

Collaboration on the Social Semantic Desktop

Gerald Reif¹, Tudor Groza², Siegfried Handschuh², Cédric Mesnage³, Mehdi Jazayeri³, and Rosa Gudjonsdottir⁴

¹ Software Evolution and Architecture Lab, University of Zurich,
Binzmuelestrasse 14, CH-8050 Zurich, Switzerland

reif@ifi.unizh.ch

<http://seal.ifi.unizh.ch/>

² DERI, National University of Ireland, Galway,
IDA Business Park, Lower Dangan, Galway, Ireland

{tudor.groza, siegfried.handschuh}@deri.org

<http://www.deri.org/>

³ Faculty of Informatics, University of Lugano,
Via Giuseppe Buffi 6, CH-6900 Lugano, Switzerland

cedric.mesnage@lu.unisi.ch

<http://www.inf.unisi.ch/>

⁴ Department of Human-Computer Interaction, KTH
Lindstedtsvägen 6, SE-100 44 Stockholm, Sweden

rosag@kth.se

<http://www.csc.kth.se>

Abstract. To accomplish the daily work people use several desktop applications to collaborate with co-workers. Each application is specialized on a specific domain, such as document management, email, or time planning. Although the data is distributed over several applications the data is highly interlinked from the user's point of view. The Social Semantic Desktop aims to take advantage of Semantic Web technologies on the computer's desktop to better support the user's mental working model and to enable collaboration over enterprise boundaries. In this paper we present our ongoing work on the Social Semantic Desktop as collaboration environment. We present the intended usage scenarios, discuss the required services and give an outlook on the architecture we envision for the Social Semantic Desktop.

1 Introduction

In our every day work people are collaborating on tasks by sharing information and by communicating with each other. To fulfill their tasks users utilize several applications, each specialized on a specific domain, such as document management, email, contacts, or time planning. Although the data is distributed over several applications the data is highly interlinked from the user's point of view: Calendar entries are used to schedule meetings; meetings are related to projects; documents are developed within projects; emails are used to arrange meetings or to discuss documents; and so on. However, this mental model of the user is not yet appropriately represented by the desktop of current operating systems.

In addition, in a collaborative working scenario the data is spread over several desktops. Current collaboration systems offer possibilities such as sharing documents with co-workers or calendar sharing. But still, the users use its desktop applications to create and edit documents or to communicate via email or chat. Even if a cooperation uses a collaboration suite within its organization that integrates these applications, there are still users outside the organization that do not have access to the system (*e.g.*, for security reasons).

The Social Semantic Desktop (SSD) [4, 8] adopts the ideas from the Semantic Web paradigm, which aims to provide a common framework to enable information to be shared across application, enterprise and community boundaries [9]. Formal ontologies capture both a shared conceptualization of desktop data and personal mental models. The Resource Description Framework (RDF) [5] serves as the common data representation format between applications. This way the SSD allows information originating from individual applications distributed over several computers to be associated with each other according to its semantic context and enables novel views on the user's data better conforming to the user's mental model.

The European Union project NEPOMUK ⁵ aims at empowering individual knowledge workers to better exploit their personal information space and to enable collaboration within social networks across organizational boundaries. NEPOMUK brings together researchers, industrial software developers, and representative industrial users, to develop a comprehensive solution for extending the personal desktop into a collaboration environment which supports both the personal information management and the sharing and exchange across social and organizational relations.

In this paper we present our ongoing work to define a reference architecture for the SSD. In Section 2 we summarize the usage scenarios gathered from the use case project partners. Section 3 abstracts the usage scenarios and presents the identified core and end-user specific services needed to realize the SSD. Section 4 gives an outlook on the architecture we envision for the SSD and Section 5 concludes the paper.

2 Usage Scenarios

The study of users' needs with regard to collaboration on the SSD is a major goal of the NEPOMUK project. These studies were carried out at the use case partners in the project which are companies and research labs working in the area of business software, biomedical research, Linux development, and educational consulting. Interviews with the use case partners resulted in 40 usage scenarios, illustrating the user expectations on the SSD. In this section we summarize the usage scenarios with the help of fictional characters and focus on collaboration aspects.

