Task Specific Semantic Views: Extracting and Integrating Contextual Metadata from the Web

Stefania Ghita, Nicola Henze, Wolfgang Nejdl

L3S Research Center / University of Hanover
Deutscher Pavillon, Expo Plaza 1
30539 Hanover, Germany
{ghita,henze,nejdl}@l3s.de

Abstract. Tasks and working scenarios on the desktop involve specific context information which is useful for finding relevant documents related to that context. Automating the process of retrieving and generating this context information is important to avoid time-consuming manual annotation not feasible in everyday work. This paper focuses on automatically extracting and integrating contextual information from web pages used in such working scenarios. The key observation is that in such scenarios we often use a set of web sites to get relevant information, implicitly syndicating their data into a coherent scenario specific information space. We show how these data can be extracted automatically from the web pages stored in local browser caches, based on appropriate query wrappers over these pages. These data are then combined into a task specific semantic view, building upon schema integration rules based on a global as view approach and view materialization, and transformed into RDF metadata for enhancing contextualized search on the desktop. We describe both the conceptual framework as well as our current prototype and conclude with a discussion of further research issues.

1 Introduction

People structure their (work) lifes according to their main activities and, emerging from these daily activities, browsing history is an important mirror of their information seeking behavior. Typically, when people search for information on the web, they do not rely on only one source of information, but many. For example, if a user searches for a publication and its relevant context, he does not only search on one web site, but will combine the information from several sites into a coherent whole. For example, on CiteSeer he will look for the papers that are cited by a specific paper as well as the ones citing it, and on DBLP he will look for the conference that the paper was published at and search for more papers in the same track. So in general, what people try to do is to collect useful information from many sites and manually syndicate this information on their desktop, hoping to have a better view over all relevant information available for specific tasks.

This information is very useful in the desktop search context, as we have discussed in [6], and is available in browser caches. However, from these pages stored as HTML documents, it is very difficult to extract the relevant information automatically. What

we really need is to have this information represented in a structured form, automatically transformed into the relevant task specific RDF context metadata, whose structure is specified using task specific ontologies. The main contribution of this paper is to show how this can be done, based on automatic extraction of relevant context information from web sites and automatic syndication of this context information into context metadata specified by task specific ontologies representing a global view over available context information. We will present an integrated system for context extraction and syndication based on the user's browsing behavior, that is able to gather information from web sites visited by the user during his activities. The system not only stores web pages in their original form, but extracts and syndicates all relevant information, thus reconstructing the relevant information space underlying these pages.

In the next section we present two motivating scenarios and discuss how we extract and syndicate information in such task specific scenarios. We will discuss in more detail in Section 3 the schemas describing web data sources and task specific ontologies, as well as the transformation steps required to extract relevant information and to syndicate it into (materialized) task specific global views. These transformation steps are further detailed in 4, where we describe how we extract web information using the Lixto toolkit and how we materialize our task specific views using mapping rules written in the TRIPLE language. We also describe how these transformation steps are automatically triggered in our Beagle^{++} desktop search infrastructure. We conclude with a discussion of related and future work.

2 Motivating Scenarios

Let us start with two working scenarios where we want to retrieve already viewed information, the first one suitable for research activities, the second suitable for movie fans.

2.1 Research Scenario

Alice is a researcher that has as her main interests peer-to-peer networks and RDF technology. Some time ago she searched for some papers about this subject on the web. These actions have been memorized and will influence her personal profile.

In order to find the necessary information, Alice uses two main information sources: CiteSeer, *citeseer.ist.psu.edu*, for the citing and cited papers, and DBLP, *http://www.informatik.uni-trier.de/~ley/db/*, for the conference information (tracks, editions). Alice discovered a paper about Edutella, **"EDUTELLA: A P2P Networking Infrastructure Based on RDF"** on the CiteSeer web site, as well as some other papers that cited this one, including ones that were written by the same author, Wolfgang Nejdl. As Alice was interested in the conference that this paper was presented at, she followed the link towards the DBLP web site. On the corresponding WWW 2002 conference page on DBLP, she looked at another paper in the same track (Query Language for Semantic Web) as the Edutella paper, *"RQL: a declarative query language for RDF"*. Another paper published by the same author was available from the WWW 2003 conference, ***"Super-Peer-Based Routing and Clustering Strategies for RDF-Based Peer-to-Peer Networks"***.

The system has stored all this browsing behavior and represented it as RDF metadata, both from the CiteSeer web page and the DBLP one. The data harvested this way contains all data from the appropriate web pages dedicated to a certain paper or conference. This merged data are very useful, as it provides all relevant information about each resource, as seen in Figure 1.

