Ontology-based Indexing and Contextualization of Multimedia Documents for Personal Information Management Applications

Annett Mitschick
Dresden University of Technology, Department of Computer Science
Chair of Multimedia Technology
Dresden, Germany
annett.mitschick@tu-dresden.de

***Abstract*—With the help of Semantic Web technologies, which ensure machine processability and interchangeability, we are able to apply semantic knowledge models to organize and describe heterogeneous multimedia items and their context. However, an ontology-based document management system has to meet a number of challenges regarding flexibility, soundness, and controllability of the semantic data model. This paper presents an integrated approach for ontology-based multimedia document management which covers the process of automated modeling of semantic descriptions for multimedia objects and their long-term maintenance, allowing for the domain-specific customization of the used ontology. Furthermore, the proposed approach addresses the problems of data validation and consolidation to ensure semantic descriptions of proper quality. We demonstrate the practicability of our concept by a prototypical implementation of a service platform for personal information management applications.**

***Keywords*-personal multimedia document management; semantic metadata; generation; maintenance;**

I. Introduction

As digital devices have found their way into nearly all domains of every-day life, the amount of digital multimedia content is increasing and becomes more valuable and important. Managing a considerable quantity of multimedia documents involves administration efforts and certain strategies for ordering and arrangement to keep track of content and structure of the collection – esp. over a long period [1]. The problem is intensified by the complex and partly high-dimensional characteristics of multimedia objects. Problems which appear when users deal with search and retrieval tasks within personal document collections mainly result from lacking expressiveness and flexibility of the structure of traditional file systems. Another problem users are facing today is an increasing *information fragmentation* [2]. A large number of desktop applications for personal document management exists, typically applying individual storage and indexing structures for specific document types (e. g., photo management software) and providing different access to the content. The reuse of metadata across personal desktop applications is rather restricted.

With the help of Semantic Web technologies, which ensure machine processability and interchangeability, we are able to apply semantic knowledge models and paths to organize and describe heterogeneous multimedia items and their context. A document collection is no longer an aggregation of separate items, but forms an individual knowledge base providing rich and valuable data for innovative PIM (personal information management) applications which present an aggregated view of the relations and links between personal documents, dates, contacts, e-mails, etc. To avoid the information fragmentation mentioned above, such PIM applications should be lightweight solutions, accessing a central ontology-based document management system. Such an ontology-based personal multimedia document management system has to meet several challenges:

- Diverse file and metadata formats are in use today and even more will evolve in the near future. Thus, it is of utmost importance that suitable document analysis, metadata and feature extraction modules can be added to the system without any difficulty. Extensibility of supported schemas or standards also means that knowledge modeling and processing modules must be flexible and configurable enough to allow for media or format specific knowledge instantiation.
- The ontology model used as a foundation of the instantiated document and context descriptions must be expressive and efficient. In the context of personal document and information management, the design of a suitable ontology model is non-trivial, only few proposals exist, and standards are still missing. Thus, the design of modules for knowledge processing, storing and provision should take into account that ontology models need to be replaced or changed. This must not result in substantial re-engineering work.
- As semantic data about documents and their relations to other resources tends to become very complex over time and therefore difficult to handle, it is necessary to integrate and apply control facilities, to enable the user to take corrective action and prevent him from being overstrained. Another result of the growing complexity of a knowledge base can be a loss of confidence – if the users are no longer able to check its correctness

themselves. Thus, appropriate support must be provided to make sure that the content of the knowledge base is sound and reliable at any time.

In this paper we present an integrated approach for ontology-based personal multimedia document management which addresses these issues, developed within the K-IMM (Knowledge through Intelligent Media Management) project [3]. After a discussion of related work in Section II, we present the process of generating semantic descriptions for personal multimedia documents in Section III and our approach for data consolidation and document life cycle management in Section IV. The architecture of the K-IMM system, including a prototypical example application for personal document and information management, is presented in Section V. Section VI concludes the paper and suggests future research directions and open issues.

II. Related Work

Existing projects with comparable goals (ontology-based document management) can be classified according to their focus on either manual ontology-based annotation or (semi-) automatic semantic data modeling. A comprehensive survey of the state of the art of semantic annotation for knowledge management is presented in [4]. Manual annotation systems mostly emerged in the context of Web document annotation, e. g., Annotea [5], SMORE [6] and CREAM [7]. Later, dedicated multimedia document annotation solutions like, e. g., Caliph&Emir [8] and AKTiveMedia [9], evolved. Most of the work on ontology-based annotation proceeds from the assumption that, before annotation starts, an appropriate ontology has to be created or assigned as a description schema (top-down approach). If this is left to the users, modality and sense of annotations depend on their intention which is even more difficult for non-ontology engineers.

Projects concerned with the problem of *automatic generation and maintenance* of semantic metadata are either targeting the automatic extraction and modeling of knowledge from documents (*Ontology Population* or *Ontology Instantiation*) based on NLP-techniques (e. g., KIM [10], ArtEquAKT [11], MediaCampaign [12]) or at the development of a so-called "Semantic Desktop". The NEPOMUK project [13] dealt with the development of a standardized, conceptual framework for "Semantic Desktops" which includes information extraction and wrapping from heterogeneous data sources, based on the Open Source project Aperture [14] (a reference implementation is Gnowsis [15]). Other projects that can be named in relation to "Semantic Desktop" are e. g., D-BIN [16], IRIS [17] and Haystack [18]. The latter turned out to be too non-restrictive to prevent data from being corrupted by the user.

At present, ontology-based solutions for multimedia document management are results of projects like aceMedia, BOEMIE or X-Media. They are either focused on automated annotation of image and video content [19], multimedia information extraction for ontology evolution [20] or large scale methodologies and techniques for knowledge management [21]. Adequate support for private users often means that a well-balanced compromise between manual and automatic annotation must be found. Presently, there is no integrated approach for ontology-based personal multimedia document management – from the content analysis to valid semantic metadata – accounting for existing context information and so-called "world knowledge". In particular, most of the existing approaches do not explicitly focus on controllability and long-term maintenance regarding data integrity and consistency, as well as *document life cycle management*. Furthermore, from a developer's point of view, the domain-specific customization and configuration (i. e., substitution of the used ontology model) is not explicitly supported.

