Talia: A Framework for Philosophy Scholars

Michele Nucci[1], Stefano David[1], Daniel Hahn[2], and Michele Barbera[2]

[1] Semedia Group - 3mediaLabs
Università Politecnica delle Marche
s.david@univpm.it, mik.nucci@gmail.com
[2] NET7
Pisa
(barbera|hahn)@netseven.it

Abstract. In this paper we present Talia, a novel implementation of a semantic digital web library system, which is part of the Discovery project. Talia deploys Semantic Web technologies and uses computational ontologies for the organisation of knowledge, which can help the definition of a state-of-the-art research and publishing environment for philosophy. Talia will provide an innovative and flexible system which enables data interoperability and new paradigms for information enrichment, data-retrieval and navigation.

1 Introduction

The human sciences have been traditionally hesitant in picking up new technologies and publication channels, instead relying on traditional printed publications. However, the search for and the acquisition of manuscripts and other source material can be difficult, expensive and time-consuming. When researchers do not have access to the original writings of their field they have to rely on secondary material, which may not be accurate and may even lead to invalid results, as describe in [1].

The Discovery Project[3] will create a distributed digital online library for the special needs of philosophers. It will contain original texts and manuscripts by Wittgenstein, Nietzsche, modern and presocratic philosophers.

In addition, the philosophers will be able to publish and peer review new material online. This gives the research community immediate access to new results and removes the burden that the traditional publishing model put on the dissemination of content.

The Discovery library will be built using the Talia system, which is being developed specifically for the Discovery project. Talia will be a distributed digital library system, consisting of a *federation* of interoperable web sites (nodes), each of which will contain content on a specific topic.

Talia uses computational ontologies to organise information and Semantic Web technology[4], like RDF(S) and triple-stores. This technology allows Talia to

[3] http://www.discovery-project.eu/

[4] See http://www.w3.org/2001/sw/ for Semantic Web initiative, related technologies and other W3C standards.

provide excellent data interoperability; it also allows to create a flexible framework for information enrichment and data retrieval that suits the specific needs of research communities. Talia will be able to use different ontologies for each node.

A flexible user interface framework will allow the content providers to configure the navigation according to their own ontologies for each node. By using the Ruby on Rails[5] framework and standard semantic technology Talia will be able to provide a novel way to quickly develop and deploy digital libraries for specialised needs.

Compared to existing digital libraries, Talia will adapt much better to changing requirements and heterogeneous metadata. Compared to other digital library systems, Talia benefits from a much better integration between the semantic technology and the user interface framework.

The paper is organised as follows. In Section 2 we briefly present the Discovery project, in Section 3 we introduce computational ontologies and in Section 4 we describe some general use-cases and requirements which have guided the Talia design and development. Finally, in Section 5 we will present the Talia architecture, focusing on its *semantic software layer*, based on Semantic Web technologies.

2 The Discovery Project

The Discovery project aims at the creation of a federation of interoperable web sites designed to aid humanities' scholars working on digital collections. The content of the Discovery federation is related to ancient and modern philosophy, but the model can easily be applied in other fields. The technical infrastructure underlying Discovery consists of two main components:

- A federation of institutional archives or content providers;
- A peer-to-peer (P2P) network of personal desktop applications;

The federated web sites (nodes) represent the foundation of the Discovery approach. These domain-specific archives and *publishers* contain the digitalised content. Each node is maintained by a different institution and acts as a content-provider and as an Open Access publisher. Each is dedicated to a specific author or theme of ancient and modern philosophy.

The sites contain both reproductions of *primary sources* and *scholarly contributions*. Primary sources are the principal subject of research by a specific community; in the Nietzsche community, for example, these are the writings of Nietzsche himself. Scholarly contributions are scientific works that contribute to the research of the subject at hand. These could be essays on the primary sources, reviews of other contributions, or other research results.

[5] `http://www.rubyonrails.org/`

As Open Access publishers, the nodes accept new secondary sources, or *contributions* related to the primary sources of their scholarly community. Contributions are peer reviewed by the board of reviewers of the node and then published on the web site, side by side with their related primary sources.

