Ontology-Based Content Model for Scalable Content Reuse

Saša Nešić, Mehdi Jazayeri
Faculty of Informatics,
University of Lugano
Lugano, Switzerland
sasa.nesic@lu.unisi.ch
mehdi.jazayeri@unisi.ch

Jelena Jovanović
School of Business Administration,
University of Belgrade
Belgrade, Serbia
jeljov@gmail.com

Dragan Gašević
School of Computing and
Information System,
Athabasca University
Athabasca, Canada
dgasevic@acm.org

ABSTRACT
The paper presents Abstract Compound Content Model (ACCM), a generic content model which we have developed aiming to facilitate interoperability, repurposing and integration of diverse platform specific content models. The ACCM model defines content units of different levels of granularity as well as a content aggregation architecture that organizes content units for content deliverables. Based on this model we have developed the ACCM ontology in order to turn the ACCM's elements (i.e., content units and content aggregations) into resources that can be directly accessed and thus reused. The paper also presents our current work on the implementation of an ACCM-based content management system and illustrates how we intend to use this system to facilitate the process of content authoring.

Categories and Subject Descriptors
I.2.4 Knowledge Representation Formalisms and Methods – *representation languages, miscellaneous*

General Terms
Design

Keywords
Content models, ontologies, interoperability, repurposing

INTRODUCTION
Content adaptation for reuse in different contexts has become an important research topic, since authoring of high quality digital content is rather an expensive and time consuming task. In order to fit the new context, the content needs to be adapted not only in a way to show or hide information that might or might not be relevant to the expected audience, but also to present it in a way that suits the users' cognitive characteristics. Therefore, there is a need for content enriched with a solid basis of declarative as well as structural knowledge. Declarative knowledge [11] describes what is known about the content's topic from the perspective of different domains. Structural knowledge describes the structure of the content, i.e., the organization and relationships among different content elements.

The majority of current solutions for content representation are platform/tool specific (e.g. PDF, PS, OpenOffice, MS Office, etc.) with a lack of declarative and structural knowledge, which limits the effectiveness of content reuse. They have their own native XML based schemas which define the structure, content and to some extent the semantics. However, the structure only allows for simple (hierarchical) representation of relationships among content units (e.g., text, graphics, images, etc.) encoded in particular schema elements. Content units are also hardly reachable from other external resources.

In order to achieve our main goal, that is scalable reuse of content across multiple contexts, we need a content, which is manageable independently of applications and storage facilities and can be located based on meaningful criteria rather than e.g., filename. Accordingly, we need to represent content in a way that will enable:

- Content knowledge acquisition, that is, gathering, organizing and structuring knowledge about the content;
- Simple content referencing;
- Access to and interaction with content units of varying granularity;
- Different relationships among content units;
- Simple content extraction and assembling;
- Transparent content evolution.

Our first step in achieving such content representation was to develop an abstract content model. Inspired by Darwin Information Architecture (DITA) [4] and Abstract Learning Object Content Model ALOCoM [8], we have developed Abstract Compound Content Model (ACCM). The model defines content units of different levels of granularity as well as the content aggregation architecture that organizes content units for content deliverables. The model considers content units as resources identified by unique version and unit identifiers. This enables clear and transparent content unit evolution regardless of the contexts in which a content unit is used. That also enables content referencing and content aggregation in more complex content structures.

The second step was to develop the ontology that enables formal representation of the ACCM model. Therefore, we have developed the ACCM-ontology. The ontological representation of the ACCM elements makes these elements directly accessible and thus reusable. In addition, content elements represented in such a way can more easily be enriched with ontological knowledge about specific domains, i.e. with ontological metadata [7]. This further enlarges the base of the content knowledge and enhances the content reusability. The ACCM-ontology also serves as a computa-

tional and platform independent content model, which allows content sharing through different platforms.

The rest of the paper is organized as follows. We begin by presenting the ACCM model and the model mappings to the DITA architecture. Subsequently, we discuss the benefits of the model and continue with the description of the ACCM-ontology. In order to illustrate our approach, we present a content authoring scenario based on the introduced model. Next, our current work on the realization of this scenario is given. Discussion on the related work and final remarks conclude the paper.

CONCEPTUAL SOLUTION

The highly dynamic online world manipulates a wide variety of digital multimedia content, developed by different communities using specific implementation technologies. There is also a plenty of different devices running different software, which are used to access the content and present it to the target audience. Much of online content was created independently of these devices and thus is not often suited to them.

If we want to develop an architecture that allows for content interoperability among different, platform specific, content models, we need a generic content model. This model should provide means for constructing uniform high-level views on various existing platforms, and can be used for repurposing or integration of platform specific content models. Accordingly, we have developed such a generic content model and named it Abstract Compound Content Model (ACCM). Figure 1 represents the model.

