Formal Metadata Semantics for Interoperability in the Audiovisual Media Production Process

Martin Höffernig and Werner Bailer

JOANNEUM RESEARCH Forschungsgesellschaft mbH
Institute of Information Systems
Steyrergasse 17, 8010 Graz, Austria
firstname.lastname@joanneum.at

Abstract. In the different stages of the production process of audiovisual media such as movies, a number of different metadata properties exist. Different metadata formats and standards are used in the stages of the process, containing metadata properties with similar and partly overlapping semantics, though not fully identical. We attempt to model the metadata properties used throughout the production process in a format independent way by creating an ontology that models these properties and the relations between them. Modeling is done on a high level, considering grouping and definition relations between the properties. We apply the proposed approach to the problem of verifying whether a set of metadata properties can be unambiguously derived from another and present a web based demo application for this use case.

1 Introduction

1.1 Motivation

A large number of different metadata properties exist throughout the audiovisual media production process. These properties are produced and consumed at different stages of the process. Typically the different devices and tools used in the chain also make use of different metadata representations. The SMPTE metadata dictionary [11] alone lists nearly 1,500 metadata properties, and there are many more that are not covered by this dictionary [1]. Other relevant standards in the media production process are for example the MXF Descriptive Metadata Scheme 1 [3], MPEG-7 Multimedia Content Description Interface [6] and EBU P_Meta [9]. Current trends such as 3D and multi-view content add additional requirements to the metadata representation, as the relations between different media properties need to be described (from high level information such as relating different views of the same scene down to precise measurements such as camera calibration parameters).

When analyzing the various properties and their definitions in different stages of the process or in different metadata formats, we encounter a number of properties which represent the same information, but modeled differently or only partly overlapping. For example, at capture, a number of parameters are available, such

as the resolution of the sensor, its aspect ratio, the temporal sampling rate, etc. At a later stage, the header of a stream might contain an identifier of the video standard used, which implies the values for a number of these parameters. In order to support content exchange and automation in the production process, it is necessary to establish metadata interoperability between the steps of the process. Due to the multitude of metadata properties and formats, which are often tailored toward the specific needs of a certain step in the process, it is utopian to expect that a single format serving the needs of all steps in the process can be defined, that will also be adopted by all the devices and tools involved. We thus need to deal with the diversity in terms of metadata and establish interoperability by well-defined semantics of the different metadata properties, so that they can be mapped between the different stages of the process.

This work presents a first step in this direction. We aim to model the concepts behind the metadata properties in the process, leaving specifics of formats such as data types aside. These things can be addressed in an additional layer on top. In the remainder of this section we discuss approaches for solving related problems. We then analyze the aspects of this interoperability problem in more detail in Section 2 and present the proposed approach in Section 3. In Section 4 we discuss a prototypical implementation of the approach for an ontology covering a small set of metadata properties and Section 5 concludes the discussion.

1.2 Related Work

In [8] the automation of media production processes by using a workflow management system is discussed. In that work, the open source workflow language YAWL (Yet Another Workflow Language [12]) has been chosen and extended to fit the area of film production (YAWL4Film[1]). YAWL4Film contains workflow patterns that support the production crew in collecting, creating, and distributing required documents and data for certain production tasks. For example, the process for a daily shooting procedure has been modeled, in which documents such as time sheets for cast members or the daily-shooting progress reports are created and distributed automatically by the workflow system. YAWL is based on XML technologies and all the data being processed during the process steps is defined using custom XML Schemata. Due to known limitations of XML Schema describing semantics [7], interoperability to existing metadata standards in the media production process is very limited.

In the different stages of the media production process, different types of metadata are needed, such as descriptive, technical, structural, composition, and editing metadata. In [10] these metadata types are listed and allocated to the concerning production stages. Furthermore, relevant metadata standards, for describing these different types of metadata have been identified. In [1] the requirements for a metadata model for audiovisual media production have been developed and discussed, and existing standards have been analyzed. The conclusion of this work is that none of the current metadata standards is able to

[1] http://www.yawl4film.com/

achieve all the defined requirements. However, interoperability between metadata standards needs to be enabled in order to exchange metadata properties between different standards.

The Simple Knowledge Organization System Reference (SKOS) provides a vocabulary to classify concepts and to describe how they relate to others concepts [5]. Semantic relations, such as narrower, broader, and related are available for describing relations between SKOS concepts. It is of course possible to build up a classification scheme using the SKOS relations. However, we want to model more complex relations, for example, if a concept defines other concepts or if a concept can be substituted by others. Describing such relations are out of the scope of SKOS[2].