One of the characteristic traits of Discovery is that the relations between primary and secondary sources are explicitly described and expressed in a machine readable way. A "Discovery structural ontology" has been created; it will be shared among all Discovery nodes. The structural ontology is the basic building block of the Discovery network as it represents the lowest common denominator of interoperability among the archives of the participating data providers.

Knowledge in Discovery will be organised by means of suitable computational ontologies, which will be used both as a means for organising knowledge concerning each site, and to relate knowledge from different sites.

3 Computational Ontologies

Ontologies have become popular in computer science as a way of organising information. This connotation of *ontology* differs to the traditional use and meaning it has in philosophy, where ontologies are considered as "A system of categories accounting for a certain vision of the world" [2]. There have been a lot of definitions of ontology in computer science (or computational ontologies), which extended and refined the one provided by Gruber [3], who defined an ontology as "a specification of a conceptualisation."[6]. We will stick to the following one, extracted from Guarino's definitions (see for example [4]): A computational ontology is *"A Formal, Partial Specification of a Shared Conceptualisation of a world (domain)"*. Intuitively, according to this definition, a computational ontology is a set of assumptions that defines the structure of a given world or domain of interest, allowing different people to use the same concepts to describe that domain.

This definition incorporates the most relevant characteristics for an ontology to be useful:

- *Formal.* An ontology needs a robust underlying theory for reasoning.
- *Partial specification.* It has a representative aspect (i.e., it presents and organises knowledge about a domain), but shall not pretend to describe every detail of the domain it should represent.
- *Shared.* An agreement shall be reached on an ontology by all interested parties, so that they can share the same ontology, and the same concepts defined in it, without the need to reinvent the wheel.
- *Conceptualisation of a world.* It should meaningfully represent a world of interest.

[6] A more up-to-date definition, together with additional readings can be found at `http://tomgruber.org/writing/ontology-definition-2007.htm`

4 Requirements and sample use cases

While the requirements for the Talia system vary significantly between the user communities, there are some general requirements that are valid for all Discovery partners. In the following we present some of the central features of Talia. These are the ones that every typical scholar using Talia will need.

4.1 General Requirements

- The users need to be able to access the primary sources (or their reproductions) online. Previously, these were often difficult and/or expensive to obtain.
- There should be a *context* to the documents that are being viewed. This means that the researcher should be able to quickly find documents that are related to the item at hand. Related, in this case, can mean for example "is cited by", "was written in the same period", "shares a common topic". This context depends heavily on the community that uses the archive.
- The user will not only be interested in relations to items in the current archive, but in relations to items in other archives as well.
- If an author wants to cite an item from the archive in her work, she must be able to refer to this item in a unique way. She also needs to have the confidence that the archive is *permanent* – so that she can cite a document without fear that it may be renamed, removed, or altered in the future.
- Researchers will be able to publish new content on the system. In order to be a viable alternative to traditional publication channels, the system must ensure at least the same level of quality. Thus, a peer review workflow has to be established that ensures the same standards as in printed publications.

4.2 Example Use Cases

One of the most striking requirements for Talia is that it needs to provide a unified platform for user communities with very diverse needs. Therefore, we will illustrate three simple use cases for three diverse user communities[7]. From these use cases, we can then explore the technical decisions made in the design of Talia.

"The Manuscript Comparers" The archive contains a lot of hand-written manuscripts by a certain philosopher. The users want to browse these like books, navigating by chapters and pages. Unfortunately, the philosopher's handwriting is unreadable for normal people, so the archive also contains transcriptions of the different paragraphs. The user wants to click on a paragraph in a manuscript and see the correct transcription. Finally, the philosopher often wrote different "versions" of the same thought in different places. The users are interested in comparing those versions, so they want to find all different versions of the current paragraph.

[7] For the sake of explanation, these use cases have been greatly simplified from the real world use cases

"The Topic Searchers" The archive contains the works of a different philosopher as an electronic text. The user wants to annotate each paragraph with the philosophical topics it deals with and then navigate the contents by topic. However, the users often disagree on the topics, so they also want to annotate the topic with their different opinions.

"The Movie Enthusiasts" The archive contains movie files concerning philosophical topics. The users want to view the videos and while watching there should be links to the current section of the video is about. All the videos refer to documents that are stored in other archives, and different parts of the videos can refer to different things.

