Integrated Project

Priority 2.4.7

Semantic based knowledge systems

The Social Semantic Desktop

Eclipse Community Involvement

Deliverable D7.3

Version 1.0

07.08.2007

Dissemination level: PU

Nature	Other
Due date	31.07.2007
Lead contractor	EDGE-IT S.A.R.L
Start date of project	01.01.2006
Duration	36 months

Authors

David Barth, EDGE-IT S.A.R.L
Costas Christidis, NTUA ICCS
Stéphane Laurière, XPertNet
Fabio Mancinelli, XPertNet
Leo Sauermann, DFKI

Mentors

Ansgar Bernardi, DFKI
Mikhail Kotelnikov, Cognium Systems

Project Co-ordinator

Dr. Ansgar Bernardi
German Research Center for Artificial Intelligence (DFKI) GmbH
Erwin-Schroedinger-Strasse (Building 57)
D 67663 Kaiserslautern
Germany
Email: bernardi@dfki.uni-kl.de, phone: +49 631 205 3582, fax: +49 631 205 4910

Partners

DEUTSCHES FORSCHUNGSZENTRUM F. KUENSTLICHE INTELLIGENZ GMBH
IBM IRELAND PRODUCT DISTRIBUTION LIMITED
SAP AG
HEWLETT PACKARD GALWAY LTD
THALES S.A.
PRC GROUP - THE MANAGEMENT HOUSE S.A.
EDGE-IT S.A.R.L
COGNIUM SYSTEMS S.A.
NATIONAL UNIVERSITY OF IRELAND, GALWAY
ECOLE POLYTECHNIQUE FEDERALE DE LAUSANNE
FORSCHUNGSZENTRUM INFORMATIK AN DER UNIVERSITAET KARLSRUHE
UNIVERSITAET HANNOVER
INSTITUTE OF COMMUNICATION AND COMPUTER SYSTEMS
KUNGLIGA TEKNISKA HOEGSKOLAN
UNIVERSITA DELLA SVIZZERA ITALIANA
IRION MANAGEMENT CONSULTING GMBH

Versions

Version	Date	Reason
0.1	04.07.2007	First draft by Stéphane Laurière
0.2	20.07.2007	Integration of contents by Leo Sauermann from the position paper on the PSEW architecture.
0.3	30.07.2007	Integration of the description of the PSEW prototype by Fabio Mancinelli.
1.0	07.08.2007	Updates taking into account the remarks by the mentors, final streamlining by M. Junker

Explanations of abbreviations on front page

Nature
R: Report
P: Prototype
R/P: Report and Prototype
O: Other

Dissemination level
PU: Public
PP: Restricted to other FP6 participants
RE: Restricted to specified group
CO: Confidential, only for Nepomuk partners

Executive summary

To be really successful, the Nepomuk project needs not only to realize and deploy an innovative approach for collaborative knowledge work, but also to integrate the Nepomuk framework into mainstream desktop environments and development platforms. The Nepomuk community initiative focuses on involving at least the KDE, Eclipse and Mozilla communities of users and developers into the Nepomuk project.

While the Nepomuk report "D7.2 - KDE Community Involvement"[1] described the outcomes of the Nepomuk-KDE initiative, the main goal of the work described in this report centers around the steps that have been undertaken to get the Eclipse community involved in Nepomuk. To this end, an analysis of the existing community and of the relevant projects already ongoing has been carried out, a Nepomuk Eclipse community platform has been started, and a RCP-based reference implementation of Nepomuk core components has been developed. These activities all together comprise the Nepomuk-Eclipse sub-project.

In order to present the project and to raise interest across the community, a dedicated Web site[2] and a public mailing-list [3] have been set up. The Nepomuk-Eclipse project was presented to several audiences interested in Eclipse technologies, and synergies with potential similar projects have been identified.

A first version of a Nepomuk demonstrator using Eclipse Rich Client Platform (Eclipse RCP) technology has been designed. Eclipse RCP is the minimal set of Eclipse plug-ins needed to build a rich desktop application. It consists of a development framework including tools for dealing with text, forms, actions and any other component needed to create advanced applications. The Nepomuk Eclipse RCP demonstrator is dubbed PSEW, standing for "P2P Semantic Eclipse Workbench". PSEW introduces a user interface for interacting with the services provided by the Nepomuk components RDF API, RDF repository and DataWrapper. As of July 2007, PSEW lets the user launch the indexing process of desktop resources, explore the extracted metadata and complete it by manual annotations. Future versions of the prototype will provide enhanced semantic features, will include the capacity to share the local indexes over a P2P network using the Nepomuk distributed index, and will put into practice the Nepomuk community detecter and labeller.

The PSEW prototype will be used and extended by several Nepomuk case studies, serving as a foundation for an Eclipse based user interface to the Nepomuk Social Semantic Desktop. The upcoming versions of PSEW will include P2P capabilities, user context handling, social features and advanced visualization capabilities.

In parallel to the further development of the PSEW prototype, the focus will be brought in the next months on the involvement of more Eclipse developers into the Nepomuk-Eclipse project, on presenting the project to a wider audience on the occasion of Eclipse related events, and on increasing the presence of the project on the Web.

The availability of the implementations of the Nepomuk specification on top of the two frameworks KDE and Eclipse RCP paves the way for fruitful comparisons. It is expected that cross-fertilization between the two communities will take place, hence bolstering the adoption and the further enhancement of the Nepomuk standards.

[1] http://nepomuk.semanticdesktop.org/xwiki/bin/view/Main1/D7-2
[2] http://nepomuk-eclipse.semanticdesktop.org
[3] https://nepomuk.semanticdesktop.org/wws/info/nepomuk-eclipse

Table of contents

1 Introduction

Nepomuk intends to realize and deploy a comprehensive solution – methods, data structures, and a set of tools – for extending the personal computer into a collaborative environment, which improves the state of art in online collaboration and personal data management and augments the intellect of people by providing and organizing information created by single or group efforts.

Community involvement

To be really successful, the Nepomuk project needs not only to realize and deploy an innovative approach for collaborative knowledge work, but also to integrate the Nepomuk framework into mainstream desktop environments and development platforms. The Nepomuk community initiative focuses on involving at least the KDE, Eclipse and Mozilla communities of users and developers into the Nepomuk project.

Eclipse community

The main goal of the work described in this report is the involvement of the Eclipse community into the Nepomuk project. To this end, an analysis of the existing community and of the relevant projects already ongoing has been carried out, a Nepomuk Eclipse community platform has been started, and a reference implementation of Nepomuk core components based on Eclipse Rich Client Platform (Eclipse RCP) [11] has been developed. These activities all together comprise the Nepomuk-Eclipse sub-project.

Involving several key desktop communities into the initiative is considered by the Nepomuk Consortium as a mean for guaranteeing that the Nepomuk standards, ontologies and APIs will be sustained beyond the end of the project itself. It is expected that the implementations of the standards on top of different frameworks will encourage a competitive spirit between the targeted platforms, thus bolstering cross-fertilization of ideas between the communities, to the benefit of the future versions of the Nepomuk specification.

Figure 1: Nepomuk targeted communities

Given the importance of the Eclipse platform eco-system in the industry and of its growing importance on the desktop, it has appeared crucial from the outset to the Nepomuk Consortium to provide a demonstrator of the Nepomuk capabilities on top of the Eclipse framework.

Eclipse RCP

Eclipse RCP is the minimal set of Eclipse plug-ins needed to build a rich desktop application. It consists of a development framework including generic tools for dealing with text, forms, actions, etc. While Eclipse RCP is not a desktop environment as such that can be directly compared to frameworks such as KDE or GNOME, the growing number of applications built on top of Eclipse RCP make the technology closer and closer to a full-fledged desktop environment. Eclipse RCP includes the capability to embed external applications such as OpenOffice.org or Mozilla Firefox. The recent migration of the Lotus application to Eclipse RCP is a good example of the capabilities of the framework for personal information management and for communication purposes.

P2P Semantic Eclipse Workbench

The Eclipse RCP implementation of Nepomuk is named PSEW, standing for "P2P Semantic Eclipse Workbench". PSEW introduces a user interface for interacting with the services provided by the Nepomuk components RDF API, RDF repository and DataWrapper. As of July 2007, PSEW lets the user launch the indexing process of desktop resources, explore the extracted meta-data and complete it by manual annotations. Future versions of the prototype will provide enhanced semantic features, will include the capacity to share the local indexes over a P2P network using the Nepomuk distributed index, and will put into practice the Nepomuk community detecter and labeller.

Overview

This document presents what has been accomplished in the Nepomuk-Eclipse project so far. It consists of the following sections:

1. Section 2 gives an introduction to Eclipse in general. It presents specific Eclipse communities and projects that are of interest to the Nepomuk-Eclipse initiative;
2. In section 3, the steps that have been undertaken to raise awareness around the project are presented;
3. Section 4 introduces the Nepomuk PSEW prototype: its objectives and its architecture;
4. Section 5 makes concluding remarks on the progress in the first phase of the project.

This report follows the report "D7.2 - KDE community involvement" [4], of which it reuses the global structure: the work carried out and the strategy for involving external contributors are similar to the Nepomuk-KDE initiative indeed, even though the target communities are different in nature.

[4] http://nepomuk.semanticdesktop.org/xwiki/bin/Main1/D7-2

2 Eclipse community overview and relevant projects

This section reminds of the main facts about the Eclipse eco-system, it underlines the importance of the Eclipse community for the Nepomuk project and it presents an overview of the main Eclipse projects that provide semantic or social features being of interest to the Nepomuk initiative.

2.1 Eclipse community

The Eclipse community is one among the largest open source community in the world.

As stated on the Eclipse portal, the Eclipse project is focused on "building an open development platform comprised of extensible frameworks, tools and runtimes for building, deploying and managing software across the lifecycle"[5].

