Semantically Enhanced Searching and Ranking on the Desktop

Paul - Alexandru Chirita, Stefania Ghita, Wolfgang Nejdl, and Raluca Paiu

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

Abstract. Existing desktop search applications, trying to keep up with the rapidly increasing storage capacities of our hard disks, offer an incomplete solution for the information retrieval. In this paper we describe our desktop search prototype, which enhances conventional full-text search with semantics and ranking modules. In this prototype we extract and store activity-based metadata explicitly as RDF annotations. Our main contributions are represented by the extensions we integrate into the Beagle desktop search infrastructure to exploit this additional contextual information for searching and ranking the resources on the desktop. Contextual information plus ranking brings desktop search much closer to the performance of web search engines. The initially disconnected sets of resources on the desktop are connected by our contextual metadata, and then a PageRank derived algorithm allows us to rank these resources appropriately. Finally, we use a detailed working scenario to discuss the advantages of this approach, as well as the user interfaces of our search prototype.

1 Introduction

The capacity of our hard-disk drives has increased tremendously over the past decade, and so has the number of files we usually store on our computer. It is no wonder that sometimes we cannot find a document any more, even when we know we saved it somewhere. Ironically, in quite a few of these cases nowadays, the document we are looking for can be found faster on the World Wide Web than on our personal computer.

Web search has become more efficient than PC search due to the boom of web search engines and due to powerful ranking algorithms like the PageRank algorithm introduced by Google [10]. The recent arrival of desktop search applications, which index all data on a PC, promises to increase search efficiency on the desktop. However, even with these tools, searching through our (relatively small set of) personal documents is currently inferior to searching the (rather vast set of) documents on the web. This happens because these desktop search applications cannot rely on PageRank-like ranking mechanisms, and they also fall short of utilizing desktop specific characteristics, especially context information.

It is thus obvious that simple indexing of the data on the desktop has to be enhanced by ranking techniques, otherwise the user has no other choice but to look at the entire result sets for her queries – usually a tedious task. The main problem with ranking on the desktop comes from the lack of links between documents, the foundation of current ranking algorithms (in addition to TF/IDF numbers). A semantic desktop offers the missing ingredients: By gathering semantic information from user activities, from the contexts the user works in[1], we build the necessary links between documents.

In this paper we enhance and contextualize desktop search based on both resource specific metadata and semantic metadata collected from different available contexts and activities performed on a personal computer. We first describe the semantics of these different contexts by appropriate ontologies in Section 3.3, and then propose a ranking algorithm for desktop documents in Section 3.4. For this latter aspect, we focus on recent advances of PageRank-based ranking, showing how local (i.e., context-based) and global ranking measures can be integrated in such an environment. We are currently implementing our prototype on top of the open source Beagle project [8], which aims to provide sophisticated desktop search in Linux. Section 4 gives a detailed description of our prototype, and shows how we extended Beagle with additional modules for annotating and ranking resources. In Section 5 we use one typical search scenario in order to illustrate how our extended Beagle search infrastructure can improve the retrieval of search results in terms of number and order of results, using contextual information and ranking based on this information.

2 Previous Work

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 [9] (proprietary, for Windows) and the Beagle open source project for Linux [8]. Yet they include *no* metadata whatsoever in their system, but just a regular text-based index. Nor does their competitor MSN Desktop Search [11]. Finally, Apple Inc. promises to integrate an advanced desktop search application (named *Spotlight Search* [2]) into their upcoming operating system, Mac OS Tiger. Even though they also intend to 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 include many additional characteristics, such as exposure information or whether a flash was used). 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 the next sections.

[1] Studies have shown that people tend to associate things to certain contexts [14], and this information should be utilized during search. So far, however, neither has this information been collected, nor have there been attempts to use it.

Swoogle [5] is a search and retrieval system for finding semantic web documents on the web. The ranking scheme used in Swoogle uses weights for the different types of relations between Semantic Web Documents (SWD) to model their probability to be explored. However, this mainly serves for ranking between ontologies or instances of ontologies. In our approach we have instances of a fixed ontology and the weights for the links model the user's preferences.

Facilitating search for information the user has already seen before is also the main goal of the *Stuff I've Seen (SIS)* system, presented in [6]. Based on the fact that the user has already seen the information, contextual cues such as time, author, thumbnails and previews can be used to search for and present information. [6] mainly focuses on experiments investigating the general usefulness of this approach though, without presenting more technical details.

The importance of semantically capturing users' interests is analyzed in [1]. The purpose of their research is to develop a ranking technique for the large number of possible semantic associations between the entities of interest for a specific query. They define an ontology for describing the user interest and use this information to compute weights for the links among the semantic entities. In our system, the user's interest is a consequence of her activities, this information is encapsulated in the properties of the entities defined, and the weights for the links are manually defined.