III. Automatic Generation of Semantic Descriptions for Multimedia Documents

The prevalent uncertainty and ambiguity of interpretation and interrelation of information sources and the various application scenarios led us to the concept of a stepwise information instantiation process [22]. Figure 1 broadly depicts the generation process, showing a sequence of distinguishable stages of data modeling which will be described in more detail in the following.

A. Document Analysis

A multimedia document is processed and analyzed by a specific *Analyzer* component, depending on its media type and file format. Available Analyzers register dynamically at runtime and are thereupon considered as providers of specific information about a certain document type. They perform the task of document pre-processing, i. e., the file format specific processing and extraction of embedded metadata and raw data (content), and the format-independent analysis of the extracted (multimedia) content. Irrespective of the type of document, each analyzing process starts with the following steps: (a) identification of the file format, (b) extraction of embedded metadata, and (c) extraction of the raw data.

The correct identification of the file format is most important for the further processing and interpretation of the content. Even if the file has an appropriate filename extension, it can not be assured that it really complies with the corresponding format (e. g., because of multiple use of filename extensions). The primary decision criterion (whether the file can be processed and analyzed by the component) must be provided by the component itself to guarantee that the content can be analyzed correctly. Of course, embedded metadata can only be extracted and analyzed if the way it is stored (or embedded) complies with a certain standard or de facto standard. The same applies to the actual raw data of the document.

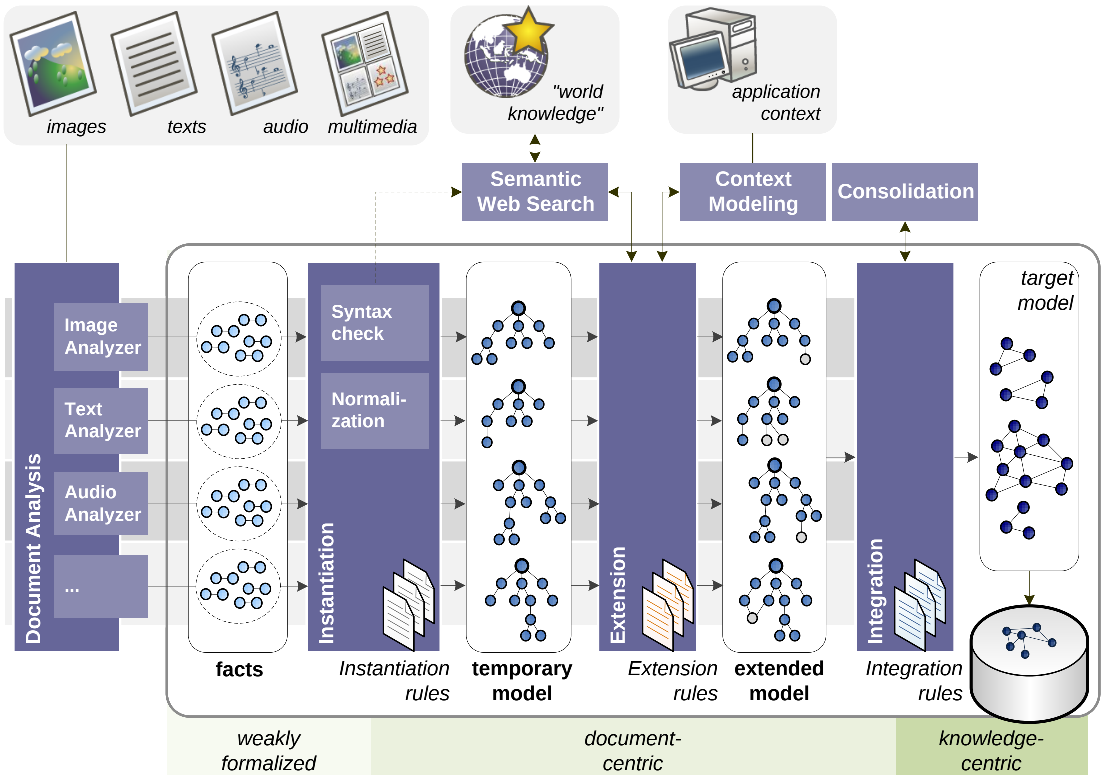

Figure 1. Data Modeling Process

The content-based exploitation and analysis of the raw data mainly comprises two subtasks: the segmentation or decomposition of the content to obtain logical parts (which can be further processed), and the determination of low- and high-level features. Following the principle of "divide-and-conquer", certain tasks of the content-based analysis of multimedia content are delegated to specialized subcomponents, allowing for reuse and substitution of particular solutions. Thus, an image which is part of a text document is extracted (by decomposition within a *Text Analyzer*) and passed to an appropriate *Image Analyzer*. Depending on the media type and file format, different techniques must be applied to extract low- and high-level features. Deducing high-level information from low-level features requires certain background information and user participation (e.g., to train classifiers for pattern recognition). For rather general fields of application, automated techniques are still missing and will hardly ever be on-hand without tight relationship to a user's context and conceptualization. Extent and complexity of the extracted data depends on how much background knowledge (rules and facts, or training data for the classification of low-level features) is available. The capabilities of an *Analyzer* component might be limited to the mere extraction of certain embedded metadata. Thus, it is possible to apply a combination of multiple *Analyzers* of different specialization to one document type.

