Chapter 2
Semantic Disclosure in an e-Science Environment

M. Scott Marshall, Marco Roos, Edgar Meij, Sophia Katrenko, Willem Robert van Hage, and Pieter W. Adriaans

Abstract The Virtual Laboratory for e-Science (VL-e) project serves as a backdrop for the ideas described in this chapter. VL-e is a project with academic and industrial partners where e-science has been applied to several domains of scientific research. Adaptive Information Disclosure (AID), a subprogram within VL-e, is a multi-disciplinary group that concentrates expertise in information extraction, machine learning, and Semantic Web – a powerful combination of technologies that can be used to extract and store knowledge in a Semantic Web framework. In this chapter, the authors explain what "semantic disclosure" means and how it is essential to knowledge sharing in e-Science. The authors describe several Semantic Web applications and how they were built using components of the AIDA Toolkit (AID Application Toolkit). The lessons learned and the future of e-Science are also discussed.

2.1 Introduction

2.1.1 Semantic Disclosure

Knowledge discovery lies at the heart of scientific endeavor. The discovery, storage, and maintenance of knowledge form the foundation of scientific progress. Consequently, if we define e-Science as "enhanced Science," then it is essential that e-Science should enhance knowledge discovery and re-use. Ultimately, e-Science is about discovering and sharing knowledge in the form of experimental data, theory-rich vocabularies, and re-usable services that are meaningful to the working scientist. Furthermore, the e-Scientist should be able to work with user interfaces that tap into knowledge repositories to provide familiar terminologies and

M.S. Marshall (✉)
Informatics Institute, University of Amsterdam, Kruislaan 403, Amsterdam, The Netherlands
e-mail: marshall@science.uva.nl

H. Chen et al. (eds.), *Semantic e-Science*, Annals of Information Systems 11,
DOI 10.1007/978-1-4419-5908-9_2,

associations from the domain of inquiry, unencumbered by data schemas, format conversion, or quirky user interfaces. For many scientists, there is the sense that if we could somehow work with conceptual icons and the terms that we are already using to think about a problem, we could more easily manipulate our ideas with the logic and rules of our own definition.

As steadily more organizations build data and knowledge repositories, there is a growing interest in information extraction and knowledge capture technologies. In order to make knowledge discovery possible, we must be able to access and harness existing knowledge. There are vast amounts of knowledge that are digitally available in both publications and on the Web. However, most knowledge is not available in a machine-readable form. The process of knowledge extraction, i.e., text mining from literature can provide us with knowledge distilled from scientific discourse. The same principle can be applied to text documents associated with a given type of data, where the knowledge extracted from the associated texts is used as a *semantic annotation* of the associated data resource. Whether performed manually or accomplished with the method that we've just described, the result is *semantic disclosure*, where meaning about a thing (i.e., resource) is disclosed in a machine-readable statement about it. The mined knowledge and associated data can then be reasoned about by new computational experiments to create new hypotheses and knowledge.

2.1.2 The Semantic Web

The *semantic stack* of the World Wide Web Consortium (W3C) was created in order to provide machine readable and interoperable knowledge exchange. The "semantic stack" is built on a set of standards that handle progressively more specific requirements. At the base is eXtended Markup Language (XML), which has provided a basis for data exchange by providing the representation, schema, and syntax of XML. On top of XML [1], the Resource Description Framework (RDF) provides a way to express statements in "triples" of "subject predicate object."[1] When the subject of one RDF statement unifies with the object of another, the statements connect together to form a graph or "web." The SPARQL Query Language for RDF, a 2008 W3C recommendation, enables query of the RDF graph to look for graph patterns. The modeling language RDF-Schema (RDF-S) provides a basis for hierarchies and subsumption reasoning ("dog isA mammal" and "mammal isA animal" implies "dog isA animal"). The Web Ontology Language (OWL) extends the basic class definitions of RDF to enable reasoning and modeling using description logic. A rule layer tops the semantic stack with the Rule Interchange Format (RIF). Although the semantic stack cannot handle all forms of knowledge, it provides a practical basis for