The main Web entry points to the Eclipse community consist of the following sites:

- Eclipse home: `http://www.eclipse.org`,
- Eclipse community wiki: `http://wiki.eclipse.org`,
- Eclipse catalogue of plug-ins: `http://eclipse-plugins.2y.net/`,
- Eclipse Planet, "a window into the world, work and lives of Eclipse hackers and contributors": `http://planeteclipse.org/`.

Figure 2: Eclipse logo

The following 2006 metrics taken from a presentation by Mike Milinkovich, executive director of the Eclipse Foundation, give an order of magnitude of the importance of the project scope[6]:

- 639 committers,
- Top level Eclipse projects[7]: 10,
- Number of sub-projects: 80,
- Web site visits: 36,000 unique visitors per day, 1,140,000 unique visitors per month (IP addresses),
- Bugs: 126K so far (typically 15K-20K per major release),
- Newsgroup postings: 9K/month,

[5] `http://www.eclipse.org/org/`
[6] `http://barcamp.org/f/EclipseLessonsLearned(BarCampOttawa).pdf`
[7] `http://www.eclipse.org/projects/`

- 70 project leaders, PMC members,
- 24,426 registered Bugzilla users,
- Estimated 2 million ++ users,
- Number of newsgroup posts: 32,223,
- Number of mailing list posts: 7,792,
- Number of Eclipse-specific blogs: 27,
- Average Bugzilla queries per day: 600,000,
- Downloads: 1 million in 40 days for last 2006 release,
- 55-65% market share of Java IDE.

Eclipse RCP being one of the largest community of developers committed to the development of an open-source framework for rich desktop applications together with communities such as KDE or GNOME, it is a natural target for Nepomuk, which aims at disseminating the Social Semantic Desktop approach to the widest possible audience of talented developers.

Since the industrial partners of the Nepomuk Consortium intends to build commercial products on top of the Nepomuk platform, an important aspect of Eclipse in the Nepomuk context is the focus on enabling the use of open-source technology in commercial software solutions. This is made possible by the fact that all Eclipse projects are licensed under the Eclipse Public License (EPL), a commercial friendly OSI approved licensed.

The Eclipse community is organized around the Eclipse Foundation, created in 2004 as "a not-for-profit, member supported corporation that hosts the Eclipse projects and helps cultivate both an open source community and an ecosystem of complementary products and services."[8]

The Eclipse Foundation consists of more than 130 members, including major Java, Linux and embedded vendors (Actuate, HP, IBM, Sybase, BEA, Borland, JBoss, SAP, RedHat, Novell, Nokia, Monta Vista, Wind River, Mentor, ENEA, QNX, ILOG, Thales, SAS, Computer Associates and others)[9]. Hundreds of commercial products are built on Eclipse technology.

One of the great strength of the Eclipse project is the so called "architecture for participation" that has been set up around the project and that has led to a vibrant community putting into practice the Eclipse development process with great efficiency. This architecture in itself is of interest to the Nepomuk work-package WP7000 since it lays down general principles that can be harnessed by the Nepomuk community initiative.

Mike Milinkovich describes the principles of this architecture for participation by quoting Tim O'Reilly:

> "What really distinguishes open source is not just source, but an architecture of participation that includes low barriers to entry by newcomers, and some mechanism for isolating the cathedral from the bazaar. This architecture of participation allows for a real free market of ideas, in which anyone can put forward a proposed solution to a problem; it becomes adopted, if at all, by acclamation and the organic spread of its usefulness."[10]

M. Milinkovich summarizes the implementation of this architecture in the Eclipse context as a set of key points:

[8] http://www.eclipse.org/org/
[9] http://www.eclipse.org/membership/exploreMembership.php
[10] http://www.oreillynet.com/pub/wlg/3017?wlg=yes

- "You need a cathedral to enable the bazaar.
  This is not about joining the project itself per se, but about making it easy for individuals to add their own ideas to the ecosystem
- Low barriers to entry
- New add-ons are first class citizens, not second class
- Empowering individuals and small groups is key. Innovation comes from committed fanatics in small groups, not corporate committees
- Laissez faire — never ever ever pretend you can pick a winner
- Provide the seeds for a broad ecosystem."[11]

Further, at a more technical level, he emphasizes that, in Eclipse RCP, "everything is a plug-in... both from the cathedral and from the bazaar", contrary to an approach that would instead differentiate between the two, as illustrated by figure 3[12].

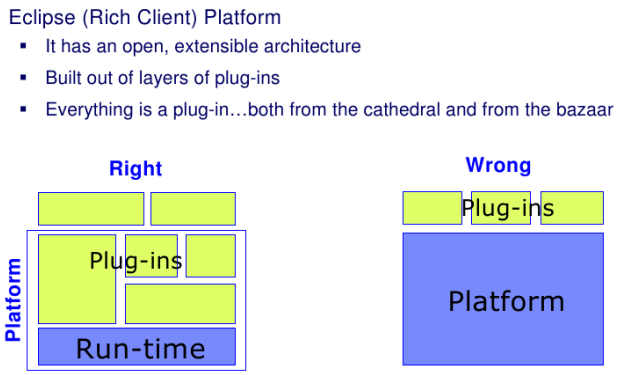

Figure 3: Eclipse RCP – Everything is a plug-in

These key points are serving as useful guidelines for the Nepomuk architecture for participation.

2.2 Eclipse Rich Client Platform

Eclipse was started as a universal platform project for integrating development tools. It became clear after the first release of the Eclipse IDE however, that many components of the Eclipse workbench were not IDE specific, and that advanced desktop applications had similar needs as the Eclipse IDE. For these reasons, the Eclipse Rich Client Platform was introduced in 2002, for providing (i) an open architecture for desktop applications, (ii) an efficient, configurable, portable user interface, (iii) product branding support, (iv) install/update support, (v) integrated help, user configuration/preferences[12].

As illustrated by figure 4, Eclipse RCP is the minimal set of Eclipse plug-ins needed to build a rich desktop application.

Eclipse RCP consists of the following components:

- Runtime: Plug-in model and extension point architecture,
- OSGi: Support for dynamic plug-ins,
- SWT: Cross-platform native widget library, with tight OS integration,

[11] http://barcamp.org/f/EclipseLessonsLearned(BarCampOttawa).pdf
[12] http://eclipse.org/rcp/slides/RCP.ppt

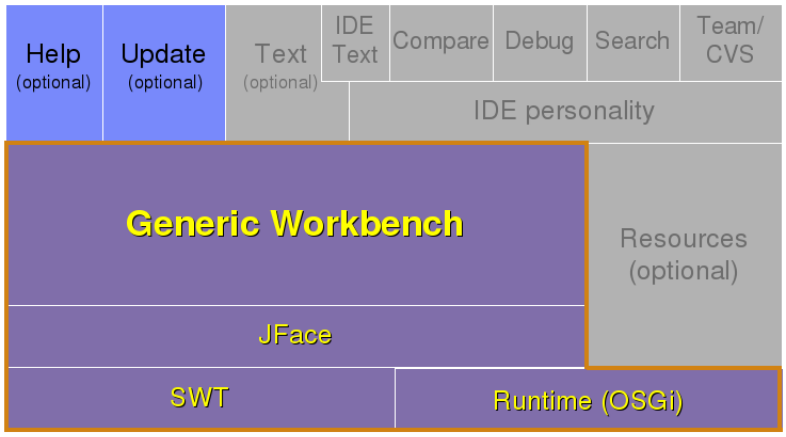

Figure 4: Eclipse Rich Client Platform components

- JFace: UI framework to simplify common tasks,
- Workbench: Highly scalable, managed UI,
- Several optional components: Help UI, Update, Intro, Cheat Sheets, Forms, GEF, EMF, GEF,

It is worth noticing that the base Eclipse RCP is relatively small: disk footprint is 5.5M, and that tool support is provided by the Eclipse plug-in development environment.

2.3 Eclipse projects related to social or semantic technologies

Several existing projects building upon the Eclipse platform – either Eclipse RCP or Eclipse OSGi server side, or both – are addressing the need for an enhanced semantic and social workbench. These projects can be categorized by the type of entity that support them, as illustrated by figure 5.

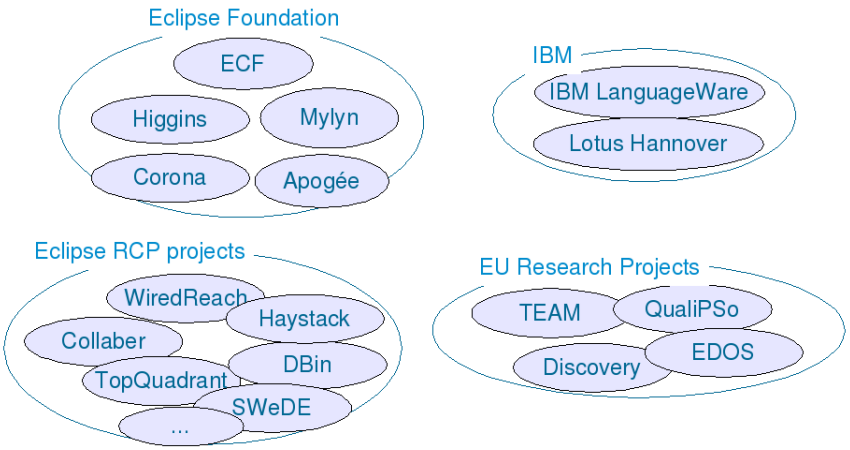

Figure 5: Eclipse communities related to social or semantic technologies

The entities and projects represented in figure 5 can be further described as follow:

- the Eclipse Foundation, endorsing the following projects:

- Apogée[13], an open-source Enterprise Content Management platform with semantic capabilities,
- Mylyn[14], a task-focused UI for Eclipse that reduces information overload and that makes multi-tasking easy,
- Eclipse Communication Framework (ECF)[15], a framework for supporting the development of distributed Eclipse-based tools and applications, providing communication containers for XMPP, IRC, JXTA, JMS and a BitTorrent implementation,
- Higgins[16], an abstraction layer for identity and social networking services,
- Corona[17], an SOA tool integration for service calls across distributed workbenches.