5 Talia

The Talia platform stores digital objects, called *sources*, which are identified by their unique URLs. Each source can have one or more data files attached to it, and the system stores information about the source objects. Talia also provides a user interface framework that allows to build web interfaces for different requirements, using modular components called *widgets*.

The communication between different Talia servers will be done through a REST interface, an approach proposed by R. Fielding in [5].

Talia is developed using the Ruby on Rails[8] web application framework and it will be publicly available from the Talia web site[9].

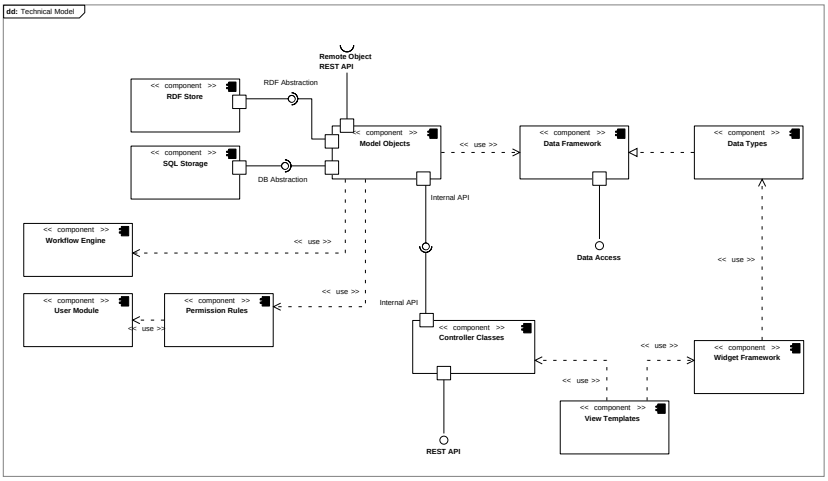

Fig. 1. Overview of the Talia system architecture

[8] http://www.rubyonrails.org

[9] http://www.talia.discovery-project.eu/

Architecture overview. Figure 1 provides an overview of the general architecture of the system and show the components of which Talia is composed:

Model Objects. The model classes provide an interface to the sources; they also contain a query mechanism to locate and retrieve sources. They can retrieve data from remote Talia nodes.

RDF Store. It contains the metadata on the sources. The data is stored as RDF triples, allowing to store any kind of descriptions about the sources.

SQL database. It contains all the information for which classic ACID requirements (see for example [6]) exist. This would contain things like the current workflow state of a source, user permissions, and the like.

Data Framework. It provides an API to access the actual data stored in the system. Different *datatypes* provide more advanced functionality for each document type, e.g., the ability to read lines from text files or a video stream from a multimedia file.

Controller Classes They contain the logic that connects the model objects with the user interface and drives the application. The controllers also provide an easy to use REST interface that allows other software to remotely interact with the library.

Widget Framework. It allows the development of widgets that can be used to build customised user interfaces. Widgets can have their own behavioural logic.

View Templates. The web site templates that render the user interface of the application. These use the data set up by the controllers, and can use widgets to place common interface elements.

Other The *Workflow Engine*, *User*, and *Permission Modules* contain logic that is need to manage the library. They are without the scope of this short introduction.

User interface. Talia will include its own framework for deploying *widgets*. These modular interface elements can be packaged independently and used as building blocks for the application's user interface. A number of common widgets will be shared by all archives, and there will be also specialised widgets for single archives that have specific needs. The archives may also modify the HTML rendering templates to customise their site's appearance.

Figure 2 shows an example of such a widget: The *context sidebar* presents the user with links to all documents related to the source currently viewed.

Dynamic Contextualisation. The *dynamic contextualisation* provides a method for different Talia nodes to exchange data. The mechanism functions as follows:

If a node adds a relation to a document on different ("foreign") node, it writes the information to its internal RDF store. Then it notifies the foreign node; and the foreign node will decide independently on whether or not to add the relation to its local RDF store.