[1] In the case of *semantic disclosure*, the subject could identify a data or service resource in order to disclose something about it, such as its *dc:creator* ("*dc*" from the Dublin Core standard, see http://dublincore.org/documents/dces/).

interoperable storage, retrieval, and exchange of knowledge. This basis is supported by a variety of implementations, many of them freely available and open source. Additional W3C standards related to RDF are also available, including RDFa for embedding RDF in web pages and Friend of a Friend (FOAF) for social networks.

The Simple Knowledge Organization System (SKOS) for vocabularies makes it possible to relate concepts as "broader" or "narrower" in a way that is intuitive and useful for structured vocabularies. One of the most useful aspects of SKOS is that it enables forward chaining across relations skos:broader and skos:narrower for the induction of hierarchies without the requirements of the more strict logic of OWL. Modeling such relations with OWL properties can result in "ontological overcommitment," for example, where an instance that is assigned a class unintentionally inherits inaccurate properties. The more vague semantics of SKOS can be convenient for modeling "associations" or relations that are not only as well defined as in OWL or RDF-S (or not defined at all yet) but also appropriate for modelling-type hierarchies. We will discuss a few applications of SKOS in the remainder of this chapter.

A common misconception about the Semantic Web is that all knowledge must be first represented in a large ontology and that some sort of large knowledge network on the scale of the Web must exist before anyone can reap the benefits. However, Semantic Web technologies and tools are already being used today to effectively manage and exchange knowledge without requiring practitioners to develop an entire software infrastructure beforehand. As steadily more people follow Linked Open Data principles[2] and learn how to apply the semantic stack, more data is becoming available as interlinked RDF and a Semantic Web is gradually emerging. An excellent introduction to Semantic Web can also be found in the book *Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL* [2].

2.1.3 Making Sense of the Digital Deluge

Many see biomedical science, with its wide spectrum of disciplines and corresponding variety of data, as the ideal proving ground for Semantic Web. Indeed, biologists seem keen to apply the new technologies being developed in the context of e-Science [3]. Understanding the human body, with its many layers of interwoven dynamic molecular systems that interact on subcellular, cellular, tissue, and organ levels may well be one of the most formidable challenges left to science. While research in the life sciences has provided us with additional knowledge about many biological phenomena, many of the mechanisms that are most essential to understanding disease remain mysteries. As an example, the gene for Huntington's Disease was the first genetic disease mapped to a chromosome with DNA polymorphisms in 1983 and isolated in 1993, yet the actual causal chain behind the progression from gene mutation to neurodegeneration is still unknown. Starting in 1991, the Human

[2] http://linkeddata.org/

Genome Project (HGP) sequenced the entire human genome, providing an initial reference sequence of human DNA in 2001. In 1993, around the time the HGP was getting started, the Web began its rapid expansion with the release of the Mosaic Web Browser. Public databases such as the GDB Human Genome Database soon became available on the Web, setting the stage for a new era of data sharing, public data curation, code sharing, bioinformatics, and computational biology. The journal *Nucleic Acids Research* now tracks more than 1,000 publicly accessible molecular biology databases [4].

2.1.4 Data Integration

A logical approach to such an abundance and variety of data is to combine data from several adjacent areas of research and look for patterns in the resulting aggregation. However, researchers hoping that the assembly of the new genome-scale data being amassed would bring new insight have experienced firsthand that the data integration of heterogeneous data is a non-trivial exercise and that scaling up only adds to the problem. Large data archiving projects have experienced similar problems, with researchers struggling to create the right data design and interface in order to avoid creating yet another "massive data graveyard."[3] Translational medicine, an effort to couple the results of fundamental life science research with clinical applications, has become a visible goal of large organizations such as the National Institute of Health (NIH) in the United States. The integration of data from "bench to bedside" (i.e., data from wet laboratory research domain to clinical domain called translational medicine) is a key socio-economic issue for the health care and pharmaceutical industries because it makes maximal use of data across multiple disciplines and enables direct knowledge sharing between disciplines. In order to reach across the boundaries of several disciplines, from life science to drug discovery and virtual screening of compounds, and from drug design to clinical treatment, translational medicine will require the bridging of many terminologies and a strong framework for data integration as its foundation.