B. Information Instantiation

As we can not predict the extent and quality of available information about multimedia documents, we need a flexible and extensible schema for the input data of the instantiation process. The most efficient way to specify descriptive information is in the form of attribute-value pairs (name and value of features or properties, like creator, modification date, but also color layout, sound intensity, etc.). A reduction to minimum structure allows a compact and uniform presentation of different sources and schemes. Furthermore, the list of attribute-value-pairs can grow dynamically. Thus, using this simple schema, an arbitrary number of Analyzer components can act as data providers. However, as the schema itself offers no validation ability, the passed input data might be incomplete or faulty, or contains redundancies or inconsistencies. To achieve an adequate level of data quality, the input data is evaluated within this process of *instantiation* (depicted in Figure 2) as follows:

1) **Filtering:** First of all, extracted data is filtered according to the requirements of the application domain. Filter criteria are defined in an editable configuration file. In case of redundant data a selection is made.
2) **Syntax check:** Syntactic errors occur if Analyzer components extract faulty data because of coding errors or problems with character sets. Examples are improper, supernumerary or missing characters, or the exceeding

of the range of values. The erroneous data is corrected or, if no automatic solution can be applied, excluded from further processing.

3) **Normalization:** Values originating from different data sources can be syntactically correct, but specified in different data formats (e. g., "`2008-01-13`" or "`01/13/2008`"). The transformation to a uniform, consistent data format is an important premise for further processing of the data.
4) **Transformation:** Finally, the filtered and normalized data is "translated" to the internal ontology model using a set of instantiation rules. In doing so, a decision is made regarding the interpretation of the mere syntactic input data by the semantic target model.

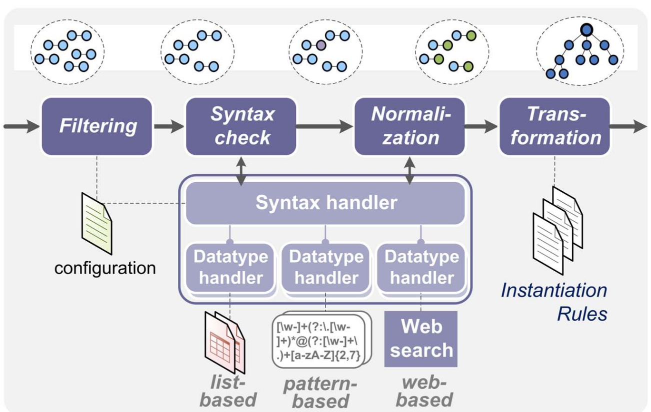

Figure 2. Information instantiation process

The process of syntax check and normalization is described in [22]. It is designed in a modular way, so that algorithms, sources and result format can be substituted and configured easily. The datatype handlers are either based on regular expressions (*pattern-based*), dictionaries (*list-based*) or web search results (*web-based*).

As for the process of *transformation*, we apply the following procedure: Each attribute-value-pair represents per se a *statement* and can be specified in RDF using the utility property `rdf:value` as predicate and a unique key as subject identifier. Hence, the resulting RDF model can be passed to a reasoner component to apply a set of configurable *Instantiation Rules*, allowing for appropriate customization.

C. Extension and Integration

In the next step, the resulting model is extended with additional data, i. e., with semantic information found on the Internet provided by a *Semantic Web Search Component*, described in more detail in [23], and current context information provided by a *Context Modeling Component*, presented in [24]. We assume that these services provide data in OWL over a standardized interface (using SPARQL) and that the ontologies are publicly available. An example for context information and a sample query is given in [24]. To allow for dynamic query composition, we introduced a template mechanism to specify SPARQL queries with the help of placeholders. An example (describing context information about an email transmission) is given in the following:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX cm: <http://mmt.inf.tu-dresden.de/crocoon/
       context-mail.owl#>
PREFIX cu: <http://mmt.inf.tu-dresden.de/crocoon/
       context-upper.owl#>

SELECT ?mail ?property ?value
WHERE
{
  ?mail rdf:type cm:Email.
  ?mail cm:hasAttachment ?d.
  ?d cu:uniqueID [[SHA1_content]].
  ?mail ?property ?value.
}
```

The placeholders, tagged with squared brackets, are replaced at runtime with adequate attributes – in this case the SHA1 hash of the document's content, e. g.,

```
?d cu:uniqueID "d07149922d9f84c097f7ccf6ed5c7b658c4229d0".
```

The result of the query can be used to extend the existing semantic information about the document. Thus, a collection of configurable *Extension Rules* is applied to the temporary data. An example of an Extension Rule (in Jena Rules syntax [25], `person` and `core` are namespaces of the target ontology model) could be:

```
[foaf1: (?P rdf:type person:Person),
        (?F rdf:type foaf:Person),
        (?F foaf:page ?homepage),
        (?F foaf:depiction ?img)
        -> (?P person:homepage ?homepage),
           (?P core:imgLink ?img)]
```

Finally, the resulting temporary model can be inserted into the system's RDF repository. The concluding step of *Integration* (cf. Figure 1) performs two tasks:

- the exploitation of interrelations within the temporary model, and
- the verification and consolidation of the new information – both in isolation as well as in context of already existing information in the repository.

At first, a set of *Integration Rules* is applied to the temporary model to deduce interrelations between instances, e. g.,

```
[html1: (?H rdf:type ex:HTMLDocument),
        (?H ex:containsURL ?img),
        (?P rdf:type ex:Image),
        (?P ex:filepath ?fp),
        equal (?fp, ?img)
        -> (?H ex:contains ?P) ]
```

Secondly, the consolidation process is invoked, which is later on described in Section IV-A. Afterwards, the generation process is completed.

IV. Maintenance

Due to the unsupervised analysis and extraction of document descriptions and context information, inserted data is likely to be of inferior quality in terms of consistency, accuracy, and redundancy. Furthermore, generated semantic descriptions become obsolete if documents are modified. In this section we describe our approach for maintaining the semantic data model, regarding consolidation and document life cycle management.

A. Data Validation and Consolidation

The semantic consolidation process within our system considers data in the context of the whole knowledge base. In general, consolidation is necessary whenever the knowledge base has been changed or extended by any automated process. It is invoked by the above-mentioned *Integration Component*. Our *Consolidation Component* is composed of three subcomponents: the *Semantic Conflicts Handler*, the *Duplication Handler*, and the *Incompletion Handler*. Their interrelation and the overall process of consolidation is described in detail in [22]. By now, the *Consolidation Component* provides high configuration ability: depending on the actual application context, the set of rules for the detection of semantic conflicts and incompletion, as well as the metrics and threshold for duplication detection can easily be adjusted or replaced. The following example rules (Jena Rules syntax) should illustrate the mode of action:

