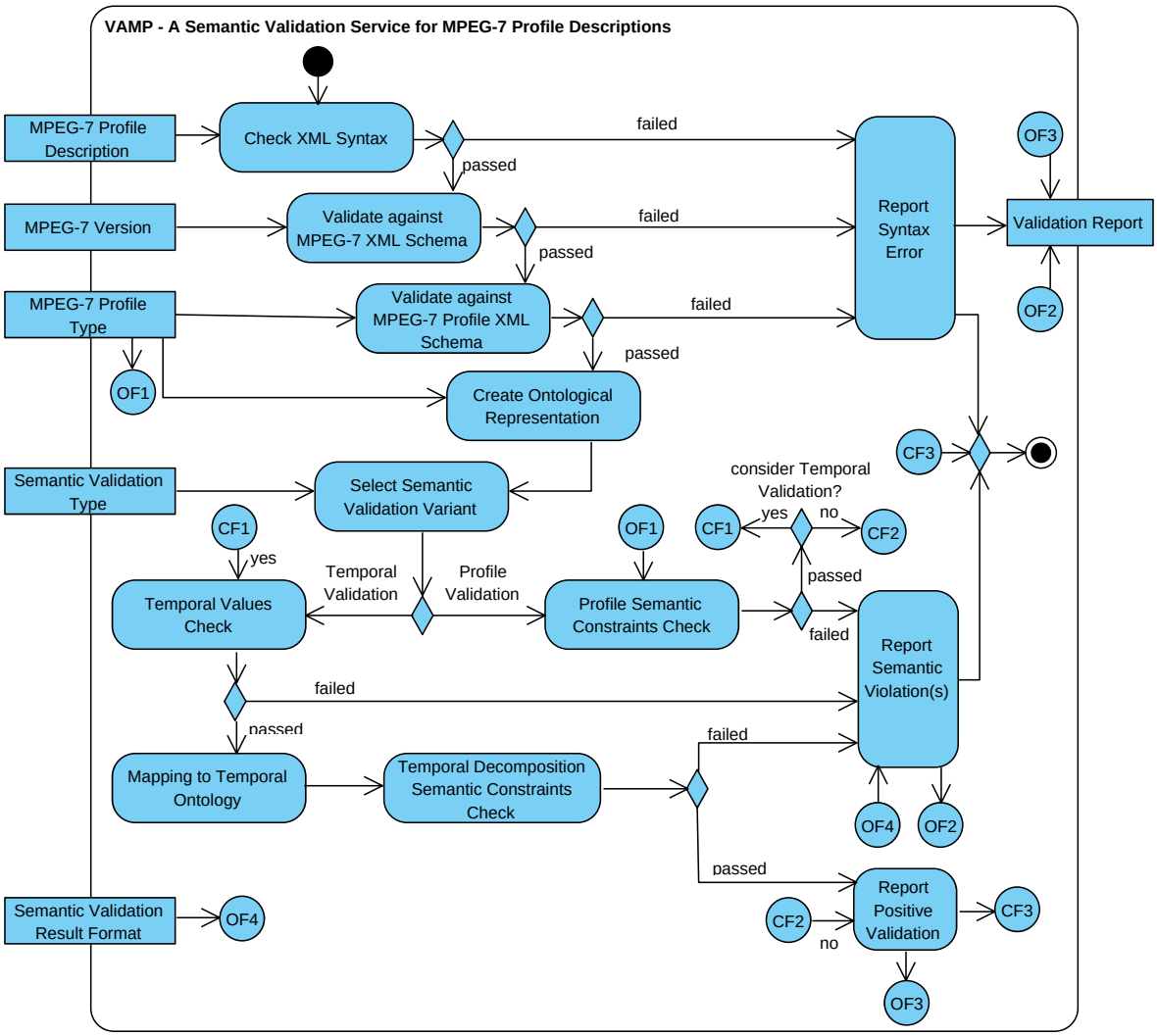

Figure 6.1: Detailed description of the validation workflow represented as UML activity diagram.

diagram, the syntax validation process is split into three different actions: *Check XML Syntax*, *Validate against MPEG-7 XML Schema*, and *Validate against MPEG-7 Profile XML Schema*. The required input parameters of these actions are the MPEG-7 profile description to be validated *(MPEG-7 Profile Description)*, the specification of the MPEG-7 XML Schema version *(MPEG-7 Version)*, and the name of the MPEG-7 profile to which the MPEG-7 profile description should conform to *(MPEG-7 Profile Type)*. Choosing different MPEG-7 XML Schema versions enables the compatibility to MPEG-7 profile descriptions that are not defined by the latest MPEG-7 XML Schema. Currently two MPEG-7 XML Schema versions and three MPEG-7 profiles are supported by VAMP. The supported profiles are the TRECVid profile, the DAVP and the AVDP (cf. Section 7.1).

The actions of the syntax validation process are processed one after the other as long as no error is detected. In case of an error, the action *Report Syntax Error* generates

a *Validation Report*, which represents the result parameter of the validation workflow. Then the validation workflow is terminated.

In case the syntax validation is passed successfully, the MPEG-7 profile description is prepared for the semantic constraints check. An ontological representation of the MPEG-7 profile description is created by instantiating the related profile ontology with respect to this description (action *Create Ontological Representation*). Afterwards, the semantic validation variant is selected. Therefore the input parameter *Semantic Validation Type* is evaluated by the action *Select Semantic Validation Variant*. An MPEG-7 profile description can be validated against the profile semantic constraints (profile validation) or temporal semantic constraints (temporal validation) exclusively or against the combination of both. In the latter case, the temporal validation is performed only, if the profile validation is passed successfully. The validation of profile-specific semantic constraints is performed by the action *Profile Semantic Constraints Check*, while the validation of the temporal semantic constraints is split into three sub-actions, which are performed successively: the detection of inconsistent temporal values *(Temporal Values Check)*, the mapping of instances from the profile ontology to the temporal ontology *(Mapping to Temporal Ontology)*, and the validation of the semantic constraints of temporal decompositions *(Temporal Decomposition Semantic Constraints Check)*. Passing the action *Temporal Values Check* successfully is a necessary precondition to execute the other two actions.

The results of the semantic constraints checks are represented by the result parameter *Validation Report*. Detected semantic inconsistencies are summarized by the action *Report Semantic Violation(s)*, while a semantically conformant MPEG-7 profile description is reported by the action *Report Positive Validation*. The format of the report is selected by the input parameter *Semantic Validation Result Format*. Finally, the validation workflow is terminated.

6.2 Class Design

The required functionality of the described actions of the VAMP workflow is provided by several classes and methods. The UML class diagram, which is depicted in Figure 6.2, shows the top-level classes and methods of VAMP. For simplicity, parameters of methods are omitted in this diagram. In the following, the classes and methods are introduced.

The class `VAMP` is responsible for the orchestration of the validation workflow. Methods for setting all input parameters (`setInputParameters()`), starting the validation (`startValidation()`), and returning the validation report (`getValidationReport()`) are provided by this class. The functionalities for the syntax validation, ontology instantiation, semantic validation, and result reporting are implemented by additional classes, which are are owned by class `VAMP` (composition relationship).

The class `MPEG7SyntaxValidator` provides methods for checking the XML well-formedness of the MPEG-7 profile description (`checkSyntax()`) and validating this description against the MPEG-7 XML Schema (`validateAgainstMPEG7Schema()`) and