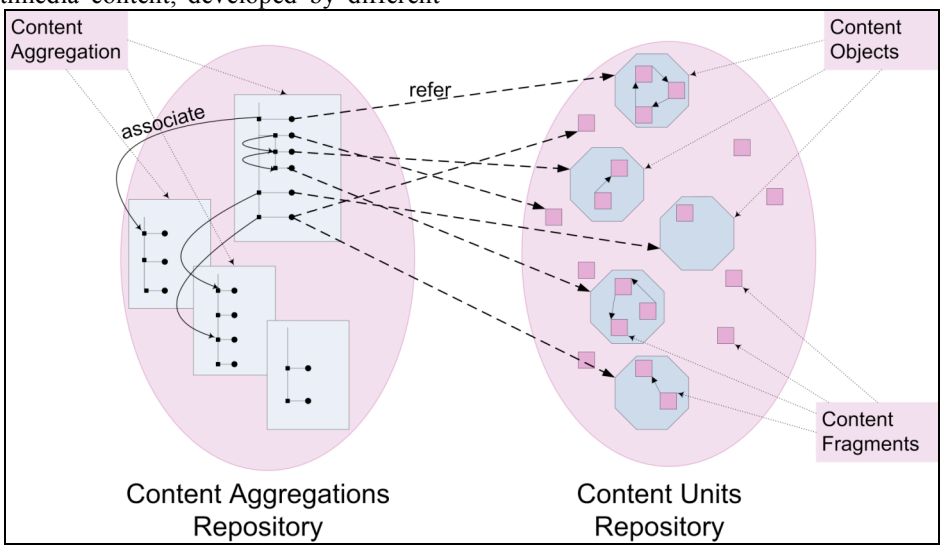

Figure 1. Abstract Compound Content Model

The ACCM model consists of the following elements:

Content fragments (CFs) are content units in their most basic form, like text, audio and video. These elements can be regarded as row digital resources. Content fragments can be further specialized into discrete (graphic, text, image) and continuous (audio, video, simulation, and animation) content fragments.

Content objects (COs) collect content fragments and add navigation. Navigation elements enable sequencing of content fragments in a content object. Besides content fragments, a content object can also include other content objects. Some examples of content objects are: paragraph, title, section, footnote, etc.

Content aggregation (CA) is a map, which represents the architecture of an information set. Specifically, it defines and organizes references to content units (i.e., CFs and COs) to indicate the relationships among them. References defined within one content aggregation can be organized into smaller groups to better reflect the content structure. In addition they can be associated with the references from the same or different content aggregations. A content aggregation can also serve as an outline or a table of content for content outputs (e.g., a slide presentation, a set of hyperlinked web pages, a PDF document, and a Word document). Content aggregations allow for scalable reuse of content across multiple contexts. Information architects and content publishers can use them to plan, develop and deliver the content.

Elements of the ACCM model (i.e., CFs, COs and CAs) are resources with unique resource and version identifiers. While the CFs and COs are used to model real-world content units, which are stored in the content repositories, CAs are documents which keep references to content units and are stored in separated repositories. In addition, all ACCM elements have metadata that describes them in a consistent fashion, facilitating their sharing and reuse. We relay on a standardized metadata set such as Dublin Core[1] (DC) for their annotation depending on the domain in which they are used, thus facilitating the content search, discovery, and retrieval. However, this metadata may capture a very limited amount of information and there is no way of guaran-

[1] http://dublincore.org/documents/dcmi-terms/

teeing that such metadata really captures relevant aspects of various domains with the breadth and the depth needed. To address this issue, we suggest enriching metadata of ACCM elements with concepts from appropriate domain-specific ontologies, i.e., with ontological metadata [7].

As we have already mentioned, the ACCM model is partially inspired by the DITA architecture [4] and ALOCoM model [8]. We reused content object and content fragment concepts defined in the ALOCoM and supplemented them by providing support for content versioning. The ACCM model introduces the new concept of content aggregation that enables different relationships among content units as well as referencing content units from different content aggregates – neither of these is supported by ALOCoM. Since both of these models (ALOCoM, ACCM) relay on DITA, in the rest of this section we outline the basic DITA concepts, and how they correspond to the ACCM model. The DITA is an XML–based architecture for creating topic-oriented, information-type content that can be reused and single-sourced in a variety of ways. DITA defines topic as a generic information type which is short enough to be specific to a single subject or answer a single question, but long enough to make sense on its own, and be authored as a unit. DITA facilitates the authoring of both context-sensitive and context-free topics. Besides the topic type DITA defines three core information types: concept, reference, and task. Another core concept of this information architecture is DITA map. DITA maps are documents that collect and organize topics for a specific delivery, including the generation of navigation files and links to related topics. The ACCM model that we propose fits well with the DITA architecture. The context-sensitive topic in DITA is analogous to the content object of the ACCM model. Although the content fragment in the ACCM does not have a direct equivalent in the DITA architecture, the context-free topic in the DITA can be treated as its equivalent on a more abstract level. Content aggregations from the ACCM are comparable with the DITA maps.