2.1.5 W3C Semantic Web for Health Care and Life Sciences Interest Group

Semantic Web offers the means to perform data integration. Indeed, the W3C Semantic Web for Health Care and Life Sciences Interest Group[4] (HCLS IG) proposes that the necessary foundation for translational medicine goals could be provided by a set of practices that make use of W3C Semantic Web standards [5]. The HCLS IG got its start in 2004, when the W3C Workshop on Semantic

[3] David Shotton, University of Oxford.

[4] http://www.w3.org/2001/sw/hcls/

Web for Life Sciences[5] brought a large community of interested parties with 115 participants, resulting in a charter for the HCLS IG in early 2005. Many workshop participants were already performing pioneering research related to Semantic Web for HCLS and banded together in task forces to create technology demonstrations and document them. The HCLS IG was rechartered in 2008 for an additional 3 years to continue its mission to develop, advocate for, and support the use of Semantic Web technologies for biological science, translational medicine, and health care. The group has approximately 100 official participants at the time of writing, with a wide range of participation from industry and academia.

Within the "BioRDF" task force of the HCLS IG, a demonstration of data integration using Semantic Web was built and first shown at a WWW conference in Banff in 2007. The demonstration successfully answered a scientific question about Alzheimer's disease to show the value of being able to query across the data of 15 public databases from the Web. The data was first aggregated into an RDF repository with care to choose an OWL design that was in line with OBO Foundry Methodology. Many researchers contributed significantly to this effort as can be seen from the list of Contributors and Acknowledgements in the HCLS Interest Group Note [6]. Additional work also demonstrated the extension of the knowledge base with SenseLab data [7]. The resulting knowledge base has reached production level in the Neurocommons [8] and continues to be developed by the BioRDF task force, with instances running at DERI Galway and at Free University Berlin.

2.1.6 Semantic Architecture

An e-Science environment brings with it an expanded set of resources and possibilities. The familiar hardware-based set of resources known to system and middleware programmers such as CPU, memory, network bandwidth, input and output devices, and disk space are augmented by "soft" resources such as data, knowledge, and services. In both grid and web environments, service-oriented architecture (SOA) supplies the advantage of data and software components that are readily available on the network for spontaneous incorporation into applications. In fact, the sheer abundance of shared heterogeneous resources can become an impediment to use and the complex tooling necessary to deploy them tends to hinder the uninitiated developer. For an end user, the resources of interest could be things as diverse as journal articles, image data, mass spectrometry data, R scripts, services, workflows, and spreadsheets. How can the user discover and select the resources that are most appropriate to the task at hand? Many factors conflate to make a complex problem of matching requirements, preferences, and policies with the resources at hand. The resource discovery problem is present at many system levels. At the application level, users must discover the knowledge resources (e.g., vocabularies), applications, and parameters that are relevant to their particular task, in their particular application

[5] http://www.w3.org/2004/07/swls-ws.html

domain. At the application development and middleware level, developers must discover services and data, preferably in a way that can be automated, in order to dynamically adjust for variable resource availability and access. In fact, the problem of data and service discovery is common to many computing environments, not only grid but also the Web and large data repositories of many sorts. An e-Science scenario could involve resources from all of the above environments but the challenge remains the same: to manage heterogeneous resources from a single user interface.