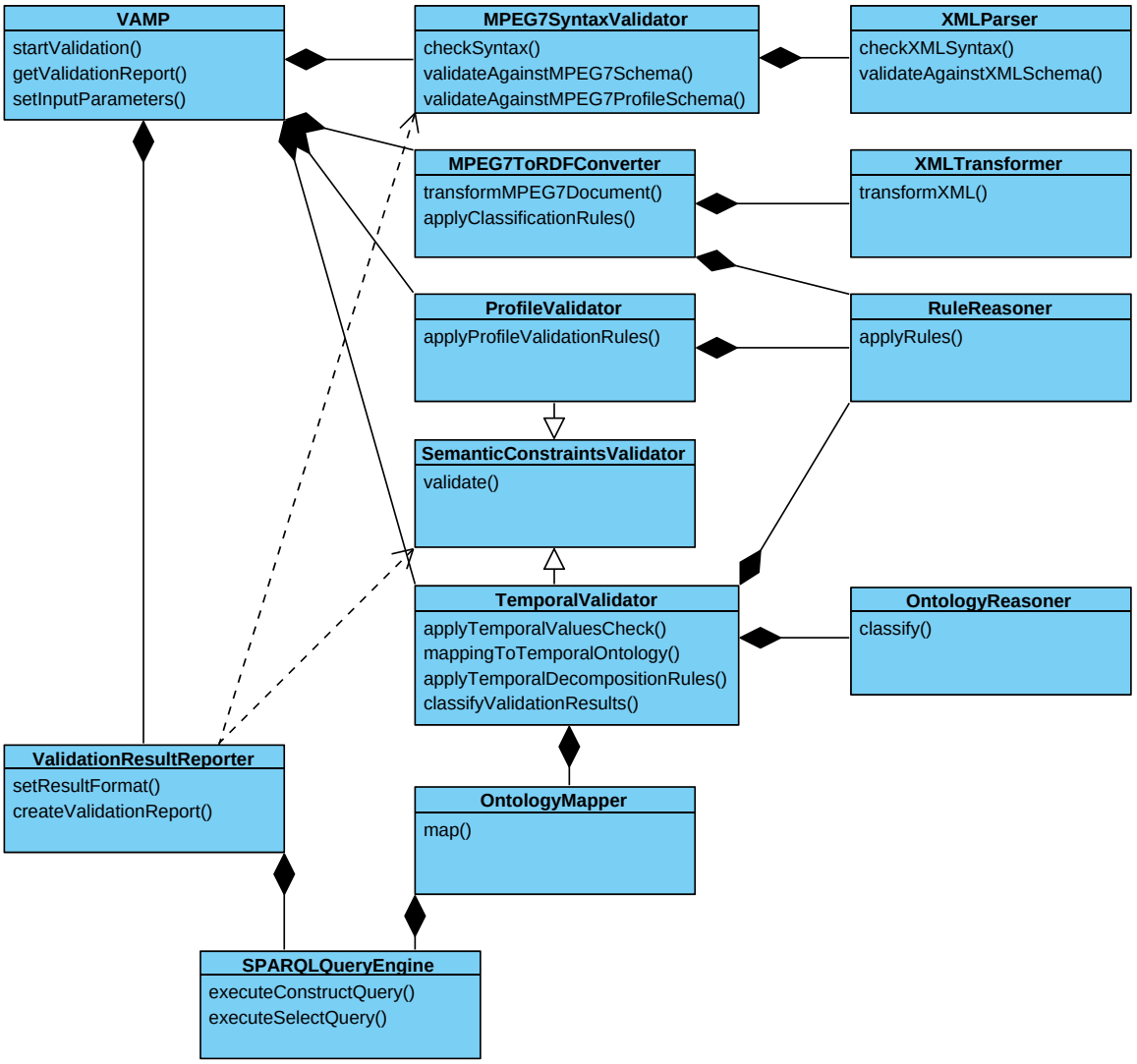

Figure 6.2: Top-level classes of VAMP.

MPEG-7 profile XML Schema (`validateAgainstMPEG7ProfileSchema()`). Therefore class `XMLParser` includes methods for executing the XML syntax and XML Schema checks.

The class `MPEG7ToRDFConverter` instantiates the profile ontology with respect to the MPEG-7 profile description to be validated. Therefore an XML transformation is executed by method `transformMPEG7Document()`, which uses the method `transformXML()` of class `XMLTransformer`. In addition, classification rules are applied by method `applyClassificationRules()`. The required functionality is provided by method `applyRules()` of class `RuleReasoner`.

The base class for the validation of the semantic constraints is class `SemanticConstraintsValidator`. Derived from this class, the sub-class `ProfileValidator` implements the validation of profile-specific semantic constraints, while the sub-class `TemporalValidator` implements the validation of temporal semantic constraints. The profile validation is executed by method `applyProfileValidationRules()`, which uses

the class `RuleReasoner` for executing the validation rules. Method `applyTemporalValuesCheck()` executes also validation rules for detecting inconsistent temporal values. The mapping from instances of the profile ontology to the temporal ontology is performed by method `mappingToTemporalOntology()`. This method uses the class `OntologyMapper` to execute a SPARQL construct query via method `executeConstructQuery()` of class `SPARQLQueryEngine`.

The different types of validation reports are created by class `ValidationResultReporter`. The format of a validation report is selected by method `setResultFormat()`. In case of reporting semantic inconsistencies, method `executeSelectQuery()` of class `SPARQLQueryEngine` is used to retrieve detected inconsistencies in the ontological representation.

6.3 Design and Implementation Details

All the required classes for providing the functionality of VAMP are implemented in Java[1]. Several open source software libraries are integrated in order to reuse functions for XML and RDF processing and for ontology and rule reasoning tasks. In the following, important implementation details of VAMP are discussed.

6.3.1 Syntax and Schema Validation

The methods for the syntax and schema validation of an MPEG-7 profile description are part of class `MPEG7SyntaxValidator`. These methods use the class `XMLParser`. The underlying XML processing and validation functionalities are provided by the Apache Xerces2[2] library. The required input parameter for checking the XML well-formedness is the MPEG-7 profile description to be validated.

6.3.2 Profile Ontology Instantiation

Instantiating the profile ontology with respect to the MPEG-7 profile description to be validated is done as a two-step process. First, an XSL transformation is applied to this description to create an initial version of the ontological representation. The result of this XSL transformation are RDF statements, which are describing new instances of the profile ontology based on the selected MPEG-7 profile description. These statements are expressed as N-Triples. The transformation is executed by method `transformXML()` of class `XMLTransformer`, which is part of class `MPEG7ToRDFConverter`. The method `transformXML()` uses the Xalan-Java[3] library for transforming XML documents. The required inputs for executing an XSL transformation are the MPEG-7 profile description and an XSLT document. Second, the ontological representation is refined by applying

[1] `http://www.java.com/en/`
[2] `http://xerces.apache.org/xerces2-j/`
[3] An XSLT processor: `http://xml.apache.org/xalan-j/`

additional classification rules. These rules are executed by method `applyRules()` of class `RuleReasoner`. This class includes libraries of the Apache Jena framework[4], for RDF data processing and rule reasoning support.

The XSLT stylesheet document contains various XSL templates for defining the conversion instructions. These templates are created by hand with respect to both the related MPEG-7 profile XML Schema and the profile ontology. XML elements and attributes of the MPEG-7 profile description are selected and corresponding instances of the classes of the profile ontology are created by these templates. When required, statements describing these instances with properties, which are defined by the profile ontology, are created. The subjects and objects of these statements can be either already existing resources and literals or are created as needed based on values of the selected XML constructs.

In Listing 6.1, the XSL template for converting `VideoSegment` elements to an instance of class `Segment` of the DAVP ontology is partly shown. This `VideoSegment` template is selected by an another template that is used for converting a `TemporalDecomposition` element. In order to reference this element by the `VideoSegment` template, the URL of the corresponding and already existing instance in the profile ontology is passed as parameter (`parentURI`). A URI for identifying the processed `VideoSegment` element is also generated and stored as variable (`elementURI`) for further use. The template `writeTypeAndXPathForElement` uses this URI to create two RDF statements: one represents the `VideoSegment` element as instance of class `Segment` in the profile ontology, while the other one stores the position of this `VideoSegment` in the MPEG-7 profile description as an XPath expression. Representing the XPath information of an XML element by the profile ontology is a requirement for precise reporting of the location of an semantic error. The parent-child relation between the involved `TemporalDecomposition` and `VideoSegment` element is also described. The resulting statement having the `parentURI` as subject, property `hasSegment` as predicate, and `elementURI` as object is created by calling template `writeRDFTriple`. Additional XML templates are selected to represent descriptors of the selected `VideoSegment` element as RDF data. The presented `VideoSegment` template selects templates for processing the attribute `id` and `StructuralUnit`, `MediaTime`, and `Decomposition` elements. The current `elementURI` is passed as parameter `parentURI` to these templates.

