

A Metadata Registry For Community Driven E-Learning Sites

Valentin Zacharias
ontoprise© GmbH
Amalienbadstr. 36 (Raumfabrik 29)
76227 Karlsruhe
zacharias@ontoprise.de

Abstract

We present the architecture and the interface of a metadata registry for a large e-learning site. The metadata registry is very simple to integrate by content and application providers and thereby tries to motivate more members of the community to contribute. It takes its inspiration from currently successful Semantic Web architectures and aims to be an evolutionary change to the web – using long established standards where possible.

1. Introduction

UNITRACC is an internet based e-learning system for the area of canalization. Currently containing content supplied by the creator of unitracc, it is developed to become a community driven platform; enabling users to recombine the contents of the site and growing from content added by the users. Already unitracc makes most of the authoring tools for knowledge entities available to teachers using the site. By uploading their content teachers can profit from the e-learning tools embedded in unitracc and can easily combine their teaching materials with the large unitracc content repository – containing not only digital versions of standard textbooks but also thousands of multimedia files. We believe, however, that not all potential content providers will think its worthwhile to spend time uploading data and that some will not want to store their content in another location – making it necessary to update it on every change to the original and causing all kinds of copyright headaches. For these content providers we plan to provide another, simple and low threshold interface to contribute content to the unitracc platform. True to the community idea the interface is not only for adding content to unitracc, but also allows for simple, low threshold access to the

stored metadata – enabling others to profit from the metadata and hopefully increasing their motivation to contribute. This paper describes the Semantic Web inspired ideas and the implementation of this interface.

This papers starts by giving an overview of the current scope and architecture of the unitracc platform. It gives a short description of the Semantic Web and describes three Semantic Web technologies that are already in widespread use and that illustrate the architectural ideas behind the unitracc metadata repository. It continues with a description of the interface to the registry and the implementation. It concludes with an overview of other approaches combining Semantic Web and e-learning and a look at the current state of the implementation and plans for future work.

2. Unitracc

Unitracc, the “**Underground Infrastructure Training and Competence Center**” is a internet based e-learning and information system for the area of canalization. Unitracc also contains a collection of web based tools that help public authorities manage and monitor underground infrastructure. The system already contains a large number of information units, especially enhanced digital versions of two standard textbooks about canalization.

Unitracc is developed by the company Prof. Dr. Ing. Stein & Partner GmbH, a leading engineering firm whose founder is also the author of many standard works of technical literature. The development has been funded in part by the German Federal Ministry of Education and Research. Access to unitracc is available on a subscription basis, the target audience ranges from beginning trainees and their teachers to architects.

The information content in unitracc is stored in

small “infobits”, each made up of some data and associated metadata. The data may be a short paragraph of text, a picture, a movie or larger multimedia applications like virtual construction sites or process simulations. Infobits are combined using so called “scripts”, a script is also an infobit and can be combined using other scripts. Scripts also contain ordering and layout information. The application was designed such that each information bit can be reused in as many different contexts as possible. The unitracc platform includes web-tools tailored for teachers to create their own multimedia teaching materials reusing existing infobits. The unitracc platform supports teachers in managing classes, exams and learning progress.

The metadata structure was developed based on ideas from Metadatastandards like LOM[1], or IMS[2] and strived to be compatible to the LOM/SCORM[3] standard which wasn’t finalized at the time the unitracc architecture was developed.

The metadata of infobits fall into three different categories:

1. Metadata for internal administration, like versioning information or size and format of multimedia data.
2. Didactic metadata allows to associate information with areas of interest, occupation groups and the learning level of the user. For example didactic metadata classifies an infobit according to its difficulty, area of interest (occupational safety, law, environment ...) and kind of information (calculation, history, construction site picture..).
3. Content metadata describe the actual content of an infobit. Content metadata consist of keywords from a keyword list with approximately 5000 entries that can be combined with a few hundred adjectives. Currently the list of content keywords is shortened and organized with Sub-/Supertopic relations. The content metadata category also contains “needed” relations between infobits. These relations state that one infobit must not be displayed without another one, a simple example would be a picture that should not be displayed without a caption.