BENEFITS OF THE ACCM MODEL

Content represented in accordance with the ACCM model has numerous benefits that are visible almost in all phases of the content lifecycle, including content creation, distribution, management, utilization, and destruction. In the rest of this section we describe some of the ACCM's major benefits.

Content repurposing - reuse in different contexts

Due to the high costs of authoring high quality digital content, the reuse of once created content has become an important research topic. However, content reuse in different contexts is a very tricky and hard to implement strategy. It implies not just content filtering based on its relevancy to the target users, but also content presentation adapted to the users' cognitive features.

The process of content authoring based on reuse of the existing content, depends on efficient mechanisms for extracting and assembling content units from different sources. However, the extraction of content units from different, platform specific, content formats (e.g. MS Office, OpenOffice, and PDF) can be a very tedious job. The proposed ACCM model serves as a conceptual base for various content repurposing solutions. In particular, it allows the construction of a uniform high-level view on various platform specific content formats. By mapping the content from platform specific models into the ACCM model and by explicitly defining relationships between the content units in this model, we enable efficient extraction and reuse of content units of varying granularity levels.

Easier and less costly transformation between platform specific formats

The ACCM model is unaffected by the specifics of different implementation technologies, so the content represented in ACCM does not have to be rebuilt, each time a new technology or a presentation format comes along. Document mark up and style-sheet languages, like HTML, XML, and Cascading Style Sheets (CSS), tend to offer the same functionality for Web content [6], since these are declarative data formats, easily transformable to other formats. However, they cover only a subset of multimedia content space and usage scenarios.

The majority of existing authoring tools provide limited services for transformation of their native format to and from other formats. Although the great majority of them support other formats through an export function, it is often the case that information is lost when exporting from one tool-specific format into the other. What is more, this happens even though both the source and the target tools have the same purpose and use almost the same terminology (e.g., MS Word and OpenOffice).

Developing export/import bridges is a difficult and costly process, as it requires detailed knowledge of both (input and output) formats and interfaces. If we have, for example, N different platform specific formats, we need N^2-N bridges in order to enable all possible transformations [5]. Considering the highly dynamic online world, it is obvious that this approach is unsuitable. The ACCM model, along with the transformation tools that map a new format into ACCM and vice versa, has great potential to solve this problem. In this way the number of all possible transformations among N targeted formats is reduced to 2N.

Content filtering

The ACCM model and its compound nature enable efficient content filtering, i.e. generation of outputs that contain only those content units that are relevant to a specific audience. Specifically, by providing direct access to each content unit within a content resource, ACCM enables selective exclusion of those units that do not meet the requirements of a particular usage context.

Simple content referencing

When using the same content unit over and over again, and when changes to the content unit should be applied to all platform specific outputs which contain it, it is easier to have references to the original content unit, instead of having too many copies of that content unit. The ACCM's content aggregation concept allows for simple referencing of content units. Therefore, when certain content units, referenced in a content aggregation change, that change can be easily propagated to all platform specific outputs based on that content aggregation (by rebuilding those outputs, so that they include the changed content units).

ACCM also enables relationships among elements from the same or different content aggregation instances. That actually enables the navigation through the associated content units originating from different content outputs. For example, by mapping the instances of platform specific formats (e.g, OpenOffice, MS Word, PDF) to instances of the ACCM content aggregation we can relate for instance, a paragraph from an MS Word document to an image from an OpenOffice slide presentation.

Transparent content evolution

By means of content unit identifier and content unit version identifier, the ACCM model allows keeping track of content unit versions and makes the process of content evolution clear and transparent [9]. These identifiers ensure that the history and traceability of a content unit is never lost. Each change of a content unit creates a new version of that content unit. The formal definitions of concepts in the ACCM model also enable the formal specification of changes, which are made to content units, either as a set of differences between two versions or as a set of change operations that can be applied to one version to create another [9]. This area is one of our main future research targets.