```
[rule1: (?P rdf:type ex:Person),
        (?P ex:bornOn ?T1),
        (?P ex:authorOf ?X),
        (?X ex:createdOn ?T2),
       greaterThan(?T1, ?T2)
       -> reportConflict(?P, ?X, '...description') ]

[label: (?O rdf:type ?C),
       noValue(?O rdfs:label)
       -> reportIncompletion(?O, 'Missing label...') ]

[fname: (?P rdf:type ex:Person),
       noValue(?P ex:familyName)
       -> reportIncompletion(?P,
              'Person has no surname...') ]
```

The first rule is an example for the detection of a semantic conflict (*"If a person, born on BirthEvent T1, is author of a Document X, created on AuthoringEvent T2, then T1 must have happened before T2.*). The second example is a domain-independent rule to detect incompletion (missing label), whereas the last one is a domain-dependent incompletion rule (missing attribute).

If a decision for conflict or duplication resolution can be made automatically, the user does not need to intervene. If a clear decision cannot be assured, the user must be involved for case-related judging. To minimize additional effort whilst providing the user with a high degree of control, it is necessary to find a compromise between fully automatic, semi-automatic and manual solution of the above-mentioned data problems. We propose two different approaches:

- Detected problems which can not be clearly solved are reported to the user (with a proposal for solution), leaving the active decision to him.
- All problems are solved automatically. Every decision is logged, in such a way that it can be undone.

Both approaches are supported by our solution as it provides machine-readable as well as human-readable problem description, according to a purpose-made ontology. These problem descriptions are produced by the above-mentioned handlers and passed to a central management component which performs the task of storage and provision, as well as solution and deletion of the conflicts depending on user feedback.

B. Document Life Cycle Management

A document's life cycle comprises all stages of a document: from its creation, processing, storage, and usage, to its disposal. In the context of personal document management, these stages are not clearly separable. Nevertheless, information about development stages of a document are quite valuable if they are related to the user's activities and events. In order to deal with the document life cycle, we specified a *Document Life Cycle Management* (DLCM) process, performing the task of modeling information about document activities which are either

1) activities which affect the document itself (creation, editing, and deletion) and according semantic information within the database needs to be updated, or
2) usage and management activities which do not affect the document itself (rendering, printing, publishing, sharing, retrieval, annotation, etc.).

An overview of the workflow is given in Figure 3. Document modifying activities invoke the application of *Update* and *Extension Rules* to the knowledge model.

In accordance to the instantiation process described in Section III-B, the modeling of metadata about the activities themselves is also a configurable, rule-based transformation process to provide flexibility and allow for substitution of the used ontology model. The input data for this modeling process should comply with a determined ontology which we call *Document Life Cycle Ontology (DLC)* – a purpose-built ontology to describe the above-mentioned document activities.

The DLC instantiation is triggered by an event handler which receives data from file system events, available context providers, and the management system itself (cf. Figure 3). To retrieve context information about a document's usage, we integrated the generic *Context Modeling Component* [24], already mentioned in Section III-B, which gathers and models cross-application context data from available context providers (e.g., from desktop applications, like e-mail clients, authoring tools, etc.).

Unlike the process of generating new semantic descriptions, as described in Section III, the update process of the

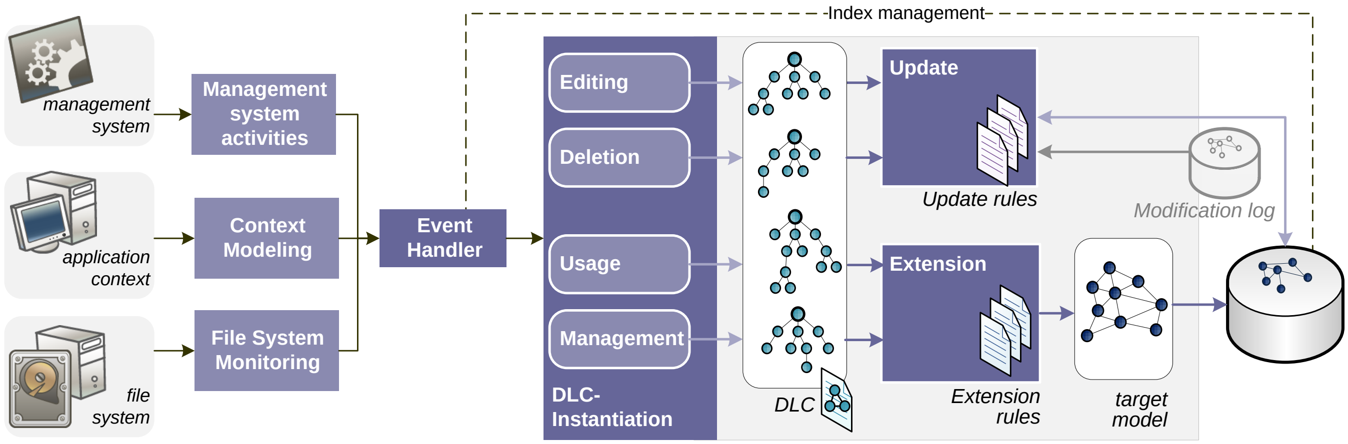

Figure 3. Document life cycle modeling process

