WeKnowIt

Emerging, Collective Intelligence for Personal,
Organisational and Social Use

FP7-215453

D5.2.2

Final Knowledge Management Methodology

Dissemination level:	Public
Contractual date of delivery:	Month 36, March 31, 2011
Actual date of delivery:	Month 36, March 31, 2011
Workpackage:	WP5 Organisational Intelligence
Task:	T5.2 Knowledge Management Methodology
Type:	Report
Approval Status:	Approved
Version:	1.0
Number of pages:	175
Filename:	D5.2.2˙kmm.tex

Abstract

The final knowledge management methodology of the WeKnowIt project comprises a set of methods for designing, combining, aligning, and using ontology-based knowledge. The first method describes the design of so-called core ontologies that provide precise semantics and rich axiomatization. A core ontology can be very generic and abstract such as the Event-Model-F (reported in deliverable D5.2.1 Prototypical Knowledge Management Methodology) for events and event relations or the Multimedia Metadata Ontology (M3O) for describing multimedia metadata. Core ontologies are in particular suitable for integrating heterogeneous information in complex application domains. In order to use core ontologies such as the M3O, they are typically specialized with respect to the requirements of the concrete application. Thus, the second method presents a four-step alignment approach for integrating existing metadata formats and metadata models with the M3O. Subsequently, an approach based on model-driven software engineering is presented to provide pragmatic access to ontological knowledge from object-oriented software systems. Finally, the use of the core ontology Event-Model-F is demonstrated at the example of the WeKnowIt ER Log Manager (WERL) application. The WERL application allows for an interactive exploration of emergency response log files by means of temporal, location, and semantic filters. Finally, the combinability of different patterns of the Event-Model-F is discussed.

co-funded by the European Union

History

Version	Date	Reason	Revised by
01	2011-02-24	First draft	Ansgar Scherp
02	2011-03-15	Review version	Ansgar Scherp
03	2011-03-31	Final version	Ansgar Scherp

Author list

Organization	Name	Contact Information
UoKob	Ansgar Scherp	Phone: +49 261 287 2717 Fax: - E-mail: scherp@uni-koblenz.de
UoKob	Carsten Saathoff	Phone: +49 261 287 2795 Fax: - E-mail: saathoff@uni-koblenz.de
UoKob	Thomas Franz	Phone: +49 261 287 2862 Fax: - E-mail: franz@uni-koblenz.de
UoKob	Steffen Staab	Phone: +49 261 287 2761 Fax: - E-mail: staab@uni-koblenz.de
UoKob	Stefan Scheglmann	Phone: +49 261 287 2718 Fax: - E-mail: schegi@uni-koblenz.de
UoKob	Daniel Eißing	Phone: - Fax: - E-mail: eissing@uni-koblenz.de
CERTH	Symeon Papadopoulos	Phone: +30 2311 257 772 Fax: - E-mail: papadop@iti.gr
Software Mind	T. Kaczanowski	Phone: - Fax: - E-mail: t.kaczanowski@softwaremind.pl

Executive Summary

This report presents the final knowledge management methodology for the organisational intelligence work package in the WeKnowIt project. In this work package, different methods have been developed for modeling, combining, aligning, and using ontological knowledge. The first method describes the design of core ontologies. Core ontologies like the Event-Model-F (reported in deliverable D5.2.1 Prototypical Knowledge Management Methodology) for events and event relations and the Multimedia Metadata Ontology (M3O) are a specific kind of ontologies that base on a foundational ontology and provide precise semantics through a rich axiomatization. Well designed core ontologies make use of ontology design patterns for modularization and providing flexibility in the knowledge modeled by the ontology. By this, core ontologies are extensible with respect to adding new functionality and allow for reuse of structural as well as domain knowledge.

A core ontology can be very generic and abstract such as the M3O. As such, it is not straightforward how to use the M3O in a concrete application. On the other hand, existing models and formats for multimedia metadata shall be reused in a multimedia application. In fact, in complex multimedia systems typically more than one metadata format or metadata model needs to be applied. Thus, we describe a method how existing multimedia metadata formats and metadata standards can be aligned with the M3O in order to integrate them and apply them together in a multimedia application.

Using ontological knowledge in software systems is a difficult task because of the differences between the logics-world of ontologies and the object-oriented world of software systems. In order to provide pragmatic access to ontological knowledge from object-oriented software systems, an approach based on model-driven software engineering has been developed. This approach is called OntoMDE and allows for the (semi-)automatic model-driven generation of application programming interfaces for ontologies.

The core ontology Event-Model-F that has been developed in the WeKnowIt project is applied in the use case of emergency response (ER) for the WeKnowIt ER Log Manager (WERL) application. WERL a web-based application for merging and managing different log files that are created by the different emergency response entities such as the control center, police department, and fire department during the incident. It allows to automatically align multiple log files and extracting ER-relevant semantic event information from log entry text. It makes use of the knowledge representation patterns of the Event-Model-F in order to facilitate information sharing and reuse and allows for an interactive exploration of the collected log files by means of temporal, location, and semantic filters. An initial description of WERL is already provided in deliverable D5.2.1 on the Prototypical Knowledge Management Methodology. The description in this report constitutes a refinement and extension of the previous document. Finally, we investigate the combinability of the different patterns of the Event-Model-F. We show the combination of the Event-Model-F pattern at the example of the WeKnowIt use case of emergency response and also refer to the use case consumer social group.

Abbreviations and Acronyms

AIMS	Atlas Incident Management System
API	Application Programming Interface
C4I	Command and Control Systems and Components
COMM	Core Ontology on Multimedia
DL	Description Logics
DS	Data Storage
DnS	Descriptions and Situations
DOLCE	Descriptive Ontology for Linguistic and Cognitive Engineering
DUL	DOLCE+DnS Ultralight
ER	Emergency Response
EXIF	Exchangeable Image File Format
M3O	Multimedia Metadata Ontology
MDE	Model Driven Engineering
MoOn	Model for Ontologies
MPEG	Moving Picture Experts Group
OAM	Ontology API Model
OWL	Ontology Web Language
RDF	Resource Definition Framework
SPARQL	SPARQL Protocol and RDF Query Language
UML	Unified Modeling Language
URI	Uniform Resource Identifier
X-COSIMO	Cross(X)-COntext Semantic Information Management Ontology
WERL	WeKnowIt ER Log Manager
WKI	WeKnowIt

Table of Contents

List of Figures

1 Introduction

This report documents the research carried out towards the final version of the WeKnowIt knowledge management methodology. It presents different methods developed in the WeKnowIt project for modeling and creating ontological knowledge, combining and aligning this knowledge, and providing pragmatic access to ontology-based knowledge in object-oriented software systems. We demonstrate how such knowledge can be used in concrete applications at the example of the core ontology Event-Model-F [28, 25] for modeling events and event relations in the WeKnowIt Emergency Response (ER) Log Manager (WERL) application.

One of the key factors that hinders integration of distributed, heterogeneous information systems is the lack of a formal basis for modeling the complex, structured knowledge that is to be exchanged. To alleviate this situation, we present in Section 2 an approach based on core ontologies. Core ontologies are characterized by a high degree of axiomatization and formal precision (cf. [4, 20]). This is achieved by basing on a foundational ontology. Foundational ontologies serve reference purposes [20] and aim at modeling the very basic and general concepts and relations [4, 20] that make up our world, e.g., objects, events, participation, and parthood. In addition, core ontologies should follow a pattern-oriented design approach (cf. [13]). By this, they are modular and extensible. Core ontologies allow for reusing the structured knowledge they define as well as integrating existing domain knowledge. The structured knowledge of core ontologies is clearly separated from the domain-specific knowledge. Such core ontologies allow for both formally conceptualize their particular fields and to be flexibly combined to cover the needs of concrete, complex application domains. We describe our design method for core ontologies in Section 2. A detailed analysis of requirements, description of benefits of the methods, and an extensive example of the simultaneous use and integration of our core ontologies at the example of a complex, distributed socio-technical system of emergency response is described in Appendix A.1 of this document.

The Multimedia Metadata Ontology (M3O) [24] is a core ontology developed following the approach above. The M3O allows for representing among others the annotation, decomposition, and provenance of multimedia metadata. As such, it is applicable to the use case of WeKnowIt and can be used in any other domain that needs to make use of semantic descriptions of media assets and multimedia content. The goal of the M3O is to integrate existing metadata standards and metadata formats rather than replacing them. To this end, the M3O provides a scaffold needed to represent multimedia metadata. A detailed description of the M3O can be found in Appendix A.2 of this document. Being an abstract model for multimedia metadata, it is not straightforward how to use and specialize the M3O for concrete application requirements and existing metadata formats and metadata standards. Thus, Section 3 introduces a step-by-step alignment method describing how to integrate and leverage existing multimedia metadata standards and metadata formats in the M3O in order to use them in a concrete application. The alignment method has been applied for three existing metadata models, namely the Core Ontology on Multimedia (COMM) [1], which is a formalization of the multimedia metadata standard MPEG-7, the Ontology for Media Resource [29] of the W3C, and the widely known industry standard EXIF for image metadata [16]. These example integrations are described in Appendix A.3 of this document.

Using ontological knowledge in object-oriented applications is a challenging task. This is due to the chasm that exists between the logics-based world of ontologies and the world of object-oriented software applications. However, using ontologies in software systems for knowledge representation and reasoning can be very useful and extend today's abilities of software systems.

Ontologies are among others used in work package 6 of the WeKnowIt project for developing the Application Programming Interface (API) of the WeKnowIt Data Storage (WKI DS) [6, 5]. The WKI DS API provides an object storage for large amounts of text, images and video content and a knowledge base for any kind of domain knowledge stored in the WeKnowIt system [5]. The WKI DS, in order to perform its functions, needs to maintain mappings between ontology concepts and object-oriented classes. Creation of such mappings, if done by hand, is very time-consuming as proved by a significant amount of changes introduced to the Common Model during the course of the WeKnowIt project. To alleviate this problem, a model-driven engineering approach for ontology application programming interfaces called OntoMDE is presented in Section 4. OntoMDE facilitates the use of ontologies in object-oriented software systems by providing a proper mapping of the logical concepts to the object-oriented application. It provides a multi-step mapping approach based on model-driven engineering (MDE) to overcome the gap between logics-based world of ontologies and the world of object-oriented software applications in order to provide pragmatic programming access to ontologies in software systems. A detailed example of applying the MDE approach for accessing ontological knowledge from object-oriented software systems is provided in Appendix A.4 of this document.

During an emergency incident, several different log files are created by members of the ER personnel to document the emergency events that occur throughout the incident. Managing and reviewing these logs is a critical task for understanding and improving the implemented ER actions. A major challenge arising in this task is the merging of log files that are created by the different members of the ER personnel for the incident under study. Extensive manual effort is necessary to identify critical information such as person names and locations in order to align and merge the incoming log entries to make them suitable for review. To alleviate this problem, the web-based application WERL (short for WeKnowIt ER Log Manager) [22] has been developed that facilitates the task of ER log merging and management by automatically aligning multiple log files and extracting ER-relevant semantic event information from log entry text. WERL has initially been description in deliverable D5.2.1 on the Prototypical Knowledge Management Methodology [27]. The description in this deliverable is a refinement and extension of the previous documentation and is presented in Section 5. The WERL application makes use of the representation features of the Event-Model-F [28, 25] in order to facilitate information sharing and reuse. Furthermore, WERL enables interactive exploration of the collected log files by means of temporal, location, and semantic filters. Preliminary evaluation of WERL by members of the Sheffield City Council Emergency Planning team confirm that the application provides them with enhanced support during the ER log management and reviewing process.

In Section 6, we investigate the combinability of the different knowledge representation patterns of the Event-Model-F. We provide examples of combining several patterns of the Event-Model-F in the WeKnowIt use case of emergency response but also refer to the use case of consumer social group.

2 Method for Designing Core Ontologies

Core ontologies are a specific kind of ontologies that can base on a foundational ontology like DOLCE [4, 12, 19]. A foundational ontology [4, 20] models the very basic and general concepts and relations that make up our world, e.g., objects, events, participation, and parthood. Through the use of a foundational ontology, core ontologies provide a precise semantics and rich axiomatization (cf. [20]). Core ontologies provide a refinement to foundational ontologies by adding detailed concepts and relations in their specific field. Well designed core ontologies make use of ontology design patterns to provide flexibility and modularize the knowledge modeled by the ontology. Similar to design patterns in software engineering [11], ontology design patterns provide a modeling solution to a recurrent modeling problem [13]. They make it easier for the ontology engineer to acquaint oneself with the ontology. By this, core ontologies are extensible with respect to adding new functionality and allow for reuse of structural as well as domain knowledge.

We have developed a method for designing core ontologies. This design approach for core ontologies is described in the following sections along the non-functional properties of core ontologies, namely axiomatization and formal precision, modularity, extensibility, reuseability, and separation of concerns (cf. [26]). An extensive example of demonstrating the use of this approach to combine several core ontologies and details on the design method are presented in Appendix A.1 of this document.

2.1 Axiomatization and Formal Precision

When designing an ontology, it is desirable to use a solid and sound modeling basis [20]. Thus, our approach foresees the use of a foundational ontology for designing the core ontology. We align the concepts and relations defined in the core ontology to the basic categories of human cognition investigated in philosophy, linguistics, and psychology [20]. These basic categories are manifested in the foundational ontology.

The alignment of the foundational ontology with the core ontology also includes the adoption and specialization of the formal semantics of the foundational ontology in the core ontology. While the axiomatization of a foundational ontology enables validating the upper-level semantics of the knowledge expressed with it, the alignment of the core ontology with the foundational ontology provides support for validating the more specific semantics of the concepts and relations defined in the core ontology.

A well-designed foundational ontology is very diligent with respect to the ontological choices to which it commits [20], e.g., the selection of the most abstract concepts that are modeled. Thus, when reusing such a foundational ontology for designing a core ontology the engineer is also requested to carefully think about his or her design choices [20]. Developing foundational ontologies is extremely hard but it needs to be conducted only once [20]. An ontology engineer should strive for applying a foundational ontology that has proven its usefulness when designing a new core ontology or domain ontology [20].

For designing our core ontologies, we have based them on the foundational ontology DOLCE and have carefully aligned our core ontologies with DOLCE's lightweight version DOLCE+DnS Ultralite (DUL)[1]. DOLCE aims at capturing the ontological categories underlying natural lan-

[1] http://ontologydesignpatterns.org/wiki/Ontology:DOLCE+DnS_Ultralite

guage and human common sense. It has a minimal core that includes only the most general concepts and patterns and is well suited for modularization. Additional theories such as Descriptions and Situations, Ontology on Information Objects, or the Ontology of Plans can be integrated when necessary [20]. We chose DOLCE as it already has proven to serve as good modeling basis for core ontologies. DOLCE has been successfully applied to design ontologies in different domains such as law, biomedicine, agriculture, and software components and web services [20].

2.2 Modularity

Even well designed core ontologies are usually large and cover more knowledge than might be required in a specific application domain [13]. Thus, concrete systems will commonly use only portions of it. In this case, it is hard to reuse only the "useful" pieces of such a monolithic core ontology [13].

Core ontologies (and domain ontologies) have a better design when applying ontology design patterns captured by the foundational ontology it uses [20]. By using ontology design patterns, it allows for selecting the parts of the ontology in a concrete application that are actually needed and used [13]. Thus, our design approach for core ontologies builds upon a foundational ontology that supports a pattern-oriented design approach [21]. DUL provides such a pattern-oriented approach.

Ontology design patterns shall not be too specific or too close to a particular domain. This would disallow the application of the pattern in other domains of the field covered by the core ontology. On the other hand, the patterns shall also not be too generic as reuse in a concrete domain would be hampered. A too generic pattern is hard to apply in a specific domain. The scope of a core ontology itself is defined through the scope of its patterns.

2.3 Extensibility

Foundational ontologies provide a high-level, abstract vocabulary of concepts and relations that are likely to be used in current and future application domains. In order to provide a solid basis for future extensions of core ontologies, a precise alignment of the concepts defined in a core ontology with the high-level concepts of a foundational ontology is conducted using our design approach. By this precise alignment, new patterns can be added to the core ontology without affecting the existing patterns. In addition to adding new patterns, the existing patterns of a core ontology can be extended. This is typically conducted by specializing the existing concepts and properties defined in the patterns. Finally, besides the already connected core ontologies within a specific application also further core ontologies can be developed and integrated if necessary.

2.4 Reuseability

For modeling complex, structured knowledge, reuse can happen on different levels, e.g., on the level of ontology design patterns, core ontologies, and domain-specific ontologies. Different patterns in the core ontology provide different descriptions of concepts defined in it. By splitting up core ontologies into different parts they allow for reusing the structured knowledge defined

within the core ontology's design patterns among different applications. This refers to the issue of extensibility discussed in Section 2.3.

In addition, the core ontologies can be combined with domain-specific knowledge. In the ideal case, domain ontologies reuse the ontology design patterns defined in core ontologies by specializations of the ontology design patterns [13]. However, our approach does not require that domain ontologies are based on a foundational ontology or a core ontology. In fact, we explicitly consider both options as it cannot be assumed that all domain ontologies are aligned with a foundational ontology or core ontology. This is achieved by using the Descriptions and Situations ontology design pattern of DOLCE. Here, the roles defined within a contextual situation can refer to a domain ontology that is either carefully aligned with DOLCE, aligned to the core ontology, or that is completely independent. In addition, also concepts classified by the role can be taken from some domain ontology.

2.5 Separation of Concerns

Our design approach supports the separation of concerns by defining the structured knowledge in the core ontology and leaving all domain-specific aspects out of it. This is achieved again with the Descriptions and Situations ontology design pattern. The structured knowledge of the concrete field of the core ontology is captured by its ontology design patterns, e.g., participation, causality, and documentation for the Event-Model-F, the annotation and decomposition in the Core Ontology on Multimedia (COMM) [1], and the communication pattern in X-COSIMO [10]. This structured knowledge is specified by the roles defined within these patterns, i.e., the `defines` relations of the `Descriptions`. The domain knowledge is only referred to by the roles classifying the events and objects used. By this, the core ontologies are independent of any concrete domain that makes use of the concepts defined by them. In addition, core ontologies provide support to include individuals defined in some domain ontologies.

3 Aligning existing Metadata Models with the Multimedia Metadata Ontology (M3O)

For representing multimedia metadata, we have developed the Multimedia Metadata Ontology (M3O) [24]. It allows for representing among others the annotation, decomposition, and provenance of multimedia metadata. The M3O is an abstract model for multimedia metadata, designed to integrate existing metadata models and metadata formats rather than replacing them. It is useful in any application domain that needs to deal with digital media assets and multimedia content like the use cases of WeKnowIt. Details of the M3O are presented in Appendix A.2 of this report.

This section introduces a method for aligning existing multimedia standards and multimedia formats with the M3O. In contrast to automatic, adaptive, or machine learning approaches for ontology alignment [8, 7, 2], we conduct a pure manual alignment. Only a manual alignment ensures the high quality of the integration and minimizes ambiguities and imprecise matching. We consider the time and effort for manual alignment manageable as we assume that each metadata format or standard has to be aligned only once and that updates to the integrated formats or standards will be infrequent and mostly incremental.

For the alignment, we propose an iterative four-step alignment method that helps ontology engineers to integrate existing metadata standards and metadata formats. In each iteration, we consecutively evolve the alignment of the format or standard with the M3O. Following an iterative approach, we are able to identify, analyze, and flexibly react to problems and challenges encountered during previous iterations of the alignment.

Each iteration consists of four steps. The first step targets the understanding of the format or standard to be integrated. The second step reorders concepts into coherent groups. The third step maps concepts and structure with the M3O. The fourth step proves and documents the validity of the alignment, and finalizes the iteration.

To introduce our alignment method, we proceed as follows: For each step, we first outline the goals and provide a brief summarization. Subsequently, we describe the core tasks to be performed within the step. The alignment method has been used to align three existing ontologies with the M3O, namely the Core Ontology on Multimedia (COMM) [1], the Ontology for Media Resource [29], and EXIF [16]. For details please refer to Appendix A.3 of this document.

3.1 Step 1: Understanding

A precise understanding of the metadata format or standard to be integrated is an important prerequisite for aligning it with the M3O. Consequently, the first step of alignment is an in-depth analysis of the structure and core concepts of the model at hand. While this advise may seem obvious, this is a task easily underestimated and problems neglected at an early stage can cause time-consuming problems along the integration process. Additional documentation, if available, will help to (re-)produce the overall structure not explicitly expressed in the formal specification. In general, we have found three distinct modeling approaches to be very common for multimedia metadata formats and metadata standards.

Pattern-based Pattern-based ontologies, e.g., COMM, provide a high degree of axiomatization and structure in a formal and precise way.

Predicate-centric In a predicate-centric approach as followed, e.g., by the Ontology for Media Resource, the ontology is mainly specified through a set of properties. Such a model offers very little structure in a machine readable format, e.g., in terms of conceptual relations between properties.

Legacy Models Other formats and standards have not yet been semantified at all.

Ambiguities that are found during the initial analysis are discussed at this point. It is not our intention to revise all modeling decisions made for the ontology to be integrated. However, we consider the alignment a good opportunity to correct some of the *bad smells* [9] discovered. Once we have reached a sufficient understanding of the format or standard to be integrated, we proceed with the grouping step.

3.2 Step 2: Grouping

Ontologies should provide structural information on the relation and groupings of concepts it defines. However, although many formats or standards provide this information in their documentation, the information is sometimes lost when the models are transformed to an ontology. By using the original specifications and documentations, we are able to preserve and recreate this grouping information, and provide them through formal specification in the aligned ontology. In principle, we distinguish three forms of available grouping information:

Explicit Grouping Pattern-based models provide an explicit grouping of concepts into coherent patterns, often accompanied by a rich axiomatization on how they relate.

Implicit Grouping For other metadata models grouping information may not be explicitly represented. This is often the case with predicate-centric approaches, e.g., the Ontology on Media Resource. In these cases, we refer to the textual documentation in order to reconstruct the implicit groupings of the properties or classes.

Recovery of Groupings In other cases grouping information is lost when transferring multimedia formats or standards to the Resource Description Framework (RDF)[2]. For example the EXIF metadata standard provides textual descriptions about groupings, e.g., in terms of pixel composition and geo location. However, this distinction got lost in the adaption to an RDF schema [18].

Once we have provided all relevant grouping relations through a formal specification, we continue with the mapping step.

3.3 Step 3: Mapping

This step achieves the mapping of the ontology's concepts and structure to the scaffold provided by the M3O. The goal of this step is to create a working ontology, which, after validation, can be published or used as basis for further iterations. For the alignment, we follow a sequence of the following three steps:

1. Mapping of Concepts If some superclass of the concept to be aligned is present in both ontologies, direct mapping of concepts is feasible. This is mainly the case for ontologies

[2] http://www.w3.org/RDF/

that share the same foundation, e.g., COMM and the M3O, which both base on the DUL foundational ontology. All axioms of the aligned concepts are preserved as long as they are applicable through the M3O. If a concept is not applicable in the M3O, we align all dependent subclasses and references to the nearest matching M3O concept.

2. **Structural Mapping** For structural mapping, we consider the functionality of the pattern or structure to be mapped. If a pattern or structure offers the same or an equal functionality than a pattern of the M3O, we can replace the pattern. By adapting the M3O pattern, we are often able to express the same functionality using a more generic approach.

3. **Removing Unnecessary Concepts** We finalize the mapping step by cleaning up unused dependencies from the ontology files. Concepts that either have no further relevance for the target context or are sufficiently covered by the M3O are removed at this point.

3.4 Step 4: Validation and Documentation

In each iteration of the alignment process, we need to check the consistency of the resulting ontology. This can be done by using a reasoner like Fact++[3] or Pellet[4]. Any problem encountered during the alignment can be resolved by reiterating the four steps of the alignment method. After proving the consistency of the resulting ontology, we finalize the process by documenting all major decisions and adjustments made during the alignment.

[3] http://owl.man.ac.uk/factplusplus/
[4] http://clarkparsia.com/pellet/

4 Model-Driven Software Engineering Support for Ontology APIs

In this section, we present the Model-driven Engineering (MDE) approach for the generation of programming access to ontologies called OntoMDE. It is motivated from the development of the Application Programming Interface (API) of the WeKnowIt Data Storage (WKI DS) [6, 5]. After implementing a first version of the WKI DS API, new requirements to the ontologies had to be taken into account by the WeKnowIt system. As such, the knowledge base of the WKI DS changed and the corresponding API had to be updated. In order to support the implementation of the WKI DS API, OntoMDE supports the developers by providing a model-driven approach for API generation. Figure 1 depicts the different steps of our OntoMDE approach. Our mapping includes three different transformations and two intermediate models, before it finally ends up in code of the application programming interface (API) of the ontology.

Section 4.1 introduces the basic concepts and principles used in OntoMDE. The first step of the OntoMDE approach is described in Section 4.2, where the ontology is transformed in a Model for Ontologies (MoOn) representation. The MoOn bases on a modification of the ECore Metamodel for OWL2[5] and the OWL-to-UML mappings described in [17, 15, 23]. We add some additional annotations to the MoOn to be able to represent information, e.g., about the concept classifications or the semantic unit specification. The second step described in Section 4.3 is the transformation from the MoOn to the Ontology API Model (OAM). The OAM is a UML class diagram of the programming access API. We have extended this model to embed information for the code generation process, e.g., to tailor the concrete API to a particular repository backend. The last step described in Section 4.4 is the generation of code from the OAM.

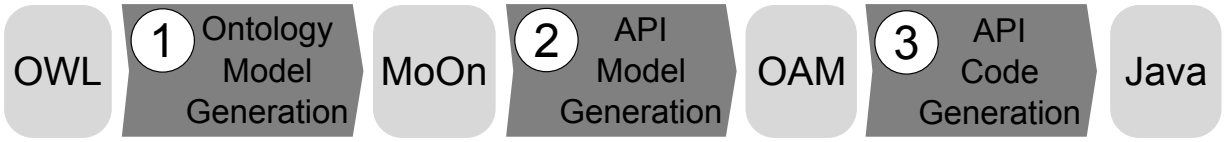


Figure 1: Model-driven API Generation Process OntoMDE

4.1 Terminology

For our OntoMDE approach for model-driven ontology API engineering, we need to introduce some terminology. Our terminology consists of four terms:

- A ***Semantic Unit*** is a triple $SU = (CO, SO, R)$ that consists of a set of *content concepts* CO, a set of *structure concepts* SO and a set of relations R between these concepts. An ontology can include multiple *semantic units*. Each *semantic unit* groups the concepts and relations for a distinct aspect of the application.
- A ***Pragmatic Unit*** groups all entities of a software system appurtenant to a specific semantic unit SU. It is represented through a tuple $PU = (C, P)$ consisting of a set of classes C and properties P.

[5]MOF-Based Metamodel for OWL2 http://www.w3.org/2007/OWL/wiki/MOF-Based_Metamodel

- ***Content Concepts*** are those concepts of a *semantic unit*, which provide the *content* the underlying pragmatic works on. Subsequently, we call *content* those entities an application, user, or developer works on. *Content concepts* can be used in multiple *semantic units*.
- ***Structure Concepts*** are all concepts of a *semantic unit* not providing content but are of structural matter for the knowledge representation. *Structure concepts* are specific to a single *semantic unit*.

4.2 Step 1: From Ontology T-Box to MoOn

In this section, we present the first intermediate model the MoOn and the transformation from an OWL-based ontology definition to MoOn. The MoOn is independent from the concrete DL language used, only the transformation is OWL2 specific. To map the single OWL constructs and their properties, we have to perform several preparation steps, these are:

- Ontologies often specialize or use concepts and properties defined in other ontologies. We have to deal with these **imports** of the ontology. To generate a MoOn for further transformations, we enrich our ontology representation with imported concepts and properties directly used in our target ontology.
- Ontologies often provide particular knowledge only in an implicit form, e.g., subsumption. We have to **inference** this implicit knowledge and make it explicit, in order to be able to map it into the MoOn. We use reasoning services to inference this implicit knowledge.
- A very important step in the preparation of the concept and property mapping, is to **de-anonymize** the constructs in the ontology. Due to the fact that object orientation does not provide for anonymous classes, we have to avoid anonymous concepts. In DL-based languages like OWL, we can easily substitute anonymous concepts with named ones. Let us assume a concept specification in the form $A \equiv (B \sqcap C) \sqcup (D \sqcap E)$. In this definition $(B \sqcap C)$ and $(D \sqcap E)$ are anonymous concepts. We de-anonymize them by introducing two new concepts $BandC \equiv B \sqcap C$ and $DandE \equiv D \sqcap E$ and substitution in the original concept specification, to get $A \equiv BandC \sqcup DandE$.
- Now that we have enriched the ontology representation with all relevant concepts and properties, we are able to create an **inheritance** structure suitable for an object-oriented representation. In OWL ontologies, we often find very complex inheritance structures, always ending up in $Thing$. In an object-oriented representation, we need a much simpler inheritance structure to support broader type declarations for constructors and methods. To adapt the inheritance structure in the MoOn to our needs, we search for least common subsumers of multiple concepts in the set of concept relevant for the MoOn. We add such subsuming concepts to this set of concepts relevant for the MoOn.

After these steps, we can transform all the relevant concepts and properties to the MoOn. We use the mappings defined in [17, 15, 23]. We integrate a mapping for number restrictions as cardinalities. When the mapping is finished, we are able to enrich this ontology model with additional information relevant for the mapping to object-orientation. For example, information about the definition of the *semantic units* and the classification of the concepts in *structure concepts* and *content concepts*.

We start with the *semantic unit* definition: With defined *semantic units* it is possible to classify the concepts in a semi-automatic process. To encapsulate concepts of a particular *semantic unit*,

we use the package mechanism of ECore. Currently, we only support user-driven *semantic unit* specification.

The next step is to classify the concepts into *structure concepts* or *content concepts*: We use ECore annotations to declare the classification of a concept in the MoOn. This step could be performed semi-automatically by preselecting all concepts involved in different *semantic units* as *content concepts*. This mechanism performs well for most of the complex ontologies like the M3O as well as for less complex ontologies. In less complex ontologies, like the Ontology for Media Resource of the W3C Media Annotations Working Group nearly all concepts provide content and we find less *structure concepts*. We use a user-driven refinement of the concept classification to support such ontologies properly.

The MoOn provides for further customizations of the previous steps. Based on the MoOn, it is possible to integrate additional concepts in the inheritance structure whether if they are from the ontology or imported. It is possible to delete single concepts or superconcepts from a *semantic unit* or from the MoOn.

4.3 Step 2: From MoOn to OAM

In this section, we present the second intermediate model, the OAM and the transformation from MoOn to OAM. The OAM is an object-oriented model of the resulting API. The API will be generated from the OAM in the subsequent step. All special characteristics and properties of the intended API are defined in this model. The UML2 class diagram is the underlying metamodel of the OAM. We have extended this metamodel with light-weighted profiles. In the profiles, we define stereotypes that could be attached to different entities in the model. These stereotypes provide additional information used during code generation.

First, we generate interfaces from the inheritance structure in the MoOn. The basic content an application works on will get its own class representations in the API. We add a class for each *content concept* implementing the relevant interfaces created in the previous step.

We generate a *pragmatic unit* class for each *semantic units* defined in the MoOn. For each *structure concept*, we add an attribute to the pragmatic unit class it belongs to. The pragmatic unit classes encapsulate the structure concepts and hide them from the API user. For each *content concept* in a *semantic unit*, we add an attribute of the type of the *content concept* class. To the added attributes, we attach the cardinalities from the related MoOn properties. For a summary of the different mappings between the ontology and the final API see the Appendix A.4.

In order to integrate classes from arbitrary legacy APIs, the OntoMDE approach allows to model such classes in OAM. A user does not have to model the whole class with all of its properties and operations, just a prototype of the class with a special stereotype. As a consequence, our new class can inherit from this modeled legacy API class model.

[5] http://mawg.joanneum.at/web/mawg.html
[5] http://www.omg.org/technology/documents/profile_catalog.htm

4.4 Step 3: Generating an API from the OAM

In the last step, we generate code from the API representation in the OAM. We can use standard code generation functionalities for arbitrary programming languages. Customizations of the generated code towards a persistence layer implies that we have computed the necessary meta information in a previous step. Once this has been done, we can generate a fully functional API for pragmatic programming access to the dedicated ontology.

5 WeKnowIt Emergency Response Log Merger (WERL)

An important task in the function of professional Emergency Response (ER) organisations is the creation, management and reviewing of log files that document the actions taken by members of the ER personnel during the course of an incident. The resulting log files constitute valuable source of information for members of the ER personnel for two functions:

- Decision making during the course of an incident: based on the collected log entries in the control room of the organisation, tactical and strategic decisions need to be made until the incident is resolved,.
- Evaluation of the implemented actions after the incident has been resolved based on reviewing the collected log entries.

Log files are generated by members of the ER personnel that are dispatched at the site(s) where the incident takes place. Due to the complexity and scale of several emergency situations, it is a frequent case to have numerous individuals dispatched at different locations. Each of them deals with a different aspect of the incident at hand and thus provides an isolated view of the incident through his/her log entries. In the end, multiple log files end up in the control room of the organisation and there is a profound need for assessing them together in order to form a complete awareness of the incident as a whole and determine the next actions in an optimal way.

Currently, this task is performed manually at the time the log entries arrive in the control room. Obviously, this is a labour intensive process that allocates precious human resources that would otherwise be exploited for the analysis of the incident. Furthermore, custom reports are created from the log files that summarise the main entities participating in the event: persons, locations and objects. Their extraction is also a manual process requiring additional human effort.

To this end, the WeKnowIt ER Log Manager (WERL) provides a novel ER Log Merging and Management application, addressing the aforementioned problems arising in the management of ER log files by automatically merging and aligning log entries produced by different members of the ER personnel during an incident. In addition, it automatically extracts semantic information from the text of each log entry, i.e., an event that happens in the course of an incident. By this, WERL enables a concise view of the important content in log files. It allows its users to interactively explore the analysed log files by online temporal, location-based and semantic filtering, resulting in enhanced reviewing capabilities for decision making and evaluation. Finally, the application makes use of the Event-Model-F [28] as a representation framework for the structure and content of the events described by the ER log files. As the Event-Model-F provides a formal specification of events and event relations, it also makes log file content shareable and reusable. Figure 2 presents a high-level view of the WERL application. The individual components depicted in the figure (Event-Model-F representation, Semantic enrichment and WERL front-end) are described in the corresponding deliverable sections that follow. This work, which extends the initial ER log management application presented in deliverable D5.2.1 [27], was published in the proceedings of the Events in MultiMedia (EiMM) workshop colocated with the ACM MultiMedia 2010 [22].

5.1 Emergency Response Event Representation

Before describing the adopted representation formalism, it is first necessary to describe the typical format of log files used within modern ER organisations. Although there is no standardised

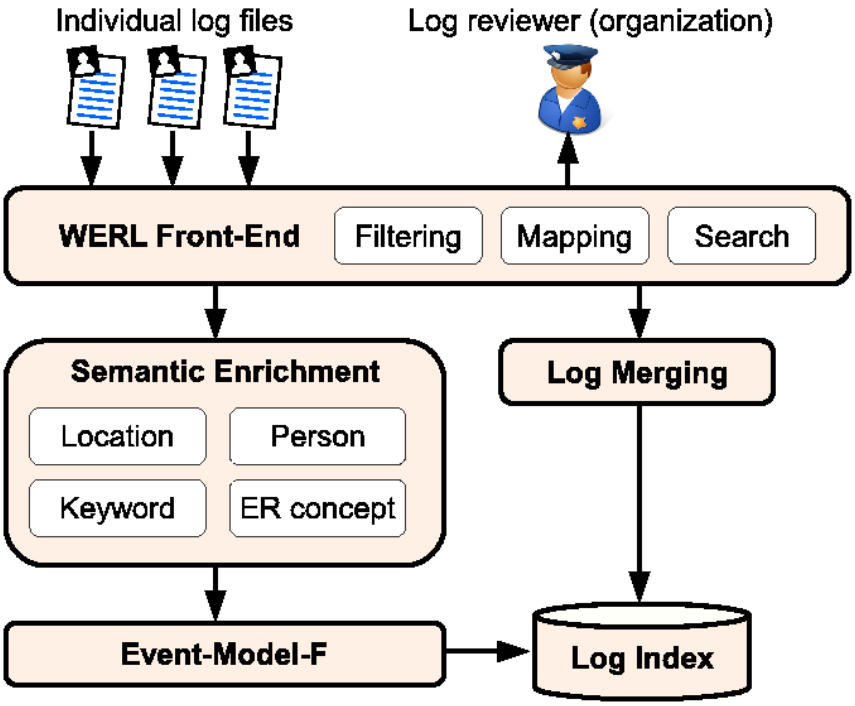


Figure 2: WERL Application Architecture

format among all ER organisations, the basic elements of ER log files are commonly present independent of the particular organisation or unit. An example log file, which was produced by the Emergency Planning Team of the Sheffield City Council, is illustrated in Figure 3.[6] The file comprises a header with information regarding the whole log file and a table of log entries that document in detail the communication details and actions of the ER personnel members involved in the incident. The log file header contains authoring information, such as log file creator, unit to which the log creator belongs, incident date and title. The individual log entries contain four fields, namely

- the time that the particular log entry was recorded,
- the source and/or destination of the message between members of the ER personnel,
- the content of the recorded message, and
- an action associated with the message.

To represent the entries in the emergency response log files, we employ the Event-Model-F [28]. The Event-Model-F is a pattern-based core ontology that bases on the foundational ontology DOLCE+DnS Ultralight [3]. It provides a formal representation of events and event relations such as participation, composition, causality, and others and has been successfully applied to the development of the SemaPlorer++ application [27] for the distributed creation and sharing of event information in emergency response. As such, it is a suitable means to represent the event information stored in the incident log files and log entries. We consider log entries as events that happen in an emergency incident. To represent these entries, we employ the participation pattern of the Event-Model-F to model the participation of living and non-living objects in the events and the documentation pattern to attach further metadata and documentary evidence to

[6] All log snapshots presented here have been fully anonymised to protect the identity and privacy of the people mentioned in the respective log files.

SHEFFIELD CITY COUNCIL - SCHEME FOR MAJOR PEACETIME EMERGENCIES

INCIDENT LOG SHEET

NAME: Person1 JOB TITLE: Role1

SERVICE AREA AND DIRECTORATE: MIRG

DATE: 15th September 2006 INCIDENT: Acid incident an Ecclesfield School

TIME	FROM/TO (Name, Position, Org.)	MESSAGE	ACTION
15.38	Person2 - via email	Possible Incident Alert	To stand by for further information
17.20	Person3 via telephone	Call out re Incident at Ecclesfield	Asked to go to Longley
17.40	Person3 / Person2	Confirmed to go to E.P.T.	
18.36	to Person2 - via email	To say arrived at EPO	
18.49	to Person2	Requesting update on WRVS and if on site (see copy email)	WRVS on site with Refreshments. Report to Person4
18.53	to Person2 - via email	Press briefing - media needing No. of evacuees to release	
18.49	to Person2 - via email	Details of explosion awaiting delivery of sand to build bunker	Other areas in South Yorkshire contacted
18.50	Via Control Room Email to Person2	Gold have had a briefing Capacity to board up & protect Gas/electricity isolated on site Will move from a police response to a L19 response Person5 with the lighting at the scene so could explode during darkness	

Figure 3: Example of Log File for Emergency Response

support the event. Each entry is considered part of a larger incident that is covered by the log file itself. Thus, we employ the composition pattern of the Event-Model-F to express that the events described by the single log entries are component events of a larger emergency incident event.

A graphical depiction of this representation is shown in Figure 4. Both the log files and the log entries are represented by an URI (incidentURI), a date/time of occurrence (dateTime), an identifier of their location (locationURI) and a set of documentation properties (additionalER-Properties). The properties describing an incident log file are:

- Incident title: a short description of the incident that triggered the emergency response procedure,
- Logger: the author of the log,
- Service area: the division of emergency response organisation, and
- Job role: the logger's role in the particular incident.

while the ones needed for a log entry description are:

- From-To: the entities identified as the sender and receiver of the message of this entry,
- Message: the message to be delivered, and
- Action: the action that needs to be done.

Below, we provide an example of how to represent a log file using the Event-Model-F. Figure 5(a) shows the modelling of the incident log file itself, an acid incident at a school in Sheffield. The representation of the first two records of the log file, i.e., the first two sub-events of the incident are depicted in Figures 5(b) and 5(c).

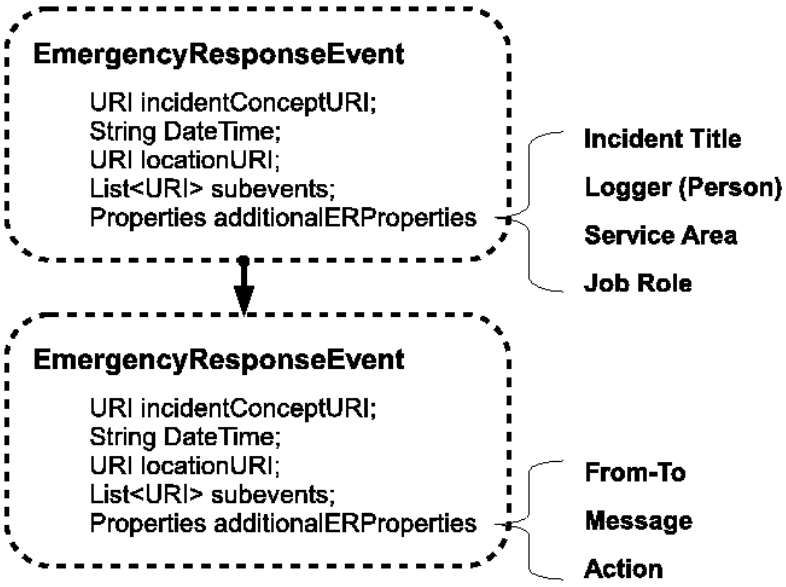


Figure 4: Event-Model-F Emergency Response Event Representation

logevent#1
URI incidentConceptURI = "**wki:events:logevent#1**"
String DateTime = "**15th September 2006**"
URI locationURI = "**geoplanet:woeid:19283**"
ArrayList<URI> subevents
"**wki:events:logentryevent#1**"
"**wki:events:logentryevent#2**"
Properties additionalERProperties
IncidentTitle = "**Acid Incident an Ecclesfield School**"
Logger = **Person1**
ServiceArea = "**MIRG**"
JobTitle = **Role1**

(a) Top-level event log

logentryevent#1
URI incidentConceptURI = "**wki:events:logentryevent#1**"
String DateTime = "**15th September 2006, 15.38**"
URI locationURI = "**geoplanet:woeid:19283**"
ArrayList<URI> subevents
"**wki:entities:person#Person2**"
"**wki:concepts:er#alert**"
Properties additionalERProperties
From-To = "**Person2 – via email**"
Message = "**Possible Incident Alert**"
Action = "**To stand by for further information**"

(b) First log entry

logentryevent#2
URI incidentConceptURI = "**wki:events:logentryevent#2**"
String DateTime = "**15th September 2006, 17.20**"
URI locationURI = "**geoplanet:woeid:19283**"
ArrayList<URI> subevents
"**wki:entities:person#Person3**"
Properties additionalERProperties
From-To = "**Person3 via telephone**"
Message = "**Call out re Incident at Ecclesfield**"
Action = "**Asked to go to Longley**"

(c) Second log entry

Figure 5: Instantiation of Two Log Entries in WERL

5.2 Semantic Enrichment

In order to facilitate the interpretation of the log entries, and determine their interrelationship, the key concepts/entities in the logs are extracted. The extracted concepts provide the information concerning the where (locations), who (person names) and what (keywords/phrases) of the incident. As training data is not available to learn the extraction patterns and inconsistencies are observed in the log entries, in terms of linguistic and syntactic style, the extraction processes applies a knowledge-intensive approach. This requires that high quality gazetteers/taxonomies are available, which are used to annotate logs with potential named-entities, manually derived extraction patterns can then utilise the text surrounding the potential annotations to determine the final semantic annotations. For a more detailed description of the text processing and rule-based document annotator see [14].

Each of the following subsections describes the techniques used in the three areas of information extraction. This information is stored according to the representation model introduced above.

5.2.1 Location

For emergency incidents there is a requirement to identify relatively fine-grained locations (i.e. at the street level). Thus, a specific Sheffield-centric gazetteer was developed. To this end, a number of high-quality, freely available (for non-commercial use), professionally produced resources were used. However, these resources are UK-specific. In order to provide generic coverage, it would be possible to use the collaborative constructed OpenStreetMap[7] data. Whilst this was considered at the time of development, it did not provide comparable levels of accuracy and completeness for the Sheffield area. Given the continual and intense development of OpenStreetMap, it is envisaged that it will develop into a comparable resource to professionally produced resources. Indeed it was widely used by a number of professional organisations during the recent Haiti Disaster[8].

During the location identification process the gazetteer place names are matched with the log text, taking into consideration the use of common abbreviations such as, rd=road, st=street, etc. However, even after identifying a location name, ambiguity may still exist due to multiple occurrences of the name, e.g. there are 12 locations called School Lane. In order to display the locations on a map, a disambiguation process attempts to uniquely identify the location. The process constructs a probability distribution of the potential locations based on the other (contextual) non-ambiguous locations in the log. The probability is increased if these contextual locations do not conflict with the potential location, i.e., they are within some distance threshold or the potential location is within the non-ambiguous location, if it refers to a wider area. If a single location receives the highest probability, given the context, it is selected as the actual location. If no unique location is identified, it is then a task of the reader to disambiguate the location. Ideally, potential ambiguities should be identified to the writer of the log so that they could disambiguate the location ensuring the correct interpretation of the log.

[7] http://www.openstreetmap.org/
[8] http://wiki.openstreetmap.org/wiki/WikiProject_Haiti

5.2.2 Person Names

The identification of person names applies a gazetteer of first names and common titles (e.g. Mr, Mrs, Dr, etc.) and case-sensitive patterns to identify likely entities. The two main issues with the process are to ensure that the gazetteer has the appropriate coverage of names/titles used in the logs, and dealing with ambiguous names. Name ambiguity can be caused by the use of common words as names, e.g. Heather, Rose, Hector, however it is presumed that the frequency of such ambiguous entities in the logs will be low. In addition names are commonly used in location names, e.g. John Street, in such cases the location takes precedence.

5.2.3 Keywords and Phrases

There are two processes employed in the identification of the keywords and phrases in the incident logs: a general term extractor utilises freely available web-services, whilst specific emergency incident terminology is extracted using an ER thesaurus.

Three web services which provide general terminology extraction were examined: Zemanta[9], OpenCalais[10], and Yahoo! Term Extraction (YTE)[11]. Both Zemanta and OpenCalais provide semantic annotations of text, extracting concepts and linking these to specific web content (from Wikipedia, Youtube, IMDB, etc.) identified by their unique URI. As these services are aimed at writers of general web content their response is limited to Wikipedia related concepts, to limit the amount of noise. YTE simply provides a list of extracted terms/phrases. There is no clear indication of the data and technologies used to derive these key terms/phrases. However, examining the results on a number of logs indicated that it tends to produce more comprehensive and pertinent extraction.

There is no standardised thesaurus for ER terminology. A number of organisations involved in Emergency Management publish glossaries of terms and acronyms which can be used to provide more specific terminology extraction. The current implementation is limited to the identification of ER acronyms, such as FLO standing for Forward Liaison Officer.

5.3 User Interface of WERL

WERL was developed after discussions with members of the Emergency Planning team of Sheffield City Council who made clear the need for an automated means of merging disparate log files and searching in them by use of pertinent criteria, such as location, person name, and ER-specific keywords.

The front-end of WERL provides online filtering capabilities for facilitating the interactive exploration of the available log entries. A snapshot of the application main screen is provided in Figure 6. At the top, a slider-based time filter is available that enables the examination of a particular time interval of the incident. In addition, standard full text search capabilities are provided for retrieving only the subset of log entries that are relevant to the input query. Most importantly, there is a series of four semantic filters that summarise the main entities found in the

[9] http://www.zemanta.com/
[10] http://www.opencalais.com/
[11] http://developer.yahoo.com/search/content/V1/termExtraction.html

log files by the text annotation component described in Section 5.2. Thus, it is possible to view only the log entries that are related to a particular location, person name, significant keyword or ER acronym. The presentation of all identified semantic entities in these lists can provide the ER user with a quick overview of the semantic content of the log file. The filtering and search operations are performed on the client side so that the log reviewing process is highly responsive and the application scales to many users by limiting server-side processing to the semantic enrichment of the input log files.

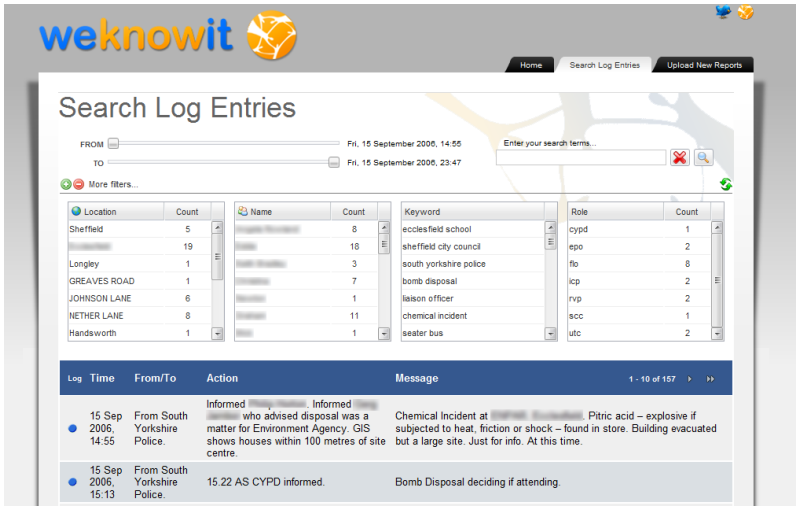


Figure 6: Snapshot of WERL Front Screen

A significant feature of WERL is the presentation of provenance information and the possibility to filter based on the provenance of log entries. As illustrated in Figure 7, beside each log entry there is a marker indicating its origin. At the bottom of the log entry list, there is an associated legend, which can also be used for filtering based on the log entry provenance. In that way, it is possible to inspect only the log entries produced by a particular log creator, thus gaining insight into his/her perspective of the incident. In larger scale incidents, involving many members of ER personel coming from different organisations (e.g. fire department, police), it is expected that more sophisticated provenance mechanisms will be necessary, e.g. provenance by organisation, unit, role in the organisation, etc.