More Information about the unitracc system can be found in [4][5].

3. Semantic Web

The idea of the Semantic Web[6] is described in the

Semantic Web Activity Statement of the W3C[7]: “The Semantic Web is a vision: the idea of having data on the web defined and linked in such a way that it can be used by machines not just for display purposes, but for automation, integration and reuse of data across various applications”. Or even shorter: The Semantic Web aims to make the Internet more user friendly by making it easier to understand for machines.

Much discussion in the academic community centers around the choosing formalism and architectures for Semantic Web applications based on the trade off between performance and expression power, we however chose to select our architecture by looking at Semantic Web applications that are already widely adopted, hoping that we can replicate their success factors. In the following we describe three particular successful Semantic Web architectures in order to explain why our meta data registry looks the way it does.

3.1 RSS

RSS or “Really Simple Syndication” is a family of protocols for webpage content description in a way that allows for simple, automated syndication.

At the center of the RSS are sites that have an interest in having their being syndicated. These sites post a file in a specific format – an RSS file – on their webpage. In this file there is a definition of one or more channels. Each channel contains some items and their description, consisting at least of a title and a link. A frequent use case is a RSS file on a Webblog page, containing one channel having the same name as the blog with the items being the titles of and links to the blog entries of the last couple of days.

By all measures RSS is a huge success: more than 400000 sites publish RSS feeds[8], the Safari, Opera and Firefox browsers come with RSS support, Apples iTunes works with one RSS format and the next versions of the Internet Explorer and the Windows operating system are expected to support it. There are many factors that contributed to RSS’s success (see [9] for a more detailed discussion), but the most important ones are probably simplicity and “web friendliness”. RSS is a very simple format: a computer scientist familiar with XML or even an amateur used to HTML can immediately read a RSS file with a text editor, understand what it means and change it to reflect the information she wants to publish. RSS is “web friendly” because its overall architecture is very similar to the traditional HTML-web. A RSS file is just like a

HTML file on a server, if a webservice allows for HTML files it will allow for RSS files. It can be uploaded using the usual FTP or WebDAV applications or can be created dynamically using one of the widely adopted scripting techniques like PHP or Python.

3.2 microformats

Microformats are a set of simple, open data formats for inclusion of structured data into webpages; they are built upon existing and widely adopted standards. Microformats exist for information about calendars and events[10], social networks[11], license information[12] and more; the most famous is the "relTag"[13] format for tags, keywords and categories. The relTag format consists of just a html link with the attribute rel="tag", giving the subject of an information unit. So for example including `Life8 ` in a webpage annotates this webpage with the tag Life8. In this architecture there is no explicit schema, everybody is free to use any tag she chooses.

Despite or rather because of its extreme simplicity the relTag microformat is to date the most successful annotation format on the web. The number of sites that publish information according to this standard is estimated to be in the millions[14], a popular tag like "Life8" is used to annotate 15000 pages.

3.3 Representational State Transfer, REST

Representational State Transfer (REST)[15] is an architectural style for distributed hypermedia systems like the world wide web. REST's proponents argue that a few key design ideas have enabled the web to be scalable and grow to its current size. They argue that these ideas can be applied to find better solutions to problems that today are usually solved using message oriented RPC architectures.

These key design principles are:

1. A stateless client server protocol
2. A set of well defined operations that apply to all pieces of information. HTTP itself defines a small set of operations, the most important of which are GET, POST, PUT and DELETE.
3. A universal syntax for resource-identification. In a system with a REST architecture, every resource is uniquely addressable through the resource's URL
4. The use of hypermedia both for application

information and application state transitions

The following simple example for a REST api is taken from Bloglines¹ – a service that can be set up to monitor and aggregate information from different news sources. The Bloglines REST api defines that a call to an URL like `http://rpc.bloglines.com/getitems?s=SUBID&n=1` (where SUBID is the id of the account that is accessed) returns an xml file with all unread news items.

