

Integrating Chemistry Scholarship with Web Architectures, Grid Computing and Semantic Web

Sashi Kiran Challa, Marlon Pierce, Suresh Marru
Community Grids Laboratory, Pervasive Technology Institute
Indiana University, Bloomington
2719 East 10th Street, Bloomington, Indiana 47408
{schalla, marpierc, smarru}@indiana.edu

Abstract — A chemist given a compound would be interested in knowing the experiments performed using the compound, journals containing the compound and also molecular properties of the compound. If there is a way to integrate this data, it would enhance the chemist's knowledge about a given compound. The Object Reuse and Exchange (ORE) specification may provide a solution to this problem. ORE is a model proposed by the digital libraries community to aggregate resources on the web. OREChem is a research project funded by Microsoft External Research that aims to apply and extend ORE to enable the integration of experimental, bibliographical and molecular properties data. OREChem targets crystallography as its primary application domain. This effort will design a prototypical, semantic-based eScience infrastructure for chemistry and chemical informatics. In this paper we describe how we have used REST as well as SOAP web services, TeraGrid cyberinfrastructure and Semantic Web technologies, such as RDF and triple stores, to facilitate the metadata integration.

Keywords - *Information Storage and Retrieval, Online Information Services – Web-based services, REST services, Rule-based databases*

I. INTRODUCTION

This paper describes a prototype infrastructure for harvesting, storing, and adding value to crystallographic metadata from the Web. Our infrastructure implementation consists of a set of network services that harvest crystallographic ATOM/XML feeds. We add value to these feeds by using the information contained in them to run computational chemistry calculations that determine fine-grained structural properties. All information is placed in a searchable triplestore. A triplestore is a framework for storing and querying Resource Description format (RDF) data [1]. The services that perform the harvesting and processing are integrated as a workflow.

To illustrate the concepts of Web-based metadata management for crystallography, we start with a motivating example. In order for a crystallographer to gather information about a particular crystal, he or she will look for experimental procedures for synthesizing

the crystal, and different journal articles related to the crystal. Also he/she would be interested in knowing the moieties that constitute that crystal. Moiety is a specific group of atoms within a crystal that is responsible for characteristic chemical reactions of that crystal structure. A crystal structure can have one or several moieties. Currently this experimental and bibliographic data are on the Web but are not discoverable in a coherent manner. Integrating experimental and bibliographic data and also the molecular properties data of all the moieties belonging to a particular crystal would support queries on all the data simultaneously. The Object Reuse and Exchange (ORE) specification and more generally Semantic Web technologies enable encoding this data in a way that it can be integrated or linked in metadata resource documents.

For the general reader to understand the core of our development work, we first provide some background on the Semantic Web. The Semantic Web is a development of the World Wide Web Consortium in which semantics are added to the information or data on the web to enable machines to make assertions about the data. RDF is the simplest data model used to represent the data in Semantic Web. An RDF triple is a subject, predicate and object statement [2]. The subject denotes the resource, and the predicate denotes traits or aspects of the resource and expresses a relationship between the subject and the object. Subject, predicate and object are represented as Uniform Resource Identifiers (URI). Objects can even be literals. The predicate is usually defined by ontology.

On the World Wide Web, a resource represents an item of interest, and every resource has a URI. A resource might have information similar to another. Thus there can be links between the resources. Open Archives Initiative Object Reuse and Exchange (OAI-ORE) [3] is an initiative in the digital libraries community that defines standards for aggregating resources on the Web. A published paper, its supporting experimental data, and its various incarnations (preprints, revised versions) are examples of OAI-ORE resources. The components of the resource do not all need to exist in one central database.

According to the ORE model an URI is assigned to an aggregation of resources. Following the Linked Data guidelines [4], a new resource is introduced to make information about the aggregation available. This new resource is called Resource Map, has its own URI, and expresses the aggregation it contains [3]. The collection URIs for a publication's various paper versions, supporting data, and other similar metadata is an example of a Resource Map. The result of an HTTP access of a Resource Map URI is a serialization of the triples describing the aggregation. This serialization may be in any of the OAI-ORE serialization syntaxes, which include RDF/XML, N3, Terse RDF Triple Language (a.k.a Turtle) [5], RDFa and ATOM. The OAI-ORE data model provides a set of base concepts that can be extended and specialized for specific domains [6]. Figures 1 and 2 show how we have extended OAI-ORE for this project. Figure 1 shows an ORE aggregation of a moiety in Turtle format. It shows two other resources that are related to the moiety; one a '.png' file, and another a 'complete.cml.xml' file. Figure 2 shows an ORE representation of a Resource Map in Turtle format.