Finally, a valuable WERL feature is the possibility to link from the log entries directly on the map of the area where the ER incident has taken place. Figure 8 depicts this capability. Such a feature is possible due to the automatic localisation of text providing geo-coordinates, as described in Section 5.2.1. Such visualisations aim to improve the ER personnel's overall situational awareness of the incident and thus allow them to make more informed decisions with respect to the next actions.

5.4 Evaluation

During the development of the WERL application in the WeKnowIt project, there have been two experts from the Emergency Planning team of the Sheffield City Council involved. The experts had access to the online application and we provided them with basic usage instructions such that they can work with the application. We discussed the application with the experts in face-to-face meetings during project meetings and received feedback on a regular basis.

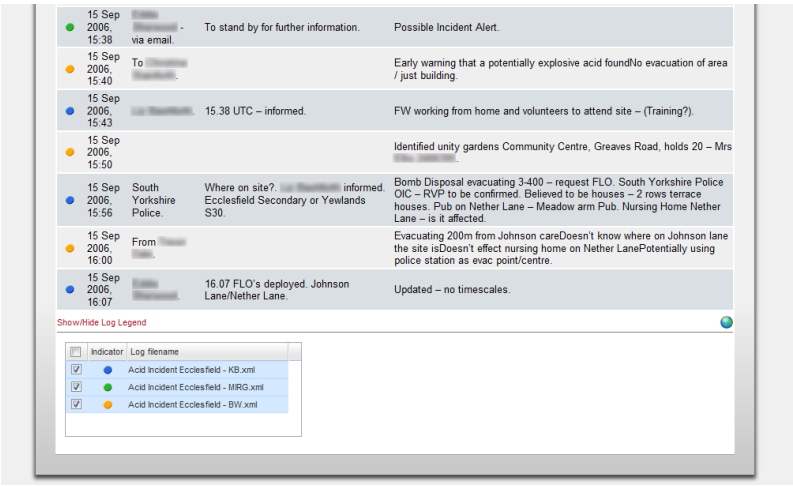


Figure 7: Snapshot of WERL Provenance Filtering

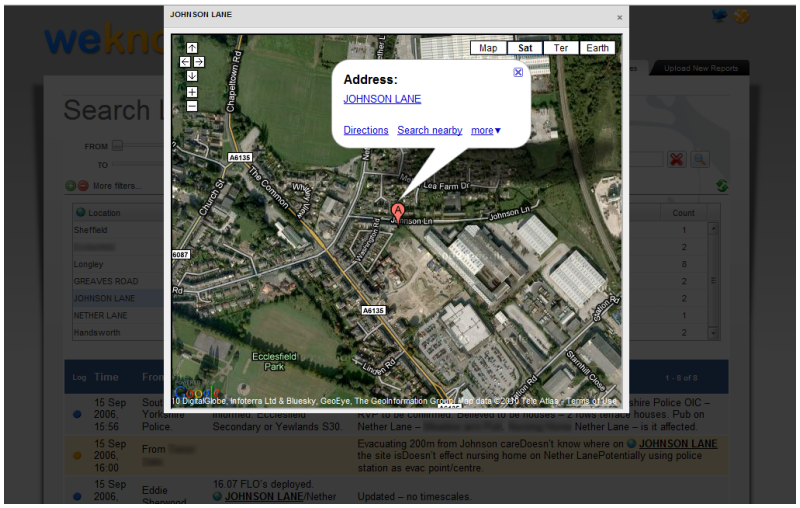


Figure 8: Snapshot of WERL Map View

For the current implementation of the WERL application, we have received an experience report by one expert of the Emergency Planning team documenting several attributes of the application as well as points for improvement and desirable features. Other than that, the pilot test was conducted in a completely unsupervised way. The general impression was that the application was easy to use and presented a number of advantages both during and post incident phase of log management. The expert expressed a desire to test the system with a larger dataset (for testing a single incident covered by three log files was available). With respect to the search facilities, the expert noted that they worked well, but there were cases where the search produced wider results than expected. On the other hand, it would be desirable for the search to pick up misspellings and typing errors, which are not uncommon in ER logs, since they are typed under time pressure.

In terms of features, the expert found useful the possibility to jump from a location reference in the text to its map-based view. In addition, he noted that it would be useful to view on the map all locations contained in the log entries under examination. An additional desirable feature for future versions of the application is the possibility to create summary reports from the log (which is supported by state-of-the-art incident management systems).

6 Combining the Event-Model-F Patterns

The Event-Model-F [28, 25] is a core ontology developed in the WeKnowIt project for modeling events and event relations such as participation of living and non-living objects in events, compositional relationships between events, causality, and correlation. In addition, it allows the annotation of events by means of some documentary evidence such as photos taken at the event. Generally speaking, documentary evidence can be any kind of objects but also some other events can be used to document an event. Finally, the Event-Model-F allows for explicitly modeling different interpretations of events, i.e., representing different point of views on the same happening. For a detailed description of the Event-Model-F and its patterns, we refer to [28, 25].

Like the design of the M3O, the design of the core ontology Event-Model-F follows the approach presented in Section 2. The Event-Model-F is based on the foundational ontology DOLCE+DnS Ultralight and makes use of ontology design patterns. Thus, the different features of the core ontology are implemented using different patterns. The Event-Model-F provides six patterns for representing events and event relationships, namely the Participation Pattern, Mereologic Pattern (for event compositions), Causality Pattern, Correlation Pattern, Documentation Pattern, and Interpretation Pattern. A particular strength of the design approach followed with the design of the core ontology is the combineability of the different patterns. The WERL application presented in Section 5 is an example where a combination of the Participation Pattern and Composition Pattern has been used together. The matrix depicted in Figure 9 gives an overview of the different, possible combinations of the patterns in the Event-Model-F. The x- and y-axis show the different patterns of the Event-Model-F, namely Participation Pattern, Composition Pattern, Causality Pattern, Correlation Pattern, Documentation Pattern, and Interpretation Pattern. A green tick means that the patterns are combinable. A yellow tilde indicates that the two patterns can be combined under some certain condition. Finally, a red cross stands for non-combineable patterns.

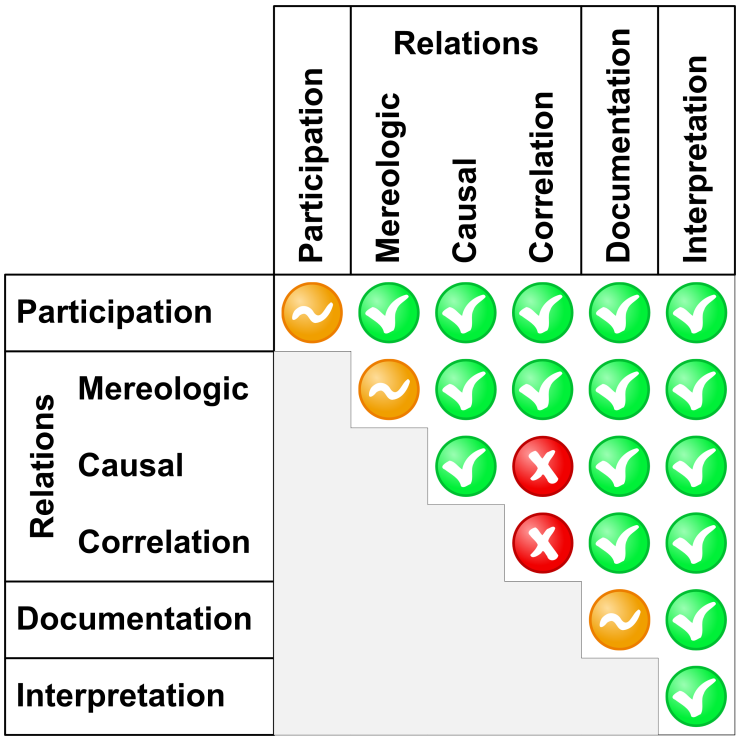


Figure 9: Combineability of the Different Patterns in the Event-Model-F

In the following, we discuss the possible combinations of the Event-Model-F patterns along the matrix depicted in Figure 9. We start with the first row and go through the columns from left to right. We only need to consider the combinations of the diagonal and above the diagonal, as it makes in principle no difference if an event that is described in, e.g., a Participation Pattern first is subsequently used in another pattern like the Mereology Pattern or vice versa. Thus, all combinations of patterns below the diagonal of the matrix shown in Figure 9 are not further considered in the remainder.

The first combination of Event-Model-F patterns is using the Participation Pattern together with (one or more) other Participation Pattern. The Participation Pattern in general is applied to express which objects participate in an event such as persons and non-living things, where the event happened, and when it happened. Applying two (or more) participation patterns on the same event can be used to express that different sources of information provide different information about which objects participate in the event, when it happened, and where. This combination of multiple Participation Patterns can be in principle conducted. However, its full potential is only enfolded when used together with the Interpretation Pattern, i.e., providing different views onto the same happening (see below).

The event described in a Participation Pattern can play the role of a composite or component in the Mereology Pattern. As composite, the event described by the Participation Pattern would be, e.g., a soccer match. The Participation Pattern then models the players participating in the match as well as its location and time. The Composition Pattern would decompose the composite event of the soccer match, e.g., along time into the event of the first half and the second half. Such composite events can be again described using the Participation Pattern to model, e.g., the players that have participated in the first half and second half, respectively.

Events described by a Participation Pattern can also be used in the Causality Pattern, either as cause event or effect event. For example, in the course of a storm there is a heavy rainfall that causes a dam to burst. Here, the event of a heavy rainfall is the cause for the event of a dam burst. The cause event of the heavy rainfall can be further described using the Participation Pattern to model, e.g., that the object of rain is participating in it at a specific place and happening at a specific period of time. The effect event of the dam burst can be described by the Participation Pattern with the objects located around the dam when the burst happened.

In a different scenario, a patient is undergoing a medical examination. Here, several correlate events can be identified such as measuring the higher blood pressure, finding out about an increasing irritation of the skin, and the patient complaining of some pain. Each of these events of the observed higher blood pressure, the increase of skin irritation, and the complain of some pain happen at the same time. Thus, they are correlate events. However, there is no causal relationship between these events. They are all effects of some other (yet unknown) causal event. The patient might have eaten something wrong, has too much stress, or something else. The correlate events can be further described using the Participation Pattern like that a specific patient is involved. In the event of observing the higher blood pressure, it might be modeled using the Participation Pattern that the patient had to take a specific medicine. In addition, it might be modeled using the Participation Pattern that there are a certain kind of bacteria involved in the event of the increasing skin irritation.

The Documentation Pattern can be used together with the Participation Pattern to document, e.g., which persons have participated in a certain event. This can be provided, e.g., by some photos taken during the dam burst or protocol that has been created during the patient examination.

As said above, the Interpretation Pattern can be used to define specific views onto events. For example, in the event of a bar fight there might be a person A saying that person B started and person C participated whereas person B is saying that person A has started and person C has not participated. This is modeled using several instantiations of the Participation Pattern (see above). The Interpretation Pattern is then used to compile the different Participation Pattern(s) and potentially other patterns like Causality Pattern and Documentation Pattern to provide the different views onto the specific event of a bar fight.

Starting with the second row, one can say that the Mereology Pattern can be applied together, either on the same level or hierarchically. Applying multiple Mereology Patterns on the same level means that one provides, e.g., different compositions of the same composite event. This is useful to model different compositions of the same event in combination with the Interpretation Pattern (see also the application of multiple Participation Pattern above). Applying the Mereology Pattern hierarchically means that an event that plays the role of a component event in one instance of the Mereology Pattern can be a composite event (i.e., play the role of a composite event) in another instance of the Mereology Pattern. This can be conducted arbitrarily often. Thus, there can be multiple layers of hierarchically applied Mereology Patterns. For example, the event of a soccer game can be divided by applying the Mereology Pattern into a first half and a second half. The first half can be again divided into several sub-events such as the foul that lead to a red card and so on.

The composite event and the component event of the Mereology Pattern can be a cause event in the Causality Pattern. For example, the composite event of an important soccer game during the FIFA World Cup is the cause for the streets being empty in the two participating countries. The component event of a foul in the soccer game is the cause for the component event of a red card in the game. Thus, the red card is an effect event.

In the case of the patient suffering from some disease, different correlate events are observed like measuring the higher blood pressure, observing the increasing skin irritation over time, and the patient complaining of some pain. These correlate events can also be understood as some composite events that happen during the event of the patient examination. A single correlation event can also be a composite event in the Mereology Pattern. For example, the increase of the patient's skin irritation is observed through several measurements (events where the medical doctor has looked at the skin irritation) that have been conducted over a period of time.

Composite events and component events can be documented using the Documentation Pattern. For example, the composite events in the soccer game and the patient examination for the increasing skin irritation can be documented using some photographs.

In addition, composite events and component events of the Mereology Pattern can play the role of an event that is interpreted using the interpretation pattern. For example, the composite event of the soccer game is interpreted differently by the media coverage in the participating countries. This applies in particular with respect to the component events of the foul and red card.

As there can be chains of causal relations, multiple Causality Pattern can be applied together. For example, in the soccer game the foul leads to the red card which might further lead to a quarrel among the players and coaches of the teams.

Correlate events are defined as events that are effects of some common causal relationship, although this cause is (initially) unknown [28]. If this causal relationship is discovered, e.g., the reason why the patient is suffering from the blood pressure, skin irritation, and pain, it can

be modeled explicitly using the Causality Pattern. However, when the common cause of the correlates is known it does not make sense to apply the Correlation Pattern any longer.

Cause events and effect events can be documented using the Documentation Pattern. For example, the cause event of the foul as well the effect event of the red card is documented by the photographers and journalists attending the game.

The events playing the role of a cause or an effect in the Causality Pattern can also be used in the Interpretation Pattern. For example, the cause event of a flooding, i.e., the heavy rainfall, can be interpreted using the Interpretation Pattern using different Participation Pattern, and Documentation Pattern. The effect event of the dam burst can be used in the Interpretation Pattern using different instances of a Participation Pattern, Mereology Pattern and Documentation Pattern.

As said, correlate events are defined as events that are effects of some common cause [28]. However, there is no causal relationship among the different correlate events. As such, it does not make sense to apply multiple Correlate Patterns together. Rather, all correlate events should be modeled using a single Correlate Pattern. Correlate events can be documented using the Documentation Pattern. For example, the examination of the patient's skin irritation can be documented using some photographs (show above).

A correlate event of a Correlation Pattern can also play the role of an event interpreted in the Interpretation Pattern. For example, there can be different interpretation of the skin irritations of the patient, i.e., different Participation Patterns describing the bacteria, chemical reactions, etc. happening in the skin of the patient.

It makes sense to apply several Documentation Patterns together, if an event that is playing the role of a documenter, i.e., documenting some other event, is the documented event in some other Documentation Pattern. For example, some journalists are taking pictures of the flooding event and risking their lives when the dam bursts. Those journalists are producing documentary evidence of the dam burst during the flooding. Some other person is observing this life risking situation and is taking some images of those journalists. Thus, here the event of observing the journalists and the objects, i.e., the photographs, created during this event, are documentary evidence for the situation of journalists risking their life are work. Multiple Documentation Pattern can be used to provide different documentary support for the same event. In this case, however, rather a single Documentation Pattern or only a few consolidated Documentation Pattern should be used.

An event that is documented can be interpreted in different ways using the Interpretation Pattern. For example, different Participation Patterns can be applied using the Interpretation Pattern in order to argue that a particular image is showing a specific person or not.

Finally, there can be several interpretation patterns of the same event. For example, in the case of a power outage during the flooding event different emergency control officers might have differing interpretations what has caused the event of the power outage [28]. One officer might believe the power outage is caused by a snapped power pole whereas the other officer considers a serious damage at the power plant as cause of the power outage. Thus, the event of the power outage and the objects participating in this event is interpreted from different points of view.

Besides the combination of patterns of the Event-Model-F only, the patterns of the Event-Model-F can also be combined with patterns of other core ontologies. For example, the combination of the Participation Pattern, Documentation Pattern, Causality Pattern, and Interpretation Pattern of the Event-Model-F together with patterns of the core ontologies COMM and X-COSIMO in

the use case of emergency response is described in Appendix A.1 of this document. The use of the Participation Pattern of the Event-Model-F together with the Annotation Pattern of the core ontology M3O [24] for annotating an image showing the atomic cloud over Nagasaki with information about the event of the atomic bombing and its participants is shown in Appendix A.2.

Further examples of combining several patterns of the Event-Model-F are documented on the web site[12] and provided as OWL ontologies. One example is about the WeKnowIt use case of a consumer social group. It describes a two-day weekend trip of a group of friends. On the first day, there is a sub-event of a dinner. On the second day are two sub-events, a visit to a museum and a travel to a sight. The tourism example applies different patterns of the Event-Model-F such as participation and composition.

A second example models a soccer game with a first halftime and second halftime. As described above, different events are happening during the game such as a foul and goal. The soccer example makes full use of all patterns of the Event-Model-F, namely participation, causality, correlation, composition, and interpretation. It further shows how a domain specific ontology can be embedded and used to describe the events happening during a soccer game.

[12] west.uni-koblenz.de/eventmodel

7 Conclusions

In this document, we have presented the final knowledge management methodology of the WeKnowIt project. We have presented a method for modeling core ontologies as specific kind of ontologies that are based on foundational ontologies and make use of ontology design patterns. Core ontologies are specifically designed to integrate heterogeneous data sources.

Based on this design method, e.g., the Multimedia Metadata Ontology (M3O) has been developed in the WeKnowIt project. It allows for representing among others the annotation, decomposition, and provenance of multimedia metadata. As the M3O is very generic, it is not straightforward how to use it in concrete applications. Thus, a method is presented to align existing metadata formats and metadata models with the M3O.

Using ontologies in software systems can be very useful and extend today's abilities of software systems. However, due to the chasm that exists between the logics-based world of ontologies and the world of object-oriented software applications it makes it hard to use ontologies in software systems as the use of ontologies for the development of the WeKnowIt Data Storage shows. To alleviate this problem, a model-driven engineering approach for ontology application programming interfaces called OntoMDE has been presented.

Finally, we have presented the web-based application WERL (short for WeKnowIt ER Log Manager) for automatically merging and managing emergency response log files. WERL makes use of the ontology design patterns of the Event-Model-F [28, 25] in order to facilitate information sharing and reuse. It allows for exploring the log files interactively by means of temporal, location, and semantic filters. We have also investigated the combinability of the different knowledge representation patterns of the Event-Model-F and showed their combination at the example of the WeKnowIt use case of emergency response but also to the use case of consumer social group.

8 References

[1] Richard Arndt, Raphaël Troncy, Steffen Staab, Lynda Hardman, and Miroslav Vacura. COMM: designing a well-founded multimedia ontology for the web. In *ISWC+ASWC*, pages 30–43, 2007.

[2] Eva Blomqvist. Ontocase-automatic ontology enrichment based on ontology design patterns. In *International Semantic Web Conference*, pages 65–80, 2009.

[3] Stefano Borgo and Claudio Masolo. Foundational choices in dolce. In Steffen Staab and Ruder Studer, editors, *Handbook on Ontologies*. Springer, second edition, 2009.

[4] Stefano Borgo and Claudio Masolo. *Handbook on Ontologies*, chapter Foundational choices in DOLCE. Springer, 2nd edition, 2009.

[5] A. Boruch, W. Ciesielski, R. Janik, T. Kaczanowski, J. Kwiecinska, A. Scherp, M. Sieprawski, and D. Zbik. D6.3: Design, architecture and implementation of knowledge base. Technical report, WeKnowIt, 2009.

[6] A. Boruch, R. Janik, T. Kaczanowski, P. Wesolowski, and F. Schwagereit. D6.2.2: Definition and implementation of apis, version 2. Technical report, WeKnowIt, 2010.

[7] Marc Ehrig. *Ontology Alignment: Bridging the Semantic Gap*, volume 4 of *Semantic Web and Beyond*. Springer, 2007.

[8] Jérôme Euzenat and Pavel Shvaiko. *Ontology matching*. Springer, 2007.

[9] Martin Fowler, Kent Beck, John Brant, William Opdyke, and Don Roberts. *Refactoring: Improving the Design of Existing Code*. Addison Wesley, 1999.

[10] Thomas Franz, Steffen Staab, and Richard Arndt. The X-COSIM integration framework for a seamless semantic desktop. In *Knowledge capture*, pages 143–150. ACM, 2007.

[11] Erich Gamma. *Design patterns : elements of reusable object oriented software*. Addison-Wesley, 2007.

[12] Aldo Gangemi, Nicola Guarino, Claudio Masolo, Alessandro Oltramari, and Luc Schneider. Sweetening Ontologies with DOLCE. In *Knowledge Engineering and Knowledge Management*, pages 166–181. Springer, 2002.

[13] Aldo Gangemi and Valentina Presutti. *Handbook on Ontologies*, chapter Ontology Design Patterns. Springer, 2nd edition, 2009.

[14] Mark A. Greenwood, José Iria, and Fabio Ciravegna. Saxon: an extensible multimedia annotator. In *LREC*, 2008.

[15] L. Hart and P. Emery. OWL Full and UML 2.0 Compared. http://uk.builder.com/whitepapers/0and39026692and60093347p-39001028qand00.htm, 2004.

[16] Japan Electronics and Information Technology Industries Association. Exchangeable image file format for digital still cameras: Exif version 2.2, 2002.

[17] Aditya Kalyanpur, Daniel Jiménez Pastor, Steve Battle, and Julian A. Padget. Automatic Mapping of OWL Ontologies into Java. In *SEKE*, 2004.

[18] Masahide Kanzaki. Exif vocabulary workspace - rdf schema, 2003. http://www.w3.org/2003/12/exif/, last update in 2007.

[19] Claudio Masolo, Stefano Borgo, Aldo Gangemi, Nicola Guarino, and Alessandro Oltramari. WonderWeb deliverable D18 ontology library; IST WonderWeb project, 2003.

[20] Daniel Oberle. *Semantic Management of Middleware*. Springer, 2006.

[21] Daniel Oberle, Steffen Lamparter, S. Grimm, D. Vrandečić, S. Staab, and A. Gangemi. Towards ontologies for formalizing modularization and communication in large software systems. *Appl. Ontol.*, 1(2):163–202, 2006.

[22] Symeon Papadopoulos, Ansgar Scherp, Neil Ireson, Ioannis Tsampoulatidis, and Yiannis Kompatsiaris. Using event representation and semantic enrichment for managing and reviewing emergency incident logs. In *Proceedings of the 2nd ACM international workshop on Events in multimedia*, EiMM '10, pages 41–46, New York, NY, USA, 2010. ACM.

[23] Tirdad Rahmani, Daniel Oberle, and Marco Dahms. An adjustable transformation from OWL to Ecore. In Dorina C. Petriu, Nicolas Rouquette, and ystein Haugen, editors, *MoDELS (2)*, volume 6395 of *Lecture Notes in Computer Science*, pages 243–257. Springer, 2010.

[24] Carsten Saathoff and Ansgar Scherp. Unlocking the semantics of multimedia presentations in the web with the multimedia metadata ontology. In *WWW '10*, pages 831–840. ACM, 2010.

[25] Ansgar Scherp, Thomas Franz, Carsten Saathoff, and Steffen Staab. F—A Model of Events based on the Foundational Ontology DOLCE+ Ultralight. In *Knowledge Capturing*, pages 137–144, 2009.

[26] Ansgar Scherp, Thomas Franz, Carsten Saathoff, and Steffen Staab. F-a model of events based on the foundational ontology DOLCE+DnS Ultralight. In *Accepted for publication at the International Conference on Knowledge Capturing (K-CAP), Redondo Beach, CA, USA*, September 2009.

[27] Ansgar Scherp, Symeon Papadopoulos, Apostolos Kritikos, Felix Schwagereit, Carsten Saathoff, Thomas Franz, Daniel Schmeiss, Steffen Staab, Simon Schenk, and Matteo Bonifacio. D5.2.1: Prototypical knowledge management methodology. Technical report, WeKnowIt, 2010.

[28] Ansgar Scherp, Carsten Saathoff, Thomas Franz, and Steffen Staab. A core ontology on events for the semantic representation of human experiences in the real world. *Multimedia Tools and Applications*, 2010. Online First, http://www.springerlink.com/content/72013082v64lv5n5/.

[29] W3C. Ontology for media resource 1.0, 2010. http://www.w3.org/TR/mediaont-10/.

Appendices

A Appendix

A.1 Appendix on Designing Core Ontologies

Designing Core Ontologies

Ansgar Scherp, Carsten Saathoff, Thomas Franz, and Steffen Staab
WeST, University of Koblenz-Landau, Germany
E-mail: {scherp|saathoff|franz|staab}@uni-koblenz.de

Abstract

One of the key factors that hinders integration of distributed, heterogeneous information systems is the lack of a formal basis for modeling the complex, structured knowledge that is to be exchanged. To alleviate this situation, we present an approach based on core ontologies. Core ontologies are characterized by a high degree of axiomatization and formal precision. This is achieved by basing on a foundational ontology. In addition, core ontologies should follow a pattern-oriented design approach. By this, they are modular and extensible. Core ontologies allow for reusing the structured knowledge they define as well as integrating existing domain knowledge. The structured knowledge of the core ontologies is clearly separated from the domain-specific knowledge. Such core ontologies allow for both formally conceptualize their particular fields and to be flexibly combined to cover the needs of concrete, complex application domains. Over the last years, we have developed three independent core ontologies for events and objects, multimedia annotations, and personal information management. In this paper, we present the simultaneous use and integration of our core ontologies at the example of a complex, distributed socio-technical system of emergency response. We describe our design approach for core ontologies and discuss the lessons learned in designing them. Finally, we elaborate on the beauty aspects of our core ontologies.

1 Introduction

Domains that require the exchange of a high amount of complex, structured knowledge such as medical systems, distributed media management, and emergency response have a high pressure for systems integration in order to facilitate efficient communication and information exchange. For example, in emergency response different entities such as an emergency hotline, police department, fire department, and emergency control center are involved. These entities need to exchange among others knowledge about what happened during an incident, tasks communicated between the entities, and media information that documents the incident. Due to the lack of appropriately integrated systems at the different emergency response entities, the complex, structured knowledge is currently exchanged via natural language on the phone. This is very error-prone and inefficient. Rather, the different, heterogeneous systems used by the emergency response entities should be integrated to provide a more efficient and effective exchange of the knowledge. One of the key factors that hinders integration of these systems is

the lack of a formal basis for modeling the complex, structured knowledge that is to be exchanged. So far, this problem has not been solved due to the lack of networked ontologies that provide a flexible means to model the complex structure of the knowledge exchanged and at the same time provide a formal semantics to that structure.

In this paper, we propose an approach based on core ontologies to alleviate this situation. An ontology allows for formally representing the relevant concepts and relations of a considered domain in a machine readable format (Oberle et al., 2009b; Oberle, 2006). Core ontologies provide a precise definition of structural knowledge in a specific field that spans across different application domains, e.g., software services, personal information management, knowledge organization, multimedia annotations, and others (Oberle, 2006). They combine a number of specific properties that have been derived from reported experiences in designing core ontologies (Oberle et al., 2007, 2006; Oberle, 2006) and the development of our own core ontologies (Arndt et al., 2009; Scherp et al., 2009a; Staab et al., 2008; Arndt et al., 2007; Franz et al., 2007). These properties are axiomatization and formal precision, modularity, extensibility, reuseability, and separation of concerns.

Axiomatization and Formal Precision. A high degree of axiomatization and formal precision is provided by core ontologies. By this, a common understanding in a particular field is established in order to ensure interoperability through machine accessible semantics. Systems can reason about the represented knowledge and carry out semantic checks on its validity. The axiomatization and formal precision is achieved by basing on a foundational ontology.

Modularity. Core ontologies should follow a pattern-oriented design approach. By this, they are modular within the field for which they are designed. Similar to design patterns in software engineering (Gamma, 2007), ontology design patterns provide a modeling solution to a recurrent ontology design problem (Gangemi and Presutti, 2009). The core ontology is a composition of such ontology design patterns with appropriate dependencies between the patterns (Gangemi and Presutti, 2009). This enables a pathway for extensibility and reuseability.

Extensibility. Being modular, a core ontology allows for adding new and updating or removing modules, i.e., ontology design patterns it defines. By this, the core ontology is able to reflect system evolution (cf. adaptability in (Vrandečić, 2009)). It is extensible towards new developments and functional requirements that arise.

Reuseability. Different systems are built for different purposes and users in different domains. Being modular, a core ontology supports reuse of its modules, i.e., the ontology design patterns despite of the different foci and domains the concrete systems have. At the same time a core ontology still guarantees formal precision of the overall knowledge it represents. In addition to the reuse of the domain-independent, structured knowledge defined by the core ontologies themselves, also existing domain knowledge can be reused. Core ontologies are able to incorporate existing domain ontologies and make use of that domain knowledge rather than requiring to remodel it.

Separation of Concerns. The structural knowledge defined in a core ontology is clearly separated from the domain-specific knowledge. This allows core ontologies to be applied in arbitrary application domains. Domain-specific knowledge such as a domain ontology on emergency response or sports can be integrated and reused without affecting the core ontology itself.

In this paper, we will show that combing these properties in a core ontology can lead to elegant solutions and interoperability in complex application domains. Due to the characteristics of their design such core ontologies can be flexibly combined to cover the needs of concrete, complex application domains. Thus, from our perspective they are to be considered *beautiful* ontologies.

Over the last years, we have developed three of those beautiful core ontologies. These core ontologies have been used over a long time, are very stable with respect to their design, and thus provide a sustainable solution for ensuring interoperability in complex socio-technical systems such as emergency response. These core ontologies are the Event-Model-F, COMM, and X-COSIMO. The core ontology Event-Model-F is designed for modeling events and objects (Scherp et al., 2009a). It allows for representing human experience and participation in real world occurrences and provides comprehensive support for modeling time and space, objects and persons, as well as mereological, causal, and correlative event relationships and event interpretations. The Core Ontology on Multimedia (COMM) (Arndt et al., 2009; Staab et al., 2008; Arndt et al., 2007) is designed for describing arbitrary digital media data. It allows for (semantic) annotations of media data and their decompositions. Finally, the Cross-Context Semantic Information Management Ontology (X-COSIMO) is designed for semantic information management and communication (Franz et al., 2007). It supports modeling the communication taking place between different persons and systems and the information associated with it such as task descriptions.

Our three core ontologies, i.e., the Event-Model-F, COMM, and X-COSIMO are based on the Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE) (Gangemi et al., 2002; Masolo et al., 2003). As foundational ontology, DOLCE aims at modeling the very basic and general concepts and relations (Borgo and Masolo, 2009; Oberle, 2006) that make up our world, e.g., objects, events, participation, and parthood. Foundational ontologies are generic across many fields (Oberle, 2006). They have a large scope and are highly reusable in different modeling scenarios (Borgo and Masolo, 2009). By their nature, foundational ontologies are much broader than core ontologies such as our Event-Model-F, COMM, and X-COSIMO. Core ontologies provide a refinement to foundational ontologies by adding detailed concepts and relations in their specific field. DOLCE already has proved to be a good modeling basis for core ontologies such as (Scherp et al., 2009a; Arndt et al., 2007; Franz et al., 2007; Oberle et al., 2006, 2007).

The Event-Model-F, COMM, and X-COSIMO have been carefully aligned with the foundational ontology DOLCE+DnS Ultralight[1] (DUL), a lightweight version of DOLCE. By this alignment, our core ontologies can be flexibly combined to cover the needs of complex application domains. Our core ontologies follow a pattern-oriented

[1] `http://ontologydesignpatterns.org/wiki/Ontology:DOLCE+DnS_Ultralite`

ontology design approach, i.e., they define a set of ontology design patterns targeted for the specific field they model. These patterns are based on the very generic patterns DUL provides such as the Descriptions and Situations (DnS) pattern and the Information Object (IO) pattern (Borgo and Masolo, 2009). By using a foundational ontology and following a pattern-oriented design approach, the core ontologies possess a solid, semantically precise basis. At the same time these core ontologies become modular and extensible with respect to their use in concrete applications and to changes in functional requirements. By applying the DnS pattern, our core ontologies allow for a clear separation of the structured knowledge captured by the core ontology and the domain knowledge provided by a domain ontology. Thus, they allow for integrating and reusing existing domain ontologies.

The remainder of the paper is organized as follows: In the next section, we motivate the need for core ontologies to model complex, structured knowledge by presenting a scenario of a complex, socio-technical system in the domain of emergency response. In Section 3, we demonstrate the simultaneous use and smooth interplay of our three core ontologies Event-Model-F, COMM, and X-COSIMO in the emergency response scenario. It demonstrates the use of the three core ontologies to model the complex, structured knowledge that needs to be exchanged between the different systems involved. The properties of core ontologies and our design approach for developing such core ontologies are presented in detail in Section 4. The concrete design of our three core ontologies the Event-Model-F, COMM, and X-COSIMO is presented in Section 5. In Section 6, we discuss the lessons learned when designing and using our core ontologies. We argue for the beauty of our core ontologies in Section 7, which lies in their ability to both formally conceptualize their particular fields and to be flexibly combined to cover the needs of concrete scenarios, before we conclude the paper.

2 Modeling and Sharing Complex, Structured Knowledge in Emergency Response

In the emergency response scenario of the EU project WeKnowIt[2] depicted in Figure 1 different emergency response entities are involved using different, heterogeneous systems. These systems need to exchange complex, structured knowledge that needs to be shared among the emergency response entities. Examples of emergency response entities are the emergency hotline, police department, fire department, emergency control center, and forward liaison officers. The emergency control center is in charge of coordinating the emergency response entities. It receives event descriptions from the emergency hotline, processes them, and communicates event descriptions with the police department and fire department. In addition, the emergency control center forwards event descriptions together with task descriptions to their forward liaison officers. Forward liaison officers are out in the field to report about a situation, verifying it, and documenting it, e.g., by taking photos and notes. As the following scenario shows, this socio-technical system for emergency response becomes very active in the case of an incident. Many different pieces of structured knowledge such as events, tasks, and

[2] http://www.weknowit.eu/

multimedia data with metadata are created, combined, and communicated between the different emergency response entities involved. Subsequently, we discuss how our core ontologies are involved in modeling this complex, structured knowledge.

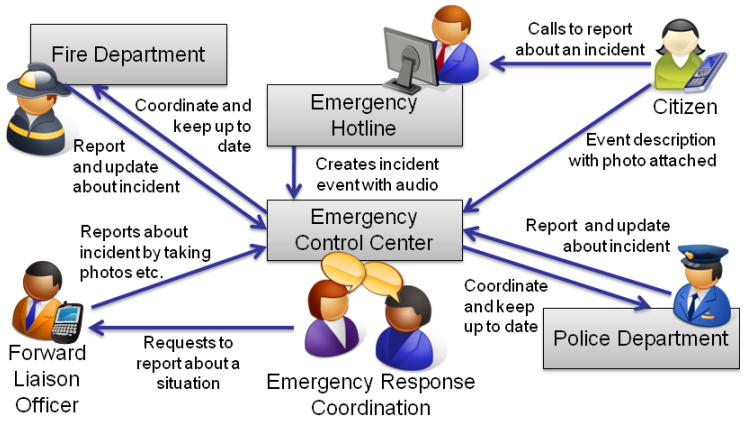


Figure 1: A distributed system for emergency response

2.1 Scenario

In an incident of a heavy storm a major flooding may happen. During the flooding a power outage occurs. Some citizens are lacking power supply and are calling the emergency hotline to report about the power outage event. The officers at the emergency hotline record these calls. In addition, they type in a textual description for each call to document the reported event in their system. The recorded calls are automatically processed by some speech recognition techniques, which creates a transcript of the calls. The algorithm also automatically distinguishes the different voices of the participants in the phone call and can automatically associate and differentiate utterances made by the citizen and the officer at the emergency hotline. Subsequently, the event description together with the processed call recording and its transcript are automatically transferred to the system of the emergency control center. The emergency control center also receives event descriptions from the systems of the police department and fire department that happen during the incident. Based on the evidence of the event descriptions, the officers in the emergency control center use their system to formulate hypothetical events that might have caused the power outage. To this end, event descriptions are analyzed, (semi-)automatically clustered, visualized, and put into relation. The officers conclude that there are two possible interpretations that might have caused the power outage, namely a snapped power pole close to the river that was reported by a citizen calling the emergency hotline or a serious problem with the power plant nearby. The correct assessment of the situation is very crucial in order to most effectively deploy the available emergency response resources. Thus, the different event interpretations modeled in the system need to be verified by the officers as soon as possible in order to confirm or reject the hypothesis. For this purpose, the officers in the

emergency control center may contact the personnel of the power plant. At the same time, the description of the hypothetical event of a snapped power pole together with a task description is sent to the mobile device of a forward liaison officer. The forward liaison officer receives the task description. She drives to the location of the events to verify it and to document it by taking photos and notes. The photos can also be tagged or otherwise annotated. The event description extended by annotated photos are send back to the system of the emergency control center.

As the scenario shows, the different entities involved in an emergency response have to share structured knowledge among each other through the different systems they use. The structured knowledge is a combination of event descriptions that happen during an incident, e.g., provided by the citizens calling the emergency hotline or annotated by forward liaison officers when verifying a situation. It also comprises task descriptions that are communicated and shared, e.g., the forward liaison officers receive task descriptions (together with the event description) to clarify a specific situation at a certain place and to document about it by taking pictures and notes. Finally, these pictures and notes taken and the metadata that is attached to the media is another kind of structured knowledge that is communicated within the distributed socio-technical system for emergency response shown in Figure 1. As the discussion shows, the structured knowledge that is to be communicated is quite complex. In addition, the entities involved with this structured knowledge, i.e., the events, media data, and tasks appear and are relevant in different contextual settings that need to be modeled. For example, an event reported by a citizen to the emergency hotline and represented appropriately may become an attachment of a message with a task description that is sent by the emergency control center to one of their forward liaison officers.

2.2 Involved Ontologies

To model the complex, structured knowledge in the emergency response scenario, i.e., the events, media data, and tasks we use and combine our three core ontologies Event-Model-F, COMM, and X-COSIMO. For representing events and the multiple relationships between them, we use the Event-Model-F (Scherp et al., 2009a) that has been developed in the WeKnowIt project. The Event-Model-F provides a formal representation of the different aspects of events in which humans participate such as time and space, composition, correlation, and documentation. Compared to existing models, the Event-Model-F differs in providing sophisticated support for modeling causality, correlation, and interpretation of events.

The Core Ontology on Multimedia (COMM) (Arndt et al., 2009; Staab et al., 2008; Arndt et al., 2007) allows to represent arbitrary digital media data such as images, videos, and audio. It supports the different kinds of annotations of media data and their decomposition into segments. COMM is highly influenced by and specifically designed to support the low-level descriptors of MPEG-7 (MPEG-7, 2001). Its roots go back to the EU project aceMedia[3], where a first attempt to model a MPEG-7 ontology has been undertaken. This ontology is not based on a direct translation of MPEG-7 but on an analysis of the MPEG-7 standard. It follows a formal approach for model-

[3] http://www.acemedia.org/

ing the multimedia annotation domain based on DOLCE (Bloehdorn et al., 2005). In the EU Network of Excellence K-Space[4], this idea of analyzing MPEG-7 in order to model a formal core ontology for multimedia has been taken up again. In contrast to the aceMedia approach, the COMM developed in K-Space is further axiomatized and based on ontology design patterns in order to acquire an easier to use and formally more sound model.

Finally, the Cross-Context Semantic Information Management Ontology (X-COSIMO) supports for modeling semantic information management and communication (Franz et al., 2007). X-COSIMO allows to represent the communication taking place between different persons and systems and the information associated with this communication like a task description. The core ontology has been developed in the X-Media project[5], which is dedicated to research on large scale and cross-media knowledge management solutions. In the X-Media project also COMM has been used and extended. Both core ontologies COMM and X-COSIMO play a key role in the shared representation of automatically extracted and newly created information in the X-Media project. Among others, the shared representation is exploited in user interfaces that enable users to deal with the diversity of the knowledge represented.

2.3 Summary

We have shown the requirement of sharing complex, structured knowledge at the example of a socio-technical system for emergency response. The structured knowledge to be modeled and exchanged within this scenario are the representation of events and objects, multimedia data and its annotations, and personal information management such as communication and tasks. In the next section, we present how such structured knowledge can be modeled with the core ontologies Event-Model-F, COMM, and X-COSIMO we have developed. We demonstrate the use of our core ontologies at the example of the emergency response scenario of the WeKnowIt project.

3 Modeling the Emergency Response Scenario

Referring to the scenario of the distributed socio-technical system for emergency response presented in Section 2, we exemplify in this section how the structured knowledge that is exchanged in this system can be modeled. We pick out a small part of the scenario and fully represent it by applying the core ontologies we have developed, namely the Event-Model-F for modeling events and objects (Scherp et al., 2009a), COMM for representing multimedia annotations (Arndt et al., 2007), and X-COSIMO for personal information management and communication (Franz et al., 2007). With modeling the scenario, we show the interplay of these core ontologies, before we discuss the properties of core ontologies in Section 4 and their concrete design in Section 5.

In the following, we consider a power outage that has happened in the course of a major flooding. Many citizens are calling the emergency response hotline such as

[4] http://kspace.qmul.net/

[5] http://www.x-media-project.org/

Paul. He calls the emergency hotline to report about an observation he made, a power pole in his street has just snapped. Shortly after, the power outage happens. Thus, Paul reports to the hotline that he thinks that the snapped power pole has caused the power outage. The officer Rita at the emergency hotline answers Paul's call. She types into her system what Paul reports, while also an automatic recording of the conversation is taken. In our ontologies, the citizen Paul is represented by the individual `paul-1` and the power pole is represented by the individual `power-pole-1`. We model the event when the power pole snapped as the individual `snapped-pp-1`, the event of the power outage as `power-outage-1`, and the event in which Paul calls the hotline as `call-1`. The officer Rita working at the emergency hotline is represented by the individual `rita-1`. She is answering the `call-1`. The overall flooding event is referenced as `flooding-1`.

Using our Event-Model-F, we model the participation of the person `paul-1` in the event of a `snapped-pp-1` as shown in Figure 2 by applying the core ontology's participation pattern. The pattern is based on the generic ontology design pattern Descriptions and Situations (DnS) (Gangemi, 2008; Gangemi and Mika, 2003). The ontology design pattern DnS provides an ontological formalization of context (Oberle, 2006; Gangemi and Mika, 2003). It allows for a formally precise representation of different, contextualized views by defining roles. Thus, besides representing the participation of a person in an event, the participation pattern shown in Figure 2 also defines that Paul plays the role of a citizen in this participation, indicated with `paul-citizen-1` which is of concept `CitizenRole`. In another situation, Paul might have a different role, e.g., if he is besides being a citizen also a professional firefighter. Thus, Paul can play the role of a `FiremanRole` in other events. It is important to note that both the `CitizenRole` and the `FiremanRole` are not defined within the participation pattern of the Event-Model-F. However, they are provided from some external, domain-specific ontologies. Thus, the participation pattern and the Event-Model-F in general allows to reuse existing domain knowledge. The use of the DnS pattern in an ontology such as the Event-Model-F can be easily recognized. It always defines a situation that satisfies a description. The situation includes the events and objects of a concrete contextual situation, i.e., the real-world entities that can be observed in a concrete situation. The description defines the roles of these events and objects in the observed situation.

In the concrete example, the situation `part-sit-snapped-pp-1` is an `EventParticipationSituation` that `satisfies` the description `part-desc-snapped-pp-1`, which is an `EventParticipation-Description`. The individual `desc-ev-snapped-pp-1` classifies the real-world event of the snapped power pole `snapped-pp-1`, which is of interest, i.e., described in the considered situation. In addition, we can model the time of the event and location of Paul when participating in the event. This is not shown in the figure, but available online as OWL ontology from our ontologies website: `http://west.uni-koblenz.de/Research/ontologies/`.

The phone call between Paul and Rita is recorded to document the event. The information that the call has been recorded is represented by the individual `audio-rec-1`. This documentary evidence that the phone call event `call-1` actually happened can be modeled using the documentation pattern of the Event-Model-F as shown in Fig-

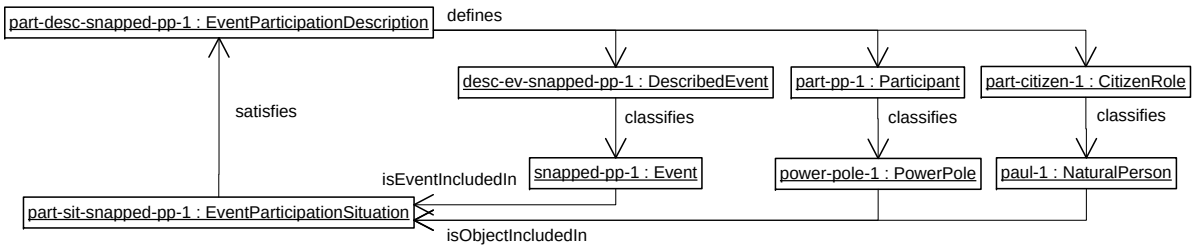


Figure 2: Citizen Paul participating in the event of a snapped power pole

ure 3. It represents the documentation of the event `call-1` by a recording of the call `audio-rec-1`. The individual `audio-rec-1` is of type `AudioData`, which is a concept taken from the COMM core ontology. The `AudioData` represents the information that is realized, i.e., contained in the audio recording. However, it is not a representation of the audio artifact such as a digital media stream that actually has been captured during the call.

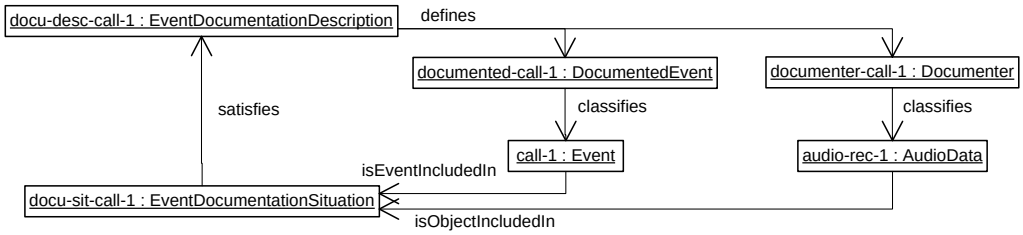


Figure 3: Documentation of Paul's call to the emergency hotline with an audio recording

In addition to the audio recording, also a description of the `snapped-pp-1` event is typed into the computer by the operators at the emergency hotline. This event description captures detailed information about the incident. For example, in the case of a flooded cellar the event description may contain detailed information for the emergency response entities like the fire department and provide specific instructions about, e.g., how to best reach the cellar, size, water level, and others. In our example, the event description is captured by the individual `text-description-1`, which is of concept `TextData`. It represents a textual description of where the snapped power pole is located and how it can be best reached. This textual description documents the `snapped-pp-1` event as depicted in Figure 4 using the Event-Model-F's documentation pattern. The `TextData` is a specialization of the concept `DigitalData` that is provided by COMM. It is a specialization of COMM towards representing textual data and is defined in the Ontology for Knowledge Acquisition (OAK) (Iria, 2009). The specialization of COMM by OAK is discussed in detail in Section 6.3.

For the course of this modeling example, it does not make a difference whether one considers `AudioData` or `TextData` to understand the role of media data for

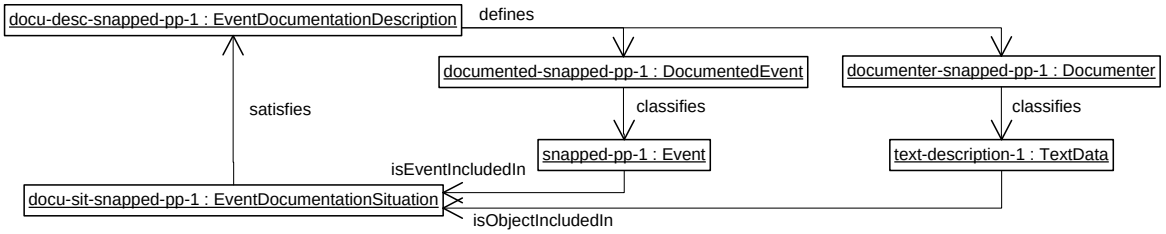


Figure 4: Documentation of the snapped power pole event with textual instructions for the emergency entities

documenting events in emergency response. In the following we concentrate on the audio recording, i.e., the `audio-rec-1` indicating that there was a recording taken during the call. Besides its mere documentation purpose, the recording of the phone call `audio-rec-1` between Paul and Rita can be replayed by the emergency response personnel in order to listen again to the conversation. In order to provide a more efficient access to the information communicated in the phone call, the audio recording `audio-rec-1` is processed by automatic classification algorithms such as a segmentation into smaller, distinct parts in which either Paul or Rita are speaking. Each part is automatically annotated with the speaker's name using automatic classification methods. Both the segmentation and the annotation with the speaker's name are modeled using the core ontology COMM. In order to conduct such a sophisticated annotation of an audio recording, we first need to provide a basic representation of the audio recording and the digital artifact created for it as shown in Figure 5.