For example, applying the `VideoSegment` template together with these templates to the `VideoSegment` element having the attribute `id="shot1_2"` (shown in Listing 3.2) results in the RDF statements presented in Listing 6.2. First, a unique URI (`:a`), which is stored in variable `elementURI`, is generated. This URI is used as subject in RDF statements to express the membership to class `Segment` (using property `rdf:type`), the position in the `MPEG-7` profile description (using property `hasXPath`), and the value of attribute `id` (using property `hasID`). The objects of these statements are literals. Another statement describes the relation of the `VideoSegment` element to its parent el-

[4] A Java framework for building Semantic Web and Linked Data applications: `http://jena.apache.org/index.html`

```
<!-- VideoSegment template -->
<xsl:template match="mpeg7:VideoSegment">
  <xsl:param name="parentURI" />

  <xsl:variable name="elementURI">
    <xsl:call-template name="generateURI" />
  </xsl:variable>

  <xsl:call-template name="writeTypeAndXPathForElement">
    <xsl:with-param name="elementURI" select="$elementURI" />
    <xsl:with-param name="class" select="$Segment" />
  </xsl:call-template>

  <xsl:call-template name="writeRDFTriple">
    <xsl:with-param name="subject" select="$parentURI" />
    <xsl:with-param name="predicate" select="$hasSegment" />
    <xsl:with-param name="object" select="$elemenURI" />
  </xsl:call-template>

  <xsl:apply-templates select="@id">
    <xsl:with-param name="parentURI" select="$elementURI" />
  </xsl:apply-templates>

  <xsl:apply-templates select="mpeg7:StructuralUnit">
      <xsl:with-param name="parentURI" select="$elementURI" />
    </xsl:apply-templates>

  <xsl:apply-templates select="mpeg7:MediaTime">
    <xsl:with-param name="parentURI" select="$elementURI" />
  </xsl:apply-templates>

  <xsl:apply-templates select="child::node()[contains(name(),'
     Decomposition')]">
    <xsl:with-param name="parentURI" select="$elementURI" />
  </xsl:apply-templates>
</xsl:template>
```

Listing 6.1: Example XSL template for converting a `VideoSegment` element to an instance of class `Segment` of the DAVP ontology.

ement (element `TemporalDecomposition` with attribute `criteria="visual shots"`). The subject of this statement is the URI of the parent element (`:b`), which is provided by parameter `parentURI`, the predicate is property `hasSegment`, and the object is the URI of the `VideoSegment` element (`:a`). In addition, the value of the corresponding `StructuralUnit` element is also represented by RDF. Thus a new URI (`:c`) is created and instantiated as member of class `StructuralUnit`. Then the statement having property `hasStructuralUnitValue` as predicate describes the actual value, while another statement designates the relation of this `StructuralUnit` element to

```
<:a> <:type> <:Segment> .
<:a> <:hasXPath> "/TemporalDecomposition/VideoSegment[2]" .
<:b> <:hasSegment> <:a> .
<:a> <:hasID> "shot1_2" .
<:a> <:hasStructuralUnit> <:c> .
<:c> <:hasStructuralUnitValue> ".:StructuralUnitCS:2005:vis.shot" .
<:a> <:hasMediaTimePoint> <:d> .
<:d> <:type> <:MediaTimePoint> .
<:d> <:hasXPath> "/VideoSegment[2]/MediaTime/MediaTimePoint" .
<:d> <:hasString> "T00:00:03:26116F30000" .
```

Listing 6.2: Resulting RDF statements when applying the `VideoSegment` template (cf. Listing 6.1) to the `VideoSegment` element denoted by `id="shot1_2"` shown in Listing 3.2. Expressed as N-Triples.

the `VideoSegment` element (using property `hasStructuralUnit`). In a similar manner, the value of element `MediaTimePoint` is represented in the profile ontology using the properties `hasMediaTimePoint` and `hasString`.

After creating the initial version of the ontological representation by applying the XSLT document to the selected MPEG-7 profile description, classification rules are applied. These rules enable a more refined classification of the previously created instances. Based on existing statements, further class memberships of instances can be expressed. For example, a given instance is classified as member of a sub-class of a class. Another task of the classification rules is to calculate the denominator and enumerator part time point and duration values. This representation variant of time values was presented in Section 5.3.2.1. Calculating the denominator and enumerator part is a requirement for mapping time values to the temporal ontology (cf. Section 6.3.4).

Some of the classification rules for refining the classification of instances of class `Segment` are shown in Listing 6.3. An instance of class `Segment` is also classified as member of class `Shot`. The classification is based on the related instance of class `StructuralValue`. First, the membership to class `StructuralUnit` is stated by rule `classifyStructuralUnit`. Therefore the object of a statement having the property `hasStructuralUnit` as predicate (`?structuralUnit`) is classified as member of class `StructuralUnit`. Afterwards, instances of class `StructuralUnit` are classified further based on property `hasStructuralUnitValue`. In rule `classifyStructuralUnitVisualShot`, the property value `urn:x-mpeg-7-davp:cs:StructuralUnitCS:2005:vis.shot` designates the membership to class `StructuralUnitVisualShot`, which is a sub-class of class `StructuralUnit`. Finally, rule `classifyShot` classifies an instances of class `Segment` as instance of class `Shot` based on the existence of an related instance of class `StructuralUnitVisualShot`.

In addition, the conversion rule `calculateMediaTimePointFraction`, which is used for calculating the enumerator and denominator part of a media time point, is also shown in Listing 6.3. The custom builtin `calculateMediaTimePointFraction` calcu-

```
[classifyStructuralUnit:
  (?segment hasStructuralUnit ?structuralUnit),
  ->
  (?structuralUnit rdf:type StructuralUnit)]

[classifyStructuralUnitVisualShot:
  (?structuralUnit rdf:type StructuralUnit),
  (?structuralUnit hasStructuralUnitValue "urn:x-mpeg-7-davp:cs:
     StructuralUnitCS:2005:vis.shot"^^xsd:string)
  ->
  (?structuralUnit rdf:type StructuralUnitVisualShot)]

[classifyShot:
  (?segment rdf:type Segment),
  (?segment hasStructuralUnit ?structuralUnit)
  (?structuralUnit rdf:type StructuralUnitVisualShot)
  ->
  (?segment rdf:type Shot)]

[calculateMediaTimePointFraction:
  (?mediaTimePoint rdf:type MediaTimePoint),
  (?mediaTimePoint hasString ?mediaTimePointString)
  calculateMediaTimePointFraction(?mediaTimePointString, ?
     mediaTimePointEnumerator, ?mediaTimePointDenominator)
  makeTemp(?fraction)
  ->
  (?fraction rdf:type Fraction),
  (?mediaTimePoint hasFraction ?fraction),
  (?fraction hasEnumerator ?mediaTimePointEnumerator),
  (?fraction hasDenominator ?mediaTimePointDenominator)]
```

Listing 6.3: Example classification rules for class `Shot`.

lates these parts based on the MPEG-7 representation of a media time point (`?mediaTimePoint`). The resulting fraction parts are stored by the variables `?mediaTimePointEnumerator` and `?mediaTimePointDenominator`. Furthermore, a new resource is created and classified as an instance of class `Fraction` for representing the enumerator and denominator part using the properties `hasEnumerator` and `hasDenominator`. Finally, this resource is linked to the related media time point (via property `hasFraction`).

Applying the classification rules (shown in Listing 6.3) to the RDF statements created by the XSL transformation (cf. Listing 6.2) results in additional statements in the ontological representation. These statements are presented in Listing 6.4. According to rule `classifyStructuralUnit` and `classifyStructuralUnitVisualShot`, `:c` is classified as instance of class `StructuralUnit` and `StructuralUnitVisualShot`. Based on this classification, `:a`, which is an instance of class `Segment`, is also classified as an instance of class `Shot`. In addition, the enumerator and denominator part of the related media time point (`:d`) is calculated and represented as instance of class `Fraction` (`:e`).

```
1 <:c> <:type> <:StructuralUnit> .
2 <:c> <:type> <:StructuralUnitVisualShot> .
3 <:a> <:type> <:Shot> .
4 <:d> <:hasFraction> <:e> .
5 <:e> <:type> <:Fraction> .
6 <:e> <:hasEnumerator> "116116"^^xsd:long .
7 <:e> <:hasDenominator> "30000"^^xsd:long .
```

Listing 6.4: Additional RDF statements after applying the classification rules shown in Listing 6.3 to the RDF statements created by the XSL transformation (cf. Listing 6.2). Expressed as N-Triples.

Instead of using classification rules, an enhanced XSLT document or a profile ontology using defined classes in combination with an ontology reasoner could also be used for some classification tasks. However, one design consideration is to keep the XSLT document as simple as possible. Furthermore, the task of the stylesheet document is to provide a basic transformation of the MPEG-7 profile description not overlooking important MPEG-7 descriptors. However, due to the amount of description tools to be considered by the stylesheet document, the number of included XSL templates is high and not easy to maintain. The creation of this stylesheet is a crucial step, since the conformance of an MPEG-7 profile description to the semantic constraints can only be checked when the related descriptors are mapped to the ontological representation. When using the profile ontology for the classification, a possible inconsistent ontology would interrupt the validation workflow. Thus classification rules are used instead and possible inconsistencies in the ontology are detected by the validation rules.