- IBM, developing the Workplace[18] application and the LanguageWare Resource Workbench[19] on top of Eclipse RCP.

  Workplace includes a set of services featuring "integrated communication and collaboration tools such as e-mail, calendaring and scheduling, awareness, instant messaging, e-learning, team spaces, Web conferencing, and document and Web content management"[20].

  LanguageWare provides a variety of linguistic functions including "language identification, spelling correction, lexical analysis, semantic annotation, dictionary look-up and fuzzy look-up, hyphenation, and normalization. LanguageWare can be used to provide efficient annotation of text using the UIMA (Unstructured Information Management Architecture) framework"[21].

- European research projects funded under the IST activities of the European Research Framework programme, such as:
  - TEAM[22], an FP6 STREP project (2006-2008) aiming at "tightening knowledge sharing in distributed software communities by applying semantic technologies". TEAM will include a "Metadata repository that enables efficient structuring and persistent storage of the acquired knowledge, as well as reasoning about its completeness and consistency, and a P2P Infrastructure for decentralised communication between local Knowledge Desktops of software developers."
  - Discovery[23], an eContentPlus project aiming to set up a peer-to-peer network of desktop client applications that allows the generation of new knowledge applying and testing different conceptual grids to the Hyper-federation contents (Discovery) applied to a corpus of philosophical contents. Discovery intends to adapt technologies such as DBin;
  - EDOS[24], an FP6 STREP project (2004-2007) whose objectives among others consists of designing the EDOS Content Dissemination System (EDOS-CDS), a P2P infrastructure for sharing and querying

[13] http://apogee.nuxeo.com
[14] http://www.eclipse.org/mylyn/
[15] http://www.eclipse.org/ecf/
[16] http://www.eclipse.org/higgins/
[17] http://www.eclipse.org/corona/
[18] www.ibm.com/software/workplace
[19] http://www.alphaworks.ibm.com/tech/lrw
[20] http://en.wikipedia.org/wiki/IBM_Workplace
[21] http://www.alphaworks.ibm.com/tech/lrw
[22] http://cordis.europa.eu/fetch?CALLER=PROJ_IST&ACTION=D&RCN=79461&DOC=24&CAT=PROJ&QUERY=1162831024619
[23] http://ec.europa.eu/information_society/activities/econtentplus/projects/cult/discovery/index_en.htm
[24] http://www.edos-project.org

metadata. EDOS-CDS provides an Eclipse RCP front-end, available from the project web site under the LGPL license;

- QualiPSo[25], an FP6 Integrated Project aiming at designing a software forge of a new generation, including P2P and semantic capabilities. QualiPSo is developing Eclipse RCP plug-ins for interacting with remote developer desktops as well as remote forge services.

- Several open-source or proprietary applications:

  - DBin[26], a "general purpose Semantic Web application that enables power users to create discussion groups where users annotate any subject of interest and share data in P2P". The DBin project implemented a way to assemble complex applications based on reusable and configurable components, called "brainlets", described by Tummarello et al. as follow in [12]:

    "Brainlets can be thought of as "configuration packages" preparing DBin to operate on a specific domain (e.g. Wine lovers, Italian Opera fans etc..). Given that Brainlet include customized user interface, the user might perceive them as full "domain applications run inside DBin" which can be installed as plug-ins. In short Brainlets define settings for:

    * The ontologies to be used for annotations in the domain
    * A general GUI layout; which components to visualize and how they are cascaded in terms of selection/reaction
    * Templates for domain specific "annotations", e.g., a "Movie Brainlet" might have a "review" template that users fill.
    * Templates for readily available "pre cooked" domain queries.
    * Templates for wizards which guide the user when inserting new domain elements (to avoid duplicated URIs etc)
    * A suggested trust model and information filtering rules for the domain. e.g. Public keys of well known "founding members" or authorities,
    * Basic RDF knowledge package for the domain

    Creating Brainlets [...] is just a matter of knowledge engineering (e.g. selecting the appropriate Ontologies) and editing of XML configuration files."

    Future versions of the Nepomuk Eclipse RCP demonstrator will draw inspiration from the principles of DBIN Brainlets for configuring the inner components of the application and for customizing their behaviour;

  - Haystack[27], an MIT iniatitve aiming at a "universal information client". Haystack lets users "define their most effective arrangements and connections between views of information." Haystack introduced the Adenine language, a domain-specific language that includes "native support for RDF data types and makes it easy to interact with RDF containers and services"[7].

  - SWeDE[28]: SWeDE stands for "Semantic Web Development Environment" and is described as "a toolkit built on the Eclipse IDE that includes an OWL editor with helpful features like syntax highlighting, autocompletion, and error-detection and that integrates existing tools like the OWL Validator, Kazuki (OWL to Java code generator), and DumpOnt (Ontology Visualizer). Going forward, SWeDE will integrate more visualization and editing tools for Semantic Web

[25] http://www.qualipso.org
[26] http://www.dbin.org
[27] http://groups.csail.mit.edu/haystack/
[28] http://owl-eclipse.projects.semwebcentral.org

documents and rules as well as provide extension points for translation tools, lightweight visual editors, and RDF models." SWeDE is hosted under the SemWebCentral forge[29], a site that comprises several open source tools for the Semantic Web and that is managed by BBN technologies[30];

- TopQuadrant[31], the company editing TopBraid Suite™, that encompasses a spectrum of Semantic Web tools;
- Collaber[32]: as stated on Collaber web site, "Collaber brings a Virtual Office Environment for your team to share files, events, tasks, manage projects [...] as if they are all located in the same location, no matter where the members are physically located.";
- Relations-RCP[33]: the Relations application is a personal wiki allowing to "create items of three types (terms, texts and persons) and set them in relation to other items already entered";
- Eclipse Resource Tagger Plug-in[34]: this plug-in "adds tagging functionality to Eclipse such that resources may be tagged similar to the content tagging in Del.icio.us and Gmail. The tagged resources may then be managed, searched, filtered, or otherwise operated on by their tag associations."

The most significant Eclipse projects among the ones mentioned above are classified below along three following dimensions in figure 6: collaboration dimension, semantic capability dimension and ease of integration dimension. Nepomuk-Eclipse aims at being the most advanced initiative along the three axis, building whenever possible upon existing infrastructure.

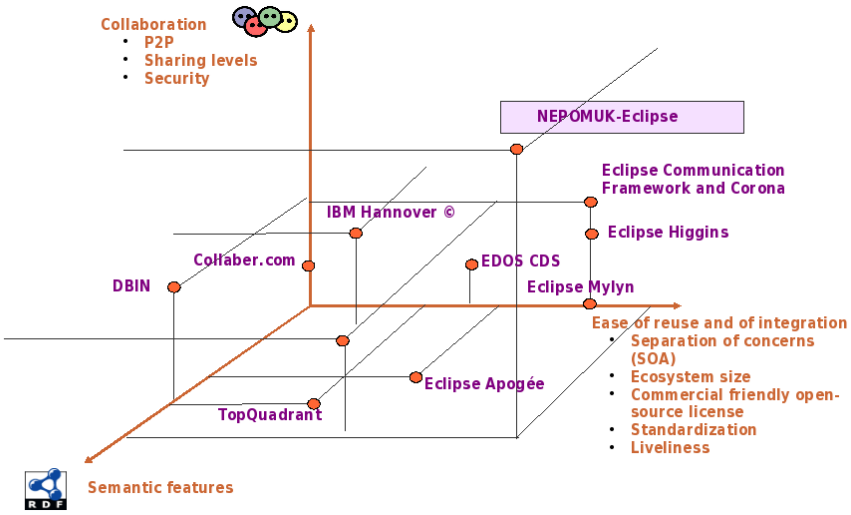

Figure 6: Classification of Eclipse projects along the collaboration, semantic and integration axis

The table below presents a Strength / Weakness / Opportunity / Threat analysis of these competing projects.

[29] http://www.semwebcentral.org/
[30] http://www.bbn.com/
[31] http://www.topquadrant.com
[32] http://www.collaber.com
[33] http://relations-rcp.sourceforge.net
[34] http://taggerplugin.sourceforge.net/

Project	Strength	Weakness	Opportunity	Threat
DBIN	P2P semantic desktop application. Includes RDFGrowth, a P2P algorithm for "collecting RDF data coming from other DBIN users" [12].	Small ecosystem. GPL, i.e. non commercial friendly. Monolithic. Limited desktop integration.	Potential for a common Nepomuk-Eclipse community. Possibility to draw inspiration from the DBIN brainlet principles for defining the views of PSEW.	
Apogée	Project endorsed by the Eclipse Foundation. Large ecosystem. Eclipse Public License. Service oriented architecture.	Poor desktop integration. Limited semantic features. No P2P features yet. Not technology neutral.	Potential for a common effort on specific services such as RDF repository, local search, resource annotation. Nepomuk standards adoption. Community building. Integration into the industry.	Single company.
ECF	Encapsulate various protocols for desktop to desktop communication and file transfer. Endorsed by Eclipse. Eclipse Public License.	No semantic features.	Use ECF for desktop to desktop file transfers and communication from Eclipse.	
EDOS-CDS	Eclipse front-end for semantic P2P indexing using KadoP "Knowledge and Data over P2P network", INRIA. Eclipse Public License.	At research stage.	Consider reusing the front-end for P-Grid. Benchmark comparison.	
Mylyn	Activity context framework.	Currently only used for code development oriented tasks.	Could be used by Nepomuk WP3000 for providing an RCP front-end to the Nepomuk task model.	

2.4 Graphical frameworks for RDF

A core benefit of the RCP platform will be to ease the development of specific applications and GUIs. As part of Nepomuk is about RDF-based annotation and browsing of desktop resources, a set of building blocks mainly oriented towards RDF editing is needed. In this context, the existing frameworks have been studied, and a new one named GnoGno has been developed (GnoGno is presented in the Nepomuk RCP demonstrator section 4).