An interesting technique for ranking the results for a query on the semantic web takes into consideration the inferencing processes that led to each result [13]. In this approach, the relevance of the returned results for a query is computed based upon the specificity of the relations (links) used when extracting information from the knowledge base. The calculation of the relevance is however a problem-sensitive decision, and therefore task oriented strategies should be developed for this computation.

3 An Architecture for Searching the Semantic Desktop

3.1 Overview

This chapter will present our 3-layer architecture for generating and exploiting the metadata enhancing desktop resources. At the bottom level, we have the physical resources currently available on the PC desktop. Even though they can all eventually be reduced to "files", it is important to differentiate between them based on content and usage context. Thus, we distinguish structured documents, emails, offline web pages, general files[2] and file hierarchies. Furthermore, while all of them do provide a basis for desktop search, they also miss a lot of contextual information, such as the author of an email or the browsing path followed on a specific web site. We generate and store this additional search input using RDF metadata, which is placed on the second conceptual layer of our architecture. Finally, the uppermost layer implements a ranking mechanism over all resources from the previous levels. An importance score is thus computed for

[2] Text files or files whose textual content can be retrieved.

each desktop item, supporting an enhanced ordering of results within desktop search applications. The entire architecture is depicted in Figure 1. In the next subsections we will describe each of its layers following a bottom-up approach.

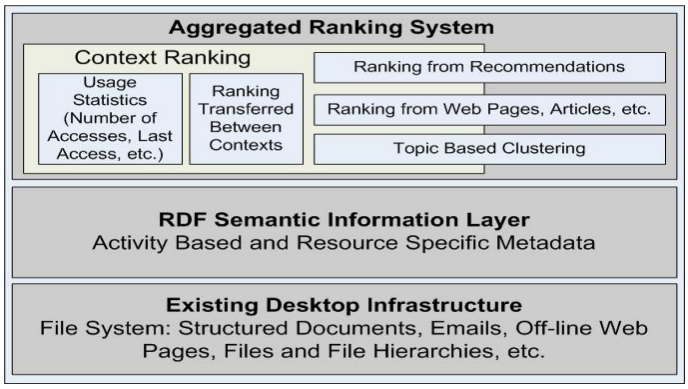

Fig. 1. Desktop Ranking System Architecture

3.2 Current Desktop Infrastructure and its Limitations

Motivation and Overview. Today the number of files on our desktops can easily reach 10,000, 100,000 or more. This large amount of data can no longer be ordered with manual operations such as defining explicit file and directory names. Automatic solutions are needed, preferably taking into account the activity contexts under which each resource was stored/used. In our prototype we focus on three main working contexts of email exchanges, file procedures (i.e., create, modify, store, etc.), and web surfing. Furthermore, we investigate an additional extended context related to research and scientific publications. In the following paragraphs, we will discuss how the resources associated to these contexts are currently encountered, and which (valuable) information is lost during their utilization. Subsequent sections will then present solutions to represent this information and exploit it for desktop search applications.

Email Context. One of the most flourishing communication mediums is certainly email communication. For example, international scientific collaboration has become almost unthinkable without electronic mail: Outcomes of brainstorming sessions, intermediate versions of reports, or published articles represent just a few of the items exchanged within this environment. Similarly, Internet purchasing or reservations are usually confirmed via email. However, if we consider the continuous increase of email exchanges, other enhanced solutions will be necessary to sort our correspondence. More, when storing emails, a lot of contextual information is lost. Most significant here is the semantic connection between the attachments of an email, its sender and subject information, as well

as the valuable comments inside its body. We propose to explicitly represent this information as RDF metadata, to enable both enhanced search capabilities for our inbox, as well as the exploitation of the semantic link between desktop files (e.g., PDF articles stored from attachments), the person that sent them to us via email or the comments she added in the email body.

Files and File Hierarchy Context and Web Cache Context. Due to space limitations, we refer the reader to [4], where we proposed several solutions to enrich the information associated to file and directory names, as well as to the previously visited resources on the Web.

Working with Scientific Publications. Research activities represent one of the occupations where the need for contextual metadata is very high. The most illustrative example is the publication itself: Where did this file come from? Did we download it from CiteSeer or did somebody send it to us by email? Which other papers did we download or discussed via email at that time, and how good are they (based on a ranking measure or on their number of citations)? We might remember the general topic of a paper and the person with whom we discussed about it, but not its title. These questions arise rather often in a research environment and have to be answered by an appropriate search infrastructure.