6.3.3 Semantic Constraints Validation

Validating the conformance of a given MPEG-7 profile description with profile-specific semantic constraints is described in Section 6.2. This approach is implemented by VAMP. The main class of this implementation is class `ProfileValidator`. In addition, the validation of temporal semantic constraints (Section 5.3) is also implemented. Class `TemporalValidator` is the related main class. For RDF data processing and rule reasoning, libraries of the Apache Jena framework are used. Ontology reasoning support is provided by Pellet[5], which is an OWL reasoner for Java. This reasoner is used by method `classify()` of class `OntologyReasoner`.

6.3.4 Temporal Ontology Instantiation

According to the workflow for validating the semantics of temporal decompositions (cf. Section 5.3.2.4), the temporal ontology has to be instantiated first. The required temporal information is already available in the ontological representation of the MPEG-7

[5] http://clarkparsia.com/pellet/

profile description to be checked. However, this information is not described with respect to the temporal ontology. Thus expressing the relevant temporal information using the temporal ontology is needed. In order to instantiate the temporal ontology, a mapping from the current representation to the temporal ontology is performed. This temporal ontology extends the ontological representation of the MPEG-7 profile description to be checked. The mapping instructions are defined by a SPARQL construct query. This query selects relevant resources in the ontological representation. Based on this selection, new RDF statements are created with respect to classes and properties of the temporal ontology. The SPARQL construct query is executed by the method `executeConstructQuery` of the Java class `SPARQLQueryEngine`, which imports Jena libraries for RDF and SPARQL data processing.

For example, the SPARQL construct query for mapping resources from a DAVP-based ontological representation to the temporal ontology is partly shown in Listing 6.5. In the `WHERE` clause of this query, relevant resources needed for describing temporal decompositions in the ontological representation are selected and stored using variables. The temporal decomposition of a segment is identified by the property `hasTemporalDecomposition`, while included child segments are selected by the property `hasSegment`. The gap and overlap information of decompositions and the time point and duration information of all involved segments are also selected. The corresponding variables are used to express temporal decompositions in terms of the temporal ontology. The resulting statements are defined in the `CONSTRUCT` clause of the SPARQL construct query. An example of this representation is depicted in Figure 3.5. A temporal decomposition is described as instance of class `ParentSegment`. For this instance, the stated gap and overlap information is described by the properties `hasAssertedGap` and `hasAssertedOverlap`. Child segments of a decomposition are denoted as members of class `Segment` and are related to this decomposition using property `hasChild`. In addition, time point and duration information is attached to all segments. After the temporal ontology are instantiated and added to the ontological representation, the validation process is started.

6.3.5 Validation Result Reporting

The class `ValidationResultReporter` creates the validation report describing the validation results. If the MPEG-7 profile description to be checked is not well-formed or does not pass the XML Schema checks, an error message is created. The underlying error reported by the Apache Xerces2 Java XML Parser is part of this message. The positive validation of the selected MPEG-7 profile description is also reported. A flagged violation of the formalized profile or temporal semantic constraints is processed by a SPARQL select query. This query (shown in Listing 6.6) is applied to the ontological representation of the MPEG-7 profile description. The query retrieves the type of error and the position of the XML element in the MPEG-7 profile description causing the error represented as XPath expression. The XPath expression, which is described by the property `hasXPath`, is created when instantiating the profile ontology (cf. Sec-

```
CONSTRUCT {
  ?decomposition rdf:type ParentSegment .
  ?childSegment rdf:type Segment .
  ?decomposition hasChild ?childSegment .
  ?decomposition hasAssertedGap ?gap .
  ?decomposition hasAssertedOverlap ?overlap .
  ?decomposition hasStartPointEnumerator ?parentMTPEnumerator .
  ?decomposition hasStartPointDenominator ?parentMTPDenominator .
  ... }
WHERE {
  ?parentSegment hasTemporalDecomposition ?decomposition .
  ?decomposition hasGap ?gap .
  ?decomposition hasOverlap ?overlap .
  ?decomposition hasSegment ?childSegment .
  ?parentSegment hasMediaStartTimePoint ?parentMTP .
  ?parentMTP hasFraction ?parentMTPFraction .
  ?parentMTPFraction hasEnumerator ?parentMTPEnumerator .
  ?parentMTPFraction hasDenominator ?parentMTPDenominator .
  ... }
```

Listing 6.5: Excerpt of the SPARQL construct query for instantiating the temporal ontology.

```
SELECT ?error ?xpath
WHERE {
  ?instance hasError ?error .
  ?instance hasXPath ?xpath . }
```

Listing 6.6: SPARQL select query to retrieve semantic violations in the ontological representation of the MPEG-7 profile description to be checked.

tion 6.3.2). The type of error is added to the ontological representation when detecting a violation of a semantic constraint using property `hasError`.

The SPARQL select query is executed by the method `executeSelectQuery` of the Java class `SPARQLQueryEngine`, which uses Jena for RDF and SPARQL data processing. Different SPARQL result formats are supported by Jena[6]. The parameter *Semantic Validation Result Format* (cf. Section 6.1) is used to select the result format. For example, retrieved semantic violations represented as JSON data are shown in Listing 6.7.

[6]Jena class `ResultSetFormatter`: http://jena.apache.org/documentation/javadoc/arq/com/hp/hpl/jena/query/ResultSetFormatter.html

```
{
  "head": {
    "vars": [ "error" , "xpath" ]
  } ,
  "results": {
    "distinct": false ,
    "ordered": true ,
    "bindings": [
      {
        "error": { "type": "uri" , "value": "#
           KeyframeWithMediaDuration" } ,
        "xpath": { "datatype": "http://www.w3.org/2001/XMLSchema#
           string" , "type": "typed-literal" , "value": "/Mpeg7/
           Description/MultimediaContent/AudioVisual/
           MediaSourceDecomposition/VideoSegment/TemporalDecomposition
           /VideoSegment" }
      } ,
      {
        "error": { "type": "uri" , "value": "#MisplacedKeyframe" } ,
        "xpath": { "datatype": "http://www.w3.org/2001/XMLSchema#
           string" , "type": "typed-literal" , "value": "/Mpeg7/
           Description/MultimediaContent/AudioVisual/
           MediaSourceDecomposition/VideoSegment/TemporalDecomposition
           /VideoSegment" }
      }
    ]
  }
}
```

Listing 6.7: Retrieval result represented as JSON data.

6.4 Applications

As a proof of concept and for demonstration purpose, VAMP is implemented as a web application for human users and as a RESTful web service for software agents. These applications are presented in the following.

6.4.1 VAMP Web Application

The VAMP web application for human users is implemented using Java servlets. The VAMP web interface[7] is depicted in Figure 6.3. First, the URL of the MPEG-7 profile description to be validated is entered. Thus this description has to be online accessible, a upload functionality is not supported by the VAMP demonstrator. However, for demonstration purposes, some demo examples are provided and can be selected alternatively. Second, the MPEG-7 XML Schema version is selected. Currently, two

[7] http://vamp.joanneum.at

different versions are available: MPEG-7 v1 was published in 2001, while its successor, MPEG-7 v2, was published in 2004. The next step is to select the MPEG-7 profile, to which the input MPEG-7 description document should conform to. Three MPEG-7 profiles are currently formalized according to the presented approach and are available in VAMP. These profiles are the TRECVid Profile, the DAVP, and the AVDP. Third, the semantic validation type is selected. Two different semantic validation types are available, which can be combined: profile validation and temporal validation. If both validation types are selected, the positive validation of the profile validation is a necessary precondition for starting the temporal validation. Finally, the validation process is initiated, by clicking the `Validate!` button.