OREChem project applies the ORE model to crystallographic scholarship data. The OREChem project is funded by Microsoft External Research and is collaboration between crystallographers and information scientists to develop and demonstrate the infrastructure, services, and applications to enable new models for research and dissemination of scholarly materials within the chemistry community [7]. Cornell University, Penn State University, Indiana University, University of Southampton, and University of Cambridge are part of this project.

The OREChem infrastructure consists of three main components: eCrystals, Crystal Eye and Computation (described here). The components are linked using web-feeds that comply with ORE's ATOM bindings. Bibliographic metadata related to crystals is extracted from journal articles by collaborators at Penn State University [8]. Experimental data is archived in the eCrystals [9] repository and that data is modeled using the Experimental Ontology developed by collaborators at University of Southampton. The Crystal Eye process developed by collaborators at University of Cambridge [10] converts CIF files [11] obtained from eCrystals into an XML format using Chemistry Markup Language (CML) [12]. CrystalEye also extracts moieties and X-ray crystallographically obtained 3D co-ordinates of a particular crystal and its moieties into the CML format. The moiety information in CML is in the form of a file published as ATOM archive feeds. Every moiety also is associated with an N3 file that has experimental, bibliographic, moieties molecular data encoded by RDF/XML. The N3 file is also published as ATOM archive feed. Indiana University is developing the computational component of the OREChem infrastructure. Section A summarizes the contribution of IU team to the overall OREChem project. In sub-sections

1, 2 we describe the workflow, the REST-style services that are the workflow's components. In Section B we describe the Open Link Virtuoso triplestore that we deployed to provide a searchable archive for the linked triples of the metadata that are produced by the project. In Section C we describe how we use the TeraGrid and Open Grid Computing Environment (OGCE) Workflow suite to build and execute the workflow. In Section III we discuss what is being achieved using this workflow, and in Section IV we discuss the future work for this project.

```
<http://wwmm.ch.cam.ac.uk/crystaleye/summary/soton/ecrystals/2010/08-06/
data/1046/1046_90mz09/moieties/1046_90mz09_moiety_1>
a    chem:MolecularEntity , ore:Aggregation ;
dcterms:source <http://wwmm.ch.cam.ac.uk/crystaleye/summary/
soton/ecrystals/2010/08-06/
data/1046/1046.cif> ;
ore:aggregates <http://wwmm.ch.cam.ac.uk/crystaleye/summary/
soton/ecrystals/2010/08-06/
data/1046/1046_90mz09/moieties/1046_90mz09_moiety_1/
1046_90mz09_moiety_1.png> ,
<http://wwmm.ch.cam.ac.uk/crystaleye/summary/soton/ecrystals/2010/08-06/
data/1046/1046_90mz09/moieties/1046_90mz09_moiety_1/
1046_90mz09_moiety_1.complete.cml.xml> ;
ore:isDescribedBy <http://wwmm.ch.cam.ac.uk/crystaleye/summary/
soton/ecrystals/2010/08-06/
data/1046/1046_90mz09/moieties/1046_90mz09_moiety_1/
1046_90mz09_moiety_1.n3> ;
```

Figure 1: ORE representation of an Aggregation of a Moiety in Turtle format

```
<http://wwmm.ch.cam.ac.uk/crystaleye/summary/soton/ecrystals/2010/
08-06/data/1046/1046_90mz09/moieties/1046_90mz09_moiety_1/
1046_90mz09_moiety_1.n3>
a    ore:ResourceMap;
dc:rights "To the extent possible under law, the
University of Cambridge has waived
all copyright and related or neighboring rights to this work.
This work is published from United Kingdom.";
dcterms:creator <http://wwmm.ch.cam.ac.uk/crystaleye>;
ore:describes "http://wwmm.ch.cam.ac.uk/crystaleye/summary/
soton/ecrystals/2010/08-06/data/1046/1046_90mz09/
moieties/1046_90mz09_moiety_1"
```

Figure 2: ORE representation of a Resource Map in Turtle format

II. ORECHEM-COMPUTATION WORKFLOW

The OREChem-computation workflow (Figure 3) that we are developing at Indiana University processes the ATOM feeds published by the Crystal Eye component in the OREChem workflow. It has two main branches. The first component facilitates storing the N3 files obtained from the ATOM feeds into the project triplestore. The second component adds additional information about crystal structure by computing detailed chemical properties like energies, geometries, and vibration frequencies using computational chemistry codes (Gaussian 09 in the current implementation).

