HyperJournal, PHP scripting and Semantic Web technologies for the Open Access

Michele Barbera[1], Francesca Di Donato[2], Giovanni Tummarello[3,4], Christian Morbidoni[3,4]

[1] Net7 – Internet Open Solutions – Pisa, Italy – www.netseven.it
[2] Dipartimento di scienze della politica, Università di Pisa, Pisa, Italy
[3] Università Politecnica delle Marche, Ancona, Italy – semedia.deit.univpm.it
[4] CNR – ISTI, Pisa, Italy

Abstract. In this article we present an high level overview of the HyperJournal project, an effort to provide novel possibilities both in Scientific Publishing and in access to Scientific Contributions, according to the Open Access movement guidelines. All the work has been implemented using the PHP Web Script language and interfacing with Java modules such as Sesame and RDFGrowth. Such interfaces, here illustrated, are of general use for project with similar needs. While the HyperJournal project itself is in its infancy stage, a first release is already available for download and public use, thus representing one of the certainly not many real and deployable examples of Semantic Web applications.

Introduction

HyperJournal is an Open Source web application written in PHP that facilitates the administration of academic journals on the Web. The project is born to fulfill to the requests of researchers and public institutions for an easy and low-cost system for the Open Publishing. Also, the project seeks to directly contribute to the Open Access movement[1], a common mission highlighted in the Berlin declaration [2] to provide knowledge dissemination and to make information widely and readily available to society. In order to reach this goal, it is stated, content and software tools must be openly accessible and compatible. Open Access has rapidly increased its number of supporters, with big institutions such as CERN and major universities undersigning the declaration and adopting an Open Access policy for their academic and scientific production. The Semantic Web initiative naturally fits in this, see for example the many similarities with the tools of OAI-PMH[4], a protocol for metadata harvesting that entitles anyone to create metadata about any page and share that metadata with everyone.

HyperJournal philosophy and goals

The primary objective of the HyperJournal project is to deliver a free information space where researchers are able to set their own selection criteria in discovering relevant information sources, instead of relying on those enforced by publishers. In order to have an impact on the real everyday work of the scholar community, especially well known to be resistant to change, innovations must certainly be introduced with caution. For this reason the first step of the HyperJournal project is to transpose the time-honored system of scholarly citation into an electronic environment.

Then, Hyperjournal adds innovative features such as the compatibility with the Protocol for Metadata Harvesting of the Open Archives Initiative to ensures maximal interoperability between the HyperJournal network and other electronic publications.

Finally, HyperJournal aims to promotes the use of metadata in order to facilitate selection of relevant information sources according to the needs and the interests of the specific browser, thus giving a previously unavailable freedom. This is one of HyperJournal's most relevant features: dynamic contextualization; cross-references contained in journal articles are automatically transformed into hypertextual, bidirectional links. By clicking on an author's name, for example, the HyperJournal system automatically searches the across the entire network of linked HyperJournals and produces a citation list that includes all the articles written by the author, all the articles the author has cited, and all the articles that cite the author.

RDF Networks of citations

In scientific works citations naturally form a "conceptual network of documents". This information however is still not encoded in machine readable languages. Inside HyperJournal, this is a task is based on RDF, where citations occurring between published articles are conceptually represented by a directed labeled graph.

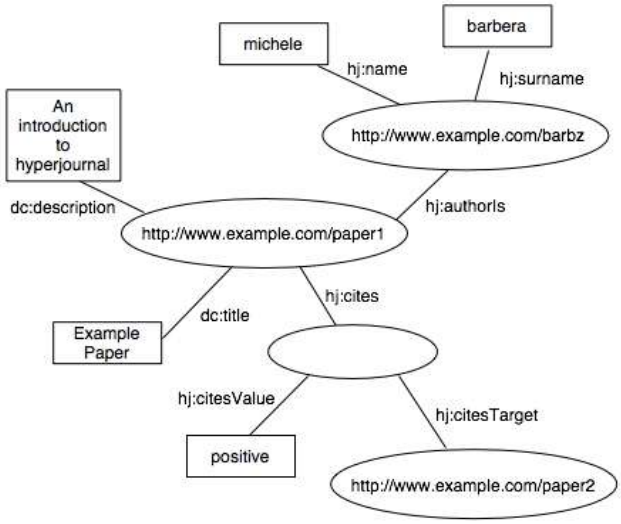

Fig. 1. An RDF graph representing a citation

During the evolution of the HyperJournal project we also developed an ontology for representing and encoding citations and cross references between articles while the Dublin Core metadata vocabulary has been used as much as possible. Once encoded in RDF, knowledge discovery techniques can help to classify and cluster citations and thus discover development trends, anomalies and dependencies among workgroups and research topics [5][6]. While this feature is undoubtedly useful for a single HyperJournal installation, it is even more so when a number of journals becomes connected and able to share citation metadata among them. Such connection happens in Hyper Journal thanks to the embedding of the RDFGrowth P2P engine for RDF metadata replication, as illustrated in the following section.