The W3C Media Annotations Working Group[3] also deals with the interoperability issue of metadata formats. Their goal is to develop a simple ontology of core metadata properties for audiovisual content and an API for accessing these properties from descriptions in a range of formats. This clearly needs mappings between the considered formats and the proposed set of properties. The working draft containing the core vocabulary to describe media resources is available at [4]. In contrast to this work we do not want to define mappings between different metadata standards, but rather to describe semantic relations between format independent metadata properties.

2 Problem Definition

Different metadata properties represented in different metadata formats exist in the media production process. However, we encounter a number of properties which represent – at least partially – the same information, but modeled differently or with only partly overlapping semantics. In order to support content exchange and automation in the production process, it is necessary to establish metadata interoperability between different metadata models and representations being used in the steps of the process. As first step to solve this interoperability issue we model the relations between metadata properties or groups of metadata properties. We then define a set of queries that our system needs to be able to answer. These queries yield information about the how metadata properties in the different stages of the process are related, they do not yet implement conversion between metadata formats.

2.1 Relations

Definition A metadata property (or group of properties) A defines another metadata property (or group of properties) B, if B can be derived without any semantic ambiguity from A by some mapping/conversion.

[2] Compare the discussion about the usage of SKOS mappings for this purpose: http://lists.w3.org/Archives/Public/public-media-annotation/2009Mar/0067.html

[3] http://www.w3.org/2008/WebVideo/Annotations/

Equivalence A metadata property (or group of properties) B is equivalent to another metadata property (or group of properties) B, if A defines B and B defines A.

At this point it is not relevant which specific metadata formats are used and what kind of data type is used to represent a specific metadata property. It is only important to model the concept represented by a metadata property and the relations between them.

In this paper we use the following notation. To formally express a group of metadata properties, the conjunction ($\wedge$) is employed. Additionally, the implication operator ($\rightarrow$) is used to express the definition relation between metadata properties (or group of) properties, and the equivalence operator ($\leftrightarrow$) describes the equivalence relation between metadata properties (or group of) properties.

As an example, assume that there are the following metadata properties of a video: number of lines, number of columns, spatial resolution, frame rate, and a video payload identifier of a container file format, that describes the video standard by an identifier[4]. Spatial resolution is equivalent to a metadata properties group containing lines and columns. Furthermore, the payload identifier defines lines, columns, and frame rate. These two statements can be formally expressed as equations 1 and 2. Since spatial resolution is equivalent to the group of lines and columns, the payload identifier also defines the resolution which is expressed in equation 3.

$$resolution \leftrightarrow lines \wedge columns. \tag{1}$$

$$payload\ identifier \rightarrow lines \wedge columns \wedge frame\ rate. \tag{2}$$

$$\begin{aligned}(payload\ identifier \rightarrow\ & lines \wedge columns \wedge frame\ rate) \wedge \\ & (lines \wedge columns \leftrightarrow resolution) \rightarrow \\ & (payload\ identifier \rightarrow resolution)\end{aligned} \tag{3}$$

2.2 Queries

Expressing definition and equivalence relations between metadata properties enables to infer information about interoperability between metadata properties. The following three types of queries have been identified[5]. It holds for all types of queries, that some queries may be answerable directly by the facts represented in the ontology while others need inference.

1. Verify whether a given metadata property is defined by another given metadata property or not. For example, it should be verified if there exists an definition relation between payload identifier and resolution (equation 4). Since there is an inferred definition the response to this query is yes.

[4] Such an property exists e.g. in the header of an MXF file.

[5] Note that in the description of the queries "metadata property" stands for single metadata properties as well as groups of metadata properties.

$$payload\ identifier \rightarrow resolution\ ? \tag{4}$$

2. Find all metadata properties that are defined by a given metadata property. An example is the query in equation 5. Lines, columns, and frame rate are direct results, while resolution is an inferred result to this query.

$$payload\ identifier \rightarrow ? \tag{5}$$

3. Find all metadata properties that define a given metadata property. As an example all metadata properties which imply lines should be listed (cf. equation 6). In this case, payload identifier and resolution are the results.

$$?\ \rightarrow lines. \tag{6}$$

To simplify matters, in all of the examples above only single metadata properties have been used as query parameters. In addition, only single metadata properties are listed as results. However, the result set could be expanded to include groups of properties, e.g. the group (lines, columns) in addition to resolution. When dealing with groups of metadata properties we can distinguish two types: those explicitly defined in our ontology and those not explicitly defined in the ontology but stated in the query or emerging from the result. For example, equation 7 contains an explicitly defined group of metadata properties since this group has been explicitly expressed in equation 1. On the other hand, the metadata group contained in equation 8 is only created in the query.

$$payload\ identifier \rightarrow (lines \wedge columns) \tag{7}$$

$$payload\ identifier \rightarrow (resolution \wedge frame\ rate) \tag{8}$$

Additionally, query results can also contain groups of metadata properties that are not explicitly defined. For example, valid results of the query in equation 5 are among others (lines $\wedge$ columns) and (lines $\wedge$ resolution).