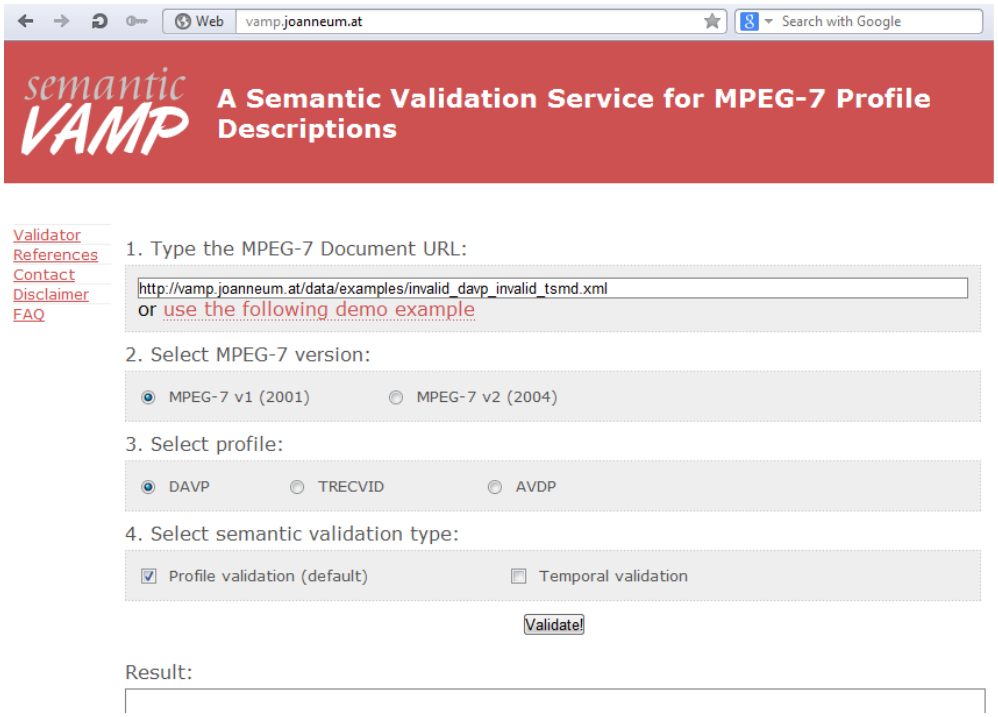

Figure 6.3: The VAMP web interface.

After the validation process is completed, a meaningful validation report is presented in the result window (cf. Figure 6.4). In case of a detected semantic error, the type of error and the involved XML element of the MPEG-7 profile description are listed. The XPath expression enables navigating directly to the error locations in the description. As additional feature, clicking an XPath expression opens a new browser window highlighting the invalid XML element in the MPEG-7 profile description. This presentation of the semantic errors in the VAMP application is established by applying an additional formatting step based on the retrieval result of the SPARQL select query (cf. Section 6.3.5).

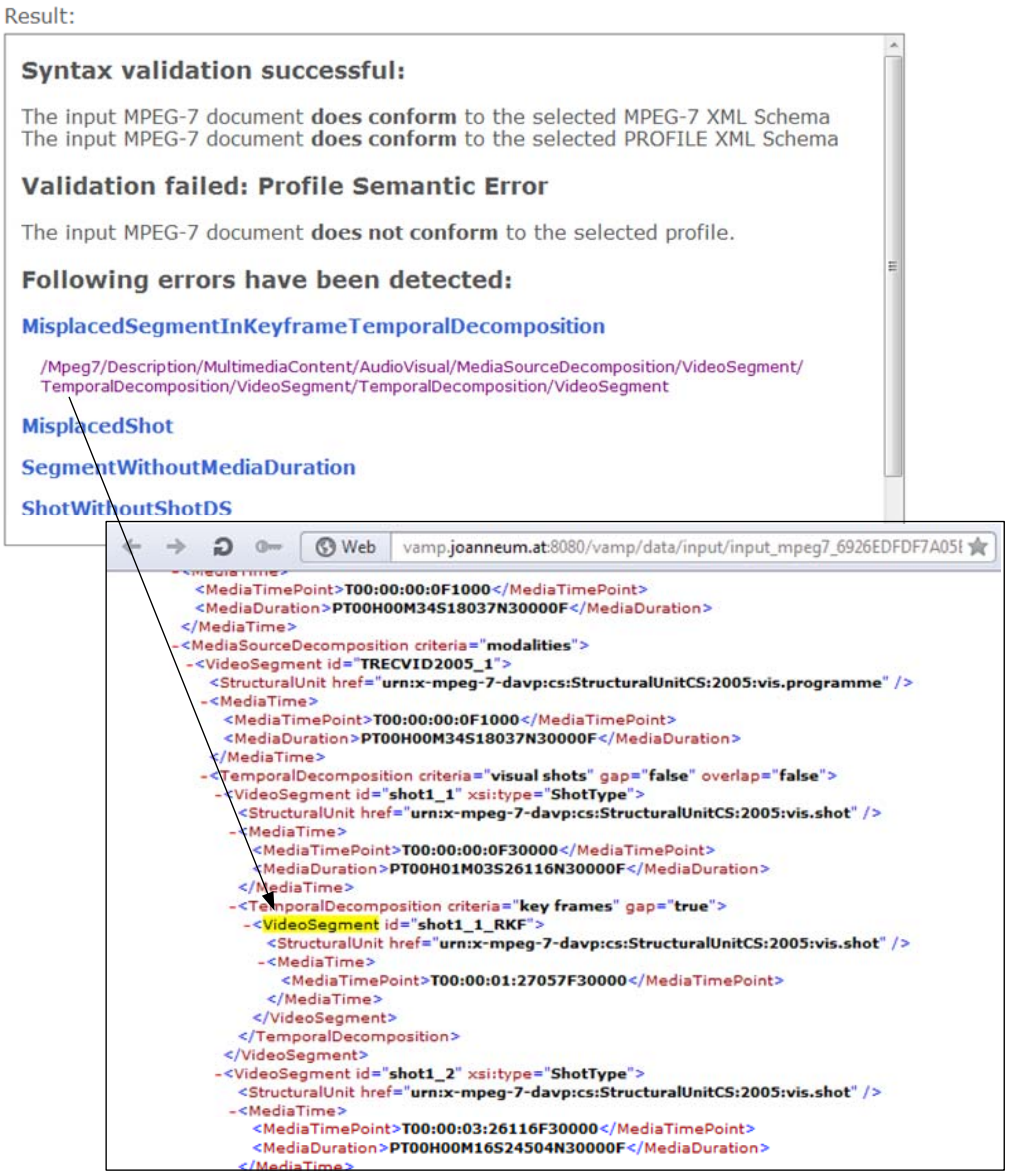

Figure 6.4: Presentation of semantic errors in the VAMP web application.

6.4.2 VAMP Web Service

In order to integrate validation service into other applications and services, a RESTful [41] web service interface for VAMP[8] is also implemented. Similar to the graphical web interface, an agent provides the MPEG-7 profile description to be validated and related validation parameters. Therefore this description is uploaded to the server and is validated. The validation result is returned expressed by the selected result format. The web service was developed during the R&D-Project PrestoPRIME[9] project at JOANNEUM RESEARCH. The specification of the web service is part of a project deliverable [20], while it was implemented by by Günter Nagler.

[8] `http://vamp.joanneum.at:8080/validator`
[9] `http://www.prestoprime.org`

6.5 Summary

In this Chapter, VAMP, which is a semantic validation service for MPEG-7 profile descriptions, was introduced. The validation workflow, the class design, and important design and implementation details were discussed. As a proof of concept, the implementation of VAMP as a web application for human users and as web service for software agents was presented. When providing the MPEG-7 profile description to be checked in combination with additional validation parameters, VAMP verifies the conformance of this description to the formalized semantic constraints. Finally, a meaningful validation report is created and returned.

Chapter 7

Conclusion

In this Chapter, the conclusion of this thesis is drawn. First, the achieved results are summarized. Then a self-assessment including a comparison of these results with the stated objectives is performed. Finally, possible future work in context of this thesis is described.

7.1 Results

In this thesis, the drawbacks of the MPEG-7 standard in general were summarized first. Then possible interoperability issues of MPEG-7 profiles were analyzed. These interoperability issues are based on the current definition of included semantic constraints expressed as English prose. If these constraints are not handled, inconsistent MPEG-7 profile descriptions can be created or processed. The semantic constraints of three MPEG-7 profiles used to describe audiovisual content were considered by this analysis. These profiles are the Audiovisual Profile (AVDP), the Detailed Audiovisual Profile (DAVP), and the TRECVid Profile, which was specified as part of this thesis.

In order to prevent the presented interoperability issues, an approach for formalizing the semantic constraints of MPEG-7 profiles was presented. Two different groups of semantic constraints are addressed by this approach, namely profile-specific semantic constraints and temporal semantic constraints. The approach is based on the application of ontologies and logical rules. Ontologies are used to model the characteristics of the different profiles and temporal concepts, while rules are used to detect and flag violations of the semantic constraints in the ontological representation of an MPEG-7 profile description.

This approach was implemented in VAMP, which is a semantic validation service for MPEG-7 profile descriptions. VAMP verifies the conformance of a given MPEG-7 description with a selected MPEG-7 profile specification in terms of syntax and semantics. The temporal semantic constraints are also considered. The required inputs for the validation process are the MPEG-7 profile description to be checked and validation parameters, such as profile and validation type. First, the MPEG-7 profile description to

be validated is syntactically validated against both the MPEG-7 XML Schema and the selected MPEG-7 profile XML Schema. A syntactically valid and well-formed MPEG-7 profile description is a necessary precondition for validating the semantic constraints. Second, instances of the profile ontology are created with respect to the MPEG-7 profile description to be validated. Based on the selected validation type, profile validation and temporal validation rules are applied After the validation process is completed, a report of all possible detected errors is provided.