⁵ This work is funded by the European Commission 6th Framework Programme in context of the EU IST NEPOMUK IP - The Social Semantic Desktop Project, FP6-027705.

Dirk gets task from Claudia. Claudia is working on a project deliverable and she identifies some tasks to be done. She adds the tasks to the project and assign them to Dirk. Dirk is notified of his new tasks he accepts the responsibility to acknowledge Claudia. Dirk realises that some tasks require more specific knowledge he decline them and suggests Martin. Claudia reassign the tasks to Martin.

Josephine follows-up the project plan Josephine is following up on an active project she is administrating. It involves Karen and a few other trainers. She accesses the project plan and browses to see if everything is on schedule. The project plan is connected to the trainers' calendars and all changes in their calendar get fed directly into the project plan.

Karen edits a document with another person Karen is to give a presentation for an existing client in a couple of days. The presentation is new and the purpose is to sum up a series of training programs she has performed for the client. When she is working on a slide she can see a new graphical layout suggestion by Josephine who is concurrently working on the presentation's graphic form. The system allows them to collaborate, make changes, discuss and explain their intentions and thoughts.

Karen shares experience Karen finds the time to take care of some administration issues. She just finished a project successfully and finds the experiences should be shared with her colleagues. She opens the course material, marks it "Shared", and adds a few keywords to make sure that people interested find the material.

3 Services of the Social Semantic Desktop

From the usage scenarios, presented in the section above, we extracted the services needed to build the SSD and clustered the services into five groups. In addition we identified services that are core to realize the SSD, whereas other services are specific to the scenarios from a use case partner. Figure 1 shows the core services of the SSD, which are discussed in the remainder of this section.

All services have in common that they operate on semantic meta-data. The semantic meta-data is encoded using RDF which uses the vocabulary that is defined in ontologies. In NEPOMUK we distinguish between the *core* and the *usage specific* ontologies. The core ontologies are used to maintain the SSD. This includes issues such as user and document management, access control, or message routing. The usage specific ontologies are used for additional services that can be build on top of NEPOMUK such as a task management system, domain specific semantic annotation of documents or a help-desk knowledge-base.

In the following we give an overview of the (overlapping) service clusters we identified for the SSD NEPOMUK:

Search enables users to *search* for resources (*e.g.*, users, documents, tasks) based on the semantic meta-data. It is further possible to query for resources that are *related* to some given resources based on similarity measures.



Fig. 1. Services of the Social Semantic Desktop.

Desktop focuses on services for the resource *management*. Resources can be created, modified, or viewed. They can be *shared* with other users and synchronized for *offline working*. For the data exchange between desktop applications NEPOMUK provides interfaces that enable services such as a Semantic Clipboard [7]. The SSD should further support *desktop sharing* for collaborative working on resources. NEPOMUK users can also define in their profile how they want to be informed about the occurrence of events (*e.g.*, via email, instant messaging, or SMS).

Profiling is based on *logging* of the user behavior. The logged data is used as *training* data to adapt the desktop behavior to the user's habits. Users can also manually *tailor* the system behavior. A core service of NEPOMUK is the *semantic annotation* of resources. The semantic annotations are either automatically extracted from the desktop applications or added by the user. The annotations provide additional information about resources or put resources into relation (*e.g.*, emails, calendar entries, and documents belong to a project). In addition, users are able to set the *trust* level to other NEPOMUK users, which for example influences the ranking of search results.

Data Analysis takes advantages of the semantically enriched data available in NEPOMUK. The *reasoning* service exploits the logic foundation of ontology based semantic meta-data and uses logical rules to deduce new knowledge. Clustering algorithms are used to *group* resources and to *sort* search results. *Keyword extraction* algorithms are used to suggest resource annotations or to search for related resources.