Fig. 2. Context sidebar in the Talia application

"Foreign" documents will be presented in the navigation in the same way as local documents. If the user follows a link to a foreign document, she will be taken to the other archive's site and she will be able to continue to browse there. If that side has accepted the contextualisation request, a back link to the original site will also be provided.

5.1 Requirements for Talia's Semantic Layer

The most characteristic aspect of Talia is that, for the first time, at least in the field of humanities, all public interfaces will be based on Semantic Web standards, which enables interoperability within the Talia Federation and with other systems.

According to the original idea and to meet the requirements coming from the analysis of use-cases, Talia will have to include a *semantic software layer*, which will be based on the following general characteristics:

- *Making metadata remotely available.* The metadata content of the archives will be made available through interfaces similar to those of URIQA[10], which will allow automated agents and clients to connect to a Talia node and ask for metadata about a resource, retrieving its description and related metadata (e.g., the author, the resource type, etc.). This type of services enables numerous types of digital content reuse.
- *Querying the system using standard query-languages.* It is possible to directly query the Talia Federation using the SPARQL Semantic Web Query Language. SPARQL provides a powerful and standard way to query RDF repositories and enables the merging of remote data sets.
- *Transformation of encoded digital contents into RDF.* The semantic software layer will provide tools to transform textually encoded material, for example the transformation of manuscripts in TEI[11] format into RDF.
- *Managing structural metadata.* "Structural" metadata describes the structure of a document (e.g., the format of a document, its publisher, etc.). It will be necessary to provide tools to manage these kinds of metadata.

[10] `http://sw.nokia.com/uriqa/URIQA.html`
[11] `http://www.tei-c.org/`

The Talia semantic software layer will also provide facilities to *enrich* the *primary sources* and the *secondary sources* with semantic information, the type and format of which depends on the kind of source and on the user community that works on the data. For example, in some archives, different "versions" of the same paragraph may exist in different documents. In order to allow the users to follow the evolution of that particular philosophical thought, the relations between these different versions must be captured by the system.

Additional trouble may arise when archives use different formats for the annotation of documents. For this reason, Talia will have to provide dedicated ontologies and tools to describe information coming from different users and the relations among them.

5.2 The Semantic Layer in Talia

In order to fulfil the requirements of the use-cases in section 4.2 and to develop the semantic software layer in Talia, it was necessary to define its general architecture as well as to choose which of the already existing Semantic Web technology to use. This choice has been influenced by the programming language used to develop the Talia system.

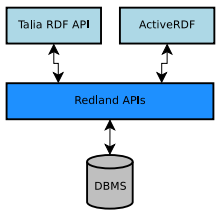

Fig. 3. General architecture of the Talia Semantic Layer

Figure 3 shows the general architecture of the Semantic Web Layer in Talia. We decided to use Redland RDF [7] as main tool to manipulate RDF data. Redland RDF provides a set of free software libraries to manage and manipulate whole RDF graphs, single triples, URIs, and literals. It also supports different storage systems, so we might choose to replace MySQL at any time without the necessity to change the Talia source code. For the first prototype of Talia we chose to use the MySQL DBMS.

On top of Redland RDF we use ActiveRDF [8], a library to access and manipulate RDF data in Ruby. ActiveRDF also provides an abstract query engine which is independent of the data source and the specific query language.

Currently ActiveRDF does not meet all requirements from the use-cases. It was therefore extended and an additional high-level API was developed. This API can interact directly with the low-level API provided by Redland. Since

ActiveRDF does not natively support the use of DBMS like MySQL or PostgreSQL, we designed and developed new dedicated data adapters. Moreover, a basic API was developed, which provides dedicated methods to delete single RDF statements on Talia sources, as well as methods to remove specific sources and their related data.

A flexible system like Talia requires facilities to easily work with properties and related values. New tools to manipulate RDF predicates have been developed, introducing a new dedicated syntax to speed up the creation, modification, and deletion of properties and groups of properties.

Digital contents of Talia archives structured by RDF can be exported in different ways. We provided Talia system with flexible and optimised tools, which currently allow RDF data export/import in RDF/XML and N-Triples formats. The feature for exporting RDF data could be easily extended included others formats like N3, plain-text, etc.