The workflow is composed of individual REST-style web services written using Java Jersey. Each service is responsible for a self-contained operation. These services are composed into a workflow in our case study, but they can also be used independently. Following Web service best practices, the inputs and outputs of the services are all explicit in the interface; for instance, if a service generates files from a feed, then the service returns URLs to all the files that it creates. Transferring files between services is an implementation detail; the URL contains the protocol necessary for the next service to obtain the file (HTTP in this case). The case study also utilized SOAP based web service to execute long running parallel Gaussian jobs on supercomputing resources. The following sub-sections provide the technical description for the web-based services built and utilized in this case study.

A. REST Web Services

The OREChem project deals with aggregating data on the Web using Semantic Web technologies and performing operations on resources. The services described here that we need to build can be represented by GET, PUT, POST, and DELETE operations. For these reasons, we chose to implement our core services as REST [13]. A REST service is invoked using a URL and every resource has a URI. For the services described here it takes few lines of code to build a REST web service using the Java Jersey library.

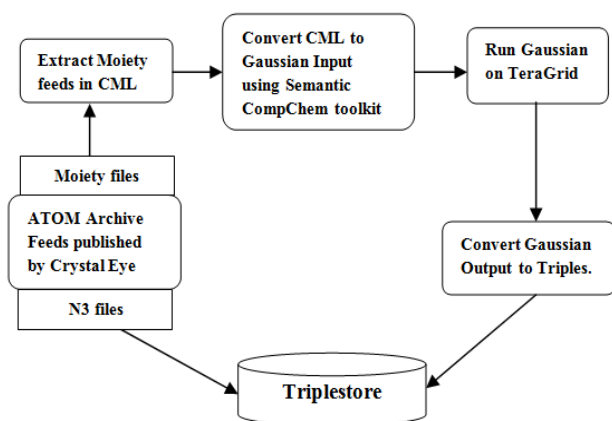


Figure 3: OREChem-computation workflow

1) *Jersey Implementation Details*: We used Java JAX-RS Jersey API (Jersey 1.2) [14] to build REST services. It has several Java annotations including @Path, @GET, @POST, @PUT, @DELETE, @Produces, @QueryParam(), and @PathParam(), that one can use to implement HTTP GET/PUT/POST/DELETE operations in one's web services. Also one could have a single class with several methods. Every method can correspond to a particular URL structure. When a particular URL is called, the corresponding method would be invoked. For instance one could have a single class returning a String array or a JSON array [15] depending on the URL structure

depending on which a particular method in the resource is invoked. Figure 4 shows one of the services we have written. In the 'MoietyHarvester' class we have 'harvestfeeds' method returning a String (comma separated values), and 'harvestfeedsJSON' method returning a JSON array. If one has 'csv' in the URL structure, harvestFeeds method will be invoked. If one has 'json' in the URL structure, harvestFeedsJSON method will be invoked. Jersey even has a separate server and client API.

2) *OREChem REST Services*: We initially developed six self-contained REST web services for this OREChem-computation workflow. However, since the collaborators at University of Cambridge started providing the data that was being generated and processed by our services, only three out of six are being used in the current implementation. In this paper we describe all the six web services and their implementation in detail. Table 1 summarizes the list of web services, their descriptions, input and output.

a) *InChIto3D* service generates 3D co-ordinates of a given IUPAC International Chemical Identifier (InChI string) [16]. InChI is a string of characters capable of uniquely representing a chemical substance. This service makes use of Open Eye Babel [17], a standard cheminformatics toolkit. The output is a String in CML format.

b) *InChIExtractor* takes the OREChem ATOM feed URL as input, looks for <chemdomain: identifier> tag in the ATOM/XML and extracts the InChI strings in the ATOM feed. It uses JDOM, JAXP libraries.