3 Proposed Approach

In this section we propose an approach for expressing the required relations between metadata properties (as discussed in Section 2). As a proof of concept the approach is applied to solve the verification query task (cf. equation 4). We propose the use of an ontology which is called *meon*[6] for the formal representation of metadata properties and the relations between them. Furthermore, logical rules are applied to infer new knowledge.

OWL-DL [2], which is a subset of the Web Ontology Language, is used to formally capture the semantics of the metadata properties and their relations. The class `Concept` models the general concept represented by a metadata

[6] prefix `meon`: http://www.20203dmedia.eu/meon#

property in the ontology. Subclasses of class `Concept` are `AtomicConcept` and `CompoundConcept`. Class `AtomicConcept` represents all single metadata properties, while class `CompoundConcept` describes groups of metadata properties containing at least two metadata properties. Property `contains` describes that an instance of class `Concept` is part of a group of metadata properties (i.e. part of an instance of class `CompoundConcept`). In order to model the definition relation between two metadata properties (or group of), the transitive property `defines` is used. This property can be applied between instances of class `Concept`. Since the equivalence relation is just the result of a bidirectional definition relation, an appropriate usage of this property is expressive enough to model also the equivalence relation. The metadata properties in the example presented in Section 2 are represented by the ontology in Figure 1[7]. In this ontology `PayloadIdentifier`, `FrameRate`, `Lines`, `Columns`, and `Resolution` are instances of class `AtomicConcept`, `CC_1` and `CC_2` are anonymous instances of class `CompoundConcept`. `CC_1` represents a group of metadata properties containing frame rate, lines, and columns. Additionally the metadata properties lines and columns form another metadata group described by `CC_2`.

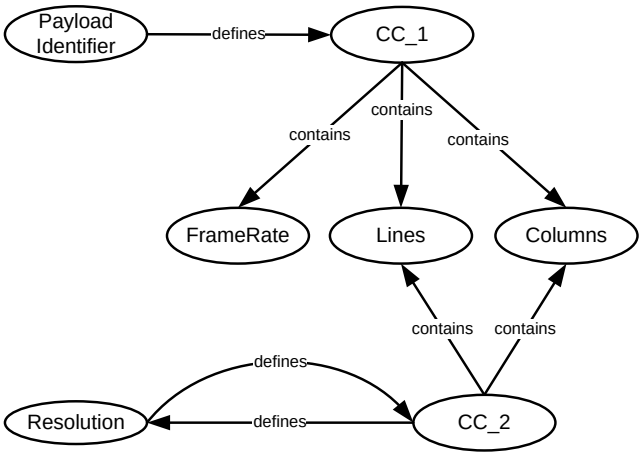

Fig. 1. Example of an ontology for describing metadata properties and their relations between.

In order to model the verification query task (cf. equation 4) as a proof of concept of the proposed approach, we split groups of metadata properties into all possible combinations and then infer new relations in the ontology. Therefore logical rules have been created to semantically express the knowledge required to solve this task. These rules make implicit knowledge in the ontology explicit by adding new instances and relations which enable the subsequent rea-

[7] For the sake of simplicity, the `meon` namespace and `rdf:type` relations have been omitted.