DLC modeling process modifies existing semantic descriptions. This also means that user-created descriptions, i. e., descriptions of particular importance, might be altered unintentionally. To avoid this, we integrated a logging mechanism which records all user-driven activities in the repository (i. e., manual instance or statement creation, deletion, or editing) using a special logging ontology. The DLC modeling process accounts for the existing *modification log* (cf. Figure 3) by accomplishing only deletion and modification rules which do not effect user-generated content. To limit the size of the modification log it is advisable to apply suitable replacement policies, e. g., deleting log entries according to their timestamp (FIFO).

V. *K-IMM:* Architecture and Implementation

Based on the concepts described in Sections III and IV, we developed a prototypical personal multimedia document management system, designed as a service platform which autonomously manages documents stored on the local file system of the user. The import and indexing of multimedia assets (of different type) is performed by background tasks. An overview of the system is depicted in Figure 4. Three layers can be distinguished and are marked in the figure accordingly: (I) multimedia document indexing and analysis, (II) semantic data modeling and storage, and (III) a domain-dependent application layer.

(I) The document analyzing components extract specific properties and features (as described in Section III-A).

(II) The extracted data is passed to the *Semantic Modeling and Consolidation* component which provides the subcomponents for information instantiation and propagation, data validation and consolidation of the knowledge base, as well as the described document life cycle management. As mentioned in Sections III-C and IV-B, a *Semantic Web Search* component [23] and a *Context Management* component are connected to it to allow for the semi-automatic extension of semantic descriptions. The results of modeling and consolidation processes are stored in a persistent RDF/OWL repository using a third-party RDF/OWL API. The *Model Management and Processing* component provides an abstraction layer which allows for the substitution of applied RDF/OWL processing frameworks on the data layer. A component for *User and Rights Management*, described in more detail in [26], allows for sharing semantic descriptions with other users (on the local computer or via a remote RDF server).

(III) The topmost component (*Data Interface*) provides access to the modeled information for miscellaneous front-end applications for personal document management. On this level, application developers can configure or replace the used domain-specific ontology model and the corresponding rule sets (more details are given in Section V-A).

The overall architecture of the K-IMM system is realized in Java based on the OSGi [27] execution environment *Equinox* [28]. The diverse system components (described above) are implemented as OSGi *Service Bundles* which makes it possible to install, register, and start services (e. g., for multimedia analysis or user interface components) at runtime and on demand. Currently, there are three prototypical document analyzing components: an *ImageAnalyzer* for digital photographs, a *TextAnalyzer* for text documents, and an *AudioAnalyzer* for music files. RDF and OWL processing and storage is based on the Jena Semantic Web Framework [29], including its inference support for the application of rules and reasoning services. Particularly, we employ Jena's general purpose rule engine, its rule syntax and the concept of *Builtin primitives* [25] to pass data to corresponding Java modules, especially for evaluation and weighting algorithms within the process of data consolidation.

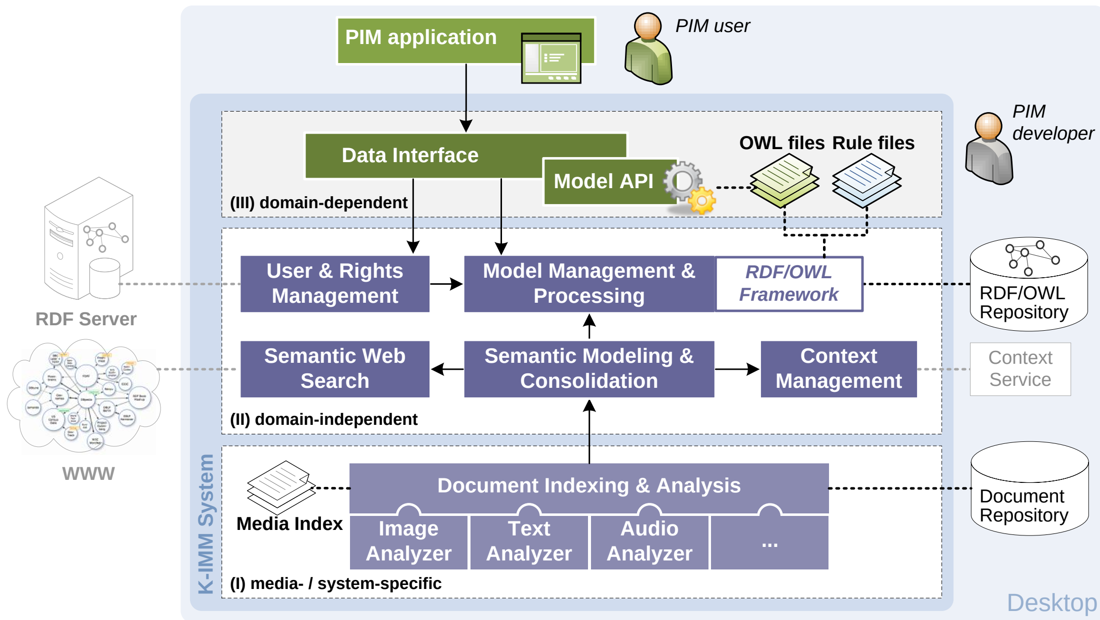

Figure 4. K-IMM Architecture (overview)

A. Domain-dependent Application Layer and Authoring Process

The Model API within the topmost layer of the architecture enables modeling, storing, and processing of instance data of the used ontology model. It provides an object-oriented access to the ontology-based dataset which is very useful for external software components (e. g., PIM applications) to access and edit information in an object-oriented way. Thus, external applications can create, modify or delete instances (e. g., if a user manually edits semantic descriptions) which is mapped by the Model API to appropriate operations on the RDF-based data layer. The Model API is *dynamically generated*, mapping OWL concepts to object-oriented classes with adequate methods for relations, with the help of an automated build-process. Thereby, at design time, the Model API is most flexible, allowing for substitution or modifications of the ontology model. Moreover, domain-specific rules used for the generation and consolidation processes, described in Sections III and IV, are kept in this layer. Thus, the developer is able to control these processes in order to meet the demands of the PIM application.