RDFGrowth: a scalable P2P engine based on the "minimum commitment" principle

The RDFGrowth algorithm [16] powers the HyperJournal ability to collect RDF metadata from other peers. Previous projects, such as [8][9], have explored P2P interactions among groups of trusted and committed peers. By this we mean that peers rely on each other to forward query requests and collecting and returning results. In contrast, RDFGrowth is designed to operate in a real world scenario of peers where cooperation is relatively frail. By this we mean that while peers are certainly expected to provide some "external service", commitment is in general minimal and in a "best effort" fashion. To obtain this, RDFGrowth, follows a peculiar philosophy: minimum external burden and maximum metadata replication. RDFGrowth will not perform any "active information hunt" such as query routing and replication. Such operations would require peers to do work on behalf of others, that means that single users could potentially cause large external burden on the network. In HyperJournal, on the contrary, the user browsing and searching is totally supported by the *local*, potentially large, metadata database, while it is the RDFGrowth algorithm that "keeps it alive" by updating it in a sustainable, "best effort" fashion. This is based on a particular kind of direct remote queries which is guaranteed to be fast and efficient to execute. As an effect, the user seems to be browsing remote semantic knowledge while is in fact browsing a "local mirror" of it. While a complete discussion is outside the scope of this overview, the following table summarizes the positive and negative aspects:

PROS:

- Enjoy maximum speed and richness in browsing the “semantic web”, since he is doing so on his local copy of it.
- Run arbitrarily complex queries, that no external peer would accept to execute on his behalf
- Apply personalized reasoning, rules and trust filtering.

CONS:

- Might require a large HD space if a broad “interest profile” is selected.
- Although the time it takes to get a new piece of information is bounded, the approach is not suitable for real time structured informations (e.g. Current weather in RDF)
- It takes some startup time
- The RDF knowledge must be “shaped” to work well with the RDFGrowth algorithm.

Clearly, these features are well suited for the HyperJournal project where information is not expected to change quickly (with respect to the workings of the algorithm) while maximum speed of querying is needed and local trust based filtering must be supported.

Phesame: a Sesame-Php5 Interface

When planning the development of the HyperJournal application, we choose Sesame [17] as RDF database. The web programming language of choice was PHP 5, both because its great diffusion and common understanding perfectly fit our objectives and because of its flexibility .
Unfortunately, and unlike PHP4, PHP 5 had no direct Java integration, something that was needed the Java modules, namely Sesame and RDFGrowth. In this section we'll take Sesame as an example and show a simple solution that makes use of the http calls and wraps them in a simple PHP class to interface with it. This class is called *Phesame*, as a pun with “Sesame” and “PHP”. Phesame is simple and lightweight, and provides a PHP interface to any locally or remotely installed Sesame repository. As such, it is different from other PHP semantic web toolkits like [10] or [11].

As outlined above, Phesame is basically a wrapper around the Sesame HTTP API, which permits various operations to be executed. Specifically, the Sesame API permits the following operations:

- Login/Logout
- Requesting a list of available repositories
- Evaluating a SeRQL-select, RQL or RDQL query
- Evaluating a SeRQL-construct query
- Extracting RDF from a repository
- Uploading data to a repository
- Adding data from the web to a repository
- Clearing a repository
- Removing statements

Not all the features implemented are needed for HyperJournal's needs. Interestingly, removing statements is not used. As Hyperjournals share RDF statements in a p2p network that grows monotonically, removing statements is pointless. Statements are instead “superseded” using techniques outside the scope of this overview.
Phesame is a PHP class, which uses the PEAR HTTP_Request library. An optional extension of Phesame exists, which introduces some utilities and a format to represent simple SeRQL queries as PHP arrays. This last class, PhesameExt, will be examined in the end of this chapter. Before a Phesame instance is ready to communicate with Sesame, we need to set some parameters regarding the location of the required repository and the login data we want to use. As mentioned, this is based on the HTTP protocol, so we need to indicate a URL to the repository. This is done by passing it to the constructor; obviously, the URL will be in the “domain:port/path/to/Sesame/servlet” format.

```
$phesame = new Phesame('127.0.0.1:8080/sesame/servlets');
$phesame->setResultFormat('xml');
$phesame->setUploadFormat('rdfxml');
$phesame->setQueryLanguage('SeRQL');
```

Other parameters are optional and are quickly explained here along with their default values and the name of the relative method:

Upload format
The format of the data that is uploaded to Sesame.
method name: setUploadFormat
possible values: rdfxml, ntriples and turtle
default value: rdfxml

Result format
Determines the format of the result. Legal values are `xml` for an XML-representation of the query result, `html` for a human-readable HTML-table presentation and rdf for an RDF-representation of the variable bindings.
method name: setResultformat
default value: xml

Query language
The query's language.
method name: setQueryLanguage
possible values: SeRQL, RQL, RDQL
default value: SeRQL

The Sesame user data can then be specified with the login method, which accepts a user/password couple.

```
$phesame->login('login', 'password');
```

A small number of repository listing and filtering methods are also present, which can be used prior to making this selection.

```
print_r($phesame->listWriteable());
```

A specific repository must then be selected with the setSelectedRepository method.

```
$phesame->setSelectedRepository('testRDFrepository');
```

Once connected with the right repository, the main methods that will be used regard querying and inserting of data.
The uploadData method, as the name may suggest, is used to put data in the repository:

```
$rdf = "<rdf:RDF
        xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
        xmlns:foaf="http://xmlns.com/foaf/0.1/"
        <foaf:Person rdf:nodeID="me">
                <foaf:name>Michele Barbera</foaf:name>
                <foaf:title>Mr</foaf:title>
                <foaf:givenname>Michele</foaf:givenname>
                <foaf:family_name>Barbera</foaf:family_name>
                <foaf:nick>barbz</foaf:nick>
        <foaf:mbox_sha1sum>
                        9859fb92c86f49b8ab9a1f0ca19978eb071c4f14
                </foaf:mbox_sha1sum>
                <foaf:homepage rdf:resource="barbera.netseven.it"/>
        </foaf:Person>
</rdf:RDF>";
$phesame->uploadData($rdf, true, false, false);
```

The simpleQuery method will accept a query in the query language specified with the setQueryLanguage method and return the corresponding result in the format specified by the setResultFormat method.

```
$serql = "SELECT Author, Paper
        FROM
```

```
            {Paper} rdf:type {foo:Paper};
        foo:keyword {"RDF", "Querying"};
                dc:author {Author}
         USING NAMESPACE
                dc  = <http://purl.org/dc/elements/1.0/>,
                foo = <http://www.foo.org/bar#>";

$result = $phesame->simpleQuery($serql);
```

The result variable will be populated by query results in the selected result format. If the format has been specified in rdf-xml we will get something similar to:

```
<?xml version="1.0" encoding="UTF-8"?>
<tableQueryResult>
  <header>
    <columnName>Paper</columnName>
    <columnName>Author</columnName>
  </header>
  <tuple>
    <uri>http://www.example.org/papers/a_paper.pdf</uri>
    <literal>Michele Barbera</literal>
  </tuple>
  <tuple>
    <uri>http://www.example.org/papers/a_paper.pdf</uri>
    <literal>Giovanni Tummarello</literal>
  </tuple>
</tableQueryResult>
```

We can use an xml library such as simpleXML to transform the query results into a PHP data structure such as an array. Finnally, any connection with Sesame can be closed with the logout method.

```
$phesame->logout('user','password');
```

To conclude our discussion on Phesame, we just wish to highlight that Sesame is a very capable tool, which is yet easy to understand and operate. Albeit very simple in its structure and functioning, it is possible that Phesame could play an important role in attract the large PHP communities to the Semantic Web thanks to the ease of accessing such powerful instrument.

Trust and Reputation

Sharing information on the HyperJournal's P2P network raises several challenges including finding appropriate sources of information the user is interested in and deciding whether that information is trustful or not. Nowadays the HyperJournal tackle the problem of selection at a federative level, where each node of the federation acts as publisher and decides whether a contribution can be published or not, according to its own notion of quality. Nevertheless subjectiveness and variableness in the idea of quality and the possibility of collaborative annotations request a more powerful solution, one that can filter information according to trust criteria already existing within relationships among researchers. Usually, the selection of information happens through trust networks, by word of mouth, acquaintances of colleagues at congresses, trust in a publisher, reviews, or the organization the author belongs to; all those elements are implicitly evaluated at the same time when the researchers has to operate their selections. If the information is semantically structured on the Web, it is possible to automatically emulate some of these processes or even introduce new ones [12].

The relations shown by the contextualization are the citations contained in the articles approved by the *peer review*. Every Journal has its own *peer review*. As journals adopting contextualization increase, the quality control decrease. Everyone can start a journal and accept low quality contributions which cite articles published in other journals. The immediate consequence is that the cited articles are submerged by scarcely relevant contextual information, the so-called “contextual noise”. A possible solution could be to allow journal publishers to select *trusted sources* in order to draw the citation metadata, so that every Journal could show the contextual information only drawing them from the trusted journals.