soning and querying steps. The Jena rules syntax[8] is used for defining the rules. One set of rules is responsible to split metadata groups into their combinations and to establish definition relations between the parent metadata group and they newly created ones. Another set of rules expresses the equivalence between metadata groups. An important prerequisite for determining the equivalence between metadata groups is that it must be possible to express the number of metadata properties contained in the group. Therefore an additional property (`countContains`) is added to the ontology, and a rule containing a custom procedural builtin[9] is used to compute the number of metadata properties for each metadata properties group. An example rule for expressing the equivalence relation between two metadata groups containing two metadata properties is shown in Figure 2. First, two ambiguous instances of class `CompoundConcept` (`?cc1` and `?cc2`) containing exactly two instances of class `AtomicConcept` are identified. Then it is verified whether in `?cc1` and `?cc2` the same instances of class `AtomicConcept` (`?ac1` and `?ac2`) are included. In case that (`?cc1` and `?cc2`) are equivalent, two new definition relations between them are added to the ontology. Another rule expresses that the `defines` relation between an instance of class `CompoundConcept` and a instance of class `AtomicConcept` also infers a definition relation (`defines`) between them. It is obvious that a metadata property is equivalent to itself. This fact is also explicitly added by a rule.

```
@prefix meon: <http://www.20203dmedia.eu/meon#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

[equivalence_between_compound_concepts_containing_2_atomic_concepts:
  (?cc1 meon:countContains "2"^^xsd:int),
  (?cc2 meon:countContains "2"^^xsd:int),
  notEqual(?cc1, ?cc2),
  (?cc1 meon:contains ?ac1),
  (?cc2 meon:contains ?ac1),
  (?cc1 meon:contains ?ac2),
  (?cc2 meon:contains ?ac2),
  notEqual(?ac1, ?ac2),
  ->
  (?cc1 meon:defines ?cc2),
  (?cc2 meon:defines ?cc1)
]
```

Fig. 2. Rule for infering the equivalence between two metadata property groups consisting of two metadata properties.

After applying the presented rules to the example ontology depicted in Figure 1, it can be derived that payload identifier defines resolution (cf. equation 3).

[8] http://jena.sourceforge.net/inference/index.html#rules
[9] http://jena.sourceforge.net/inference/index.html#RULEextensions

The extended example ontology after applying the rules is shown in Figure 3. First a new anonymous instance `CC_4` of class `CompoundConcept`, which is a subgroup of `CC_1`, is added to the ontology by the according rule. Additional subgroups of `CC_1` are created as well but for simplicity they are not shown in Figure 3. Then the equivalence between `CC_4` and `CC_2` is computed (cf. rule shown in Figure 2). According to the transitive behavior of property `defines` there is now a definition relation between the instances `PayloadIdentifier` and `Resolution`.

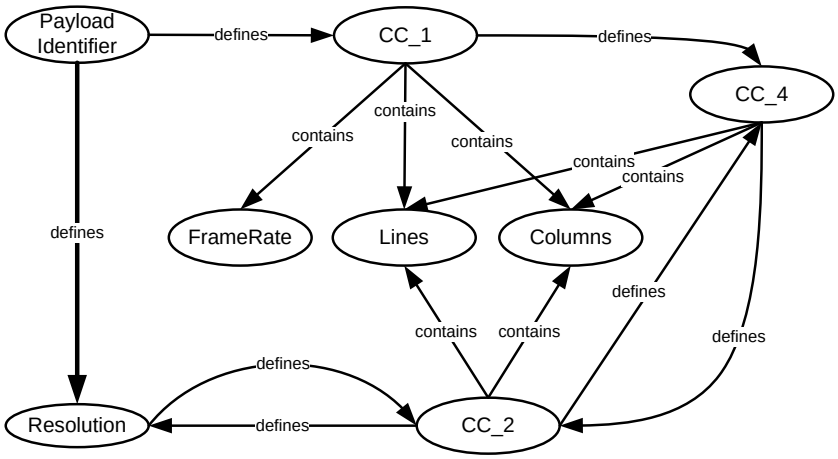

Fig. 3. Extended ontology after applying rules to example ontology (shown in Figure 1) in order to accomplish the verification query task.

4 Demo Application

To demonstrate our approach to solve the verification query task we have implemented a web based demo application. As described in Section 3, the presented Jena rules are applied to the ontology containing the metadata properties. Then new definition relations are inferred. Although transitive reasoning is a basic task for an OWL-DL reasoner[10], this task is also performed by a Jena rule. The reason for this design decision is that it would be the only usage of an OWL-DL reasoner during the whole process, and due to performance considerations this task has been moved to the rule reasoning block. The next step is to perform a SPARQL[11] select query to verify whether there is a definition relation between the selected metadata properties or not. For example, the SPARQL select query shown in Figure 4 is used to query for relations between the metadata properties

[10] e.g. http://clarkparsia.com/pellet
[11] http://www.w3.org/TR/rdf-sparql-query/

payload identifier and resolution. Finally it is checked if the definition relation, represented using the property `defines`, is part of the SPARQL query result in order to determine the result of a verification query task.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX meon: <http://www.20203dmedia.eu/meon#>