Social aspects focus on the *sharing* of resources in information spaces (*e.g.*, sharing of documents and calendar entries with co-workers). The access to the resources in the shared information spaces is controlled by the *user group management* and the *access right management*. NEPOMUK provides further a publish/subscribe system where users can subscribe to arbitrary events (*e.g.*, document modified, user profile changed, task completed) which trigger the notification of the user. NEPOMUK supports the *social interaction* between users via synchronous and asynchronous communication techniques.

Although the services introduced above were requested by the use case partners in the NEPOMUK project, not all services will be implemented within the project. Some services such as desktop sharing have been identified as useful but recognized as beyond the scope of the project.

4 Architecture

In this section we present an outlook on the NEPOMUK architecture. The architecture, as shown in Figure 2, is organized in three layers. The lowest layer is the Communication Layer. This layer provides an *Event Based System*, which is responsible for the distribution of the events on between the NEPOMUK peers. The events carry an RDF graph as payload describing the cause of the event. The *Messaging System* routes the messages to receiver. The *Peer-to-Peer File Sharing System* enables the shared information space. It will be based on GridVine [2]. GridVine is based on P-Grid [1] and provides a distributed index which supports RDQL search queries.

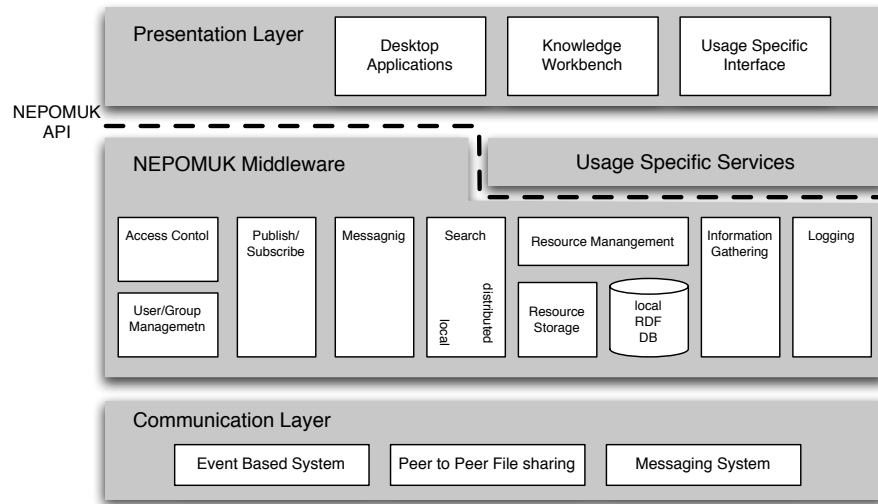


Fig. 2. Layered NEPOMUK Architecture.

On top of the communication layer is the NEPOMUK Middleware. The *Publish/Subscribe System* allows users to subscribe to events in the NEPOMUK system. The subscriptions are stored as SPARQL queries [6] which are matched against the RDF payload of the events. When the subscription, *i.e.*, the SPARQL query, matches the event, the *Messaging System* looks up the preferred notification media (*e.g.*, email, instant messaging, SMS) and delivers the mes-

sages. The Messaging System is further used for synchronous and asynchronous communication between NEPOMUK users.

The *Resource Management* is responsible to control the insertion, modification, and deletion of resources on the NEPOMUK desktop. A resource can be a user, a document, a calendar entry, an email, and so on. It is responsible to store the RDF meta-data in the *local RDF DB* and to maintain the reference to the original resource in the *Resource Storage*. The Resource Storage can be maintained by a desktop application (*e.g.*, the emails managed by an email client). A resource can either be manually added to the NEPOMUK desktop or the *Information Gatherer* extracts the information from desktop applications such as email clients or calendar applications. The Information Gatherer will be implemented based on Aperture [3]. For local queries and for offline working the meta-data is stored in the local RDF DB. If a resource is shared with other users in an information space, the meta-data is also uploaded to the distributed index of the peer-to-peer file sharing system. The *Search* service can either issue a local search in the local RDF DB or a distributed search in the underlying peer-to-peer system. The *Logging* service writes user actions for further analysis to local RDF DB.