Yahoo and eBay are among the many internet sites that offers REST interfaces. Amazon, that offers both SOAP and REST interfaces, observes that 80% of the actual request are made using the REST apis[16].

4. Unitracc metadata registry

Starting from the above described observation we are currently building a metadata registry with the central goal of simplicity for content providers. The registry tries to encourage contributions by not only making uploading simple but also by making it easy for everyone to profit from the contributions to the community.

The metadata registry stores the textual content and the annotations of information units. The information units are annotated with topics from a topic hierarchy that is also maintained by the metadata registry. The registry offers a REST api to change the taxonomy. Each topic is identified by a unique URL that can also be used to access and change information about this topic, again determined through a REST api (described in section 4.1 and 4.3).

There are three different ways to add documents and their metadata to the registry: Sites within the community will be crawled periodically and pages that contain references to topics in the taxonomy will be added to the index. The second possibility is to directly alert the registry to the existence of a new page through a POST request (described in section 4.2). The last possibility is to change the stored document data directly by sending a simple XML files to the registry (described in section 4.4).

Each document added to the registry also has a unique URL in the unitracc domain that can be used to access and change information about this document.

The unitracc metadata registry also offers a quite sophisticated search function (described in section A).

¹ <http://www.bloglines.com>

4.1 Getting information from the registry

All URLs for the topics are of the format `http://unitracc.de/topic/[topicID]`. These URLs are also a starting point when interacting with the registry. Calling such an URL returns a HTML page describing the topic, its name, a description and super- and subtopics. HTTP get parameters can be used to augment the registries response:

- “format” with possible values `html` and `xml` allows to get versions suited for human or computer processing.
- “docs”, with an integer value asks the registry to return a number of documents annotated with this topic. “from” can be used to give an index where the list begins. “search” can be used to search in the documents annotated with the current topic for a search string.
- “news” returns a RSS file with recent changes to this topic and new additions. This parameter cannot be combined with “format” or “docs”.
- “trans”, true or false. If documents annotated with subtopics of the current one should be returned or if changes and additions to subtopics of the current one should be returned.
- “lan”, currently “en” or “de” allows to choose the language of topic descriptions.

For example `http://unitracc.de/topic/leak_test?format=xml&docs=20&from=15&trans=false`, returns the documents 15-35 from the documents annotated with topic `leak_test`. The list is returned in a simple, self-explanatory xml format; documents annotated with subtopics of `leak_test` are not returned. Documents are sorted by the time they were added to the index, newest come first.

The URL can be extended with additional topicID: `http://unitracc.de/topic/leak_test/image/civil_engineering?docs=20&trans="true"` returns the last 20 documents that were annotated with the topics `leak_test`, `image` and `civil engineering` (or subtopics of these).

Another entry point are URLs of the form

`http://unitracc.de/document/url=[DOCURL]` where `DOCURL` is a URL known to the system. Calling such an URL returns the information stored about the document at that location, i.e. topics, the time it was indexed and the content of the document stored in the text index. Depending on the form parameter this is returned as `xml` or `html`.

The third entry point at `/search` offers the usual text search interface. Search terms entered into this interface are first analyzed for references to topics, i.e. for each topic a number of lexical references exist (name and synonyms) and an occurrence of one of these in the search string counts as reference to this topic. The index is then searched for documents that contain the search terms and/or are annotated with the topics (or their sub-/supertopics) referenced in the search string. A document that is annotated with the exact topic referenced in the search string is ranked higher than a document annotated with a sub-/supertopic. A document whose content contains a search term is ranked lower than a document that is annotated with a referenced topic.

4.2 Posting Information

We will allow sites that annotate their content with the unitracc topics to register with our site and we will crawl these sites periodically. Metadata about information on these sites can be included by using the above described relTag microformat. So including `Leak Test ` in such a site annotates this page with the topic `leak test`.