3.3 RDF Semantic Information Layer

Motivation and Overview. People organize their lives according to preferences often based on their activities. Consequently, desktop resources are also organized according to performed activities and personal profiles. Since, as described above, most the information related to these activities is lost on our current desktops, the goal of the RDF semantic information layer is to record and represent this data and to store it in RDF annotations associated to each resource. Figure 2 depicts an overview image of the ontology that defines appropriate annotation metadata for the context we are focusing on in this paper. The following paragraphs will describe these metadata in more detail.

Email Metadata. Basic properties for the email context are referring to the date when an email was sent or accessed, as well as its subject and body text. The status of an email can be described as seen/unseen or read/unread. A property "reply to" represents email thread information, the "has attachment" property describes a 1:n relation between an email and its attachments. The "sender" property gives information about the email sender, which can be associated to a social networking trust scheme, thus providing valuable input for assessing the quality of the email according to the reputation of its sender.

File and Web Cache Specific Metadata. For these, we again refer the reader to our previous work [4] describing the ontologies associated to these two activity contexts. An overview can be found in the lower half of Figure 2.

Scientific Publications Metadata. The Publication class represents a specific type of file, with additional information associated to it. The most common fields are "author", "conference", "year", and "title", which comprise the regular information describing a scientific article. Additionally, we store the paper's CiteSeer ID (if any). The publication context is connected to the email context,

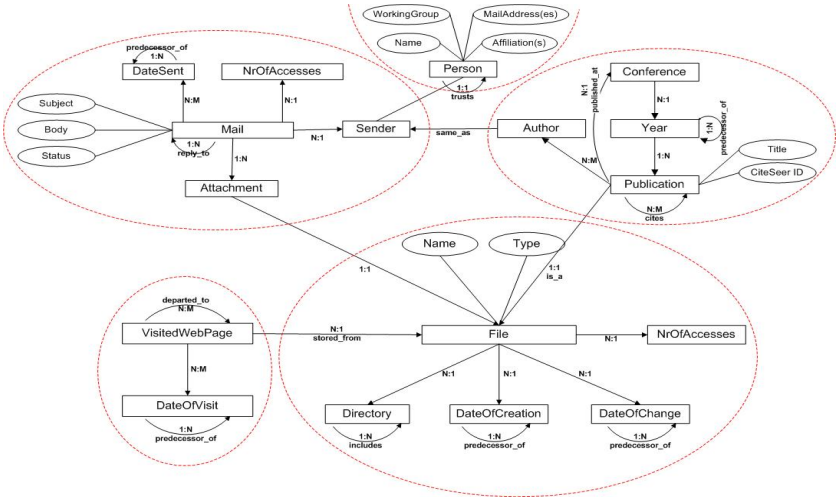

Fig. 2. Contextual Ontology for the Semantic Desktop

if we communicate with an author or if we save a publication from an email attachment. Of course, since each publication is stored as a file, it is also connected to the file context, and thus to the file specific information associated to it (e.g., path, number of accesses, etc.).

3.4 Aggregated Ranking System

Motivation and Overview. As the amount of desktop items has been increasing significantly over the past years, desktop search applications will return more and more hits to our queries. Contextual metadata, which provide additional information about each resource, result in even more search results. A measure of importance is therefore necessary, which enables us to rank these results. The following paragraphs describe such a ranking mechanism, developed based on the Google PageRank algorithm [10].

Basic Ranking. Given the fact that rank computation on the desktop would not be possible without the contextual information, which recreates the links among resources, annotation ontologies should describe all aspects and relationships among resources influencing the ranking. The identity of the authors for example influences our opinion of documents, and thus "author" should be represented explicitly as a class in our publication ontology.

Second, we have to specify how these aspects influence importance. ObjectRank [3] has introduced the notion of authority transfer schema graphs, which extend schemas similar to the ontologies previously described, by adding weights and edges in order to express how importance propagates among the entities and resources inside the ontology. These weights and edges represent authority transfer annotations, which extend our context ontologies with the information

we need to compute ranks for all instances of the classes defined in the context ontologies.

Figure 3 depicts our context ontology plus appropriate authority transfer annotations. For example, authority of an email is split among the sender of the email, its attachment, the number of times that email was accessed, the date when it was sent and the email to which it was replied. If an email is important, the sender might be an important person, the attachment an important one and/or the number of times the email was accessed is very high. Additionally, the date when the email was sent and the previous email in the thread hierarchy also become important. As suggested in [3], every edge from the schema graph is split into two edges, one for each direction. This is motivated by the observation that authority potentially flows in both directions and not only in the direction that appears in the schema: if we know that a particular person is important, we also want to have all emails we receive from this person ranked higher. The final ObjectRank value for each resource is calculated based on the PageRank formula (presented in Section 4.3).

