KNOWLEDGE-MASHUPS AS NEXT GENERATION WEBBASED SYSTEMS
Converging Systems Via Self-explaining Services

Thomas Bopp, Birger Kühnel
University of Paderborn, Germany

Thorsten Hampel
University of Vienna, Austria

Christian Prpitsch, Frank Lützenkirchen
University of Duisburg-Essen, Germany

Keywords: Converging systems, webservices, mediation, knowledge organization, digital repository.

Abstract: Webservice-based architectures are facing new challenges in terms of convergence of systems. By example of a web service integration of a digital repository/library, systems of knowledge management in groups, and learning management systems this contribution shows new potentials of flexible, descriptive webservices. Digital libraries are understood in their key position as searching, structuring, and archiving instances of digital media and they actively provide services in this sense. The goal of this article is to introduce services suitable for everyday use for coupling different classes of systems. Conceptually, the requirements of a possible standard in the area of convergence of knowledge management, digital libraries, and learning management systems are discussed. The results are publish and search services with negotiation capabilities with a low-barrier for adoption.

1 INTRODUCTION

Mashups - the integration of different web-based services and tools is one of the key challenges of our modern information society - not only the Web 2.0 development. Thus, a big challenge for the information society lies in the combination and cooperation of different tools, systems and services. In this context service oriented architectures (SOA) play an important role and show that the understanding of computer systems as monolithic applications changes towards flexible, multi-layered service infrastructures. The emphasis of such systems is the interaction between different web-based services using established standards. A common base for interoperability are web services that allow integration of services through an asynchronous, web-based protocol (SOAP). Web services are self-explanatory and offer functionality on a high abstraction level.

A critical look on existing software architectures however shows a different picture than the one described above. Service-oriented architectures are used within certain classes of systems, but there are only a few interoperable services between different classes. Most of these services offer one-way reading capabilities (e.g. search services of Amazon and Google), but do not allow to write (the state of the system is not affected). On the other hand, if a system offers a broad range of functions, it lacks mechanisms for a flexible integration of those services, because there is no standardized API for most systems. Therefore, the idea of SOA respective web services allowing the flexible and scalable integration of web-based services is not achieved in todays architectures.

Starting from scenarios of interoperability (Bopp et al., 2006) of digital repositories, knowledge management systems, and planning systems, we understand a digital repository as a persistent location for documents. The repository provides document (search) and allows their storage (publish and archive). Our research project mistel deals with convergence of systems and ways to establish new standardized web services between the system classes of knowledge management, digital repository and planning system. The sample implementation connects DuE-Publico, sTeam, and ELM, but our approach is not bound to these specific systems. Moreover, we identified different requirements and restrictions to be able to incorporate as many systems as possible. Therefore, it is important to interoperate on different levels and to find the minimal common base for such a connection. In this context for example a keyword search serves as a minimum search capability a system must offer for a service provider, as well as a service client. In order to match the different capabilities of the connecting systems we introduce a negotiation phase, which serves as an additional explanation for web services. Based on the results of this phase the service client may decide in which way it contacts the service provider. Therefore, mistel creates a new unterstanding of web-based integration of services which can be described as a mixture of low level mashups and high level web service integration. Our web services are flexible enough to communicate different levels of interoperability. On the other hand they are easy to use (such as mashups) to be integratied in most application out of the knowledge management/digital library domain.

2 SERVICE ORIENTED ARCHITECTURE

The Web 2.0 movement is mainly identified by new social ways of user-centred forms of semantic structuring of web content. Here, tags and folksonomies [1] stand for social forms of contextualization and organization of the web. Simultaneously, a new generation of successful web based tools, such as Flickr, YouTube, or MySpace implement the ideas of user-generated content and show the user new possibilities of powerful ways of interaction on the net (for example rich-client user interfaces such as Flickr).

Even more important in our understanding is the paradigm seeing an application or tool not as a single application in web 2.0, but understanding web based applications as flexible services. Unfortunately this idea is so far only focused on a small number of applications such as the above-mentioned tools. The broad spectrum of existing E-Learning environments, knowledge management, planning systems and digital libraries are still separated in its system classes with only very limited and inflexible possibilities for mashup.