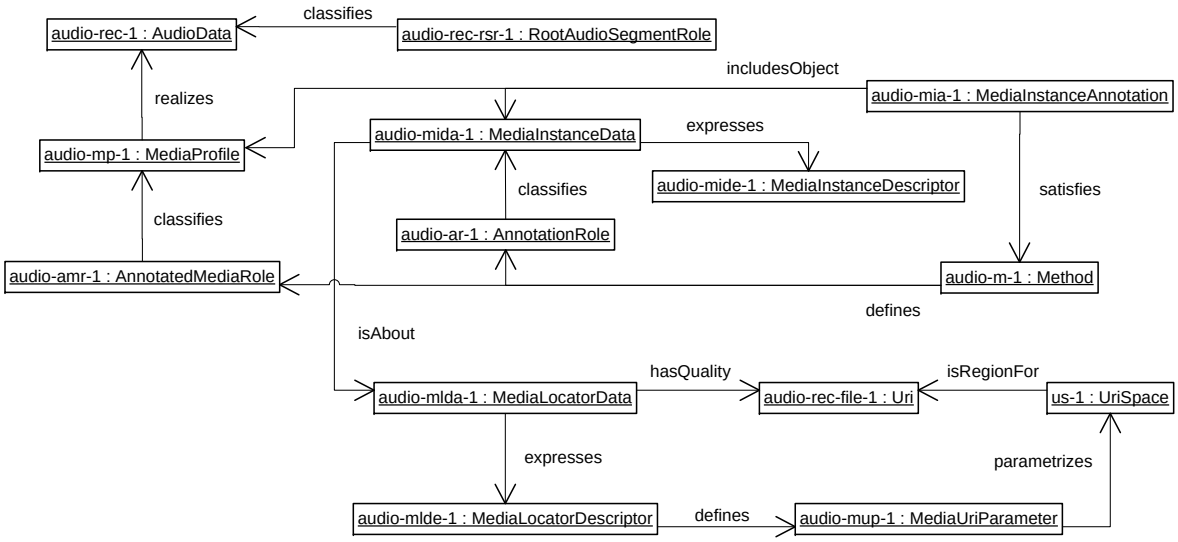


Figure 5: Representation of an audio recording and the file that realizes the recording

An underlying paradigm of COMM is the distinction of the information represented by any kind of multimedia asset and its digital (or even non-digital) realization. The recording is modeled in COMM as AudioData, a sub concept of InformationObject and represents the information that expresses some state of affairs (in DUL modeled as a Description). This information is clearly separated from the actual realization of that recording. The same information might be stored as .wav or .mp3 file, or even be stored on tape. This separation of the information object and its digital realization is based on the generic ontology of Information Objects (IO) (Borgo and Masolo, 2009). The modeled information, i.e., the fact that the call has been recorded and its digital realization is brought together by the individual audio-mp-1 of the concept MediaProfile. The concept of a media profile originates from the multimedia annotation standard MPEG-7 (MPEG-7, 2001). It allows for modeling various metadata such as color histograms, file type, and file location. This metadata can be complex including nesting of metadata. In COMM, we represent this nested metadata structure with StructuredDataDescriptions (see digital data pattern in Section 5.2.1).

In the example of Figure 5, we see two of such StructuredDataDescriptions, namely audio-mide-1 (a MediaInstanceDescriptor) and audio-mlde-1 (a MediaLocator-Descriptor). As shown in the figure, the individual audio-mida-1 of the type MediaInstanceData expresses the MediaInstanceDescriptor and is about the individual audio-mlda-1 of type MediaLocatorData. The individual audio-mlda-1 expresses a MediaLocatorDescriptor, represented by the individual audio-mlde-1. This about relation between audio- mida-1 and audio-mlda-1 models the nesting of the MediaInstanceDescriptor, which might, among others, contain a MediaLocatorDescriptor as subelement.

The location of the digital realization in form of some audio file is represented as an Uri named audio-rec-file-1. This Uri serves as quality of the MediaLocatorData represented by audio-mlda-1 as discussed above. A quality is always located in some region that represents the space of all possible values. Since the location is identified by some Uri, the corresponding region is the space of all Uris, represented by the individual us-1 of type UriSpace. Since the audio-mlda-1 expresses the MediaLocatorDescriptor, we link the quality to the description using the MediaUriParameter, which parametrizes the UriSpace. The audio-rec-1 itself is further classified as RootAudioSegmentRole, which indicates that the information object audio-rec-1 refers to the whole recording, in contrast to other AudioData representing parts of the whole recording. In COMM, audio data is always of type AudioData regardless of whether it refers to a whole or a part. The latter is represented using the decomposition pattern.

As the recording contains the voices of both Paul and Rita participating in the call-1, an automatic speaker change detection is employed to automatically segment the recording into several parts that refer to the different speakers. To model the two voices, we apply the COMM decomposition pattern to audio-rec-1 in order to represent the different segments of the recording. As an example, Figure 6 depicts the decomposition of audio-rec-1 into three seg-

ments identified as audio-segment-1 to audio-segment-3. The whole recording audio-rec-1 plays the InputSegmentRole, while the parts play an AudioSegmentRole in this pattern. Please note that in a real phone conversation at an emergency hotline there will typically be more than three speaker changes, i.e., more than three segments detected. However, for the purpose of demonstration it is already sufficient to consider only three segments. Please note further that in the COMM decomposition pattern, we do not identify the concrete person speaking in a segment. We only represent the number of different segments in the call. The annotation of the individual segments with the concrete person speaking is done only in the subsequent step.

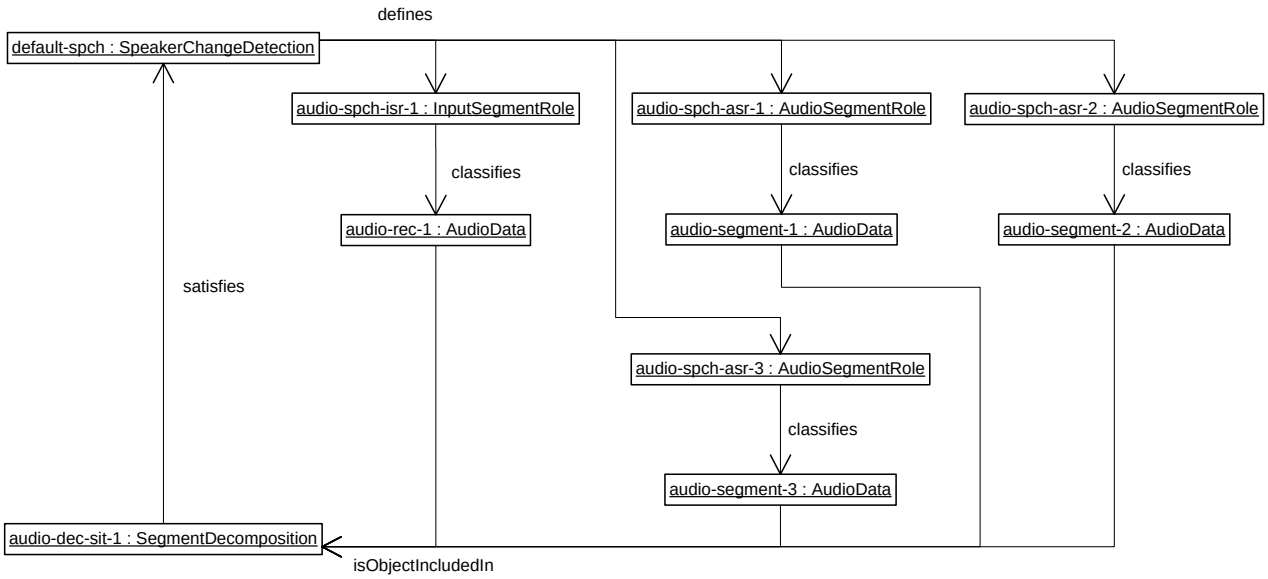


Figure 6: Decomposition of the call recording into segments

To annotate the individual segments of the audio recording with the person speaking, an algorithm for speech detection is applied. It leverages the known characteristics of the officer's voice Rita to distinguish her from Paul's voice. At the example of audio-segment-2, Figure 7 shows the semantic annotation of the second audio segment with the speaker's voice, paul-1. Following the DnS pattern, the SemanticAnnotation satisfies a Method, which represents the semi-automatic method audio-speaker-method-1 for assigning a person to a segment. The audio-speaker-method-1 defines an AnnotatedDataRole which classifies the AudioData to be annotated, in our example audio-segment-2. It further defines a SemanticLabelRole, which classifies the semantic annotation label for the audio segment. In this case, the individual paul-1 representing the citizen Paul.

Similar, also the segments of the audio recording are annotated with rita-1 speaking. Figure 8 shows at the example of audio-segment-1 the annotation of

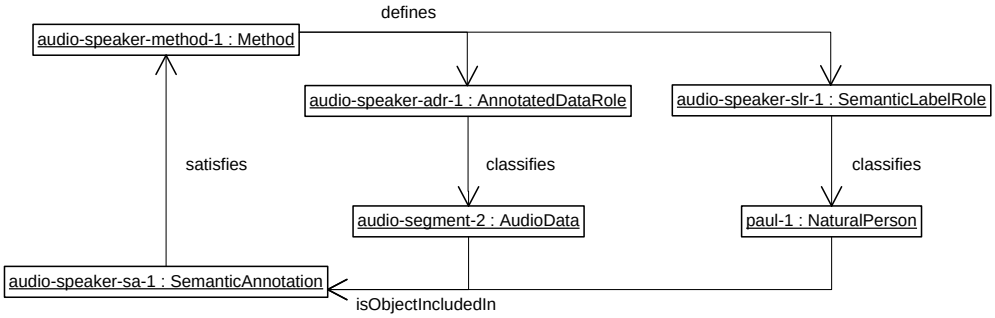


Figure 7: Annotation of a call segment with the speaker Paul

a segment with `rita-1`. The system at the emergency hotline has its own repository for the operators at the hotline and uses its own concept `Person` instead of DUL's concept `NaturalPerson`. This example shows that our core ontologies can reuse existing domain ontologies. This issue is discussed later in Section 6.4. Please note that in the case where multiple domain ontologies are reused, an alignment between these ontologies is necessary.

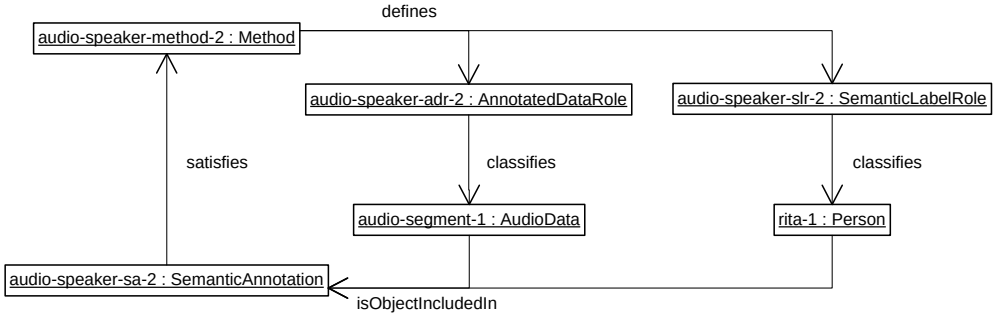


Figure 8: Annotation of a call segment with the speaker Rita

In his phone call to the emergency hotline, Paul tells Rita that he thinks that the event of a snapped power pole represented by the individual `snapped-pp-1` has caused the event of a `power-outage-1` that shortly after occurred. This subjective but plausible causal relationship between events can be modeled using the causality pattern of the Event-Model-F as indicated in Figure 9. This is represented by two roles, one classifying the causing event `cause-snapped-pp-1` and the other classifying the effect event `effect-power-outage-1`. The causality pattern also foresees to attach a justification to the causal relationship, here the `laws-of-physics-1`.

In order to represent that the causal relationship modeled above is Paul's interpretation of how the `power-outage-1` happened, the Event-Model-F allows for representing different contextual views on events by using the interpretation pattern as shown in Figure 10. We interpret the `power-outage-1` event by assem-

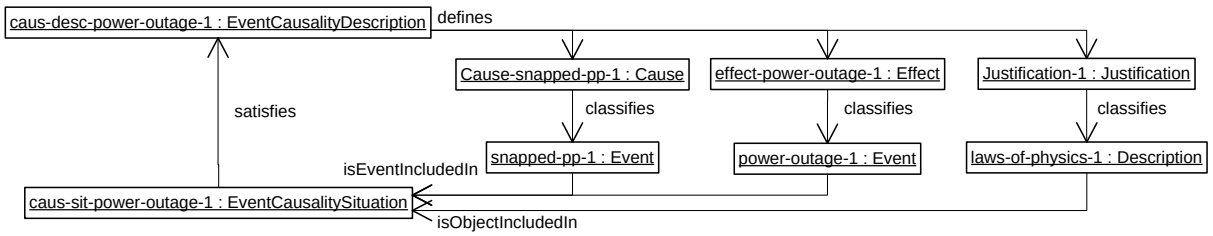


Figure 9: Causal relationship between a snapped power pole and a power outage

bling the different instantiations of the Event-Model-F patterns. These instantiations of the patterns are identified by the individuals `part-sit-snapped-pp-1` and `caus-sit-poweroutage-1` from Figure 2 and Figure 9.

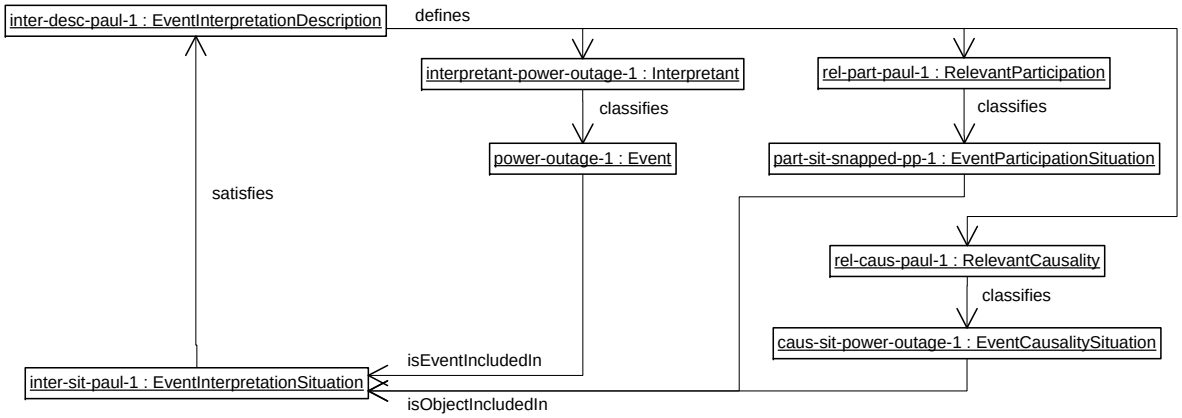


Figure 10: Representing Paul's interpretation that a snapped power pole has caused the power outage

Subsequently to the call between Paul and Rita, the description of the `call-1` event created by Rita in the emergency hotline system and the annotation of the event with the `audio-rec-1` is transferred to the emergency control center. The emergency control center also receives Paul's interpretation of the cause for the `power-outage-1`, namely the event of a snapped power pole he observed. At the emergency control center, Henry represented as `henry-1` works as emergency coordination manager. Among others, he supervises and filters incoming event descriptions. He reads the event description just created by Rita and listens to her conversation with Paul. As the snapped power pole might indeed have caused a large power outage in the city, Henry decides to send an urgent email message represented as `message-1` with a task description `task-1` to his forward liaison officer Marie. Henry attaches the event interpretation `inter-sit-paul-1` of Paul to the message as well as the audio file of the recorded call `audio-rec-1`. The forward liaison officer Marie represented by `marie-1` receives the message and drives to the location where the power

pole snapped. Marie checks the power cables of the snapped pole and sees that it only serves a few houses in the neighborhood with electricity. Thus, she concludes that the `snapped-pp-1` event cannot have caused the large `power-outage-1` of the city that happened and that Paul's interpretation is likely wrong. Marie takes out her camera to take a picture of the snapped power pole to document her observation. She attaches the picture together with some manually added tags as documentary support for the `snapped-pp-1` event. The results of Marie's investigation are send back to Henry, who decides that Marie's interpretation of the `snapped-pp-1` is the right one and that the power outage must have been caused by some other reason.

With our core ontology X-COSIMO, we represent communications, processes, and associated tasks. The message exchanged between `henry-1` and his forward liaison officer `marie-1` is modeled using the communication pattern and is shown in Figure 11. It provides a contextual view on communication where `henry-1` and `marie-1` are playing the roles `sender-1` and `recipient-1` of the email message, respectively. Attached to the message is Paul's interpretation of the power outage `inter-sit-paul-1` and the `audio-recording-1` is also represented.

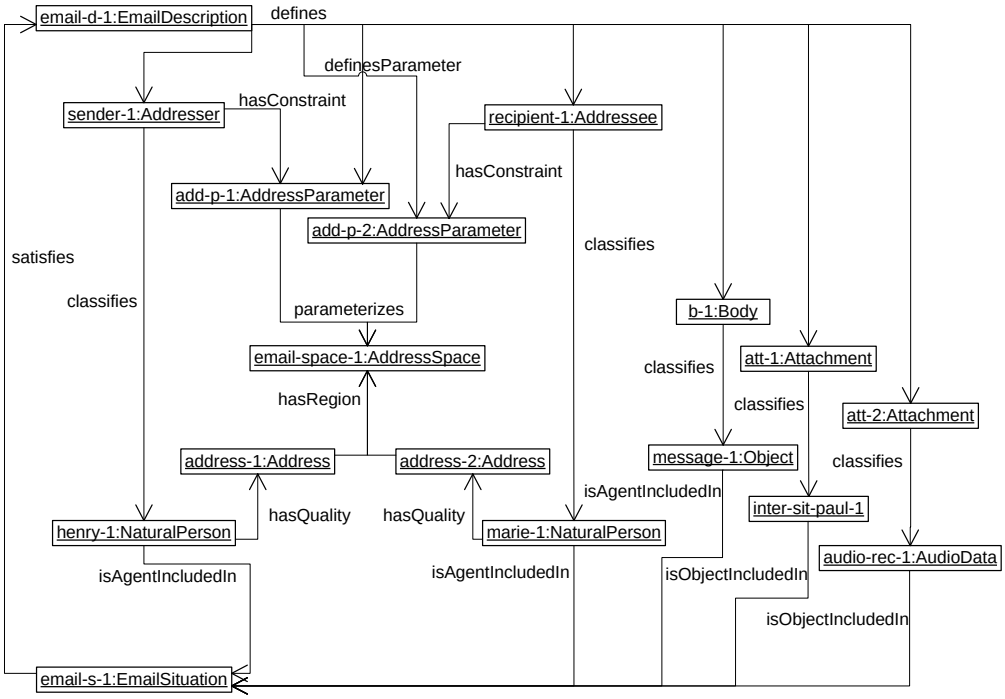


Figure 11: Henry sends Marie a message with a task description

The `task-1` assigned to Marie is of concept `InspectionTask` and is about to inspect the reported incident of a snapped power pole. The concept `InspectionTask` is defined in a domain ontology for tasks in emergency response, which is reused here. The assignment of the `InspectionTask` to Marie is modeled

in X-COSIMO as shown in Figure 12. The action `inspect-pp-1` is associated to `task-1` and assigned to `marie-1`, who is considered as the task owner while Henry is considered in the role of an information provider. Several inputs to the task are available and accordingly represented as well: Paul's interpretation of the power outage `inter-sit-paul-1`, the recorded phone call `audio-recording-1`, and the `message-1` that is described above.

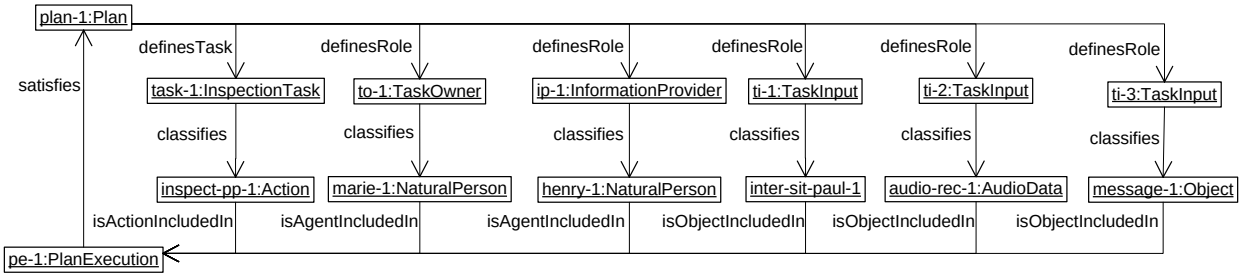


Figure 12: Description of the task sent to Marie

Finally, from the viewpoint of Henry being a member of the emergency control center, all events that happen are parts of the major `flooding-1` event. For example, the parts of the `flooding-1` incident are the events of a snapped power pole `snapped-pp-1`, a power outage `power-outage-1` that occurred, phone calls the emergency hotline receives such as `call-1`, and many others. This composition of the large `flooding-1` event by smaller events is depicted in Figure 13. It uses the mereology pattern of the Event-Model-F. The composite event is `flooding-1` with its three example component events `snapped-pp-1`, `power-outage-1`, and `call-1`. The temporal constraints by which the composite event is composed into the three and further subevents is not shown in Figure 13. It is important to note that the `flooding-1` is of concept `Flooding`, which is not defined within the mereology pattern. Rather, a domain ontology for emergency response developed by a partner in the WeKnowIt project is being reused here within the Event-Model-F.

By the concrete example of a flooding event, we have shown how the complex, structured knowledge required in our scenario can be modeled. Multiple core ontologies are involved and combined to represent the knowledge shared in the scenario, namely the Event-Model-F, X-COSIMO, and COMM. They are designed such that they allow for an optimal interplay and integration. By this, we allow for an effective exchange of emergency response information between the different emergency response entities in our scenario and the systems these entities use. For the design of our core ontologies, we follow a specific ontology design approach, which we present in the following section.

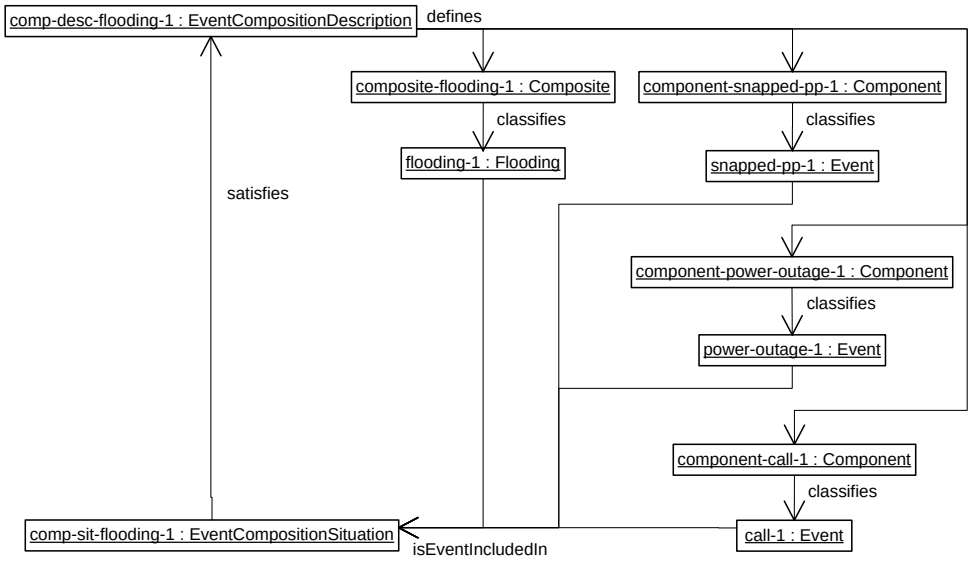


Figure 13: Modeling the large flooding event by composition of smaller events

4 Designing Core Ontologies

In order to provide a proper design approach for core ontologies, it is necessary to clarify what distinguishes a core ontology from foundational ontologies and domain ontologies. Thus, in Section 4.1, we briefly discriminate the concept of core ontologies from foundational ontologies and domain ontologies. In Section 4.2, we describe our design approach for building core ontologies. Essential choice for this design approach are the use of a foundational ontology as modeling basis and the use of ontology design patterns. The approach has been successfully applied to develop our three core ontologies Event-Model-F, COMM, and X-COSIMO.

4.1 Definition of Foundational Ontologies, Core Ontologies, and Domain Ontologies

An ontology is a special kind of information object that allows for formally representing the relevant concepts and relations of a considered domain in a machine readable format (Oberle et al., 2009b; Oberle, 2006). Thus, ontologies are a means to explicitly specify conceptual models with logic-based semantics (Oberle, 2006). In addition, ontologies are often referred to have a collaborative aspect (Oberle et al., 2009b; Pinto et al., 2009), i.e., the formal conceptualization should be expressed by a shared view and consensus between several parties. To be a *shared* conceptualization is of high importance for ontologies intended to support large-scale interoperability (Oberle et al., 2009b) such as the core ontologies Event-Model-F, COMM, and X-COSIMO used in the emergency scenario in Section 2. In the literature, we find different types and classifications of ontologies such as (Oberle, 2006; Gangemi et al., 2002). In the context

of this work, we follow the three-layered architecture of ontology libraries (Gangemi et al., 2004) and discriminate between foundational ontologies, core ontologies, and domain ontologies (Oberle, 2006). Core ontologies may use foundational ontologies as well as leverage domain ontologies. Thus, in the following sections, we briefly analyze and define the nature of foundational ontologies, core ontologies, and domain ontologies and discuss their relation to each other.

4.1.1 Foundational Ontologies

Foundational ontologies are generic across many fields (Oberle, 2006). They have a large scope and are highly reusable in different modeling scenarios (Borgo and Masolo, 2009). Thus, foundational ontologies serve reference purposes (Oberle, 2006) and aim at modeling the very basic and general concepts and relations (Borgo and Masolo, 2009; Oberle, 2006) that make up our world, e.g., objects, events, participation, and parthood. Foundational ontologies are heavyweight as they are rich in axiomatization (Borgo and Masolo, 2009), precisely defining the concepts in the ontology and their relations (Oberle, 2006). Synonyms of the term foundational ontology are generic ontology, upper level ontology, and top-level ontology (Euzenat and Shvaiko, 2007; Oberle, 2006).

Examples of foundational ontologies are the ABC ontology and model (Lagoze and Hunter, 2001), the Basic Formal Ontology (BFO)[6] (Masolo et al., 2003), DOLCE (Borgo and Masolo, 2009; Gangemi et al., 2002; Masolo et al., 2003), the Object-Centered High-level REference ontology (OCHRE) (Schneider, 2003), the General Formal Ontology (GFO) (Herre et al., 2006), the OpenCyc ontology[7] (Lenat et al., 1990), and the Suggested Upper Merged Ontology (SUMO) (Niles and Pease, 2001). A detailed discussion of most of these foundational ontologies can be found in (Oberle, 2006).

Heavyweight foundational ontologies can have lightweight ones (Oberle, 2006), e.g., DOLCE and its lightweight version the DOLCE+DnS Ultralight (DUL) ontology[8]. The main purpose of heavyweight foundational ontologies is to serve as reference ontologies during development time (Oberle, 2006). A lightweight version of the foundational ontology is applied to facilitate reasoning at run time (Oberle, 2006). Foundational ontologies can be used as starting point for building core ontologies and domain ontologies (Oberle, 2006).

4.1.2 Core Ontologies

In contrast to foundational ontologies that span across many fields and model the very basic and general concepts and relations (Borgo and Masolo, 2009; Oberle, 2006) that make up our world, core ontologies provide a detailed abstract definition of structured knowledge in one of these fields, e.g., medicine, law, software services, personal information management, multimedia annotations, and others. By their nature, foundational

[6] http://www.ifomis.org/bfo

[7] http://www.cyc.com/cyc/opencyc/

[8] Available from: http://ontologydesignpatterns.org/wiki/Ontology:DOLCE+DnS_Ultralite

ontologies are much broader than core ontologies. Core ontologies can be based on foundational ontologies and provide a refinement to foundational ontologies by adding detailed concepts and relations in their specific field. However, core ontologies are still very generic and span across a set of domains in a specific field (Oberle, 2006).

Examples of core ontologies are the MPEG7 ontology[9] for multimedia annotations (Hunter, 2005), a core ontology for software components and web services (Oberle et al., 2009a; Oberle, 2006; Mika et al., 2004), our core ontology Event-Model-F for capturing human experiences in terms of events and objects (Scherp et al., 2009a), the X-COSIMO ontology for personal information management (Franz et al., 2007), and the Core Ontology for Multimedia (COMM) for modeling multimedia annotations (Arndt et al., 2009; Staab et al., 2008; Arndt et al., 2007).

Core ontologies are situated in between the two extremes of foundational ontologies and domain ontologies (Oberle, 2006), described next. As foundational ontologies serve as a good modeling basis for core ontologies, so do core ontologies for domain ontologies.

4.1.3 Domain Ontologies

Finally, with domain ontologies we find representation of knowledge that is specific for a particular domain (Euzenat and Shvaiko, 2007; Oberle, 2006). Domain ontologies use terms in a sense that is relevant only to the considered domain and which is not related to similar concepts in other domains (Euzenat and Shvaiko, 2007). Domain ontologies can be very complex, i.e., they can comprise a very large number of concepts and relations. They can make use of and can be based on foundational ontologies or core ontologies by specializing their concepts in the domain ontology (Oberle, 2006). Domain-specific ontologies can be used as external sources of background knowledge (Euzenat and Shvaiko, 2007), e.g., in combination with core ontologies.

Examples of domain ontologies are a soccer ontology developed in the SmartWeb project (Oberle et al., 2007), the Foundational Model of Anatomy[10] as a domain-specific medical ontology describing the anatomy of the human body (Rosse and Mejino, 2003), the RadLex Lexicon for Radiology[11] (Kundu et al., 2009), and other medical ontologies such as Snomed (Cote et al., 1993), the Gene Ontology (Ashburner, 2000), and Galen (Rector and Horrocks, 1997).

4.1.4 Summary

The relation between foundational ontologies, core ontologies, and domain ontologies is illustrated by the ontology stack in Figure 4.1.4. The bottom of the figure shows the foundational ontologies. They may be used and refined by core ontologies in the middle layer. The further, core ontologies as well as foundational ontologies can be used for defining semantically precise domain ontologies. The borderline from core ontologies to domain ontologies is not clearly defined. Core ontologies intend to be generic within a field that spans across multiple domains (Oberle, 2006). Similarly,

[9] http://metadata.net/mpeg7/
[10] http://sig.biostr.washington.edu/projects/fm/index.html
[11] http://www.radlex.org/

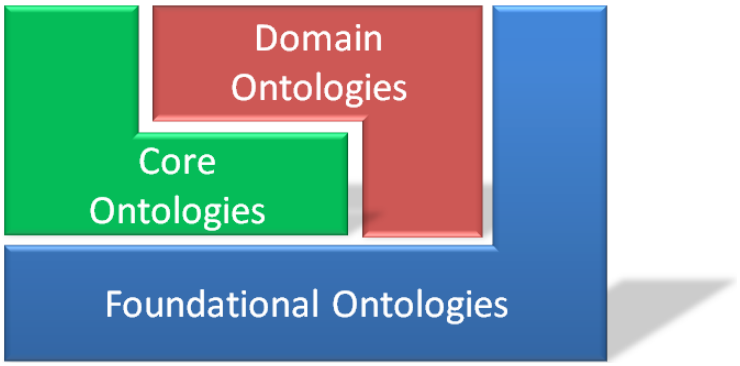


Figure 14: Ontology stack of foundational ontologies, core ontologies, and domain ontologies

also the distinction between foundational ontologies and core ontologies is not clearly defined (Oberle, 2006). However, the distinction is meaningful and useful for building ontology libraries as foundational ontologies, core ontologies, and domain ontologies serve different purposes (Gangemi et al., 2004).

4.2 Design Approach for Core Ontologies

We have developed a guideline describing the design approach of core ontologies. This design approach for core ontologies is described in the following sections along the non-functional properties identified, namely axiomatization and formal precision, modularity, extensibility, reuseability, and separation of concerns.

4.2.1 Axiomatization and Formal Precision

When designing an ontology, it is desirable to use a solid and sound modeling basis (Oberle, 2006). Thus, our approach foresees the use of a foundational ontology for designing the core ontology. We align the concepts and relations defined in the core ontology to the basic categories of human cognition investigated in philosophy, linguistics, and psychology (Oberle, 2006). These basic categories are manifested in the foundational ontology.

The alignment of the foundational ontology with the core ontology also includes the adoption and specialization of the formal semantics of the foundational ontology in the core ontology. While the axiomatization of a foundational ontology enables validating the upper-level semantics of the knowledge expressed with it, the alignment of the core ontology with the foundational ontology provides support for validating the more specific semantics of the concepts and relations defined in the core ontology.

A well-designed foundational ontology is very diligent with respect to the ontological choices to which it commits (Oberle, 2006), e.g., the selection of the most abstract concepts that are modeled. Thus, when reusing such a foundational ontology for designing a core ontology the engineer is also requested to carefully think about his or

her design choices (Oberle, 2006). Developing foundational ontologies is extremely hard but it needs to be conducted only once (Oberle, 2006). An ontology engineer should strive for applying a foundational ontology that has proven its usefulness when designing a new core ontology or domain ontology (Oberle, 2006).

For designing our core ontologies, we have based them on the foundational ontology DOLCE and have carefully aligned our core ontologies with DOLCE's lightweight version DOLCE+DnS Ultralight (DUL). DOLCE aims at capturing the ontological categories underlying natural language and human common sense. It has a minimal core that includes only the most general concepts and patterns and is well suited for modularization. Additional theories such as Descriptions and Situations, Ontology on Information Objects, or the Ontology of Plans can be integrated when necessary (Oberle, 2006). We chose DOLCE as it already has proven to serve as good modeling basis for core ontologies. DOLCE has been successfully applied to design ontologies in different domains such as law, biomedicine, agriculture, and software components and web services Oberle (2006).

4.2.2 Modularity

Even well designed core ontologies are usually large and cover more knowledge than might be required in a specific application domain (Gangemi and Presutti, 2009). Thus, concrete systems will commonly use only portions of it. In this case, it is hard to reuse only the "useful" pieces of such a monolithic core ontology (Gangemi and Presutti, 2009).

Core ontologies (and domain ontologies) have a better design when applying ontology design patterns captured by the foundational ontology it uses (Oberle, 2006). By using ontology design patterns, it allows for selecting the parts of the ontology in a concrete application that are actually needed and used (Gangemi and Presutti, 2009). Thus, our design approach for core ontologies builds upon a foundational ontology that supports a pattern-oriented design approach (Oberle et al., 2006). DUL provides such a pattern-oriented approach.

Ontology design patterns shall not be too specific or too close to a particular domain. This would disallow the application of the pattern in other domains of the field covered by the core ontology. On the other hand, the patterns shall also not be too generic as reuse in a concrete domain would be hampered. A too generic pattern is hard to apply in a specific domain. The scope of a core ontology itself is defined through the scope of its patterns.

4.2.3 Extensibility

Foundational ontologies provide a high-level, abstract vocabulary of concepts and relations that are likely to be used in current and future application domains. In order to provide a solid basis for future extensions of core ontologies, a precise alignment of the concepts defined in a core ontology with the high-level concepts of a foundational ontology is conducted using our design approach. By this precise alignment, new patterns can be added to the core ontology without affecting the existing patterns. In addition to adding new patterns, the existing patterns of a core ontology can be extended. This

is typically conducted by specializing the existing concepts and properties defined in the patterns. Finally, besides the already connected core ontologies within a specific application also further core ontologies can be developed and integrated if necessary.

4.2.4 Reuseability

For modeling complex, structured knowledge, reuse can happen on different levels, e.g., on the level of ontology design patterns, core ontologies, and domain-specific ontologies. Different patterns in the core ontology provide different descriptions of concepts defined in it. By splitting up core ontologies into different parts they allow for reusing the structured knowledge defined within the core ontology's design patterns among different applications. This refers to the issue of extensibility discussed in Section 4.2.3.

In addition, the core ontologies can be combined with domain-specific knowledge. In the ideal case, domain ontologies reuse the ontology design patterns defined in core ontologies by specializations of the ontology design patterns (Gangemi and Presutti, 2009). However, our approach does not require that domain ontologies are based on a foundational ontology or a core ontology. In fact, we explicitly consider both options as it cannot be assumed that all domain ontologies are aligned with a foundational ontology or core ontology. This is achieved by using the Descriptions and Situations ontology design pattern of DOLCE. Here, the roles defined within a contextual situation can refer to a domain ontology that is either carefully aligned with DOLCE, aligned to the core ontology, or that is completely independent.

4.2.5 Separation of Concerns

Our design approach supports the separation of concerns by defining the structured knowledge in the core ontology and leaving all domain-specific aspects out of it. This is achieved again with the Descriptions and Situations ontology design pattern. The structured knowledge of the concrete field of the core ontology is captured by its ontology design patterns, e.g., participation, causality, and documentation for the Event-Model-F, the annotation and decomposition in COMM, and the communication pattern in X-COSIMO. This structured knowledge is specified by the roles defined within these patterns, i.e., the `defines` relations of the `Descriptions`. The domain knowledge is only referred to by the roles classifying the events and objects used. By this, the core ontologies are independent of any concrete domain that makes use of the concepts defined by them. In addition, core ontologies provide support to include individuals defined in some domain ontologies.

5 Examples of Core Ontologies

In this section, we describe the design of our core ontologies Event-Model-F, COMM, and X-COSIMO with focus on the parts relevant to model the emergency response scenario in Section 3. Our core ontologies are based on the foundational ontology

DOLCE+DnS Ultralight (DUL). DUL defines the class DUL:Event next to the disjoint upper classes DUL:Object, DUL:Abstract, and DUL:Quality. The definition of Event has been specialized from the formal definition in DOLCE as an entity that exists in time. The class Object stands for entities that exist in space such as living things as well as non-living and social and cognitive entities. A Quality[12] is a characteristic of an object or an event. It has a value that is represented as a point or area in some Abstract. The class Abstract represents value spaces, e.g., the space of natural numbers or the time of a day. Typically, we do not prescribe specific Abstracts that are to be used. We rather refer to the generic Abstracts already defined in DUL such as the regions DUL:TimeInterval, DUL:SpatioTemporalRegion, and DUL:SpaceRegion.

For modeling our core ontologies, we make use of the ontology design patterns Descriptions and Situations (DnS) and the ontology of Information Object (IO) (Borgo and Masolo, 2009). The DnS pattern provides an ontological formalization of context (Oberle, 2006; Gangemi and Mika, 2003). With DnS one can reify Events and Objects and describe the n-ary relation that exists between multiple individuals of them. Thus, it allows for a formally precise representation of different, contextualized views on events. The IO ontology pattern describes the relation between an information object such as a poem, song, and a story and their actual physical realization in form of a printed book, recorded track, and a movie taken (Oberle, 2006). We describe the design of our three core ontologies the Event-Model-F, COMM, and X-COSIMO in the following sections.

5.1 Event-Model-F—Core Ontology of Events for Representing Human Experience

The core ontology Event-Model-F for representing human experience allows for modeling the different relationships between events and objects. The requirements to the core ontology for events have been derived from existing event models in various domains such as music, journalism, multimedia, news, cultural heritage, and knowledge representation (Wang et al., 2007; Raimond and Abdallah, 2007; IPTC, 2008; Doerr et al., 2007; Mueller, 2008; Francois et al., 2005; Jain, 2008; Ekin et al., 2004). Identified requirements are representing (1) participation of living and non-living objects in events, (2) temporal duration of events, and (3) spatial extension of objects. In addition, three kind of event relationships shall be supported, namely (4a) mereological (composition of events), (4b) causal, and (4c) correlation. The common model shall also support the experiential aspect, i.e., the (5) annotation of events with sensor data such as media data, and allow for (6) different interpretations of events. Existing models almost fully support participation, time and space, and the experiential aspect. However, they substantially lack in the mereological, causal, and correlation relationships, and event interpretations. Here, we find different limitations or even no support, e.g., only simple mereological relationships in (Raimond and Abdallah, 2007) and causal relationships in (Doerr et al., 2007). Correlation is not considered at all and event interpretations are only mentioned in (Jain, 2008) but remain future work.

[12] Also called trope, see http://plato.stanford.edu/entries/tropes/

With respect to the requirements, we introduced specialized instantiations of the DnS pattern. Here, the participation of objects in events (1) is implemented by the participation pattern. It also provides for modeling the absolute time and location of events (2) and objects (3). The mereology pattern, causality pattern, and correlation pattern implement the structural relationships between events (4a-4c). In addition, the mereology pattern allows for modeling the relative temporal relations and relative spatial relations between events (2) and objects (3). In order to express such relative temporal relations between events, one can facilitate the provided means of DOLCE such as the formalization of Allen's Time Calculus[13]. The documentation pattern provides for annotating events (5). It can be seamlessly linked with other ontologies, e.g., the Core Ontology for Multimedia (Arndt et al., 2007) for precisely describing digital media data like images and videos. Finally, the interpretation pattern supports different event interpretations (6).

We use the DnS pattern for representing occurrences in the real world, i.e., the events and objects we are modeling. These occurrences are subject to discussion and interpretation and may not be objectively observable. The DnS pattern allows for representing different opinions about events and their participating objects. This feature is not provided by DOLCE's participation relation. In the following, we present the ontology patterns of the Event-Model-F that have been employed to model the scenario in Section 3 and illustrate them in diagrams. This comprises almost all patterns of the Event-Model-F, expect from the correlation pattern. A complete description of the Event-Model-F can be found in (Scherp et al., 2009a).

5.1.1 Participation Pattern

The participation pattern of the Event-Model-F enables to formally express the participation of objects in events. As shown in Figure 15, participation is expressed by an `F:EventParticipationSituation` that `satisfies` an `F:EventParticipationDescription`. The situation includes the `Event` being described and the `Objects` participating in this event. The `EventParticipationDescription` classifies the described event and its participants by using the concepts `F:DescribedEvent` (specialized from `DUL:EventType`) and the object role `F:Participant` (specialized from `DUL:Role`). The concept `DescribedEvent classifies` the `Event` that is described by the participation pattern, e.g., the event of a flooded cellar. Likewise, instances of `Participant` classify objects as participants of the event. For example, the citizen calling the emergency hotline to report about the flooded cellar. Instances of `Participant` can be roles defined in some domain ontology as indicated in Figure 15. For example, an emergency response ontology that defines the role of a person being affected, i.e., the emergency subject such as a `CitizenRole`, and the role describing the rescue staff such as a `FiremanRole`. Besides the role an object can play in a specific participation pattern, also the described event and its participating objects themselves can be defined in some domain ontology as indicated in Figure 15.

[13] Available from: `http://wiki.loa-cnr.it/index.php/LoaWiki:Ontologies`

The parameter F:LocationParameter describes the general spatial region where the objects are located. It parametrizes a DUL:SpaceRegion and defines a property DUL:isParameterFor to the Participant role. The Object that is classified by the Participant has a Quality with the property DUL:hasRegion of a DUL:SpaceRegion. Thus, using the F:LocationParameter we can define the location(s) represented by DUL:SpaceRegions that are relevant for describing the event in a given context. For example, when quenching a house fire all firemen have their specific location within and around the building. The F:LocationParameter can then be used to describe in general that the firemen where at that specific house, e.g., in form of some longitude-latitude rectangular. Thus, we do not need to explicitly state where the individual firemen are. The F:TimeParameter describes the general temporal region when the event happened. It parametrizes a DUL:TimeInterval and defines a property DUL:isParameterFor to the DescribedEvent role. For example, one can state that the house fire happened on June 13, 2006.

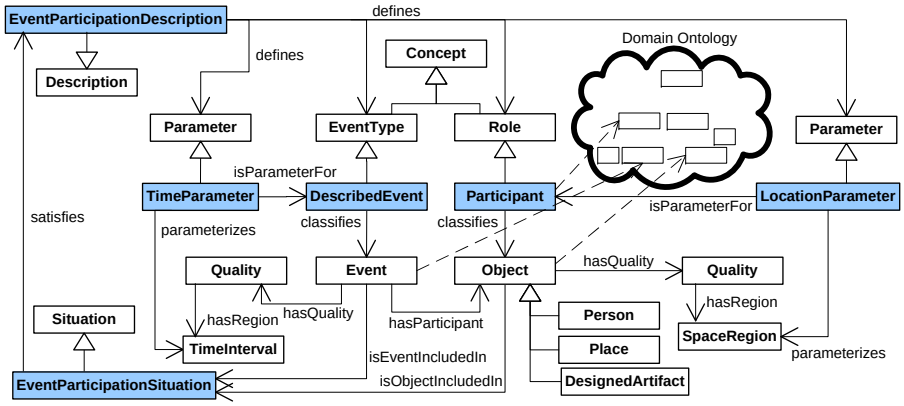


Figure 15: Participation pattern

5.1.2 Mereology Pattern

Events are commonly considered at different abstraction levels depending on the view and the knowledge of a spectator. For instance, the event of a flooded cellar may be considered as such or as part of the larger event of a flooding in which many other (smaller) incidents occur. The mereology pattern shown in Figure 16 enables expressing such mereological relations as composition of events. The composite event is the "whole" and the component events are its "parts". Formally, a F:EventCompositionSituation includes one instance of an event that is classified by the concept F:Composite and many events classified as its F:Component(s). Accordingly, an EventCompositionSituation satisfies a F:CompositionDescription that defines the concepts

`Composite` and `Component` for classifying the composite event and its component events.

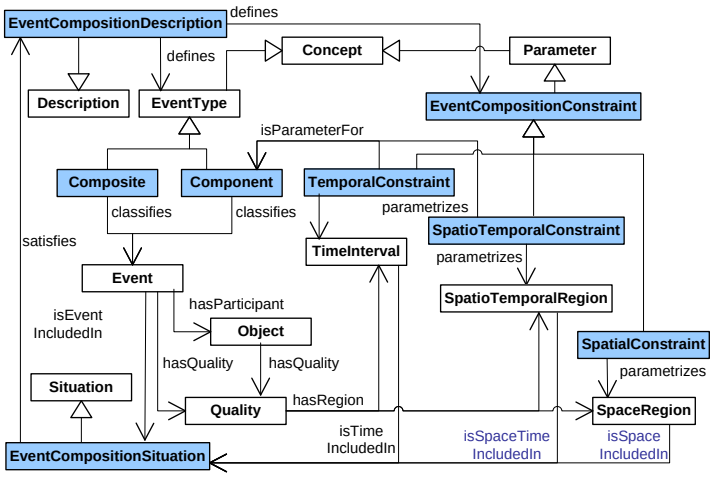


Figure 16: Mereology pattern

Events that play the `Component` role may be further qualified by temporal, spatial, and spatio-temporal constraints. As events are formally defined as entities that exist in time and not in space (Scherp et al., 2009a), constraints including spatial restrictions are expressed through the objects participating in the component event. For instance, a `Component` event may be required to occur within a certain time-interval, e.g., the second week of June 2009. Depending on its objects, a `Component` event may also happen in a certain spatial region. For example, the flooding of a town should be composed of events that have objects associated to it, which have some certain range of longitude and latitude. Finally, events and the objects bond to it may be qualified by a spatio-temporal quality like the progress of a flood that extents over time and space, starting with a high water level located in some area of a river and extending spatially over time into other areas. Any such constraints are formally expressed by one or multiple instances of the `F:EventCompositionConstraint`. Thus, with the composition pattern, events may be arbitrarily temporally related to each other, i.e., they might be disjoint, overlapping, or otherwise ordered. In order to express such relative temporal relations between events, one can facilitate the provided means of DOLCE such as the formalization of Allen's Time Calculus[14].

5.1.3 Causality Pattern

Causality is the philosophical problem investigating the existence of any special "tie" binding causes and effects together (Itkonen, 1983). It can be questioned either as "Why" or "How" (Itkonen, 1983). An answer to this question is (or better said claims to be) a causal explanation. What explains, is the cause and that what is explained is

[14]Available from: `http://wiki.loa-cnr.it/index.php/LoaWiki:Ontologies`

the effect (Itkonen, 1983). Events are the most natural concept to serve for defining causal relations (Quinton, 1979). In fact, causes and effects are two specific types of events (Itkonen, 1983; Lombard, 1986). A causal relationship is always justified by some (maybe implicit) underlying causal theory.

We designed a causality pattern as depicted in Figure 17. The pattern defines two `EventTypes` called `F:Cause` and `F:Effect` which `classify Events`. It further defines a `DUL:Description`, which is classified by a `F:Justification`. By this, the pattern explicitly expresses the causal relationship between the cause and the effect under the justification of some theory. A theory might be an opinion, a scientific law, or not further specified. For example, during a heavy storm, a power outage event might occur caused by a snapped power pole event. The `Justification` of this causal relationship is the laws of physics.

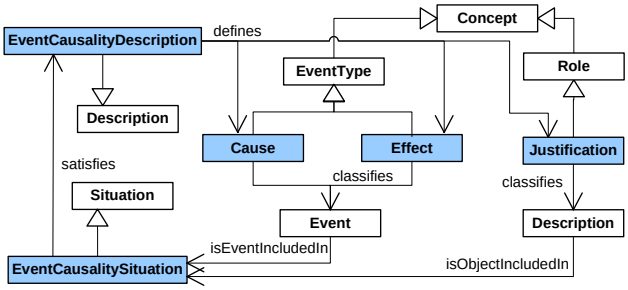


Figure 17: Causality pattern

5.1.4 Documentation Pattern

Documentary evidence for an event may be given by arbitrary objects, e.g., some sensor data or media data, or by other events. Formally, this relation is expressed in F by the documentation pattern depicted in Figure 18. It defines the concept `F:DocumentedEvent` that classifies the documented event and the concept `F:Documenter` that classifies the documentary evidence for that event. This evidence can be expressed by any specialization of an `Object`, e.g., a digital photo taken with a cell phone during an incident, or a specialization of `Event`. For example, digital media data like images and videos can be classified as `Documenter` and precisely described using the Core Ontology for Multimedia (Arndt et al., 2007) described in Section 5.2. Objects are documented via the events in which they participate (see participation pattern in Section 5.1.1).