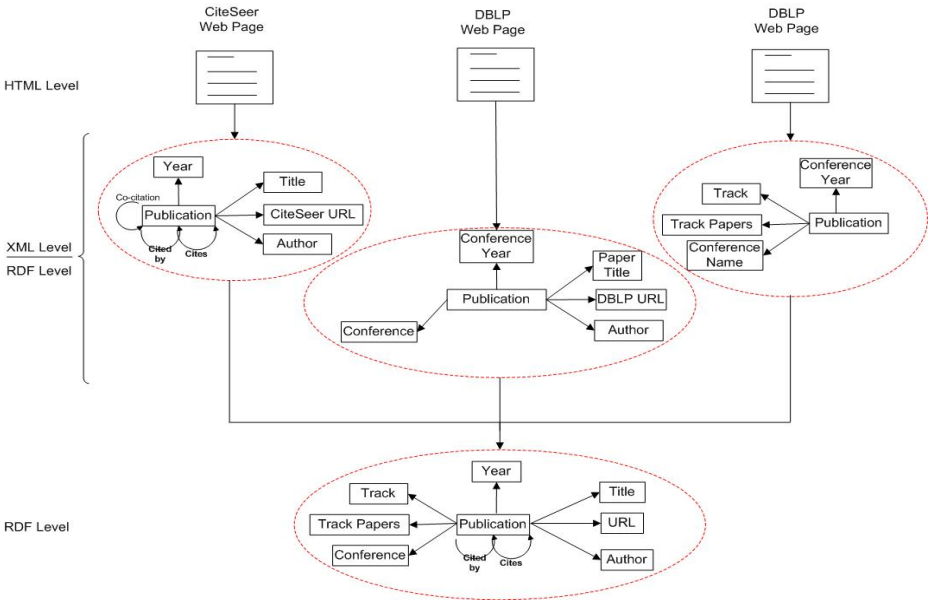

Fig. 1. Syndicated Data View for Research Scenario

When Alice searches again for Edutella resources, she does not only find the paper she retrieved, but additionally get the stored context of this paper. In particular, this includes some publications that cited this paper, and especially the ones with the same author: ***"Super-Peer-Based Routing Strategies for RDF-Based Peer-to-Peer Networks"***, ***"Super-Peer-Based Routing and Clustering Strategies for RDF-Based Peer-to-Peer Networks"***, ***"Role Oriented Models for Hypermedia Construction"***. The system is also able to make the connections between different resources based on the research scenario context metadata. The system knows that Alice viewed the paper both on CiteSeer as well as on the DBLP site, and displays additional data that was extracted from that web site, including the conference that the paper has been presented at, some other papers that were presented in the same track, and other editions of the same conference (WWW).

2.2 Movies Scenario

In our second scenario we investigate how the context is useful for movie fans. Let us take a look at Bob who is a big fan of Harrison Ford and likes science-fiction movies. He

is usually browsing the IMDB (*www.imdb.com*) web site for information and his personal history includes that that he has viewed Harrison Ford's page so many times. For purchasing products from the internet, Bob uses Amazon (*www.amazon.com*), which also has links to the IMDB site.

Some time ago, Bob ordered the "**Star Wars**" CD. Now he wants to buy the DVD, too, and he will be provided with the relevant context. When searching for this DVD, the system realizes that Bob has in the past browsed information about this movie and displays the joined information from the two intensely visited web sites, Amazon and IMDB. For example, the director and the actors for this movie will be displayed, together with other movies that these actors participated. Since Bob is a big fan of Harrison Ford, whenever he views the page on Amazon about the "Star Wars" movie, he will automatically see information retrieved from the web pages of the movies that Harrison Ford participated, as viewed on IMDB. Among the displayed metadata, Bob can also see the rating from IMDB (7.7/10) or the price that he paid for the CD at the time he had bought it. He will also receive some additional movie recommendations from IMDB ("Shaft") and from Amazon ("Psycho").

Besides these suggestions, Bob will also receive additional valuable information, i.e. comments from users of the IMDB web site. Not any recommendations, though, since the system will be able to choose among the multitude of comments about this movie the one made by his favorite commenter, "*Grann-Bach*". This comment is also very highly rated by the users of the IMDB system (22/30), even though on the IMDB page some other user's comment would appear. This will be done because Bob always reads his comments and this fact is memorized in his personal history. So whenever Bob wants to buy something, additional information about the product will be made available to him, based on information extracted from his past activities (see Figure 2).

3 Relevant Data and Transformation Steps

As seen in Figure 2, we can partition the process behind our motivating scenarios into distinct steps, further detailed in the rest of this paper. The first step takes care of the extraction of information from various web sites, each web site having a specific schema for their content, as discussed in Section 3.1. The data retrieved in an XML format will contain the relevant web page information for each context and will be transformed into RDF data using XSLT (Section 3.2). After this transformation, the data are syndicated and materialized (based on appropriate mapping rules) into task specific semantic views, and can be used to answer queries based over these views.