Nevertheless, E-Learning-systems and infrastructures nowadays have reached a high standard. At the same time, it becomes obvious that the interaction between different classes of systems and an integration of other classes into an E-Learning-system becomes the critical task of for the next generation of cooperative tools on the web. In the example of a university-wide infrastructure, E-Learning-infrastructures span a number of service providers and service-consumers, and are highly process oriented. In this context various systems from study organization, examination administration, to planning/ authoring systems, and knowledge organization are involved. Here, digital repositories take a special role, because previously media as part of learning systems was limited to the specific system. We see that the above-mentioned idea of Web 2.0 as a mashup of different services and tools is not only limited to classical tools and services on the web nor is limited to new business models for services on the web. The idea of service integration is the crucial task for todays service architectures, such as the architecture for organizing knowledge as part of a university infrastructure. The same ideas can be transferred to most infrastructures of larger organizations.

The mistel approach solves the problems of *knowledge-SOA* from that end. With the new paradigm of service oriented architectures there is the possibility of opening up digital repositories for new scenarios of use. In this context, the consequences of a coupling between digital repositories and cooperative learning environments have to be explored. A digital repository is regarded as a natural partner in finding, structuring, and persistently archiving digital media. Previously, it has been shown by scenarios (Bopp et al., 2006) how services of a digital repository can be integrated into modern forms of cooperative knowledge management. The focus of this integration lies in the combination of different systems in a standardized way. Such a coupling must fulfil a number of requirements, mostly in the negotiation of the provided services.

As part of the next chapter we will show how the above-mentioned integration of different classes of systems as a knowledge-SOA, a mashup of services, is accomplished. Here, new forms of flexible web services have to be developed.

[1]The term folksonomy is generally attributed to Thomas Vander Wal.

3 DIGITAL LIBRARIES, REPOSITORIES AND LEARNING OBJECTS

The presented approach to interoperating systems assumes a lose coupling of service-oriented architectures. Its central service provider is the digital repository/digital library, which provides access to documents and collections of documents. In this context, we distinguish between single documents (standalone), derivatives of a document, and collections of documents. In general, documents are not isolated, but there may exist connections between different documents. For example, one document might be a version of another document in a different language (this is a derivative of the original document). Another example are several HTML-document including images and stylesheets, which all together describe a single topic and thus can be considered a single document. This is important, when searching for documents, because such a collection should be displayed as a single search result. Moreover, it must be retrieved as a whole, because of the dependencies between the contained documents.

Starting with scenarios of inquiry, and publishing, the search for documents should enable a user to find appropriate documents, collections, and derivatives. Thus, if there are several derivatives found of a document, the user must be able to identify those documents and choose one of them (for example in a language the user is able to read and in a format that can be processed). Technically, the system must also be able to display collections of documents and to import those documents as a whole.

Besides the digital repository/library, Learning Object Repositories (LORs), Learning Management Systems (LMS), Knowledge Management, and Planning Systems have been considered. Digital libraries are able to store any kind of document, but LORs are focused on learning objects and often restricted to a limited set of learning object formats and metadata formats (Neven and Duval,). They are capable of consistently extracting parts of SCORM-Learning-Environments (Dodds, 2004) out of their contexts. Searching for documents is usually done as a keyword search, which is also the easiest method.

LMS differ from digital libraries/repositories by their editing capabilities for learning objects. The organization in a LMS is typically hierarchical and can technically be seen as a classification.

The requirement for providing standardized and open interfaces is derived from the interconnection and the task sharing of the participating systems. Even a simple life-cycle of a learning-unit from creation in the planning/authoring-system, transfer to a LMS, and finally to archival storage in a digital repository/library already shows a couple of required interfaces. This also includes search functionality for finding learning units in all those places.

Currently, there are only a few existing solutions, which can be considered for interconnecting different classes of systems. The standardized import and export is limited to the context of E-Learning with SCORM (Edutella (Nejdl et al., 2002), Ariadne GLOBE (Simon et al., 2005), and ADL CORDRA (Rehak et al.,)), which is not suitable to transport any kind of document. There are no standardized interfaces for transferring documents between arbitrary systems.

The standard OAI-PMH (Lagoze and de Sompel, 2001) of the Open Archives Initiative defines how metadata of digital repositories can be exchanged. However, an exchange of documents itself is not specified. Moreover, it is not suited for usage within E-Learning, because metadata formats like Dublin Core are not applicable because of missing didactic details.