c) *FeedsHarvester* is wrapped around a tool developed by Crystal Eye group, called "crystaleye harvester" [18]. This tool fetches user specified number of moiety feeds into a local directory from the Crystaleye "moiety" repository. Each moiety feed that is fetched is a directory containing "cml.xml" file and "N3" file. This web service is written in such a way that the files that are fetched by the crystaleye harvester tool are made available as a String array of URLs or as a JSON array of URLs. A user can choose the desired output format by choosing a particular URL structure. This service fetches the N3 or triples files as URLs. By keeping track of the unique ids of moiety and triples files, we can make sure that the files that are already harvested or fetched at one time are not fetched again.

d) *CML2GaussianSemCompChem* is a service that transforms the contents of moiety CML file into Gaussian Input format. This transformation is brought about by using the "Semantic Comp Chem" toolkit [19], a cheminformatics toolkit being developed by the collaborators at University of Cambridge. The input to this service is the moiety CML file HTTP URL. This service does an HTTP GET operation on this moiety

CML file URL and transforms the contents into Gaussian input format [20]. The output of this service is a single HTTP URL to the generated Gaussian input file.

```
@Singleton
@Path("/cml3d")
public class MoietyHarvester {
    @GET @Path("/csv") http://gf18.ucs.indiana.edu/FeedsHarvester/cml3d/csv?parameters
    @Produces("text/plain")
    public String harvestfeeds(@QueryParam("harvester") String harvester,
    @DefaultValue("10") @QueryParam("numofentries") String num_entries){
    .....
    }
    @GET @Path("/json") http://gf18.ucs.indiana.edu/FeedsHarvester/cml3d/json?parameters
    @Produces("application/json")
    public JSONArray harvestfeedsJSON(@QueryParam("harvester") String harvester,
    @DefaultValue("10") @QueryParam("numofentries") String num_entries){
    .....
    }
}
```

Figure 4: Snapshot of REST service written using Jersey. Note the difference in the structure of the URL, difference in output formats & different Method names.

e) *ATOM2RDF* is a service that transforms data encoded as ATOM/XML into RDF/XML using Saxon-XSLT processor [21]. The input to this service is the ATOM feed URL and the output is the http URL to the RDF/XML file generated after the Saxon transformation. This service makes use of the GRDDL-compliant [22] XSLT style-sheet written for this project.

f) *RDFIntoVirtuoso* is a service that places triples into triplestore. URL to the N3 triples files is given as input to this service. It performs an HTTP GET operation on that file and writes its contents into a temp file before using the Jackrabbit [23] WEBDAV client's PUT method to place the contents onto the triplestore. The output of this service is the success or failure message for the triples uploaded.

These web services are deployed on one of a UNIX-based server. Software is managed using SVN on SourceForge [24]. We use Apache Maven as our build system, simplifying deployment. In the current implementation, FeedsHarvester, CML2GaussianSemCompChem, RDFIntoVirtuoso are the services that we are using to compose the OREchem-computation workflow.

B. Virtuoso triplestore

A triplestore is a framework for storing and querying RDF data in the Semantic Web technologies stack. It provides a mechanism for persistent storage and access of RDF graphs, such as those generated from OREchem atom feeds. There are several commercial and open source triplestores available. We chose Open Link's Virtuoso triplestore for this project. Virtuoso is an Object-Relational Database Management System

Table 1: List of RESTful services developed as part of the workflow. *-Currently being used in the workflow.

Web Service	Description	Input	Output
InChExtractor	Extracts InChIs by parsing the ATOM Feeds	ATOM feed URL	String of InChIs
InChIto3D	Generates 3D coordinates of an InChI. (Open Eye Babel)	InChI string	3D coordinates (CML)
FeedsHarvester *	Fetches the moiety feeds from Crystal Eye. (crystal-eye harvester)	harvester name, no. of feeds	JSON or CSV (CML URL's)
CML2GaussianSemCompChem *	Generates Gaussian Input file. (Semantic Comp Chem)	POST CML file URL	Gaussian Input file URL
ATOM2RDF *	ATOM to RDF/XML, SAXON-XSLT	ATOM feed URL	RDF/XML triples file URL
RDFIntoVirtuoso *	Places triples into Triple Store. (Jackrabbit WebDAV Client)	POST RDF/XML triples file	GRAPH IRI (SPARQL)

(ORDBMS) extended into an RDF triplestore [25]. One can upload the RDF files from the command line using the 'isql' utility or one can use the WEBDAV interface called "Conductor". Virtuoso also provides a web server and an interface for performing SPARQL [26] queries on the data uploaded into the triplestore. SPARQL is a query language for RDF. 5 billion triples is the current capacity for the open source version of Virtuoso product. We have installed the UNIX version of the Open Link Virtuoso triplestore (version 5.0.12) on one of our CentOS Unix machines.