3.1 Schemas for Web Page Content

Data driven HTML pages contain two kinds of information: about the structure of the page (repetition of items) - data items are listed as rows of a table, or are structured in distinct sections - and about what is presented within this structure - the actual information. The first type (partially) reflects the structure of the database that were used to generate the web page. All pages generated from databases and a lot of other ones repeat the same structural items so that we can recognize different information items

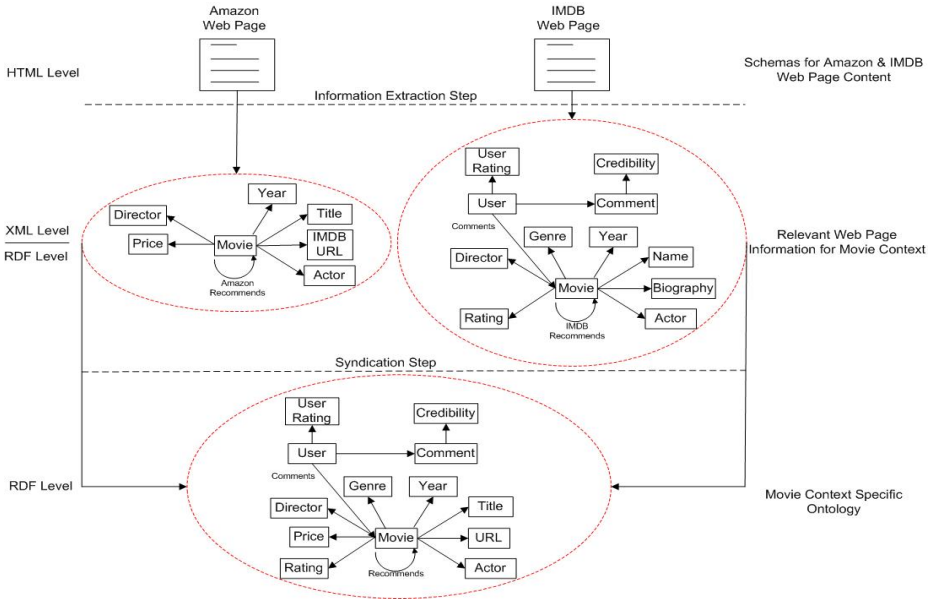

Fig. 2. Syndicated Data View for Movies Scenario

rather easily. For example, a Hotel Information Server has web pages structured based on the databases entries: the description of an individual Hotel with details like general information, pictures, contact address, etc. In the case of our research scenario, the information about scientific publications is often presented in the same style (see Figure 3), and includes title of the paper, year of publication, authors, and cited papers.

As a first step, we need local schemas for the web pages interesting for a specific scenario. Appropriate collections of web pages share some structure for presenting the content, e.g. all pages from CiteSeer that present information about a publication belong to a class, together with the web pages that are associated to this page by dedicated links (in Figure 3 these are the links under the rubric "cited by" plus the according web pages). In a (manual) preparation step, we analyze each of these collections for expressive metadata, and design a small, local ontology which describes the objects of discourse of each of these collections. We can then harvest information about entities and their different attributes. For example, for papers we have publications and conferences and the attributes for these entities. Similarly, the IMDB and Amazon web sites reflect the underlying database entries: movies, actors, directors. Such a local schema for the CiteSeer source is depicted in Figure 4.

In all these cases, the information on the web page is semi-structured, and this will allow us to construct machine-readable metadata from these web pages in a semi-automated manner (see Section 4.1). This is done based on the reconstructed schema and an appropriate query on the HTML page which extracts information according to that schema.

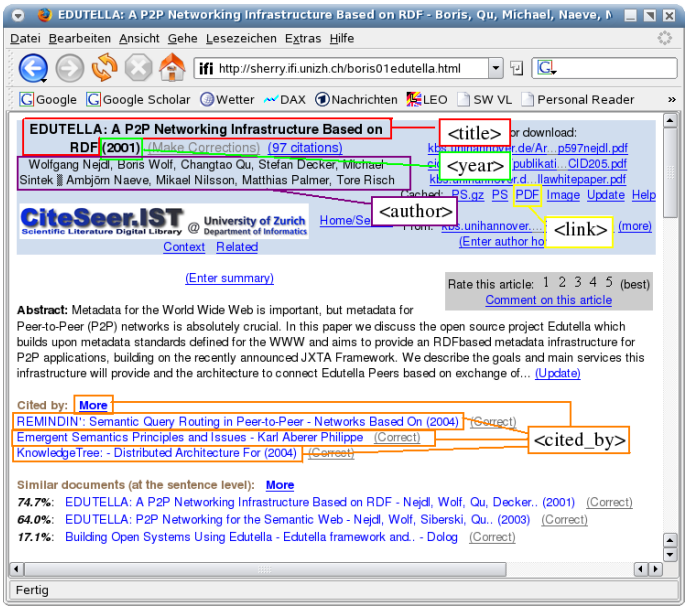

Fig. 3. CiteSeer Web Page

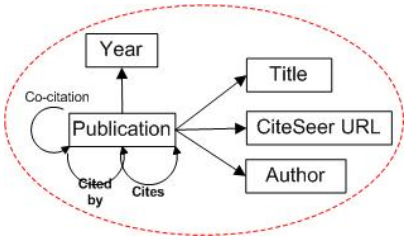

Fig. 4. CiteSeer Local Schema

3.2 Task Specific Semantic Views

Depending on the tasks and context the user is working in, his context includes all relevant information from the local schemas which we described in the last section. This context can be represented by a task specific semantic view integrating the relevant data from the local sources we discussed in the previous section. This semantic view specifies all metadata needed of this context, and is described using ontologies. Let us take a look on two ontologies appropriate for our two example scenarios.