Some digital libraries/repositories like Aleph (Ex Libris), or the open source system DuEPublico offer proprietary interfaces for distributed searching in several installations of these systems. These interfaces are not appropriate for searching in different repositories, because they are arranged around the peculiarities of the specific system. Importing documents is not possible with any of the described systems.

In conclusion, there is an absence of standardized interfaces between different classes of systems. Existing standards are not adequate in terms of openness for using them in the context of the three classes described above. Thus, we can derive requirements towards an open architecture: It needs to be independent from the organization of documents within a repository, and must not depend on a certain metadata standard. Moreover, it should be easy to implement and be able to adapt to different requirements. Therefore, it should be possible to describe the capabilities of a system/service.

4 INTEGRATION OF SERVICES

The goal of our work is to achieve integration of services between different systems, which obviously requires a common basis. Here, the starting point is searching through a digital repository with its documents. In contrast to other functions search is a public service of a repository, which generally does not require authentication of users (only for certain documents or areas).

1 illustrates the coupling of the different systems.

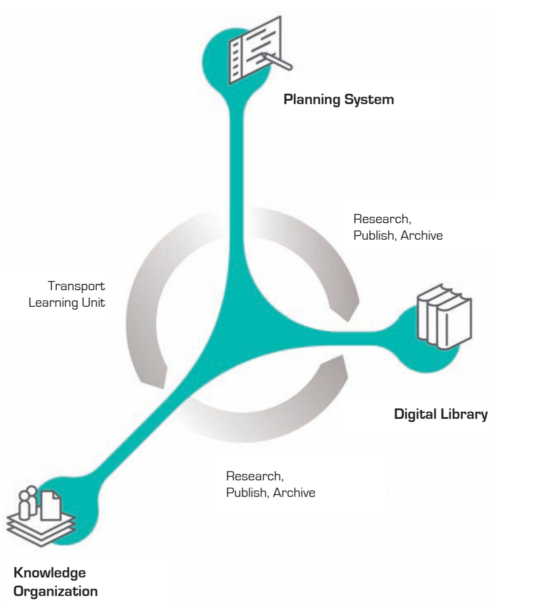

Figure 1: Coupling Knowledge Management, Digital Library and Planning System.

Initially, we identified the most important services between the systems as publish, search, and archive in the digital library. The only connection between planning system and knowledge management is export/transport of learning units.

The integration of services was initially implemented as a prototype to attain technical expertise and test the first version of our specification. The following substantial characteristics have been considered in this first version:

- Simplicity: The service needs to be simple in use in order to integrate a large number of systems. An extension of the functionality should be possible with different layers of complexity. A low-barrier solution is very important for widespread adoption of the specification.
- Openness: Existing standards need to be considered. This includes transmission protocols, encryption and compression methods, and metadata formats.
- Adaptability: If possible, the absence of certain features should not result into the exclusion of a system. In fact, a common base should be found that enables communication between systems.

Technically, web services offer a basis for a loose coupling of systems, which meets all the requirements described above. Especially, it must be pointed out that the implementation of the services guarantee the openness of the system and do not use any specific functionality of any of the systems. The negotiation between the system's capabilities is the central quality of our approach. This includes the goal to guarantee communication between service provider and service requester by finding a common basis for the participants. The solution for this is a meta service which encapsulates different services and standards, and provides additional information about the individual services. This meta service handles the communication in three phases. In the first phase the service requester connects and authenticates (the authentication is handled within the SOAP header).

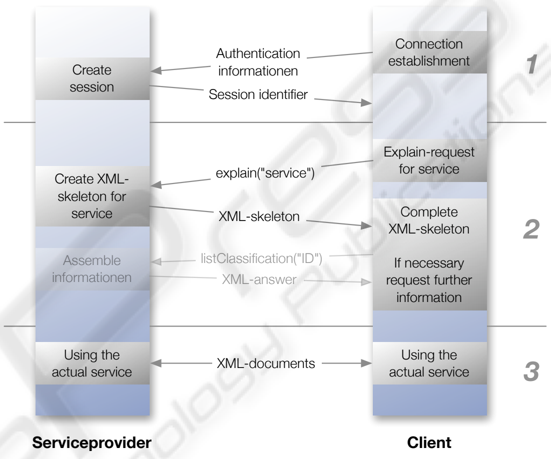

Figure 2: 3-phase communication-structure between client and service-provider.