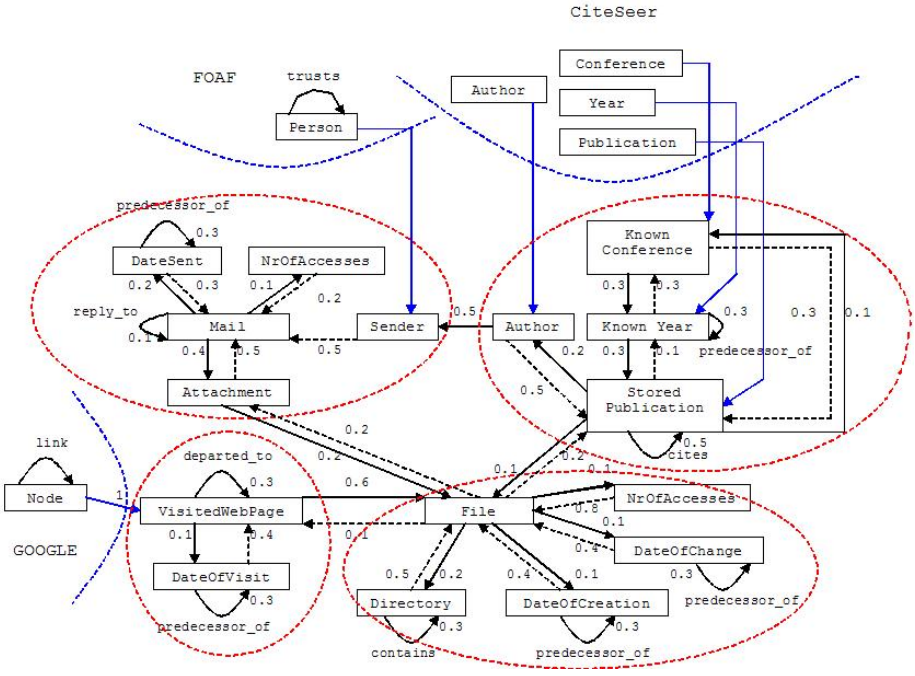

Fig. 3. Contextual Authority Transfer Schema

Using External Sources. For the computation of authority transfer, we can also include additional external ranking sources to connect global ranking computation and personalized ranking of resources on our desktop. These exter-

nal ranking sources are used to provide the seed values for the calculation of the personal ranking. Our prototype ontology includes three global ranking services, one returning Google ranks, the second one ranks computed from the CiteSeer database and the last one from the social network described with FOAF.

The ObjectRank value for each resource is calculated based on the PageRank formula and the seed values for this computation integrate information from external ranking systems and personalized information. We use the following external ranking systems as the most relevant for our purpose:

- *Ranking for articles.* Co-citation analysis is used to compute a primary rank for the article [12]. Because of the sparse article graph on each desktop this rank should be retrieved from a server that stores the articles (in our case all metadata from CiteSeer and DBLP).
- *Recommendations.* We may receive documents from other peers together with their recommendations. These recommendations are weighted by a local estimate of the sender's expertise in the topic [7].

Personalization. Different authority transfer weights express different preferences of the user, translating into personalized ranking. The important requirement for doing this successfully is that we include in a users ontology all concepts, which influence her ranking function. For example, if we consider a publication important because it was written by an author important to us, we have to represent that in our context ontology. Another example are digital photographies, whose importance is usually heavily influenced by the event or the location where they were taken. In this case both event and location have to be included as classes in our context ontology. The user activities that influence the ranking computation have also to be taken into account, which translates to assigning different weights to different contexts.

4 Prototype

Our current prototype is being built on top of the open source Beagle desktop search infrastructure, which we extended with additional modules: metadata generators, which handle the creation of contextual information around the resources on the desktop, and a ranking module, which computes the ratings of resources so that search results are shown in the order of their importance. The advantage of our system over existing desktop search applications consists in both the ability of identifying resources based on an extended set of attributes – more results, and of presenting the results according to their ranking – to enable the user to quickly locate the most relevant resource.

4.1 Current Beagle 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. This relies on notification functionalities provided by the kernel. 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. Much of this basic notification functionality is provided on Linux by an inotify-enabled Linux kernel, which is used by Beagle.

Our prototype keeps all the basic structure of Beagle and adds additional modules that are responsible for generating and using the contextual annotations enriching the resources on our desktop. The main components of the extended Beagle prototype are Beagled^{++} and Best^{++}, as seen in Figure 4, "++" being used to denote our extensions. Beagled^{++} is the main module that deals with indexing of resources on the desktop and also retrieving the results from user queries. Best^{++} is responsible for the graphical interface, put on top of Beagled^{++}, communicating through the DBus daemon. Whenever a DBus daemon launches, a new bus is initialized and to this one both Beagled^{++} and Best^{++} connect and register. Then Beagled^{++} starts a global scheduler thread and the inotify thread, which are responsible for the indexing of new resources found on the desktop. The query driver is then started and waits for queries. The Best^{++} interface is also initialized, using this module queries will be transmitted to Beagled^{++} and answers visualized.