5.1.5 Interpretation Pattern

The perception of events as occurrences in the real world heavily depends on the context and point of view of the observer. Such different, context-dependent event interpretations can be described formally by instantiating the different Event-Model-F

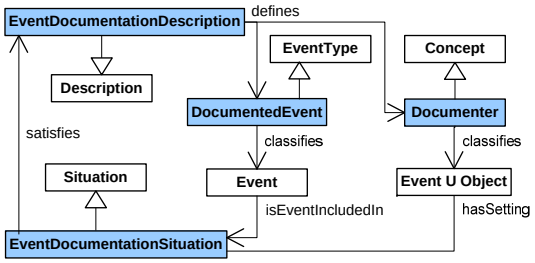


Figure 18: Documentation pattern

patterns presented so far and binding them together with the interpretation pattern depicted in Figure 19. Each pattern models a single, specific interpretation of an event by associating *participations*, *mereological*, *causal*, and *correlative* relationships, as well as *documentations* relevant in the context of a specific *interpretation*. In the emergency scenario, two emergency control officers might have differing interpretations of the power outage. One might be convinced that the power outage is due to a snapped power pole, while the other might think of a more serious case of a damaged power plant. Both consider the same event of a power outage, however, consider it from different points of view that involve other events and objects in different patterns.

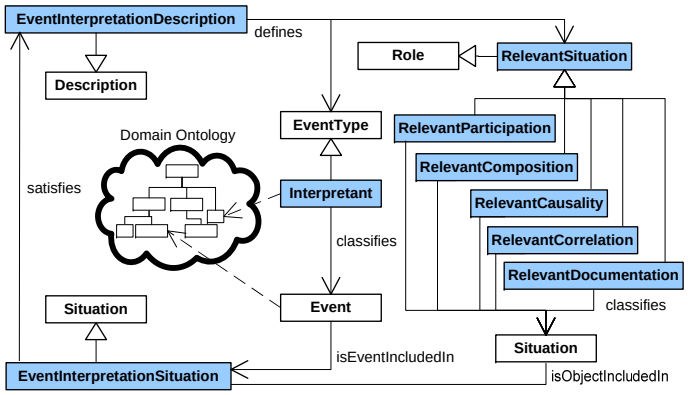


Figure 19: Interpretation pattern

Formally, the interpretation pattern shown in Figure 19 defines a `F:Interpretant` that is specialized from `EventType` and classifies the interpreted `Event`. The `Interpretant` might be defined in some domain ontology and determines how an event is interpreted, e.g., as emergency incident in the case of the emergency control center or as news event described in a news paper. Within each interpretation, we classify the `F:RelevantSituations`, namely the situations satisfying the participation, mereology, causality, correlation, and documentation. These are defined as specializations of `RelevantSituation`.

5.2 COMM—Core Ontology for Multimedia

The Core Ontology for Multimedia (COMM) (Arndt et al., 2007) models the domain of multimedia content and annotation, and is based on MPEG-7. In contrast to other approaches to modeling MPEG-7 as an ontology (Hunter, 2005; Bloehdorn et al., 2005; Garcia and Celma, 2005; Isaac and Troncy, 2004; Tsinaraki et al., 2004), COMM is not designed as a one-to-one mapping, but provides a set of patterns that cover the core and repetitive building blocks of MPEG-7. In the following, we discuss the patterns of COMM used for the emergency response scenario in Section 3. Three repetitive structures have been identified in MPEG-7:

Decomposition. MPEG-7 provides descriptors for spatial, temporal, spatio-temporal and media source decompositions of multimedia content into segments. A segment is the most general abstract concept in MPEG-7 and can refer to a region of an image, a piece of text, a temporal scene of a video, or even to a moving object tracked during a period of time.

Annotation. MPEG-7 defines a very large collection of descriptors that can be used to annotate a segment. These descriptors can be low-level visual features, audio features or more abstract concepts. They allow the annotation of the content of multimedia documents or the media asset itself.

Nested Data Structures. Descriptors in MPEG-7 are nested structures containing different kinds of data. Except for the semantic annotation, data refers to strings or numerical values such as the encoding of an image or the values of a color histogram.

These three structures are modeled as patterns of our ontology COMM. The patterns are the decomposition pattern, annotation pattern, and the digital data pattern. Before we discuss the patterns in detail, we will introduce some central concepts that are present in all the patterns:

Digital Data. Within the domain of multimedia annotation, the notion of digital data is central—both the multimedia content being annotated and the annotations themselves are expressed as digital data. We consider `DigitalData` entities of arbitrary size to be `InformationObject`s, which are used for communication between machines. The IO design pattern states that `Description`s are expressed by `InformationObject`s, which have to be about facts (represented by individuals of type `Entity`). These facts are settings for `Situation`s that have to satisfy the `Description`s that are expressed by `InformationObject`s. This chain of constraints allows the modeling of complex data structures to store digital information.

Multimedia Data. This encapsulates the MPEG-7 notion of multimedia content such as images, audio, text, open document format (ODF) documents, and is a subconcept of `DigitalData`. `MultimediaData` is an abstract concept that has to be further specialized for concrete multimedia content types (e.g. `AudioData` corresponds to the data representing the recorded audio signal). According to

the IO pattern, MultimediaData is realized by some physical Media, e.g., MediaProfile, which contains information about the storage location, file type, and others. This concept is needed for annotating the physical realization of multimedia content.

Media. Media objects represent the realizations of the information represented as MultimediaData. In the context of electronic devices this refers to files accessible via some protocol, but might in a more general setting also refer to physical realizations such as a painting.

Method. A Method refers to some manual process, and its subclass Algorithm to a (semi-)automatic process that processes some input and generates output. Examples are the extraction of features, e.g., a DominantColorExtractionAlgorithm, or the manual annotation of content with a semantic concept.

Input/Output Roles. Methods always define InputRoles and OutputRoles. The input role classifies the object that is processed, while the output role classifies the results. For example, a segmentation defines exactly one InputSegmentRole which classifies the image that is being segmented and at least one OutputSegmentRole which classifies the resulting subsegments.

In the following subsections, we present the three patterns of COMM. For expressing concrete descriptors of MPEG-7 a specialization of the concepts involved in each pattern is required, but for easier comprehension we discuss the pattern on a more abstract level. The details are comprehended by the ontology itself which is available through our website: http://west.uni-koblenz.de/Research/ontologies/.

5.2.1 Digital Data Pattern

The digital data pattern is defined as depicted in Figure 20. DigitalData expresses StructuredDataDescriptions, which define meaningful labels for the information contained by DigitalData. This information is represented by literals such as scalars, matrices, strings, rectangles, or polygons. In DOLCE terms, these values are represented as a Quality that is associated with the DigitalData and located in Regions. In the context of a Description, these Regions are parametrized by Parameters. StructuredDataDescriptions define StructuredDataParameters, for which the qualities located in the parametrized Regions assign values to the DigitalData. Referring to the example in Figure 5 (cf. Section 3), we see that the formalization of data structures so far is not sufficient. Complex MPEG-7 types can include nested types that again have to be represented by StructuredDataDescriptions. In our example, the MediaInstanceDescriptor contains the MediaLocatorDescriptor. The digital data pattern covers such cases by allowing a DigitalData instance to be about another DigitalData instance which expresses the nested StructuredDataDescription.

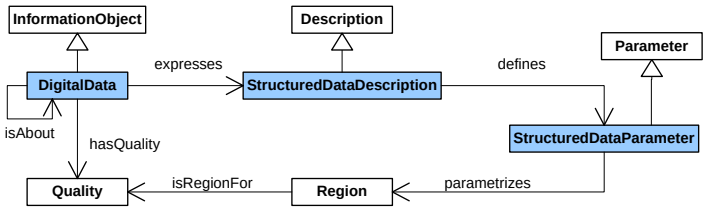


Figure 20: Digital data pattern

5.2.2 Decomposition Pattern

Following the DnS pattern, we consider the decomposition of `MultimediaData` to be a `Situation` that satisfies a `Method`. More specifically the `Situation` is a `SegmentDecomposition`. The description refers to an algorithm such as a `SegmentationAlgorithm` (cf. Figure 6) or to a manual segmentation `Method` like a tool that allows the user to draw a bounding box around a depicted face. Of particular interest with respect to the decomposition pattern are the roles that are defined by a `SegmentationAlgorithm` or a `Method`. The input to a segmentation is `MultimediaData` that is classified by an `InputSegmentRole`, while the `MultimediaData` referring to the output segments are classified by `OutputSegmentRoles`. These data entities have as setting the `SegmentDecomposition` situation. `OutputSegmentRoles` as well as `SegmentDecompositions` are then specialized for specific types of media (according to the segment and decomposition hierarchies of MPEG-7 (MPEG-7 (2001), part 5, section 11). The decomposition pattern is depicted in Figure 21. Please note that the concept `TextData` is actually not provided by COMM. It has been added to the decomposition pattern by the Ontology for Knowledge Acquisition (OAK) (Iria, 2009) in order to support the annotation and decomposition of textual data.

In terms of MPEG-7, unsegmented (complete) multimedia content also corresponds to a segment. Consequently, annotations of complete multimedia content start with a root segment. In order to designate `MultimediaData` instances that correspond to these root segments the decomposition pattern provides the `RootSegmentRole` concept. Note that `RootSegmentRoles` are not defined by `Methods` which describe `SegmentDecompositions`. They are rather defined by `Methods` which cause the production of multimedia content. These methods as well as annotation modes which allow the description of the production process (e.g. MPEG-7 (2001), part 5, section 9) are currently not covered by our ontology. Nevertheless, the prerequisite for enhancing the COMM into this direction is already given.

The decomposition pattern also reflects the need for localizing segments within the input segment of a decomposition as each `OutputSegmentRole` `requires` a `MaskRole` that classifies some `DigitalData` which expresses one `LocalizationDescriptor`. The latter describes the location of a segment, e.g., start and end time of an audio segment. In our example, we did not include this required information in order to reduce the complexity of the diagram.

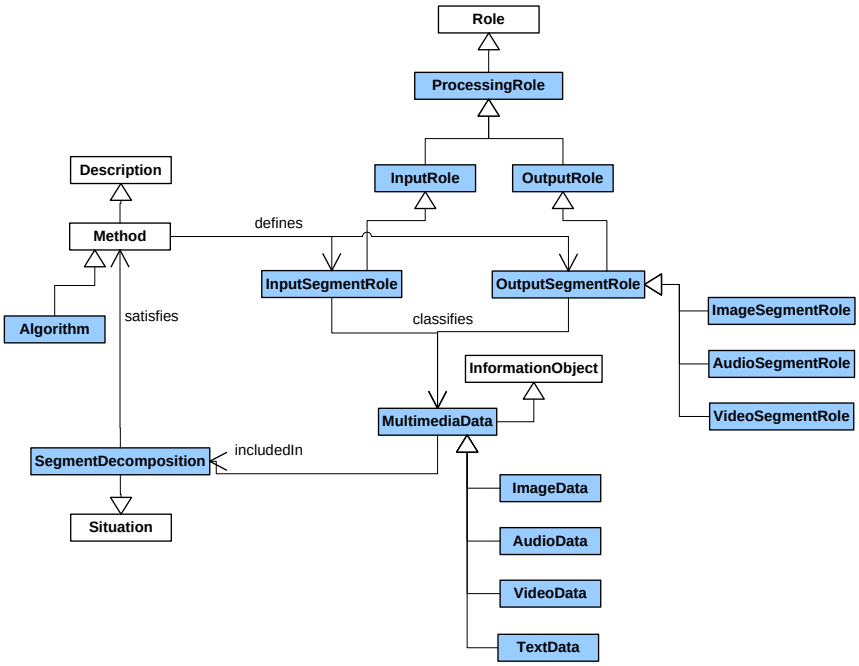


Figure 21: Decomposition pattern

5.2.3 Annotation Pattern

In this section, we describe the attachment of metadata, i.e., annotations to both `MultimediaData` or `Media` as depicted in Figures 22, 23, and 24. In general, we distinguish three annotation patterns. The content annotation pattern models the annotation of `MultimediaData` with metadata represented as `DigitalData`, i.e., in general MPEG-7 descriptors such as a dominant color descriptor. The media annotation pattern is used to attach metadata to the media, e.g., the file name or file size. Finally, the semantic annotation pattern formalizes the semantic annotation of `MultimediaData`. A semantic annotation refers to some individual of a domain ontology.

Each annotation pattern consists of an `Annotation` (subclass of `Situation`) that satisfies a `Method`. The `Method` defines an `InputRole` that is specialized to either `AnnotatedDataRole` or `AnnotatedMediaRole`, depending on what is annotated. Furthermore it specifies at least one `OutputRole`, which is specialized to `AnnotationRole` for the content annotation pattern and media annotation pattern and specialized to `SemanticLabelRole` in case of the semantic annotation pattern, respectively. In the following, we discuss the individual patterns in more detail.

We start with the content annotation pattern (cf. Figure 22). As `ContentAnnotations` we understand `Situations` that include annotated media as `MultimediaData` classified by an `AnnotatedDataRole`. The metadata is represented by `DigitalData`, which is classified by `AnnotationRoles`.

The roles are defined by an `Algorithm` or a `Method`, respectively. The actual metadata that is represented by a `DigitalData` entity depends on the `StructuredDataDescription` that the metadata comprises. These structured descriptions are formalized using the digital data pattern (see Section 5.2.1) and typically refer to MPEG-7 descriptors, although this is not required. The content annotation pattern is used to represent rather technical metadata such as extracted low-level features.

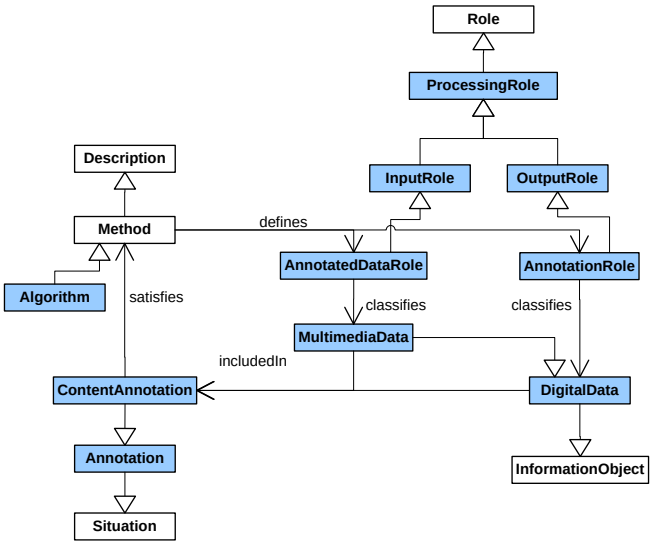


Figure 22: Content annotation pattern

The media annotation pattern forms the basis for describing the physical instances of multimedia content (cf. Figure 23). It differs from the content annotation pattern in only one respect: it is the `Media` that is being annotated and therefore plays an `AnnotatedMediaRole`. The situation is specialized to a `MediaAnnotation`. An example of the application of this pattern is given in Figure 5 in Section 3. The example depicts a file realizing an audio recording. The audio recording is annotated with the MPEG-7 media profile descriptor, which in the example contains the URI to the file.

Finally, COMM also provides the semantic annotation pattern depicted in Figure 24. MPEG-7 provides the means to model semantics as MPEG-7 descriptors (see MPEG-7 (2001), Part 5, Section 12). However, in the context of an ontology-based approach like COMM the integration of domain-specific ontologies is more appropriate then using the MPEG-7 descriptor for semantics. Thus, for the semantic annotation COMM relies on domain-specific ontologies that represent, e.g., real world entities that are depicted in the annotated multimedia content. Consequently, the semantic annotation pattern specializes the content annotation pattern to allow the connection of multimedia descriptions with domain descriptions provided by independent domain ontologies.

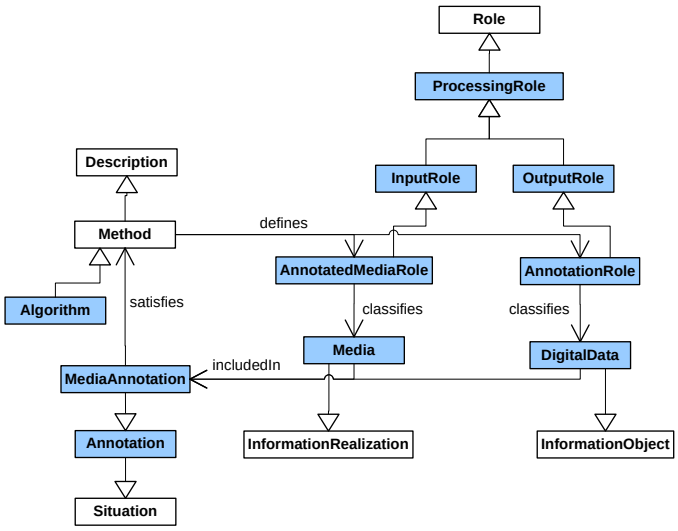


Figure 23: Media annotation pattern

5.3 X-COSIMO—Core Ontology for Personal Information Management

The Cross-Context Semantic Information Management Ontology (X-COSIMO) provides a formally precise representation of personal information and associated tasks to foster its reuse across personal information management applications such as task managers, email clients, and file management tools. It is used among others for personal information management in the X-COSIM semantic desktop (Franz et al., 2007). In the following, we detail on two of the patterns provided by X-COSIMO that are employed to describe the emergency scenario, namely the communication pattern and the task pattern.

5.3.1 Communication Pattern

The aim of the communication pattern is to enable a unified view onto communication and to represent information dealt in the context of communication for reuse in further contexts while maintaining information linkage. For achieving the first, the communication pattern provides a conceptual view of communication as developed by Jakobson (Jakobson, 1960). In his work, Jakobson defines the concept of a `Message` that is about something which he calls `Context`. The context is transmitted via a `Contact`—a connection between the `Addresser` and `Addressee`—and that is expressed within a `Code`. Such a model generalizes communication so that arbitrary communication modes such as chat, phone, and email can be represented consistently. For achieving the second goal, the communication pattern is designed using the DnS pattern as a `CommunicationDescription` (cf. Fig. 25) that represents the communication model of Jakobson.

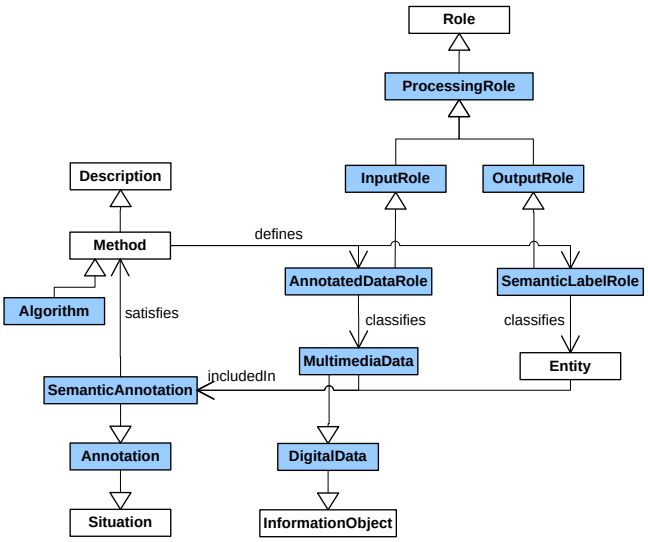


Figure 24: Semantic annotation pattern

The `CommunicationDescription` defines the roles `Addresser`, `Addressee`, `Contact`, and `Message` as common roles for any kind of communication as defined by Jakobson. Jakobson's concept of `Context` and `Code` are not defined by the description as existing ontologies for media annotation and decomposition such as COMM express these aspects in a more general way and are applicable also to other contexts than communication. Additionally, a `CommunicationDescription` also defines the `CommunicationParameter` `ConversationStart` and `AddressParameter`, which express constraints on the classified `CommunicationEvent` and the `Agents` that occur in the `CommunicationSituation`. Precisely, it requires that `Agents` have an `Address` quality that is contained in the `AddressSpace` `parameterized` by the `AddressParameter`. Similarly, `CommunicationEvents` are required to have a `TemporalQuality` that is within the `TimeInterval` `parameterizedBy` the parameter `ConversationStart`. The quality then expresses temporal information about the time at which a conversation took place. `Agents` have the role of addressers and addressees, while descriptions of protocols play the role of `Contact`, e.g., a description of the simple mail transfer protocol plays the role of contact for email communication. The `Message` role is played by an `InformationObject` that is separated from its realization that is accessible by a `DigitalRealization`.

Next to the communication roles, the `CommunicationDescription` also defines a `CommunicationCourse` which classifies a `CommunicationEvent`. A `CommunicationEvent` can be a conversation that spans several exchanges of messages. While the `CommunicationDescription` represents the abstract model of Jakobson, it subsumes more specific descriptions of particular communication modes, e.g., representing email communication or instant message communication.

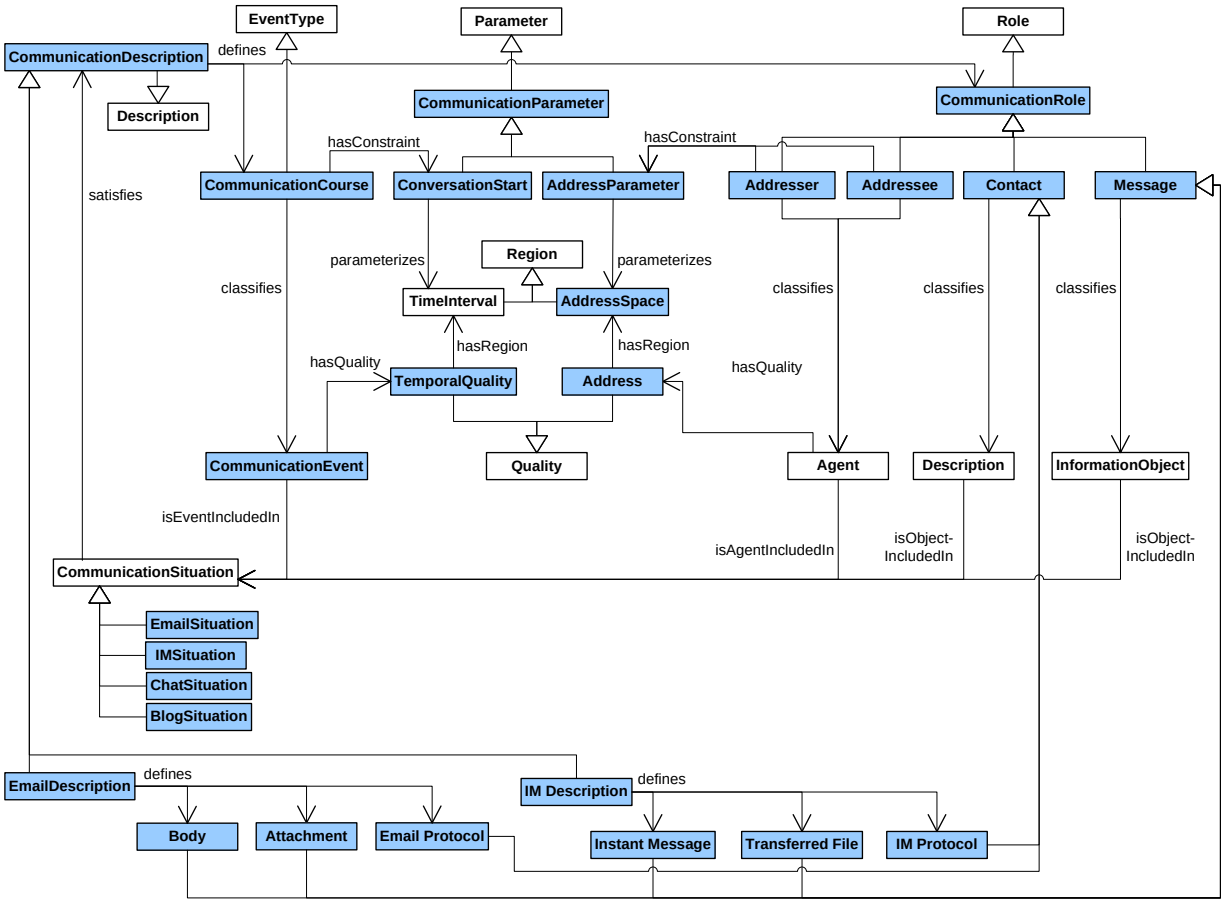


Figure 25: Communication pattern

Descriptions of such communication modes define more specific roles than the abstract `CommunicationDescription`, e.g., the `EmailDescription` defines the role of an `Attachment` and of an `EmailProtocol` as more specific roles of `Message` and `Contact`.

Applying the communication pattern for representing communication information, it can be reused in further contexts, classified according to any classification scheme and linked with further information. Moreover, disadvantages from the separation of different communication modes are alleviated as messages of any mode are represented by a common conceptual model for communication that results in a unified view onto communication. As an example, a conversation where `a-1` sends an email to `b-1` while `b-1` replies to `a-1` by an instant message would be represented by the instantiation of two `CommunicationDescriptions`, one `EmailDescription` and one `IMDescription`. These would define roles such as `Addresser` and `Addressee` that are played by the instances of the class `Agent` representing `a-1`

and b-1. A CommunicationCourse will also be defined that classifies a CommunicationEvent that has all the players of the roles defined by the particular communication description as participants. That same CommunicationEvent is also classifiedBy a communication course in the second description, i.e., the IMDescription. Within the communication course of the instant messaging, the IMDescription has the players of the roles as further participants. Thus, conversational threads can be represented in spite of the use of different communication modes that are based on different addressing mechanisms, communication protocols, and message serializations.

5.3.2 Task Pattern

The task pattern of X-COSIMO has been designed by extending the plan ontology (Gangemi et al., 2005). The refinement is conducted alongside the requirements imposed by the functionalities towards process support as defined in the X-Media project (Uren et al., 2006). Similar to the communication pattern, the DnS pattern is employed to represent a task-specific context onto the information, agents, and actions that take part in the execution of processes. The task pattern is depicted in Figure 26.

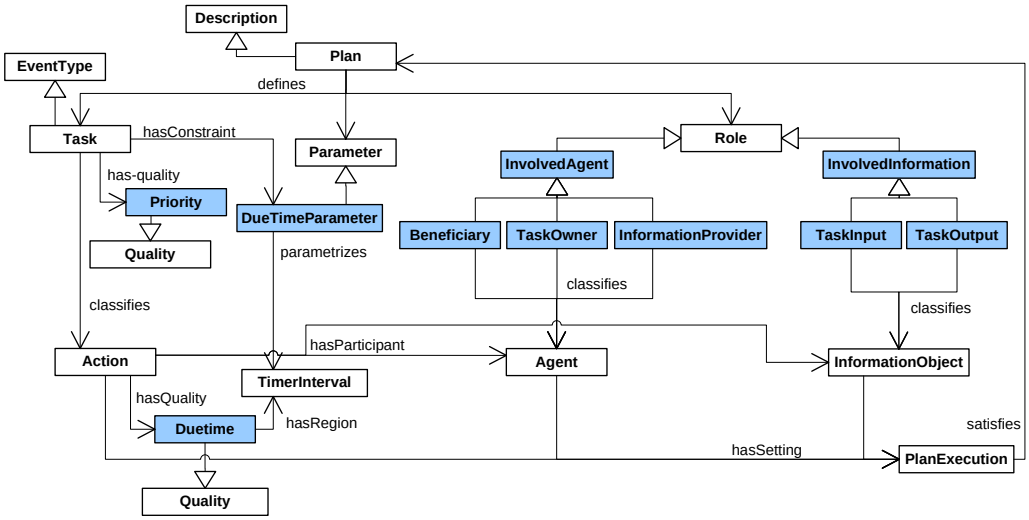


Figure 26: Task pattern

The notion of a task is expressed in terms of DnS as a Task that classifies Actions. Actions represent activities that take place within the execution of a Plan. For instance, the event that is classified as a communication course with respect to communication may be regarded as an action with respect to a particular task and process. Different information objects and agents take part in an action. This is represented using the hasParticipant property. As tasks are commonly prioritized, the quality Priority associates a priority value to a task. Actions are associated to a

Duetime, modeled as a quality of an action. The value of a due time is parametrized by a DueTimeParameter that is a constrained for a task.

Several notions of agents are also defined by the pattern, namely Beneficiary, TaskOwner, and InformationProvider. These are roles of agents and distinguish between different attitudes and stati of agents that take part in a process. The role of a TaskOwner classifies agents that take responsibility for a task, e.g., by ensuring timely or correct execution of a task. The role of an InformationProvider characterizes contributing participation as given, e.g., by domain experts that are consulted to gain further inside knowledge required to execute a task successfully. A similar notion of some agent providing information could also be represented by an additional sub-task associated to the information provider. However, we have introduced the role to make the status of such agents explicit. The role of a Beneficiary classifies agents that do not directly participate in actions, but anyway have an attitude towards a task. To give an example, a Beneficiary can be a customer that orders a product. While the customer does not participate in the manufacture of the product, the customer might expect certain quality levels, delivery times, and thus influences the process of manufacture. Next to agents, information can be involved in the execution of a process, either as input or as output of a process. Subclasses of InvolvedInformation classify information objects accordingly. The concepts like Task and Beneficiary are always defined by an instance of a Plan. A Plan is subclass of a Description that defines agent-driven-roles, tasks, and involved information. The corresponding Situation is given by the class PlanExecution that provides a setting for the classified activities, agents, and information.

6 Comparison with Core Ontology Properties

We have motivated the need for representing and sharing complex, structured knowledge by the use of core ontologies at the example of a socio-technical system for emergency response in Section 2. The use and design of our core ontologies Event-Model-F, COMM, and X-COSIMO has been shown in Section 3. These core ontologies have been designed, implemented, and evaluated over the last years in different European projects such as the WeKnowIt project, the X-Media project, and the network of excellence K-Space. Based on our and reported experiences in designing core ontologies, we have introduced our design approach for core ontologies in Section 4. The concrete design of our core ontologies following this approach has been described in Section 5.

In this section, we compare our Event-Model-F, COMM, and X-COSIMO with the properties of core ontologies introduced in Section 1. These properties are axiomatization and formal precision, modularity, extensibility, reuseability, and separation of concerns. We show examples of how the properties have been put into practice and summarize with a discussion from designing and applying our core ontologies in European projects.

6.1 Axiomatization and Formal Precision

Core ontologies shall provide a high degree of axiomatization and formal precision. This is required to establish a common understanding in a particular field and to ensure interoperability through machine accessible semantics. The axiomatization and formal precision of our core ontologies is achieved by basing the Event-Model-F, COMM, and X-COSIMO on the foundational ontology DOLCE. By this, our core ontologies inherit the rich axiomatization and formal precision of the foundational ontology. In addition, further axioms have been added to the core ontologies to precisely define the structured knowledge in the particular field they cover. These axioms have been specified using description logics (Baader et al., 2003). When precisely specifying the semantics of ontology design patterns with description logics one encounters the question of how many axioms are needed in order to sufficiently define the semantics of a pattern. This is of particular interest for patterns that base on the Descriptions and Situations (DnS) pattern. The `Description` defines a number of concepts that determine the roles of the entities in the pattern. These roles make up the structured knowledge represented by the pattern. For example, the axiomatization of the `EventCausalityDescription` of the causality pattern described in Section 5.1.3 defines the three roles `Cause`, `Effect`, and `Justification` as shown below (a full axiomatization can be found in (Scherp et al., 2009c)).

$$EventCausalityDescription \sqsubseteq \forall defines.(Cause \sqcup Effect \sqcup Justification)$$

No other roles except `Cause`, `Effect`, and `Justification` are allowed in the causality pattern. The other patterns of the Event-Model-F are defined following the same approach. Also the patterns of X-COSIMO are specified very tightly using description logics. For example, the communication pattern described in Section 5.3.1 defines that there can only be a `CommunicationCourse`, `ConversationStart`, `AddressParameter`, `Adresser`, `Adressee`, `Contact`, and `Message`. Other roles are excluded. This tight axiomatization is required for a semantically precise specification and use of the patterns. For example, an ontology engineer might want to use the pattern in a context in which it is not designed for. He might want to add some concepts to it that do not fit the design of, e.g., the causality pattern like introducing aspects of correlation to it. By a high amount of axiomatization, we reduce the risk of wrongly applying the ontology design patterns.

On the other hand, it might be interesting in the future to add some additional information to the patterns like provenance information, i.e., meta-information about the method and parameter setting by which the concrete pattern instantiations have been created. For example, the annotation pattern of COMM described in Section 5.2.3 could be extended to provide information about the source and confidence of the annotations. Thus, the current axiomatization of COMM does not define a restricted set of roles in its patterns. For example, the semantic annotation pattern allows to integrate further roles to the already defined ones, namely `AnnotatedDataRole` and `SemanticLabelRole`.

It is a principal design decision whether the patterns of a core ontology are closed with respect to additional roles such as the Event-Model-F and X-COSIMO or if roles can be added to the patterns like in COMM. There is no universal design solution to

the question of how many axioms are needed in order to sufficiently define the semantics of a pattern. However, the discussion shows that the amount of axiomatization of a pattern is dependent on how much one wants to exactly define and restrict the pattern's context of use and how open it shall be for future extensions. In the case of the Event-Model-F and X-COSIMO, a very tight specification of the patterns has been chosen in order to best capture the theories that are represented by the structure of the patterns like the philosophical question of causality in the case of the causality pattern in the Event-Model-F and Jakobson's communication theory in the case of the communication pattern of X-COSIMO. With the annotation pattern of COMM, only a loose specification has been conducted in order to allow extensions of COMM towards various specific features of existing metadata models and metadata formats.

The axiomatization of our core ontologies is available online from our website and can be investigated in detail in the ontologies themselves: `http://west.uni-koblenz.de/Research/ontologies/`. In addition, an extensive description of the axiomatization of the Event-Model-F can be found in (Scherp et al., 2009c).

6.2 Modularity

Foundational ontologies model the very basic and general concepts and relations (Borgo and Masolo, 2009; Oberle, 2006) that make up our world. They are hard to learn and to understand how to apply them to build a core ontology or domain ontology. Thus, there is a high learning curve when applying a foundational ontology. To alleviate this situation, a good foundational ontology like DOLCE already makes use of ontology design patterns such as DnS and IO.

Based on the very generic ontology design patterns provided by DOLCE, we have carefully chosen the scope of our core ontology design patterns. The patterns provide a solution to a recurrent ontology modeling problem such as participation of objects in events in the Event-Model-F, the annotation of media data in COMM, and the communication between agents in X-COSIMO. Thus, the scope for each pattern is chosen that its functionality can be encapsulated and provided by a distinct service. They are neither too generic such that they cannot be applied or specialized in a specific domain nor are they too specific such that they are limited in their use. Finally, it is also important to mention that there is no overlap between the scope of the patterns. Thus, there are no redundant patterns. By defining the different ontology design patterns in a well-chosen scope, we provide a modularization of our core ontologies.

6.3 Extensibility

Modularity is a prerequisite to allow core ontologies to be extensible towards new developments and functional requirements that arise (cf. adaptability in (Vrandečić, 2009)). Throughout the lifecycle of ontologies, they need to be extended and modified in order to adapt them to new requirements that may have emerged over time. With their pattern-oriented design, the core ontologies Event-Model-F, COMM, and X-COSIMO are well prepared to handle such new requirements and extensions. With respect to extensibility of our core ontologies, several examples and different work can

be mentioned. We describe how we have dealt with newly arising requirements and how we have extended our core ontologies towards them.

An example for extending existing ontology design patterns is from the X-Media project. Here, new requirements for media annotation have been raised. While the COMM ontology supports the description of image data and video data as shown in Section 5.2, the scenarios in X-Media additionally required to describe annotations of textual and numerical data, e.g., textual data given in the open document format. The annotation of portions of the data, e.g., a text passage or a subset of numerical data, also required to describe the decomposition of the data into segments. The extension of COMM towards these media types has resulted in the Ontology for Knowledge Acquisition (OAK) (Iria, 2009). Concepts defined in COMM have been specialized in OAK in order to adapt the core ontology to the requirements on the integration of knowledge acquisition tools. Analogue to the concepts `Image` and `Video`, the concept `Text` has been introduced in COMM. Like `Image` and `Video`, it is a subclass of `Media` that realizes `TextData`. `TextData` is a specialization of the COMM concept `MultimediaData`, which represents information objects. `MultimediaData` already has subclasses such as `ImageData` and `VideoData` as shown in Section 5.2.2. Furthermore, new locators for textual data have been defined, analogue to the locators already existing for image data and video data. For the description of the annotation and decomposition of the newly added media types, existing COMM patterns have been reused.

With respect to adding additional ontology design patterns to an existing core ontology, we consider the Event-Model-F. In the initial design of our Event-Model-F, we did not foresee the documentation pattern described in Section 5.1.4. Our technical report documents this initial design of the Event-Model-F where the documentation pattern is missing (Scherp et al., 2009b). The documentation pattern allows to associate documentary evidence with an event, thus it constitutes the missing link that connects the Event-Model-F with COMM. The documentation pattern of the Event-Model-F described in Section 5.1.4 and used for modeling the emergency response scenario in Section 3 has only been added later (Scherp et al., 2009a). To extend the initial version of the Event-Model-F by an additional pattern and integrating this additional pattern with the core ontology the following steps have been performed: In a first step, the existing patterns of the Event-Model-F have been checked to which extend they already provide the functionality that shall be added with the new pattern. As there is no overlap, the documentation pattern has been designed and carefully aligned to DOLCE+DnS Ultralight in a second step. Thus, we did not need to change or adapt the existing patterns. By using DOLCE as common basis, the documentation pattern could directly be used with the Event-Model-F.

Finally, our core ontologies are extended in a research project conducted in collaboration with the arts department of the University of Koblenz-Landau. The researchers of the arts department are interested in the history of art and conducting iconographic research of scientific art pieces. They analyze the depiction of scientific concepts such as the atom model and track how the depiction has changed over time due to scientific inventions and discoveries. A web-based system has been developed for the collaborative discussion of the scientific art pieces and representing different interpretations on them depending on the point of views the researchers might have. In this work, a

pattern-based domain ontology for the history of scientific art has been developed. This domain ontology is based on DOLCE+DnS Ultralight and uses and extends multiple patterns provided by the Event-Model-F and COMM. For example, the life of a person that is relevant in the history of science such as researchers and inventors are modeled as specialization of the Event-Model-F composition pattern. To model a person's life, the pattern defines specific events such as birth, death, events in education, inventions, and scientific discoveries. In addition, the interpretation of scientific art pieces is not always clear and is typically subject to discussion. Here, the Event-Model-F interpretation pattern is applied to model the different opinions the experts have. The metadata information about the digital media data representing the scientific art pieces in the system are modeled using COMM. Here, no changes or specializations of the core ontology have been conducted. Thus, COMM is reused without modifications. There are also new patterns introduced for the domain ontology. These patterns model additional aspects that are not covered by the scope of our three core ontologies. For example, an ontology design pattern is added for modeling the change of qualities of events and objects over time like the name of a researcher that has changed due to marriage. The patterns of the domain ontology are axiomatized using description logics (Baader et al., 2003) to provide a formal semantics of the structural knowledge captured by the system.

6.4 Reuseability

Closely related to extensibility is the property of reuseability. Being modularly defined, a core ontology supports reuse of its ontology design patterns in various different domains. At the same time it still guarantees formal precision of the overall knowledge it represents. Reuseability of our core ontologies is considered in two kinds: First, the structured knowledge defined by core ontologies can be reused (and extended) towards newly arising requirements and specific domains. The structured knowledge of our three core ontologies Event-Model-F, COMM, and X-COSIMO have already been reused in different applications and domains as the discussion in Section 6.3 on extensibility shows. This reuse of the existing ontology design patterns has been conducted without any changes to the particular patterns. For example, the modeling of the emergency response scenario in Section 3 demonstrates the use of various ontology design patterns such as the decomposition pattern of COMM, the task pattern of X-COSIMO, and the participation pattern of the Event-Model-F without the need to modify them. In addition, the existing ontology design patterns have been reused by specializing concepts defined in the patterns. This is the case with the example of COMM to add support for textual data, which has been added with the development of OAK. For example, the concept `TextData` has been introduced as specialization of `MultimediaData` and the concept `Text` as subclass of `Media`.

Second, besides the reuse of the structured knowledge defined in the core ontologies, they are also able to incorporate existing domain knowledge. They make use of that domain knowledge rather than requiring to remodel it. The emergency response scenario modeled in Section 3 demonstrates the reuse of domain ontologies within our core ontologies. This reuse of domain knowledge takes place with the roles defined in the patterns. For example, modeling the participation of the citizen Paul in the event of

a snapped power pole in Figure 2 reuses a domain ontology for roles in emergency response. Thus, in the concrete example the citizen `paul-1` is classified by the concept `CitizenRole` taken from the domain ontology. Other roles in this ontology are the `FiremanRole` that `paul-1` can play in other situations as he is also a professional fireman.

Another possibility for reusing domain knowledge takes place with the real-world entities that are described by the patterns, i.e., classified by the roles. For example, the composition pattern of the Event-Model-F depicted in Figure 13 shows a representation of a larger flooding event that is composed of smaller incident events. Here, the `Composite` role classifies the individual `flooding-1` that is of event type `Flooding`. The `Flooding` event is taken from a domain ontology for emergency response that is provided by one of the partners in the WeKnowIt project. Similar to this example is the use of the domain-specific concept `PowerPole` to represent the `power-pole-1` in the participation pattern at the beginning of Section 3. The `PowerPole` is defined in an external domain ontology and reused here.

Besides the demonstrated reuses of domain knowledge in the Event-Model-F, domain ontologies are also reused in COMM and X-COSIMO. In the modeling example of Section 3, an existing repository containing operators at the emergency hotline is reused to annotate the segments of the audio recordings. In Figure 8, Rita is identified in the call segment `audio-segment-1` and thus `rita-1` is associated with the segment. The repository of operators at the hotline is not aligned with DUL. It uses the concept `Person` to represent the individual operators. Thus, the `SemanticLabelRole` in Figure 8 classifies the concept `Person` instead of DUL's `NaturalPerson`. Although `rita-1` is of concept `Person`, we can reuse the individual as label in the semantic annotation pattern. Since `rita-1` plays only in the context of a semantic annotation the `SemanticAnnotationRole`, we do not cause any unwanted inferences or infer any additional class memberships. Besides such closed data repositories like the repository containing the operators at the hotline, one can also make use of linked open data such DBpedia (Bizer et al., 2009).

In the task pattern of X-COSIMO depicted in Figure 12 a domain-specific ontology for tasks in emergency response is applied. Such an ontology allows to distinguish between different types of `Tasks` such as to confirm a situation, inspect a problem, contacting people, and others. The example in Figure 12 shows an `InspectionTask` that is to be carried out by the floating liaison officer `marie-1` to inspect a snapped power pole.

6.5 Separation of Concerns

The structured knowledge of our core ontologies is clearly separated from the domain-specific knowledge that is reused. As we have shown in the previous section, our core ontologies allow to apply different kinds of domain-specific knowledge and domain-specific ontologies. By the separation of concerns, the domain-specific knowledge is integrated and reused without affecting the core ontology itself. This is achieved by using the DnS ontology design pattern for representing the structured knowledge of the concrete field captured by our core ontologies (see Section 4.2.5). The roles that are defined by the `Description` specify the structure of the ontology design pattern,

i.e., represent the structured knowledge of the core ontology. The domain knowledge is only referred to by the roles defined in the pattern. In addition, the individuals and the concepts of the entities classified by the roles may come from a domain ontology. By using the DnS pattern, this domain specific knowledge does not affect the structured knowledge of the core ontology. Independent whether domain ontologies are used for the roles or entities in the patterns, the roles that are defined by the `Description` remain. In addition, with respect to the separation of concerns it does not make a difference if the used domain ontology is aligned to DOLCE or not. The separation of concerns in our core ontologies Event-Model-F, COMM, and X-COSIMO is shown in Section 6.4 at the examples of reusing domain ontologies. The examples demonstrate the reuse of existing domain ontologies for the roles defined in the DnS-based patterns of the Event-Model-F, COMM, and X-COSIMO and the entities classified by the roles in these patterns.

The separation of concerns is nicely visualized in the example of the participation pattern in Section 5.1.1. The `EventParticipationDescription` defines the different roles required in an event participation situation, namely a `DescribedEvent` and a `Participant`. As shown in Figure 15, subconcepts of the `Participant` role can be defined in some domain-specific ontology. For example, an emergency response ontology that defines different emergency roles like the `CitizenRole` and `FiremanRole`. Besides reusing domain ontologies for specifying the roles in the participation pattern, also the classified entities such as the described event and its participating objects in the participation pattern can be provided by some domain ontology. Such a domain ontology can provide both individuals as well as concepts that are reused with the pattern. Independent of the domain ontologies actually used, the participation pattern always foresees two roles of exactly one `DescribedEvent` role and at least one `Participant` role.

6.6 Concluding Discussion

The core ontologies we have developed, namely COMM, X-COSIMO, and Event-Model-F are employed in larger European projects as briefly described in Section 2. Partly, they are in use since 2007. Thus, it is first of all important to mention that since their creation no major changes on the principal design of our core ontologies were necessary. In the course of applying and extending our core ontologies, no additional patterns had to be introduced. Thus, no essential patterns or functionality is missing in our core ontologies to serve the purpose they have been designed for. This means, the scope of our core ontologies has been well chosen. For example, COMM is designed to provide support for annotations of media. In the X-Media project the requirement of new media types arised that so far were not supported by COMM, namely text and numerical data (see Section 6.3). However, no new patterns needed to be added in COMM for providing support for text and numerical data. Rather, the existing patterns for decomposition and annotation could be reused. For the interpretation of scientific art pieces, the modeling of peoples' life with events was implemented as specialization of the Event-Model-F composition pattern. Thus, none of the additions required the modification of existing patterns of the core ontology. In addition, none of them introduced inconsistencies or violated the intentional use of any of the patterns defined by

our core ontologies. In cases where new ontology design patterns have been created such as the change of qualities over time, we can state that these resulted from requirements that are not within the scope of our core ontologies presented here. Rather, such newly introduced patterns complement our three core ontologies and are already designed such that they are interoperable with them. This shows that our core ontologies are of some stability and allow for reuse of the structured knowledge defined in the ontology.

Designing core ontologies based on an existing foundational ontology is a cumbersome and tedious task. It requires excellent knowledge of the foundational ontology used as modeling basis. In addition, it requires to precisely align the core ontology with the foundational ontology. Core ontology designers also need in depth knowledge about the field that is to be covered by the core ontology they design. Here, an extensive analysis of the related work is necessary, i.e., the analysis of existing models and ontologies in the field as well as existing applications from different domains in the field. Based on this analysis, the functional requirements to the core ontology can be derived. A core ontology covers a specific field that is an abstraction of different concrete domains (see Section 4.1). Thus, it requires a reasonable level of abstraction when designing it.

7 Beautiful Aspects of Our Core Ontologies

We have presented a design approach for developing core ontologies in Section 4.2 that meets all requirements posed in Section 1. These requirements are axiomatization and formal precision, modularity, extensibility, reuseability, and separation of concerns. We have exemplified core ontologies developed according to this design approach in Section 5 and have highlighted how they fulfill the requirements in Section 6. Core ontologies that have been created following this design approach and fulfilling the requirements above can be easily combined with each other to describe highly complex knowledge structures although they have been designed for different purposes and for different fields. Concrete examples demonstrating how our core ontologies are combined in the emergency response scenario are presented in Section 7.1. In Section 7.2, we compare our approach with the use of other, more domain-oriented ontologies to model complex scenarios and argue for the benefits of our core ontologies.

7.1 Combining Core Ontologies

The core ontologies Event-Model-F, COMM, and X-COSIMO presented in Section 5 have been designed for different purposes and to provide support for different fields. However, by the nature of their design they can be flexibly combined. The ability for a flexible combination is essential for facilitating the representation of complex, structured knowledge that spans across several different domains. As an example of the application of our core ontologies, we have illustrated how they support the representation of complex, structured knowledge in an emergency response system in Section 3. However, they have also been applied successfully to support complex systems in the aviation and automotive industry. In the following, we refer to the concrete emergency

scenario from Section 3 to highlight the beauty in the integration and smooth interplay of our core ontologies. It becomes visible when different patterns of different core ontologies share common real-world entities, namely events and objects.

For example, the audio recording at the emergency hotline illustrates these aspects. The information that an audio recording took place during a phone call is used within the documentation pattern of the Event-Model-F as shown in Figure 3. It is represented by `audio-rec-1` and is associated to the event of `paul-1`'s call to the emergency hotline to report about a snapped power pole. The same individual `audio-rec-1` representing the audio recording is used within the digital data pattern of COMM as depicted in Figure 5. The `audio-rec-1` of concept `AudioData` is a specialization of `DigitalData`, which is defined in the digital data pattern. Again, the same `audio-rec-1` is used in the communication pattern and task pattern of X-COSIMO for modeling a message and task description send between two emergency response officers as shown in Figures 11 and 12.

Another example demonstrating the interplay of the Event-Model-F and COMM is the representation of the citizen `paul-1`. The individual `paul-1` is used in the participation pattern of the Event-Model-F as shown in Figure 2. It represents the call of `paul-1` to the emergency response hotline to report about a snapped power pole. During the call a recording of the conversation between `paul-1` and the officer at the hotline answering the call takes place. In order to provide efficient access to the different parts of the conversation, the recorded call is segmented and annotated with the speakers' names. Here, the participation pattern of the Event-Model-F interplays with the semantic annotation pattern of COMM as shown in Figure 7. Each segment of the audio call is annotated with a semantic label representing the person speaking. In our example, the `audio-segment-2` is annotated with the individual `paul-1`.

The interplay between the Event-Model-F and X-COSIMO is nicely shown in Figure 10. It depicts the attachment of Paul's interpretation `inter-sit-1` for the power outage event to the task pattern. The interpretation is modeled using the Event-Model-F, whereas the task pattern is defined in X-COSIMO. The task pattern is depicted in Figure 12 and takes Paul's interpretation for the power outage as `TaskInput`. The example shows that not only some individuals can be shared between our different core ontology patterns. It demonstrates that even entire instantiations of the patterns can be integrated and reused in other patterns. In our example, an instantiation of the event interpretation pattern of the Event-Model-F identified by the individual `inter-sit-1` is reused in the task pattern of X-COSIMO.