Another mechanism could be based on user decisions instead on publishers'. Users would select the sources they regard as trustworthy, in order to draw contextual information. This however raises a problem of information overload and needs a greater process of selection as authors should constantly update their trustworthy sources list. This problem could be partially solved by exchanging the source list according to mutual trust among the researchers. If Alice trusts Bob, and Bob updates his trust list with a new record, Alice's might automatically update ("reputation-based" trust models. [13])

In HyperJournal, we're currently experimenting both the scenarios outlined above using the FOAF ontology to represent groups and individuals. The nature of the HyperJournal's networks fits naturally with a reputation based system to select information sources. Once the information is delivered and locally available to a peer, it is used to decide whether that information is trustful or not. Several approaches have been proposed based on public key certifications [14] [15]. While these approaches seems very promising, in order to be applied in our project they require the local data sources to be able to manage provenance information. To be able to locally compute whether signed statements received from other sources are trustful or not we must be able to filter out statements coming from untrusted sources during the very execution of query to the local data source. For this reason we started a collaboration with the Sesame team to explore the possibility of implementing a provenance and context model into the SeRQL query language.

Conclusions and future works

In this article we presented an high level overview of the HyperJournal project, and effort to provide novel possibilities in scientific publishing and consultation.
While the HyperJournal project itself is in its infancy stage and will need further development and evaluation, all the work has been implemented using the Web Script language PHP and interfacing with Java modules such as Sesame and RDFGrowth using extensions that have been here illustrated and are of general use for project with similar needs.
All the source code here presented is available for download under an approved Open Source licence.

References

[1]P. Suber, *Open Access Overview*, http://www.erlham.edu/peters/fos/owerview.html

[2]*Berlin Declaration on Open Access to Knowledge in the Sciences and Humanities*, october 2003, http://www.zim.mpg.de/openaccess-berlin/berlindeclaration.html

Dublin Core Metadata initiative, http://dublincore.org/

[4]*The Open Archive Protocol for Metadata Harvesting*, http://www.openarchives.org

[5]E. Garfield, *Citation Indexing: Its Theory and Application in Science, Technology, and Humanities*. Wiley, New York, 1979

[6]E. Schallehn, M. Endig, K. Sattler, *Citation Linking in Federated Digital Libraries*, Proc. 3rd Int. Workshop on Engineering Federated Information Systems, EFIS'00, Dublin, Ireland, 2000

C.L. Giles, K. Bollacker, and S. Lawrence, *CiteSeer: An Automatic Citation Indexing System*, Digital Libraries 98: Third ACM Conf. on Digital Libraries, ACM Press, New York, 1998, pp. 89-98.

[8] Wolfgang Nejdl, Boris Wolf , *EDUTELLA: A P2P Networking Infrastructure RDF* 2002 WWW2002, Honolulu

[9] Min Cai, Martin Frank , *RDFPeers: A Scalable Distributed RDF Repository based on A Structured Peer-to-Peer Network* 2004 13th International World Wide Web Conference WWW2004, New York

[10]*pOWL*, http://powl.sourceforge.net/

[11]*RAP*, http://www.wiwiss.fu-berlin.de/suhl/bizer/rdfapi

[12]J. Golbeck, B. Parsia, J. Hendler, *Trust Networks on the Semantic Web*, Proceedings of Cooperative Intelligent Agents 2003, Helsinki, Finland.

[13])P.A. Bonatti, N. Shahmehri, C. Duma, D. Olmedilla, W. Njedl, M. Baldoni, C. Baroglio, A. Martelli, V. Patti, P. Coraggio, G. Antoniou, J. Peer, N.E. Fuchs, *Rule-based policy specification: State of the art and future work*, Project deliverable D1, Working group I2, EU NoE REWERSE, 2004

[14]A. Chirita, S.Idreos, M. Koubarakis, W. Njedl, *Publish/Subscribe for RDF-Based P2P Networks*, Proceedings of the 1st European Semantic Web Simposyum, Heraklion, Greece, 2004.

[15]J. Carrol, C. Bizer, P. Hayes, P. Stickler, *Semantic Web publishing using named graphs*, Workshop on trust, security and reputation on the Semantic Web, Hiroshima, Japan, 2004.

[16]G. Tummarello, C. Morbidoni, J. Petersson, F. Piazza, M. Mazzieri, P. Puliti, *Toward widely deployable Semantic Web P2P: tools, definitions and the RDFGrowth algorithm*, ISWC'04 workshop on Semantic Web Technology for Mobile and Ubiquitous Applications, 7th November 2004, Hiroshima, Japan

[17] Jeen Broekstra, Arjohn Kampman, and Frank van Harmelen *Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema*. First International Semantic Web Conference (ISWC 2002)