The intended authoring process comprises the following four steps:

(1) Building the application ontology in OWL (e. g., using Protégé [30]),
(2) Specifying the configuration settings for
   - the instantiation process (with regard to the documentation of available Analyzer bundles),
   - duplication handling (similarity metrics), and
   - the compilation of SPARQL queries for the acquisition of context information and semantic web search results,
(3) Specifying rule sets for
   - consolidation (semantic conflicts, incompletion),
   - updating the document descriptions (DLC),
   - extension (regarding available context information and "world knowledge"),
   - integration (establishing relationships between documents),
(4) Implementing the front end application based on the object-oriented data interface.

A graphical representation is depicted in Figure 5.

The benefit of the system and its application layer is the separation of concerns: a declarative configuration of the application domain and its "business logic", and the imperative programming of the front end application. Developing suitable authoring tools, based on a linear, guided authoring process, is obviously worthwhile. Furthermore, it would also be possible to introduce distinct authoring roles, e. g., the *ontology designer* (a domain-expert, responsible for steps (1) and (3)), the *process designer* (responsible for step (2)) or the *user interface developer* (responsible for step (4), in general the most laborious task).

In the following we present an application example which shows the feasibility of our approach.

B. A K-IMM-based Desktop Application

Based on the exemplary implementation of the K-IMM System, we set up a desktop application based on the

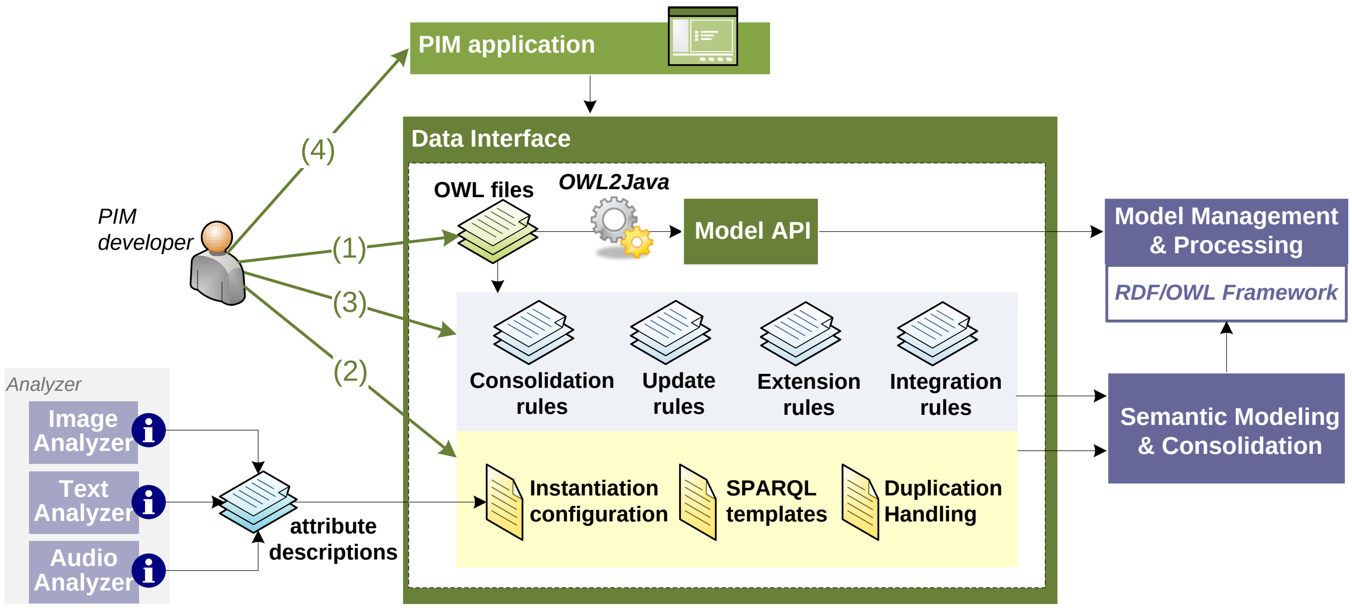

Figure 5. Authoring process of the PIM developer

Eclipse Rich Client Platform (RCP) [31]. The application uses the Data Interface and Model API to retrieve semantic information managed by the K-IMM System, modeled and managed as described in Sections III and IV. The used ontology is a purpose-built PIM ontology, consisting of 87 concepts and 173 properties. The necessary configuration and Jena Rule files include about 500 lines of code (LOC), whilst in contrast, the sophisticated graphical user interface (GUI) expectedly comprises more than 20.000 LOC in Java. A screenshot of the application is shown in Figure 6.

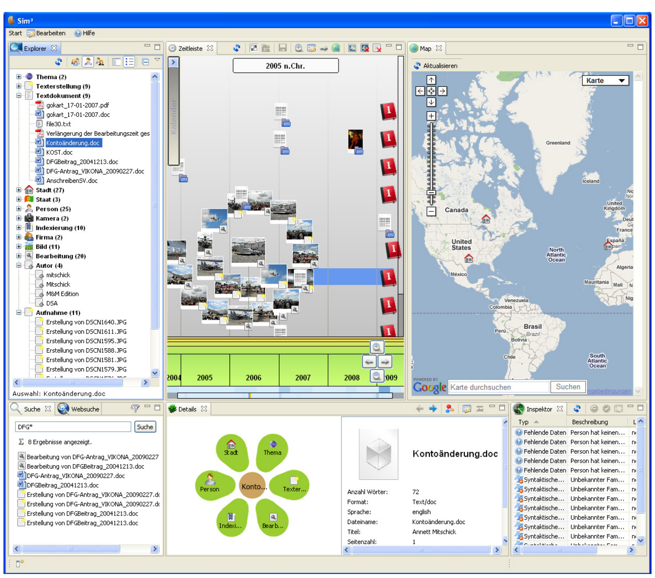

Figure 6. Screenshot of the demo application

The GUI provides several views to present and edit the available instances. Resources which have relations to spatial or temporal information (e. g., creation date/place) are visualized as pictograms in a *geographical view* (on the right), based on the Google Maps API [32], and in a *timeline view* (in the middle) which can be zoomed smoothly for different levels of detail [33]. The application allows for the unrestrained edition and creation of semantic descriptions, providing dynamically generated dialogs with appropriate data type verification.

The GUI also contains a so-called *Inspector* view in the bottom right corner. It presents currently existing and automatically detected problems with a human-readable description and proposed solution (cf. Section IV-A). Thus, the user can solve a conflict or delete the problem record with just one mouse click.

Furthermore, the application features a Semantic Web search widget, depicted in Figure 7. It illustrates the application of semi-automated gathering and integration of "world knowledge" found on the Semantic Web. As an example, the figure shows the extension of the person instance "Peter Jackson" using information from DBpedia.org [34] – with one mouse click.

VI. Conclusions and Future Work

In this paper we presented an integrated approach for ontology-based personal multimedia document management which covers the whole process of automated modeling of semantic descriptions for multimedia objects: document analysis, information instantiation, context-aware extension and integration, data consolidation, and observance of the document's life cycle. These aspects have been described in Sections III and IV. They provide the basis for the design

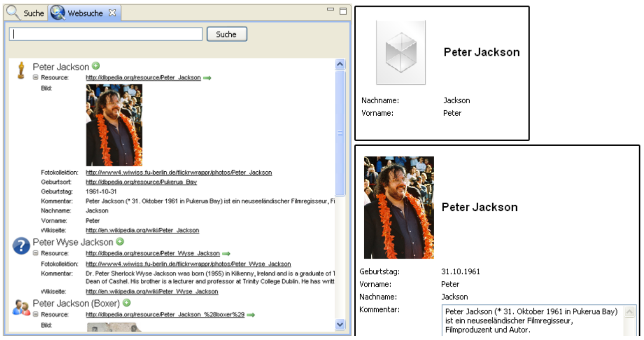

Figure 7. Detail of the demo application showing the widget for Semantic Web search