In the second phase the interoperability is negotiated by examining the capabilities of the participating systems. Finally, in the third phase a service is called based on the interoperability observations from the previous phase. The actual call of the service might take several communication steps.

The negotiation and description of the provided services takes place in the second phase. This is characterized by an explain-function, which is similarly part of the SRW specification (SRW, 2005). The SRW-explain method solely describes the SRU-Server, e.g. the Endpoint URL of the web service and its port number, the supported SRW version, or the name and description of the server. Unlike this procedure, the explain method, which is introduced in this article, is strictly related to the individual services. For each of the provided services all parameters are described by an XML description, which basically matches the syntax of a query for this service. For example, a search query includes the format of the metadata for the result of the query and the classifications used for searching.

The following figure 2 illustrates this 3-phase communication pattern between the client, wanting to use the service and the service provider. The description and negotiation takes place in the second phase by using an XML-skeleton as the exchange-dataformat of the explain-method.

4.1 Search

As a common basis for the integrated services each repository allows certain types of search queries. Usually, this includes the classical keyword search, which is the easiest possible type of query in terms of usage and implementation. Such a simple search does only return unspecific results. Thus, it is advantageous to support a simple syntax as well as one or more complex query syntax.

In general, we have to distinguish between the query which is send from the client to the service and the result of this query. Both have to follow certain specifications and standards. Therefore, a system has to support different syntaxes for the query and metadata standards for the result. A well established standard for metadata is Dublin Core (Dublin Core Metadata Initiative, 2005), which only supports a small set of metadata (simple DC) compared to LOM (IEEE, 2002). Therefore, it can be considered a simple standard, which is easily supportable by any application. Moreover, it is possible to find a transformation from the metadata of knowledge organization and planning system to Dublin Core. Typically, a digital repository uses much more metadata to describe a document than other systems.

There are already a few existing standards for search queries. For example, SRW (SRW, 2005) specifies searching for documents with paging and assembly of the result. Our approach is to use such standards within our service layer and to offer several possibilities for searching, retrieving and publishing documents. Therefore, it is necessary to negotiate which methods and standards to use when calling a service.

To start the negotiation process for the search-function, the user's client has to call the function *explain* which is provided as part of our mistel-web service. The explain-function requires the name of the function, the client wants to get information about input parameters, e.g. *search* for the search explanation. The code listing 3 shows the exemplary XML-skeleton code that the client receives upon the call of *explain(„search“)*. The XML response uses some syntax expressions, which are freely adopted from the XML Schema Definition (XSD). The code in listing 3 offers two different search methods (*keyword* or *SRW*) which are encapsulated by the tag *(*method). Within the method tags you will find tags for all necessary parameters for the respective method. The *choice* tags are derived from XSD syntax and mean that the client can choose between the two different methods *keyword* or *srw*. The attribute *id* denotes the name of the tags, the choice should be applied to. The remaining two attributes *minOccurs* and *maxOccurs* describe the amount of tags the client has to choose. In the search example, the choice tag for *method* has *minOccurs=1* and *maxOccurs=1* which means, that the client has choose at least one method but also not more than one i.e. exactly one of the two available methods. The choice tag is always used when the client has the possibility to choose from different available options.

A keyword-search needs at least one keyword to search for, which is wrapped by the *query* tag. This tag has the attribute *use="required"* which means the client has to provide the information within this tag. The following *choice* tag offers the different metadata formats for the search result, that are available. In the example the client has to choose exactly one of the two possibilities. The block of *querymeta* is labeled by *use="optional"* meaning that all information grouped within this tag are not necessary. The optional information are the maximum number of results (*maxresults*), the offset for the start of the resultset (*startresults*), the direction to sort the results ascending or descending according to a given field (*sortby*) or a transformation stylesheet for displaying the resultset (*transformation*). This stylesheet is determined by a specific URL, where it is stored. The second method, the SRW-search only needs a SRW-query as input, because all further information are encoded in this query.

The whole explain-XML-code of the search-method uses the textstring ### as a placeholder for content, the client has to add to the marked places in the XML-skeleton. A method *getExplainPlaceholder* can be called to query this character string.