Data intensive applications such as the Nepomuk Eclipse RCP demonstrator require a sound framework for editing various types of resources. Several existing frameworks such as Visual Basic™, Delphi™, Microsoft™ .NET™ visual editor exist for dealing with forms. All these popular and major Rapid Application Development (RAD) systems rely on visual components that are connected to a specific field of a database source, typically a relational or an object database. The specificity of dealing with RDF data is that RDF is very flexible in nature and RDF data is subject to evolution. This means the user interface should ideally evolve smoothly along the changes brought to the underlying semantic graph.

This section presents the frameworks that have been considered for describing the editors to be used to manipulate data through the Nepomuk Eclipse RCP demonstrator .

2.4.1 Haystack Adenine

The Adenine programming language was introduced in the frame of the Haystack project[5] to describe user interfaces and actions dynamically in RDF. Due to the dynamicity of RDF data, it is clear that using an interpreted language to bind RDF data to a user of interface is a powerful solution.

In Haystack, the data formats and the user interface are described using RDF, while the actions to be taken when manipulating collection of data items (addition, removal etc.) are described using Adenine.

The Adenine approach reduces the coding effort vastly, since Adenine-defined tasks can be reused easily across the application. Also, the Adenine descriptions can virtually be interpreted to generate a totally different user interface than the Eclipse RCP one, such as an HTML interface.

On the negative side however, the use of Adenine is seriously hindered by the problems arising when debugging it, and by the very high hardware requirements needed to run the Haystack software smoothly (1GB RAM and 2.0GHz CPU being not enough).

2.4.2 Fresnel

The experience from Haystack Adenine was used to generate a more generic standard: the Fresnel language[15]. Fresnel is an RDF vocabulary that allows ontology engineers to specify how resources of an ontology should be rendered in a user interface. The vocabulary is published on the W3C portal and has the role of a quasi-standard in the domain of visualizing RDF [6].

The main concepts of Fresnel consists of the following:

- A Fresnel Lens defines a view on data, content selection and ordering.
- A Fresnel Format defines how the content should be formatted and if additional data should be displayed.
- A Cascading Style Sheet (CSS) contains styling instructions for fonts, colours, and borders.

Here is an example of a Lens and a Format (example extracted from [6]):

```
:PersonLensafresnel:Lens;
  fresnel:classLensDomainfoaf:Person;
  fresnel:showProperties(
    foaf:name
    foaf:mbox
    [rdf:typefresnel:PropertyDescription;
     fresnel:alternateProperties(
     foaf:depictionfoaf:imgp3p:image)
     ] ) .
:mboxFormatafresnel:Format;
  fresnel:propertyFormatDomainfoaf:mbox;
  fresnel:label"Mailbox";
  fresnel:valuefresnel:externalLink;
  fresnel:valueFormat[fresnel:contentAfter",""].
```

An implementation can then interpret the RDF data using the Fresnel description and render the data. Figure 7 shows a rendering.

As Fresnel is defined in RDF, it is extensible, making it is possible to define custom formats and lenses. Fresnel was used successfully in Nepomuk for cre-

Figure 7: A Fresnel lens interpreted to render information about a contact person

ating both HTML and SWT editors for configuring the Aperture data sources. New elements were even added to the framework on that occasion, to allow the editing of passwords in Fresnel, and to add support for checkbox widgets.

Several Fresnel implementations are available, making it possible to reuse UI Fresnel descriptions in Web and desktop applications, without any assumption on the programming language of the target application. This is clearly an advantage in the context of Nepomuk, where several technologies are handled, and severl types of user interfaces are targeted.

On the negative side, Fresnel lacks clear concepts on how GUIs for editing data can be generated using Frensel. A combination of the GnoGno framework and of Fresnel may be considered for flexible form generators with fine-grained rendering.

3 Nepomuk Eclipse community involvement

Beside the implementation of an Eclipse demonstrator show-casing an implementation of the Nepomuk platform, one of the main tasks of the Nepomuk-Eclipse project is to involve Eclipse RCP developers into the Nepomuk project. This section gives an overview of the Eclipse community activities in the Nepomuk-Eclipse project.

A communication platform has been set up, face-to-face meetings have been organized, and fruitful collaboration with two other activities have taken place, as described in the following paragraphs.

3.1 The Nepomuk-Eclipse web portal

In order to present the advancement of the project, and also to raise interest across the community, a dedicated Web site has been created at the following location: `http://nepomuk-eclipse.semanticdesktop.org`.

Figure 8 shows the home page of the Nepomuk-Eclipse Web site.

Figure 8: Nepomuk-Eclipse web site

The Web site consists of a wiki that registered members can edit. This wiki is intended to be a source of information for developers and users of the Nepomuk-Eclipse project. It is open to the public, inviting the community to take part in the initiative.

The Web site visits are measured using the online Google Analytics[35] service. The table below reports the main figures pertaining to the site traffic. It is expected that these figures will grow significantly by the end of 2007 with the announcement of the availability of the first Eclipse RCP based Nepomuk demonstrators.

[35] `http://www.google.com/analytics/`

Indicator	Value
Average number of visits per month	103
Average number of page views per month	300
Average number of page views per visit	3
Total number of wiki pages	60
Total number of pages tagged "Eclipse" on the SemanticDesktop.org Web site (`http://www.semanticdesktop.org/xwiki/bin/Wiki/Eclipse`)	28

Figure 9 shows the Google Analytics dashboard of the Nepomuk-Eclipse Web site for the month of July 2007.

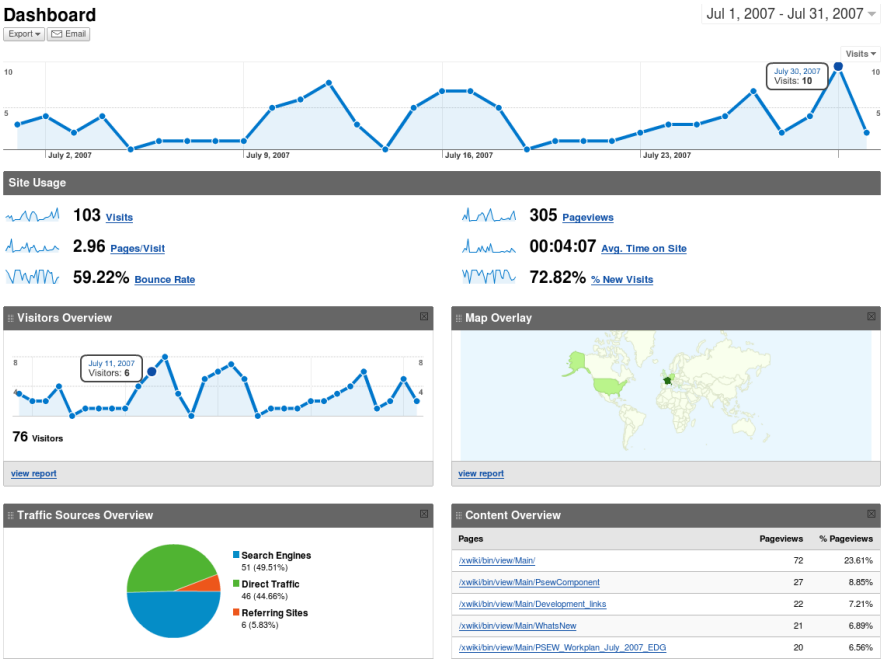

Figure 9: Nepomuk-Eclipse Web site visit analysis – July 2007

In addition to the Nepomuk-Eclipse Web site, a public mailing-list has been set up and has already received attention from several external users or developers. The mailing-list is located at the following URL: `https://nepomuk.semanticdesktop.org/wws/info/nepomuk-eclipse`. Anyone can subscribe to the list either using the web interface, or by sending an email to the SYMPA mailing-list manager at `mailto:sympa@semanticdestkop.org`, with the subject "subscribe nepomuk-eclipse". Beside the Nepomuk internal participants, 10 persons have subscribed to the list, expressing their interest for the initiative either directly via the list or through private emails to the list owners. The number of participants is expected to grow significantly by the end of 2007 following several announcements that will be made on relevant mailing-lists. The creation of a permanent IRC or Jabber channel dedicated to the Nepomuk-Eclipse sub-project will be considered.

3.2 Nepomuk-Eclipse IBM workshop

An Eclipse RCP workshop has been organized by the Nepomuk IBM team in the Dublin IBM Center for Advanced Studies[36], with the goal to provide training on the Eclipse RCP platform and its opportunities for the integration of several Nepomuk components. The first day consisted of a presentation of

[36] `http://www.ibm.com/ibm/cas/sites/dublin/`

Eclipse RCP, with more specific discussion around how RCP can be exploited in the Semantic Web paradigm, and a possible Eclipse reference implementation for Nepomuk. In addition to several Eclipse IBM experts, an external Eclipse speaker was present. The second day consisted of a hands-on workshop for giving attendees a basic understanding of how to develop an RCP application.

Following the Eclipse workshop at IBM Dublin, DFKI, DERI and Cognium Systems worked with IBM LanguageWare[3] labs to design a demonstrator for Lotusphere 2007 (January 21-25, Orlando, Florida), showing integration of the new Lotus Notes client with Semantic Desktop capabilities harnessing the Nepomuk platform. The demonstration was presented to IBM customers, partners and press.

3.3 Semantic Desktop Hands-on Session 2007, Berlin

The Nepomuk-Eclipse project was presented at the second Berlin Semantic Desktop hands-on session[37] with an audience of approximately 50 people. Discussions with the authors of DBin have taken place on the topic of possible future contributions of DBin maintainers to the Nepomuk-Eclipse implementation.