7.2 Comparison with other Ontologies

Besides our core ontologies, there exist also other ontologies for modeling events such as the Event Ontology[15] and the Linking Open Descriptions of Events (LODE)[16] ontology. For representing multimedia metadata we find ontologies like the W3C Ontology for Media Resource[17]. Finally, there are also ontologies supporting knowledge

[15] `http://motools.sourceforge.net/event/event.html`
[16] `http://linkedevents.org/ontology/`
[17] `http://www.w3.org/TR/2010/WD-mediaont-10-20100309/`

management such as the Personal Information Model (PIMO)[18]. These ontologies are suitable for being applied in the specific domain they have been designed for. Some of these ontologies are designed within the context of a specific project such as the Event Ontology in the context of a music management system and PIMO as part of the semantic desktop NEPOMUK. Other ontologies like LODE and the Ontology for Media Resource aim at integrating existing ontologies in their domain. The complexity of these ontologies is often low, i.e., they typically consist only of a few concepts and properties. Some of these ontologies exist already for a while like the Event Ontology that has been first released in 2004 and is already well known in the community.

These ontologies may be a good choice for modeling the domain of emergency response as they provide for representing events, multimedia metadata, and personal information. However, these ontologies are not designed to collaborate and to be combined in a complex, socio-technical system such as emergency response. Although some of these ontologies reuse existing ontologies like the Event Ontology reuses the Friend-of-a-Friend[19] vocabulary it remains unclear how they should be connected with each other. For example, the Event Ontology does not know about PIMO and the Ontology for Media Resource. LODE and the Ontology of Media Resource allow for integrating other ontologies, but this is limited to ontologies in their specific domain. They also do not know how to handle PIMO and other ontologies in further domains.

If one wants to combine these ontologies in a concrete information system they have to be harmonized and integrated a posteriori. However, this can become a big challenge as the ontologies typically do not follow a systematic development approach and are often semantically ambiguous. For example, the semantics of the `factor` and `product` properties of the Event Ontology is quite vague and no axiomatization of the causal relation expressed with these properties is provided. The property `location` of the Ontology of Media Resource is provided to express where a specific media asset has been recorded. However, this property may cause confusion as it is often mixed up with the location the content is about. For example, US president Barack Obama may make some statements in a video captured at the White House in Washington about events happening in the area of the Middle East. Matching the concepts and properties of the different ontologies also becomes a challenging task. For example, it is unclear which properties of the Ontology of Media Resources that are related to *agents* such as `contributor`, `creator`, `publisher`, and `targetAudience` should be aligned with the `involvedAgent` property of LODE to describe events. In the worst case, the ontologies that shall be combined need to be redesigned from scratch.

In contrast, our core ontologies provide a precise semantics by a rich axiomatization and follow a common design approach. By the nature of their design, our core ontologies allow to be combined in a complex scenario such as emergency response although the core ontologies have been designed for different fields. This is achieved by using a common foundational ontology. Thus, the beauty of our core ontologies lies in their ability to be combined within complex scenarios. This is achieved just by the way how we design the core ontologies. We belief that it is better to design

[18] `http://www.semanticdesktop.org/ontologies/pimo/`
[19] `http://www.foaf-project.org/`

ontologies such that they are prepared for being combined with others like our core ontologies rather than integrating ontologies a posteriori in order to use them together in a concrete information system.

8 Conclusions

We have presented the design and use of three core ontologies, the Event-Model-F, COMM, and X-COSIMO. These core ontologies are characterized by a high degree of axiomatization and formal precision. This is achieved by basing on the common foundational ontology DOLCE+DnS Ultralight. Our core ontologies follow a pattern-oriented design approach, which make them modular and extensible. In addition, they support reuse of the structured knowledge defined within the core ontologies as well as support reuse of existing domain ontologies. By applying the Descriptions and Situations pattern of DOLCE+DnS Ultralight they support the separation of concerns, i.e., clearly separating the structured knowledge defined in the core ontology from the domain-specific aspects. Due to these characteristics, our core ontologies allow for both formally conceptualize their particular fields and to be flexibly combined to cover the needs of concrete, complex application domains. Thus, from our perspective they are to be considered *beautiful* ontologies.

With the design of our core ontologies, we contribute to the overall discussion and methodologies for improving ontology design and ontology quality such as OntoClean (Guarino and Welty, 2002) and DILIGENT (Pinto et al., 2009) and bring the field further forward from an art to an engineering discipline. As such, our core ontologies are of interest to the ontology engineering community. We assume that in the future many further beautiful core ontologies will emerge and that the need to combine multiple core ontologies to support the information exchange in complex applications will increase. With beautiful core ontologies such as the Event-Model-F, X-COSIMO, and COMM, we can fulfill the requirement of future information systems to model and exchange very complex, structured information in a heterogeneous setting of different client applications and systems in use. The core ontologies Event-Model-F, COMM, and X-COSIMO have been created in the Web Ontology Language[20] (OWL) and axiomatized using description logics (Baader et al., 2003). The core ontologies as well as tool support we have developed are available online from our website: `http://west.uni-koblenz.de/Research/ontologies/`.

Acknowledgements

This research has been co-funded by the EU in FP7 in the WeKnowIt project (215453) and in FP6 in the X-Media project (026978).

[20] `http://www.w3.org/2004/OWL/`

References

Arndt, R., Troncy, R., Staab, S., and Hardman, L. (2009). *Handbook on Ontologies*, chapter COMM: A Core Ontology for Multimedia Annotation. Springer, 2nd edition.

Arndt, R., Troncy, R., Staab, S., Hardman, L., and Vacura, M. (2007). COMM: Designing a well-founded multimedia ontology for the web. In *ISWC*. Springer.

Ashburner, M. (2000). Gene ontology: Tool for the unification of biology. *Nature Genetics*, 25:25–29.

Baader, F., Calvanese, D., McGuinness, D. L., Nardi, D., and Patel-Schneider, P. F., editors (2003). *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press.

Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., and Hellmann, S. (2009). DBpedia: A crystallization point for the web of data. *Web Semant.*, 7(3):154–165.

Bloehdorn, S., Petridis, K., Saathoff, C., Simou, N., Tzouvaras, V., Avrithis, Y., Handschuh, S., Kompatsiaris, Y., Staab, S., and Strintzis, M. G. (2005). Semantic annotation of images and videos for multimedia analysis. In *European Semantic Web Conference*. Springer.

Borgo, S. and Masolo, C. (2009). *Handbook on Ontologies*, chapter Foundational choices in DOLCE. Springer, 2nd edition.

Cote, R., Rothwell, D., Palotay, J., Beckett, R., and Brochu, L. (1993). The systematized nomenclature of human and veterinary medicine. Technical report, SNOMED International, Northfield, IL: College of American Pathologists.

Doerr, M., Ore, C.-E., and Stead, S. (2007). The CIDOC conceptual reference model: a new standard for knowledge sharing. In *Conceptual modeling*, pages 51–56. Australian Computer Society, Inc.

Ekin, A., Tekalp, A. M., and Mehrotra, R. (2004). Integrated semantic-syntactic video modeling for search and browsing. *IEEE Transactions on Multimedia*, 6(6):839–851.

Euzenat, J. and Shvaiko, P. (2007). *Ontology matching*, chapter Classifications of ontology matching techniques. Springer.

Francois, A. R. J., Nevatia, R., Hobbs, J., and Bolles, R. C. (2005). VERL: An ontology framework for representing and annotating video events. *IEEE MultiMedia*, 12(4).

Franz, T., Staab, S., and Arndt, R. (2007). The X-COSIM integration framework for a seamless semantic desktop. In *Knowledge capture*, pages 143–150. ACM.

Gamma, E. (2007). *Design patterns : elements of reusable object oriented software*. Addison-Wesley.

Gangemi, A. (2008). Norms and plans as unification criteria for social collectives. *Autonomous Agents and Multi-Agent Systems*, 17(1):70–112.

Gangemi, A., Borgo, S., Catenacci, C., and Lehmann, J. (2005). Task taxonomies for knowledge content. Deliverable 07 of the EU FP6 project Metokis, http://metokis.salzburgresearch.at/.

Gangemi, A., Fisseha, F., Keizer, J., Lehmann, J., Liang, A., Pettman, I., Sini, M., and Taconet, M. (2004). A core ontology of fishery and its use in the fishery ontology service project. In Borgo, A. G. S., editor, *EKAW*04 Workshop on Core Ontologies in Ontology Engineering*, Northamptonshire, UK. CEUR Proceedings Vol. 118.

Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., and Schneider, L. (2002). Sweetening Ontologies with DOLCE. In *Knowledge Engineering and Knowledge Management*, pages 166–181. Springer.

Gangemi, A. and Mika, P. (2003). Understanding the semantic web through descriptions and situations. In *CoopIS/DOA/ODBASE*, pages 689–706. Springer.

Gangemi, A. and Presutti, V. (2009). *Handbook on Ontologies*, chapter Ontology Design Patterns. Springer, 2nd edition.

Garcia, R. and Celma, O. (2005). Semantic integration and retrieval of multimedia metadata. In *Knowledge Markup and Semantic Annotation Workshop*, Galway, Ireland.

Guarino, N. and Welty, C. (2002). Evaluating ontological decisions with OntoClean. *Commun. ACM*, 45(2):61–65.

Herre, H., Heller, B., Burek, P., Hoehndorf, R., Loebe, F., and Michalek, H. (2006). General formal ontology (GFO): A foundational ontology integrating objects and processes. part I: Basic principles (version 1.0). Onto-Med Report 8, Research Group Ontologies in Medicine (Onto-Med), University of Leipzig.

Hunter, J. (2005). *Multimedia Content and the Semantic Web: Standards, Methods and Tools*, chapter Adding Multimedia to the Semantic Web: Building an MPEG-7 ontology, pages 261–283. John Wiley & Sons.

IPTC (2008). EventML. http://iptc.org/.

Iria, J. (2009). A core ontology for knowledge acquisition. In *European Semantic Web Conference*, pages 233–247.

Isaac, A. and Troncy, R. (2004). Designing and using an audio-visual description core ontology. In *Workshop on Core Ontologies in Ontology Engineering, CEUR Proceedings Vol. 118*, Whittlebury Hall, Northamptonshire, UK.

Itkonen, E. (1983). *Causality in Linguistic Theory*. Indiana Univ. Press.

Jain, R. (2008). EventWeb: Developing a human-centered computing system. *Computer*, 41(2):42–50.

Jakobson, R. (1960). *Style in Language*, chapter Closing statement: Linguistics and Poetics, pages 350– 377. Cambridge, Mass., MIT Press.

Kundu, S., Itkin, M., Gervais, D., Krishnamurthy, V., Wallace, M., Cardella, J., Rubin, D., and Langlotz, C. (2009). The IR radlex project: An interventional radiology lexicon. *Vascular and Interventional Radiology*, 20(4):433–435.

Lagoze, C. and Hunter, J. (2001). The ABC ontology and model. In *Proc. of the Int. Conf. on Dublin Coreand Metadata Applications*, pages 160–176. National Institute of Informatics, Tokyo, Japan.

Lenat, D. B., Guha, R. V., Pittman, K., Pratt, D., and Shepherd, M. (1990). Cyc: toward programs with common sense. *Commun. ACM*, 33(8):30–49.

Lombard, L. (1986). *Events: A metaphysical study*. Routledge & Kegan Paul.

Masolo, C., Borgo, S., Gangemi, A., Guarino, N., and Oltramari, A. (2003). WonderWeb deliverable D18 ontology library; IST WonderWeb project.

Mika, P., Oberle, D., Gangemi, A., and Sabou, M. (2004). Foundations for service ontologies: aligning owl-s to dolce. In *World Wide Web*. ACM.

MPEG-7 (2001). Multimedia content description interface. Technical report, Standard No. ISO/IEC n15938.

Mueller, E. T. (2008). *Handbook of Knowledge Representation*, chapter Event Calculus. Elsevier.

Niles, I. and Pease, A. (2001). Towards a standard upper ontology. In *Formal Ontology in Information Systems*, pages 2–9, New York, NY, USA. ACM.

Oberle, D. (2006). *Semantic Management of Middleware*. Springer.

Oberle, D., Ankolekar, A., Hitzler, P., Cimiano, P., Sintek, M., Kiesel, M., Mougouie, B., Baumann, S., Vembu, S., and Romanelli, M. (2007). DOLCE ergo SUMO: On foundational & domain models in the SmartWebIntegrated Ontology. *Web Semant.*, 5(3):156–174.

Oberle, D., Grimm, S., and Staab, S. (2009a). *Handbook on Ontologies*, chapter An Ontology for Software. Springer, 2nd edition.

Oberle, D., Guarino, N., and Staab, S. (2009b). *Handbook on Ontologies*, chapter What is an Ontology? Springer, 2nd edition.

Oberle, D., Lamparter, S., Grimm, S., Vrandečić, D., Staab, S., and Gangemi, A. (2006). Towards ontologies for formalizing modularization and communication in large software systems. *Appl. Ontol.*, 1(2):163–202.

Pinto, H. S., Tempich, C., and Staab, S. (2009). *Handbook on Ontologies*, chapter Ontology Engineering and Evolution in a Distributed World Using DILIGENT. Springer, 2nd edition.

Quinton, A. (1979). Objects and events. *Mind*, 88(350):197–214.

Raimond, Y. and Abdallah, S. (2007). The event ontology. http://motools.sf.net/event.

Rector, A. L. and Horrocks, I. R. (1997). Experience building a large, re-usable medical ontology using a description logic with transitivity and concept inclusions. In *Workshop on Ontological Engineering*. AAAI Press.

Rosse, C. and Mejino, J. L. V. (2003). A reference ontology for bioinformatics: The foundational model of anatomy. *Biomedical Informatics*, 36:478–500.

Scherp, A., Franz, T., Saathoff, C., and Staab, S. (2009a). F-a model of events based on the foundational ontology DOLCE+DnS Ultralight. In *Accepted for publication at the International Conference on Knowledge Capturing (K-CAP), Redondo Beach, CA, USA*.

Scherp, A., Franz, T., Saathoff, C., and Staab, S. (2009b). A model of events based on a foundatiional ontology. Technical Report 2/2009, Arbeitsberichte aus dem Fachbereich Informatik, University Koblenz-Landau, Koblenz, Germany. ISSN 1864-0850.

Scherp, A., Papadopoulos, S., Kritikos, A., Schwagereit, F., Saathoff, C., Franz, T., Schmeiss, D., Staab, S., Schenk, S., and Bonifacio, M. (2009c). D5.2.1 prototypical knowledge management methodology (intermediate report). Technical report, WeKnowIt Project. http://www.weknowit.eu/sites/default/files/D5.2.1.pdf.

Schneider, L. (2003). Designing foundational ontologies - the object-centered high-level reference ontology OCHRE as a case study. In *Conceptual Modeling - ER 2003, 22nd International Conference on Conceptual Modeling*. Springer.

Staab, S., Scherp, A., Arndt, R., Troncy, R., Grzegorzek, M., Saathoff, C., Schenk, S., and Hardman, L. (2008). Semantic multimedia. In *Reasoning Web; San Servolo Island, Venice, Italy*. Springer.

Tsinaraki, C., Polydoros, P., and Christodoulakis, S. (2004). Integration of OWL ontologies in MPEG-7 and TVAnytime compliant semantic indexing. *Advanced Information Systems Engineering, 16th International Conference, CAiSE04*, pages 398–413.

Uren, V., Petrelli, D., Cimiano, P., Franz, T., Lanfranchi, V., Gilardoni, L., Busse, J., Slavazza, P., Dadzie, A.-S., Ciravegna, F., Lei, Y., Rabus, D., and Thanh, T. D. (2006). X-media deliverable 4.1: Specification of knowledge sharing systems. Technical report, X-Media Consortium.

Vrandečić, D. (2009). *Handbook on Ontologies*, chapter Ontology Evaluation. Springer, 2nd edition.

Wang, X., Mamadgi, S., Thekdi, A., Kelliher, A., and Sundaram, H. (2007). Eventory – an event based media repository. In *Semantic Computing*, pages 95–104. IEEE.

A.2 Appendix on the Multimedia Metadata Ontology (M3O)

M3O—Describing the Semantics of Rich Multimedia Presentations with the Multimedia Metadata Ontology

Ansgar Scherp, Carsten Saathoff
University of Koblenz-Landau, Germany
Tel.: +49-261-287-2717
Fax: +49-261-287-100-2717
{scherp,saathoff}@uni-koblenz.de

Abstract

Existing metadata models and metadata standards are not sufficient to describe the semantics of rich, structured multimedia content in formats such as SMIL, SVG, LASeR, XMT-Ω, and Flash. They are either conceptually too narrow, focus on a specific media type only, cannot be used and combined together, or are not practically applicable. This hampers the retrieval of such presentations by search engines today and makes their archival and management a difficult task. Existing W3C standards for rich multimedia presentations like SMIL and SVG and the industry standards LASeR and XMT-Ω foresee the use of the Resource Description Framework to describe the semantics of the multimedia content. However, until today there is no appropriate metadata model or best practice available that explains how to describe and annotate rich, structured multimedia content. To fill this gap, we propose the Multimedia Metadata Ontology (M3O) for annotating rich, structured multimedia presentations. The M3O provides a generic modeling framework for representing sophisticated multimedia metadata. It allows for integrating the features provided by the existing metadata models and metadata standards. With the M3O, we make the semantics of rich multimedia presentations machine-readable and machine-understandable. An application programming interface (API) for our M3O provides the tool for adding rich semantic descriptions of multimedia content into arbitrary multimedia applications. The API can also be used within generic multimedia tool support like the SemanticMM4U framework for the multi-channel generation of semantically-rich multimedia presentations. The framework has been successfully applied for the development of multimedia applications in domains such as personalized sports news, context-aware tourist guides, and the generation and semantic enrichment of personal photo albums.

1 Introduction

Multimedia metadata and semantic annotation of multimedia is a key-enabler for improved services on multimedia content. If there is no or only limited metadata and annotations provided, the archival, retrieval, and management of multimedia content becomes very hard if not practicably infeasible. Rich, structured multimedia content is encoded by the combination of at least one continuous media asset like audio and video and one discrete media asset such as text and image [56]. The media assets are arranged in time and space into a coherent multimedia presentation in formats such as the Synchronized Multimedia Integration Language (SMIL) [63], Scalable Vector Graphics (SVG) [65], Lightweight Application Scene Representation (LASeR) [32], Extensible MPEG-4 Textual Omega (XMT-Ω) [36], and Flash [1]. Annotation of such rich, structured multimedia content is the association of metadata to the structured content and its media assets [40].

Rich, structured multimedia presentations constitute a "black box" for information retrieval today. They cannot or only to a limited extend be understood by search engines. Multimedia formats such as the W3C standards SMIL and SVG, the industry format Flash by Adobe, and the MPEG standards like LASeR and XMT-Ω forsee the use of Semantic Web technologies for annotating the content using the Resource Description Framework (RDF) [60]. However, there is currently no appropriate model provided or best practice available that explains how to describe and annotate such rich, structured multimedia content in the web. The existing metadata models such as [27, 5, 39, 25, 26] and metadata standards like [34, 2, 30, 42] are either conceptually too narrow, semantically ambiguous, focus on a specific media type only, cannot be used and combined with each other, or are not practically applicable for the semantic description of rich multimedia presentations in the web. For example, image descriptions using EXIF [34] cannot be combined with MPEG-7 [42] descriptors. In IPTC [30], the location fields are defined to contain the locations the content is "focusing on". However, it remains unclear what "focusing on" actually means. For instance, consider an image from the atomic bombing of the city of Nagasaki[1] in Japan in 1945. This image is about the city of Nagasaki since it documents an event taking place in that city. But it is also about the USA since they have dropped the atomic bomb on the city of Nagasaki. Distinguishing these different roles a location can play is impossible with IPTC and others. Here, support for semantic annotations with formally defined background knowledge needs to be provided. However, this is hardly found in the existing models. In addition, none of the existing models and standards explicitly support the distinction between information objects and information realizations [13]. An information object such as an image is often available in different formats and resolutions, i.e., in different information realizations. However, this feature is often requested by conceptual multimedia metadata models [33, 25]. In addition,

[1] http://commons.wikimedia.org/wiki/File:Nagasakibomb.jpg, from Wikimedia Commons. The image is in the public domain.

most metadata models focus on a single media type only. Thus, they ignore the type's relation to other media types and the media assets' context within a rich, structured multimedia presentation. These metadata models are not designed to be combined with each other. This results in a disconnectedness of today's models. However, this combination is required by multimedia applications that integrate, e.g., image data and video data, in particular in the open world of the web. In addition, most existing models do not support representing both high-level semantic annotation with background knowledge as well as the annotation with low-level features extracted from the multimedia content.

This situation is very unfortunate as the authoring of rich, structured multimedia content can be quite expensive and providing support for annotating rich content not only improves archival, retrieval, and management of the content, but also allows for better reuse. With the Multimedia Metadata Ontology (M3O), we propose an approach for annotating rich, structured multimedia content in the web and unlocking its semantics by making it machine-readable and machine-understandable [46, 45]. The M3O allows for a sophisticated semantic description of rich, structured multimedia content. It provides a generic modeling framework that can accommodate and integrate the features provided by the different multimedia metadata models and metadata standards we find today. The M3O bases on Semantic Web technologies and thus can be easily integrated with today's presentation formats like SMIL and SVG. For designing the M3O, we conducted an analysis of related work and extracted the common data structures that underlie the existing metadata models and metadata standards. We represent these common data structures in form of *ontology design patterns* (ODPs) [20] based on the formal upper-level ontology DOLCE+DnS Ultralight[2]. Similar to design patterns in software engineering [19], ODPs provide a modeling solution to a recurrent ontology design problem [20]. DOLCE+DnS Ultralight is a lightweight version of the Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE) [13]. As upper-level ontology, DOLCE aims at modeling the very basic and general concepts and relations [13, 44] that make up our world such as objects, events, participation, and parthood. Part of DOLCE+DnS Ultralight is the ODP Descriptions and Situations (DnS) [20]. It provides a formal representation of contextualized views on the relations of a set of individuals by assigning roles or types to the individuals that are only valid within a given context. By employing ontology design patterns and basing on a formal upper-level ontology, the M3O can serve as a reference framework for modeling the annotation of rich, structured multimedia content.

Implementing the M3O using Semantic Web technologies is a promising approach as it allows for representing sophisticated multimedia annotations. Semantic Web technologies ease the use of formal domain ontologies, leverage the employment of reasoning services, and provide the means to exploit the rapidly growing amount of Linked Open Data (http://linkeddata.org/) available on the web. The M3O is agnostic to the source of the annotations and decompositions.

[2] http://www.ontologydesignpatterns.org/wiki/Ontology:DOLCE+DnS_Ultralite

They may be generated by automatic processes or manually created by humans. The distinguishing features of our Multimedia Metadata Ontology are:

- The explicit distinction of information objects and information realizations. This is important to explicitly deal with the different kinds of realizations an information object can have.
- Support for annotating both information objects and information realizations. Thus, to attach metadata to the multimedia content and its media assets.
- Representing high-level annotations as well as low-level annotations. This feature is required to support a big variety of multimedia annotations as it is supported by today's models and formats.
- Support for decomposing the rich, structured multimedia presentations into its single media assets. Like annotation, this can be applied on information objects and information realizations.
- Representing collections of information objects and information realizations. Collections are supported by many media management applications and tools, but rarely supported by the metadata models.
- Capturing of provenance information for the annotations, decompositions, collections, and the origins of the media assets themselves. Provenance information is very valuable and important for an improved archival, retrieval, and management of the content.
- Representing of confidence information of annotations, decompositions, collections, and origins of media assets. Confidence information is an explicit representation of the level of trustworthiness of information. It is important to support as often it is not clear to which extend one can rely on the provided information such as an annotation or decomposition.

It is important to note that we do not propose yet another model on multimedia metadata. Rather than replacing any of the existing models, the M3O aims at integrating and representing the metadata and data structures that underlie the existing approaches. This is achieved by the formal nature of the M3O and by following a pattern-oriented design approach. The problem of annotating rich, structured multimedia presentations such as SMIL, SVG, and Flash is not new. However, until today it remains an unsolved problem and there is no appropriate model or best practice available that can be used to describe and annotate structured content in the web. With the M3O, we aim at providing a model for annotating structured multimedia content and to unlock its semantics for a better archiving, retrieval, and management of the content. The M3O continues our prior work on multimedia annotation and bases on the experiences gained with developing the Core Ontology on Multimedia [5].

The remainder of the paper is organized as follows: In the next section, we motivate the need for the M3O by a concrete scenario. The related work is

reviewed in Section 3. The requirements to the M3O are presented in Section 4. They have been derived from the scenario and the related work. The ontology design patterns of the M3O are introduced in Section 5. The fulfillment of the requirements by our Multimedia Metadata Ontology and further non-functional features are discussed in Section 6. Section 7 describes the experiences made with defining the patterns of the M3O and argues for the benefits following our approach. The axiomatization of the M3O using description logics [6] is discussed in Section 8. In Section 9, we discuss the scenario from Section 2 again and demonstrate the application of the M3O to the concrete scenario. In Section 10, we demonstrate how annotations represented in the M3O are integrated into SMIL, SVG, and Flash. The use and implementation of the M3O in our SemanticMM4U framework for the multi-channel generation of semantically-rich multimedia presentations is described in Section 11, before we conclude the paper.

2 Scenario

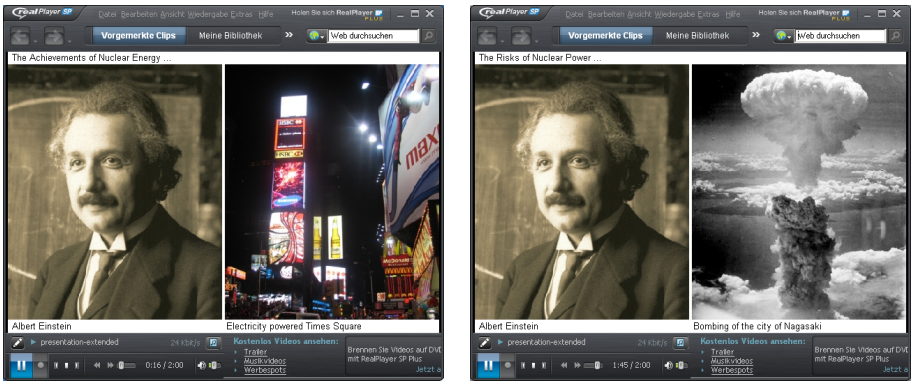
In this section, we introduce a small scenario that demonstrates the added benefit of rich semantic annotations for structured multimedia content. The scenario involves John, a nuclear physicist, who was asked by his son's school principal to give a talk about the history of nuclear energy for the schools anniversary celebration. He prepares a multimedia presentation in SMIL (cf. Figure 1) about the history of nuclear energy, but also mentions the downsides of this technology. Such a SMIL presentation is a composition of several individual media assets like images and texts into a coherent, structured multimedia document. It possesses a spatial layout of the media assets, arranges them into a temporal course, and also allows for some interaction with the content. The presentation is rendered using the RealPlayer[3].

Among others, the presentation contains two parts that discuss and visualize the positive effects of nuclear energy and the risks. The first part (cf. Figure 1a) shows a picture of Albert Einstein[4] and a photo of the Times Square in New York. This part of the presentation serves as metaphor for the achievements reached by the discovery of nuclear energy in which Einstein played a central role. By the peaceful use of nuclear energy, it can serve large cities like New York with electricity, which is one of the basic supplies for a high quality of living.

In the second part of our SMIL presentation (cf. Figure 1b), we replace the photo of the Times Square by a picture showing the atomic bombing of the city of Nagasaki in Japan in 1945. The picture of Einstein remains unchanged. However, the contextual use in which the picture of Einstein is shown is completely different. He is now used as a scientist who contributed to the invention of such a terrifying weapon. Instead of showing the advantages of nuclear energy,

[3] RealNetworks, Inc., http://www.real.com/realplayer/

[4] http://en.wikipedia.org/wiki/File:Einstein1921_by_F_Schmutzer_4.jpg, the image is in the public domain.



(a) (b)

Figure 1: An image of Albert Einstein combined with an image of the Times Square and an image of the nuclear bomb cloud expressing contrary views on *nuclear energy*. The screenshot on the left hand side showing Albert Einstein and the Time Square in New York serves as a metaphor for scientific development and quality of living achieved with nuclear energy. Albert Einstein and the nuclear bomb cloud depicted in the screenshot on the right hand side is a metaphor for the risks and destructive nature of nuclear energy.

this part of the presentation serves as metaphor for the risks and the potential destructive power of nuclear energy.

By the change of contextual use, the media assets transmit a totally different message and express different semantics [53]. John uses these two parts to discuss with the class that scientific results can often be used to do positive and negative things, independently of what the scientists originally wanted to achieve. He hopes to trigger some discussion and reflection about this topic by using one of the best known scientists in such a contradictory manner.

John usually publishes all of his writings and presentations in the web using different platforms and formats. In order to support searching for his presentation on the web, he annotates his works. Everything of potential interest about the presentation shall be annotated. For providing such a comprehensive semantic description of this multimedia presentation, there are different kinds of annotations involved: (i) John would like to annotate the different parts of the presentation individually. (ii) He wants to express that the two parts discussed above are about the positive and negative aspects of nuclear energy, respectively. (iii) In addition, he would like to annotate the individual media assets of the presentation such as the pictures of Einstein, the Times Square, and the atomic cloud with background information. For example, he likes to annotate the picture showing the atomic cloud with the historic event of the bombing of the city of Nagasaki in 1945. In addition, he may want to annotate text assets used in the presentation with their bibliography (this is not further considered

in this paper for reasons of brevity). (iv) Furthermore, John would like to add metadata to the images that represent among others the place of capture or the creator. (v) John reuses some images from Wikipedia and his personal photo archive, so he would like to point to the locations where these images can be accessed individually. (vi) The images that John uses to present the potentials and risks of nuclear energy such as the Times Square in New York and the depiction of the atomic bombing of Nagasaki are part of a collection of images John has been collecting for a while already. He wants to explicitly represent that these pictures are part of a collection that he has created. (vii) John also adds provenance information about himself to the presentation and the metadata such that other people can assess the source of the statements made in the presentation. This provenance information allows people to judge whether they trust the provided information or not. (viii) Finally, also explicit information about the confidence of annotations is added to the semantic description of the presentation. For example, the location annotation of the image showing the Times Square is annotated with a high level of trust as John has seen many other images of the times square before. In addition, the GPS information extracted from the image supports that it has been taken in New York.

John uses the built-in functionality of his favorite multimedia authoring tool to create these annotations using Linked Open Data vocabularies and appropriate ontologies. (ix) He publishes his presentations including the annotations in the formats SMIL and Flash on the web. He wants the metadata to reflect that the SMIL and Flash files are realizations of the same presentation. Some weeks later, John sees on the web that the journalist Mary is working on an article about the risks that accompany scientific research. (x) Marie has initiated to collect media that depict examples of scientists that are or were involved in the development of potentially dangerous technologies. John decides to add some images of his collection to Marie's media collection. Since Marie requires reliable information, she is only interested in media provided by experts. Due to the rich annotation and provenance information John provides with his collection, Mary decides that she can trust this source. She selects images of John's presentation and reuses them in her article.

The metadata created and associated with multimedia presentations such as in the scenario above can be used for retrieval of multimedia content in several ways. In the following, we describe some example queries related to the scenario and explain their usefulness.

- Show me presentations that relate Albert Einstein to social risks. This query is conduced by John when he prepares his talk about the history of nuclear energy. He wants to know if other people have created similar presentations or articles about that issue, which he could use for inspiration or reuse.

- Show me all pictures that Mary contributed to the collection. This query is interesting when one wants to know the provenance of contributions to a collection. For example, one might want to explicitly see only contributions of a specific person or source.

- Show me presentations that use Ferdinand Schmutzer's images. This query is conducted when someone wants to know, e.g., how popular Schmutzer's work is.

The scenario puts different requirements to the metadata model used to represent the semantics of the multimedia content shown in the presentation. Further requirements are derived from the related work, which is presented in the following section.

3 Related Work

Metadata are generally defined as "data about data" (cf. [23]). Numerous metadata models and metadata standards have been proposed in research and industry. These models come from different backgrounds and with different goals. They vary in the domain for which they have been designed and can be domain-specific or for general purpose. The existing metadata models also focus on a specific single media type such as image, text, or video and are not designed for annotating rich, structured multimedia presentations such as SMIL, SVG, and Flash. In addition, the metadata models differ in the complexity of the data structures they provide. With standards like EXIF [34], XMP [2], and IPTC [30], we find metadata models that provide (typed) key-value pairs to represent metadata of the media type image. Harmonization efforts like the Metadata Working Group[5] are very much appreciated. However, they remain on the same technological level and do not extend their effort beyond the single media type of image. Similar standards exist for other media types, such as ID3 [43] for audio files. Like EXIF, ID3 provides a predefined list of key-value-pairs to annotate audio files and allows for defining custom metadata fields. Yahoo! SearchMonkey Media [66] is an RDF vocabulary providing a set of classes and a set of properties describing media assets. It does not provide any additional functionality compared to EXIF or ID3. However, it covers further media types like video and text. Other metadata models like Dublin Core [16] support hierarchical modeling of key-value pairs. It can be used to describe arbitrary resources. However, it is designed to annotate only entire documents and not parts of it. In addition, as it is very generic it covers only a small fraction of the metadata needed to sufficiently annotate rich, structured multimedia content.

With MPEG-7 [42], we find a comprehensive metadata standard that aims at covering mainly decomposition and description of low-level features of audiovisual media content. MPEG-7 also provides basic means for semantic annotation. Several approaches have been published providing a formalization of MPEG-7 as an ontology [15], e.g., by Hunter [28, 27], Bloehdorn et al. [10], Garcia and Celma [21], Isaac and Troncy [31], DS-MIRF [57], or the Core Ontology on Multimedia (COMM) [5]. In contrast to other approaches to modeling MPEG-7 as an ontology [28, 10, 21, 31, 57], COMM is not designed as a one-to-one mapping,

[5] http://www.metadataworkinggroup.org/

but provides a set of patterns that cover the core and repetitive building blocks of MPEG-7. Although these ontologies provide clear semantics for the multimedia annotations, they still focus on MPEG-7 as the underlying metadata standard.

Most metadata models lack in supporting structured multimedia content. Annotation of such structured multimedia content is in principle possible with MPEG-7 by considering the multimedia content as a media stream that can be decomposed. However, conducting such a decomposition for a complex structured multimedia presentation is not very practical in MPEG-7 due to the nature of annotations in MPEG-7 and the complexity involved with these annotations. In MPEG-7 a structured multimedia presentation such as a Flash or SMIL presentation is considered an audiovisual media stream. In order to associate MPEG-7 annotations with this stream, the audio-visual stream would be appropriately decomposed and annotated with the MPEG-7 metadata. This way of annotating media content is not enforced by MPEG-7 and could be done differently. However, it would be done like this in order to provide support for annotations similar to the way metadata are associated with Flash or SMIL. For example, the multimedia presentation from Section 2 rendered by the Real Player would be temporally decomposed into some time-variant streams of the first and second part of the presentation. In addition, the streams would be spatially decomposed into the left image, right image, and the textual media assets involved. Each stream would be annotated in MPEG-7 with the appropriate metadata at the time when it is rendered to the users. This approach is not very practicable, as the annotations have to be associated to the content each time the presentation is rendered. However, the multimedia annotations are available a-priori to the rendering of the presentations. Thus, they can already be associated with and stored together with the multimedia content beforehand. This approach is followed by today's presentation formats such as SMIL and SVG and is implemented with the M3O.

From the existing metadata standards and metadata models, only the Functional Requirements for Bibliographic Records (FRBR) [29] and COMM consider the separation of information objects and information realizations [13]. Examples of information objects are stories, stage plays, or narrative structures. John's presentation, e.g., could be seen as an information object of a narrative structure telling the history of nuclear energy. Each information object is realized by different so-called information realizations [13]. Only a realization brings something abstract such as a message into the real world and makes it perceivable by humans (or any kind of agent). However, it is not fully supported in FRBR as the annotations can only be applied on the information objects. Also COMM does not fully support the separation of information objects and information realizations. The decomposition and annotation can only be applied on information object level. Thus, it is not possible to individually annotate, e.g., the different realizations of the same image information object. This is very unfortunate, as the decomposition of information realizations depends on the realization's resolution and others.

The existing metadata models and metadata formats also hardly integrate high-level and low-level features, i.e., the integration of representing both the features that can be extracted from the media assets as well as the annotation with semantic background knowledge. This is unfortunate, as studies have shown the need for semantic annotation [24, 55] and conceptual queries, e.g., in image retrieval [39, 25, 26]. Finally, a multitude of conceptual models for multimedia metadata have been proposed such as [33, 37]. However, they have not been operationalized or published as a standard or ontology.

The list of metadata models and metadata standards is far from being complete and is beyond the scope of this work. The examples discussed have been selected as representative to show the variety of the different metadata models and metadata formats for multimedia content that exist today. A comprehensive overview of multimedia metadata models and standards can be found in a report [11] of the W3C Multimedia Semantics Incubator Group. Subsequently, the current W3C Media Annotations Working Group has published an ontology for media resources on the web [38] that aims at providing a mapping vocabulary for a number of existing standards, including EXIF, IPTC, ID3, or Yahoo! SearchMonkey Media. However, the technical drawbacks of the existing standards are not tackled by this working group. Nonetheless, the proposed mapping vocabulary is a great source for defining refinements and specific extensions of the M3O. Building on top of this work will provide a core of mappings to many existing standards.

To summarize, the major standards for multimedia metadata are too focused on a single media type and too limited in their support for rich semantic annotations. MPEG-7 was developed to tackle at least the former problem, but is also clearly focused on decomposition and annotation with low-level features. Additionally, MPEG-7 is known to contain a large number of redundancies and ambiguities [5], which makes interoperability of MPEG-7 descriptions produced by different tools an issue. Furthermore, despite the existence of semantic annotation in MPEG-7, the provided solution is not integrated with the standards of the Semantic Web and thus hardly interoperable. The existing multimedia metadata ontologies such as [28, 10, 21, 31, 57] were developed in order to overcome these limitations. However, these ontologies are all based on MPEG-7. Even the COMM, which is a refactoring of MPEG-7 and not a mere mapping of MPEG-7 features to ontologies and that followed a formal ontology design approach, still contains a number of design artifacts and shortcomings. With the M3O, we aim at integrating the existing metadata models and metadata formats rather then replacing them. It specifies a formal reference model for metadata integration and metadata exchange and provides the means for describing rich semantics of structured multimedia content.

4 Requirements on a Multimedia Metadata Model

From the scenario and the discussion of related work, we derive the principal requirements that need to be supported for annotating rich, structured multi-

media content such as the SMIL presentation in the scenario. These need to be reflected by our multimedia metadata ontology.

REQ-1: Identification of Resources In the scenario, the presentation is published in different formats on different platforms (cf. (ix) in Section 2). Furthermore, the presentation uses other media elements that are available separately (cf. (v)). In order to be able to link these different resources and to coherently integrate the metadata, a universal identification mechanism is required that allows for identifying resources on the web. Only such a mechanism guarantees that the SMIL version of Johns presentation can be linked to the Flash version of the same presentation in a way that allows to infer the fact that both files realize the same presentation.

REQ-2: Separation between Information Objects and Information Realizations On the conceptual level, multimedia content conveys information to the consumer. As such, the multimedia content plays the role of a message that is transmitted to a recipient. Such a message can be understood as an abstract information object [13]. The multimedia content in our scenario above is, e.g., realized as a SMIL and a Flash presentation (cf. (ix)). This separation between information objects and information realizations is important, since it provides a clean distinction between the semantics and the data. What a media item expresses is independent of how it is realized. If we transformed the SMIL presentation into a video, it would not change its message.

REQ-3: Annotation of Information Objects and Information Realizations The model needs to support the annotation of multimedia content (cf. (ii)—(iv)). This support for annotation needs to be provided for both information objects and information realizations. Examples are annotations in the style of typed key-value pairs as provided by EXIF or the semantic annotation with background knowledge from DBpedia [9] for describing the multimedia content. In our example, the image from the Times Square would be annotated with the geo-coordinates it was taken at. The location where an image has been taken does not depend on how it is realized, e.g., as JPG or other. Thus, the location annotation is applied on the information object level and not attached to the information realizations. Specifically the attachment of low-level metadata such as resolution, color histograms, and others require means to represent arbitrary, possibly complex data values. Some low-level metadata such as color information or the file size are typically attached to the realization, since they depend on the concrete realization. The realization of an image as a JPG will very likely have another file size as the realization as a PNG.

REQ-4: Decomposition of Information Objects and Information Realizations Multimedia content can be decomposed into its constituent parts. The presentation in the scenario can be decomposed into the two parts discussing the chances and risks of nuclear energy (cf. (i)). Each part can be decomposed

into the images and other media assets they contain. The decomposition is important to refer to only the relevant parts of the presentation when applying some annotation. In the example above, e.g., only the first part is about the positive aspects of nuclear energy and the second about the negative ones. It is required to attach the annotations to the corresponding parts. If we annotated the presentation globally, we would not be able to relate the use of Albert Einstein and the Times Square in the context of the positive aspects of nuclear energy. A global annotation would just express the fact that the presentation is about positive and negative effects and that Albert Einstein, the Times Square, and the Nagasaki Bombing are depicted. It would not provide the same semantics as if we annotated the presentation parts individually. Decomposition can be applied arbitrarily often, i.e., one can create a hierarchy of parts. A clear separation between the information object and information realization is also important for the decomposition. For example, addressing a component is depending on the realization, e.g., physically addressing the first part is different in SMIL and in Flash. Whereas this does not matter for the information object.

REQ-5: Organizing Multimedia in Collections Multimedia content and in particular media assets are often organized in collections that are about a specific topic or theme. Different existing systems such as the media sharing platform Flickr[6] and Picasa[7] support the organization of media assets along a common concept. In the scenario, John uses media assets in his presentation to discuss about the advantages and disadvantages of nuclear energy that have been taken from such a collection (cf. (vi)). In addition, Marie has initiated an image collection of scientists that are or have been involved in the development of potentially risky technology (cf. (x)). The information that images are part of a common collection shall be made explicit. It can be used to provide a different view onto a presentation by showing, e.g., only those images of a presentation that originate from a collection. In addition, the collection where the images origin from is an entry point to search and browse for further related images a user might be interested in.

REQ-6: Representation of Provenance Information Provenance is of crucial importance in order to assess the source of an information (cf. (vii)) and to judge its reliability. This is in particular true for information taken from the web. It is relevant for the annotation of multimedia content and its media assets, their decomposition, as well as collections. Provenance information is very valuable and important for an improved archival, retrieval, and management of the content. However, is not mandatory to the annotation, decomposition, and collections and thus should be defined optional. John manually annotates the presentation with a concept from DBpedia about social risks. He includes himself as the provenance of this annotation. John also provides himself as creator of his collection of images about the potentials and risks of nuclear energy.

[6] http://www.flickr.com/
[7] http://picasa.google.com/

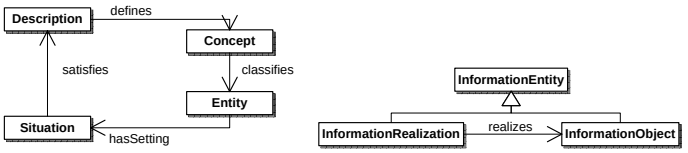
With additional knowledge about John, e.g., coming from his FOAF [14] file, a user can judge whether he has the required expertise. Based on the provenance information, the journalist Mary in our scenario can judge whether John has the expert knowledge she requires. Besides the manual provision of provenance information, it can also be provided by automatic methods. For example, an annotation or decomposition of an image can be produced by some analysis algorithm. Provenance information would include the algorithm type, the parameters it was run with, or maybe a reference to the training data that has been used.

REQ-7: Representation of Confidence Information In addition to provenance information, one may also want to add information about the confidence of an annotation, decomposition, or collection. Confidence information is an explicit expression of a level of trust about some other information. Thus, it is related to provenance information. In our scenario, John states with a high confidence level that the image of the Times Square he uses in his presentation actually has been taken in New York at that place (cf. (viii)). Thus, John provides some confidence information about the location annotation of the image based on his knowledge and experience. Such kind of confidence information is valuable to have and should be explicitly represented in an information system. It allows users to judge the trustworthiness of an information without looking at the entire provenance chain. Besides the manual provision of confidence information, it can also be provided by automatic methods. A classification algorithm for images writes the confidence of the classifications into the multimedia metadata. For example, an outdoor shot has been detected with an accuracy of 91% whereas a building has been determined with the confidence of 64%.

5 Multimedia Metadata Ontology

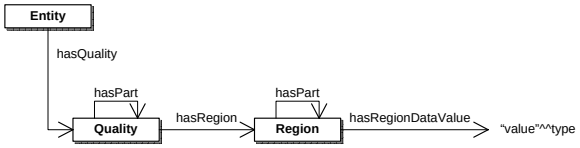
For defining our Multimedia Metadata Ontology (M3O), we leverage Semantic Web technologies and follow a pattern-oriented ontology design approach [20]. Our goal is not to provide an ontological representation of a *specific* metadata standard or conceptual model but to provide a common ontology covering media annotation and multimedia metadata in general. Thus, we have analyzed the existing standards and models in Section 3 in order to identify their common underlying structures. These standards are limited and different with respect to the type of media and the kind of metadata they support. However, common to them is that they assign *some metadata* to *some media*. Therefore, we provide a *pattern* that allows to accomplish exactly the assignment of arbitrary metadata to arbitrary media.

From our analysis, we have identified a set of patterns required to express metadata for multimedia content. These patterns model the basic structural elements of existing metadata formats and conceptual models such as structure, annotation, information realization, complex data values, provenance, and con-



(a) Description and Situation Pattern

(b) Information Realization Pattern



(c) Data Value Pattern

Figure 2: Patterns of the foundational ontology DOLCE+DnS Ultralight

fidence. In order to realize a specific metadata standard or metadata model in M3O, these patterns need to be specialized. The patterns base on the foundational ontology DOLCE+DnS Ultralight (DUL) [13] and are formalized using Description Logics [6]. By this, we provide clear semantics of the patterns and their elements. We have achieved an improved formal representation of the metadata compared to existing models. In addition, such a generic model is not limited to a single media type such as images, video, text, and audio. It provides support for structured multimedia content, i.e., a combination of several media assets coherently arranged in time and space as it can be created with today's multimedia presentation formats such as SMIL, SVG, and Flash.

In the following, we introduce three basic patterns from DOLCE+DnS Ultralight that we use for our model. Subsequently, we present the patterns provided by M3O for multimedia annotation, decomposition, collections, provenance, and confidence. We conclude this section by comparing the ontology against the requirements from Section 4.

5.1 DOLCE+DnS Ultralight (DUL) Patterns

The Descriptions and Situation (DnS) pattern [20] allows for the representation of contextualized views on the relations of a set of individuals and is depicted in Figure 2a. Since annotations might only be valid, interesting, or trusted within certain contexts, we have to consider the annotation itself within a context. This modeling requires reification, i.e., we have to be able to make statements about predicates. The DnS pattern gives us a formally sound reification mechanism. It provides a formally defined model to view relations among individuals within a context and assign roles or types that are only valid within this context.

The DnS pattern consists of a Situation that satisfies a Description. The Description defines the roles and types present in a context, called Concepts. Each Concept classifies an Entity. The entities are the individuals that are relevant

in a given context. The Concept can be seen as the type of the Entity within the specific context. Each Entity is connected to the situation via the hasSetting relation. Furthermore, the concepts can be related to other concepts by the isRelatedToConcept relation in order to express their dependency. The DnS pattern therefore expresses an n-ary relation among a set of entities. The concepts determine the roles that the entities play within this context.

As an example, we consider a semantic annotation expressing a view on the relation of Albert Einstein and the development of a nuclear bomb. One might be interested in the source of this annotation in order to judge whether it is trustworthy or not. Using a simple relation would not suffice since additional information about the context in which this annotation is asserted needs to be provided. Using Descriptions and Situations, we express the fact that a relation between the media and its annotation is valid within the context of the user, i.e., that the user who provides the annotation asserts this relation. We can then restrict a search only to certain users or types of users we trust.

The Information Realization Pattern in Figure 2b models the distinction between information objects and information realizations [13]. It consists of the InformationRealization that is connected to the InformationObject by the realizes relation. Both are subconcepts of InformationEntity, which allows treating information in a general sense. We will use this in our annotation and decomposition patterns, since the structure of an annotation or a decomposition is the same on both the information object and information realization levels. Information objects represent abstract ideas that are supposed to be communicated. An information object might be the image taken during holidays, in which case the abstract idea is the scene that we perceive and want to show (i.e. communicate) to our family or friends. Another example is the presentation from our scenario in Section 2, where the abstract idea that shall be communicated is the history of nuclear energy and the associated advantages and disadvantages. We have to realize this information in order to communicate it. The realization of an information object is called information realization. The presentation, e.g., is realized as both a SMIL file and a Flash file, but the message or abstract idea being communicated is the same. In other words, the same information can be realized in different ways and each realization communicates the same abstract idea or message.

This view on information objects and their realizations is already a simplification. From a philosophical point of view, it is debatable whether the information does change or does not change when it is realized using different media types. As an example, we consider a conference talk and a video recording of it. Is the same message conveyed by the live talk and the video, i.e., are both realizations of the *same* information object? The context of the two realizations and thus the experience the viewers make is certainly different. Thus, the different contextual settings of the talk and the video might already change the message. In the context of this work, we are mainly interested in information realization in the context of digital media and focus on modeling the storage of the same media item in different formats, resolutions, and others. These different versions are assumed to be very similar and therefore it

seems acceptable to assume that they realize the same information object. The problem of modeling information objects and their realization within different contexts is of fundamental nature. It should be addressed and solved on the level of a foundational ontology like DOLCE+DnS Ultralight and not within a multimedia core ontology such as the M3O.