4.2 Publish

Despite the fact that there are a lot of different standards for searching documents, there is still no standard how to publish documents to digital repositories. Due to that, we propose a method for publishing documents in this section which obviously has some similarities to a search. For publishing, various parameters have to be negotiated:

- What should be published?
- Where to publish it?
- Which metadata describe the document?
- Who publishes?

Metadata as well as the target are important for publishing and searching. Additionally the document to

```
<search xmlns="http://www.systemkonvergenz.de">
    <choice id ="methods" minOccurs="1" maxOccurs="1">
        <method name="keyword">
            <query use="required">###</query>
            <choice id="classifications" minOccurs="0" maxOccurs="unbounded">
                <classification id="###" type="PATH" minOccurs="1" maxOccurs="1"/>
            </choice>
            <choice id="metadatas" minOccurs="1" maxOccurs="1">
                <metadata format="dc"/>
                <metadata format="rdf"/>
            </choice>
            <querymeta use="optional">
                <maxresults use="optional">###</maxresults>
                <startresults use="optional">###</startresults>
                <choice id="sortbys" minOccurs="0" maxOccurs="1">
                    <sortby order="asc">###</sortby>
                    <sortby order="desc">###</sortby>
                </choice>
                <transformation use="optional">
                    <choice id="locations" minOccurs="1" maxOccurs="1">
                        <location type="http">###</location>
                        <location type="soap-attachment">###</location>
                    </choice>
                </transformation>
            </querymeta>
        </method>
        <method name="srw">
            <query use="required">###</query>
        </method>
    </choice>
</search>
```

Service: Keyword-Search

Classifications
List of available classifications for search

Metadata formats
List of available metadata-formats for the answer

Query-Metainformation
Pagination-, sort- and transformationinformation

Service: SRW-Search
Only needs a SRW-Query

Listing 3: XML-Code of the explain for the search-method “search”.

be published needs to be transferred to the service of the digital repository/library. In this context the format of the document is important: is it a single document without any further relations, a collection of documents, or a derivative of an existing document.

The transfer of documents also needs negotiation between the service provider and service consumer about the supported protocols. There are several different protocols for document transfer including http, ftp, webdav and soap. Again, those standards are used within our services instead of inventing new technologies for upload and download. Generally, we distinguish between push and pull of documents. The service provider might download the document, which is provided by the service consumer at some location, or might provide a location for uploading the document. A SOAP attachment is useful for publishing small documents, because this method lacks streaming functionality. When a document is published it also needs to be classified within the structure of the repository. Information about the target system is required, which explains the used classifications and metadata formats. Then, it is possible to choose a classification and metadata format.

In order to receive the XML-skeleton of the required parameters of the publish method, the client has to call the explain method of the web service using the parameter *publish*. The given listing 4 shows a possible example of a XML-skeleton received from the explain method. The structure is quite similar to the one shown in listing 3 and also uses *choice*-blocks and *use* attributes.

In general, one can divide a publish process into three different possibilities: adding a new document, changing or overwriting existing document or deleting one. These three options are grouped into the *actions* choice-block, whereas the *minOccurs=0* indicates, that no action is also allowed. This implies, that the action of adding a new document is used as default action. After that, the client has to specify the kind of document he wants to upload. In the example he has three different types (*document, collection, versions*) from which he has to choose exactly one. The type *document* is used for a single file (e.g. a PDF document), whereas *collection* and *versions* stand for a package of several files. The type *collection* should be used for packages of files, that need a special conversion or treatment on the server, because they consist of different types of files (e.g. SCORM packages). In contrast, *versions* should be used for packages that contain different formats of the same file (e.g. a german and an english version of a document).

After that, the *source* block starts, that contains all the necessary information about how to receive the document that should be published. Most of these information are of course required. The *packaging* block provides the client with a list of available packaging or compression formats that the server accepts. Within the optional *mimetype* tag the client may add the specific mimetype of the document he wants to publish. The following choice block of locations gives a list all possible types of transportation that are supported by the server and the client has to choose exactly one and specify the location where to find the document he wants to publish. Additionally he is able to use the *checksum* tag to add a MD5 checksum to test the data integrity after transfer.