Research Ontology. Figure 5 depicts an overview image of the ontology that defines appropriate context metadata for the research scenario. The ”Publication” class represents a specific type of file, with additional information associated to it. The most important attributes are “Author”, “Title”, as well as relationships regarding citing and

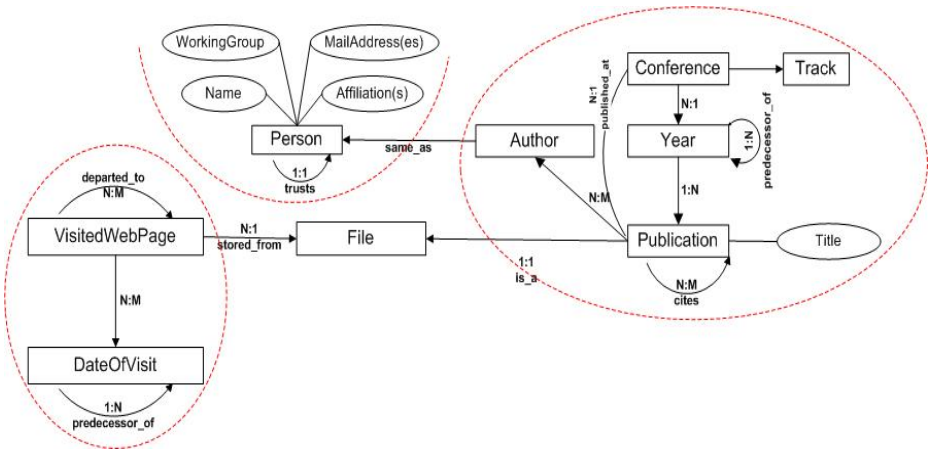

Fig. 5. Research Ontology

cited papers. These attributes and relations can easily be retrieved from a CiteSeer web page, "Conference" and "Year" harvesting is done more reliable from the DBLP site. As each publication is stored as a file, it is also connected to the file context, and thus to the file specific information like path, number of accesses, etc. Additionally, it is possibly connected to visited web pages / URLs a publication has been downloaded from. Authors are persons, which are modeled based on the FOAF ontology, as members of interest groups (foaf:Group).

If we take a look at Figure 1 we see that some attributes are omitted from the more web page-specific ontologies, such as "Co-citations" retrieved from the CiteSeer web page, and that some attributes have different names, even though they represent the same information, such as "Title" and "Paper Title". So this ontology represents a specific view on the local data sources, appropriate for the task context.

Movies Ontology. The central part of the movies ontology is the "Movie", as seen in Figure 6. It is associated to the participating actors, the director, the year it was released and its genre (comedy, action, thriller, science fiction, etc.) The IMDB site also provides a "Rating" computed with the help of the ranks provided by different registered users. An interesting feature of this database is the fact that the system also makes recommendations for other movies. Users comment on movies, these comments have a credibility value based on their usefulness to other users. As in the research scenario, this ontology again represents a syndicated view of the Amazon and the IMDB ontologies, as seen in Figure 2.

3.3 Transforming Web Page Content Information into Task Specific Metadata

Combining data from different sources and providing the user with a unified view of these data is known as the "data integration" problem [19, 16, 4]. The set of sources (in our context the set of visited web pages) contain the relevant data, while a global schema

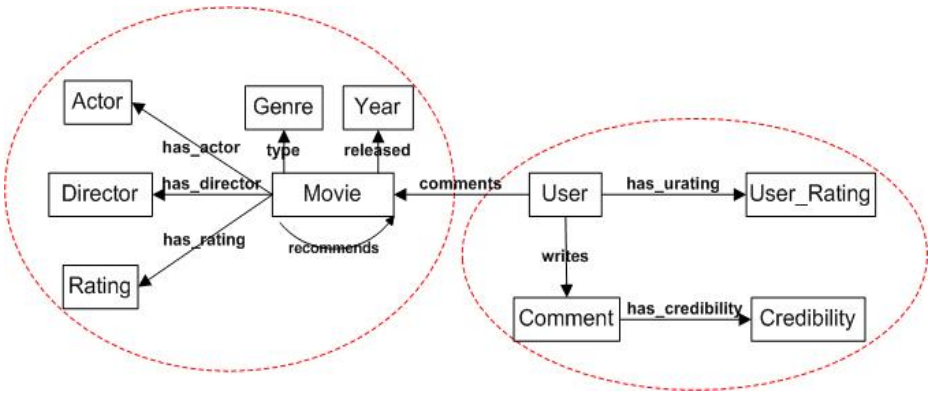

Fig. 6. Movies Ontology

(in our context the task specific ontologies) provide a unified view of the underlying sources. For modeling the relation between the sources and the global schema we will use the global-as-view approach [8, 10], which describes the mappings between local sources and the global schema as a set of assertions

$$g \rightsquigarrow q_S$$

where g represents an element of the global schema and q_S a query over the sources. Such a mapping explicitly specifies how to query the local sources for each element contained in a query over the global schema, or alternatively, how to materialize the global schema based on the instances from the local data sources.