of a personal multimedia document management system, presented in Section V. It consists of three layers, separating *system-specific document analysis*, *domain-independent semantic data modeling and storage*, and the *domain-dependent application interface*. Thus, our solution allows for the domain-specific customization and substitution of the used ontology model and the corresponding modeling rules and configurations. The proposed approach is perfectly flexible regarding domain-specific alterations done by application developers, or regarding future document or metadata formats. We have proven the practicability of our concept by a prototypical implementation of the K-IMM system as an OSGi-based service platform for personal information management applications. As a demo application we created a comprehensive RCP front end, described in Section V-B.

We hope that this approach helps researchers and developers who pursue similar objectives. Of course, the benefit of this concept heavily depends on the usability and "added value" of suitable applications for personal information management. However, the rich, ontology-based datasets, which are automatically generated, consolidated and managed by the K-IMM system, form a proper basis to create advanced and lightweight front ends.

In the near future we will concentrate on more detailed performance and usability evaluation within our K-IMM-based personal desktop application for ontology-based multimedia document management. Additionally, as we have already adopted our system in other application scenarios (e. g., within the professional domain of construction process management addressed in the project BauVOGrid [35]), we would like to evaluate its feasibility within further application domains. Finally, to ameliorate and simplify the development process it would be reasonable to work on a convenient authoring tool which supports ontology design, specification of rules and configurations, as well as several authoring roles.

References

[1] A. Mitschick and K. Meißner, "Generation and maintenance of semantic metadata for personal multimedia document management," in *Proceedings of the First International Conference on Advances in Multimedia (MMEDIA 2009)*, Colmar, France, July 2009, pp. 74–79.

[2] D. R. Karger and W. Jones, "Data Unification in Personal Information Management," *Communications of the ACM, Special Issue: Personal information management*, vol. 49, no. 1, pp. 77–82, Jan. 2006.

[3] K-IMM Project, Chair of Multimedia Technology, TU Dresden, URL: http://mmt.inf.tu-dresden.de/K-IMM/, last accessed: 06/16/2010.

[4] V. Uren, P. Cimiano, J. Iria, S. Handschuh, M. Vargas-Vera, E. Motta, and F. Ciravegna, "Semantic annotation for knowledge management: Requirements and a survey of the state of the art," *Journal of Web Semantics*, vol. 4, no. 1, 2006.

[5] J. Kahan and M.-R. Koivunen, "Annotea: an open RDF infrastructure for shared Web annotations," in *WWW '01: Proceedings of the 10th international conference on World Wide Web*. New York, NY, USA: ACM, 2001, pp. 623–632.

[6] A. Kalyanpur, J. Golbeck, J. Hendler, and B. Parsia, "SMORE - Semantic Markup, Ontology, and RDF," Mindswap, Tech. Rep., Nov. 2002.

[7] S. Handschuh and S. Staab, "Authoring and Annotation of Web Pages in CREAM." in *Proceedings of the Eleventh International World Wide Web Conference, WWW2002*, 2002, pp. 462–473.

[8] M. Lux, "Semantische Metadaten: Ein Modell fr den Bereich zwischen Metadaten und Ontologien," Ph.D. dissertation, Graz University of Technology, 2006.

[9] A. Chakravarthy, F. Ciravegna, and V. Lanfranchi, "Cross-media Document Annotation and Enrichment," in *Proceedings of the 1st Semantic Authoring and Annotation Workshop (SAAW2006)*, 2006. [Online]. Available: http://www.dcs.shef.ac.uk/ ajay/publications/paper-camera-workshop.pdf

[10] B. Popov, A. Kiryakov, A. Kirilov, D. Manov, D. Ognyanoff, and M. Goranov, "KIM - Semantic Annotation Platform," in *International Semantic Web Conference*, ser. Lecture Notes in Computer Science, D. Fensel, K. P. Sycara, and J. Mylopoulos, Eds., vol. 2870. Springer, 2003, pp. 834–849.

[11] M. J. Weal, H. Alani, S. Kim, P. H. Lewis, D. E. Millard, P. A. S. Sinclair, D. C. De Roure, and N. R. Shadbolt, "Ontologies as facilitators for repurposing web documents," *International Journal of Human Computer Studies*, vol. 65, pp. 537–562, 2007.

[12] M. Yankova, H. Saggion, and H. Cunningham, "A Framework for Identity Resolution and Merging for Multi-source Information Extraction," in *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odjik, S. Piperidis, and D. Tapias, Eds. Marrakech, Morocco: European Language Resources Association (ELRA), May 2008.