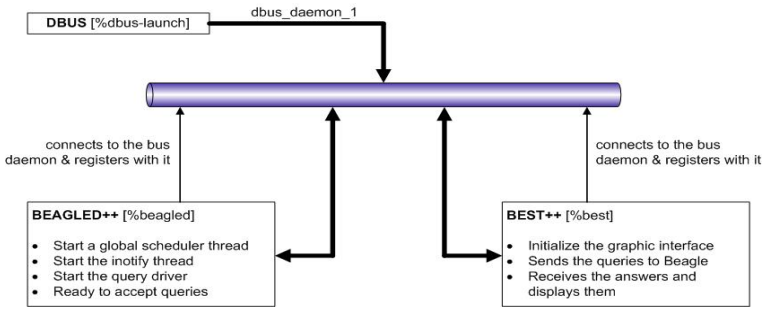

Fig. 4. Extended Beagle Desktop Search

4.2 Extending Beagle with Metadata Generators

Depending on the type and context of the file / event, metadata generation is performed by appropriate metadata generators, as described in Figure 5. These applications build upon an appropriate RDFS ontology as shown in [4], describing the RDF metadata to be used for that specific context. Generated metadata are either extracted directly (e.g. email sender, subject, body) or are generated using the appropriate association rules plus possibly some additional background knowledge. All of these metadata are exported in RDF format, and added to a metadata index, which is used by the search application together with the usual full-text index.

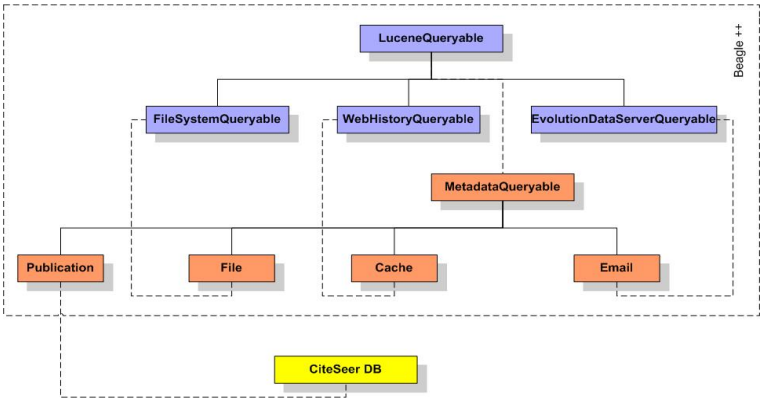

Fig. 5. Beagle Extensions for Metadata Support

The architecture of our prototype environment includes four prototype metadata generators according to the types of contexts described in the previous sections. We added a new subclass of the LuceneQueryable class, MetadataQueryable, and, from this one, derived four additional subclasses, dealing with the generation of metadata for the appropriate contexts (Files, Web Cache, Emails and Publications). The annotations we create include the corresponding elements depicted in the ontology graph Figure 2. They are described in detail in [4]. A new one is the publication metadata generator, described in the next paragraph.

Publication Metadata Generator. For the experiments described in this paper, we have implemented a metadata generator module, which deals with publications. For each identified paper, it extracts the title and tries to match it with an entry into the CiteSeer publications database. If it finds an entry, the application builds up an RDF annotation file, containing information from the database about the title of the paper, the authors, publication year, conference, papers which cite this one and other CiteSeer references to publications. All annotation files corresponding to papers are merged in order to construct the RDF graph of publications existing on one's desktop.

4.3 Extending Beagle with A Ranking Module

Each user has his own contextual network / context metadata graph and for each node in this network the appropriate ranking as computed by the algorithm described in section 3.4. The computation of rankings is based on the link structure of the resources as specified by the defined ontologies and the corresponding metadata. We base our rank computation on the PageRank formula

$$r = d \cdot A \cdot r + (1 - d) \cdot e \tag{1}$$

applying the random surfer model and including all nodes in the base set. The random jump to an arbitrary resource from the data graph is modeled by the

vector e. A is the adjacency matrix which connects all available instances of the existing context ontology on one's desktop. The weights of the links between the instances correspond to the weights specified in the authority transfer annotation ontology. Thus, when instantiating the authority transfer annotation ontology for the resources existing on the users desktop, the corresponding matrix A will have elements which can be either 0, if there is no edge between the corresponding entities in the data graph, or they have the value of the weight assigned to the edge determined by these entities, in the authority transfer annotation ontology. Additionally, in the case of publications, for the rank computation we also take into account the ratings extracted from the CiteSeer database, with the aid of our publication metadata generator. This values are used as seed values for the calculation of the personalized rankings.