As a proof of concept, VAMP is available as a web based application for human users and as a REST-style web service for embedding the validation functionality into other applications. All resources required for the validation process implemented in VAMP are listed in Appendix A. Currently the DAVP, the AVDP, and the TRECVid Profile are integrated in VAMP. The formalization of the DAVP and TRECVid Profile was done along with this thesis. In addition, the AVDP was formalized with respect to the presented work by Günter Nagler during the R&D-Project PrestoPRIME.

The proposed formalization approach for MPEG-7 profiles can be a trigger for considering a refinement of the textual descriptions of semantic constraints in the profile definition. In some cases they are defined fuzzy leaving space for different interpretations. For example, describing a valid decomposition hierarchy and not explicitly addressing all variants may result in different interpretations of the use of such hierarchy. When formalizing the related semantic constraints by a profile ontology and validation rules probably ambiguities may become evident.

Parts of the described work was already published in some papers. The approach for formalizing temporal semantic constraints was published as a workshop paper [54], while the general validation approach of VAMP was published as a journal paper [79].

7.2 Self-Assessment and Future Work

As a proof of concept, the semantic constraints related to the structural and temporal description of MPEG-7 profile descriptions are currently formalized and available in VAMP. Therefore the structural semantic constraints of three MPEG-7 profiles, namely the DAVP, AVDP, and TRECVid Profile are considered. The semantic constraints required for the validation of media information and the integration of classification schemes (cf. Section 3.3.3) is implemented rudimentarily only. The completion of the formalization process by following the presented approach is one of the next steps of future work.

Possible refinement steps for improving the usability of the ontologies are also identified. The three available MPEG-7 profile ontologies describe the segmentation of content in similar ways. Classes and properties for describing segment and decomposition types are defined in each profile separately. These classes and properties can be concentrated by an own segment ontology, which then can be imported by the profile ontologies. If required, the imported classes and properties can be extended. In the same way, already established ontologies and vocabularies can also be considered for

defining the profile ontologies. Additionally, a separate ontology expressing the semantic error types can be defined that is also reused by the profile ontologies. For example, the Simple Knowledge Organization System (SKOS) [63] can be used to express a hierarchy of these errors types. In addition, detailed explanations of the error types can be added. Such explanations are currently available for the AVDP only. However, they are formulated as part of the validation rules, which is not the best solution in terms of reusability and maintainability.

The instantiation process for the creation of the ontological representation can be enhanced in order to keep the XSLT document as simple as possible. This can be achieved by shifting possible instantiation tasks performed by this stylesheet document to the classification rules. However, it has to be considered that the instantiation process is a crucial step, since the conformance of an MPEG-7 profile description to the semantic constraints can only be checked when the related descriptors are correctly mapped to the ontological representation.

The calculation of the actual gap and overlap attributes of temporal decompositions is also a subject of potential improvements. Currently a set of rules is used to calculate the required properties in the temporal ontology (cf. Section 5.3.2.2). The calculations are based on the pairwise comparison of the temporal information of the involved segments. These segments are selected on an arbitrary basis, which is inefficient compared to a processing strategy in a sorted order. If a temporal order of the segments is available, segments that are not in the time range of the parent segment can be easier and faster detected. The same is true for finding gaps between the parent segment and the first respectively last segment in the temporal sequence. However, a sorted order can be described by the temporal ontology, but an additional classification step is required. The calculation rules have to be extended also in order to select the temporal segments in an ordered way. This would increase the complexity of these rules. Possible new rules have to be created also. Instead of applying rules, an alternative approach based on a SPARQL select query in combination with Java code importing Jena libraries can be used for the calculation of the gap and overlap attributes. First, all involved temporal segments are retrieved by the SPARQL query. Then these segments are ordered and compared programmatically in order to detect gaps, overlaps, and invalid parent-child relations. Finally, RDF statements representing the calculation results are added to the temporal ontology.

The inefficient calculation of the actual gap and overlap attributes of temporal decompositions contributes to observed performance issues of VAMP. The validation of larger MPEG-7 profile documents, approximately starting at a file size of 2 MB and including more than 50 decompositions and 200 segments, is time-consuming along with a high main memory consumption. Thus total processing times of more than 3 hours and out of memory errors resulting in an unexpected termination of the validation process were occurred. In case a long validation time is a minor issue, the VAMP web service can be chosen since it acts asynchronously using a polling strategy. An immediate validation result is not always be expected by this approach. The MPEG-7 profile document to be validated is uploaded on the server and queued first. Then the

validation process is started. The termination of this process is recognized by frequently polling the current validation status. Finally, the validation result is retrieved. However, as a future work, an intensive investigation is required in order to reveal the exact cause of the performance issues and to propose improvements.

In addition, an extended usage evaluation of the available VAMP application is also scheduled as part of the future work. Therefore the log files of more than 2500 validations have to be considered. An evaluation of the initial VAMP web service was already reported in [79]. It turned out that only 2% of 467 validations passed successfully both the profile XML Schema and the semantic validation. 46% were not valid with respect to the MPEG-7 XML Schema and 20% did not conform to the selected MPEG-7 profile schema. 32% were not well-formed XML documents and 16% could not be validated because of invalid URLs of the MPEG-7 profile description to be validated or internal errors probably based on performance issues.

When formalizing semantic constraints of MPEG-7 profiles based on the interpretation of their textual descriptions, the question of consistency with respect to flexibility and strictness arises. If the semantic constraints are projected to be very strict, the use of not explicitly addressed structures in resulting MPEG-7 profile descriptions is prevented. Even if such structures are used as extensions and do not interfere with the semantic constraints defined in the profile. Thus it could be an option to introduce different levels of conformance to the semantics of a profile. This approach of semantic levels is also part of future work. The idea is to define several levels of strictness in terms of semantic constraints for each profile which can then be used depending on application requirements. The definition starts with the most "liberal" semantic level, which formalizes the most basic semantic constraints of the profile. These constraints should only solve interoperability problems by avoiding ambiguities, but not unnecessarily restrict the use of optional elements or extensions. Based on this simple definition, stricter levels can be derived by adding further constraints.

Formalizing the semantic constraints of MPEG-7 profiles is not only useful for semantically validating corresponding descriptions, but also for establishing mappings between profiles or heterogeneous MPEG-7 descriptions. Such mapping instructions are based on the profile ontologies. A first approach is implemented by mapping instances of the profile ontologies to the temporal ontology (cf. Section 6.3.4). This approach can be refined to relate parts of an MPEG-7 profile descriptions to other parts defined by different MPEG-7 profiles or other domain vocabularies such as EXIF or ID3. Multimedia applications that need to index multimedia metadata from heterogeneous sources could benefit from such mappings. Formalizing the semantics of the profiles used for representing this metadata allows to express mappings between based on the stated semantics.

Another use case to be considered as future work addresses the validation of temporal decompositions of other metadata formats. The presented validation approach is generic since the temporal ontology models general properties of a temporal multimedia content structure independent of the actual description format or standard. Thus the approach

can also be applied to the validation of descriptions using other standards that support the concept of temporal segmentation.

Similar to the temporal decomposition, the spatial and the spatio-temporal decompositions suffer from the same limitations in MPEG-7. For example, if a region of an image is decomposed into sub-regions, the sub-regions must lie inside the parent region. Wrongly stated values of the `gap` and `overlap` attributes in a related description can also be occurred. Thus implementing a validating strategy for spatial and the spatio-temporal decompositions in VAMP is also desirable and part of future work. However, the actual implementation is more difficult than for temporal decompositions due to the two-dimensional nature of the spatial regions. Having a common solution for the spatial and the temporal case, a spatio-temporal validation can also be established.

Appendix A

VAMP Resources

In the following, the required XML Schemas, XSLT documents, ontologies, rules, and SPARQL documents needed for the validation process implemented in VAMP are listed. The base URL of these documents is `http://vamp.joanneum.at/data/`.

XML Schema	URL
MPEG-7 v1	`xsd/mpeg7_xsd/mpeg7_2001/mpeg7-2001.xsd`
MPEG-7 v2	`xsd/mpeg7_xsd/mpeg7_2004/mpeg7-2004.xsd`
TRECVid	`xsd/trecvid_xsd/trecvid-2001.xsd`
DAVP (2005)	`xsd/davp_xsd/davp_2005/davp-2005.xsd`
DAVP (2009)	`xsd/davp_xsd/davp_2009/davp-2009.xsd`
AVDP	`xsd/avdp_xsd/avdp_2010/avdp-2010.xsd`

Table A.1: MPEG-7 XML Schemas.