Simple addition of metadata

As mentioned earlier, we assume that content elements of the ACCM model are annotated with standardized metadata. However, the same content can be used for several purposes, and the metadata relevant for one purpose may be different from the one required for another purpose. Since different users reuse the same content, it is important to understand the individual needs of each user, but standard metadata (e.g. DC and IEEE LOM) still plays a small role in distinguishing one user from another. Therefore, this metadata has to be extended with information derived from data about different aspects of interactions between the content and its users. In other words, the content needs to be made aware of its usage during its lifetime. Different users' behavior, when they reuse the same content, defines the content behavior itself.

The ACCM model defines content elements as resources, which can easily be accessed and their usage (meta)data can be simply attached to them. This metadata will improve the content semantics, simplify further content analysis and enable reasoning that will determine content behavior. The content, aware of its structure and past behavior, can be considered an intelligent content. To be aware of its past behavior means that the content is accompanied with information about actions in which it has been used. This information along with structural information can be used in planning algorithms [12], which are executed by intelligent agents to achieve some goals. Represented in the ACCM model, the content has the potential to achieve this goal.

ONTOLOGY BASED SOLUTION

Figure 2a represents the views on content from different levels of abstraction. The top level of the abstraction is the computation-independent content model, in our case the ACCM model. Underneath the ACCM model is the ACCM-ontology that is used to formally represent the model and serves as a platform-independent model. Next layer towards the bottom is defined as a set of platform specific models (schemas). At the bottom of this layered architecture, we have content data as instances of platform specific models.

The ACCM ontology was developed in the Ontology Web Language (OWL), using the Protégé[2] ontology development tool and is available at [10]. The first step in building the ontology was the definition of the components of the ACCM model, since we take this model as the starting point. Accordingly, the ACCM ontology distinguishes between the content fragment, content object, and content aggregation concepts. These three concepts are formally represented by *accm:ContentFragment*, *accm:ContentObject*, and *accm:ContentAggregation* classes, respectively. Content object and content fragment are sub-concepts of a more general content unit concept, which is represented by the *accm:ContentUnit* class in the ontology. Moreover, *accm:ContentUnit* and *accm:ContentAggregation* are sub-concepts of the root *accm:ContentType* class. Figure 2b and 2c present the core concepts and properties of the ACCM ontology, in the form of a Resource Description Framework (RDF) graph.

The ontology defines a number of content fragment types. First, the distinction is made between continuous and discrete content fragments, formally represented by *accm:ContinuousCF* and *accm:DiscreteCF* classes respectievly. Those classes are subclasses of the *accm:ContentFragment* class. The discrete content fragments are further specialized into text, images, graphics and tables, while the continuous content fragments are specialized into audio, video, animations, and simulations. All of them are represented in the ontology with appropriate classes which are subclasses of the *accm:ContinuousCF* and *accm:DiscreteCF* classes.

Besides content fragment types, the ACCM ontology defines a number of content object types, based on the DITA architecture. We defined the generic topic type *accm:Topic*