5 In-Depth Scenario and Results

We have considered a set of test scenarios for validating our assumptions about the benefits of our metadata enhanced search engine. Let us look at one of them in more detail. We assume that Bob and Alice are two team members of a computer science research institute, both being interested in semantic web technologies. Alice is currently writing a paper about searching and ranking on the semantic desktop and she wants to find some good papers on this topic, which she remembers she stored sometime ago on her desktop. Figure 6 depicts a small part of the set of publications existing on Alice's desktop, along with the contextual metadata associated with them. Papers are connected among each others by the 'cites'- relationship. By dotted lines we represent the papers that are not stored on Alice's desktop, but are referenced by some of the saved ones. In order to find them she makes use of the extended desktop search engine and issues a query with the searched terms **'semantic desktop'**. Let us look at the results she will find.

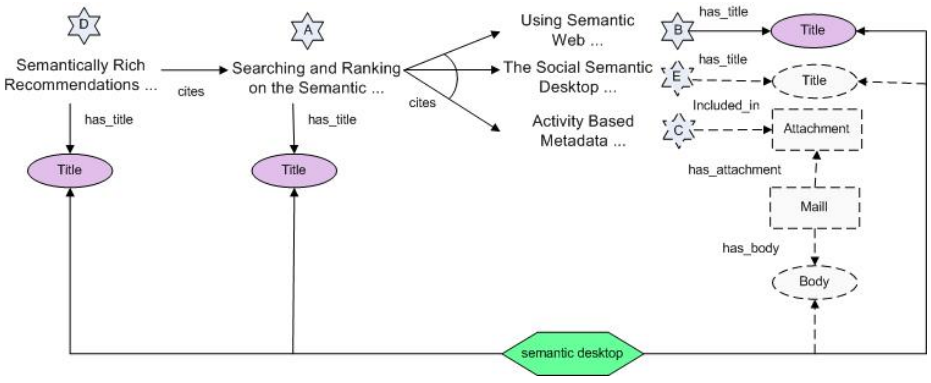

Fig. 6. Papers example

The next three subsections discuss this scenario in more detail. We describe the type of hits our extended Beagle engine returns, show how we display these results together with their additional metadata and how ranking information is used to order the search results.

5.1 Direct Hits vs. Metadata Hits

The hits returned by any normal search engine (direct hits) would be the ones that contain the searched keywords in the title or in the content of the publication, in our case publications **A**, **B** and **D** ("Searching and Ranking on the **Semantic Desktop**", "Using Semantic Web Technologies to Build a **Semantic Desktop**", "Semantically Rich Recommendations in Social Networks for Sharing and Exchanging Semantic Context"). Traditional search engines would not be able to retrieve the other two publications. Publication **C** is included as email attachment, and not indexed by Beagle. However, the corresponding email text contains the searched keywords, and therefore we can retrieve it as an indirect hit (from the metadata as for each email we automatically store the email text as metadata for all attachments). Publication **E** is not even stored on the computer but is among the cited publications by another stored publication(**A**) which contains the keywords in the title, and therefore will be returned as a metadata hit("The Social **Semantic Desktop**").

We extend the Best interface provided by Beagle to include the metadata hits together with the direct ones. The direct hits are shown as Beagle normally does, with the occurrences of the searched terms emphasized. For the indirect hits we display the resources whose associated metadata include the query terms. Our example shows the first 5 out of a total of 20 results. As depicted in Figure 7, the first hit is an email having as attachment a publication not stored on the computer but refers to the topic in the body of the email, which contains the searched keywords. The last result in this picture is another indirect hit but the resource that is displayed is neither explicitly nor implicitly (e.g. in the email attachment) stored on the computer. This is why the user is redirected to the results provided by Google when searching for the item that it is pointed to by the metadata file.