Type	URL
SPARQL select query	`sparql/has_error.query`

Table A.2: SPARQL select query for semantic error reporting.

Profile	XSLT URL
TRECVid	`xslt/xslt_trecvid/trecvid2rdf.xsl`
DAVP (2005)	`xslt/xslt_davp/davp2rdf.xsl`
DAVP (2009)	`xslt/xslt_davp/davp2rdf_v2.xsl`
AVDP	`xslt/xslt_avdp/avdp2rdf_schema2004.xsl`

Table A.3: XSLT documents for creating the ontological representation.

Profile	Profile ontology URL
TRECVid	`owl/trecvid.owl`
DAVP	`owl/davp.owl`
AVDP	`owl/avdp.owl`

Table A.4: Profile ontologies.

Profile	Rules URL
TRECVid	`rules/trecvid_classification.rules`
DAVP	`rules/davp_classification.rules`
AVDP	`rules/avdp_classification.rules`

Table A.5: Profile classification rules.

Profile	Rules URL
TRECVid	`rules/trecvid_validation.rules`
DAVP	`rules/davp_validation.rules`
AVDP	`rules/avdp_validation.rules`

Table A.6: Profile validation rules.

Profile	Rules URL
TRECVid	`rules/trecvid_tsmd_validation.rules`
DAVP	`rules/davp_tsmd_validation.rules`
AVDP	`rules/avdp_tsmd_validation.rules`

Table A.7: Temporal values validation rules.

Profile	SPARQL construct query URL
TRECVid	`mapping_trecvid_tsmd/trecvid_2_tsmd.query`
DAVP	`mapping_davp_tsmd/davp_2_tsmd.query`
AVDP	`mapping_avdp_tsmd/avdp_2_tsmd.query`

Table A.8: SPARQL construct queries for instantiating the temporal ontology.

Type	URL
Temporal Ontology	`tsmd/temporal_segments.owl`
Validation Rules	`tsmd/temporal_rules_step_1.rules`
	`tsmd/temporal_rules_step_2.rules`

Table A.9: Temporal ontology resources.

Bibliography

[1] Information technology – MPEG systems technologies – Part 1: Binary MPEG format for XML. ISO/IEC 23001-1:2006.

[2] Information technology – Multimedia content description interface – Part 10: Schema definition. ISO/IEC 15938-10:2005.

[3] Information technology – Multimedia content description interface – Part 11: MPEG-7 profile schemas. ISO/IEC TR 15938-11:2005.

[4] Information technology – Multimedia content description interface – Part 12: Query format. ISO/IEC 15938-12:2008.

[5] Information technology – Multimedia content description interface – Part 2: Description definition language. ISO/IEC 15938-2:2002.

[6] Information technology – Multimedia content description interface – Part 3: Visual. ISO/IEC 15938-3:2002.

[7] Information technology – Multimedia content description interface – Part 4: Audio. ISO/IEC 15938-4:2002.

[8] Information technology – Multimedia content description interface – Part 5: Multimedia description schemes. ISO/IEC 15938-5:2003.

[9] Information technology – Multimedia content description interface – Part 5: Systems. ISO/IEC 15938-1:2002.

[10] Information technology – Multimedia content description interface – Part 6: Reference software. ISO/IEC 15938-6:2003.

[11] Information technology – Multimedia content description interface – Part 7: Conformance testing. ISO/IEC 15938-7:2003.

[12] Information technology – Multimedia content description interface – Part 8: Extraction and use of MPEG-7 descriptions. ISO/IEC TR 15938-8:2002.

[13] Information technology – Multimedia content description interface – Part 9: Profiles and levels. ISO/IEC 15938-9:2005.

[14] Information technology – Multimedia content description interface – Part 9: Profiles and levels, AMENDMENT 1: Extensions to profiles and levels. ISO/IEC 15938-9:2005.

[15] Information technology – Multimedia Content Description Interface. ISO/IEC 15938, 2001.

[16] Metadata Production Framework Specifications (v. 2.0.2E). Technical report, NHK Science and Technical Research Laboratories, Dec. 2008. http://www.nhk.or.jp/strl/mpf/english/index.htm.

[17] Richard Arndt, Raphaël Troncy, Steffen Staab, Lynda Hardman, and Miroslav Vacura. COMM: Designing a Well-Founded Multimedia Ontology for the Web. In *6th International Semantic Web Conference (ISWC'07)*, pages 30–43, Busan, South Korea, 2007.

[18] Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the EL Envelope. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, IJCAI'05, pages 364–369, San Francisco, CA, USA, 2005. Morgan Kaufmann Publishers Inc.

[19] 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, New York, NY, USA, 2003.

[20] Werner Bailer, Martin Höffernig, Kurt Majcen, Günter Nagler, and Hemlut Mülner. Deliverable ID4.0.2b - interface specification of metadata conversion and deployment services , 2011. project deliverable PrestoPRIME. http://www.prestoprime.org.

[21] Werner Bailer and Peter 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.

[22] Werner Bailer, Peter Schallauer, and Helmut Neuschmied. MPEG-7 Detailed Audiovisual Profile - Description of the Profile, October 2007. http://mpeg7.joanneum.at/wp-content/uploads/2012/07/mpeg7davp0.6.pdf.

[23] Dave Beckett. RDF/XML Syntax Specification (Revised) W3C Recommendation 10 February 2004. W3C Recommendation, 10 February 2004. http://www.w3.org/TR/rdf-syntax-grammar/.

[24] David Beckett and Tim Berners-Lee. Turtle - Terse RDF Triple Language. W3C Team Submission, 28 March 2011. http://www.w3.org/TeamSubmission/2011/SUBM-turtle-20110328/.

[25] Anders Berglund. Extensible Stylesheet Language (XSL) Version 1.1. W3C Recommendation, December 2006.
http://www.w3.org/TR/xsl11/.

[26] Tim Berners-Lee, Roy Thomas Fielding, and Larry Masinter. Uniform Resource Identifier (URI): Generic Syntax. *Network Working Group*, 66(3986):1–61, 2005.

[27] Chris Bizer and Richard Cyganiak. RDF 1.1 TriG. W3C Recommendation, 25 February 2014.
http://www.w3.org/TR/trig/.

[28] Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler, and Francois Yergeau. Extensible Markup Language (XML) 1.0 (Fifth Edition). W3C Recommendation, 26 November 2008. http://www.w3.org/TR/xml/.

[29] Dan Brickley and R.V. Guha. RDF Schema 1.1. W3C Recommendation, 16 January 2014.
http://www.w3.org/TR/2014/REC-rdf-schema-20140225/.

[30] Jeen Broekstra and Arjohn Kampman. SeRQL: An RDF Query and Transformation Language, August 2004.

[31] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Tractable Reasoning and Efficient Query Answering in Description Logics: The *DL-Lite* Family. *J. of Automated Reasoning*, 39(3):385–429, 2007.

[32] Gavin Carothers. RDF 1.1 N-Quads. W3C Recommendation, 25 February 2014.
http://www.w3.org/TR/n-quads/.

[33] Gavin Carothers and Andy Seaborne. RDF 1.1 N-Triples. W3C Recommendation, 25 February 2014.
http://www.w3.org/TR/n-triples/.

[34] Richard Cyganiak, David Wood, and Markus Lanthaler. RDF 1.1 Concepts and Abstract Syntax. W3C Recommendation, 25 February 2014.
http://www.w3.org/TR/rdf11-concepts/.

[35] Neil Day and José M. Martínez. Introduction to MPEG-7 (v4.0). ISO/IEC JTC1/SC29/WG11N4675, March 2002.

[36] DCMI Usage Board. DCMI metadata terms. DCMI recommendation, Dublin Core Metadata Initiative, December 2006. Published online on December 18th, 2006 at
http://dublincore.org/documents/2006/12/18/dcmi-terms/.

[37] Mike Dean and Guus Schreiber. OWL Web Ontology Language: Reference. W3C Recommendation, 10 February 2004.
http://www.w3.org/TR/owl-ref/.

[38] M. Duerst and M. Suignard. RFC 3987: Internationalized Resource Identifiers (IRIs). RFC 3987 (Proposed Standard), see http://www.ietf.org/rfc/rfc3987.txt, January 2005.

[39] David C. Fallside and Priscilla Walmsley. XML Schema Part 0: Primer Second Edition. W3C Recommendation 28 October 2004. http://www.w3.org/TR/xmlschema-0/.