[2] http://protege.stanford.edu/

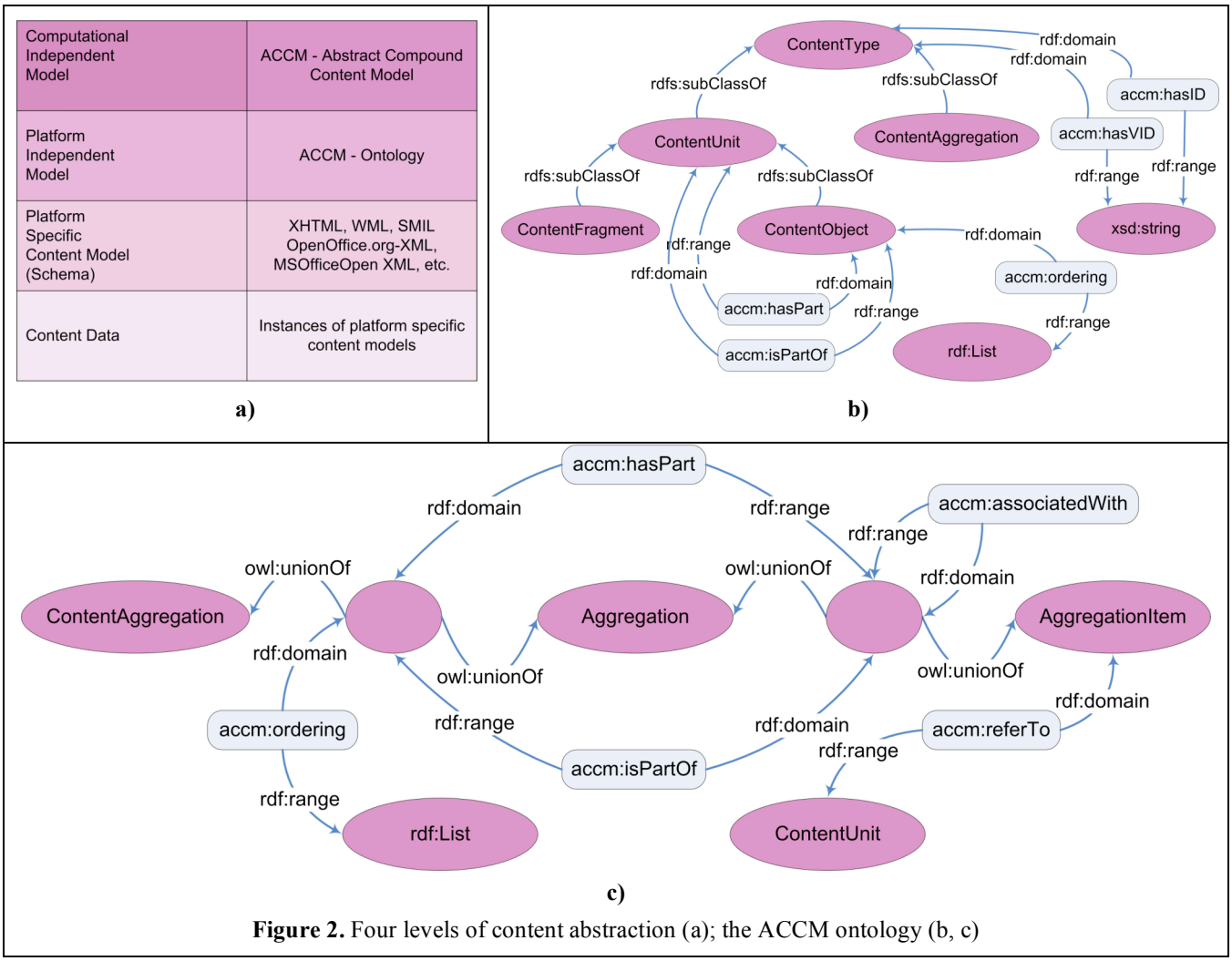

Figure 2. Four levels of content abstraction (a); the ACCM ontology (b, c)

and the three core information types of the DITA architecture: accm:*Concept*, *accm:Reference* and *accm:Task*. Many of the DITA building blocks are also included in the ontology as subclasses of the *accm:ContentObject* class such as *accm:Definition*, *accm:Section*, and *accm:Paragraph*, *accm:List* and *accm:Comment*.

The content aggregation type (formalized as the *accm:ContentAggregation* class) collects and organizes references to content fragments and content objects. In order to enable the aggregation of references to content units and to define relationships among them, we have introduced two additional concepts: aggregation and aggregation item, formally represented by *accm:Aggregation* and *accm:AggregationItem* classes. An aggregation item keeps a reference to an instance of *accm:ContentUnit* via the *accm:refersTo* property, and can be part of either an aggregation or a content aggregation. Each instance of *accm:Aggregation* aggregates instances of *accm:AggregationItem* and belongs to an instance of *accm:ContentAggregation*.

The ACCM ontology defines three types of relationships between its classes: 1) aggregational relationships; 2) navigational relationships; 3) associative relationships.

Aggregational relationships are expressed with the *accm:hasPart* property and its inverse *accm:isPart* property. As Figure 2c shows, the domain of the property is a union of *accm:Aggregation* and *accm:ContentAggregation* classes, while the range is defined as a union of *accm:Aggregation* and *accm:AggregationItem* classes. However, we also use the *accm:hasPart* property to define content objects (Figure 2b) as ordered aggregations of content units (content fragments and content objects). Accordingly, the domain of the property also includes the *accm:ContentObject* class, whereas the range is extended with the *accm:ContentUnit* class. We have used OWL restrictions to constrain the range of the *accm:hasPart* property for certain classes. For instance, we used OWL universal restriction (*owl:allValuesFrom*) to restrict the range of the *accm:hasPart* property in the context of the *accm:ContentAggregation* class to a union of *accm:Aggregation* and *accm:AggregationItem* classes. Navigational relationships are expressed with the *accm:ordering* property that defines the order of components in the form of an *rdf:List*. The domain of this property is defined as a union of *accm:ContentObject*, *accm:Aggregation*, and *accm:ContentAggreagtion* classes.

The range is an *rdf:List* composed of instances from the domain. In the context of content objects those are instances of *accm:ContentFragment* and *accm:ContentObject*. In the context of an aggregation and a content aggregation, those are instances of *accm:Aggregation* and *accm:AggregationItem*.