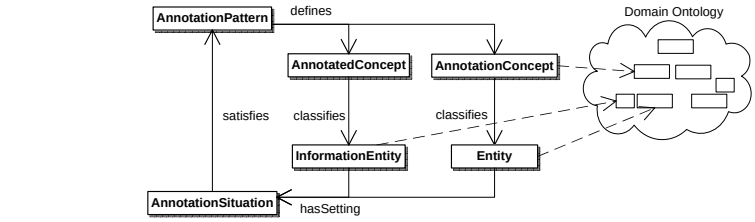
With ontologies, we can use abstract concepts and clearly identifiable individuals to represent data and to perform inferencing over the data. However, we also need the means to represent concrete data values such as strings or numerical values. In DUL there exists the concept Quality in order to represent intrinsic attributes of an Entity, i.e., attributes that only exist together with the Entity. Regions are used in order to represent the values of Qualities and the data space they come from. In DUL, there are different ways to encode concrete data values using Qualities and Regions. However, this discussion is beyond the scope of this paper. With the Data Value Pattern, we propose one of these options to be used in M3O. We argue that it is important to have a single, well defined way of representing concrete data values in order to reduce the risk of ambiguities. The Data Value Pattern (depicted in Figure 2c) assigns a concrete data value to an attribute of that entity. The attribute is represented by the concept Quality and is connected to the Entity by the hasQuality property. The Quality is connected to a Region by the hasRegion relation. The Region models the data space the value comes from. We attach the concrete value to the Region using the relation hasRegionDataValue. The data value is encoded using typed literals, i.e., the datatype can be specified using XML Schema Datatypes [8].

As an example, we like to represent the EXIF metadata brightnessValue representing the brightness of an image. We model Brightness as a quality since it is an attribute of the image. As BrightnessRegion, we use the real numbers and attach the concrete brightness value to the region using a specialization of hasRegionDataValue with a minInclusive and maxInclusive restriction on the interval $[-99.99, 99.99]$.

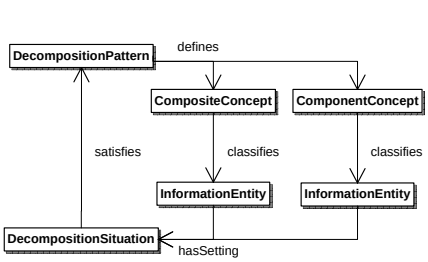
Using the hasPart relation, we can also express structured data values such as supported in MPEG-7. We can, e.g., represent the dominant color of an image as a Quality having a value from the Region RGBColorSpace. The RGBColorSpace has as part the individual color spaces, e.g., the RedColorSpace. Finally, we attach the color index to these subregions.

5.2 Annotation Pattern

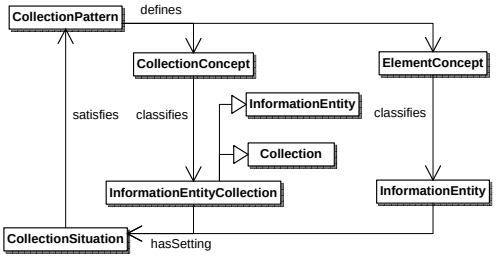
Annotations are understood in M3O as the attachment of metadata to an information entity (see Section 1). Thus, annotations are metadata of information objects and information realizations, respectively. As we have discussed in Section 3, metadata comes in various forms such as low-level descriptors obtained by automatic methods, non-visual information covering authorship and technical details, or semantic annotation aiming at a formal and machine-understandable representation of the contents. Our annotation pattern models basic structure that underlies all types of annotation. This allows for assigning arbitrary



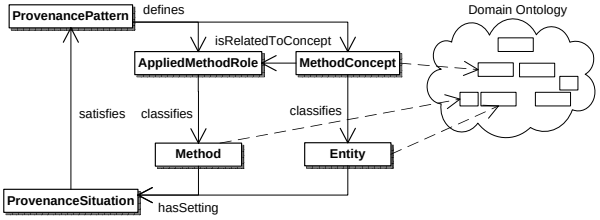
(a) Annotation Pattern



(b) Decomposition Pattern



(c) Collection Pattern



(d) Provenance Pattern

Figure 3: Patterns of the Multimedia Metadata Ontology

annotations to information entities, while providing the means for modeling provenance and context.

The Annotation Pattern depicted in Figure 3a is a specialization of the DnS pattern. It consists of an AnnotationSituation that is a specialization of the Situation concept and satisfies an AnnotationPattern, which is a specialization of the Description concept. The description defines at least one AnnotatedConcept that classifies each InformationEntity that is annotated by an instantiation of this pattern. Thus, the annotation pattern allows to annotate multiple individuals of InformationEntity within a single instantiation of the pattern. The InformationEntity has the AnnotationSituation as its setting. Each metadata item is represented by an Entity that is classified by an AnnotationConcept. A metadata item is an Entity that is used for describing some other Entity, i.e., it is data that describes some resource [23]. There has to be at least one AnnotationConcept defined by an AnnotationPattern.

Instances of Entity can be defined in some domain ontology as indicated in Figure 3a. For example, an annotation entity of an image can be the Creator of an image defined in some domain ontology (cf. example in Figure 8). Besides the Entity that is used to annotate an InformationEntity, also the role this Entity plays in the annotation and the InformationEntity itself can be defined in some domain ontology as Figure 3a depicts. For example, there might be a specialization of the Annotation Pattern that introduces concrete media types such as Image or Presentation that are of interest in a particular application (cf. Figure 5 and 8).

The annotations of an InformationEntity can be single individuals such as those defined in some external domain-specific ontologies. For example, the Creator of an image in the example shown in Figure 8 is Ferdinand Schmutzer represented by the individual F_Schmutzer. The annotation defined in the Annotation Pattern of Figure 3a can be in principle any arbitrary Entity. Thus, besides single individuals also complex annotations can used, possibly leveraging other core ontologies. For example, the image of the bombing of the city of Nagasaki can be annotated with a complex representation of the event of the atomic bombing (cf. the example in Figure 9) using the core ontology Event-Model-F [54]. The Event-Model-F provides ontology design patterns for representing the participation of living and non-living objects in events, the relation between events such as mereology, causality, and correlation, the interpretation of events and others.

The M3O makes no assumption about the source of the annotation and both manually and automatically created annotations are supported. We support both low-level and high-level annotations. Low-level annotations use the Data Value Pattern in order to represent metadata such as color histograms, while the high-level annotations reuse concepts and individuals from arbitrary domain ontologies. By the use of the DnS pattern, the annotation pattern already provides for a contextualized view on annotations. The relation between the annotated InformationEntity and the annotating metadata Entity only exists within the context expressed by the DnS pattern. There can be arbitrary instantiations of the Annotation Pattern used to describe a single InformationEntity. These different annotations may origin from different users. With the use of the DnS pattern, it is possible to differentiate the annotations from the different users and view only those that one is actually interested in. This provenance information of the annotation can be explicitly captured by the use of the Provenance Pattern described in Section 5.5.

Annotations can be subjective or objective, depending on the contextual point of view. Certain types of annotations, primarily those of technical nature like annotations using EXIF metadata, might be considered objective. Semantic annotations are rather subjective. Whether another person finds them appropriate depends on a variety of factors, such as context, trust, or whether the source is known or not. With our annotation pattern, we support the modeling of both subjective and objective annotations. The border between the kinds of annotation is rather vaguely defined. We support the interpretation of the trustworthiness of annotations and thus their objectivity by using the Provenance Pattern.

5.3 Decomposition Pattern

Our Decomposition Pattern models the decomposition of information entities, e.g., the decomposition of a SMIL presentation into its logical parts or the segmentation of an image. After a decomposition, there is a whole, called the composite, and there are the parts, called the components. We call this pattern Decomposition Pattern, as from a metadata point of view we decompose the (multi-)media into parts, which we want to annotate further.

The Decomposition Pattern shown in Figure 3b is a specialization of the DnS pattern. It introduces a DecompositionPattern as specialization of the Description concept and a DecompositionSituation as specialization of the Situation concept. The DecompositionPattern defines exactly one CompositeConcept and at least one ComponentConcept. The CompositeConcept classifies an InformationEntity, expressing that it is the whole. Each ComponentConcept classifies an InformationEntity, asserting that they are parts of the whole. All classified entities have the DecompositionSituation as setting. Like with annotations, the provenance of a decomposition can be added to the Decomposition Pattern by applying the Provenance Pattern from Section 5.5.

It is important to note that in cases of structured multimedia content there is already composition information available in the media itself. A SMIL file, e.g., contains information about how single media assets are arranged. However, with M3O we aim at representing metadata about parts of the media that are not necessarily equal to or included in the physical structure defined in the SMIL file (see Appendix A).

With the Decomposition Pattern, we provide means to attach metadata to the multimedia presentation and its constituents, i.e., the media assets. However, the Decomposition Pattern does not address the challenge of referring to parts of media assets such as clips and frames in videos via URIs in the web. This is the goal of the Media Fragments working group of the W3C, which developed a mechanism to address specific parts of videos by standardizing the way fragments are specified within an URI [58].

5.4 Collection Pattern

The Collection Pattern allows to represent an arbitrary collection of information entities that are semantically coherent, i.e., a set of information entities that share one or more common properties. For example in the scenario in Section 2, the journalist Marie initiates a media collection depicting scientists of potentially dangerous technologies. The Collection Pattern supports the collaborative creation of such collections. In addition, it is aware of the source of the information entities that are added to it.

The Collection Pattern depicted in Figure 3c introduces a CollectionPattern as specialization of the Description concept and a CollectionSituation specializing the Situation concept of the DnS pattern. The CollectionPattern defines exactly one CollectionConcept classifying an InformationEntityCollection. An InformationEntityCollection represents the actual collection and is a specialization of InformationEntity.

In addition, it is also a subconcept of Collection indicating that it is a collection of entities that share common properties. The InformationEntityCollection is a first-order entity. It might be annotated using the Annotation Pattern described in Section 5.2 or decomposed using the Decomposition Pattern as introduced in Section 5.3. In addition to the CollectionConcept, the CollectionPattern defines some ElementConcepts that classify InformationEntity, i.e., the elements of the collection. Finally, also provenance information can be added to a collecting using the Provenance Pattern described below. As such, it is possible to distinguish the origin of some InformationEntity that have been added to a collection. It allows Marie to choose media from experts and sources the trusts.

Collections and decompositions express different mereological relations between entities. A collection can be understood as a set of entities that share some common property. In contrast, a decomposition expresses a parthood relation among entities and their constituents. Therefore, collections and decompositions refer to different mereological relations between entities and thus should be modeled differently. Finally, the annotation of entities denotes a common description of these entities. Again, the annotation relationship is different from the mereological relations expressed by a collection and decomposition.

5.5 Provenance Pattern

An important aspect of the semantic description of rich, structured multimedia content is the provenance of the description, i.e., its source. With the Provenance Pattern, we allow for an explicit representation of provenance information along with a reason or justification of this information. The Provenance Pattern can be applied to describe the source of annotations, decompositions, as well as collections of information entities. Thus, it works in combination with the Annotation Pattern, Decomposition Pattern, and Collection Pattern described in Sections 5.2 to 5.4. Provenance information is optional, it is not always required or needed in a specific application context.

The Provenance Pattern is depicted in Figure 3d. It introduces a ProvenancePattern specialized from the Description concept and a ProvenanceSituation that is a subconcept of Situation. The ProvenancePattern defines one or more AppliedMethodRoles and optional MethodConcepts. The AppliedMethodRole classifies a Method that represents how an annotation, decomposition, or collection has been created. For example, it can be an algorithm for image segmentation or a manual annotation. The Provenance Pattern can define one or more AppliedMethodRoles in order to express multiple provenance information. The optional MethodConcepts defined by the ProvenancePattern provide further details of the applied Method. They classify Entity such as the parameters used when applying the Method. Thus, the MethodConcepts are related to the AppliedMethodRole, which is represented using the property isRelatedToConcept. Besides representing parameters used to run an automatic algorithm, the Entity can also point to a model used for some classifier, give a high level explanation in case of a manual annotation, or represent the creator of a collection. In case of concrete data values for the Entity parameters, the Data Value Pattern described in Sec-

tion 5.1 is used. Depending on the provenance of an annotation, decomposition, or collection, we might trust the information or not.

Like the Annotation Pattern, also the Provenance Pattern can integrate some domain ontology as indicated in Figure 3d. Instances of a Method can be defined in some domain ontology such as ManualAnnotation and AutomaticAnnotation (cf. examples in Figures 5 and 8). In addition, also the MethodConcept can be defined in some domain ontology like the AuthorRole indicating that a ManualAnnotation is provided by the author of a presentation such as john-1 (cf. examples in Figure 5). Finally, also the Entity indicating the parameters of a Method can be taken from some domain knowledge.

The provenance information modeled with the Provenance Pattern is applied to one, many, or all entities of the pattern the Provenance Pattern is attached to. To express to which entities the provenance information refers to, the relation isRelatedToConcept pointing from the AppliedMethodRole of the Provenance Pattern to, e.g., the AnnotationConcept of the Annotation Pattern is used. For example, if there are several AnnotationConcepts present in a single Annotation Pattern, which is valid, the isRelatedToConcept relation indicates to which of these AnnotationConcepts the provided provenance information refers to. Another example where provenance information is applied to the Collection Pattern is provided in Section 9.

Closely related to provenance is the notion of confidence in some information, i.e., the degree of certainty that a provided information is correct. It can be considered as some kind of specific provenance information. Confidence information can be provided by some algorithms and explicitly represented using the Provenance Pattern's Entity. It is the *output* parameters of the Method applied. For example, an image classifier potentially knows its performance in classifying images in a set of concepts it supports. Confidence information can be very specific such as a value in the interval of $[0, 1]$, where values close to 0 represent low confidence and a value nearby 1 indicate high confidence. However, it can also be only be vaguely specified such as a set of three confidence values like low, medium, and high. To represent confidence information, the Data Value Pattern is applied.

5.6 Comparison with Existing Metadata Models and Standards

A principle difference of our M3O approach with most of the existing metadata models and formats is that we use ontology design patterns and that these patterns base on the DnS pattern of DUL. The DnS pattern introduces roles that classify the entities one wants to make statements about. For example, in the Annotation Pattern the roles AnnotatedConcept and AnnotationConcept are introduced. By using the DnS pattern and roles, we allow to provide clear distinctions of different, contextual views onto the entities that are considered. For example, an entity such as john-1 can play the role of an AnnotationConcept in some contextual situation. He can also become an AnnotatedConcept in a different setting where john-1 is used to annotate an image that depicts John. Another

benefit of the M3O and its DnS-based approach is that it provides explicit points where less formal metadata such as domain ontologies and Linked Data can be integrated. These points are visualized in Figure 3a and Figure 3d. Regarding COMM, we can say that the DnS-based patterns of the M3O are simpler than the DnS-based patterns of COMM. The reason for this is that the M3O patterns have been alleviated from the influence of a single metadata format such as MPEG-7 in the case of COMM. Thus, the M3O patterns are more generic and lightweight with respect to the number of concepts and relations modeled for representing the different M3O features.

The structure of the M3O patterns and COMM patterns look similar as they all base on DnS. However, the functionality the patterns of the M3O provide is very different from each other and ranges from annotation, over decomposition, to provenance. In contrast, in COMM we find three patterns of similar structure that all deal with the same feature, namely annotating multimedia content. These are the Media Annotation Pattern, Content Annotation Pattern, and Semantic Annotation Pattern. The only difference is the particular kind of annotations that is supported. Whereas the Media Annotation Pattern allows to describe information realizations, the Content Annotation Pattern supports annotating information objects, and the Semantic Annotation Pattern provides for representing annotations of information objects with MPEG-7 descriptors. Thus, the patterns defined in COMM show partially overlapping functionality. As discussed in Section 3, COMM does not consistently provide a separation between information objects and information realizations. In the Semantic Annotation Pattern, MPEG-7 descriptors cannot be associated with information realizations. All descriptions, including format specific low-level features of the realizations, are attached to the information object. In addition, COMM has no support for collections of information entities and no explicit support for provenance information. These limitations have been removed with the M3O and the functionality significantly extended.

6 Fulfillment of Requirements

We have presented the ontology design patterns underlying our Multimedia Metadata Ontology M3O. We now compare the requirements discussed in Section 4 with the provided patterns to verify that our ontology supports all required features.

REQ-1 is supported by the use of Semantic Web technologies. In the Semantic Web resources are identified by URIs, which can be used as universal identifiers. Typically `http`-URIs are used. They are dereferencable and provide besides identification also an access mechanism. *REQ-2* is covered by the Information Realization Pattern, which models the distinction between the information object and it realizations. The annotation pattern addresses *REQ-3*. It provides the means to attach arbitrary metadata to both information objects and information realizations. Complex data values can be represented using the Data Value Pattern. The decomposition as formulated in *REQ-4* is provided by

the Decomposition Pattern. It provides decomposition on both levels. Support for organizing multimedia content in collections as formulated in *REQ-5* is provided by the Collection Pattern. It can be applied on information entities, i.e., information objects as well as information realizations. The *REQ-6* is satisfied by the Provenance Pattern. It provides a formalized means for representing the method how an annotation, decomposition, or collection has been created and the parameters applied with this method. Finally, the Provenance Pattern also supports *REQ-7* to represent confidence information. Being some specific kind of provenance information, it can be modeled as parameters of the method applied.

Besides the functional requirements, the M3O also provides a number of non-functional features [5, 54]. These non-functional features are rich axiomatization, partly due to the formal basis of DUL, modularity, extensibility, reusability, and separation of concerns. The M3O can be classified as a core ontology, which means that the ontology is modeled independently of a specific domain, but focused on an aspect orthogonal to many domains, namely media annotation. We clearly separate concerns by using small and reusable patterns such as annotation and separation of information objects and their realizations. An important aspect is the independence of specific domain ontologies. We can incorporate arbitrary domain ontologies or Linked Data sources such as DBpedia.

7 Experience with Defining the M3O and its Benefits

From the scenario in Section 2, we have derived the requirements to a multimedia metadata ontology described in Section 4. In addition, to define the patterns of the M3O, we have also conducted an extensive study of related work that is discussed in Section 3. Choosing the DnS pattern as basis for the M3O patterns is of fundamental nature as it has an impact to the design and structure of the entire core ontology. However, DnS has already been successfully used for the design of existing core ontologies like [54, 18, 44]. Thus, it has proofed to be a good modeling choice. Identifying and defining the Annotation Pattern and Decomposition Pattern of the M3O has been without complications. Both features are well understood and established in the existing metadata models and metadata formats.

In the course of designing the M3O, we have discussed whether the different features of our metadata model such as the Annotation Pattern and Decomposition Pattern are applicable to InformationObjects and InformationRealizations. Interestingly, these features make sense for both InformationObjects and InformationRealizations. For example, the InformationObject of an image could be annotated with its creator and concrete realizations of this image with the specific locations where they are stored in the web or the color depth and resolution that is used.

It is noteworthy that the Provenance Pattern has been initially integrated with the Annotation Pattern and Composition Pattern [46, 45]. This redundancy has been removed by introducing a distinct Provenance Pattern that can be attached to the Annotation Pattern and Composition Pattern. Regarding the Provenance Pattern, the question is how to attach provenance information to the patterns. We identified two principle solution approaches: The first one would attach the Provenance Pattern to the Situation concept of the patterns. This would allow to associate provenance information with the patterns without modifying the original patterns. The second option is to attach the Provenance Pattern to the M3O patterns like it is described in the article by extending the DnS-based patterns with respect to defining new roles and specializing existing roles. The difference in these two options is how the defines property of the M3O patterns can be axiomatized. In the second case, axiomatization of the defines property can be only very limited, i.e., it cannot be stated that there can only be AnnotedConcepts and AnnotationConcepts but it must be allowed to define further roles to add the provenance information. In the first case, axiomatization of the defines property can be much stricter as the provenance information is associated to the patterns through the Situation concept. However, this is not a good modeling solution as the Situation is not designed for such a purpose. Consequently, for the design of the M3O, we have decided in favor of the second option and piggybacking the Provenanve Pattern to the other M3O patterns by extending the DnS structure.

Finally, modeling collections is a requirement that is raised from existing multimedia management applications. However, it is hardly reflected by today's metadata models. Based on the experience with the previous patterns, identifying the scope and designing the Collection Pattern was conducted without problems. Subsequently, we recognized that provenance information could also be added to a collection. Thus, we also explicitly foresee the attachment of the Provenance Pattern to the Collection Pattern.

The M3O provides a model to annotate structured multimedia content. Although RDF is foreseen to conduct annotations in multimedia presentation formats such as SMIL, SVG, Flash, LASeR, XMT-Ω, and others, until now there is to the best of our knowledge no method that actually describes how to annotate the content. This is very unfortunate as retrieval of rich, structured multimedia presentations in the web is very hard. If multimedia authoring tools would be enabled to add more metadata to the presentations as it is possible with the M3O and as it is foreseen with today's formats, search of these presentations could be improved. Search engines like Yahoo! and Google already collect and use semantic metadata in formats such as RDFa [62] or Microformats [7] to improve visualization of their search results. First studies show that leveraging the structured metadata improves the click-through rate of web pages.[8] Rich semantic annotations in the M3O describing the multimedia presentations may also be collected by search engines and used for improving multimedia search.

[8]See: http://developer.yahoo.net/blog/archives/2008/07/ and http://priyankmohan.blogspot.com/2009/12/online-retail-how-best-buy-is-using.html

The M3O is not designed to replace existing multimedia metadata models, formats, and standards but rather aims at integrating them. This integration is supported by the formal nature of the M3O. We have developed a four-step, iterative process that allows ontology engineers to align existing multimedia metadata formats and metadata standards with the M3O [17]. Example mappings have been performed following this alignment process with the metadata models and metadata standards COMM, EXIF, and the W3C Ontology of Media Resource. In the first step, an in-depth analysis of the metadata format or metadata standard to be integrated with the M3O is conducted. This is an important prerequisite for successfully aligning the existing standard or format with the M3O. In the second step, structural information regarding the metadata model is analyzed in order to provide useful semantic groupings of the properties and concepts the model defines. In the third step, the concepts and properties are mapped with the M3O. Result of this step is a first integrated ontology that is validated and used as basis for further iterations. Finally, in the fourth step, the design decisions and adjustments made during the alignment process are documented. This mapping between the existing metadata models and the M3O has to be performed only once for each model. We assume that multimedia metadata standards do not change very often and if they are updated, the changes are typically incrementally. Thus, conducting a manual alignment of the existing metadata models with the M3O seems to be feasible in practice.

As the M3O targets at modeling multimedia metadata, we do not support the modeling of other fields such events or communication. Rather, the M3O is designed to be integrated with other core ontologies that support the modeling of events and communication. For example, the M3O can be smoothly integrated with our core ontology Event-Model-F [54] for a sophisticated representation of events and event relationships as the detailed example in Figure 9 shows.

Due to its formal nature, the M3O also allows to better integrate the canonical processes along the media production chain [22] such as premeditate, annotate, query, organize, publish, and distribute. Different metadata is to be exchanged between these processes. This metadata can be precisely defined by using the M3O. Thus, the M3O supports the idea and goal of the canonical processes for a better integration of the processes in the media production chain.

8 Axiomatization of the M3O

The Multimedia Metadata Ontology (M3O) has been created in the Web Ontology Language (OWL) [59]. It has been axiomatized using description logics [6]. By using the foundational ontology DOLCE+DnS Ultralight as modeling basis, the M3O inherits the rich axiomatization and formal precision of the foundational ontology. In addition, further axioms have been added to the core ontologies to precisely define the patterns defined in Section 5. Goal of such an axiomatization is to precisely specify the semantics of an ontology and its patterns, respectively. A rich axiomatization prevents the patterns to be applied

in a context in which they have not been designed for. However, a too restrictive axiomatization may also potentially limit the extensibility of the pattern towards new and useful requirements. Thus, the axiomatization of core ontologies like the M3O is a trade-off between semantic precision and protecting the patterns against wrongly applying them vs. providing support for extensibility and integration of future functionality.

Of particular interest is the axiomatization of ontology design patterns that base on the DnS pattern. The axiomatization of the Description of such DnS-based patterns defines some Concepts. These Concepts are the roles played by the entities that are included in the pattern. For example, the annotation pattern of the M3O introduced in Section 5.2 defines that there exists the roles AnnotatedConcept and AnnotationConcept as shown by the axiom below. The cardinality of the AnnotatedConcept and AnnotationConcept within the pattern is greater or equal one. In addition, further axioms are defined to precisely specify the semantics of the Annotation Pattern.

$$AnnotationPattern \sqsubseteq \exists defines.AnnotatedConcept \sqcap \exists defines.AnnotationConcept$$

It is important to note that the specification of the roles in the Annotation Pattern is not restricted to the two concepts above. Thus, in concrete instantiations of the Annotation Pattern further roles can be be defined and included. Such an open axiomatization of the roles in the ontology design pattern is required to extend the Annotation Pattern by further functionality. For example, one might want to add provenance information to the annotation, i.e., meta-information about the method and parameter setting by which the concrete annotations have been created. The remaining ontology design patterns of our M3O, namely the Decomposition Pattern, Collection Pattern, and Provenance Pattern are specified in the same way. They define the roles of the patterns and their cardinality. However, we do not restrict the set of roles that are allowed in the pattern.

The specific design of our ontology patterns is based on the generic DnS pattern. As discussed above, the axiomatization of our patterns allows us to use them solitary or in combination. For example, we can either use the Annotation Pattern, Decomposition Pattern, and Collection Pattern as they are defined in Sections 5.2 to 5.4 or combine them with the Provenance Pattern. This nesting of the Provenance Pattern with the other ontology design patterns of the M3O is possible as all our design patterns are based on the same foundational ontology DUL and specialize the DnS pattern. Thus, each pattern introduces their specific subconcepts of the DnS concepts Description and Situation. In addition, the patterns do not restrict the roles defined within the patterns. By this common basis and the ability to introduce new roles, the patterns are aligned and can be easily combined. A full axiomatization of the M3O can be found in Appendix B.

9 Modeling the Scenario with M3O

Having introduced the M3O, we now show its application to the scenario in Section 2. To this end, we briefly describe what happens when John creates and annotates his presentation and which patterns of the M3O are involved. While John annotates the multimedia content, the different features of the M3O are used such as applying the Decomposition Pattern and Provenance Pattern. For example, when John annotates the two parts of the presentation about the positive side and negative effects of nuclear energy (see (ii) in the scenario), the decomposition pattern is applied to define these two parts. The parts themselves again use the decomposition pattern to define which media assets they include. However, this is hidden from John by the multimedia authoring tool he uses. The authoring tool allows him to define the different sections of his presentation and add some annotations to it. When John publishes his presentation on the Web in the Flash format and SMIL format (see (v) in the example), the authoring tool applies in the background the Information Realization Pattern of DUL to describe that there are two realizations of the same information object. In his presentation, John uses images of a media collection about nuclear energy (see (vii) in the scenario). To describe this collection, the Collection Pattern of the M3O is applied. Finally, provenance information about the origin of the presentation is added by John (see (viii) in the scenario). A concrete multimedia authoring tool such as Adobe Flash [4] could obtain this information from the general authorship settings similar to other tools like Microsoft Word [41] and Adobe Acrobat [3]. The information about John's confidence on the correctness of the annotations such as the location of the New York picture is manually added by the author using M3O's Confidence Pattern (see (ix) in the scenario).

The following examples present the core aspects of our model, namely the information realization, decomposition, and annotation of multimedia content. The concrete objects are referred to as individuals, which is common in the context of ontologies and the Semantic Web. Each individual has a type that refers to a concept of some ontology. Within the diagrams we introduce in the following, each box represents an individual and its type. For example, the presentation-realization-1:SMILFile in Figure 4 refers to some individual presentation-realization-1 of type SMILFile. Both the ontology and the concrete annotations are represented using RDF. Concepts and individuals are identified by URIs. However, for easier presentation we will omit the namespace completely.

We start with an example of how to apply the Information Realization Pattern in order to represent the two basic levels of our model, i.e., the information object and the information realization. In this example, we consider two realizations of our presentation, namely one in SMIL and one in Flash. Therefore, we represent the fact that the presentation is realized by a SMIL file and also by a Flash file. In Figure 4, we can see that there is one individual presentation-1 of type Presentation, which is a subclass of InformationObject. The files are represented by the individuals presentation-realization-1 and presentation-realization-2, which realize the presentation. They are of type SMILFile and FlashFile, which

are subclasses of InformationRealization. Ideally, the full URI of the realization are derefereancable, i.e., a client can directly retrieve the respective realization.

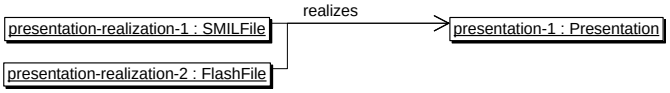


Figure 4: Application of the Information Realization Pattern to represent two realizations of a multimedia presentation

In the next step, we annotate the whole presentation with its topic, which is in this case represented by a Wikipedia article on the risk society using the Annotation Pattern. In Figure 5, the application of the Annotation Pattern is shown. The AnnotatedConcept classifies the individual presentation-1 and expresses that this is the information object being annotated. The AnnotationConcept classifies the individual Risk_Society from DBPedia, a Semantic Web representation of the Wikipedia article. We are not limited to using DBPedia, but can use any domain ontology and Linked Data source.

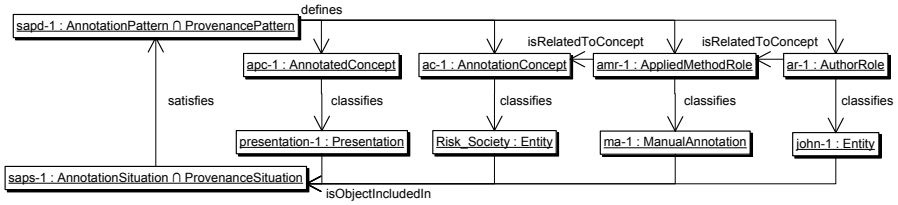


Figure 5: Semantic annotation of the whole presentation using the Annotation Pattern and Provenance Pattern

The pattern shows the benefit of using the DnS-based approach. We cannot only express the annotation as a relation between the information object and its annotation, but we can treat this relation within a context. We exemplify the support of our patterns for context and provenance by including information about the creator of the annotation. The AppliedMethodRole classifies a ManualAnnotation, and thus expresses that this presentation was labeled manually. We specify the author of this annotation by classifying some individual john-1 using the AuthorRole. The AuthorRole isRelatedToConcept the AppliedMethodRole, expressing that john-1 is the author of this manual annotation. Please note that the concepts such as ManualAnnotation and AuthorRole are subconcepts of the DOLCE+DnS Ultralight concepts Method and Entity and should be provided by some specialization of the M3O core patterns.

Subsequently, we present the decomposition of the presentation into logical components that we want to annotate further. In Figure 6, we show the logical decomposition of the presentation into two parts representing the positive and negative aspects of nuclear energy, respectively. We further demonstrate the

decomposition of the first part into the two images of Albert Einstein and the Times Square.

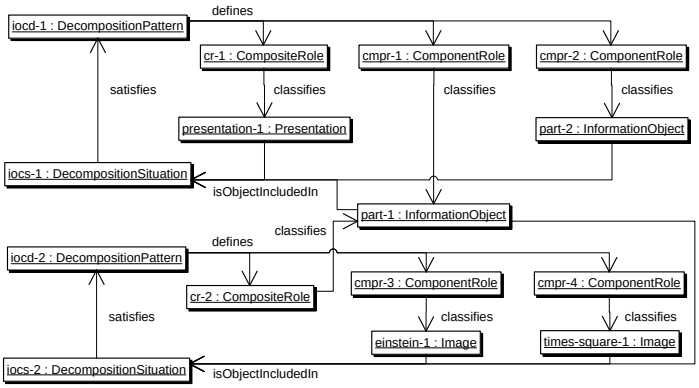


Figure 6: Two-layered decomposition of the presentation and its first part using the Decomposition Pattern

The upper part of Figure 6 shows the first composition, the lower half the second one. We see that the DecompositionPattern defines the CompositeRole and two ComponentRoles. The CompositeRole classifies the individual presentation-1, i.e., the information object representing our presentation. This relation represents the fact that the presentation is the Composite, i.e., the *whole* in this decomposition. The ComponentRoles classify the two InformationObjects named part-1 and part-2, representing the two logical *part*s of the presentation. The lower part of Figure 6 shows how part-1 is further decomposed into the two Images, represented by einstein-1 and times-square-1. The concept Image is a specialization of InformationObject representing images. We see that the individual part-1 plays the ComponentRole in the first composition and the CompositeRole in the second one. Being a component or a composite is therefore depending on the context. Thus, from a modeling point of view our M3O approach is advantageous as it considers properties such as being a component or a composite only within a specific context.

We demonstrate the annotation of an image with GPS-based location information from the EXIF [34] metadata standard as shown in Figure 7. EXIF allows to embed mainly technical metadata directly into images, e.g., in JPEG files. It is also part of the widely known and industry-driven Extensible Metadata Platform (XMP) [2] and used there for annotating images with GPS-based location information. The annotation is conducted by the EXIFAnnotationPattern, a specialization of the Annotation Pattern for EXIF. In addition, the Provenance Pattern is applied to provide the provenance of the annotation. EXIF metadata mainly focuses on the capturing conditions of the image. Thus, it might be tempting to attach all EXIF metadata to the information realization. But actually, EXIF provides many metadata that does not belong the realization. If we consider, e.g., the camera model, camera make, location, date, and time,

they clearly are properties of the information object. Even if we transform the original image into other formats, the camera model used to take the original image and the other properties do not change. Therefore, it is a immanent feature of the image that it was taken with a certain camera, and not of the concrete realization. Information such as the dimensions and color depth of the image is indeed a property of the realization and might change in different realizations. A standard like EXIF should therefore not be transformed into a M3O representation directly, but rather based on an analysis of the available features. There exist other ontologies that provide some vocabulary for representing EXIF metadata as RDF [35]. However, since they just perform a one-to-one mapping of EXIF tags to RDF properties and therefore add no additional semantics, we decided to not build on them, but use EXIF directly.

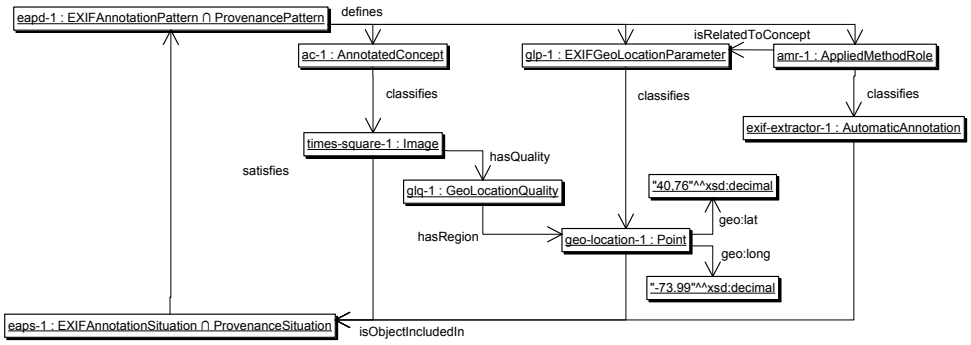


Figure 7: Using the Annotation Pattern and Provenance Pattern for representing geo-information extracted from the EXIF header

Figure 7 depicts the annotation of an Image information object with the location on which the image was taken. We attach this information to the information object, since the location is independent of the image realization and is therefore a property of the information object. In order to represent the coordinates, we employ the Data Value Pattern and introduce the GeoLocationQuality. In addition, the AnnotationConcept is specialized to EXIFGeoLocationParameter that parametrizes a Point from the WGS84 vocabulary [61]. The concept Point is used here as Region and represents the data space of all geo coordinates. We use the WGS84 properties geo:lat and geo:long to represent the latitude and longitude. Please note that the Region, the Quality, and the WGS84 relations are not specific to the EXIF descriptor. They could be reused in other annotations that represent geolocations. The provenance of the location information is the exif-extractor-1, a common EXIF extractor tool. Thus, it is an AutomaticAnnotation that is classified by a AppliedMethodRole.

An alternative way of modeling the geolocation would be introducing subregions for latitude and longitude to the region Point using the hasPart relation of DUL. These sub-regions LatitudeRegion and LongitudeRegion would then each have a hasRegionDataValue for the latitude and longitude information. However,

it is questionable that introducing two sub-regions for modeling the latitude and longitude of a geo point is of any added value for the knowledge representation. Thus, a trade-off is needed to decide how deep one wants to model the knowledge while still allowing an efficient processing of the data. In the example, it seems reasonable to integrate the WGS84 domain ontology rather than introducing further sub-regions. In general, we recommend to integrate domain knowledge where possible and keeping a good trade-off between semantic precision and efficient processing of the knowledge representation.

The annotation of concrete realizations of the information object times-square-1 is very similar to the annotation shown in Figure 7. Instead of annotating the information object times-square-1, the realizations of this image information object, namely individuals of ImageRealizations are annotated. Metadata that is typically associated with these realizations are the image resolution and the location where the file is stored. A figure depicting the annotation of image realizations is omitted for reasons of brevity.

Whenever the Annotation Pattern shall provide support for some specific annotation entities like the location an image has been captured, a specialization of the Annotation Pattern is created as Figure 7 shows. To this end, subconcepts of the AnnotationConcept role are created such as the EXIFGeoLocationParameter. Based on this specialization, a concrete information system will know how to interpret the concrete Entity used for annotation.

The example depicted in Figure 8 shows the annotation of the image of Einstein einstein-1 with information about its Creator, namely Ferdinand Schmutzer represented by the individual F_Schmutzer. This annotation is represented using the Annotation Pattern. The provenance is a ManualAnnotation conducted by john-1, which is modeled using the Provenance Pattern. As the information is provided by Wikipedia, John states a high confidence degree to his annotation. Thus, the ManualAnnotation has a ConfidenceQuality that uses as ConfidenceRegion the interval of $[0, 1]$. In the example, the confidence value of 0.95. The confidence is modeled using the XSD datatype float and applying the minInclusive and maxInclusive restrictions on it.

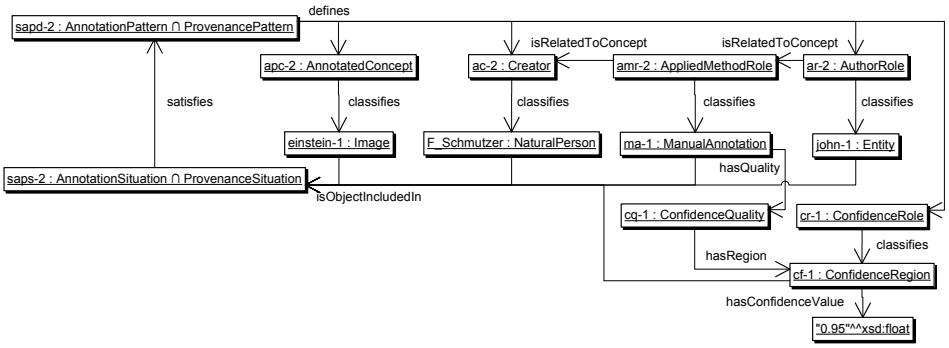


Figure 8: Using the Annotation Pattern and Provenance Pattern for annotating the image of Einstein with the image's creator and high confidence by John

In addition to the annotation of an image such as einstein-1 with an individual from Wikipedia, the Annotation Pattern can also be used to annotate an Image (or other information entity) with some complex knowledge. The example depicted in Figure 9 demonstrates the annotation of the image nagasaki-cloud-1 with some complex annotations. The image nagasaki-cloud-1 shows the atomic cloud of the bombing of the city of Nagasaki. It is annotated with a complex representation of the event of that bombing using the Participation Pattern of the core ontology Event-Model-F [54]. The Participation Pattern captures the living and non-living objects participating in an event. It also allows to represent the time and location when this event happened (not shown in the figure for reasons of brevity). As the Figure 9 shows, the Participation Pattern of the Event-Model-F is connected with the Annotation Pattern of the M3O through the EventParticipationSituation. The Participation Pattern defines the event nagasaki-bombing-1 and the different objects that have participated in the event. These are sweeney-1, the 393rd Squadron commander Major Charles W. Sweeney who was pilot of the B-29 airplane superfortress-bockscar-1 that dropped the bomb called fat-man-1 onto the city of Nagasaki on August 9, 1945.

Please note that the combination of the Annotation Pattern and the Participation Pattern does not ensure that the image actually shows the event. It rather denotes that the image is annotated with some information about an event. The image could have been taken, e.g., before or after the event, or even be unrelated to the event at all. The exact relation can be specified by specializing the AnnotationRole. For example, there can be an AboutRole which is instantiated with "science", or a SpecificOfRole that would be "Albert Einstein", and a GenericOfRole that is "person".

Please note that the InformationObject einstein-1 is saying that there is a particular image taken of Einstein that has been created by Schmutzer. This InformationObject can be provided in different InformationRealizations such as PNG and JPG. These InformationRealizations can even be created by different people. However, independent of the creators of the concrete realizations, the creator of the InformationObject remains the same. By applying the Information Realization Pattern, we are even able to explicitly annotate the different creators of the InformationObject and its InformationRealizations.

Finally, the example depicted in Figure 10 demonstrates the use of the Collection Pattern and Provenance Pattern. The journalist Marie has initiated an ImageCollection about scientists that are or have been involved in the development of potentially dangerous technology. The collection-1 has several elements of which two of them are shown in the figure. One is the image of Wernher von Braun von-braun-1 provided by Marie and the other one is einstein-1, an image of Einstein added by John.

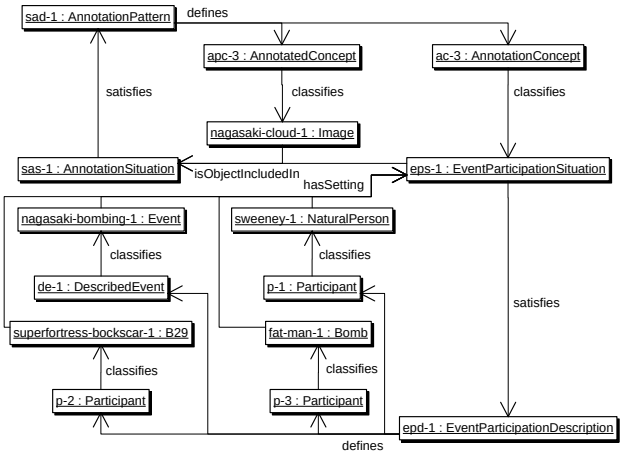


Figure 9: Using the Annotation Pattern of M3O and the Participation Pattern of the Event-Model-F to annotate the image showing the atomic cloud over Nagasaki with the event of the atomic bombing and its participants

10 Embedding M3O in Multimedia Presentations

In this section, we demonstrate how the metadata represented in M3O is embedded into rich, structured multimedia presentations. The M3O ontology is represented in OWL [59]. As such, the patterns of the ontology can be represented in RDF, which can be serialized in different formats. The best known is probably RDF/XML, which is also recommended by the W3C. This allows us to directly embed RDF metadata within formats such as SMIL and SVG, which already provide appropriate means for embedding XML-based metadata. However, in general, the representation of M3O-based metadata is independent of a specific serialization format like RDF/XML.

With the Metainformation Module, SMIL explicitly forsees the integration of XML-based metdata to describe the SMIL presentation [63]. The XML-serialized M3O is embedded into the SMIL presentation's <header> by using the <metadata>-tag (cf. lines 2-4 of Listing 1). The embedded RDF has to represent both the information object and the information realization levels. As the example in Listing 1 shows, the embedded RDF uses the xml:base attribute to set the base URI of the RDF part to http://example.com/john/nuclear representing the information object level (cf. line 6). The SMIL file itself has the URI http://example.com/john/nuclear.smil. This URI also denotes the location from which the file can be retrieved. Within the RDF part, we use abbreviated URIs, such as #presentation-1 (eg. in line 11). Using the xml:base this is concatenated to the full URI http://example.com/john/nuclear#presentation-1. On the information realization level, we address parts of the SMIL file using the URI of the file and

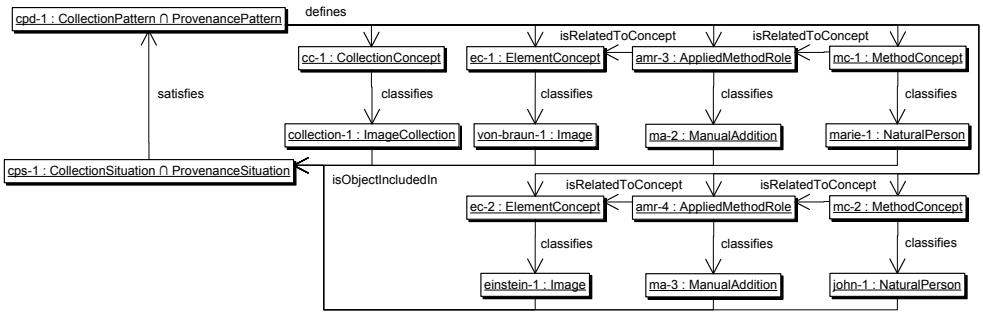


Figure 10: Using the collection pattern for representing an image collection of scientists

adding the value of the respective xml:id attribute using a hash sign (cf. line 17 for a RDF snippet referencing a SMIL element and line 29 for its definition). The URI http://example.com/john/nuclear.smil#scientificAchievements identifies the first part of the SMIL document with the xml:id scientificAchievements.

Listing 1: Embedding M3O into SMIL as RDF/XML

```
<smil xmlns="http://www.w3.org/2006/SMIL30/...">
<head>
<!-- Metadata -->
<metadata id="meta-rdf">
<rdf:RDF
   xml:base="http://example.com/john/nuclear"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:dul="http://www.loa-cnr.it/ontologies/DUL.owl#"
   xmlns:m3odec="http://m3o.semantic-multimedia.org/ontology/
       decomposition.owl#"
   xmlns:m3oann="http://m3o.semantic-multimedia.org/ontology/annotation.
       owl#"
   xmlns:m3osmil="http://m3o.semantic-multimedia.org/ontology/smil.owl#">

<dul:InformationObject rdf:about="#presentation-1">
  <dul:isObjectIncludedIn rdf:resource="#ds-1"/>
</dul:InformationObject>
<m3osmil:SMILFile rdf:about="http://example.com/john/nuclear.smil">
  <dul:realizes rdf:resource="#presentation-1"/>
</m3osmil:SMILFile>
<m3osmil:SMILElement rdf:about="http://example.com/john/nuclear.smil#
    scientificAchievements">
  <dul:realizes rdf:resource="#part-1"/>
</m3osmil:SMILElement>
<!-- more, e.g. #part-1, #image-1, ... -->
</rdf:RDF>
</metadata>
<!-- layout -->
</head>
<body>
  <!-- Presentation content -->
  <seq id="mainPresentation">
    <par xml:id="scientificAchievements">...</par>
    <par xml:id="scientificRisks">...</par>
  </seq>
</body>
</smil>
```

Embedding the M3O metadata into other presentation formats like SVG works in principle similar to the integration into SMIL. SVG also provides a <metadata>-tag that can be used to embed XML-serialized RDF in the SVG-header. As with SMIL, the individual parts of the SVG presentation such as the media assets can be annotated. This approach of embedding M3O annotations can also be applied on emerging multimedia formats such as LASeR and XMT-Ω. LASeR is based on SVG and reuses its <metadata> element for describing and embedding RDF-based multimedia metadata. XMT-Ω is a high level description of the binary MPEG-4 format and is based on SMIL. It uses SMIL's metadata model to represent metadata in RDF. Finally, also the frame-based and binary presentation format Flash allows for integrating metadata with the presentation [51]. Flash is in principle also tag-based like SMIL and SVG [50]. Examples of tags in Flash are adding and removing an image within a frame, starting and stopping a sound, and others. Thus we can use the M3O to annotate the frames and tags in Flash.

Please note that in the M3O, we do neither model the temporal structure nor the spatial structure of the media in the multimedia presentations. However, we provide the means to refer to parts in the presentation and media assets used. In the case of SMIL, the metadata described in the M3O is connected with the content elements of the annotated presentation via the xml:id of the tags used like <image> and <video> or <par> and <seq>. Thus, the metadata description is independent from the timing aspects and interactivity of the presentation. In other formats such as Flash, this connection is established by the frame number within the presentation. Further examples demonstrating the serialization of M3O in RDF/XML are shown in Appendix A of this document.

11 Application Programming Interface and Tool Support for M3O

Our principle idea is that application developers who want to use the M3O will not need to write SPARQL [64] queries in order to use the M3O knowledge base. Rather, we aim at providing an easy to use application programming interface (API) that completely hides the complexity and formal nature of the M3O from the application developers. To this end, we have developed a model driven engineering (MDE) tool for API generation that uses the ontology definition to (semi-)automatically generate an API in programming languages such as Java for it. A detailed description of how APIs for OWL-based ontologies such as the M3O are generated has been documented in a technical report [49]. Thus, in principle application developers do not need to understand the M3O in order to use it, to create annotations, or search for annotations. However, it is of course possible to write one's own SPARQL queries against the knowledge base.

Besides generating APIs for ontologies like the M3O, we have also developed an API to the M3O by hand. The rationale behind this is to be able to compare the efforts needed for (semi-)automatically generating the API code with the

handcrafted implementation. The handcrafted M3O API has been integrated with the SemanticMM4U framework for the multi-channel generation of multimedia presentations. The API allows for adding rich semantic descriptions of multimedia content into arbitrary multimedia applications. It has been developed using Winter, a flexible approach for mapping arbitrary RDF patterns to Java objects [48]. Winter allows to annotate Java classes with SPARQL queries and automates the construction of Java objects from the retrieved data. The SemanticMM4U framework provides a generic multimedia tool support for the multi-channel generation of semantically-rich multimedia presentations in formats like SMIL, SVG, Flash, and others [52, 53]. The framework has been successfully applied for the development and generation of rich, structured multimedia presentations in application domains such as personalized sports news, context-aware tourist guides, and the generation and semantic enrichment of personal photo albums [12]. The framework uses existing multimedia metadata and allows to derive new semantics while the multimedia presentations are created. Although the SemanticMM4U allows for the multi-channel generation of semantically-rich multimedia presentations, a proper model that describes how the generated presentations shall be annotated has still been desperately missing. This gap of a generic model and a reference framework for semantic annotation of rich, structured multimedia presentations is now being filled by the M3O. The integration of the M3O API into the SemanticMM4U framework is transparent to the applications that use the framework. The concrete applications automatically benefit from the use of the M3O to semantically describe the generated multimedia presentations without the need of any modifications of the applications.