Using these mappings, we can materialize instances of the task specific ontologies when the user browses new web pages, or reformulate queries over the task specific ontologies during search time. Obviously, the second alternative is not really useful in our context as users have come to expect nearly instantaneous access to search results from web search engines.

As we assume exact views and do not allow integrity constraints in the global schema, our data integration algorithm can exploit the "single database property" [4] which means that all instances of relations of the global schema can be computed by the corresponding views over the sources, using the mapping rules as transformation rules. So view materialization is the more feasible approach for our application, and we just have to remember how the derived global schema instances depend on the source data in order to recompute the views when source data change. This is similar to view materialization in data warehouses [7].

The global database is thus constructed by merging the important information from the relevant sources of information, i.e. the information extracted from the web pages browsed are merged into the global database containing our activity driven metadata as specified by the task specific ontologies. We can easily see that the data is not only a projection or a subset of the data provided by one site, but another representation of the information. When we map from the global to the local level, we can have differ-

ent transformations from the different local schemas. We represent these mappings as discussed in [15]:

$$Year_{global}(Paper, Year) \rightarrow Conference_{DBLP}(Paper, Conference), \\ Conference\_Year_{DBLP}(Conference, Year).$$

$$Title_{global}(Movie, Name) \rightarrow Title_{Amazon}(Movie, Name).$$
$$Title_{global}(Movie, Name) \rightarrow Name_{IMDB}(Movie, Name).$$

4 Metadata Extraction and Transformation

Now that we know how our global ontologies look like and how the data extracted from the web pages is structured, we have to describe how exactly we transform data from local data sources into RDF instances corresponding to the global ontologies. We need the following two steps to go from web pages to contextual metadata:

- *extract* task related metadata from distributed, inhomogeneous sources into local schemas (Section 4.1)
- *transform* this gathered metadata into one, common context schema (Section 4.2)

Even when web pages change, their structure tends to stay the same so any wrappers or transformation rules remain valid. Metadata are then extracted automatically when the web page is visited again. Of course, initial effort has to be invested for completely new wrappers or ontologies.

4.1 Extraction of Web Information Using Lixto

As described in Section 3.1, we are interested in the structured information containted in web pages. We use the Lixto Toolkit [12] for handling this extraction part. The Visual Wrapper from Lixto [2] provides a methodology and tool for the visual and interactive generation of query wrappers - programs, that automatically extract data from semi-structured data sources like web pages and transform them into XML. Lixto wrappers contain queries in the Elog query language, which is based on monadic Datalog plus extensions for regular expressions. The extractor, using as input an HTML document and a previously constructed program, generates as its output a pattern instance base, a data structure which encodes the extracted instances as hierarchically ordered trees and strings. For our publication scenario, an excerpt of the XML extraction of the example page depicted in Figure 3 looks as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<document>
    <Publication>
        <Title>Super-Peer-Based Routing and Clustering Strategies for
                RDF-Based Peer-to-Peer Networks</Title>
        <Author>Wolfgang Nejdl</Author>
        <Author>Martin Wolpers</Author>
        <Author>Wolf Siberski</Author>
        <Author>Christoph Schmitz</Author>
        <Author>Mario Schlosser</Author>
        <Author>Ingo Brunkhorst</Author>
```

```
        <Author>Alexander Loeser</Author>
        <Year>2003</Year>
        <Link>http://citeseer.ist.psu.edu/..../nejdl03superpeerbased.pdf</Link>
        <Citations>
            <CitationTitle>Chord: A Scalable Peer-to-Peer Lookup Service for
                           Internet Applications</CitationTitle>
            <CitationTitle>A Scalable Content-Addressable Network</CitationTitle>
            <CitationTitle>Mediators in the Architecture of Future
                           Information Systems</CitationTitle>
            <CitationTitle>The TSIMMIS Project: Integration of Heterogeneous
                           Information Sources</CitationTitle>
            <CitationTitle>A Measurement Study of Peer-to-Peer File Sharing
                           Systems</CitationTitle>
            ........
        </Citations>
        <CitedBy>
            <CitedByTitle>Self-Organization of a Small World by Topic</CitedByTitle>
            <CitedByTitle>Semantic Query Routing and Processing in P2P Database
                          Systems: The ICS-FORTH SQPeer Middleware</CitedByTitle>
            <CitedByTitle>Top-k Query Evaluation for Schema-Based Peer-to-Peer
                           Networks</CitedByTitle>
            <CitedByTitle>Super-Peer-Based Routing Strategies for RDF-Based
                          Peer-to-Peer Networks</CitedByTitle>
            .........
        </CitedBy>
    </Publication>
</document>
```

For this XML output, a small part of the Lixto extractor, which harvests data about the title of a paper, is shown in the following snippet:

```
<StringSourceDef description="" maxInstances="-1" name="Title"
                 parentName="TitleLine">
    <ExtractionRules>
        <StringExtractionRule description="" parent="TitleLine">
            <Head description="">
                <I>
                    <Var name="0"/>
                </I>
                <O>
                    <Var name="1"/>
                </O>
            </Head>
            <AtomChain>
                <SubText description="">
                    <I>
                        <Var name="0"/>
                    </I>
                    <STD>
                        <SimpleSTD>
                            <RE pattern="(([A-Z]|[A-Z]+|[A-Z]\w+).*(\s|\-)([A-Z]\w+|[A-Z]+|[a-z]+))"/>
                        </SimpleSTD>
                    </STD>
                    <O>
                        <Var name="1"/>
                    </O>
                </SubText>
            </AtomChain>
        </StringExtractionRule>
    </ExtractionRules>
</StringSourceDef>
```

In a second step, the extracted XML data are then transformed to RDF using an XSLT script, resulting in the following format:

```
<rdf:Description rdf:about="http://citeseer.ist.psu.edu/569523.html">
    <dc:publisher>University of Hannover</dc:publisher>
    <dc:title>Super-Peer-Based Routing and Clustering Strategies for RDF-Based
              Peer-to-Peer Networks</dc:title>
    <dc:creator>
        <rdf:Seq>
              <rdf:li rdf:resource="#Wolfgang Nejdl"/>
              <rdf:li rdf:resource="#Martin Wolpers"/>
              <rdf:li rdf:resource="#Wolf Siberski"/>
              <rdf:li rdf:resource="#Christoph Schmitz"/>
              <rdf:li rdf:resource="#Mario Schlosser"/>
              <rdf:li rdf:resource="#Ingo Brunkhorst"/>
              <rdf:li rdf:resource="#Alexander Loeser"/>
        </rdf:Seq>
    </dc:creator>
    <dc:date>2003</dc:date>
    <dc:identifier>http://citeseer.ist.psu.edu/..../nejdl03superpeerbased.pdf
                  </dc:identifier>
    <citeseer:cites>Chord: A Scalable Peer-to-Peer Lookup Service for
                    Internet Applications</citeseer:cites>
    <citeseer:cites>A Scalable Content-Addressable Network
                    </citeseer:cites>
    <citeseer:cites>Mediators in the Architecture of Future Information
                    Systems</citeseer:cites>
    <citeseer:cites>The TSIMMIS Project: Integration of Heterogeneous
                    Information Sources</citeseer:cites>
    <citeseer:cites>A Measurement Study of Peer-to-Peer File Sharing
                    Systems</citeseer:cites>
    .........
    <citeseer:cited_by>Self-Organization of a Small World by Topic
                       </citeseer:cited_by>
    <citeseer:cited_by>Semantic Query Routing and Processing in P2P Database
                       Systems:The ICS-FORTH SQPeer Middleware</citeseer:cited_by>
    <citeseer:cited_by>Top-k Query Evaluation for Schema-Based Peer-to-Peer
                       Networks</citeseer:cited_by>
    <citeseer:cited_by>Super-Peer-Based Routing Strategies for RDF-Based
                       Peer-to-Peer Networks</citeseer:cited_by>
    ..........
</rdf:Description>
```

4.2 Transformation into Task Specific Semantic Views Based on the Mapping Rules

After the extraction and transformation of data according to our local schemas, we then have to transform these data into the global schema, which gives us a unified view on *all* the local sources that we can query for each scenario. The ontologies described in Section 3.2 provide these task specific semantic views, specifying the final format for the contextual metadata for each scenario.

This transformation step is facilitated by the mapping rules (see Section 3.3) which provide the translation between the local properties and relations identified on the web sites (e.g., CiteSeer local ontology) and the properties and relations that are specified in the syndicated ontology (e.g., research ontology). The mapping rules are necessary for all items we want to keep for the global view. In order to materialize our task specific semantic views, we translated the mapping rules in Section 3.3 into TRIPLE rules [18]. The following examples show how contextual information described in the research ontology and in the movie ontology can be constructed from data extracted from the local sources (DBLP, Amazon and IMDB in this case):

```
FORALL PAPER,YEAR  PAPER[global:has_published->YEAR] <-
    EXISTS CONFERENCE conference(PAPER,CONFERENCE)@DBLP.