Actions a user performs on the shared information space have to be approved by the *Access Control* System. Depending on the group membership of a user, maintained in the *User/Group Management*, the Access Control grants the privileges to perform the action. The access rights, the user, and the group data are stored as RDF graphs in the distributed index of the peer-to-peer system. This data is encoded using the access right ontology and the user/group ontology, which belong to the NEPOMUK core ontologies.

The NEPOMUK Middleware provides the core services of the NEPOMUK architecture. These services can be accessed via the NEPOMUK Application Programming Interface (API). An application programmer can build usage specific services on top of the NEPOMUK API. By using the functionality provided by the API, the programmer can implement new functionality according to the end-users' business requirements. Hence, the basic set of services provided by the NEPOMUK API can be customized and extended by businesses and organizations. For example, a company might be interested in integrating task management system whereas another might be interested in having document versioning support for resources.

The top layer of the architecture is the presentation layer. It provides a user interface to the services provided by the NEPOMUK desktop. The presentation layer is built using the NEPOMUK API. Many desktop applications are possible sources for resources that should be managed by NEPOMUK. Therefore, each desktop application should integrate support for the NEPOMUK Middleware. Since this assumption does not hold for most of the applications, we developed plugins and add-ons to enable a seamless integration for popular applications such as the MS Office Suite, which for example extract email or calendar data and adds them as resources to the NEPOMUK desktop.

In addition, the Knowledge Workbench is the central place to browse, query, view, and edit resources and their meta-data. This way the Knowledge Workbench aims to replace current file management tools such as the MS File Explorer. If the SSD is extended by usage specific services, the application programmer has also to provide the corresponding user interface in the Presentation Layer.

5 Conclusion

In this paper we presented our ongoing work on defining the software architecture for the SSD. We pointed out that the use of Semantic Web technologies improve the collaboration between users and helps to overcome community and enterprise boundaries. The presented NEPOMUK architecture provides the core functionalities of the SSD. The possibility to extend the SSD with usage specific ontologies increases the flexibility to build usage specific services on top of the core NEPOMUK Middleware.

References

1. K. Aberer, P. Cudré-Mauroux, A. Datta, Z. Despotovic, M. Hauswirth, M. Puceva, and R. Schmidt. P-grid: a self-organizing structured p2p system. *SIGMOD Record*, 32(3):29–33, 2003.
2. K. Aberer, P. Cudré-Mauroux, M. Hauswirth, and T. V. Pelt. Gridvine: Building internet-scale semantic overlay networks. In *3th International Semantic Web Conference ISWC 2004*, pages 107–121. Springer Verlag, 2004.
3. Aperture a java framework for getting data and metadata, Last visited March 2007. <http://aperture.sourceforge.net/>.
4. S. Decker and M. Frank. The networked semantic desktop. In *Workshop on Application Design, Development and Implementation Issues in the Semantic Web at the 13th International World Wide Web Conference*, New York, USA, May 2004. CEUR Workshop Proceedings. <http://CEUR-WS.org/Vol-105/>.
5. D. B. ed. *RDF/XML Syntax Specification (Revised)*. W3C Recommendation, 10 February 2004. <http://www.w3.org/TR/rdf-syntax-grammar/>.
6. E. Prud'hommeaux and A. S. eds. SPARQL query language for RDF. W3C Working Draft, 4. October 2006. <http://www.w3.org/TR/rdf-sparql-query/>.
7. G. Reif, M. Morger, and H. Gall. Semantic clipboard - semantically enriched data exchange between desktop applications. In *Semantic Desktop and Social Semantic Collaboration Workshop at the 5th International Semantic Web Conference ISWC06*, Athens, Georgia, USA, November 2006.
8. L. Sauermann. Using semantic web technologies to build a semantic desktop. Master's thesis, Vienna University of Technology, 2003.
9. World Wide Web Consortium (W3C) Semantic Web activity homepage. <http://w3c.org/sw>.