Associative relationships are enabled with the *accm:AssociatedWith* property, in which both the domain and the range are a union of *accm:Aggregation* and *accm:AggregationItem* classes. This kind of relationships allows the instances of *accm:Aggregation* and *accm:AggregationItem* to be associated with the other instances of those concepts within the same content aggregation. Actually, the associative relationships enable the links among similar instances, based on the given criteria, within a content aggregation. This also allows a random access or alternative paths through content aggregations.

In addition, the ACCM-ontology enables unique identification of all content elements represented in the ontology (content fragments, concept objects, and content aggregations) with *accm:hasID* and *accm:hasVID* properties having the *accm:ContentType* class as their domain.

AN APPLICATION EXAMPLE

In this section, we first present a content authoring scenario in order to illustrate the benefits that stem from the usage of the ACCM model for content representation. Subsequently, we present our current work on the implementation of an ACCM-based content management system (CMS) that supports this scenario.

A content authoring scenario

Let us suppose that Mark is working in a selling department of the company HappyContentSale and has to write a monthly report about his contacts with the company's clients and the contracts he has negotiated with them during the previous month. To write the report he has to gather data from different sources: his email correspondence with the clients, the meeting minutes, the signed contracts, etc. This further means that he has to spend a considerable time, first on searching his email-box for clients' emails, his file system for meeting minutes, the company's content management system for contracts, etc. Subsequently, he has to skim read through all those files in order to locate those parts that might be relevant for the report and make a lot of copy-and-paste actions. In addition, since required files are often in different formats, he would have to do a lot of formatting to make everything look smoothly in the final document.

Let us now suppose that Mark has an ACCM-based authoring tool (Figure 3) that enables him to create the monthly report in a quick and efficient manner. Mark would first sketch an outline of the document (i.e., report in this example) he intends to create. The tool, internally maps this outline into an ACCM content aggregation. For each 'leaf' item of the outline (i.e., each aggregation item in term of the ACCM model), Mark assigns a domain concept that should be addressed in that part of the document. The concept can be specified either in the terms of domain ontologies (in this case, it would be the ontology of commerce and an ontology covering the company's products and services) or in plain text. In the latter case, the tool would internally search domain ontologies for concepts labeled with the specified terms/phrases or some of their synonyms (the synonyms can be obtained through a lexical ontology, such as Wordnet[3]). Besides the domain concept(s), Mark might specify the type of document that the needed content originates from (such as notes, meeting minutes, reports, etc). The specified concept and the document type are internally transformed into a query, expressed, for example, in the SPARQL[4] RDF query language, that the will be used for searching the ACCM-based content repository. After Mark has finished the outline, the tool forwards the search queries (i.e., the queries formulated for each leaf of the outline) to the underlying CMS. After the search and retrieval processes have finished, the tool fetches the results from the CMS and present them to Mark. Mark then selects the outline's leafs, one after the other and the tool presents a list of retrieved content units for each selected leaf. For each content unit in the list, its metadata is presented – key concepts, the document it originates from, etc. By selecting a content unit from the list, Mark gets a preview of its content. In this manner he can select content units for each part of the outline and in that way assemble the whole document.

After the document is built, Mark can choose the output formats – he might want it in a regular document format (such as .doc, or .pdf), but also he might want to share it with his colleagues through the company's Wiki, so he would need it in the HTML format. The tool supports this as well: it offers export of the assembled document in multiple formats, where each delivery is based on the mappings between ACCM, as platform/tool independent format, and the desired tool specific format (doc, pdf, html). In addition, the tool stores the newly created outline (i.e., content aggregation) for future (re)use. It keeps a repository of content aggregation templates for those types of documents that users frequently create (e.g. monthly reports, work schedules, etc). So, when Mark has to create a monthly report for the next month, he will be able to retrieve an existing outline template (i.e. content aggregation) and then fill it in with new content. Of course, in order to be able to use the advantages offered by the described authoring tool, Mark has to upload his documents into the ACCM-based CMS which lies beneath the authoring tool and maintains the ACCM-based content repository. When uploading a new document Mark should provide a small set of the document's metadata - primarily, the document type (e.g., report, work agenda, meeting minutes), the creation date and the key concepts.

Mark is just an example of a knowledge worker, but all knowledge workers—whether they are historians, physi-

[3] http://wordnet.princeton.edu/

[4] http://www.w3.org/TR/rdf-sparql-query/

cists, financial advisors, or simply individuals researching a topic for personal reasons – face the problem of efficient knowledge location, retrieval and (re)use [1].

Tool support

As the above presented scenario illustrates, our aim is to facilitate the process of content authoring by providing users with the technology that enables them to reuse existing content in an efficient manner. Technically speaking, this goal requires the existence of a CMS that enables efficient storage, indexing, search and retrieval of not only complete documents, but also smaller chunks of content, i.e. content units as they are defined in the ACCM ontology. In other words, the system would have to manage a repository with ACCM ontology-aware content.