The remaining tags are used to describe the properties the new document should receive in it’s new environment after publishing it. Essential is that the

```xml
<publish xmlns="http://www.systemkonvergenz.de">
    <choice id="actions" minOccurs="0" maxOccurs="1">
        <action id="###" operation="add"/>
        <action id="###" operation="delete"/>
        <action id="###" operation="update"/>
    </choice>
    <choice id="types" minOccurs="1" maxOccurs="1">
        <type>document</type><type>collection</type><type>versions</type>
    </choice>
    <source use="required">
        <choice id="packagings" minOccurs="1" maxOccurs="1">
            <packaging>zip</packaging><packaging>tar</packaging>
        </choice>
        <mimetype use="optional">###</mimetype>
        <choice id="locations" minOccurs="1" maxOccurs="1">
            <location type="http" method="GET">###</location>
            <location type="soap-attachment">###</location>
        </choice>
        <choice id="checksums" minOccurs="0" maxOccurs="1">
            <checksum algorithm="md5">###</checksum>
        </choice>
    </source>
    <name use="required">###</name>
    <choice id="metadatas" minOccurs="1" maxOccurs="1">
        <metadata format="dc">###</metadata>
        <metadata format="rdf">###</metadata>
    </choice>
    <choice id="classifications" minOccurs="1" maxOccurs="unbounded">
        <classification id="###" type="ROOM" minOccurs="1" maxOccurs="1"/>
    </choice>
    <choice id="permissions" minOccurs="1" maxOccurs="unbounded">
        <permission group="###" id="r" user="###"/>
        <permission group="###" id="w" user="###"/>
        <permission group="###" id="x" user="###"/>
    </choice>
</publish>
```

Add / Delete / Update
Specification of the operation

Distinction of the type
Currently in three different ones

Specification of the source

Packet formats
Supported formats

Mime-type of the document

Source of the document
Could be available as download or SOAP attachment

Checksums
For verification of a correct transmission

Name
For the document

Metadata formats
Which are supported by the target system

Classifications
For classifing the new document

Permissions
For the new document on the target system

Listing 4: XML-Code of the explain for the publish-method “publish”.

document acquires a *name* on the server. The client is able to add a list of metadata information to the published document and should add at least on classification from the given list. The value of the *maxOccurs* attribute is *unbound*, which means, that the maximum number of classifications is not restricted. Each classification itself is also labeled with the bounding attributes which might allow, that one type of classification is used several times with different values. By using the mistel web service, it is possible to get more detailed information about one classification type. With the concluding choice block the client should determine the permissions for the document. The server lists all available permissions which are specified by an unique ID, e.g. *r*, *w* and *x* and the client has to set these accordingly to the desired state.

5 EVALUATION

The focus of this paper is to give a good overview of the proposed new mistel web service integrating different classes of systems of the knowledge management and digital library domain. Hence, we are not able to present our sample application in great detail. Currently we integrated the mistel web service in our mediarena composer application, which is a powerful cooperative knowledge management system (Niehus et al., 2006) (see http://flywheel.open-steam.org/). As part of using this tool for the cooperative structuring of so called virtual knowledge spaces users are able to search for relevant documents in various digital libraries directly from the mediarena application. As proposed in this paper mediarena also allows the storage of documents (e.g. learning materials) in the digital library. Our first user experiences show that the seamless integration (reducing discontinutes in the use of digital media) is a great success - even on a relatively low level of supported search functionality. - As part of the Web 2.0 movements mistel proves, that the flexible integration of web-based services inflates a great potential on user centred tools and services.

6 RELATED WORK

Melnik et al. choose a different approach to couple services of digital libraries (Melnik et al., 2000). An infrastructure for mediation between the services is used to translate the queries (in this case search-queries) in order to make them understandable for the service provider. Therefore, it is possible to couple a client with various target systems, that use different formats and protocols.

The eduSource Project (Hatala et al., 2004) follows the IMS DRI speficiation (Global Learning Consortium, 2003) and in this context implements ECL (eduSource Communication layer) as a communication layer. Besides this ECL protocol a gateway framework is provided to include existing protocols and formats. Here, it is distinguished between different layers, which include the communication protocol, the query language (for example ECL or OAI), and the metadata. These layers are basically quite similar to the different phases of negotiation described in this work.