3.4 Synergy with the Hyena project

DFKI has entered into discussions with Axel Rauschmayer, author of the Hyena project [38], Hypergraph Editor and Navigator. Hyena is described as "an RDF editor that allows one to mix and match RDF vocabularies to flexibly model any kind of data.". Hyena can be run as either an Eclipse plugin or an Ajax-based web application.

3.5 Nepomuk-XWiki RCP workspace engine client

XPertNet[39] has been involved in the Nepomuk project as a subcontractor of Edge-IT since January 2007. XPertNet is the editor of the open-source software XWiki[40], a framework supporting the easy creation of collaborative web applications harnessing the wiki principles.

XWiki provides typical workspace services supporting workgroup collaborations: XWiki lets users form groups for sharing documents within dedicated workspaces. In 2007, the development of an Eclipse RCP client for interacting with an XWiki workspace has started. The figure 11 shows a screenshot of this RCP client as of July 2007.

The principles of the Nepomuk platform have been advertised to the XWiki community of developers, with the objective to use the Nepomuk OSGi components for introducing semantic capabilities and advanced community features to the XWiki engine. A prototype of XWiki embedding the Nepomuk WikiModel component[41] has been developed. Nepomuk-Eclipse team will consider building upon the XWiki RCP plugin for introducing the wiki metaphor and more generally the WP1000 innovative concepts[4] to the Eclipse RCP desktop as extensions to the PSEW application.

[37] http://www.semanticdesktop.org/xwiki/bin/Wiki/SemDeskHandsOn2007April/
[38] http://hypergraphs.de/
[39] http://www.xwiki.com
[40] http://www.xwiki.org
[41] http://nepomuk.semanticdesktop.org/xwiki/bin/Components/WikiModel

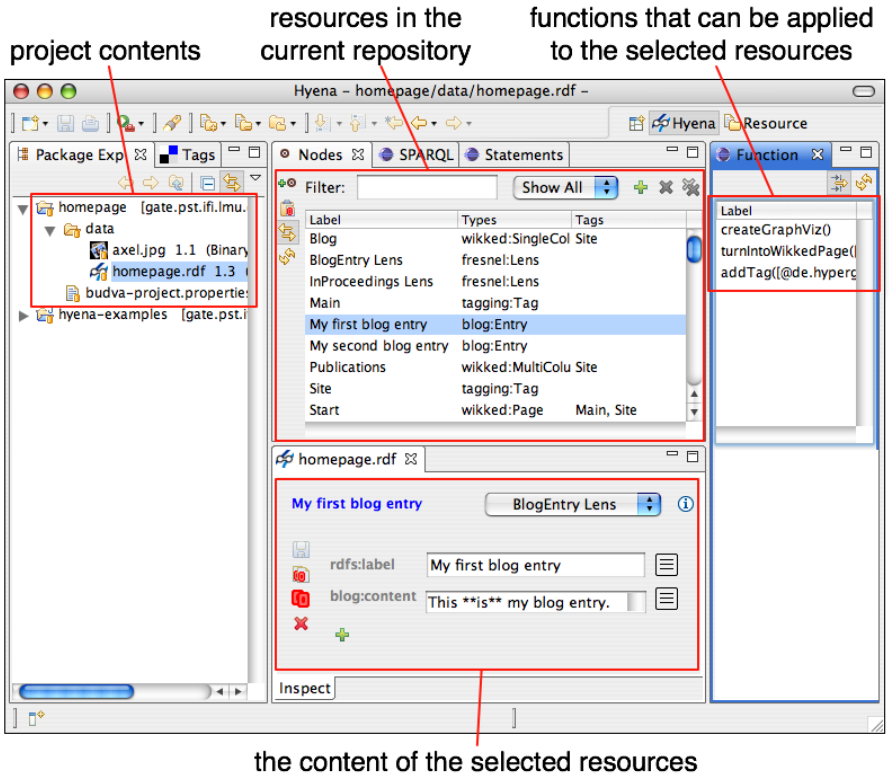

Figure 10: Hyena screen example

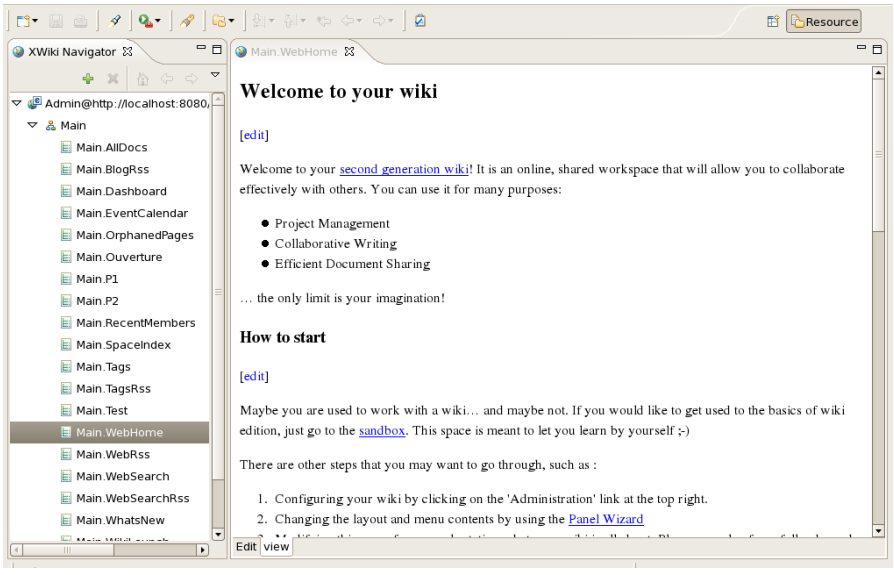

Figure 11: XWiki Eclipse RCP

4 Nepomuk-Eclipse prototype: PSEW

4.1 Introduction

To make the Nepomuk ideas graspable in the context of Eclipse RCP, a prototype was needed. The novel Nepomuk services also need to be available in Eclipse. The generated prototype is described in the following section. Based on this prototype, the related communities can see how to embed and access Nepomuk in their environments.

4.2 Objectives

Several workbenches for managing semantic personal information have been burgeoning since the foundations of the Semantic Web have been laid down. The paper "Overview and Outlook on the Semantic Desktop" presents the principal aspects of Semantic Desktop user interfaces [9].

Typical semantic workbenches have an architecture that can be roughly illustrated by the figure 12: the RDF data model is applied to data originating from a local desktop or from Internet data sources. The workbench then provides at least the following services: resource browsing, resource annotation using a wiki or a form-based editor, resource search using semantic criteria, resource annotation availability across all the desktop applications.

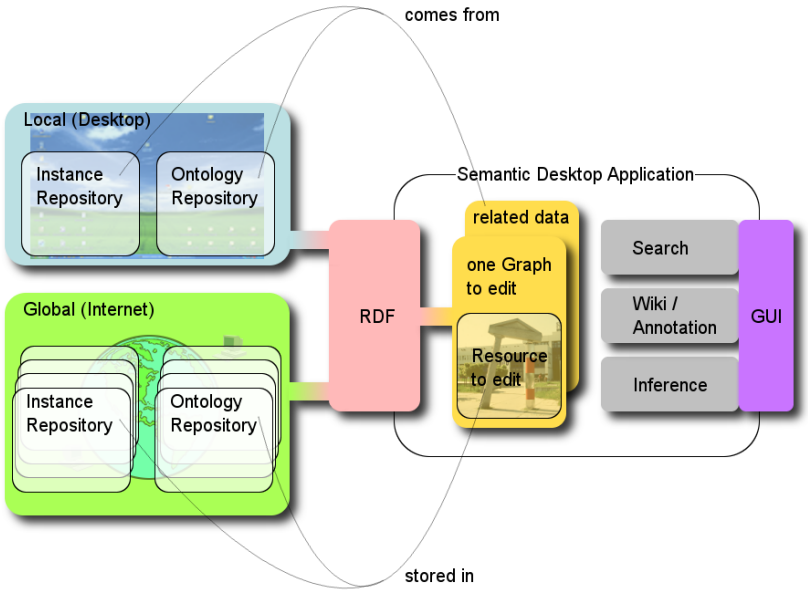

Figure 12: A typical Semantic Desktop application architecture

A user interface for accessing the services provided by such a workbench typically looks like the one shown in figure 13: an address bar allows the user to navigate to resources directly, a search box allows full-text search, a navigation tree lets the user browse the available resources. Once a resource is selected, it is shown centrally, letting the user browse and add annotations, as well as other contextual information.

This UI abstraction is the common base of many existing desktop Semantic Web applications such as the Protégé ontology editor, DBIN, Haystack, OpenIris, Gnowsis, but also the Microsoft™ Windows™ file explorer and Microsoft™ Outlook.

The design of the first version of the PSEW prototype was inspired by this type

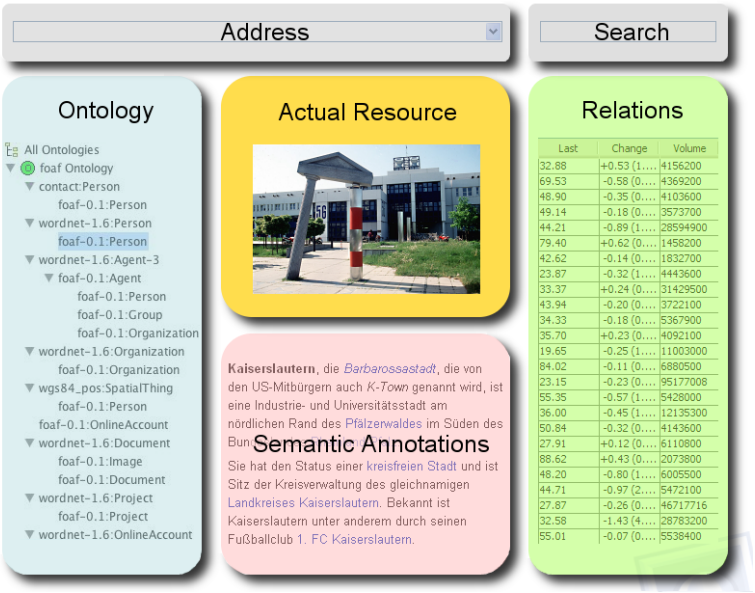

Figure 13: A typical Semantic Desktop application user interface

of user interface.