In the next step, we plan to evaluate an automatically generated API for the M3O and the handcrafted M3O API with respect to development effort and time. In addition, we plan to apply the MDE-driven API generation process on specializations of the M3O such as the examples provided in Section 9.

12 Conclusions and Future Work

In this paper, we have presented the Multimedia Metadata Ontology (M3O) as a generic modeling framework for rich, structured multimedia presentations. Unlike existing metadata models, the M3O is not bound to a specific media type and allows for integrating the features of the different models and standards we find today. The M3O strictly separates information objects from their realizations, a requirement that is often requested for multimedia annotation [33, 25]. The model supports annotation and decomposition of the multimedia presentations on both levels, the information objects and information realizations. It supports both the representation of high-level semantic annotation with background knowledge as well as the annotation with low-level features extracted from the multimedia content. In addition, it allows to capture and represent provenance information about the annotations and decompositions.

It is important to note that with the M3O, we do not propose yet another model on multimedia metadata. Rather, we aim at providing a general modeling framework for multimedia metadata that comprises the features of today's metadata models and metadata standards. Due to its formal nature and pattern-based approach it is well suited for this task and provides the basis needed to host and integrate the different existing metadata approaches. The M3O is available in OWL at http://west.uni-koblenz.de/m3o and is formalized in description logics [6]. It can be used via a Java-based API and has been integrated in our SemanticMM4U framework for the multi-channel generation of semantically-rich multimedia presentations [52, 53] in multimedia formats such as SMIL, SVG, and Flash.

The M3O is part of the portfolio of key technologies used in the start-up company Kreuzverweis (http://www.kreuzverweis.com). Kreuzverweis aims at developing a next generation media management solution based on semantic technologies and is funded by the EXIST Research Transfer Programme of the German government.

Acknowledgement We kindly thank our students Stefan Scheglmann, Daniel Eißing, and Viktor Wart for implementing the M3O and its application programming interface.

References

[1] Adobe Systems, Inc. Flash file format, July 2008. http://www.adobe.com/licensing/developer/.

[2] Adobe Systems, Inc. XMP Specifications, 2008. http://www.adobe.com/devnet/xmp/.

[3] Adobe Systems, Inc. Adobe Acrobat, 2010. http://www.adobe.com/products/acrobat/.

[4] Adobe Systems, Inc. Adobe Flash Professional CS5, 2010. http://www.adobe.com/products/flash/.

[5] R. Arndt, R. Troncy, S. Staab, L. Hardman, and M. Vacura. COMM: designing a well-founded multimedia ontology for the web. In *ISWC+ASWC*, pages 30–43, 2007.

[6] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, 2003. ISBN 0-521-78176-0.

[7] F. Berriman, D. Cederholm, T. Çelik, R. Khare, R. King, K. Marks, and B. Ward. microformats, 2010. http://microformats.org/.

[8] P. V. Biron and A. Malhotra. XML Schema Part 2: Datatypes Second Edition, W3C Recommendation. October 2004. http://www.w3.org/TR/xmlschema-2/.

[9] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann. DBpedia - a crystallization point for the Web of Data. *J. Web Sem.*, 7(3):154–165, 2009.

[10] S. Bloehdorn, K. Petridis, C. Saathoff, N. Simou, V. Tzouvaras, Y. Avrithis, S. Handschuh, Y. Kompatsiaris, S. Staab, and M. G. Strintzis. Semantic annotation of images and videos for multimedia analysis. In *European Semantic Web Conference*, pages 592–607. Springer, May 2005.

[11] S. Boll, T. Bürger, O. Celma, C. Halaschek-Wiener, E. Mannens, and R. Troncy. Multimedia Vocabularies on the Semantic Web. W3C Multimedia Semantics Incubator Group Report (XGR), July 2007. http://www.w3.org/2005/Incubator/mmsem/XGR-vocabularies/.

[12] S. Boll, P. Sandhaus, A. Scherp, and U. Westermann. Semantics, content, and structure of many for the creation of personal photo albums. In *ACM Multimedia*, pages 641–650, 2007.

[13] S. Borgo and C. Masolo. *Handbook on Ontologies*, chapter Foundational choices in DOLCE. Springer, 2nd edition, 2009.

[14] D. Brickley and L. Miller. The Friend Of A Friend (FOAF) vocabulary specification, November 2007. http://xmlns.com/foaf/spec/.

[15] S. Dasiopoulou, V. Tzouvaras, I. Kompatsiaris, and M. G. Strintzis. Enquiring MPEG-7 based multimedia ontologies. page 331–370, Oct. 2009.

[16] Dublin Core Metadata Initiative. DCMI Metadata Terms, Jan. 2008. http://dublincore.org/documents/dcmi-terms/.

[17] D. Eißing, A. Scherp, and C. Saathoff. Integration of existing multimedia metadata formats and metadata standards in the M3O. In *Int. Conf. on Semantic and Digital Media Technologies; Saarbrücken, Germany*. Springer, 2010.

[18] T. Franz, S. Staab, and R. Arndt. The X-COSIM integration framework for a seamless semantic desktop. In *Knowledge capture*, pages 143–150. ACM, 2007. ISBN 978-1-59593-643-1.

[19] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design patterns: elements of reusable object-oriented software*. Programmer's Choice. Addison-Wesley, July 2004.

[20] A. Gangemi and V. Presutti. *Handbook on Ontologies*, chapter Ontology Design Patterns. Springer, 2009.

[21] R. Garcia and O. Celma. Semantic integration and retrieval of multimedia metadata. In *Knowledge Markup and Semantic Annotation Workshop*, Galway, Ireland, 2005.

[22] L. Hardman, Z. Obrenovic, F. Nack, B. Kerhervé, and K. W. Piersol. Canonical processes of semantically annotated media production. *Multimedia Syst.*, 14(6):327–340, 2008.

[23] B. Haslhofer and W. Klas. A survey of techniques for achieving metadata interoperability. *ACM Comput. Surv.*, 42(2):1–37, 2010. ISSN 0360-0300.

[24] L. Hollink, G. Nguyen, G. Schreiber, J. Wielemaker, B. Wielinga, and M. Worring. Adding spatial semantics to image annotations. In *Knowledge Markup and Semantic Annotation*, 2004.

[25] L. Hollink, A. T. Schreiber, B. J. Wielinga, and M. Worring. Classification of user image descriptions. *International Journal of Human-Computer Studies*, 61(5):601 – 626, November 2004.

[26] L. Hollink, G. Schreiber, and B. Wielinga. Patterns of semantic relations to improve image content search. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(3):195–203, 2007.

[27] J. Hunter. Enhancing the semantic interoperability of multimedia through a core ontology. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(1):49–58, January 2003.

[28] J. Hunter. *Multimedia Content and the Semantic Web: Standards, Methods and Tools*, chapter Adding Multimedia to the Semantic Web: Building an MPEG-7 ontology, pages 261–283. John Wiley & Sons, 2005. ISBN 0470857536.

[29] Int. Federation of Library Associations and Institutions. Functional requirements for bibliographic records. Technical report, IFLA, 2009. http://www.ifla.org/publications/functional-requirements-for-bibliographic-records.

[30] International Press Telecommunications Council. "IPTC Core" Schema for XMP Version 1.0 Specification document, 2005. http://www.iptc.org/.

[31] A. Isaac and R. Troncy. Designing and using an audio-visual description core ontology. In *Workshop on Core Ontologies in Ontology Engineering, CEUR Proceedings Vol. 118*, Whittlebury Hall, Northamptonshire, UK, October 2004.

[32] ISO/IEC. Information technology — Coding of audio-visual objects — Part 20: Lightweight Application Scene Representation (LASeR) and Simple Aggregation Format (SAF), Apr. 2007. http://www.mpeg-laser.org/documents/DRAFT_LASER_2ND_ED.pdf.

[33] A. Jaimes and S.-F. Chang. A conceptual framework for indexing visual information at multiple levels. In *IS&T/SPIE Internet Imaging*, volume 3964, 2000.

[34] JEITA. Exchangeable image file format for digital still cameras, April 2002. http://www.jeita.or.jp/.

[35] M. Kanzaki. Exif data description vocabulary, 2003. http://www.kanzaki.com/ns/exif, last update in 2007.

[36] M. Kim, S. Wood, and L.-T. Cheok. Extensible MPEG-4 textual format (XMT). In *ACM workshops on Multimedia*, pages 71–74, New York, NY, USA, 2000. ACM. ISBN 1-58113-311-1.

[37] S. S. Layne. Some issues in the indexing of images. *Journal of the American Society for Information Science*, 45(8):583–588, 1994.

[38] W. Lee, T. Bürger, F. Sasaki, V. Malaisé, F. Stegmaier, and J. Söderberg. Ontology for Media Resource 1.0. W3C Media Annotations Working Group, Working Draft, June 2009. http://www.w3.org/TR/2009/WD-mediaont-10-20090618/.

[39] M. Markkula and E. Sormunen. End-user searching challenges indexing practices in the digital newspaper photo archive. *Information Retrieval*, 1 (4):259–285, January 2000.

[40] Merriam-Webster, Inc. Metadata, 2009. http://www.m-w.com/dictionary/metadata.

[41] Microsoft Corporation. Office, 2010. http://office.microsoft.com/.

[42] MPEG-7. Multimedia content description interface. Technical report, Standard No. ISO/IEC n15938, 2001.

[43] M. Nilsson and M. Mutschler. ID3, 2009. http://www.id3.org/.

[44] D. Oberle. *Semantic Management of Middleware*. Springer, 2006.

[45] C. Saathoff and A. Scherp. M3O: The multimedia metadata ontology. In *Workshop on Semantic Multimedia Database Technologies; Graz, Austria*, 2009.

[46] C. Saathoff and A. Scherp. Unlocking the semantics of multimedia presentations in the web with the multimedia metadata ontology. In *World Wide Web; Raleigh, North Carolina, USA*, pages 831–840. ACM, 2010.

[47] C. Saathoff, M. Grzegorzek, and S. Staab. Labelling image regions using wavelet features and spatial prototypes. In *Int. Conf. on Semantic and Digital Media Technologies; Koblenz, Germany*, volume 5392 of *Lecture Notes in Computer Science*, pages 89–104. Springer, 2008.

[48] C. Saathoff, S. Scheglmann, and S. Schenk. Winter : Mapping RDF to POJOs revisited. In *Poster & Demo Session, ESWC 2009, Heraklion, Greece, May 31 - June 3*, Heraklion, Greece, 2009.

[49] S. Scheglmann, A. Scherp, and S. Staab. Model-driven generation of apis for owl-based ontologies. Technical report, Fachbereich Informatik, Universität Koblenz-Landau, 2010. http://www.uni-koblenz.de/~fb4reports/2010/2010_07_Arbeitsberichte.pdf.

[50] A. Scherp. *A Component Framework for Personalized Multimedia Applications*. OlWIR, Oldenburg, Germany, Feb. 2007. ISBN 978-3-939704-11-9. PhD Thesis, Available from http://ansgarscherp.net/dissertation/.

[51] A. Scherp. Semantics support for personalized multimedia content. In *Internet and Multimedia Systems and Applications*, pages 57–65. IASTED, Mar. 2008.

[52] A. Scherp. Canonical processes for creating personalized semantically rich multimedia presentations. *Multimedia Syst.*, 14(6):415–425, 2008.

[53] A. Scherp and R. Jain. An ecosystem for semantics. *IEEE MultiMedia*, 16 (2):18–25, 2009.

[54] A. Scherp, T. Franz, C. Saathoff, and S. Staab. F—A Model of Events based on the Foundational Ontology DOLCE+ Ultralight. In *Knowledge Capturing*, pages 137–144, 9 2009.

[55] G. Schreiber, I. Blok, D. Carlier, W. van Gent, J. Hokstam, and U. Roos. A mini-experiment in semantic annotation. pages 404–408, 2002.

[56] R. Steinmetz and K. Nahrstedt. *Multimedia Systems*. Springer, 2004.

[57] C. Tsinaraki, P. Polydoros, and S. Christodoulakis. Integration of OWL ontologies in MPEG-7 and TVAnytime compliant semantic indexing. *Advanced Information Systems Engineering*, pages 398–413, 2004.

[58] D. Van Deursen, R. Troncy, E. Mannens, S. Pfeiffer, Y. Lafon, and R. Van de Walle. Implementing the media fragments URI specification. In *International conference on World wide web; Raleigh, North Carolina, USA*, pages 1361–1364. ACM, 2010. ISBN 978-1-60558-799-8.

[59] W3C. OWL web ontology language overview, January 2004. http://www.w3.org/TR/owl-features/.

[60] W3C. RDF Primer, Feb. 2004. http://www.w3.org/TR/REC-rdf-syntax/.

[61] W3C. Basic geo (WGS84 lat/long) vocabulary, 2006. http://www.w3.org/2003/01/geo/.

[62] W3C. RDFa Primer, Oct. 2008. http://www.w3.org/TR/xhtml-rdfa-primer/.

[63] W3C. SMIL 3.0, Dec. 2008. http://www.w3.org/TR/SMIL/.

[64] W3C. SPARQL query language for RDF, Jan. 2008. http://www.w3.org/TR/rdf-sparql-query/.

[65] W3C. SVG, Apr. 2009. http://www.w3.org/TR/SVG/.

[66] Yahoo! SearchMonkey Media, 2009. http://developer.yahoo.com/searchmonkey/smguide/searchmonkey-media.html.

Appendix

A Further Examples showing the Representation of M3O in RDF/XML

This section provides further examples of how metadata modeled in M3O can be represented in RDF. The examples show an XML-serialization of the RDF descriptions and thus can be directly embedded into formats such as SMIL, SVG, LASeR, and XMT-Ω.

The decomposition pattern is shown in Listing 2. We see again the abbreviated URIs for the whole presentation #presentation-1 in line 4 as well as the URIs of the two parts in lines 9 and 14. This decomposition is on the information object level. The information objects are linked to the realizations, i.e., the SMIL file itself and the two elements of the SMIL presentation using the information relaization pattern. This is shown in Listing 1 in lines 14 and 18.

Listing 2: Embedding M3O Decomposition into SMIL

```
<m3odec:DecompositionPattern rdf:about="#iocd-1">
  <dul:defines>
    <m3odec:CompositeRole rdf:about="#cr-1">
      <dul:classifies rdf:resource="#presentation-1" />
    </m3odec:CompositeRole>
  </dul:defines>
  <dul:defines>
    <m3odec:ComponentRole rdf:about="#cmpr-1">
      <dul:classifies rdf:resource="#part-1" />
    </m3odec:ComponentRole>
  </dul:defines>
  <dul:defines>
    <m3odec:ComponentRole rdf:about="#cmpr-2">
      <dul:classifies rdf:resource="#part-2" />
    </m3odec:ComponentRole>
  </dul:defines>
</m3odec:DecompositionPattern>
<dul:InformationObject rdf:about="#part-1">
  <dul:isObjectIncludedIn rdf:resource="#ds-1" />
</dul:InformationObject>
<dul:InformationObject rdf:about="#part-2">
  <dul:isObjectIncludedIn rdf:resource="#ds-1" />
</dul:InformationObject>
<m3odec:DecompositionSituation rdf:about="#ds-1">
  <dul:satisfies rdf:resource="#dp-1" />
</m3odec:DecompositionSituation>
```

The M3O's Decomposition Pattern is not concerned with the actual organization of the media assets in time, space, and interaction. This is the task of the multimedia presentation format used such as SMIL and Flash. As such, the Decomposition Pattern only separates the different logical units of the presentation and gives them an identifier in order to conduct annotations, further decompositions, and others on it. To illustrate this, consider the two presentation parts part-1 and part-2. Whether part-1 is rendered before part-2 or vice versa, or the spatial layout of the images shown in part-2 are changed, or a mixture of both does not matter for the annotation of the presentation parts and images using the Annotation Pattern. Thus, the organization of the media assets in time

and space in a coherent manner is the responsibility of the presentation format like SMIL, whereas the M3O is in charge of describing the logical units of the presentations for the purpose of annotations. Of course, parts of the logical arrangements of the media content can be conducted in SMIL itself using, e.g., the <par> and <seq> tags. However, this is not necessarily the case and from a viewers perspective the same multimedia presentation can be realized using different arrangements of the <par> and <seq> tags or using different tags like the <img> tag and the begin and end attributes. However, such spatial and temporal relations can be introduced into the Annotation Pattern by specializing it. For example, the spatial relation can be used to represent that an image placed above a text is depicting a person and refers to a curriculum vitae of that person, namely the text below. Such relations on the annotations can be used, e.g., to conduct spatial reasoning [47].

In Listing 3, we see how to represent the fact that image-1 is realized by some file from Wikipedia. Please note that within the SMIL body even a local copy might be used. But from a metadata perspective it might be more appropriate to link to the original version on the web. Even both could be included.

Listing 3: Referring to Media Assets with M3O in SMIL.

```
<m3osmil:JPEGFile rdf:about="http://en.wikipedia.org/wiki/
    File:Einstein1921_by_F_Schmutzer_4.jpg">
  <dul:realizes rdf:resource="#image-1" />
</m3osmil:JPEGFile>
```

This example demonstrates the annotation of the first part with the individual Nuclear_power from DBPedia in Listing 4. As discussed, our modelling approach allows for the use of arbitrary background knowledge (cf. [54]).

Listing 4: Semantic Annotations for the First Part of the Presentation into SMIL

```
<m3odec:AnnotationConcept rdf:about="#cmpr-1">
  <dul:classifies rdf:resource="http://dbpedia.org/resource/
      Nuclear_power"/>
</m3odec:AnnotationConcept>
```

As final example, Listing 5 demonstrates embedding the semantic annotation of the image of Einstein with the creator of the image as shown in Figure 8. In addition, we represent that the annotation has been conducted by John and has a high confidence.

Listing 5: Semantic Annotation of the Image of Einstein with Creator and Confidence

```
<m3oann:AnnotationPattern rdf:about="#sapd-1">
  <dul:defines>
    <m3oann:AnnotatedConcept rdf:about="#apc-2">
      <dul:classifies>
        <m3oann:Image rdf:about="#einstein-1" />
      <dul:classifies>
    </m3oann:AnnotatedConcept>
  </dul:defines>
  <dul:defines>
    <m3oann:AnnotationConcept rdf:about="#ac-2">
```

```
        <dul:classifies>
          <m3oann:Creator rdf:about="http://de.wikipedia.org/wiki/
              Ferdinand_Schmutzer">
        </dul:classifies>
      </m3oann:AnnotationConcept>
  </dul:defines>
</m3oann:AnnotationPattern>

<m3opro:ProvenancePattern rdf:about="#sapd-1">
  <dul:defines>
      <m3opro:AppliedMethodRole rdf:about="#amr-2">
          <dul:classifies>
            <m3opro:ManualAnnotation rdf:about="#ma-1">
                <m3opro:hasQuality>
                  <m3opro:ConfidenceQuality rdf:about="cq-1">
                    <m3opro:hasRegion rdf:about="#cf-1">
                  </m3opro:ConfidenceQuality>
                </m3opro:hasQuality>
            </m3opro:ManualAnnotation>
          </dul:classifies>
      </m3opro:AppliedMethodRole>
  </dul:defines>
  <dul:defines>
      <m3opro:AuthorRole rdf:about="#a2-2">
          <m3opro:isRelatedToConcept rdf:resource="#amr-2">
          <dul:classifies rdf:resource="#john-1">
      </m3opro:AuthorRole>
  </dul:defines>
  <dul:defines>
      <m3opro:ConfidenceRole rdf:about="#cr-1">
          <dul:classifies>
            <m3opro:ConfidenceRegion rdf:about="#cf-1">
                <m3opro:hasConfidenceValue xsd:float="0.95">
            </m3opro:ConfidenceRegion>
          </dul:classifies>
      </m3opro:ConfidenceRole>
  </dul:defines>
</m3opro:ProvenancePattern>
```

B Axiomatization of the M3O in Description Logics

B.1 Decomposition Pattern

$$AnnotatedConcept \sqsubseteq Concept \tag{1}$$
$$AnnotatedConcept \sqsubseteq =1(classifies.InformationEntity) \tag{2}$$
$$AnnotatedConcept \sqsubseteq \forall classifies.InformationEntity \tag{3}$$
$$AnnotatedConcept \sqsubseteq =1(isDefinedIn.AnnotationPattern) \tag{4}$$
$$AnnotatedConcept \sqsubseteq \forall isDefinedIn.AnnotationPattern \tag{5}$$
$$\tag{6}$$

$$
\begin{aligned}
AnnotationConcept &\sqsubseteq Concept && (7)\\
AnnotationConcept &\sqsubseteq =1(classifies.Entity) && (8)\\
AnnotationConcept &\sqsubseteq =1(isDefinedIn.AnnotationPattern) && (9)\\
AnnotationConcept &\sqsubseteq \forall isDefinedIn.AnnotationPattern && (10)\\
& && (11)
\end{aligned}
$$

$$
\begin{aligned}
AnnotationSituation &\sqsubseteq Situation && (12)\\
AnnotationSituation &\sqsubseteq =1(isSettingFor.(InformationEntity\sqcap && (13)\\
&\quad \exists classifiedBy.AnnotatedConcept)) && \\
AnnotationSituation &\sqsubseteq \geq 1(isSettingFor.(InformationEntity\sqcap && (14)\\
&\quad \exists classifiedBy.AnnotationConcept)) && \\
AnnotationSituation &\sqsubseteq =1(satisfies.AnnotationPattern) && (15)\\
AnnotationSituation &\sqsubseteq \forall satisfies.AnnotationPattern && (16)\\
& && (17)
\end{aligned}
$$

$$
\begin{aligned}
AnnotationPattern &\sqsubseteq Pattern && (18)\\
AnnotationPattern &\sqsubseteq =1(defines.AnnotatedConcept) && (19)\\
AnnotationPattern &\sqsubseteq \geq 1(defines.AnnotationConcept) && (20)\\
AnnotationPattern &\sqsubseteq =1(isSatisfiedBy.AnnotationSitation) && (21)\\
AnnotationPattern &\sqsubseteq \forall(isSatisfiedBy.AnnotationSituation) && (22)\\
& && (23)
\end{aligned}
$$

B.2 Decomposition Pattern

$$
\begin{aligned}
ComponentConcept &\sqsubseteq Concept && (24)\\
ComponentConcept &\sqsubseteq =1(classifies.InformationEntity) && (25)\\
ComponentConcept &\sqsubseteq \forall classifies.InformationEntity && (26)\\
ComponentConcept &\sqsubseteq =1(isDefinedIn.DecompositionPattern) && (27)\\
ComponentConcept &\sqsubseteq \forall isDefinedIn.DecompositionPattern && (28)\\
& && (29)
\end{aligned}
$$

$$
\begin{aligned}
CompositeConcept &\sqsubseteq Concept && (30)\\
CompositeConcept &\sqsubseteq =1(classifies.InformationEntity) && (31)\\
CompositeConcept &\sqsubseteq \forall classifies.InformationEntity && (32)\\
CompositeConcept &\sqsubseteq =1(isDefinedIn.DecompositionPattern) && (33)\\
CompositeConcept &\sqsubseteq \forall isDefinedIn.DecompositionPattern && (34)\\
& && (35)
\end{aligned}
$$

$$DecompositionSituation \sqsubseteq Situation \quad (36)$$
$$DecompositionSituation \sqsubseteq \geq 1(isSettingFor.(InformationEntity \sqcap \exists classifiedBy.ComponentConcept)) \quad (37)$$
$$DecompositionSituation \sqsubseteq = 1(isSettingFor.(InformationEntity \sqcap \exists classifiedBy.CompositeConcept)) \quad (38)$$
$$DecompositionSituation \sqsubseteq = 1(satisfies.DecompositionPattern) \quad (39)$$
$$DecompositionSituation \sqsubseteq \forall satisfies.DecompositionPattern \quad (40)$$
$$(41)$$

$$DecompositionPattern \sqsubseteq Pattern \quad (42)$$
$$DecompositionPattern \sqsubseteq = 1(defines.CompositeConcept) \quad (43)$$
$$DecompositionPattern \sqsubseteq \geq 1(defines.ComponentConcept) \quad (44)$$
$$DecompositionPattern \sqsubseteq = 1(isSatisfiedBy.DecompositionSitation) \quad (45)$$
$$DecompositionPattern \sqsubseteq \forall(isSatisfiedBy.DecompositionSituation) \quad (46)$$
$$(47)$$

B.3 Collection Pattern

$$InformationEntityCollection \sqsubseteq InformationEntity \sqcap Collection \quad (48)$$
$$(49)$$

$$CollectionConcept \sqsubseteq Concept \quad (50)$$
$$CollectionConcept \sqsubseteq = 1(classifies.InformationEntityCollection) \quad (51)$$
$$CollectionConcept \sqsubseteq \forall classifies.InformationEntityCollection \quad (52)$$
$$CollectionConcept \sqsubseteq = 1(isDefinedIn.CollectionPattern) \quad (53)$$
$$CollectionConcept \sqsubseteq \forall isDefinedIn.CollectionPattern \quad (54)$$
$$(55)$$

$$ElementConcept \sqsubseteq Concept \quad (56)$$
$$ElementConcept \sqsubseteq = 1(classifies.InformationEntity) \quad (57)$$
$$ElementConcept \sqsubseteq \forall classifies.InformationEntity \quad (58)$$
$$ElementConcept \sqsubseteq = 1(isDefinedIn.CollectionPattern) \quad (59)$$
$$ElementConcept \sqsubseteq \forall isDefinedIn.CollectionPattern \quad (60)$$
$$(61)$$

$$
\begin{aligned}
CollectionSituation &\sqsubseteq Situation && (62)\\
CollectionSituation &\sqsubseteq =1(isSettingFor. && (63)\\
&\quad (InformationEntityCollection\sqcap && (64)\\
&\quad \exists classifiedBy.CollectionConcept)) &&\\
CollectionSituation &\sqsubseteq \geq 1(isSettingFor.(InformationEntity\sqcap && (65)\\
&\quad \exists classifiedBy.ElementConcept)) &&\\
CollectionSituation &\sqsubseteq =1(satisfies.CollectionPattern) && (66)\\
CollectionSituation &\sqsubseteq \forall satisfies.CollectionPattern && (67)\\
& && (68)
\end{aligned}
$$

$$
\begin{aligned}
CollectionPattern &\sqsubseteq Pattern && (69)\\
CollectionPattern &\sqsubseteq =1(defines.CollectionConcept) && (70)\\
CollectionPattern &\sqsubseteq \geq 1(defines.ElementConcept) && (71)\\
CollectionPattern &\sqsubseteq =1(isSatisfiedBy.CollectionSitation) && (72)\\
CollectionPattern &\sqsubseteq \forall(isSatisfiedBy.CollectionSituation) && (73)\\
& && (74)
\end{aligned}
$$

B.4 Provenance Pattern

$$
\begin{aligned}
MethodConcept &\sqsubseteq Concept && (75)\\
MethodConcept &\sqsubseteq =1(classifies.Entity) && (76)\\
MethodConcept &\sqsubseteq \geq 1isRelatedToConcept.AppliedMethodRole && (77)\\
MethodConcept &\sqsubseteq =1(isDefinedIn.ProvenancePattern) && (78)\\
MethodConcept &\sqsubseteq \forall isDefinedIn.ProvenancePattern && (79)\\
& && (80)
\end{aligned}
$$

$$
\begin{aligned}
AppliedMethodRole &\sqsubseteq Role && (81)\\
AppliedMethodRole &\sqsubseteq =1(classifies.Method) && (82)\\
AppliedMethodRole &\sqsubseteq \forall classifies.Method && (83)\\
AppliedMethodRole &\sqsubseteq =1(isDefinedIn.ProvenancePattern) && (84)\\
AppliedMethodRole &\sqsubseteq \forall isDefinedIn.ProvenancePattern && (85)\\
AppliedMethodRole &\sqsubseteq \geq 1isRelatedToConcept.Concept && (86)\\
& && (87)
\end{aligned}
$$

$$
\begin{aligned}
ProvenanceSituation &\sqsubseteq Situation && (88)\\
ProvenanceSituation &\sqsubseteq =1(isSettingFor. && (89)\\
&\quad (Method\sqcap && (90)\\
&\quad \exists classifiedBy.AppliedMethodRole)) &&\\
ProvenanceSituation &\sqsubseteq \geq 1(isSettingFor.(Entity\sqcap && (91)\\
&\quad \exists classifiedBy.MethodConcept)) &&\\
ProvenanceSituation &\sqsubseteq =1(satisfies.ProvenancePattern) && (92)\\
ProvenanceSituation &\sqsubseteq \forall satisfies.ProvenancePattern && (93)\\
& && (94)
\end{aligned}
$$

$$
\begin{aligned}
ProvenancePattern &\sqsubseteq Pattern && (95)\\
ProvenancePattern &\sqsubseteq \geq 1(defines.AppliedMethodRole) && (96)\\
ProvenancePattern &\sqsubseteq \geq 1(defines.MethodConcept) && (97)\\
ProvenancePattern &\sqsubseteq =1(isSatisfiedBy.ProvenanceSitation) && (98)\\
ProvenancePattern &\sqsubseteq \forall(isSatisfiedBy.ProvenanceSituation) && (99)\\
& && (100)
\end{aligned}
$$

A.3 Appending on Integrating Existing Multimedia Metadata Formats and Metadata Standards and the M3O

Integration of Existing Multimedia Metadata Formats and Metadata Standards in the M3O

Daniel Eißing, Ansgar Scherp, Carsten Saathoff
WeST, University of Koblenz-Landau, Germany
{eissing,scherp,saathoff}@uni-koblenz.de

Abstract

With the Multimedia Metadata Ontology (M3O), we have developed a sophisticated model for representing among others the annotation, decomposition, and provenance of multimedia metadata. The goal of the M3O is to integrate existing metadata standards and metadata formats rather than replacing them. To this end, the M3O provides a scaffold needed to represent multimedia metadata. Being an abstract model for multimedia metadata, it is not straightforward how to use and specialize the M3O for concrete application requirements and existing metadata formats and metadata standards.

In this paper, we present a step-by-step alignment method describing how to integrate and leverage existing multimedia metadata standards and metadata formats in the M3O in order to use them in a concrete application. We demonstrate our approach by integrating three existing metadata models: the Core Ontology on Multimedia (COMM), which is a formalization of the multimedia metadata standard MPEG-7, the Ontology for Media Resource of the W3C, and the widely known industry standard EXIF for image metadata.

1 Introduction

A densely populated jungle with a myriad of partially competing species of different colors and size—this might be a good characterization of today's world of multimedia metadata formats and metadata standards. Looking at the existing metadata models like [1–5] and metadata standards such as [6–10], we find it hard to decide which of them to use in a complex multimedia application. They focus on different media types, are very generic or designed for a specific application domain, and overlap in the functionality provided.

However, building a complex multimedia application often requires using several of these standards *together*, e.g., when different tools have to be integrated along the media production process [11]. The integration among tools requires interoperability of different metadata standards, which is a requirement that is not sufficiently satisfied by existing formats and standards. With

XMP [7], there exists an important initiative to enable interoperability along the production process of images. Nevertheless, this work is limited with respect to the functionality provided and focuses on the media type image only [12]. Overall, the XMP initiative is an important step but more is required to facilitate multimedia metadata interoperability along the media production process.

To solve this problem, we have developed the Multimedia Metadata Ontology (M3O) [12]. The M3O is a sophisticated model for representing among others the annotation, decomposition, and provenance of multimedia content and multimedia metadata. The goal of the M3O is to provide a framework for the integration of existing metadata formats and metadata standards rather than replacing them. The M3O bases on a foundational ontology and by this inherits its rich axiomatization. It follows a pattern-based ontology design approach, which allows the M3O to arrange the different functionalities for representing multimedia metadata into smaller, modular, and reusable units.

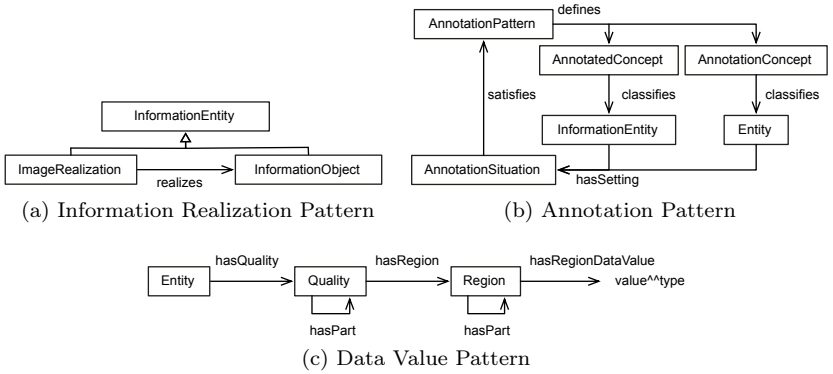
However, the M3O was designed as an abstract model providing a scaffold for representing arbitrary multimedia metadata. As such, the integration of existing standards is not straightforward, and we are confronted with a gap between the formal model and its application in concrete domains. In this paper, we fill this gap and present a step-by-step alignment method describing how to integrate existing formats and standards for multimedia metadata and the M3O. We describe the tasks that have to be performed for this integration and apply the integrated ontology to a concrete modeling task. We demonstrate this integration at the example of the Core Ontology on Multimedia (COMM) [2], which is a formalization of the multimedia metadata standard MPEG-7 [9], the recently released Ontology for Media Resource [13] of the W3C, and the widely known and adopted industry standard EXIF [6] for image metadata.

2 Introduction to the Multimedia Metadata Ontology

The Multimedia Metadata Ontology (M3O) [12] provides a generic modeling framework to integrate existing multimedia metadata formats and metadata standards. The M3O is modeled as a highly axiomatized core ontology basing on the foundational ontology DOLCE+DnS Ultralight (DUL) [14]. DUL provides a philosophically grounded conceptualization of the most generic concepts such as objects, events, and information. The axiomatization is formulated in Description Logics [15].

The M3O follows a pattern-based approach to ontology design. Each pattern is focused on modeling a specific and clearly identified aspect of the domain. From an analysis of existing multimedia metadata formats and metadata standards [12], we have identified six patterns required to express the metadata for multimedia content. These patterns model the basic structural elements of existing metadata models and are the Decomposition Pattern, Annotation Pattern, Information Realization Pattern, Data Value Pattern for representing

complex values, Collection Pattern, and Provenance Pattern. Basing a model like the M3O on ontology design patterns ensures a high degree of modularity and extensibility, while at the same time a high degree of axiomatization and thus semantic precision is retained. In order to realize a specific multimedia metadata format or metadata standard in M3O, these patterns need to be specialized. In the following, we discuss three patterns of the M3O in more detail, namely the *Information Realization Pattern*, *Annotation Pattern*, and *Data Value Pattern*, which we will mainly refer to in the upcoming sections.



(a) Information Realization Pattern

(b) Annotation Pattern

(c) Data Value Pattern

Figure 1: Three Example Patterns of the Multimedia Metadata Ontology (M3O)

Information Realization The information realization pattern in Figure 1a models the distinction between information objects and information realizations [14]. Consider a digital image that is stored on the hard disk in several formats and resolutions. An information object represents the image as an abstract *concept* or *idea*, namely the information object of an image. Different files may *realize* this same abstract idea. As shown in Figure 1a, the pattern consists of the InformationRealization that is connected to the InformationObject by the realizes relation. Both are subconcepts of InformationEntity, which allows treating information in a general sense as we will see in the Annotation Pattern.

Annotation Pattern Annotations are understood in the M3O as the attachment of metadata to an information entity. Metadata comes in various forms such as low-level descriptors obtained by automatic methods, non-visual information covering authorship or technical details, or semantic annotation aiming at a formal and machine-understandable representation of the contents. Our Annotation Pattern models the basic structure that underlies all types of annotation. This allows for assigning arbitrary annotations to information entities while providing the means for modeling provenance and context. In Figure 1b, we see that an annotation is not modeled as a direct relationship between some media item and an annotation. It is defined by a more complex structure, which

is inherited by the Descriptions and Situations Pattern of DUL. Basically, a Descriptions and Situations Pattern is two-layered. The *Description* defines the structure, in this case of an annotation, which contains some entity that is annotated and some entity that represents the metadata. The *Situation* contains the concrete entities for which we want to express the annotation. The pattern allows us to add further concepts and entities into the *context* of an annotation, e.g., expressing provenance or confidence information. On the top half, we see that the AnnotationPattern defines an AnnotatedConcept and an AnnotationConcept. The AnnotatedConcept classifies an InformationEntity and thus expresses that the information entity is the subject of the annotation. The AnnotationConcept classifies some Entity, which identifies this entity as the annotation or metadata. The entity can be some complex data value, e.g., representing some low-level features represented using the Data Value Pattern, but also some concept located in a domain ontology such as DBpedia[1]. All the entities have as setting the AnnotationSituation, which satisfies the AnnotationPattern.

Data Value Pattern In ontologies we mainly use abstract concepts and clearly identifiable individuals to represent data. However, we also need the means to represent concrete data values such as strings and numerical values. In DUL, there exists the concept Quality in order to represent attributes of an Entity, i.e., attributes that only exist together with the Entity. Regions are used to represent the values of an Quality and the data space they come from. The Data Value Pattern depicted in Figure 1c assigns a concrete data value to an attribute of that entity. The attribute is represented by the concept Quality and is connected to the Entity by the hasQuality property. The Quality is connected to a Region by the hasRegion relation. The Region models the data space the value comes from. We attach the concrete value to the Region using the relation hasRegionDataValue. The data value is encoded using typed literals, i.e., the datatype can be specified using XML Schema Datatypes [16].

3 Alignment Method

This section introduces our method for aligning multimedia standards and multimedia formats with the M3O. The method has been derived from our experiences applying and specializing the M3O for three existing multimedia formats and standards, namely COMM [2], the Ontology for Media Resource [13], and EXIF [6].

In contrast to automatic, adaptive, or machine learning approaches for ontology alignment [17–19], we conduct a pure manual alignment, as only a manual alignment ensures the high quality of the integration and minimizes ambiguities and imprecise matching. We consider the time and effort for manual alignment manageable, as we assume that each metadata format or standard has to be aligned only once and that updates to the integrated formats or standards will be infrequent and mostly incremental.

[1] `http://dbpedia.org/`

For the alignment, we propose an iterative four-step alignment method that helps ontology engineers to integrate existing metadata standards and metadata formats. In each iteration, we consecutively evolve the alignment of the format or standard with the M3O. Following an iterative approach, we are able to identify, analyze, and flexibly react to problems and challenges encountered during previous iterations of the alignment.

Each iteration consists of four steps. The first step targets the understanding of the format or standard to be integrated. The second step reorders concepts into coherent groups. The third step maps concepts and structure with the M3O. The fourth step proves and documents the validity of the alignment, and finalizes the iteration.

To introduce our alignment method, we proceed as follows: For each step, we first outline the goals and provide a brief summarization. Subsequently, we describe the core tasks to be performed within the step, and provide concrete examples that show its relevance and application for concrete ontologies.

3.1 Step 1: Understanding

Summary A precise understanding of the metadata format or standard to be integrated is an import prerequisite for aligning it with the M3O. Consequently, the first step of alignment is an in-depth analysis of the structure and core concepts of the model at hand. While this advise may seem obvious, this is a task easily underestimated and problems neglected at an early stage can cause time-consuming problems along the integration process. Additional documentation, if available, will help to (re-)produce the overall structure not explicitly expressed in the formal specification.

Detailed Description and Examples In general, we have found three distinct modeling approaches to be very common for multimedia metadata formats and metadata standards.

Pattern-based Pattern-based ontologies, e.g., COMM, provide a high degree of axiomatization and structure in a formal and precise way. Through our analysis, we understand the patterns used and the functionality they provide. This allows us to compare the patterns of the ontology to be integrated with those provided by the M3O.

Predicate-centric In a predicate-centric approach, as followed e.g. by the Ontology for Media Resource, the ontology is mainly specified through a set of properties. Such a model offers very little structure in a machine readable format, e.g., in terms of conceptual relations between properties. However, by analyzing the documentation, we infer additional information about intended groupings of properties and the structural composition of the format or standard to be integrated.

Legacy Models Other formats and standards have not yet been semantified at all. By analyzing the concepts and relations expressed in the specification

of the format, we decide how the core concepts can be expressed in a formal and precise way using the M3O.

Ambiguities that are found during the initial analysis are discussed at this point. It is not our intention to revise all modeling decisions made for the ontology to be integrated. However, we consider the alignment a good opportunity to correct some of the *bad smells* [20] discovered. Once we have reached a sufficient understanding of the format or standard to be integrated, we proceed with the grouping step.

3.2 Step 2: Grouping

Summary Ontologies should provide structural information on the relation and groupings of concepts it defines. However, although many formats or standards provide this information in their documentation, the information is sometimes lost when the models are transformed to an ontology. By using the original specifications and documentations, we are able to preserve and recreate this information grouping, and provide them through formal specification in the aligned ontology.

Detailed Description and Examples In principle, we distinguish three forms of available grouping information:

Explicit Grouping Pattern-based models provide an explicit grouping of concepts into coherent patterns, often accompanied by a rich axiomatization on how they relate. As an example, the definition of a color histogram annotation in COMM specifies a ColorQuantizationComponentDescriptorParameter that groups the concepts ColorComponent and NumberOfBinsPerComponent.

Implicit Grouping For other metadata models grouping information may not be explicitly represented. This is often the case with predicate-centric approaches, e.g., the Ontology on Media Resource. In these cases, we refer to the textual documentation in order to (re-)construct the implicit groupings of the properties or classes. As an example, the documentation of the Ontology on Media Resource offers a textual description on the grouping of its properties, e.g., in terms of identification or creation. However, this information is not accessible in the RDF representation as proposed by the W3C. By defining the appropriate axioms, we have appended the implicit grouping information in a formal and explicit way, e.g., by stating that an IdentificationAnnotation hasPart some TitleAnnotation, LanguageAnnotation, and LocatorAnnotation.

Recovery of Groupings In other cases grouping information is lost when transferring multimedia formats or standards to RDF. For example the EXIF metadata standard provides textual descriptions about groupings,

e.g., in terms of pixel composition and geo location. However, this distinction got lost in the adaption to an RDF schema [21]. For the alignment with the M3O, we have reconstructed the grouping information and appended it to the model in a formal and explicit way.

Once we have provided all relevant grouping relations through a formal specification, we continue with the mapping step.

3.3 Step 3: Mapping

Summary This step achieves the mapping of the ontology's concepts and structure to the scaffold provided by the M3O. The goal of this step is to create a working ontology, which, after validation, can be published or used as basis for further iterations.

Detailed Description and Examples For the alignment we follow a sequence of the following three steps:

1. **Mapping of Concepts** If some superclass of the concept to be aligned is present in both ontologies, direct mapping of concepts is feasible. This is mainly the case for ontologies that share the same foundation, e.g., COMM and the M3O, which both base on the DUL foundational ontology. All axioms of the aligned concepts are preserved as long as they are applicable through the M3O. If a concept is not applicable in the M3O, we align all dependent subclasses and references to the nearest matching M3O concept. As an example, the COMM DigitalData concept, which is a subclass of the DUL InformationObject, was removed during the alignment. The dangling dependencies and references have been resolved by subclassing or referencing the InformationObject instead.

2. **Structural Mapping** For structural mapping, we consider the functionality of the pattern or structure to be mapped. If a pattern or structure offers the same or an equal functionality than a pattern of the M3O, we can replace the pattern. By adapting the M3O pattern, we are often able to express the same functionality using a more generic approach. As an example, COMM proposes the Digital Data Pattern to express data values in a digital domain. A similar functionality is provided by the M3O Data Value Pattern, which expresses data values through the generic concepts of Quality and Region. The COMM Digital Data Pattern can be considered a special case of expressing data values and therefor has been replaced using the M3O Data Value Pattern instead.

   In the same manner, we simplify the structural composition of the existing model by merging multiple concepts and patterns that offer the same or an equal functionality. As an example, COMM defines three annotation patterns. Each deals with a different aspect of multimedia annotation, although they vary only slightly in their structural composition. We have

aligned those patterns by adapting the M3O Annotation Pattern. The domain specific concepts that result from the separation into three distinct patterns have been preserved by subclassing the corresponding concepts of the M3O Annotation Pattern. This simplifies the structure of the model, while also preserving the original functionality.

3. **Removing Unnecessary Concepts** We finalize the mapping step by cleaning up unused dependencies from the ontology files. Concepts that either have no further relevance for the target context or are sufficiently covered by the M3O are removed at this point. An example, the COMM AnnotatedMediaRole offers an equal functionality as the M3O AnnotatedInformationRealizationConcept. We therefore have removed COMM's AnnoatedMediaRole and replaced any formal relation that involves the concept.

3.4 Step 4: Validation and Documentation

In each iteration of the alignment process, we need to check the consistency of the resulting ontology. This can be done by using a reasoner like Fact++[2] or Pellet[3]. Any problem encountered during the alignment can be resolved by reiterating the four steps of the alignment method. After proving the consistency of the resulting ontology, we finalize the process by documenting all major decisions and adjustments made during the alignment.

3.5 Summary

In this section, we have proposed a four-step method for aligning multimedia metadata formats and multimedia metadata standards with the M3O. In the following Sections 4-6, we demonstrate the alignment of three existing formats and standards by applying our method. They are the Core Ontology on Multimedia, the Ontology for Media Resource, and the EXIF metadata standard.

4 Example 1: Core Ontology on Multimedia (COMM)

The Core Ontology on Multimedia (COMM) [2] is a formal specification of the MPEG-7 metadata standard [9]. In contrast to other approaches to modeling MPEG-7 as an ontology COMM is not designed as a one-to-one mapping, but provides a set of patterns that cover the core and repetitive building blocks of MPEG-7. The central challenge of the alignment of COMM and M3O is understanding the patterns of COMM and mapping them to the scaffold provided by the M3O. This section describes the experiences and challenges of aligning COMM and the M3O, using the four-step alignment method proposed above.

[2] http://owl.man.ac.uk/factplusplus/
[3] http://clarkparsia.com/pellet/

4.1 Application of the Alignment Method

Understanding COMM follows a pattern-based approach and builds on the DUL foundational ontology. Some of the core patterns, i.e., the Descriptions and Situations Pattern, are shared between COMM and the M3O. Others, e.g., the Digital Data Pattern, form major structural differences.

COMM defines five structural patterns, namely the Content Annotation Pattern, Media Annotation Pattern, and Semantic Annotation Pattern for media annotation, the Decomposition Pattern for media (de-)composition, and the Digital Data Pattern, which expresses annotations in a digital domain. Domain specific knowledge is separated from the core concepts and defined in separate ontologies, e.g., concepts concerning annotation of visual entities are defined in the *visual ontology*.

Some ambiguities that were found in the initial analysis have been resolved at this point. As an example, COMM specifies concepts such as NumberOfBinsPerComponent that are specialization of both Parameter and Region. While this may not be syntactically incorrect, it violates the DnS pattern of DUL. In the DnS pattern, a Parameter parametrizes a Region. Thus, these two concepts should not have common sub-concepts. To solve this problem, we have removed the superclass relations to the Parameter concept and introduced a parametrizes relation. For example, COMM specified a ColorComponent and NumberOfBinsPerComponent, which are subclasses of both the ColorQuantizationComponentDescriptorParameter and the Region concept. We have removed the superclass relation from the ColorComponent and NumberOfBinsPerComponent to the ColorQuantizationComponentDescriptorParameter, which instead now parametrizes these concepts.

Grouping Following a pattern-based design, COMM already provides a rich degree of conceptual groupings and their axiomatization in a machine readable format. However, reusability can be improved by redistributing concepts among the six ontologies of COMM, *core*, *datatype*, *localization*, *media*, *visual*, and *textual* respectively. As an example, the concept RootSegmentRole, located in the COMM *core ontology*, is not used in any pattern definition and has therefore been relocated to the *localization ontology*.

Mapping The main challenge of aligning COMM and the M3O concern the differences of the patterns used and how to relate them. Although some principles are shared between the ontologies, there are also major differences, e.g., the Digital Data Pattern of COMM and the Information Realization Pattern of the M3O.

Often COMM patterns have been replaced using a more generic pattern of the M3O. As an example, Figure 2 displays the adaptation of the COMM Digital Data Pattern through the M3O. For the alignment, we have decided that the functionality of the Digital Data Pattern, i.e., expressing data values, can be maintained by adopting the M3O Data Value Pattern instead. All related concepts have either been removed or mapped to the next matching M3O

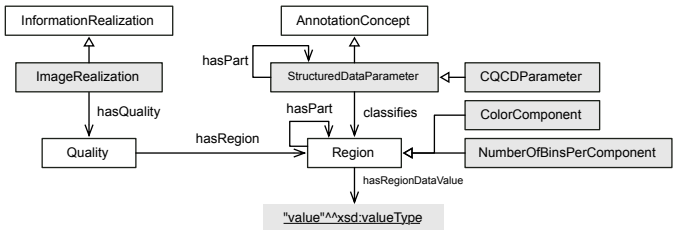


Figure 2: Excerpt of COMM subsequently to the integration with the M3O. White boxes show the concepts of the M3O or DUL, whereas grey boxes represent concepts of COMM aligned to the M3O.

concept. As an example, the StructuredDataDescription concept has been removed as it held no further relevance in the context of the Data Value Pattern. The StructuredDataParameter concept on the other hand has been preserved as specialization of the M3O Annotation Pattern. To accommodate StructuredDataParameters with the M3O, we consider StructuredDataParameters as subclass of the AnnotationConcept. Through parametrizing the appropriate Region, we can constrain the range applicable for a specific StructuredDataParameter. The value itself is expressed using the hasRegionDataValue relation. In a similar manner, the three annotation patterns of COMM have been replaced through the M3O Annotation Pattern, and all dependent concepts have been mapped to the M3O Annotation Pattern instead.

Validation and Documentation The alignment of COMM and the M3O has been validated using Fact++ and Pellet reasoner. The results have been documented in a publicly accessible wiki page available at: http://semantic-multimedia.org/index.php/COMM_integration.