7 CONCLUSION

This article describes the integration of different services into a unified framework, which focuses on an extended descriptiveness of the services. This enables a loose coupling of services with negotiation about formats and protocols. Our specification does not dictate the usage of a specific standard, but suggests the intersection of available metadata fields, query standards and classifications. In order to guarantee a communication between service provider and service consumer each service needs to support the most simple methods like keyword search and simple Dublin Core. On this basis interoperability is achieved between services of different systems.

Apart from the negotiation capabilities of our service specification, we introduced a method for publishing documents. Minimum requirements are a small set of metadata, the name of the document, and the document itself. The transfer of the document data is accomplished using standard protocols like http or ftp. Furthermore, the metadata can be negotiated between service provider and service consumer.

The combination of search and publish services successfully enables a loose coupling of digital libraries, knowledge management systems, and planning/authoring systems. Technically, publish and search is similar to exporting and retrieving documents. Thus, the minimum requirement of exchanging documents is fulfilled with the described specification. With dynamic negotiation of communication between service provider and consumer we also met the requirements of simplicity, openness and adaptability.

ACKNOWLEDGEMENTS

The project is funded by the *Deutsche Forschungsgemeinschaft* (DFG).

REFERENCES

Bopp, T., Hampel, T., Hinn, R., Ltzenkirchen, F., Prpitsch, C., and Richter, H. (2006). Alltagstaugliche mediennutzung erfordert systemkonvergenz in aus- und weiterbildung. In *E-Learning - alltagstaugliche Innovation?*, volume 38 of *Medien in der Wissenschaft*, pages 87–96.

Dodds, P. (2004). Scorm 2004 overview. Technical report, Advanced Distributed Learning (ADL).

Dublin Core Metadata Initiative (2005). Dcmi metadata terms. http://dublincore.org/documents/dcmi-terms.

Global Learning Consortium (2003). Digital repository interoperability - core functions information model. http://www.imsglobal.org/digitalrepositories/driv1p0/imsdri_infov1p0.html.

Hatala, M., Richards, G., Eap, T., and Willms, J. (2004). The interoperability of learning object repositories and services: Standards, implementations and lessons learned. In *Proceedings of the 13th international World Wide Web Conference on Alternate Track Papers & Posters*, pages 19–27, New York, NY, USA. ACM Press.

IEEE (2002). Draft standard for learning object metadata. http://ltsc.ieee.org/wg12.

Lagoze, C. and de Sompel, H. V. (2001). The open archives initiative: building a low-barrier interoperability framework. In *Proceedings of the 1st ACM/IEEE-CS joint Conference on Digital Libraries*, pages 54–62, New York, NY, USA. ACM Press.

Melnik, S., Garcia-Molina, H., and Paepcke, A. (2000). A mediation infrastructure for digital library services. In *Proceedings of the Fifth ACM International Conference on Digital Libraries*, pages 123–132.

Nejdl, W., Wolf, B., Qu, C., Decker, S., Naeve, A., Nilsson, M., Palmer, M., and Risch, T. (2002). Edutella: A p2p networking infrastructure based on rdf. In *Proceedings of the 11th International Conference on World Wide Web*, pages 604–615, New York, NY, USA. ACM Press.

Neven, F. and Duval, E. Reusable learning objects: A survey of lom-based repositories. In *Proceedings of the 10th Conference on Multimedia*, pages 291–294.

Niehus, D., Hampel, T., and Sprotte, R. (2006). medi@rena: An eclipse based rich client application for open-steam and its real world usage. In *Proceedings of the World Conference on Educational Multimedia, Hypermedia and Telecommunications*, pages 1304–1309.

Rehak, D., Dodds, P., and Lannom, L. A model and infrastructure for federated learning content repositories. In *Proceedings of the 1st Workshop on Interoperability of Web-based Educational Systems*.

Simon, B., Massart, D., van Assche, F., Ternier, S., Duval, E., Brantner, S., Olmedilla, D., and Miklos, Z. (2005). A simple query interface for interoperable learning repositories. In Simon, B., Olmedilla, D., and Saito, N., editors, *Proceedings of the 1st Workshop on Interoperability of Web-based Educational Systems*, pages 11–18, Chiba, Japan. CEUR.

SRW (2005). Srw/u - search / retrieve web services. http://www.loc.gov/z3950/agency/zing/srw.