Two possibilities exist to alert the metadata repository to the presence of new content: a HTTP POST to the base uri `http://unitracc.de/topic/` with a parameter “document” that specifies the url of the changed document. The document at the supplied URL is then fetched, searched for any topics specified in the relTag microformat and added to the index. To prevent SPAM the HTTP user agent of a post request must be a key that is obtained by registering with unitracc².

The second possibility is to make such a POST request on the URL of a topic; this stores the information that a particular document has the topic on which the POST request was called. So making a POST request on URL `http://unitracc.de/topic/leak_test` with the parameter `document="www.leaktestingspec.com"` adds the information that `www.leaktestingspec.com` is about leak testing, even if the page has no annotations on it. If the registry hasn't seen the url before, the document at this url is also fetched and the content is added to the full text index.

² To further simplify things and allow for easy testing out of every browser, we allow to simulate such post requests by using GET with parameters `action="post"` and `user_agent="..."`

4.3 Changing the topic hierarchy

The information about topics (like name, description, supertopics) can be changed by making a HTTP PUT request on the correct URL and sending the xml document describing the new state of the topic as parameter “topic”. A complete interaction could look like this: a GET request on `/topic/leak_test?format=”xml”` returns the xml description of the topic. The client downloads the xml file, changes it and then calls HTTP PUT on `/topic/leak_test` with the changed xml file as parameter. If the URL the PUT request is called upon did not yet correspond to a topic, a new topic is created. A HTTP DELETE request can be used to delete topics. Obviously these functions are not available without using the correct key as HTTP user agent and we do not see this functionality as part of the functions offered to the community but rather as a way to separate the actual registry from a web based taxonomy editor.

4.4 Changing document data

PUT and DELETE requests can also be made on URLs corresponding to documents. DELETE requests allow to remove documents from the index. A PUT request with the xml document containing the topics and the content of the document allows to fine-tune what gets into the index and to include things that would not be accessible to a crawler (for example content for which some kind of authentication is needed).

4.5 Implementation

The untracc metadata repository is implemented as Servlets using Java 5 and Apache Tomcat 5.5.9³. The HTMLParser⁴ package is used to analyze retrieved documents. The actual index of the documents and their metadata is stored using the open source software Lucene⁵.

Lucene is a text search engine that works on “Documents” containing a number of “Fields”, each “Field” containing string values. Queries work by comparing query terms to the values of fields (contained in, similar to ...), different parts of the query can get different weights depending on how

much this part should contribute to the overall ranking. Parts of the query can be designated as mandatory. The Lucene document representing a untracc metadata registry document and its metadata contain the following fields

- `p_title` and `u_title`: Parts of the document marked as headline, HTML title or set as bold text. They are stored in a stemmed (`p_title`) and a not stemmed form. These fields are used to give an extra boost to documents that highlight one of the search terms.
- `p_content`, `u_content`: The content of the document, stemmed and not stemmed.
- `Topic`: the IDs of the topics this document is annotated with.
- `URL`
- `Date`

If a query asks not only for concrete topics, but also for sub-/supertopics the query is expanded using an in-memory model of the topics and their relations.

5. Limitations and future work

One important limitation is that the metadata registry described here lacks build in support for the hierarchical structure of the “script” infobits – for example it would be desirable that a movie that is used in a chapter annotated with some topic can be found through this topic even if the movie itself is not annotated. However, adding such support while still keeping the entire index in Lucene proves to be complicated and for the time being we have decided to leave any support for the hierarchical infobit structure outside the scope of this system.

Another limitation and area for future work is the support for changes to the topics and their relations. In the current implementation the deletion of a topic has no effect on documents annotated with this topic. In the end this effectively removes annotations from all document, making them harder to find. For the future we want to add support for more sophisticated changes to the topics, for example allowing the merging of two topics.

6. Related Work

That Semantic Web ideas offer the possibility to build more distributed learning systems while still retaining powerful metadata has been observed for a while now[17]. Some work has been done to bring the established LOM standard into the Semantic Web by

³ <http://jakarta.apache.org/tomcat/>

⁴ <http://htmlparser.sourceforge.net/>

⁵ <http://lucene.apache.org/>

creating a binding between RDF and LOM [18], but this format would be too big and complicated for the more casual contributors we are targeting.