SELECT DISTINCT ?property WHERE
{ meon:PayloadIdentifier ?property meon:Resolution }
```

Fig. 4. SPARQL select query to verify equation 4.

The user interface of our web application[12] is shown in Figure 5. The web application has been created using the Google Web Toolkit (GWT)[13] and is deployed on Apache Tomcat 6.0[14]. All OWL processing tasks (including rule reasoning and executing SPARQL queries) are performed using Jena 2.6.0[15]. After loading an ontology all single metadata properties are displayed. The user selects the metadata properties to be mapped and performs the verification.

5 Conclusion and Future Work

We have analyzed the problem of mapping metadata properties between stages of the audiovisual media production process with possible (partly) different semantics. Three basic types of queries have been identified. The proposed approach consists of an ontology modeling simple as well as compound metadata properties as well as their relations. Jena rules are used to infer implicit knowledge about the metadata properties. As a proof of concept, the approach has been successfully applied to the verification query type, i.e. verifying whether a metadata property can be unambiguously derived from another. A web application has been implemented as a demonstrator.

The next step is to also implement the other query types. In particular, groups of metadata properties implicitly defined in the query (but not modeled in the ontology) need to be supported. In addition, the ontology will be extended to cover a wide range of metadata properties used in the audiovisual media production process, as well as their relations. Furthermore, if we also describe the relation between the metadata properties in our ontology and specific formats and consider data type issues, the approach could be useful to enable automatic conversion between metadata formats.

[12] The application can be accessed from http://meon.joanneum.at.
[13] http://code.google.com/webtoolkit/overview.html
[14] http://tomcat.apache.org/index.html
[15] http://jena.sourceforge.net/

Fig. 5. meon web application.

Acknowledgements

The research leading to this paper was partially supported by the European Commission under the IST contract FP7-215475, "2020 3D Media: Spatial Sound and Vision" (http://www.20203dmedia.eu).

References

1. Werner Bailer and Peter Schallauer. Metadata in the audiovisual media production process. In Michael Granitzer, Mathias Lux, and Marc Spaniol, editors, *Multimedia Semantics - The Role of Metadata*, volume 101 of *Studies in Computational Intelligence*, pages 65–84. Springer, Jun. 2008.
2. Mike Dean and Guus Schreiber. OWL Web Ontology Language: Reference. W3C Recommendation, 10 February 2004. http://www.w3.org/TR/owl-ref/.
3. Material Exchange Format (MXF) - Descriptive Metadata Scheme-1. SMPTE 380M, 2004.
4. WonSuk Lee, Tobias Bürger, Felix Sasaki, Véronique Malaisé, and Florian Stegmaier. Ontology for Media Resource 1.0. W3C Working Draft, June 2009. Editor's draft at http://www.w3.org/2008/WebVideo/Annotations/drafts/ontology10/ First-Draft/Overview.html, to appear at http://www.w3.org/TR/mediaont-10.
5. Alistair Miles and Sean Bechhofer. SKOS Simple Knowledge Organization System Reference. W3C Candidate Recommendation, 17 March 2009.

6. Information Technology - Multimedia Content Description Interface (MPEG-7). ISO/IEC 15938, 2001.
7. Frank Nack, Jacco van Ossenbruggen, and Lynda Hardman. That Obscure Object of Desire: Multimedia Metadata on the Web (Part II). *IEEE Multimedia*, 12(1), 2005.
8. Chun Ouyang, Marcello La Rosa, Arthur H.M. ter Hofstede, Marlon Dumas, and Katherine Shortland. Toward web-scale workflows for film production. *IEEE Internet Computing*, 12(5):53–61, 2008.
9. EBU P_META 2.0 Metadata Library. EBU Tech 3295-v2, Jul. 2007.
10. Konstantin Schinas, Wolfgang Schmidt, Franz Höller, Herwig Zeiner, Werner Bailer, and Michael Hausenblas. D3.2.1 Metadata in the Digital Cinema Workflow and its Standards. Public deliverable, IP-RACINE (IST-2-511316-IP), 2005. http://www.ipracine.org/documents/Del_3_2_1_metadata.pdf.
11. Metadata dictionary registry of metadata element descriptions. SMPTE RP210.11, 2004.
12. W. M. P. van der Aalst, L. Aldred, M. Dumas, and Ter A. H. M. Hofstede. Design and implementation of the YAWL system. *Proceedings of the 16th International Conference on Advanced Information Systems Engineering (CAiSE'04)*, 2004.