All the files uploaded into the triplestore are stored in a location called 'rdf_sink'. The triples are stored in a table with four columns: GraphID, Subject, Predicate, and Object with IRI_ID, IRI_ID, IRI_ID, and ANY as respective data types [25]. For every file uploaded into the triplestore there is a unique GRAPH IRI created using which one can perform SPARQL queries for that particular file. However it is useful to be able to perform SPARQL queries on all the RDF triples in the triplestore; in other words we need to be able to query the whole RDF graph. By setting the "virt: sponger" property to "yes" and "rdf: graph" property to the Internationalized Resource Identifier [27] (IRI) we desire, we are able to have an IRI that can be used to query all the RDF triples in the triplestore.

C. TeraGrid & OGCE Workflow Suite

TeraGrid is an NSF funded Cyberinfrastructure that links major supercomputing and storage facilities with high-speed research networks. The TeraGrid also provides a unified administration layer, single sign on across resources, and a common middleware infrastructure for remotely accessing the resources. TeraGrid includes some of the largest supercomputers for academic use in the world (<http://www.top500.org/list/2010/06/100>). We use the TeraGrid resources for running the Gaussian09 on the

moiety CML files. MPI version of Gaussian09 is available on the TeraGrid.

To submit jobs to TeraGrid as well as to compose a workflow, we use the OGCE Workflow Suite [28]. It consists of tools to wrap command-line applications as lightweight web services and to execute and monitor the workflows. The Generic Service toolkit (also called GFac), XRegistry and XBaya are the three main OGCE tools. GFac allows users to wrap any command-line application as a SOAP protocol-based web service. XRegistry is the information repository of the workflow suite enabling users to register, search and access application and workflow deployment descriptions. XBaya provides a workflow graphical user interface, a Java Application invoked via Java Network Launching Protocol using Java Web Start. The XBaya GUI can be used for composing, executing and monitoring workflows from web services created by the GFac and other web service creation tools. This workflow suite facilitates submissions and management of jobs such as Gaussian runs on TeraGrid.

For this project, we registered OREChem REST Services in the XRegistry and built a workflow using XBaya workflow composer. XBaya is a SOAP workflow orchestration engine, in order to orchestrate the REST services; the services were wrapped as SOAP services using OGCE GFac toolkit. This is a heavy weight approach in translating the simpler REST services to SOAP, but the case study also needed to execute the Gaussian on TeraGrid and use of XBaya facilitated that. As discussed in the Future Work Section, the OGCE team is developing a hybrid REST and SOAP workflow orchestration capabilities in XBaya facilitating the REST services to be composed into workflows without wrappers. It takes around 10mins to 2 hours to run Gaussian, varying with the input files from the supplied moieties. A snapshot of the XBaya workflow composer is in Figure 5.

Figure 5 shows both the components of the OREChem-computation workflow. In the first component of the workflow, when the 'FeedsHarvester' service is run, it outputs a String array or a JSON array of moiety CML files as URLs. The files can be processed in parallel making use the workflow "for each" capabilities. Using 'For Each', 'End For Each' System components in XBaya further computations are performed on each of the HTTP URLs.

Each CML file URL is taken as input by the 'CML2GaussianSemCompChem' service, which transforms the contents to Gaussian input format and outputs a URL for each of the generated Gaussian input file. This URL array is then taken by the Gaussian Model service that submits the file to Gaussian09 on TeraGrid. (Currently these jobs are submitted to Gaussian09 on Abe Cluster at the National Center for Supercomputing

Applications [29]. Abe is one of the supercomputers part of the TeraGrid Cyberinfrastructure.) Once the Gaussian job is completed the output file is published as a URL that is taken as input by the service that transforms this Gaussian output into triples. This service is yet to be implemented.