5.2 Rankings

Results. As Figure 7 shows, the results are displayed according to the computed resource rankings. The first hit has the highest rank among the resources in the result set. The rank values are also shown so that the user has an impression about how relevant the results are. In our case the most important result is the email Alice received from Bob including in its attachment a publication. Alice has exchanged her context with Bob, as well as with Caroline, Dan and Tom. All these persons have a high level of trust for Bob and therefore, by exchanging context information, in the resulting graph the node corresponding to Bob will have many incoming links. This translates into a high rank value for Bob. As we suggested in Figure 3, certain percentages of Bob's rank will flow towards all

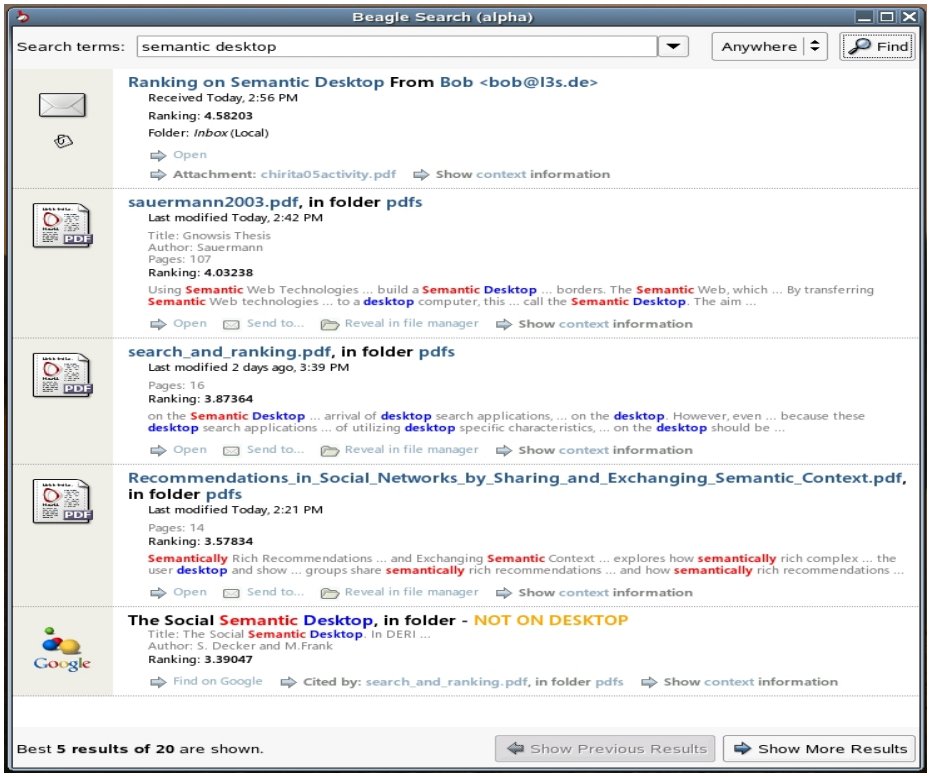

Fig. 7. Beagle^{++} Main Window

nodes that Bob's node points to. Since Bob is the sender of the email which was identified by Beagle as a match for the searched terms, the rank of this hit has a very high value.

We additionally observe that the last hit from this partial set of results, in spite of the fact that is not stored on the desktop, has also a high rank because it is cited by the third hit of this query. Its high rank is also influenced by other resources that the publication receives links from.

Discussion. Ranking algorithms are very much influenced by the structure of their underlying graph. More specifically, different distributions of links yield rather different resource orderings. Luckily, the cached web pages and the scientific publications do exhibit a web-like link distribution (i.e., power-law), and are thus suitable for such a ranking scheme as ours. However, even though in our initial experiments from [4] and from this paper we found the desktop resource graphs to show an encouraging structure (also when emails and regular files have been included), we are currently validating these results on a larger scale of users and search scenarios.