4.3 Nepomuk core components

The PSEW prototype is built on top of the infrastructure provided by the following Nepomuk components:

- RDF2Go[42]
  RDF data is handled in PSEW using the RDF2Go API[43], which provides the following services:
  - Represents the basic concepts of RDF as objects: Nodes, Resources, Literals, Statements,
  - Stores and loads data from databases,
  - Stores and loads data from serialized forms,
  - Run SPARQL queries,
  - Inference.

  RDF2Go is a Java RDF API that abstracts from all the APIs and offers features common to all of them. It does not implement the functionality itself, instead it forwards to existing APIs. RDF2Go can be seen as a JDBC API for RDF databases.

  The existing approaches use different and incompatbible RDF frameworks (DBIN uses Sesame v1, Hyena uses a custom made RDF framework, Boca is an extension of Jena), which causes a problem when integrating them. For Nepomuk, the use of the RDF2Go abstraction framework was chosen, that can wrap the underlying RDF layer.

- RDFRepository component[44]
  The Nepomuk RDF repository [45] is the central metadata store in Nepomuk. It contains all information from the file system from the DataWrapper, ontologies, and the Personal Information Model (PIMO) of the user.

[42] `org.ontoware.rdf2go`
[43] `http://rdf2go.semweb4j.org/`
[44] `org.semanticdesktop.nepomuk.comp.rdfrepository`
[45] `https://dev.nepomuk.semanticdesktop.org/wiki/RdfRepository`

All data handled locally by a Nepomuk Semantic Desktop is stored in this repository. As of July 2007, storage of binary files is not yet part of the repository. The component uses the Sesame[1] backend.

The Nepomuk RDF repository component provides the following features:

- Storing and querying RDF Data,
- SPARQL support,
- Inference engine for subclass/subpropery/inverseProperty.

The following features will be available in the next version of the component:

- full-text search using LuceneSail,
- Full NRL inference,
- NRL Validation.

- Aperture DataWrapper component[46]
  The component integrates the handling of Aperture[10] data crawlers, the definition and management of data sources and their configuration, and the management and synchronization of the crawling processes that perform the extraction of the meta-data from the defined data sources.

The previously mentioned components are built on the top of the OSGi[2] component model, which eases their integration in the Eclipse RCP framework. Their functionalities are made available in the form of OSGi services that, once registered, can be acquired by other components and, in our case, by the PSEW.

4.4 PSEW description

This section describes the initial implementation of the PSEW prototype, starting from the functionalities and a general architectural view, down to some implementation choices and techniques.

The actual prototype consists of of different components:

- `org.semanticdesktop.nepomuk.psew`: this component provides an RCP shell where all the other components will be hosted. Moreover, the `org.semanticdesktop.nepomuk.psew` component contains all the configuration parameter for packaging the PSEW as a standalone application;
- `org.semanticdesktop.nepomuk.psew.datawrapper.ui`: this component contains all the graphical user interface elements and the logic for managing the Aperture DataWrapper component;
- `org.semanticdesktop.nepomuk.psew.explorer`: this component contains all the graphical user interface elements for navigating and searching the RDF repository containing the meta-data extracted by the Aperture DataWrapper component.

Both the `datawrapper.ui` and `explorer` components are actually independent of the RCP shell and they can be deployed in other environments as well. For example, since they use declarative mechanisms for providing additions to the Eclipse platform, they could be integrated directly in any Eclipse product by simply installing the component JAR archives in the plugin repository. In

[46] `org.semanticdesktop.nepomuk.comp.datawrapper.aperture`

this way, these components can be used and reused in different contexts, depending on the user's needs.

The following sections detail the characteristics of the `datawrapper.ui` and `explorer` components, and present the PSEW interface, shown in figure 14.

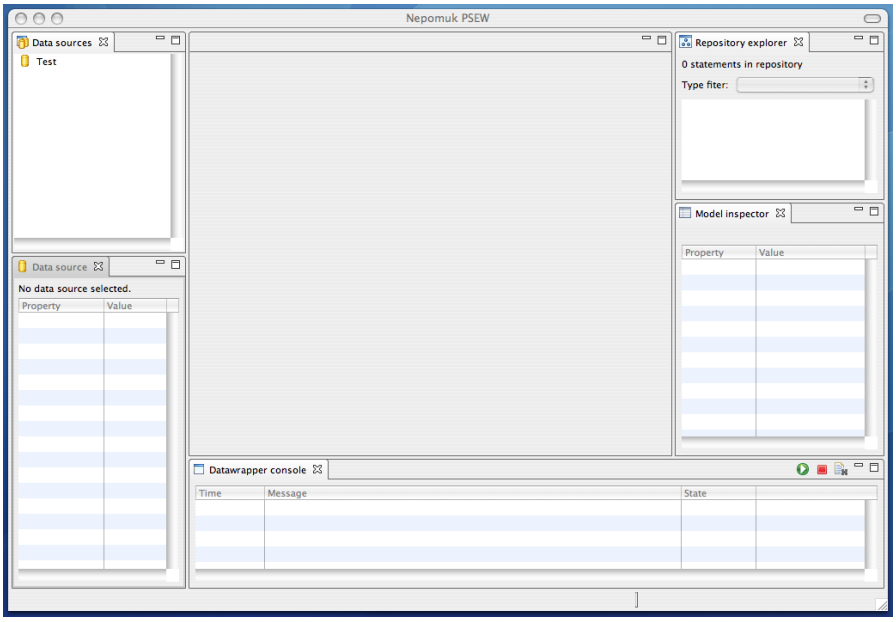

Figure 14: The main PSEW window

4.5 The `datawrapper.ui` component

The `datawrapper.ui` component is the bridge that links the PSEW interface to the Nepomuk Aperture DataWrapper component. It provides the following elements:

- A data sources view that shows all the configured data sources that will be crawled, and that will provide the meta-data the end user could navigate.
- A data source information view that shows, for each data source its current configuration parameters.
- A datawrapper console that is used to control the Nepomuk Aperture DataWrapper component by starting and stopping the crawling processes, and to receive and show feedback from that component about its current state and the operations that are being carried on.
- A preferences editor that is used to edit and change the configuration of the available data sources.
- A set of wizards that guides the user in the creation of new data sources.

Figure 14 shows the data sources, data source information and DataWrapper console views in the left and bottom part of the interface.

Figure 15 shows the class hierarchy of the `datawrapper.ui` component. It is worth noticing the content of the `util` package and the `DataSourceForm` class. All these classes provides a generic infrastructure for automatically generating the forms that are needed for exposing to the end user, all the characteristics of a given data source. This form generation is based on the Fresnel framework introduced earlier in the section 2.

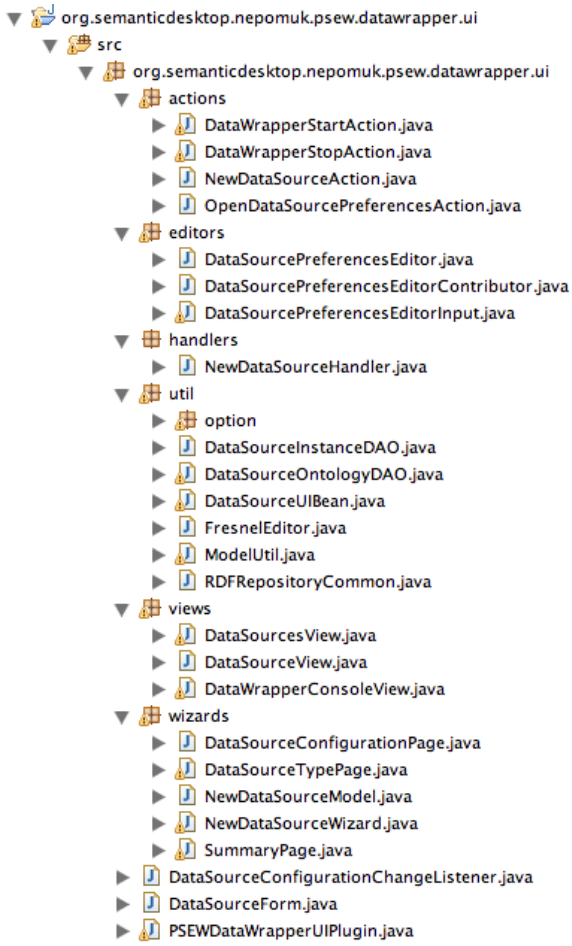

Figure 15: The `datawrapper.ui` component's class hierarchy

By querying the underlying Fresnel models that are associated to a given data source type, the `FresnelEditor` class constructs a set of objects that contains the structural properties of a data source. The `DataSourceForm`, starting from this list, is able to build the actual user interface that exposes all the data source properties to the user by using widgets arranged in a form (Figure 16).