Finally triples are put onto the Virtuoso triplestore using the 'RDFIntoVirtuoso' service. In the second component of the workflow, when the 'FeedsHarvester' service is run, it outputs String Array or a JSONArray of the URLs of N3 files. Using XBaya's 'For Each' and 'EndForEach' System Components each of these URLs is taken as inputs by 'RDFIntoVirtuoso' service, which stores the contents of the N3 files into the triplestore. Within XBaya there is a window in which the success or failure of each of the components in the workflow is displayed.

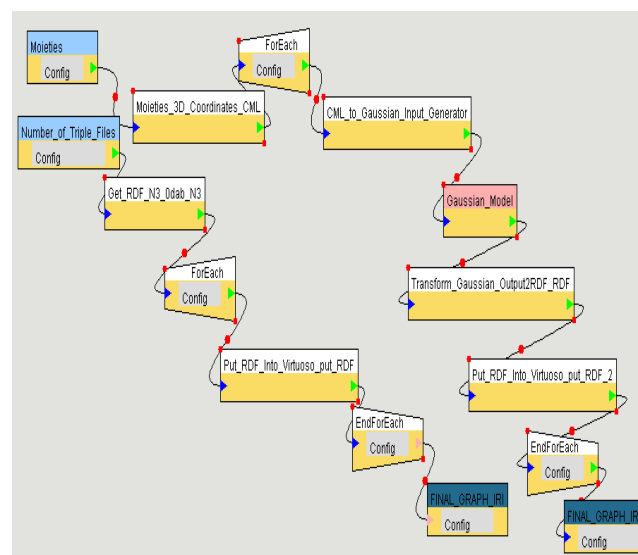


Figure 5: Snapshot of the current OREChem-computation workflow composed in the XBaya workflow composer

III. CONCLUSION

Resources to experimental data, bibliographic data and molecular properties data of crystal moieties are being integrated using the ORE, RDF model. We have developed individual REST-style services to (i) fetch moiety CML and ORE N3 atom feeds; (ii) generate Gaussian input files from moiety CML files and submit Gaussian jobs to Gaussian09 on TeraGrid; (iii) convert Gaussian09 output to triples, and store these triples as well ORE N3 triples into a triplestore. We have integrated these services into workflow that we execute using tools from the OGCE project. By using both REST-style and SOAP/WSDL Web architectures, TeraGrid Cyberinfrastructure, Semantic Web technologies such as the Virtuoso triplestore, and the OGCE workflow orchestration and execution tools, we are able to process, enhance, store, and query integrated data provided by our collaborators. The triplestore now

contains information about experimental procedures, journal articles and molecular properties of crystal moieties. Different resources are now linked, thus creating a huge RDF graph of Chemistry scholarship. SPARQL queries can now be performed on the web server at [30]. One can now get all the related information about a particular crystal by performing a SPARQL query that fetches all the triples related to the crystal. Also, one can write SPARQL queries to get all the crystal structures that are synthesized using a particular experimental procedure and have a particular moiety molecular property.

IV. Future Work

Currently only the Crystal Eye group from Cambridge University is publishing ORE ATOM feeds with OREChem data. We anticipate adding additional feeds from other OREChem partners.

Our workflow must be manually initiated by a user, and the feed data must be pulled from the Crystal Eye server. Push-based models (or, equivalently, event-driven systems) would be more efficient. One solution for push models in REST and ATOM-based systems is Google's PubSubHubbub [31]. Using PubSubHubbub, the start of the OREChem-computation workflow could be automated. PubSubHubbub is based on a publisher/subscriber protocol in which a publisher is subscribed to a Hub, and also the subscriber is subscribed to the Hub. Hub can be a Google App Engine or a Red Hat Linux Server. As the publisher publishes a new entry, the hub gets the new content and it pushes this new content to the subscriber. This requires either a modification to Crystal Eye so that it acts as a publisher, or else we must develop a proxy intermediary that acts as the publisher.

This study has combined both REST and SOAP based services in a single eScience Workflow. The current case study workflow is orchestrated in the OGCE workflow tools using a heavyweight SOAP Web Service workflow execution. In the future, the OGCE workflow tools will be enhanced to support the simpler REST protocol and accordingly orchestrate the REST and SOAP hybrid workflow in a more efficient manner utilizing REST orchestrations for the REST components in the workflows and utilizing SOAP based standards for SOAP components.