The OAI metadata harvesting protocol[19] is a very simple approach to sharing metadata about e-learning materials. It is similar to our approach in its focus on simplicity and the use of simple xml files that are accessed via plain HTTP. Its scope however is much smaller than that of the work described here – the metadata harvesting protocol only defines how metadata can be packaged, not the kind of metadata, metadata registry nor ways to change the metadata.

The edutella system[20] is to date probably the most famous application of Semantic Web technologies to an e-learning scenario. It defines a Peer-to-Peer network in which peers exchange data and queries in RDF. Most ideas from edutella, however, are not applicable to our scenario because we could not expect our contributors to permanently run a Java based edutella peer.

7. References

[1] IEEE Learning Technology Standardization Committee, Draft Standard for Learning Object Metadata, 18 April 2001.

[2] IMS (Instructional Management Systems) Project from Educause. <http://www.imsproject.org/>

[3] ADL Sharable Courseware Object Reference Model, SCORM <http://www.adlnet.org/>

[4] R. Stein, "Berichte: Leitungsbau und- instandhaltung: Internet-Fachportal UNITRACC" in *Bautechnik*, Volume 81, Issue 3, pages 216-217

[5] R. Stein, "KSIUnderground – Kommunale Serviceplattform für Abwasser-Infrastrukturen", *Proceeding of the 9th international symposium on information and communication technologies in urban and spatial planning and impacts of ICT on physical space* (CORP2004)

[6] T. Berners-Lee, J. Hendler, O. Lassila: "The Semantic Web", *Scientific American* 284(5): 34-43. 2001

[7] W3C: Semantic Web Vision, <http://www.w3.org/2001/sw/>

[8] Syndic8 <http://www.syndic8.com/stats.php>

[9] V. Zacharias, M. Sibley: "Semantic Announcement Sharing", *Proceedings of the FGWM 2004*

[10] hCalendar, <http://microformats.org/wiki/hcalendar>, 2005

[11] XFN "XHTML friends network", <http://www.gmpg.org/xfn/>, 2005

[12] relLicense: <http://microformats.org/wiki/rellicense>, 2005

[13] T. Celik: "relTag, Draft Specification" : <http://microformats.org/wiki/rehtag>, 2005

[14] B. Gibson: "Imitation is the surest sign of success" http://thecommunityengine.com/home/archives/2005/07/imitation_is_th.html, 2005

[15] Roy Fielding: "Architectural Styles and the Design of Network based Software Architectures", Dissertation available at http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm, 2000

[16] E. Vlist: "XML Europe 2004: Refactoring XML", <http://www.xml.com/pub/a/2004/05/05/xml.eu.html>, 2004

[17] L. Stojanovic, S. Staab, R. Studer: "eLearning based on the Semantic Web", WebNet2001, World Conference on the WWW and Internet, October 23-27, 2001, Orlando, Florida, USA

[18] M. Nilsson, M. Palmer, J. Brase: "The LOM RDF binding – principles and implementation" *Proceedings of the 3rd annual ARIADNE conference*, Katholieke Universiteit Leuven, Belgium, 2003

[19] S. Warner: "Exposing and harvesting metadata using the OAI metadata harvesting protocol: A tutorial" *High Energy Physics Webzine*, issue 4, June 2001, <http://libraray.cern.ch/HELPW/4/papers/3/>, 2001

[20] W. Nejdil, B. Wolf, C. Qu, S. Decker, M. Sintek, A. Naeve, M. Nilsson, M. Palmer, and Tore Risch. Edutella: A P2P networking infrastructure based on RDF. <http://edutella.jxta.org/reports/edutella-whitepaper.pdf>, November 2001.

8. Acknowledgement

This work was supported in part by the German Federal Ministry of Education and Research under the ksi_underground project.