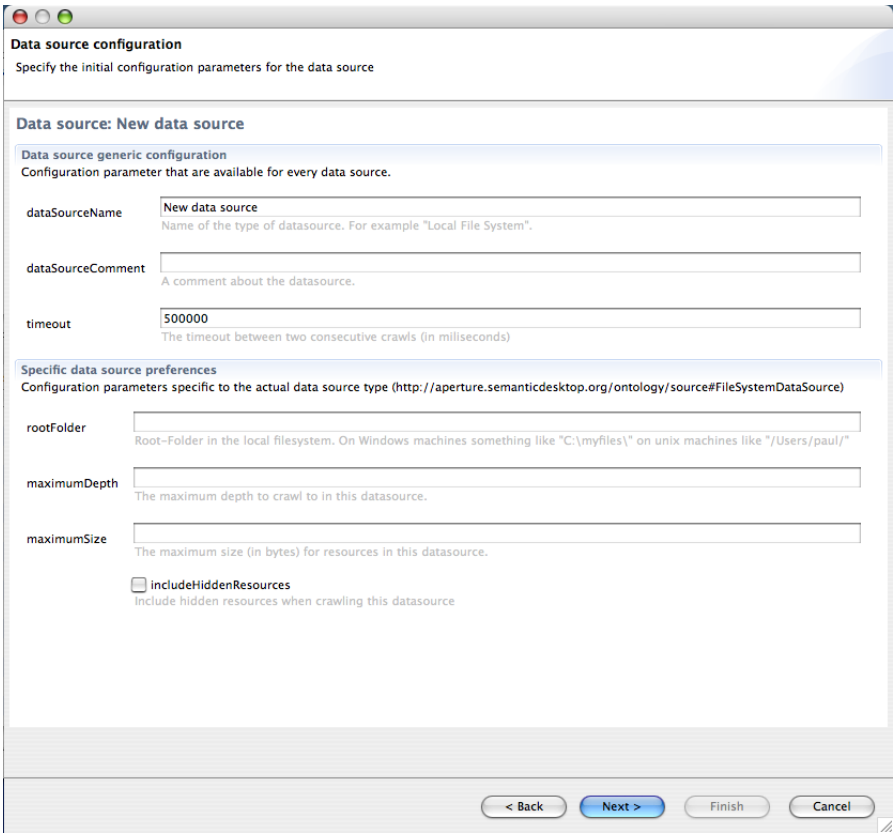

Figure 16: New data source wizard: data source configuration

The information provided to the user contains the correct rendering of a widget depending on the property type (text fields, option, etc.), a description of the property and a mechanism for validating the user input, e.g., preventing the user to insert a string where an integer is required.

The same infrastructure is used both in the wizard for creating new data sources, and in the editor for modifying already available data sources (figure 17)

This mechanism is totally generic with respect to the available data sources and can be used to provide a configuration interface for new data source types that will be added in the future, provided that these data sources are accompanied by the relative Fresnel description. Figure 18 shows the currently available data source types in the Aperture framework, as provided by the Aperture DataWrapper component.

4.6 The `explorer` component

The `explorer` component provides the user interface and the functionalities for navigating the RDF store where the crawled metadata is stored, handled by the Nepomuk RDFRepository component. It provides the following elements:

- A search editor for performing simple keyword search on the whole crawled meta-data,

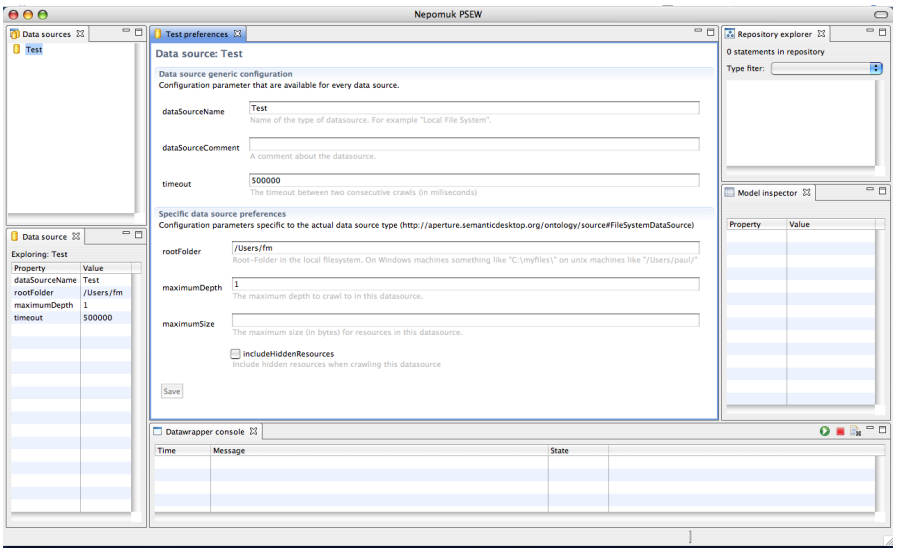

Figure 17: Data source preferences

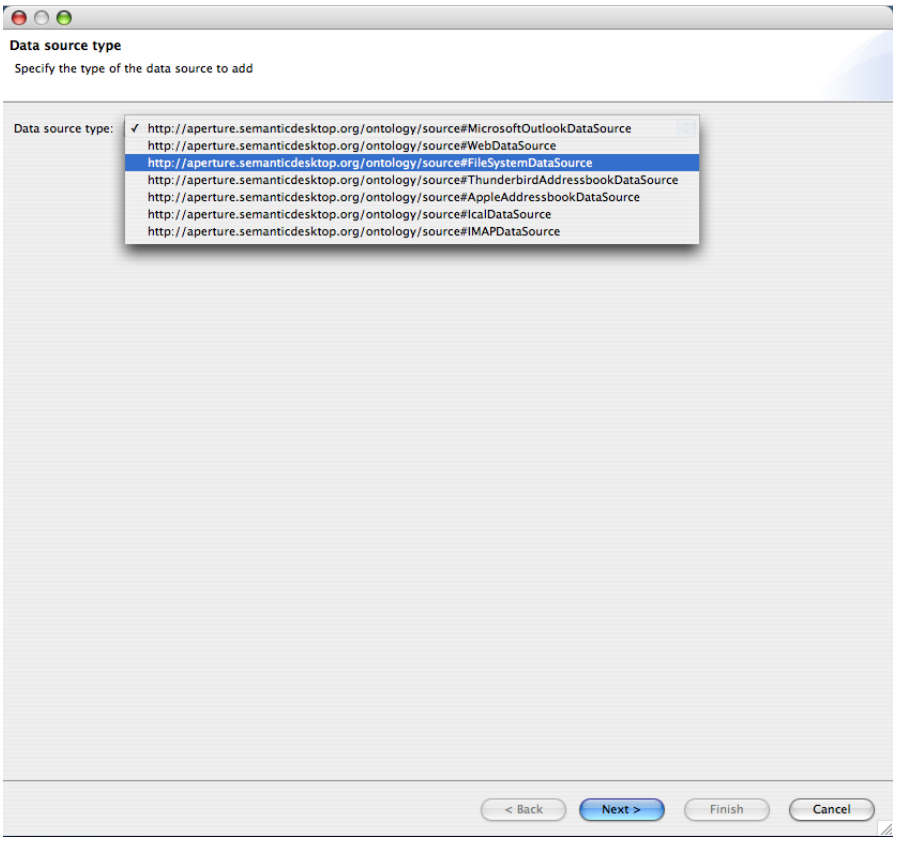

Figure 18: New data source wizard: data source type choice

- A repository explorer view that shows all the available RDF models present in the repository and that correspond to the crawled resources,
- A model inspector view that provides information about the content of a given RDF model.

Figure 14 shows the repository explorer and the model inspector views in the right part of the interface, while figure 19 shows the search editor.

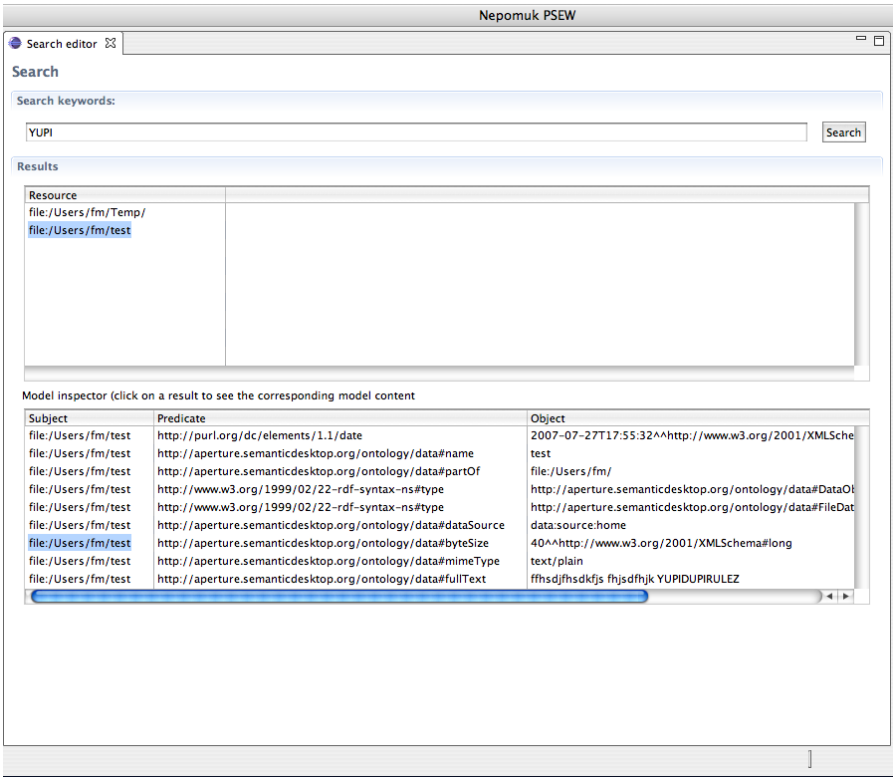

Figure 19: Meta-data keyword search

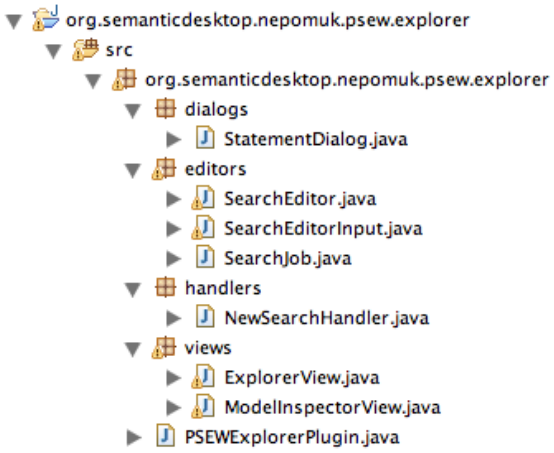

Figure 20: The `explorer` component's class hierarchy

Figure 20 shows the class hierarchy of the `explorer` component. The explorer view can perform some declarative filtering that is dynamic on the type of the resources that are present in the RDF repository. Currently it is possible to filter the meta-data that is displayed in the explorer view by their type. In the next version of PSEW, more advanced filtering mechanism based on the attributes provided by the underlying ontologies will be used.