[40] Lee Feigenbaum, Gregory Todd Williams, Kendall Grant Clark, and Elias Torres. SPARQL 1.1 Protocol. W3C Recommendation, 21 March 2013. http://www.w3.org/TR/2013/REC-sparql11-protocol-20130321/.

[41] Roy T. Fielding and Richard N. Taylor. Principled design of the modern web architecture. *ACM Trans. Internet Technol.*, 2(2), May 2002.

[42] International Organization for Standardization. Definition of MPEG-7 Description Profiling. ISO/IEC JTC1/SC29/WG11 N6079, October 2003.

[43] Ned Freed and Nathaniel Borenstein. RFC2046: Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types, 1996.

[44] Tim Furche, Francois Bry, James Bailey, Sebastian Schaffert, Benedikt Linse, Renzo Orsini, Ian Horrocks, Michael Krauss, and Oliver Bolzer. Survey over Existing Query and Transformation Languages, Revision 2.0. http://rewerse.net/deliverables/m24/i4-d9a.pdf, 2006.

[45] Roberto Garcia and Oscar Celma. Semantic Integration and Retrieval of Multimedia Metadata. In 5^{th} *International Workshop on Knowledge Markup and Semantic Annotation (SemAnnot'05)*, Galway, Ireland, 2005.

[46] Paul Gearon, Alexandre Passant, and Axel Polleres. SPARQL 1.1 Update. W3C Recommendation, 21 March 2013. http://www.w3.org/TR/2013/REC-sparql11-update-20130321/.

[47] Bernardo Cuenca Grau, Ian Horrocks, Boris Motik, Bijan Parsia, Peter Patel-Schneider, and Ulrike Sattler. OWL 2: The Next Step for OWL. *Journal of Web Semantics*, 6(4):309–322, 2008.

[48] Benjamin Grosof, Ian Horrocks, Raphael Volz, and Stefan Decker. Description Logic Programs: Combining Logic Programming with Description Logic. In *Proceedings of the WWW2003 Conference, Budapest, Hungary*. ACM, 2003.

[49] W3C SPARQL Working Group. SPARQL 1.1 Overview. W3C Recommendation, 21 March 2013. http://www.w3.org/TR/sparql11-overview/.

[50] Thomas R. Gruber. Ontology. In *Encyclopedia of Database Systems*, pages 1963–1965. Springer, 2009.

[51] Michael Hausenblas. Multimedia Vocabularies on the Semantic Web. W3C Incubator Group Report, July 2007. http://www.w3.org/2005/Incubator/mmsem/XGR-vocabularies/.

[52] Ivan Herman, Ben Adida, Manu Sporny, and Mark Birbeck. RDFa 1.1 Primer - Second Edition. W3C Working Group Note, 22 August 2013. http://www.w3.org/TR/rdfa-primer/.

[53] Pascal Hitzler, Markus Krötzsch, Bijan Parsia, Peter F. Patel-Schneider, and Sebastian Rudolph. OWL 2 Web Ontology Language Primer (Second Edition) . W3C Recommendation, 11 December 2012. http://www.w3.org/TR/owl2-primer/.

[54] Martin Höffernig, Michael Hausenblas, and Werner Bailer. Semantics of Temporal Media Content Descriptions. In *Multimedia Metadata Applications Workshop (M3A)*, pages 155–162, Graz, Austria, 2007.

[55] Matthew Horridge and Peter F. Patel-Schneider. OWL 2 Web Ontology Language Manchester Syntax (Second Edition). W3C Working Group Note, 11 December 2012. http://www.w3.org/TR/owl2-manchester-syntax/.

[56] I. Horrocks, P.F. Patel-Schneider, and F. van Harmelen. From SHIQ and RDF to OWL: The making of a Web Ontology Language. *Journal of Web Semantics*, 1(1):7–26, 2003.

[57] Ian Horrocks, Oliver Kutz, and Ulrike Sattler. The Even More Irresistible $\mathcal{SROIQ}$. In *Proc. of the 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2006)*, pages 57–67. AAAI Press, 2006.

[58] Ian Horrocks and Peter F. Patel-Schneider. Reducing OWL Entailment to Description Logic Satisfiability. In *International Semantic Web Conference (ISWC-2003)*, 2003. Submitted.

[59] Jane 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.

[60] International Organization for Standardization. Representations of dates and times, second edition. ISO 8601, 15 December 2000.

[61] Gregory Karvounarakis, Sofia Alexaki, Vassilis Christophides, Dimitris Plexousakis, and Michel Scholl. RQL: A Declarative Query Language for RDF. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, pages 592–603. ACM Press, 2002.

[62] Frank Manola and Eric Miller. RDF (Ressource Description Framework) Primer. W3C Recommendation, 10 February 2004.
http://www.w3.org/TR/rdf-primer/.

[63] Alistair Miles and Sean Bechhofer. SKOS Simple Knowledge Organization System Reference. W3C Recommendation, 18 August 2009.
http://www.w3.org/TR/2009/REC-skos-reference-20090818/.

[64] Boris Motik. On the Properties of Metamodeling in OWL. *Journal of Logic and Computation*, 17(4):617–637, 2007.

[65] Boris Motik, Bijan Parsia, and Peter F. Patel-Schneider. OWL 2 Web Ontology Language XML Serialization (Second Edition). W3C Recommendation, 11 December 2012.
http://www.w3.org/TR/owl2-xml-serialization/.

[66] Boris Motik, Peter F. Patel-Schneider, and Bernardo Cuenca Grau. OWL 2 Web Ontology Language Direct Semantics (Second Edition). W3C Recommendation, 11 December 2012.
http://www.w3.org/TR/owl-direct-semantics/.

[67] Boris Motik, Peter F. Patel-Schneider, and Bijan Parsia. OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax (Second Edition). W3C Recommendation, 11 December 2012.
http://www.w3.org/TR/owl-syntax/.

[68] Boris Motik, Ulrike Sattler, and Rudi Studer. Query Answering for OWL-DL with Rules. *Journal of Web Semantics*, 3(1):41–60, 2005.

[69] Jacco van Ossenbruggen, Frank Nack, and Lynda Hardman. That Obscure Object of Desire: Multimedia Metadata on the Web (Part I). *IEEE Multimedia*, 11(4), 2004.

[70] Peter F. Patel-Schneider and Ian Horrocks. OWL 1.1 Web Ontology Language Overview. W3C Member Submission, 19 December 2006.
http://www.w3.org/Submission/2006/SUBM-owl11-overview-20061219/.

[71] Fernando Pereira. MPEG-7 Requirements Document v.16. ISO/IEC JTC1/SC29/WG11/N4510. Pattaya, Thailand, December 2001.

[72] Silvia Pfeiffer and Uma Srinivasan. TV Anytime as an application scenario for MPEG-7. In *Workshop on Standards, Interoperability and Practice*, Los Angeles, California, USA, 2000.

[73] Phillipe Salembier and Thomas Sikora. *Introduction to MPEG-7: Multimedia Content Description Interface*. John Wiley & Sons, Inc., New York, NY, USA, 2002.

[74] Konstantin Schinas, Wolfgang Schmidt, Franz Höller, Herwig Zeiner, Werner Bailer, and Michael Hausenblas. Metadata in the Digital Cinema Workflow and its Standards Report. project deliverable IP-RACINE D3.2.1, June 2005.

[75] Michael Schneider. OWL 2 Web Ontology Language RDF-Based Semantics (Second Edition). W3C Recommendation, 11 December 2012.
http://www.w3.org/TR/owl-rdf-based-semantics/.

[76] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A Practical OWL-DL Reasoner. *Web Semant.*, 5(2):51–53, June 2007.

[77] Manu Sporny, Dave Longley, Gregg Kellogg, Markus Lanthaler, and Niklas Lindström. JSON-LD 1.0. W3C Recommendation, 16 January 2014.
http://www.w3.org/TR/2014/REC-json-ld-20140116/.

[78] Raphaël Troncy, Werner Bailer, Michael Hausenblas, Philip Hofmair, and Rudolf Schlatte. Enabling Multimedia Metadata Interoperability by Defining Formal Semantics of MPEG-7 Profiles. In *1st International Conference on Semantics And digital Media Technology (SAMT'06)*, pages 41–55, Athens, Greece, 2006.

[79] Raphaël Troncy, Werner Bailer, Martin Höffernig, and Michael Hausenblas. VAMP: A Service for Validating MPEG-7 Descriptions w.r.t. to Formal Profile Definitions. *Multimedia Tools and Applications*, 46(2-3):307–329, Jan. 2010.

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