To organise information and support the semantic enrichment of the content provided by partners with different needs, it was decided not to develop a common ontology for the whole project. Instead, we defined only a very broad *structural ontology*, which contains only some general concepts and relations to link documents, and each Discovery content partner will develop its own *domain ontology* for the specific needs of each archive. For this reason we also developed basic tools to manage and load ontologies for the first prototype. These tools will be upgraded for the final version of Talia, including features which will allow to remove old ontologies from the system and to synchronise and update different versions of the same ontology.

6 Related Works

Talia is directly related to the Hyper platform, which was used for the HyperNietzsche[12] archive. HyperNietzsche and the Hyper platform are described in [1].

HyperNietzsche provides a way of navigating the digital Nietzsche archive. The software was specifically developed for the community of Nietzsche scholars, with many different features tailored for the needs of the Nietzsche researchers.

Hyper did not use semantic web technology extensively, and was a highly specialised solution. Modifying this code base for the needs of the Discovery partners will not be feasible; thus the Hyper platform will be retired during the Discovery project and will be superseded by Talia.

The authors are also aware of other semantic digital library projects, such as JeromeDL [9], BRICKS [10] and Fedora [11]. Especially the latter ones provide robust frameworks for digital library applications. However, the focus is mainly on providing an infrastructure; none of the projects offers the kind of strong integration between the user interface framework and the semantic library backend that is required for Discovery.

[12] `http://www.hypernietzsche.org/`

7 Conclusion and future work

We have presented a prototype of the Talia platform, an application which aims at easing scholarly research in philosophy. We have also presented, as a use case, the motivation for the development and implementation of Talia and the role played by computational ontologies and Semantic Web technologies. By using ontologies to organise information and to formally describe the relations among the digital contents it has been possible to develop a very adaptable, state-of-the-art research and publishing environment for the philosophy.

A lot of effort is currently put into the development of Talia, and the release of an alpha version is planned for the beginning of 2008. A preliminary date for a beta release is the end of June, 2008.

Acknowledgements

This work has been supported by Discovery, an ECP 2005 CULT 038206 project under the EC eContentplus programme.

References

1. D'Iorio, P.: Nietzsche on new paths: The hypernietzsche project and open scholarship on the web. In Fornari, C., Franzese, S., eds.: Friedrich Nietzsche. Edizioni e interpretazioni. Edizioni ETS, Pisa (2007)
2. Calvanese, D., Guarino, N.: Ontologies and Description Logics. "Intelligenza Artificiale - The Journal of the Italian Association for Artificial Intelligence" **3**(1/2) (2006)
3. Gruber, T.R.: A translation approach to portable ontology specifications. Knowledge Acquisition **5**(2) (1993) 199–220
4. Guarino, N.: Formal Ontology and Information Systems. In Guarino, N., ed.: 1st International Conference on Formal Ontologies in Information Systems, IOS Press (1998) 3–15
5. Fielding, R.T.: Achitectural Styles and the Design of Network-based Software Architectures. PhD thesis, UC Irvine (2000)
6. Gray, J.: The transaction concept: Virtues and limitations. In: 7th International Conference on Very Large Data Bases, 19333 Vallco Parkway, Cupertino CA 95014, Tandem Computers (1981) 144–154
7. Beckett, D.: The Design and Implementation of the Redland RDF Application Framework. In: 10th International World Wide Web Conference (WWW2001), Hong Kong. (2001)
8. Oren, E., Debru, R., Gerke, S., Haller, A., Decker, S.: ActiveRDF: Object-Oriented Semantic Web Programming. In: 16th International World Wide Web Conference (WWW2007), Banff, Alberta, Canada. (8-12 May, 2007) 817–823
9. Kruk, S., Woroniecki, T., Gzella, A., Dabrowski, M., McDaniel, B.: Anatomy of a social semantic library. In: European Semantic Web Conference. Volume Sematic Digital Library Tutorial. (2007)
10. Risse, T., Knežević, P., Meghini, C., Hecht, R., Basile, F.: The bricks infrastructure - an overview. In: The International Conference EVA, Moscoww (2005)
11. Fedora Development Team: Fedora open source repository software: White paper. White paper, Fedora Project (2005)