2.1.7 The Virtual Laboratory for e-Science Project

The Virtual Laboratory for e-Science[6] (VL-e) is a project with academic and industrial partners where e-science has been applied to several domains of scientific research. Adaptive Information Disclosure[7] (AID), a subprogram within VL-e, is a multi-disciplinary group that concentrates expertise in information extraction, machine learning, and Semantic Web – a powerful combination of technologies that can be used to extract and store knowledge in a Semantic Web framework that enables effective retrieval via both keyword and semantic search. In order to support metadata and knowledge management, AID has created a set of web services as generic components that support the building of applications that are customized to a particular domain. The web services and the applications built around them comprise the AIDA Toolkit (AID Application Toolkit). The AIDA Toolkit and its applications have been developed in cooperation with project partners from several application domains and address a variety of use cases. The AIDA Toolkit has been applied to use cases in bioinformatics, medical imaging, and food informatics during the first 4 years of the VL-e project.

Several AIDA applications have been created that can be executed from four different interfaces: Taverna (workflows), a Taverna plugin, a web interface, and a Java application for accessing grid resources called the VBrowser. The AIDA Toolkit and its applications have been developed in cooperation with project partners from several application domains and address a variety of use cases. In the remaining sections, we will describe our experiences and the lessons learned while designing, building, and applying the AIDA Toolkit to use cases in bioinformatics, medical imaging, and food informatics during the first 4 years of the VL-e project. We will describe how we created a workflow for hypothesis support in biology through information extraction and how this leads to issues in computational experimentation such as the choice of knowledge representation that enables knowledge re-use and knowledge provenance, as well as the need to support semantic types in workflows. We will also discuss the quest for food terminologies that would be useful to our Food Informatics partners and how this finally led to a Web browser interface

[6] http://www.vl-e.nl

[7] http://adaptivedisclosure.org/

with access to customized (Lucene) indexes and multiple terminologies, among them a SKOS translation of a food ontology that was specially developed by the Food Informatics Consortium. The same combination of customizable indexes with structured vocabularies for search has also been applied in a Java application that provides access to grid resources, paving the way to perform semantic retrieval and annotation of grid resources. This Java application can be used by medical imaging researchers to manage image data that is stored and transported on the grid. We also describe how we extended it to semantically annotate and retrieve medical images.

2.2 The AIDA Toolkit

For the purpose of knowledge management, we would like to perform knowledge capture. In the context of a laboratory, knowledge capture can be regarded as the process of collecting and managing related knowledge resources, especially those related to an experiment. In this case, knowledge capture can be directly compared to resource management, where resource aggregation requires a way to associate disparate resource types and the creation of intuitive methods for retrieval. In a laboratory, the resource types can include various types of raw data, such as image data, as well as ontologies and vocabularies for annotation of those resources. In contrast, in a knowledge base, the emphasis is more on the collection of facts and rules than that of data, although links to the evidence on which the knowledge is based are generally desired. Of course, this practical distinction between knowledge capture in a laboratory and a knowledge base should eventually give way to a complete chain of evidence from data and data provenance to distilled fact.

In order to build and maintain a knowledge base, a knowledge engineer needs methods to extract, represent, and manipulate knowledge resources such as facts and rules. Ideally, round-trip knowledge engineering would be possible, where facts in the knowledge base could be directly extracted from the data, and knowledge base maintenance would consist of updating the evidence with new data in order to generate any new facts that would follow from it. The set of web services that we describe here are components of AIDA in a service-oriented architecture that cover the basic functionality necessary to create a knowledge management application. In particular, a user can combine them in a flexible way with other web services providing search, extraction, and annotation functionality.

The AIDA Toolkit is directed at groups of knowledge workers that cooperatively search, annotate, interpret, and enrich large collections of heterogeneous documents from diverse locations. It is a generic set of components that can perform a variety of tasks such as learn new pattern recognition models, perform specialized search on resource collections, and store knowledge in a repository. W3C standards are used to make data accessible and manageable with Semantic Web technologies such as OWL, RDF(S), and SKOS. AIDA is also based on Lucene and Sesame. Most components are available as web services and are open source under an

Apache license. AIDA is composed of three main modules: Storage, Learning, and Search.