In order to have an ACCM-based content repository, we need to transform content from different platform/tool specific formats into the format compliant with the ACCM ontology. This is why we need tools we call disaggregators. They take as input content in any platform/tool specific format and convert it into ACCM ontology compliant output.

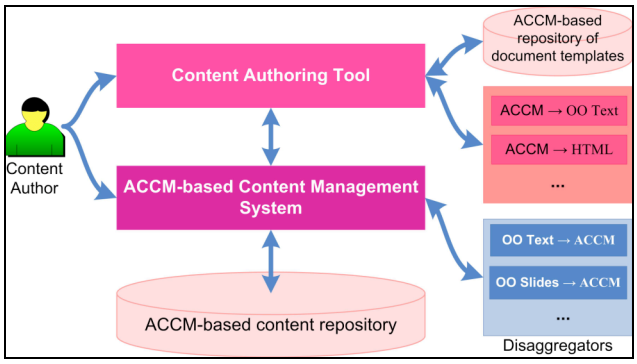

Figure 3. The core components of the proposed ACCM-based content authoring system

In addition, we have to provide users with technical support that facilitates the process of searching the content repository for a particular content unit – an image, a graphic, a table or a paragraph rather then for complete documents. The provision of such support is highly dependent on semantic annotation of the content available from the repository, where semantic annotation is considered as a process of tying semantic models and natural language content [2]. In other words, content units have to be annotated with concepts of appropriate domain ontologies. Therefore, we argue for content structuring according to the ACCM ontology as well as for its semantic markup compliant with the appropriate domain ontologies.

Mapping content into the ACCM model

Although we have not developed all the tools yet that would support the proposed authoring scenario, we have undertaken some initial implementation steps. Our current focus is on the tools for extracting/transforming tool-specific content into ACCM ontology-aware content, i.e. the tools that we referrer to as disaggregators.

Extracting ACCM ontology-aware content from the content produced by real-world authoring tools (e.g. MS PowerPoint, OpenOffice etc.) is a complex task since it must consider a huge variety of source formats and implementation techniques. Furthermore, it requires knowledge of the internal organization (i.e. structure) of the format we are mapping into the ontology-aware content.

So far, we have implemented mappings from OpenOffice (OO) slides and text documents into the ACCM ontology-aware content. Here we briefly explain how we perform the transformation of OO text documents. The transformation of OO slide presentations is done in a similar manner.

Decomposition of an OO text document into content units of the ACCM ontology is a two-step process: the first one consists of transforming the OO text document into a corresponding XML document which is parsed in the second step in order to extract its structure and content units (such as section, paragraph, and table) and represent them in accordance with the ACCM ontology. The original OO XML document is much more presentation than structure oriented, so it is very hard to identify and extract the document structure from it. Therefore, in the first transformation step we create an intermediary XML document that allows us to clearly identify the structural units of the OO document. Furthermore, it facilitates identification of formatting styles that were applied to each unit. This kind of information, when available, for example, helps us identify the structuring of the manuscript in terms of sections and subsections – without it, we would not be able (at least not without a substantial text analysis) to recognize whether the coming section is a subsection of the current one or it is a completely new section. Accordingly, the generated ontological representation of the OO document (the result of the second transformation step) captures even the most fine-grained structural units, such as table cells, table headings, footnotes and the like. To create ontology-aware content (i.e. instances of the ACCM ontology) out of extracted content units, we use Jena[5] – a Java-based framework for the Semantic Web – since it defines an extensive Java API for working with OWL ontologies. The resulting ontological instances of content units are loaded into an ACCM-aware repository of content units.

During the second step of the transformation process, besides creating ontological instances of the extracted content units, we also perform annotation of each content unit. As suggested in the above given scenario, we annotate a content unit with data about the document it originates from, the creation date and the key concepts. Annotations are stored as RDF representation of appropriate Dublin Core elements (e.g., *dc:author*, *dc:date*, *dc:subject*). The process of annotation is based on the approach we adopted in one of our previous projects [3] and which can be summarized as a combination of shallow text processing and a top-down approach in which metadata for describing content units of

[5] http://jena.sourceforge.net/

a document are derived from the metadata of the document itself. We are currently experimenting with different semantic annotation tools, such as KIM[6], in order to choose the one that we will use to automatically annotate content units with concepts of domain ontologies.