4.2 Application of the Aligned Ontology

Figure 3 demonstrates the application of StructuredDataParameters using COMM aligned with the M3O. We specify a ColorQuantizationComponentDescriptorParameter (CQCDParameter) as part of the RGBHistogramAnnotationConcept. The CQCDParameter parametrizes the ColorComponents and NumberOfBinsPerComponent, which are considered part of the RGBHistogramRegion. The hasRegionDataValue relation expresses the primitive value for this annotation, e.g., an unsigned int for the NumberOfBinsPerComponent concept. Staying in line with the specification of the M3O Data Value Pattern, we consider the use of StructuredDataParameters optional. Thus, we do not specify that an AnnotationConcept must specify any StructuredDataParameters in a hasPart relation but recommend using them as they add an additional layer of formal expressiveness.

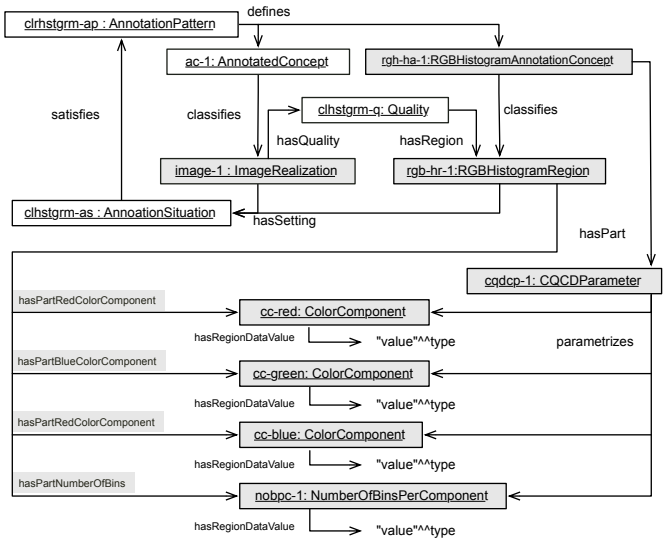


Figure 3: Application of COMM after its integration with the M3O

5 Example 2: Ontology for Media Resource

The Ontology for Media Resource [13] developed by the W3C defines a core vocabulary for multimedia annotation. The ontology targets at an unifying mapping of common media formats like EXIF [6] or Dublin Core [10]. The core challenge for this alignment of the Ontology for Media Resource concerns the mapping of properties to either information object or information realization as provided by the Information Realization Pattern of the M3O.

5.1 Application of the Alignment Method

Understanding The Ontology for Media Resource presents a property-centric approach to ontology modeling and consists of 28 predicates including properties like *title* and *language*. Some properties are specified in further detail, e.g., through *role* or *type* properties. Only entities such as multimedia items and persons are represented as resources. Any other information such as roles or types are represented using primitive values, e.g., strings. The Ontology for Media Resource defines neither structural patterns nor formal logical constraints beyond the domain and range specification for each property. Unlike the M3O, there is no distinction between information object and information realization.

Grouping The Ontology for Media Resource's documentation on the web provides a number of conceptual groupings for certain aspects of multimedia description, e.g., identification or fragmentation. However, this information is not accessible in a machine readable format. With the alignment of the Ontol-

ogy for Media Resource and the M3O, we have provided grouping information by defining the appropriate axioms. For example an IdentificationAnnotation concept hasPart some TitleAnnotationConcept, LanguageAnnotationConcept, and LocatorAnnotationConcept.

Mapping For mapping the ontology for Media Resource to the M3O, we define a subconcept of the AnnotationConcept for each predicate of the ontology. For example, we define a LocatorAnnotationConcept to match the *locator* property. Concrete values are expressed using the Data Value Pattern of the M3O. To this end, we define appropriate Region concepts. In the case of the LocatorAnnotationConcept, we define a LocatorRegion with the property hasRegionDataValue and an URI specifying a concrete location on the web.

Of primary concern for this alignment is the mapping with the Information Realization Pattern. By taking into account the difference between information objects and information realizations, we can improve semantic precision of the aligned ontology. To this end, we examine each attribute of the Ontology for Media Resource for its inherent meaning and constrain it to the appropriate concept of the Information Realization Pattern of the M3O. As an example, the *locator* property of the Ontology for Media Resource annotates media files that are locatable on the web. This is a quality only applicable for information realizations and is expressed in the definition of the LocatorAnnotationConcept.

We express the *type* property of the Ontology for Media Resource through specialization, e.g., by specifying an ImageRealization, a subclass of the InformationRealization, as the type for the considered media item. Finally the *fragments* facet of the Ontology for Media Resource has been modeled using the Decomposition Pattern of the M3O. The functionality indicated by the *namedFragments* property can be obtained by decomposing multimedia items using the M3O Decomposition Pattern and by using the M3O Annotation Pattern to annotate the resulting fragment with a FragmentLabelAnnotationConcept.

Consistency Checking and Documentation The resulting ontology has been validated using Fact++ and Pellet.

5.2 Application of the Aligned Ontology

Figure 4 demonstrates the application of the aligned ontology. We explicitly distinguish between an ImageObject and an ImageRealization that realizes the ImageObject. The specific type for each media is expressed through specialization of the corresponding InformationObject and InformationRealization concepts. The ImageObject is annotated with some TitleAnnotationConcept, where the title "Mona Lisa" is expressed using the Data Value Pattern. The ImageRealization is annotated with some LocatorAnnotationConcept that parametrizes a Region for an URI locatable on the web.

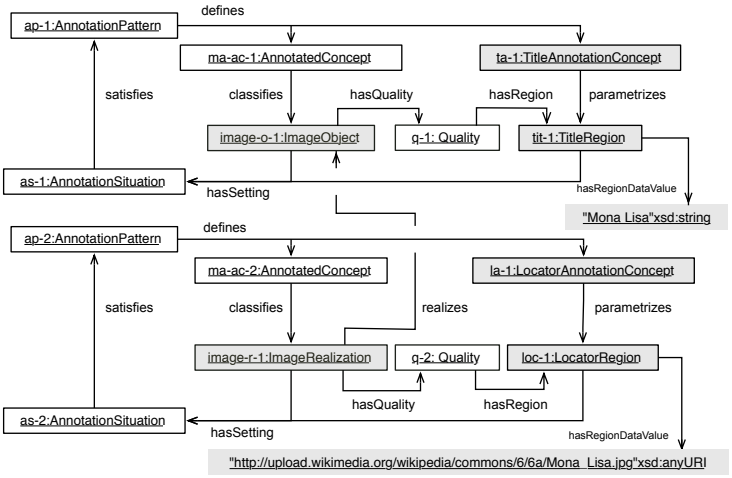


Figure 4: Application of the Ontology on Media Resource after the alignment

6 Example 3: EXIF

EXIF is a common metadata standard for images and supports mainly technical metadata [6]. It is embedded directly into media assets such as JPEG files. The following section presents the alignment of EXIF and the M3O.

6.1 Application of the Alignment Method

Understanding The key-value based metadata specified in EXIF is binary encoded into the header of, e.g., JPEG files. Consequently the mapping of the non-semantified concepts onto the scaffold of the M3O posed the major challenge for this particular alignment. Thus, for aligning EXIF and the M3O, we first needed to semantify the key-value pairs of EXIF.

Grouping The EXIF metadata standard has been translated to a RDF Schema [21] by the W3C through an one-to-one mapping. Here, each key of the EXIF specification has been directly mapped to a corresponding property. This approach ignores the groupings of metadata keys that is provided in the original EXIF specification such as pixel composition and geo location. For the alignment, we have reconstructed this grouping information.

Mapping Mapping EXIF to the M3O follows a similar procedure as conducted with the mapping of the Ontology of Media Resource. Special consideration is provided on how to map EXIF properties to information objects and information realizations. For example, locations have been constrained to information objects, as they convey information on where the original picture has been taken. Image resolutions describe a quality of a concrete realization,

e.g., a JPEG file, and are therefore associated with the information realization instead. Specific properties can be referred to by using preexisting vocabularies, e.g., the WGS84 vocabulary [22] for GPS information. As EXIF restrains itself to describing qualities of multimedia items all keys have been mapped as specialization of the M3O Annotation Pattern.

Consistency Checking and Documentation We have tested the validity of the resulting ontology using Fact++ and Pellet.

6.2 Application of the Aligned Ontology

The example as shown in Figure 5 defines an EXIFAnnotationPattern concept that allows us to represent EXIF compliant annotations. In this case, the EXIFAnnotationPattern defines an EXIFGeoParameter that parametrizes a GeoPoint. Within this construct, we accumulate all parameters that can be specified in regards to *GPS Attribute Information* as specified in EXIF. Going conform with the Data Value Pattern, we express the GeoPoint through geo:lat and geo:long, which specify primitive values of type xsd:decimal. In this case, we want to represent the location at which the image was created and thus attach the information to the information object.

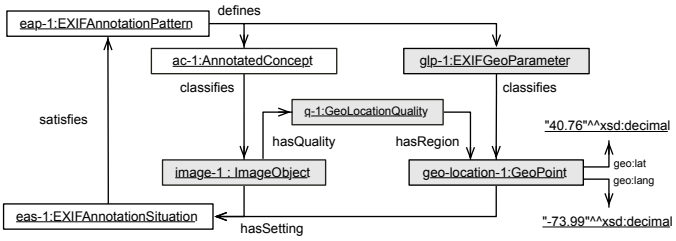


Figure 5: Application of the EXIF metadata standard using M3O.

7 Related Work

We shortly review the state of aligning different metadata standards. For a detailed analysis of existing metadata formats and metadata standards, we refer to the original M3O paper [12] and specifically to the report of the W3C Multimedia Semantics Incubator Group [23] and the overview of the current W3C Media Annotations Working Group [24].

Numerous metadata models and metadata standards with different goals and backgrounds have been proposed in research and industry. Most focus on a single media type such as image, text, or video, differ in the complexity of the data structures they provide, and provide partly overlapping and partly complementary functionality. With standards like EXIF [6], XMP [7], and IPTC [8] we

find metadata models that provide (typed) key-value pairs to represent metadata of the media type image. An example for a more complex standard is MPEG-7 [9]. MPEG-7 provides a rich set of complex descriptors that mainly focus on expressing low-level features of images, audio, and video.

The drawbacks of these standards are the lacking interoperability and the alignment between them. Harmonization efforts like the Metadata Working Group [25] or the Media Annotations Working Group [24] try to tackle these issues and develop a common vocabulary. However, they remain on the same technological level and do not extend their effort beyond the single media type of images and do not provide a generic framework suitable for arbitrary metadata formats and arbitrary metadata standards. XMP aims at an integrated standard for image metadata. However, it tackles the problem from a different point of view. While XMP also aims at providing a framework for multimedia metadata, it focusses on images only and does not consider other media types or structured multimedia content. Another major difference is that XMP stays on the level of standards such as EXIF or IPTC and does not take into account requirements such as provenance of metadata, decomposition, or information realization.

Several approaches have been published providing a formalization of MPEG-7 as an ontology [26], e.g., by Hunter [1] or the Core Ontology on Multimedia [2]. Although these ontologies provide clear semantics for the multimedia annotations, they still focus on MPEG-7 as the underlying metadata standard. More importantly, these ontologies basically provide a formalization of MPEG-7, but do not provide for the integration of different standards.

The alignment method discussed in this paper is fully manual. There are numerous publications about (semi-)automatic alignment and matching methods [17–19]. However, these methods typically do not provide the high accuracy we require in an alignment of different metadata standards and are usually applied to problems such as ontology learning or the alignment of domain models. The M3O is a core ontology, i.e., an ontology providing some underlying structure for specific aspects of an application. The method presented in this paper shows how to align existing metadata formats and metadata standards with such a core ontology. The goal of this work is producing a specialization of the M3O that inherits the same level of formal precision and conciseness. Achieving this goal with an automatic method currently seems not realistic.

8 Conclusions

In this paper, we have shown how the generic scaffold provided by the Multimedia Metadata Ontology (M3O) can be specialized to integrate existing multimedia metadata formats and metadata standards. To this end, we have developed a four-step alignment method that describes the tasks to be performed. We have demonstrated the applicability of our approach at the example of three existing metadata models, the Core Ontology on Multimedia, the Ontology for Media Resource of the W3C, and the industry standard EXIF for image metadata. The experiences made in conducting the alignment with the M3O have

been described. The results are also continuously documented on our wiki: http://www.semantic-multimedia.org/index.php/M3O:Main#Mappings

Acknowledgment. This research has been co-funded by the EU in FP7 in the WeKnowIt project (215453).

References

[1] Hunter, J.: Enhancing the semantic interoperability of multimedia through a core ontology. IEEE Transactions on Circuits and Systems for Video Technology **13**(1) (2003) 49–58

[2] Arndt, R., Troncy, R., Staab, S., Hardman, L., Vacura, M.: COMM: designing a well-founded multimedia ontology for the web. In: ISWC+ASWC. (2007) 30–43

[3] Markkula, M., Sormunen, E.: End-user searching challenges indexing practices in the digital newspaper photo archive. Information Retrieval **1**(4) (2000)

[4] Hollink, L., Schreiber, A.T., Wielinga, B.J., Worring, M.: Classification of user image descriptions. International J. of Human-Computer Studies **61**(5) (2004) 601 – 626

[5] Hollink, L., Schreiber, G., Wielinga, B.: Patterns of semantic relations to improve image content search. Web Semantics: Science, Services and Agents on the World Wide Web **5**(3) (2007)

[6] Japan Electronics and Information Technology Industries Association: Exchangeable image file format for digital still cameras: Exif version 2.2 (2002)

[7] Adobe Systems, Inc.: XMP Specifications (2008) `http://www.adobe.com/devnet/xmp/`.

[8] International Press Telecommunications Council: "IPTC Core" Schema for XMP Version 1.0 Specification document (2005) `http://www.iptc.org/`.

[9] MPEG-7: Multimedia content description interface. Technical report, Standard No. ISO/IEC n15938 (2001)

[10] Dublin Core Metadata Initiative: DCMI Metadata Terms (2008) `http://dublincore.org/documents/dcmi-terms/`.

[11] Hardman, L., Obrenovic, Z., Nack, F., Kerhervé, B., Piersol, K.W.: Canonical processes of semantically annotated media production. **14**(6) (2008) 327–340

[12] Saathoff, C., Scherp, A.: Unlocking the semantics of multimedia presentations in the web with the multimedia metadata ontology. In: WWW '10, ACM (2010) 831–840

[13] W3C: Ontology for media resource 1.0 (2010) http://www.w3.org/TR/mediaont-10/.

[14] Borgo, S., Masolo, C.: Foundational choices in DOLCE. In: Handbook on Ontologies. 2nd edn. Springer (2009)

[15] Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F., eds.: The Description Logic Handbook. Cambridge University Press (2003)

[16] Biron, P.V., Malhotra, A.: XML Schema Part 2: Datatypes Second Edition, W3C Recommendation. (2004) `http://www.w3.org/TR/xmlschema-2/`.

[17] Euzenat, J., Shvaiko, P.: Ontology matching. Springer (2007)

[18] Ehrig, M.: Ontology Alignment: Bridging the Semantic Gap. Volume 4 of Semantic Web and Beyond. Springer (2007)

[19] Blomqvist, E.: Ontocase-automatic ontology enrichment based on ontology design patterns. In: International Semantic Web Conference. (2009) 65–80

[20] Fowler, M., Beck, K., Brant, J., Opdyke, W., Roberts, D.: Refactoring: Improving the Design of Existing Code. Addison Wesley (1999)

[21] Kanzaki, M.: Exif vocabulary workspace - rdf schema (2003) `http://www.w3.org/2003/12/exif/`, last update in 2007.

[22] Brickley, D.: Basic Geo (WGS84 lat/long) Vocabulary (2006)

[23] Boll, S., Bürger, T., Celma, O., Halaschek-Wiener, C., Mannens, E., Troncy, R.: Multimedia Vocabularies on the Semantic Web (2007)

[24] Media Annotations Working Group: `http://www.w3.org/2008/WebVideo/Annotations/`.

[25] The Metadata Working Group: `http://www.metadataworkinggroup.org/`.

[26] Dasiopoulou, S., Tzouvaras, V., Kompatsiaris, I., Strintzis, M.G.: Enquiring MPEG-7 based multimedia ontologies. (2009) 331–370

A.4 Appendix on Model-driven Engineering Approach for Programming Access to Ontologies

OntoMDE: Model-driven Engineering of Programming Access to Ontologies

Stefan Scheglmann and Ansgar Scherp and Steffen Staab
Institute for Web Science and Technologies
University of Koblenz-Landau, Germany
{schegi, scherp, staab}@uni-koblenz.de

Abstract

Using ontologies in object-oriented applications is a challenging task. This is due to the chasm that exists between the logics-based world of ontologies and the world of object-oriented software applications. However, using ontologies in software systems for knowledge representation and reasoning can be very useful and extend today's abilities of software systems. To facilitate use of ontologies in object-oriented software systems, a proper mapping of the logical concepts to the object-oriented application is needed. Such a mapping has to overcome the fundamental differences between the semantics described in the ontology and the pragmatics, i.e., class structure, functionalities, and behavior needed in the software system. In this paper, we present a multi-step mapping approach based on model-driven engineering (MDE) to overcome this gap and to provide programming access to ontologies in software systems.

1 Introduction

Ontologies provide powerful means for knowledge representation and reasoning and are useful for various application domains. However, using ontologies in object-oriented software systems is a challenging task due to the differences between the logics-based world of ontologies and the object-oriented world.

For data sources such as relational databases (RDB) and Web Services, a popular strategy accessing data from an application is using Data Access Objects (DAOs). These DAOs abstract and encapsulate the access to the data and provide interfaces to the application developer. In the RDBs, the Active Record Pattern [2] is a widespread pattern implementing the DOA strategy. Due to Fowler, the inventor of the Active Record Pattern, an Active Record is an "object that wraps a row in a database table or view, encapsulates the access and adds domain logic on that data[2]". This object provides an abstraction of the query language used to retrieve the data.

For accessing ontological knowledge from object-oriented software systems, there are similar solutions like ActiveRDF [7] and Jastor[1]. However, these tools are limited with respect to their functionality. The existing approaches typically provide only a one-to-one mapping of the concepts and properties of the ontology to classes and fields of an object-oriented implementation. Often, not all concepts and relations are necessarily useful for the application developer. Logical abstractions that ease the use of ontological knowledge are not provided. It remains unclear how the ontological knowledge has to be used, i.e., in which order the different classes have to be instantiated and the methods provided to be called. Thus, existing approaches do not provide a reasonable mapping of the ontological knowledge to the object-oriented world due to a lack of understanding how the ontology should be used in a concrete practical setting.

Pragmatics is the field that studies the ways in which context contributes to meaning [1]. In our situation, the context is the object-oriented software application and we want to understand how it influences and contributes to the meaning of the ontology. The goal is to understand how the ontology should be used within the context of a concrete application. In this paper, we present an approach to define the pragmatics needed to access and update ontological knowledge from object-orient software systems. We apply a model-driven engineering (MDE) approach that conducts the mapping from ontologies to the software in multiple steps. This approach takes into account the characteristics of ontologies such as concept and property relations, subsumption, import of ontologies, and also allows integrating legacy programming interfaces.

The remainder of the paper is organized as follows: In the following section, we define our terminology and introduce a scenario and running example. In Section 3, we introduce some related work and show how our approach differs from it. In Section 4, we define the requirements to a programming access to ontologies and to our MDE process. Based on these requirements, we describe our approach in Section 5. In Section 6, we provide information about our implementation and the technologies used, before we conclude the paper.

2 Scenario and Example Ontology

In this section, we present a scenario to motivate our work and use it as run-through example for subsequent sections. Basing on this scenario, we present an excerpt of an ontology as input to our process. As output, we present a programming access to the ontology in form of an API.

To be able to explain the coherences between the semantics in the ontology and the pragmatics of the API, we need a distinct terminology. We introduce this terminology here in order to make use of it in the scenario and example. Our terminology consists of four terms:

[1] `http://jastor.sourceforge.net/` last visit December 6, 2010

- A ***Semantic Unit*** is a triple $SU = (CO, SO, R)$ that consists of a set of *content concepts* CO, a set of *structure concepts* SO and a set of relations R between these concepts. An ontology can include multiple *semantic units*. Each *semantic unit* groups all concepts and relations manipulated by a aspect of the context of the application, a pragmatic.
- A ***Pragmatic Unit*** groups all entities of a software system appurtenant to a specific semantic unit SU. It is represented through a tuple $PU = (C, P)$ consisting of a set of classes C and properties P.
- ***Content Concepts*** are those concepts of a *semantic unit*, which provide the *content* the underlying pragmatic works on. Subsequently, we call *content* those entities an application, user or developer works on. *Content concepts* could be used in multiple *semantic units*.
- ***Structure Concepts*** are all concepts of a *semantic unit* not providing content but are of structural matter for the knowledge representation. *Structure concepts* are specific to a single *semantic unit*.

2.1 Scenario: An Ontological Multimedia Annotation Framework

Jim works for a multimedia company and is responsible for the integration of knowledge-base access in a media annotation framework. The media annotation framework should support the user in decomposing and annotating multimedia content such as multimedia presentations and video clips for the web. Jim shall use an ontology for representing multimedia metadata. He has not been involved in the design of the ontology. His task is to define the pragmatics, i.e., how to access and update the knowledge base from the application. He has to consider that further specializations toward domain specific requirements could result in changes of the implementation.

2.2 Example: Ontology-based Modeling of Multimedia Metadata

Figure 1a shows an excerpt of the ontology used by Jim to model the multimedia metadata. It models the decomposition of a multimedia Presentation into Images. The images are annotated with an EXIF[2] Geo-Point by a specific Device. Figure 1 shows also further concepts such as CompositeConcept classifying the Presentation as composite and ComponentConcept that classifying the Image as component in the decomposition.

In Figure 1b, we show the inheritance structure of some of the *content concepts*. As we can see from the different namespaces, the m3o:Presentation,

[2] `http://www.exif.org/` last visit December 6, 2010

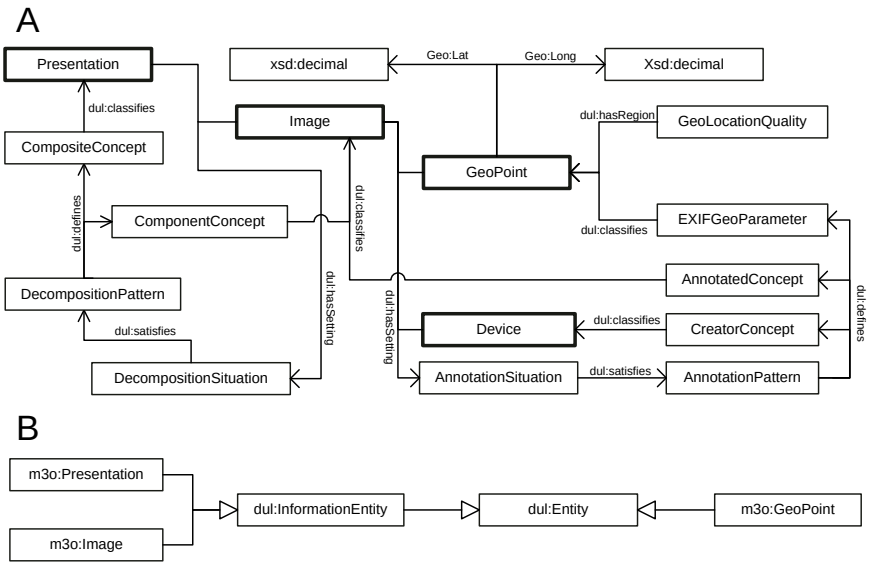


Figure 1: **Ontology-based Annotation and Decomposition of a Multimedia Presentation**

m3o:Image and m3o:GeoPoint concepts and their superconcepts dul:InformationEntity and dul:Entity are defined in different ontologies. The inheritance and import relationships shown in this figure are important for a proper API representation.

The example is based on our Multimedia Metadata Ontology (M3O) [11] for representing annotation, decomposition, and provenance information of multimedia data. All of the M3O concepts inherit from the DOLCE+DnS Ultralite [5] ontology. With the M3O, not only the Presentation can be decomposed into Images, but also an Image can be decomposed into segments. In addition, besides the annotation of a single Image, also the segments as well as the Presentation itself can be annotated. Finally, by attaching information about the creator of an annotation such as Device for Image, the ontology also provides for modeling provenance information.

2.3 Pragmatics of the Ontology

In order to use the ontology in the application, Jim has to define the relations between the pragmatics of the software system and knowledge schema of the ontology. In our example, Jim identifies annotation and decomposition as the main pragmatics provided by the application. Based on these pragmatics, Jim is now able to identify the corresponding semantic units in the ontology. For example in case of annotation the *semantic unit* includes all concepts, AnnotationDe-

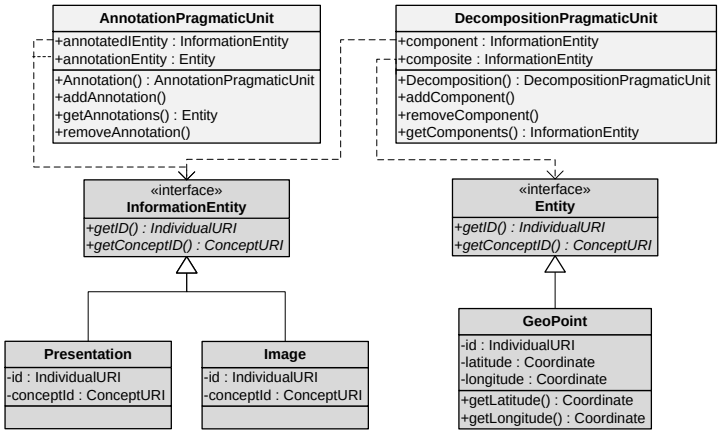


Figure 2: **Model of an API for the Running Example**

scription, AnnotationSituation, AnnotatedConcept, AnnotationConcept, the Image as annotated entity and the GeoPoint as annotation entity. Subsequently, Jim classifies the concepts of the *semantic units* in *content concepts* and *structure concepts*. In case of the annotation *semantic unit* Image and GeoPoint provide the content and AnnotationDescription, AnnotationSituation, AnnotatedConcept and AnnotationConcept the structure. Jim wants to hide the *structure concepts* away from the application developers to reduce complexity.

For the implementation, Jim defines a class for each of the two pragmatics Annotation and Decomposition, as shown in Figure 2. In addition, he defines a class for each content concept the application works on, in this case Presentation, Image and GeoPoint. To minimize the repercussions of new domain specific specializations, Jim decides to build an architecture that allows direct access to the ontology content representations and functionalities while encapsulating the rest of the ontology behind interfaces that retain unchanged. For this reason, Jim also implements interfaces representing the least common subsumer of multimedia content and metadata in the inheritance hierarchy of the ontology, Figure 1*b*. This least common subsumer interfaces are the InformationEntity interface implemented by Presentation and Image and the AnnotationEntity interface implemented by the GeoPoint class. These interfaces support the pragmatic use of arbitrary multimedia content for annotation and decomposition.

3 Related Work

The idea of generating code from an ontology definition is not new. Multiple frameworks like ActiveRDF [7], AliBaba[3], OWL2Java [4], Jena/Jastor[4], OntoJava[5], and others were developed in the past. An overview can be found at Tripresso[6], a project web site on mapping RDF to the object-oriented world. These frameworks use a simple mapping model transforming each concept of the ontology into a DAO class of a specific programming language like Java or Ruby. Properties are mapped to fields. For the ontology in our example, these frameworks create DAOs for each of the concepts, e.g., a DOA for Image as well as for DecompositionSituation. The hasSetting relationships between these two concepts will be represented as a hasSetting field of type DecompositionSituation in the Image class. Unlike in Figure 2, the result of such an approach did not provide dedicated DOAs for the pragmatics, like annotation and decomposition.

Only Àgogo [8] goes a different way. Àgogo is a programming language independent model driven approach for automatically generating ontology APIs. It introduces an intermediate step based on a Domain Specific Language (DSL). This DSL captures domain concepts necessary to map ontologies to an object-oriented representations.

Our approach bases on the experiences of the API generation frameworks mentioned above. We use the mappings proposed in the literature such as [6, 4] and apply a model-driven approach like Àgogo. But to be able to generate specific APIs for multiple application scenarios, the application developer must have full control and customization facilities on all levels of the generation process. To achieve this, our approach suggest to introduce two different intermediate step one for control and customization on ontology level and one for the same tasks on API level. This is needed in order to leverage legacy APIs.

4 Requirements for Programming Access to Ontologies

In this section, we analyze the requirements for the generation of programming access to ontologies. The requirements may be distinguished into requirements to the programming access, typically in form of an API, and the therefrom derived requirements to the generation process of this API.

[3] `http://www.openrdf.org/doc/alibaba/2.0-alpha4/` last visit December 6, 2010
[4] `http://jastor.sourceforge.net/` last visit December 6, 2010
[5] `http://www.aifb.uni-karlsruhe.de/WBS/aeb/ontojava/` last visit December 10, 2010
[6] `http://semanticweb.org/wiki/Tripresso` last visit June 12, 2010

4.1 Requirements to the Pragmatic Programming Access

(R1) Concept Representations

Programming access for ontologies using Data Access Objects (DAOs) has to represent the instances of the ontology concepts as DAOs. For this reason, we have to provide DAO class representations for the ontology. The most naive way, realized by the several frameworks mentioned in the related work section, is to map each ontology concept to a DAO class representation. Such an approach neglects the fact that only some of the concepts provide content for an application, like those Jim identifies in the first step of 2.3. The rest of the concepts are structural, they do not provide content to the application. A programming access should provide direct access to concepts providing content, while ensconcing and encapsulating structure concepts. Form an application point of view, structure concepts are needed to ensure proper serialization. For the encapsulation, we need to classify all concepts, we want to map as content concepts or structure concepts.

(R2) Property Representations

Most of the frameworks using concepts-to-class mapping represent inter-concept relationships as fields of the corresponding concept classes. In programming access to ontologies often complex relationship structures have to be instantiated at once. For example, Jim wants to provide an annotation functionality, instantiating representations for all concepts and relations involved in the declaration of an annotation. A simple field representation for properties in combination with concept-to-class mapping could lead to problems when working on complex semantic units. Often such *semantic units* include cyclic dependencies. These dependencies make secured instantiation impossible without further control structures.

(R3) Inheritance Structure

Inheritance relationships that classify ontology concepts into sub-concepts and super-concepts are of special interest for the programming access. As we have seen in Section 2.3, Jim has implemented additional interfaces for the least common subsumers of the multimedia contents and the metadata entities. These interfaces allow us to apply basic functionality to multiple content classes, e.g., any multimedia content that is of type InformationEntity could be annotated or decomposed. Thus, for generating programming access to ontologies, we need information how to abstract a lean and useful inheritance structure from the ontology. We need to know which concepts of the ontology inheritance structure have to be represented in the API.

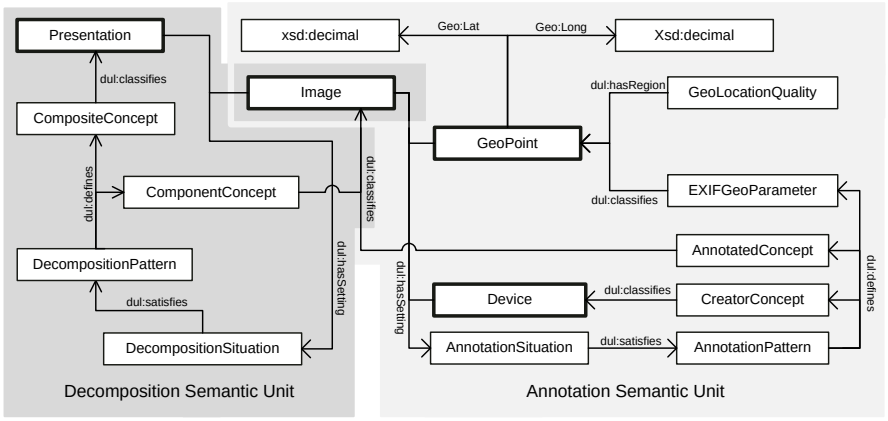


Figure 3: **Semantic Units in the Running Example**

(R4) Pragmatic Units

In order to provide pragmatic units in the API, we need to know to which semantic unit it refers to. A programming access should provide functionalities to the application developer to access and modify the structured knowledge. To provide useful functionality, we have to be able to identify the underlying concept-relation sets relevant for the different pragmatics. In the example decribed in Section 2.2, we found two semantic units, for decomposition and annotation.

(R5) Method Behavior

In our example, Jim's API provides the functionalities for annotating and decomposing multimedia presentations. Jim wants to ensure that each decomposition has at least two known components. For our example, there have to be at least two components of type Image for the presentation decomposition. Thus, when calling methods, like addComponent(...) and removeComponent(...) on the Decomposition class, we need to ensure that this constraint is fulfilled. If this constraint is not fulfilled, an API should raise an exception and notify the application about the inconsistency. This behavior of the API is specified in the so-called method contracts.

However, the logic world of ontologies with its open world assumption does not require the existence of at least one component individual. Here, the context provided by the application as mentioned in Section 1, contributes to the meaning (semantics) of the ontology. In the case described above, it is more restrictive than the semantics of the ontology. We can distinguish between three possible variants how to deal with ontology restrictions.

(A)

In the first case like described above, the behavior of the **API is more restrictive** in terms of ensuring consistency of the represented information than the ontology. It is not possible to create a decomposition with less than the minimum number of components using the API whereas this does not cause any inconsistencies for the ontological knowledge representation.

(B)

In some cases, it is also possible that the **ontology is more restrictive** than the API. This is the case when the API cannot ensure consistency and needs to fall back to a reasoner running in the background to decide whether a specific operation is allowed or not. For example, there might be an API for modeling the configuration of a car's equipment. Depending on the different extras such as air conditioning and radio, the car has to be equipped with a different generator. Here, it must be possible to create different configurations using the API, i.e., a specific selection of extras and generator. Then, the reasoner is applied to determine whether the given configuration is a valid one. Thus, in this case it is possible to create invalid car configurations using the API and determining this invalidity only by using the reasoner.

(C)

In the third case, the **API always calls a reasoner** for each operation it performs. Whenever the application calls one of the APIs functionalities, the reasoner is used to check the validity of the operation. Based on the result the reasoner computes, the API adapts its behavior and either raises an exception or not. This behavior should also be configurable by the application developer.

These differences in the behavior of the ontology and the API need to be considered to specify the API method behavior. In the remainder of this paper, we consider the first case where the API is more restrictive than the ontology. The other two options remain for futurework.

4.2 Requirements to the MDE generation Process of Programming Access to Ontologies

A model-driven generation process for programming access has to generate as much as possible automatically. Where this is not possible it relies on the developer in defining additional conditions. Finally the process has to be adapted to these condition to generate proper programming access.

(R6) Semi-automatic generation

Often the classification of concepts or their assignment to semantic units are driven by the context of the concrete application or by user decision. The process has to provide different functionalities for the user to support him in influencing the generation process and the structure of the generated code.

(R7) Import

The generation process, has to deal with import the instructions in the ontology documents. Often ontologies specialize or make direct use of concepts defined in other ontologies. For example, all concepts from the M3O and thus from our example inherit from concepts defined in the DOLCE+DnS Ultralite [5] ontology. A generation process has to manage all these imports and decide which are important for the generation process.

(R8) De-Anonymization of Concepts

A concept specification may consist of a number of anonymous or named concepts and unary or binary operators. Due to the lack of support for anonymity in object-oriented environments problems arise when we try to map anonymous concepts. So, we constrain that only named concepts are allowed in ontologies our process should be performed on. For most of the ontologies this is not the case. So we have to de-anonymize the ontologies first.

(R9) Legacy API integration

Often applications or frameworks provide classes for their basic content, e.g., in our example the presentation, images or metadata. Let us assume Jims application already uses an particular image class from a legacy API. In this case, Jim does not want to implement a completely new image class to represent the Image concept from the ontology. Instead, he wants to embed the ontology access functionalities into the image class provided by the legacy API. For such a case, we need mechanisms to integrate the ontology access functionalities into classes from arbitrary legacy APIs.

(R10) User-driven Customizations

The API generation process has to provide user-driven control mechanisms. The user wants to create an API for his personal needs while considering the semantics of the ontology. For nearly all the requirements in Section 4.1, we defined a limited leeway, in which the user of the process has to decide how the mapping should be conducted. For example, for the mapping of an inheritance structure to an API or for customizations in concept classification and assignment, we

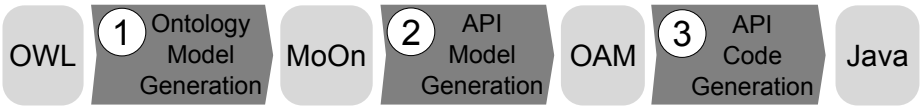


Figure 4: **The API Generation Process**

need decision mechanisms. These mechanisms should give the user the opportunity to decide, e.g., which elements in a concrete inheritance structure should be mapped to the inheritance structure of the API.

(R11) Multi-step generation process

To influence concept classification or the definition of semantic units defined in Section 4.1, the developer needs access to the generation process on ontology level. For user-driven customizations like in R10 and to influence the structure of the generated code, the developer needs access on the level of code generation. To achieve this, we need intermediate steps in the generation on which these customizations could be performed.

5 Approach: Model-driven Engineering of Programming Access and Update of Ontologies

In this section, we present the Model-driven Engineering (MDE) approach for the generation of programming access to ontologies. To fulfill the different requirements defined in the previous section, we implemented our approach as a multi step process (R11). Figure 4 depicts the different steps of our process. Our mapping includes three different transformations and two intermediate models, before it finally ends up in API Code.

In the first step the ontology is transformed in a Model for Ontologies (MoOn) representation. The MoOn bases on a modification of the ECore Metamodel for OWL2[7] and the OWL-to-UML mappings described in [4, 3, 9]. We add some additional annotations to the MoOn to be able to represent information, e.g., about the concept classifications or the semantic unit specification.

The second step is the transformation from the MoOn to the Ontology API Model (OAM). The OAM is a UML class diagram of the programming access API. We extended this model to embed information for the code generation process, e.g., to tailor the concrete API to a particular repository backend. The last step is the generation of code from the OAM.

[7]MOF-Based Metamodel for OWL2 `http://www.w3.org/2007/OWL/wiki/MOF-Based_Metamodel`

5.1 Step 1: From Ontology T-Box to MoOn

In this section, we present the first intermediate model the MoOn and the transformation from an OWL-based ontology definition to MoOn. The MoOn is independent from the concrete DL language used, only the transformation is OWL2 specific.

To map the single OWL constructs and their properties, we have to perform several preparation steps, these are:

- Ontologies often specialize or use concepts and properties defined in other ontologies. We have to deal with these **imports** of the ontology, see requirement R7. To generate a MoOn for further transformations, we enrich our ontology representation with imported concepts and properties directly used in our target ontology.

- Ontologies often provide particular knowledge only in an implicit form, e.g., subsumption. We have to **inference** this implicit knowledge and make it explicit, in order to be able to map it into the MoOn. We use reasoning services to inference this implicit knowledge.

- A very important step in the preparation of the concept and property mapping, is to **de-anonymize** the constructs in the ontology (R8). Due to the fact that object orientation does not provide for anonymous classes, we have to avoid anonymous concepts. In DL based languages as OWL, we can easily substitute anonymous concepts with named ones. Let us assume a concept specification in the form $A \equiv (B \sqcap C) \sqcup (D \sqcap E)$. In this definition $(B \sqcap C)$ and $(D \sqcap E)$ are anonymous concepts. We de-anonymize them by introducing two new concepts $BandC \equiv B \sqcap C$ and $DandE \equiv D \sqcap E$ and substitution in the original concept specification, to get $A \equiv BandC \sqcup DandE$.

- Now that we have enriched the ontology representation with all relevant concepts and properties, we are able to create an **inheritance** structure suitable for an object oriented representation (R3). In OWL ontologies, we often find very complex inheritance structures, always ending up in Thing. In an object-oriented representation, we need a much simpler inheritance structure to support broader type declarations for constructors and methods. To adapt the inheritance structure in the MoOn to our needs, we search for least common subsumers of multiple concepts in the set of concept relevant for the MoOn. We add such subsuming concepts to this set of concepts relevant for the MoOn.

After these steps, we can transform all the relevant concepts and properties to the MoOn. We use the mappings defined in [4, 3, 9]. We integrate a mapping for number restrictions as cardinalities. When the mapping is finished, we are able to enrich this ontology model with additional information relevant for the

mapping to object-orientation. For example, information about the definition of the *semantic units* and the classification of the concepts in *structure concepts* and *content concepts*, see Section 2.3.

We start with the *semantic unit* definition (R4). With defined *semantic units* it is possible to classify the concepts in a semi-automatic process. To encapsulate concepts of a particular *semantic unit*, we use the package mechanism of ECore. Currently, we only support user-driven *semantic unit* specification. For our example, Jim defines the two *Semantic Units* annotation and decomposition. Figure 3 shows the affiliation of the single classes to the different *semantic units*.

The next step is to classify the concepts into *structure concept* or *content concepts* (R1). We use ECore annotations to declare the classification of a concept in the MoOn. This step could be performed semi-automatically by preselecting all concepts involved in different *semantic units* as *content concepts*. For our example all concepts inheriting from InformationEntity or Entity itself are *content concepts*. *Structure concepts* are only involved in a single *semantic unit*. For the annotation *semantic unit* in our example, the concepts AnnotationPattern, AnnotationDescription, AnnotatedConcept and AnnotationConcept are *structure concepts*. This mechanism performs well for most of the complex ontologies like the M3O as well as for less complex ontologies. In less complex ontologies, like the Multimedia Metadata Ontology of the W3C Media Annotations Working Group nearly all concepts provide content and we find less *structure concepts*. We use a user-driven refinement of the concept classification to support such ontologies properly.

The MoOn provides for further customizations of the previous steps, (R6, R9). Based on the MoOn, it is possible to integrate additional concepts in the inheritance structure whether if they are from the ontology or imported. It is possible to delete single concepts or superconcepts from a *semantic unit* or from the MoOn.

5.2 Step 2: From MoOn to OAM

In this section, we present the second intermediate model, the OAM and the transformation from MoOn to OAM.

The OAM is an object-oriented model of the resulting API. The API will be generated from the OAM in the subsequent step. All special characteristics and properties of the intended API are defined in this model (R2, R5). The UML2 class diagram is the underlying metamodel of the OAM. We extended this metamodel with light-weighted profiles. In the profiles, we define stereotypes that could be attached to different entities in the model. These stereotypes provide additional information used during code generation.

[7] `http://mawg.joanneum.at/web/mawg.html`

[7] `http://www.omg.org/technology/documents/profile_catalog.htm`

First, we generate interfaces from the inheritance structure in the MoOn (R3). The basic content an application works on will get its own class representations in the API. We add a class for each *content concept* implementing the relevant interfaces created in the previous step.

We generate a *pragmatic unit* class for each *semantic units* defined in the MoOn. For each *structure concepts*, we add an attribute to the pragmatic unit class it belongs to (R2). Figure 2, shows the generated classes for our running example. The pragmatic unit classes encapsulates the structure concepts and hides them from the API user. For each *content concept* in a *semantic unit*, we add an attribute of the type of the *content concept* class. To the added attributes, we attach the cardinalities from the related MoOn properties. This enables us to support dedicated method behavior in the code generation step. For example, a removeComponent(...) method that ensures the existence of at least two components in a decomposition (R5).

Table 1 summarizes the different mappings between the Ontology and the final API for the *content* and *structure concepts* as well as for the *semantic units*

Table 1: **Overview of Mappings between Ontology and API**

Ontology	**API**
Content concepts	Content classes & pragmatic unit class attribute
Content individuals	Content objects
Structure concepts	Class attributes
Structure individual	Individual URI and concept URI
Semantic unit	Pragmatic unit class

As we have mentioned in the requirement R9, we also want to be able to integrate classes from arbitrary legacy APIs and to embed our ontology access functionalities into these classes. The OAM provides ideal conditions to perform these tasks. In the OAM, it is possible to model classes from legacy APIs. A user do not has to model the whole class with all of its properties and operations, just a prototype of the class with a special stereotype is sufficient. So that our new class can inherit from this modeled legacy API class model. Let us assume the application Jim contributes to uses AWT[8] image class. Then Jim has to model a simple class named Image stereotyped with the concrete package name *java.awt*. The generated image class inherits from the AWT image class while implementing all interfaces for ontology access. So, Jim has a class completely compatible with his application embedding all ontology access functionalities.

[8] `http://java.sun.com/products/jdk/awt/`

[8] `http://download.oracle.com/javase/1.4.2/docs/api/java/awt/Image.html`

5.3 Step 3: Generating an API from the OAM

In the last step, we generate code from the API representation in the OAM. We can use standard code generation functionalities for arbitrary programming languages. Customizations of the out-coming code towards a persistence layer implies that we have computed the necessary meta information in a previous step. Once this has be done, we could generate a fully functional API for pragmatic programming access to the dedicated ontology.

6 Implementation

To demonstrate the feasibility of our process, we have implemented it for the generation of Java-based APIs. In our implementation, we generate the meta-information required for the WINTER persistence layer. It is easy to adopt this step to nearly every existing persistence layers, like ActiveRDF [7] or Jastor[9].

The implementation of our model-driven API generation process uses a set of plug-ins for the Eclipse software development toolkit[10]. A first plug-in loads the OWL-based ontology and represents it using MoOn. All operation on the OWL ontology are implemented using the OWL API[11]. The MoOn representation leverages the Model Development Tools (MDT) which is part of the Eclipse Modeling Framework Project (EMF)[12]. This framework provides, an implementation of the Meta Object Facility (MOF) [13] ECore, to support the development of modeling tools. The MDT provides a graphical model representation and manipulation framework for ECore, the Graphical Editing Framework (GEF)[14] and the Graphical Modeling Project (GMP)[15]. The EMF also provides facilities for building code from a structured data model. A second plugin realizes the transformation from MoOn to OAM. Based on the OAM a third plugin realizes the code generation using Jet. We have implemented plugins to perform the user-driven customizations on the MoOn and OAM.

In our implementation, we support the generation of Java code that uses the Winter[10] persistence API. For the code generation, we use Java Emitter Templates (JET)[16]. JET is a generic template engine to generate various different textual output such as Java code.

[9] `http://jastor.sourceforge.net/` last visit December 6, 2010
[10] `http://www.eclipse.org/` last visit December 5, 2010
[11] `http://owlapi.sourceforge.net/` last visit December 5, 2010
[12] `http://www.eclipse.org/modeling/emf/` last visit December 5, 2010
[13] `http://www.omg.org/mof/`last visit December 5, 2010
[14] `http://www.eclipse.org/gef/` last visit December 5, 2010
[15] `http://www.eclipse.org/modeling/gmp/` last visit December 5, 2010
[16] `http://www.eclipse.org/modeling/m2t/?project=jet` last visit December 5, 2010

7 Conclusion

We have presented a multi-step model-driven approach to generate application programming interfaces (APIs) from logics-based ontologies. To this end, we first prepare the ontology and transform it to our Model for Ontologies (MoOn), which is based on ECore. The MoOn allows user-driven customizations on the ontology representation to reflect the needs in a specific application context. In a subsequent step, the MoOn is transformed to the Ontology API Model (OAM), which is based on UML. From the OAM, we can directly generate our API code. Unlike other approaches, we do not just perform a naive one-to-one mapping of the ontology concepts and properties to the API classes and fields. Instead, we follow a model-driven approach that provides support for the characteristics of ontologies such as concept and property relations, subsumption, import of ontologies, and also allows integrating legacy programming interfaces. By this,we provide the pragmatics required to use the ontological knowledge in the context of a concrete application. With our approach, we alleviate the developers from the tedious and time-consuming API development task such that they can concentrate on developing the application's functionalities.

For our future work, we plan to provide support for integrating and manipulating T-Box knowledge via the generated API and we want to integrate the support for different method behaviors (see R5). At the moment, we only support behavior for methods where the application context is more restrictive that the ontology.

Acknowledgements This research has been co-funded by the EU in FP7 in the WeKnowIt project (215453).

References

[1] Harry Bunt and Bill Black. The abc of computational pragmatics, 2000.

[2] Martin Fowler. *Patterns of Enterprise Application Architecture*. Addison-Wesley Longman, Amsterdam, 2002.

[3] L. Hart and P. Emery. OWL Full and UML 2.0 Compared. `http://uk.builder.com/whitepapers/0and39026692and60093347p-39001028qand00.htm`, 2004.

[4] Aditya Kalyanpur, Daniel Jiménez Pastor, Steve Battle, and Julian A. Padget. Automatic Mapping of OWL Ontologies into Java. In *SEKE*, 2004.

[5] C. Masolo, S. Borgo, A. Gangemi, N. Guarino, A. Oltramari, and L. Schneider. WonderWeb deliverable D17. the WonderWeb library of foundational ontologies and the DOLCE ontology. Technical report, ISTC-CNR, 2002.

[6] OMG. *Ontology Definition Metamodel*. Object Modeling Group, May 2009. `http://www.omg.org/spec/ODM/1.0/PDF`.

[7] Eyal Oren, Renaud Delbru, Sebastian Gerke, Armin Haller, and Stefan Decker. Activerdf: object-oriented semantic web programming. In *WWW*. ACM, 2007.

[8] Fernando Silva Parreiras, Carsten Saathoff, Tobias Walter, Thomas Franz, and Steffen Staab. 'a gogo: Automatic Generation of Ontology APIs. In *IEEE Int. Conference on Semantic Computing*. IEEE Press, 2009.

[9] Tirdad Rahmani, Daniel Oberle, and Marco Dahms. An adjustable transformation from owl to ecore. In Dorina C. Petriu, Nicolas Rouquette, and ystein Haugen, editors, *MoDELS (2)*, volume 6395 of *Lecture Notes in Computer Science*, pages 243–257. Springer, 2010.

[10] Carsten Saathoff, Stefan Scheglmann, and Simon Schenk. Winter : Mapping RDF to POJOs revisited. In *Poster and Demo Session, ESWC 2009*, Heraklion, Greece, 2009.

[11] Carsten Saathoff and Ansgar Scherp. Unlocking the Semantics of Multimedia Presentations in the Web with the Multimedia Metadata Ontology. In *WWW*. ACM, 2010.