The same remark applies to the search editor, that at present uses a simple pattern matching on strings in order to discover which meta-data contains the requested keyword. A more elaborated ontology-based searching mechanism will be provided in the following versions.

Figure 21 illustrates how an annotated resource is rendered using the GnoGno framework described below.

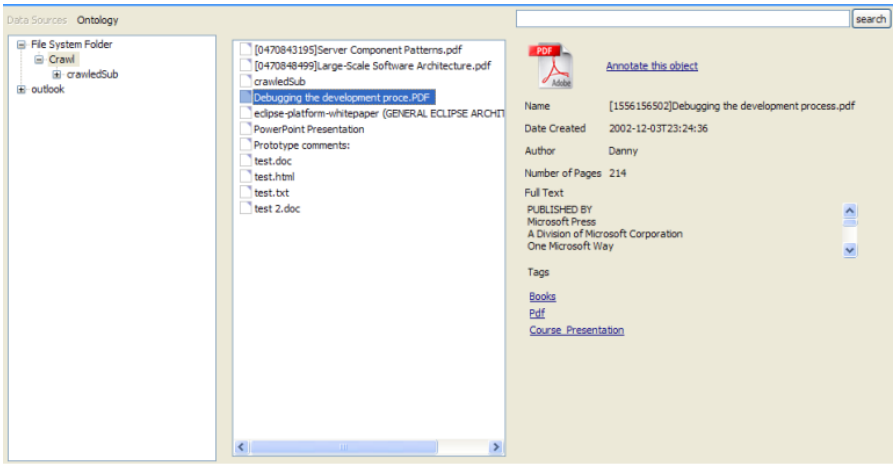

Figure 21: Example of an annotated resource

4.7 The PIMO browser component

The Nepomuk Personal Information Model (PIMO) consists of an ontology of common concepts related to personal information management (People, Topics, Places, ...) and of a framework to represent and to use those concepts in the context of the other Nepomuk ontologies. An example ontology is shown in figure 22.

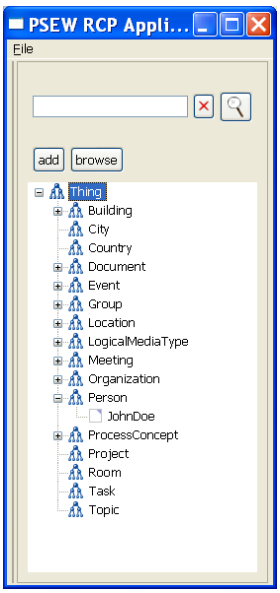

Figure 22: Example ontology browser

4.8 GnoGno framework

As mentioned in section 2, since existing frameworks for handling graphically RDF resources were not satisfactory, a new one, named GnoGno[8], has been designed by the Nepomuk Consortium.

GnoGno consists of a set of components to easily create Swing or SWT/JFace user interfaces that are backed by RDF2Go objects allowing to use any RDF framework such as Jena or Sesame.

The GnoGno GUI framework was initiated by Leo Sauermann in the frame of the Gnowsis application design, and was re-architectured and re-implemented in the context of the Nepomuk project.

GnoGno provides data binders that connect GUI widgets directly to RDF data models. For example, a text editor widget can be bound to a certain property of a resource, the widget will then display the object value of this resource and change the value in the data model if the user edits the value in the widget.

Additionally to the visual components, a layer between GUI and RDF data model is added. An RDF model is wrapped using a ModelDataSet, a selected resource from this model using a ResourceDataSet. The RDF model is based on RDF2Go[14], which allows GnoGno to be used with any underlying RDF store. The GUI widgets are then connected to the ModelDataSet and ResourceDataSet, not directly to the RDF data. The widgets update the visual representation when a new model is loaded or when a resource from this model is selected, so a user interface can, once created, be used to show data of different instances. GnoGno supports both Swing and Eclipse widgets.

GnoGno also allows the generation of form-based editors: edit controls for properties of a resource can be generated based on the ontology used. The figure 23 illustrates a form for editing metadata about a project.

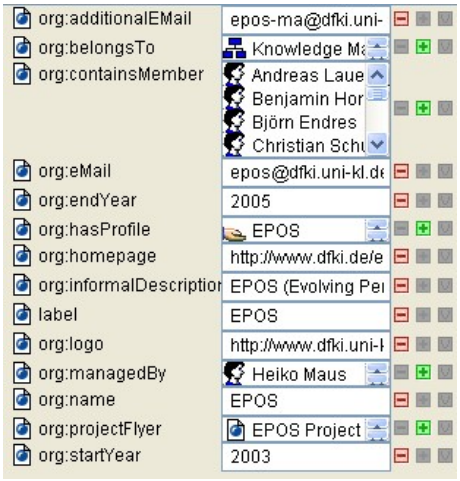

Figure 23: GnoGno Thing Dialog

More information about GnoGno can be found on the project home page: `http://gnowsis.opendfki.de/wiki/GnognoComp`.

A screencast showing how to use the GnoGno components is available from the following page: `http://www.dfki.uni-kl.de/~sauermann/2007/05/31/edited_web.html`.

Further improvements are under work, to make the GnoGno components of interest to a broader community.

4.9 Next steps

The next steps related to the development of PSEW will mainly consist of bringing the following enhancements:

- Advanced annotation capabilities,
- Addition of P2P metadata indexing capabilities by using the Nepomuk distributed index component[13],
- Design of a rich semantic wiki editor on top of the Eclipse text framework,
- Introduction of i-mapping features,
- Inclusion of the WP3000 personal task manager assistant,
- User context service on top of Mylyn,
- Integration of the WP5000 desktop ranker component.

5 Conclusion

During the first 18 months of the project, sound foundations have been laid down for the involvement of the Eclipse community into the Nepomuk project: the analysis of the existing community has led to a better understanding of the ongoing projects, the Nepomuk-Eclipse community platform has raised awareness on the initiative, and the first version of the PSEW demonstrator has been made ready for public release. It is expected that the second half of the Nepomuk project will see the Nepomuk-Eclipse community grow significantly and emulate constructively with the Nepomuk-KDE community.

The first version of the PSEW demonstrator showcases some of the capabilities of the Nepomuk platform. The availability of similar features in the two frameworks KDE and Eclipse RCP paves the way for interesting comparisons indeed, all the more as Eclipse RCP is progressively evolving toward a full-fledged desktop window system. It is expected that cross-fertilization between the communities will take place, hence bolstering the adoption and the further enhancement of the Nepomuk standards.

The PSEW prototype will be used and extended by several Nepomuk case studies, serving as a foundation for an Eclipse based user interface to the Nepomuk Social Semantic Desktop.

In parallel to the further development of the PSEW prototype, the focus will be brought on the involvement of more Eclipse developers into the Nepomuk-Eclipse project, by presenting the project to a wider audience on the occasion of Eclipse related events, and by increasing the presence of the project on the Web.

References

[1] Aduna. Sesame. http://www.openrdf.org/.

[2] OSGi Alliance. Osgi. http://www.osgi.org.

[3] IBM. Languageware. http://www-306.ibm.com/software/globalization/topics/languageware/index.jsp.

[4] Mikhail Kotelnikov, Alexander Polonsky, Malte Kiesel, Max Volkel, Heiko Haller, Mikhail Sogrin, Par Lannero, and Brian Davis. Nepomuk deliverable d1.1: Interactive semantic wikis. Technical report, Cognium Systems, DFKI, FZI, IBM, KTH, NUIG, 2006.

[5] MIT. Haystack. http://groups.csail.mit.edu/haystack/.

[6] Emmanuel Pietriga, Christian Bizer, David Karger, and Ryan Lee. Fresnel: A browser-independent presentation vocabulary for rdf. In Isabel F. Cruz, Stefan Decker, Dean Allemang, Chris Preist, Daniel Schwabe, Peter Mika, Michael Uschold, and Lora Aroyo, editors, International Semantic Web Conference, volume 4273 of Lecture Notes in Computer Science, pages 158–171. Springer, 2006.

[7] Dennis Quan. Metadata programming in adenine. http://groups.csail.mit.edu/haystack/documentation/adenine.pdf.

[8] Leo Sauermann. Gnogno gui framework. http://gnowsis.opendfki.de/wiki/GnognoComp.

[9] Leo Sauermann, Ansgar Bernardi, and Andreas Dengel. Overview and outlook on the semantic desktop. In Dennis Quan Stefan Decker, Jack Park and Leo Sauermann, editors, Proceedings of the 1st Workshop on The Semantic Desktop at the ISWC 2005 Conference, 2005.

[10] Aperture Team. Aperture, a java framework for getting data and metadata. http://aperture.sourceforge.net/.

[11] Eclipse RCP team. Eclipse rcp. http://eclipse.org/rcp/.

[12] Giovanni Tummarello, Christian Morbidoni, and Michele Nucci. Enabling semantic web communities with dbin: An overview. In Proceedings of the ISWC, pages 943–950, 2006.

[13] Renault John Vasilios Darlagiannis, Roman Schmidt and Ekaterini Ioannou. Nepomuk deliverable d4.1 - distributed search system - basic infrastructure. Technical report, EPFL and L3S, 2006.

[14] Max Völkel. Rdf2go. http://rdf2go.semweb4j.org/.

[15] W3C. Fresnel - display vocabulary for rdf. http://www.w3.org/2005/04/fresnel-info/.