RELATED WORK

Our work on the development of ACCM as a platform neutral, abstract content model is related to the work conducted by the LTSC Computer-managed Instruction (CMI) Working Group in the scope of the RAMLET (Resource Aggregation Model for Learning, Education and Training) project[7]. This project aims at producing an IEEE standard that is supposed to facilitate interoperability among a variety of resource aggregation formats and specifications for learning, education, and training applications. The group is currently focused on providing mappings, in the form of an OWL ontology, which will allow for transformations between a number of specifications (IMS Content Packaging, METS, and IETF Atom have been identified so far). Whereas the RAMLET project is focused exclusively on educational resources, our focus is much broader, since the ACCM model and ontology are applicable to any kind of content, regardless of its intended usage. In addition, RAMLET is developing a generic model for structuring aggregates of assets, while ACCM is intended for defining the content and structure of a single asset.

ACCM model was partly inspired by the Abstract Learning Object Content Model (ALOCoM) [8]. ALOCoM has served as the basis for supporting learning content personalization[8] as well as learning content authoring[9]. Unlike ALOCoM which is an abstract content model for learning content, ACCM is applicable to any kind of content. Actually, ALOCoM can be considered as a specialization of ACCM for the e-learning domain.

CONCLUSION

In this paper, we have presented an architecture that allows for content interoperability among different, platform specific content models and provides means for organizing content by defining relationships among its content units. Such an architecture requires a generic content model, which we provide in the form of the ACCM model. In order to formally represent this model we have developed the ACCM-ontology. Content represented in accordance with the ACCM ontology can be easily enriched with ontological knowledge about specific domains (i.e. ontological metadata). That knowledge, along with the explicitly defined content structure, allows for scalable reuse of content across multiple contexts, which is one of our main goals.

We are currently working on the implementation of an ACCM-based content management system and are focused on content transformation from different tool-specific contents (e.g. OpenOffice, Word documents) to ACCM-ontology compliant content. Further, we plan to extend some existing authoring tools so that they can generate ontology-aware content and allow semantic markup of the ACCM components, using concepts from appropriate domain ontologies. Once we achieve a solid foundation of such content, we plan to work on the development of a recommendation system that will guide content authors through the authoring process. Finally, our plans also include the work on tracking the evolution of the ACCM-ontology aware content.

[6] http://ontotext.com/kim

[7] http://www.ieeeltsc.org/working-groups/wg11CMI/ramlet/Pub

[8] http://iis.fon.bg.ac.yu/TANGRAM/

[9] http://ariadne.cs.kuleuven.be/alocom/

REFERENCES

[1] Warren, P., "Knowledge Management and the Semantic Web: From Scenario to Technology," IEEE Intelligent Systems, January/February 2006, pp. 53-59.

[2] Cunningham, H., Bontcheva, K., Li,Y., "Knowledge management and human language: crossing the chasm," Journal of Knowledge Management, Vol.9, No. 5, 2005, pp. 108-131.

[3] Jovanović, J., Gašević, D., Devedžić, V., "Ontology-based Automatic Annotation of Learning Content," International Journal on Semantic Web and Information Systems, Vol. 2, No. 2, 2006, pp. 91-119.

[4] Priestley, M., "DITA XML: a reuse by reference architecture for technical documentation," In Proc. of the 19th Int'l Conf. on Computer Doc., Sante Fe, USA, 2001, pp. 152-156.

[5] S. Brodsky, "XMI Opens Application Interchange," IBM White paper, 1999.

[6] Obrenović, Z., Starčević, D., Selić, B., "A Model-Driven Approach to Content Repurposing," IEEE MultiMedia, Vol. 11, No. 1, 2004, pp. 62-71.

[7] Handschuh, S., Volz, R., Staab, S., "Annotation for the Deep Web," IEEE Intelligent Systems, Vol.18, No. 5, 2003, pp. 42-48.

[8] Verbert, K., Jovanović, J., Duval, E., Gašević, D., Meire, M., "Ontology-based Learning Content Repurposing: the ALOCoM Framework," International Journal on E-Learning, Vol. 5, No. 1, 2006, pp. 67-74.

[9] Nešić, S., Gašević, D., Jazayeri, M.,"An Ontology-Based Framework for Author-Learning Content Interaction," In Proc. of the 6th IASTEAD Int'l Conf. on WBE, Chamonix, France, 2007, pp. 359-364.

[10] [ACCM ontology] ACCM ontology, available at < http://www.inf.unisi.ch/phd/nesic/accm/>

[11] Gašević, D., Djurić, D., Devedžić, V., Model Driven Architecture and Ontology Development, Springer, 2006.

[12] Kealbling, L., Littman, M., Cassandra, A., "Planning and Acting in Partially Observable Stochastic Domains," Artif. Intell. Vol. 101, No. 1-2, 1998, pp. 99-134.