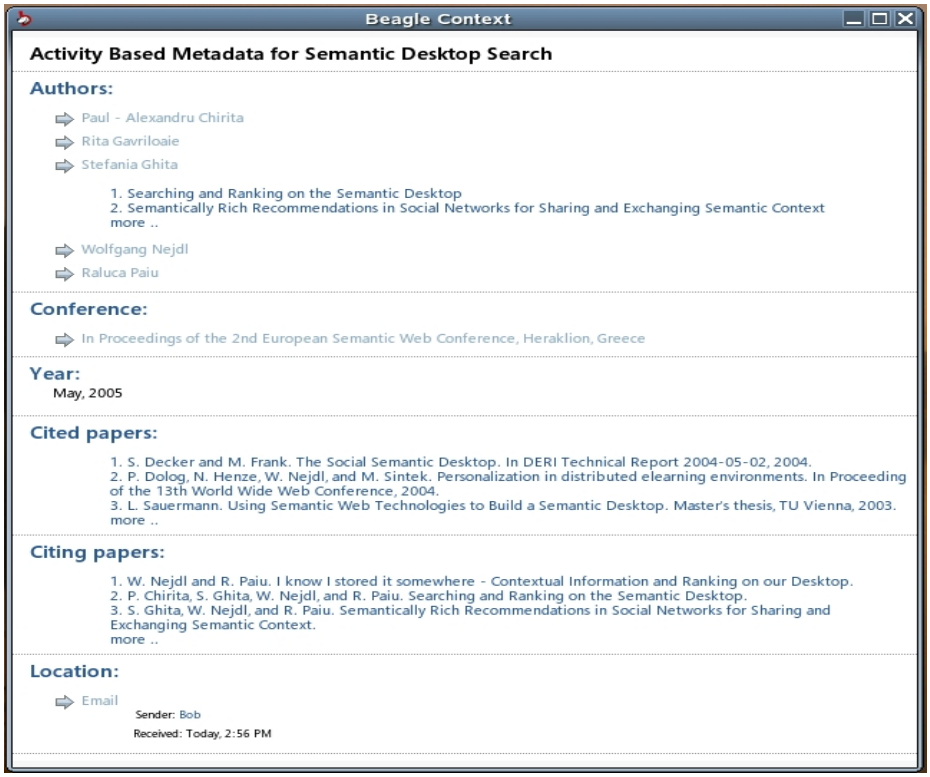

Fig. 8. Beagle^{++} Metadata Window

5.3 Metadata Visualization

Whenever a user clicks on the 'Show context information' link for a certain result, the corresponding metadata can be visualized, both for direct or indirect hits. A new window pops up displaying a list of details that correspond to the ontology related to the type of resource. Alice chooses to visualize the first result returned by Beagle, representing an email from Bob and having as attachment a publication. Since the interesting resource for this query is the PDF file in the attachment, the metadata window displays the annotations corresponding to publications together with other contextual information associated with it. The publication "Activity Based Metadata for Semantic Desktop Search" 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.

6 Conclusions and Future Work

We presented two main contributions that enhance traditional desktop search, focusing on how regular text-based desktop search can be enhanced with semantics / contextual information and ranking exploiting that information. Searching for resources then will not only retrieve explicit results but also items inferred from the users' existing network of resources and contextual information. Maintaining the provenance of information can help the search engine take into account the recommendations from other users and thus provide more retrieved results. The ranking module, by exploiting contextual information, improves retrieval and presentation of search results, providing more functionality to desktop search.

There are quite a few interesting additional contexts, that are worth investigating in the future: metadata embedded in multimedia files, the relations between objects embedded within each other (a presentation including pictures, tables, charts, etc.), or chat history. A further interesting question we want to investigate in the future is how to learn contextual authority transfer weights from user feedback on ranked search results.

References

1. B. Aleman-Meza, C. Halaschek, I. B. Arpinar, and A. Sheth. Context-aware semantic association ranking. In *Semantic Web and Databases Workshop*, 2003.
2. Apple spotlight search. http://developer.apple.com/macosx/tiger/spotlight.html.
3. A. Balmin, V. Hristidis, and Y. Papakonstantinou. Objectrank: Authority-based keyword search in databases. In *VLDB*, Toronto, Sept. 2004.
4. P. A. Chirita, R. Gavriloaie, S. Ghita, W. Nejdl, and R. Paiu. Activity based metadata for semantic desktop search. In *In Proceedings of the 2nd European Semantic Web Conference*, Heraklion, Greece, May 2005.
5. L. Ding, T. Finin, A. Joshi, R. Pan, R. S. Cost, Y. Peng, P. Reddivari, V. C. Doshi, and J. Sachs. Swoogle: A search and metadata engine for the semantic web. In *in Proceedings of the Thirteenth ACM Conference on Information and Knowledge Management*, Washington, DC, Nov. 2004.
6. S. Dumais, E. Cutrell, J. Cadiz, G. Jancke, R. Sarin, and D. C. Robbins. Stuff i've seen: A system for personal information retrieval and re-use. In *Proc. of the 26th Intl. ACM SIGIR Conference*, Toronto, July 2003.
7. S. Ghita, W. Nejdl, and R. Paiu. Semantically rich recommendations in social networks for sharing and exchanging semantic context. In *ESWC Workshop on Ontologies in P2P Communities*, 2005.
8. Gnome beagle desktop search. http://www.gnome.org/projects/beagle/.
9. Google desktop search application. http://desktop.google.com/.
10. Google search engine. http://www.google.com.
11. Msn desktop search application. http://beta.toolbar.msn.com/.
12. A. Sidiropoulos and Y. Manolopoulos. A new perspective to automatically rank scientific conferences using digital libraries. In *Information Processing and Management 41 (2005) 289Z12*, 2005.
13. N. Stojanovic, R. Studer, and L. Stojanovic. An approach for the ranking of query results in the semantic web. In *ISWC*, 2003.
14. J. Teevan, C. Alvarado, M. Ackerman, and D. R. Karger. The perfect search engine is not enough: A study of orienteering behavior in directed search. In *CHI*, 2004.