FORALL MOVIE,NAME  MOVIE[global:has_title->NAME] <-
    title(MOVIE,NAME)@AMAZON
    OR name(MOVIE,NAME)@IMDB.
```

4.3 Triggering These Transformation Steps

Our current Beagle^{++} prototype [6, 5] is being built on top of the open source Beagle desktop search infrastructure, which we extended with additional modules (metadata generators) handling the creation of contextual information, and a ranking module, which computes the ratings of resources so that search results are shown in the order of their importance. Compared to existing desktop search applications this makes it easier to find relevant resources based on the additional contextual information and to use link-based ranking algorithms like PageRank operating on the context information graph, in addition to traditional TF/IDF measures.

Beagle Event Based Architecture. The main characteristic of our extended desktop search architecture is metadata generation and indexing on-the-fly, triggered by modification events generated upon occurrence of file system changes. Events are generated whenever a new file is copied to hard disk or stored by the web browser, when a file is deleted or modified, when a new email is read, etc, and according to the type of events, we trigger the appropriate annotation steps. Much of this basic notification functionality is provided in Linux by an inotify-enabled Linux kernel, which is used by Beagle.

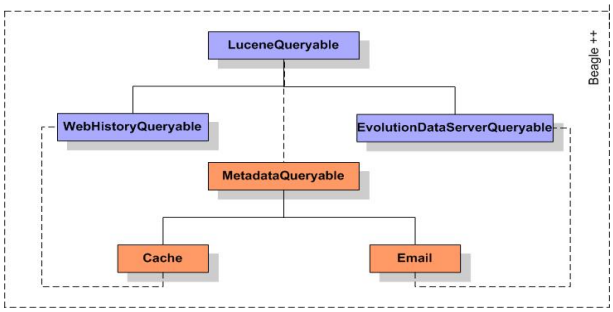

Fig. 7. Beagle Extensions for Metadata Support

Web Cache Metadata Generator. Figure 7 shows how additional metadata generators are integrated into our Beagle^{++} prototype. The queryable responsible for the web cache annotation is WebHistoryQueryable. Each URL typed in the web browser that is not in the cache will be transmitted by Beagle^{++} to the Lixto wrapper that harvests the data according to the appropriate local ontologies. The XML data are then translated

via XSLT into RDF and then are materalized into the global semantic views with the help of the TRIPLE mapping rules, stored into a RDF file, and indexed appropriately. If the URL is in the cache, the relevant metadata will be displayed.

4.4 Metadata Visualization

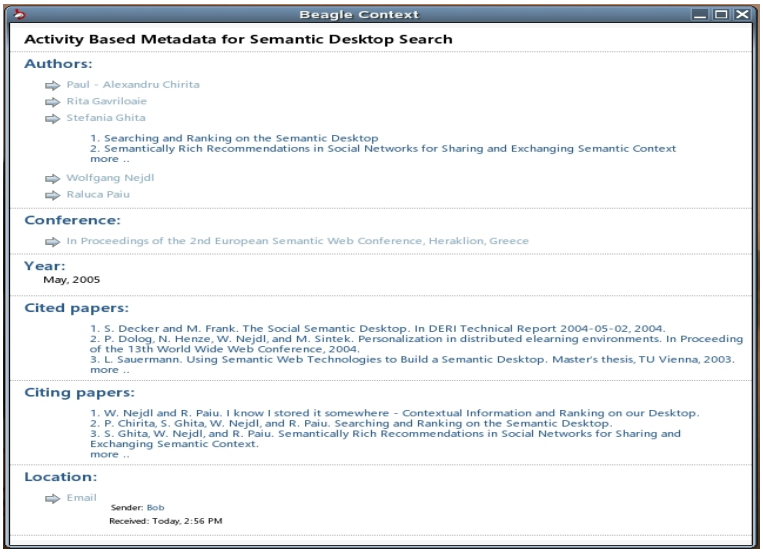

Fig. 8. Beagle^{++} Metadata Window

Let's suppose that Alice searched for the words "**semantic desktop**", and she chose among the results the paper "Activity Based Metadata for Semantic Desktop Search". When visualizing this result, the corresponding metadata can be seen as well. A new window pops up displaying a list of details that correspond to the ontology related to the type of resource. The result is stored on the desktop as a file sent as an attachment by Bob. The metadata window displays the annotations corresponding to publications together with other contextual information associated with it, retrieved from all the other sources (e.g., DBLP, CiteSeer). The publication has 5 authors and for each of the authors we can further display the next level of metadata. For example, in Figure 8, Alice extended author S. Ghita and she can see other publications of this author. Additionally, she can see that the publication was presented at the ESWC conference in 2005, its referenced publications and the ones that cited it. Information related to the provenance of this resource is also shown, the email it was saved from and its sender.

We are currently extending our prototype to be able to display metadata on arbitrary ontologies.

5 Related Work

Our approach integrates ideas from various fields: metadata extraction and syndication in the web as well as recent achievements for the semantic desktop.

In [14], the authors describe an approach for personalized content syndication, featuring a central content syndicator instance which answers user requests. Our approach differs from this approach as we do not focus on content brokerage but on metadata brokerage, and incrementally construct metadata which we use for modeling a user's context and preferences. Therefore, formats for content syndication like RSS [17] are not expressive enough to create the metadata needed. A related approach creating metadata descriptions on behalf of a web extraction process is described in [3]. The author creates RDF descriptions about publication information from dedicated sites in an automated process, as well as new views on the data based on these descriptions and additional background knowledge available for this application.

One of the most interesting semantic search efforts concerning metadata enrichment of results and their visualization is being performed in the TAP project [13]. TAP builds upon the TAPache module, which provides a platform for publishing and consuming data from the Semantic Web. Its knowledge base is updated with the aid of the onTAP system, which includes 207 HTML page templates, being able to read and extract knowledge from 38 different high quality web sites. The key idea in TAP is that for specific searches, a lot of information is available in catalogs and backend databases, but not necessarily on web pages crawled exhaustively by Google. The semantic search results are independent of the results obtained via traditional information retrieval technologies and aim to augment them. In contrast, metadata information in our scenarios reflects contextual and activity-based metadata information available on our desktop.