[13] T. Groza, S. Handschuh, K. Moeller, G. Grimnes, L. Sauermann, E. Minack, C. Mesnage, M. Jazayeri, G. Reif, and R. Gudjonsdottir, "The NEPOMUK Project - On the way to the Social Semantic Desktop," in *Proc. of I-Semantics' 07*. JUCS, 2007, pp. pp. 201–211. [Online]. Available: http://www.dfki.uni-kl.de/ sauermann/papers/groza+2007a.pdf

[14] Aperture, Aduna Software, URL: http://www.aduna-software.com/technology/aperture, last accessed: 06/16/2010.

[15] L. Sauermann, G. A. Grimnes, M. Kiesel, C. Fluit, H. Maus, D. Heim, D. Nadeem, B. Horak, and A. Dengel, "Semantic Desktop 2.0: The Gnowsis Experience," in *Proc. of the ISWC Conference*, Nov 2006, pp. 887–900.

[16] G. Tummarello, C. Morbidoni, and M. Nucci, "Enabling Semantic Web Communities with DBin: An Overview." in *Proc. of ISWC 2006*, ser. Lecture Notes in Computer Science. Athens, GA, USA: Springer, 2006.

[17] A. Cheyer, J. Park, and R. Giuli, "IRIS: Integrate. Relate. Infer. Share." in *Proc. of 1st Workshop on The Semantic Desktop. 4th International Semantic Web Conference*, Nov. 2005, p. 15.

[18] D. R. Karger, K. Bakshi, D. Huynh, D. Quan, and V. Sinha, "Haystack: A Customizable General-Purpose Information Management Tool for End Users of Semistructured Data," in *In CIDR*, 2005.

[19] S. Dasiopoulou, C. Saathoff, P. Mylonas, Y. Avrithis, Y. Kompatsiaris, S. Staab, and M. G. Strintzis, "Introducing context and reasoning in visual content analysis: An ontology-based framework," in *Semantic Multimedia and Ontologies*. Springer Verlag, 2008.

[20] S. Castano, A. Ferrara, S. Montanelli, and D. Lorusso, "Instance matching for ontology population," in *Proc. of the 16th Italian Symposium on Advanced Database Systems, 22-25 June 2008, Mondello, Italy*, 2008, pp. 121–132.

[21] P. Buitelaar, P. Cimiano, A. Frank, M. Hartung, and S. Racioppa, "Ontology-based information extraction and integration from heterogeneous data sources," *Int. Journal of Human-Computer Studies*, vol. 66, no. 11, pp. 759 – 788, 2008.

[22] A. Mitschick and K. Meißner, "Metadata Generation and Consolidation within an Ontology-based Document Management System," *Int. Journal of Metadata, Semantics and Ontologies*, vol. 3, no. 4, pp. 249–259, 2008.

[23] A. Mitschick, R. Winkler, and K. Meißner, "Searching Community-built Semantic Web Resources to Support Personal Media Annotation," in *Proc. of SemNet 2007, Int. Workshop located at ESWC2007*, Innsbruck, Austria, 2007.

[24] S. Pietschmann, A. Mitschick, R. Winkler, and K. Meißner, "CroCo: Ontology-Based, Cross-Application Context Management," in *Proc. of SMAP 2008*. Prague, CZ: IEEE Computer Society, December 2008.

[25] Jena Rules Documentation, Jena Semantic Web Framework, URL: http://jena.sourceforge.net/inference/, last accessed: 06/16/2010.

[26] A. Mitschick and R. Fritzsche, "Publishing and Sharing Ontology-Based Information in a Collaborative Multimedia Document Management System," in *WISE Workshops*, Nancy, France, Dec. 2007, pp. 79–90.

[27] D. Marples and P. Kriens, "The Open Services Gateway Initiative: An Introductory Overview." 2001. [Online]. Available: http://www.adammathes.com/academic/computer-mediated-communication/folksonomies.html

[28] Equinox Project, Eclipse Foundation, URL: http://www.eclipse.org/equinox, last accessed: 06/16/2010.

[29] J. J. Carroll, I. Dickinson, C. Dollin, A. Seaborne, K. Wilkinson, and D. Reynolds, "Jena: Implementing the semantic web recommendations," in *Proc. of WWW Alt. '04*, New York, USA, 2004.

[30] Protégé Ontology Editor and Knowledge Acquisition System, URL: http://protege.stanford.edu/, last accessed: 06/16/2010.

[31] Eclipse Rich Client Platform (RCP), Eclipse Foundation, URL: http://www.eclipse.org/rcp, last accessed: 06/16/2010.

[32] Google Maps API, Google Code, URL: http://code.google.com/apis/maps/, last accessed: 06/16/2010.

[33] R. Dachselt and M. Weiland, "TimeZoom: A Flexible Detail and Context Timeline," in *Proc. of CHI '06*. New York, NY, USA: ACM, 2006, pp. 682–687.

[34] DBpedia Knowledge Base, Linking Open Data community project, URL: http://dbpedia.org/, last accessed: 06/16/2010.

[35] A. Gehre, P. Katranuschkov, and R. Scherer, "Semantic Support for Construction Process Management in Virtual Organisation Environments," in *ECPPM 2008 - eWork and eBusiness in Architecture, Engineering and Construction - Proc. of the 7th European Conference on Product and Process Modelling (ECPPM)*, S. R. Zarli A., Ed. Taylor & Francis Group, Netherlands, Sep. 2008.