2.2.1 Storage – The Metadata Storage Module

AIDA includes components for the storage and processing of ontologies, vocabularies, and other structured metadata in the Storage module (see Annotation, Storage, and Ontology editing in Fig. 2.1). The main component is RepositoryWS, a service wrapper for Sesame[8] – an open source framework for storage, inferencing, and querying of RDF data on which most of this module's implementation is based [9]. ThesaurusRepositoryWS is an extension of RepositoryWS that provides convenient access methods for SKOS thesauri. The Sesame RDF repository offers an HTTP interface and a Java API. In order to be able to integrate Sesame into workflows we created a SOAP service that gives access to the Sesame Java API. We accommodate for extensions to other RDF repositories, such as the HP Jena, Virtuoso, Allegrograph repositories, or future versions of Sesame, by implementing the Factory design pattern. This pattern will allow parallel implementations of the Repository service to coexist.

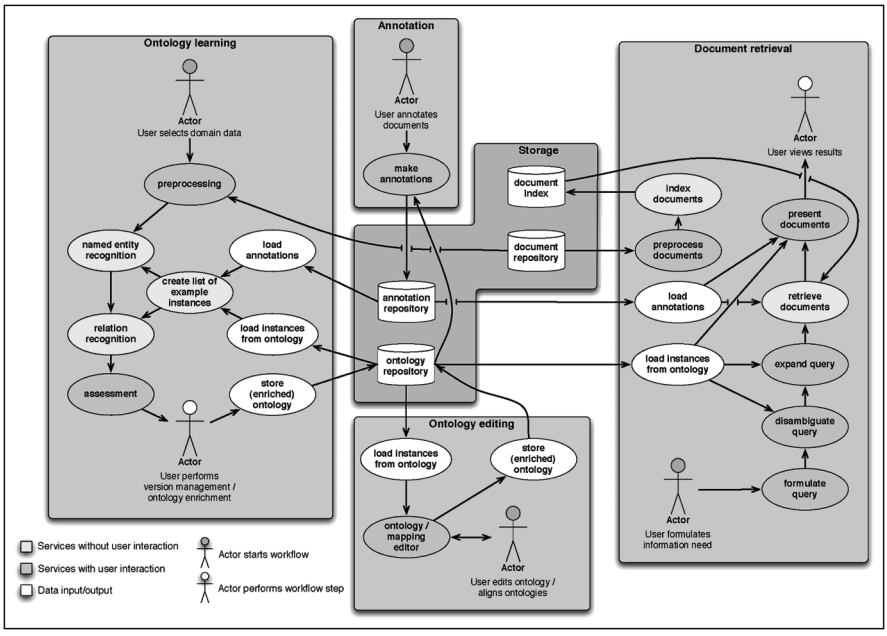

Fig. 2.1 AIDA components can be applied in a variety of activities

[8]Sesame and related RDF software is available from http://openrdf.org

RepositoryWS creates access to operations that enable the manipulation of an RDF repository, but does not contain any specific reasoning facilities for RDF-based knowledge representation languages such as OWL or FOAF. In order to support simplified access to domain vocabularies, we implemented a set of convenience methods for SKOS on top of RepositoryWS called ThesaurusRepositoryWS. SKOS is an RDF-based language for the representation of thesauri. ThesaurusRepositoryWS contains operations that enable querying for terms that represent a concept, their synonyms, broader, narrower, and related concepts, and mappings to concepts in other thesauri. Currently, the most common use of the thesaurus services is for browsing and searching vocabularies that have been stored in the repository. By starting at the top concepts (i.e., the "broadest concepts") of a vocabulary and progressively showing "narrower" concepts, an interactive hierarchical view of the vocabulary is provided in web browsers and the VBrowser application. Two example web service clients that make use of thesaurus operations have also been made: ThesaurusSearch for matching strings to concepts (i.e., searching for concepts in a thesaurus) and ThesaurusBrowser for looking up related concepts (i.e., navigating a thesaurus).