ACKNOWLEDGEMENTS

We thank Microsoft External Research for their funding of this project. Also thanks to all the members of the OREChem Project; Alex Wade, Lee Dirks from Microsoft Research; Carl Lagoze from Cornell University; Na Li, Prasenjit Mitra from Penn State University; Jeremy Frey, Simon Coles, Mark Borkum

from University of Southampton; Peter Murray Rust, Nick Day from University of Cambridge.

REFERENCES

- [1] Triplestore Wiki
<http://en.wikipedia.org/wiki/Triplestore>
- [2] McBride, B. "RDF primer"
<http://www.w3.org/TR/rdf-primer/>, 2004, Feb 10.
- [3] Lagoze, C. V. "ORE User Guide", 2008
<http://www.openarchives.org/ore/1.0/primer>
- [4] Chris Bizer, Richard Cyganiak, Tom Heath, "How to publish linked data on the web",
<http://www4.wiwiw.fu-berlin.de/bizer/pub/LinkedDataTutorial/>, 2007, Jul 27.
- [5] "Turtle primer"
<http://www.w3.org/TeamSubmission/turtle/>
- [6] Lagoze, C. V. "ORE User Guide – Resource Map Implementation in Atom"
<http://www.openarchives.org/ore/1.0/atom>, Oct 2008.
- [7] Microsoft Research OREChem Project
<http://research.microsoft.com/en-us/projects/orechem/>
- [8] ChemXSeer, <http://chemxseer.ist.psu.edu/>
- [9] eCrystals, <http://ecrystals.chem.soton.ac.uk/>
- [10] CrystalEye <http://wwwmm.ch.cam.ac.uk/crystaleye/>
- [11] IUCr's CIF files, <http://www.iucr.org/resources/cif>
- [12] CML Wiki
http://en.wikipedia.org/wiki/Chemical_Markup_Language
- [13] Fielding, R. REST
http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm
- [14] Jersey 1.2
<https://jersey.dev.java.net/nonav/documentation/latest/user-guide.html>
- [15] JSON format, <http://www.json.org/fatfree.html>
- [16] InChI Wiki,
<http://wwwmm.ch.cam.ac.uk/inchifaq/>
- [17] Open Eye Babel,
http://openbabel.org/wiki/Main_Page
- [18] Day, N. Crystal Eye Harvester Bit Bucket Page,
<http://bitbucket.org/ned24/crystaleye-harvesters/src/>
- [19] Peter Murray Rust et al, Semantic Comp Chem,
<http://bitbucket.org/gigadot/semantic-compchem/>
- [20] Young, D. Retrieved from "The absolute beginners guide to gaussian"
<http://www.ccl.net/cca/documents/dyoung/topics-orig/gaussian.html>
- [21] SaxonHE Documentation,
<http://www.saxonica.com/documentation/using-xsl/commandline.html>
- [22] GRDDL Primer
<http://www.w3.org/TR/grddl-primer/>
- [23] Jackrabbit [Online] <http://jackrabbit.apache.org/>
- [24] OREChem Sourceforge
<https://orechem.svn.sourceforge.net/svnroot/orechem>
- [25] Erling, O. "Implementing a SPARQL compliant RDF triplestore using SQL-ORDBMS"
<http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main/VOSRDFWP>

- [26] "SPARQL: query language for RDF"
<http://www.w3.org/TR/rdf-sparql-query/>
- [27] W3C IRI Internationalization
<http://www.w3.org/International/O-URL-and-ident.html>
- [28] OGCE Workflow Suite
<http://www.collab-ogce.org/ogce/index.php/Workflow>
- [29] Abe NCSA [Online]
https://www.teragrid.org/web/user-support/compute_resources?p_p_id=hardwareresources_WAR_userinfoportlet_INSTANCE_v9AO&p_p_lifecycle=1&p_p_state=normal&p_p_mode=view&p_p_col_id=column-1&p_p_col_pos=1&p_p_col_count=4&hardwareresources_WAR_userinfoportlet_INSTANCE_v9AO_id=50
- [30] SPARQL Endpoint [Online]
<http://gf18.ucs.indiana.edu:8890/sparql>
- [31] PubSubHubbub [Online]
<http://code.google.com/p/pubsubhubbub/>