The difficulty of accessing information on our computers has prompted several first releases of desktop search applications during the last months. The most prominent examples include Google desktop search [11] and the Beagle open source project for Linux [9]. They do not exploit metadata information, but rely on a regular text-based index. Apple Inc. has integrated an advanced desktop search application (named *Spotlight Search* [1]) into their new operating system, Mac OS Tiger. Even though they did add semantics into their tool, only explicit information is used, such as file size, creator, last modification date, or metadata embedded into specific files (images taken with digital cameras for example). While this is indeed an improvement over regular search, it still misses contextual information often resulting or inferable from explicit user actions or additional background knowledge, as discussed in this paper.

6 Conclusions and Future Work

In this paper we discussed how relevant data can be automatically extracted from web sites visited by the user during his work and syndicated into task specific semantic views, which represent contextual information relevant for specific tasks and contexts. This contextual information can be exploited to enhance desktop search beyond full-text indexing, leading to more search results as well as to richer result representation.

We are currently putting all implementation pieces together in our Beagle++ prototype, and will evaluate it in more depth in the application scenarios described in this

paper. Additionally, we intend to investigate in more detail how to incrementally update views whenever the content of revisited web pages has changed, in order to keep our contextual information consistent.

References

1. Apple spotlight search. http://developer.apple.com/macosx/tiger/spotlight.html.
2. Robert Baumgartner, Sergio Flesca, and Georg Gottlob. Declarative information extraction, web crawling, and recursive wrapping with lixto. In *6th International Conference on Logic Programming and Nonmonotonic Reasoning*, Vienna, Austria, 2001.
3. Robert Baumgartner, Nicola Henze, and Marcus Herzog. The Personal Publication Reader: Illustrating Web Data Extraction, Personalization and Reasoning for the Semantic Web. In *ESWC*, Heraklion, Greece, May 29 - June 1 2005.
4. Andrea Cal, Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. On the expressive power of data integration systems. In *21st Int. Conf. on Conceptual Modeling*, 2002.
5. P.-A. Chirita, S. Ghita, W. Nejdl, and R. Paiu. Semantically enhanced searching and ranking on the desktop. In *ISWC, November*, 2005.
6. Paul Alexandru Chirita, Rita Gavriloaie, Stefania Ghita, Wolfgang Nejdl, and Raluca Paiu. Activity based metadata for semantic desktop search. In *Proceedings of 2nd ESWC*, Heraklion, Greece, May 2005.
7. Rada Chirkova, Alon Y. Halevy, and Dan Suciu. A formal perspective on the view selection problem. In *Proceedings of the 27th International Conference VLDB*, pages 59–68, Rome, Italy, September 2001. Morgan Kaufmann.
8. Hector Garcia-Molina, Yannis Papakonstantinou, Dallan Quass, Anand Rajaraman, Yehoshua Sagiv, Jeffrey D. Ullman, Vasilis Vassalos, and Jennifer Widom. The TSIMMIS approach to mediation: Data models and languages. *Journal of Intelligent Information Systems*, 8(2):117–132, 1997.
9. Gnome beagle desktop search. http://www.gnome.org/projects/beagle/.
10. Cheng Hian Goh, Stéphane Bressan, Stuart Madnick, and Michael Siegel. Context interchange: new features and formalisms for the intelligent integration of information. *ACM Transactions on Information Systems*, 17(3):270–270, 1999.
11. Google desktop search application. http://desktop.google.com/.
12. Georg Gottlob, Christoph Koch, Rober Baumgartner, Marcus Herzog, and Sergio Flesca. The Lixto Data Extraction Project — Back and Forth between Theorie and Practice. In *ACM Symposium on Principles of Database Systems (PODS)*, volume 23. ACM, June 2004.
13. R. Guha, Rob McCool, and Eric Miller. Semantic search. In *Proceedings of the 12th International Conference on WWW*, pages 700–709. ACM Press, 2003.
14. W. Kießling, W.-T. Balke, and M. Wagner. Personalized content syndication in a preference world. In *EnCKompass Workshop on E-Content Management*, Eindhoven, The Netherlands, 2001.
15. Maurizio Lenzerini. Data integration: A theoretical perspective. In *PODS*, pages 233–246, 2002.
16. Alon Y. Levy, Alberto O. Mendelzon, Yehoshua Sagiv, and Divesh Srivastava. Answering queries using views. In *Proceedings of the 14th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 95–104, San Jose, Calif., 1995.
17. RDF Site Summary specification. http://web.resource.org/rss/1.0/.
18. Triple, an rdf rule language. http://triple.semanticweb.org/.
19. Jeffrey D. Ullman. Information integration using logical views. *Theoretical Computer Science*, 239(2):189–210, 2000.