Most RDF manipulation will occur within workflows or applications that access RepositoryWS or ThesaurusRepositoryWS. Because most of our applications require user interaction, several examples of user interactions have been made available in AIDA clients such as HTML web forms, AJAX web applications, and a Firefox toolbar. The clients access RepositoryWS for querying RDF through the provided Java Servlets. An RDF web page demonstrates how to access web service clients from HTML forms. A combination of AJAX code and Java Servlets was used to create a web-based AIDA Thesaurus Browser. The AIDA Thesaurus Browser was used to create Recall samples for the Ontology Alignment Evaluation Initiative.[9] Another example can be found in the XUL Firefox extension for the annotation of web pages, that also access RepositoryWS through Java Servlets. Another example annotation client, similar to the Firefox extension, was implemented as an interactive web page that demonstrates auto-completion on labels of RDF Classes and Properties.

In HCLS IG work on federation of knowledge bases, we added several features to the Storage module. Our applications can now automatically detect and connect to a repository that is either Sesame, Virtuoso, or AllegroGraph. What was originally the Thesaurus Browser in our client user interface has now become the Repository Browser, because it can browse OWL hierarchies, as well as SKOS. Currently, autodetection also works to determine the contents of the repository: the SKOS specification from 2004 is attempted, followed by SKOS 2008, OWL classes, and a number of other specialized patterns. It is also possible to supply the entity types and relations as analogs to the SKOS:Concept and SKOS:narrower that were originally used in the ThesaurusRepositoryWS in functionality now called the *SKOS Lense*. This enables the user to customize the hierarchical browsing

[9] http://www.few.vu.nl/~wrvhage/oaei2007/food.html

to specialized types of RDF. The new functionality made it possible to easily access a wide variety of RDF from several different repository types by simply entering a URL to the repository host and choosing a repository or named graph.

The web services in Storage have recently been updated from the Sesame 1.2 Java API to the Sesame 2.0 Java API. Some of the new features that Sesame 2.0 provides, such as SPARQL support and named graphs, have been added to our web service API's and incorporated into our applications.

2.2.2 Learning – The Machine Learning Module

AIDA includes several components which enable information extraction from text data in the Learning module. These components are referred to as learning tools. The large community working on the information extraction task has already produced numerous data sets and tools to work with them. To be able to use existing solutions, we incorporated some of the models trained on the large corpora into the named entity recognition web service NERecognizerService. These models are provided by LingPipe [10] and range from the very general named entity recognition (detecting locations, person, and organization names) to the specific models in the biomedical field created to recognize protein names and other bio-entities. We specified several options for input/output, which give us an opportunity to work with either text data or the output of the search engine Lucene. The latter scenario is beneficial for a user who intends first to retrieve documents of his interest and then to zoom into pieces of text which are more specific. Output can be presented as a list of named entities or as the annotated sentences.

However, such solutions may not comply with the users' needs to detect named entities in domains other than the biomedical domain. To address this problem, we offer LearnModel web service whose aim is to produce a model given the annotated text data. A model is based on the contextual information and use learning methods provided by Weka [11] libraries. Once such a model is created, it can be used by the TestModel web service to annotate texts in the same domain. Splitting the entire process in two parts is useful from several perspectives. First of all, to annotate texts (i.e., to use TestModel), it is not necessary for a user to apply his own model. Given a large collection of already created models, he can compare them based on the 10-fold cross-validation performance. Another attractive option for creating models is to use sequential models, such as conditional random fields (CRFs), which have gained increasing popularity in the past few years. Although hidden Markov models (HMM) have often been used for labeling sequences, CRFs have an advantage over them because of their ability to relax the independence assumption by defining a conditional probability distribution over label sequences given an observation sequence. We used CRFs to detect named entities in several domains like acids of various lengths in the food informatics field or protein names in the biomedical field [12].

Named entity recognition constitutes only one subtask in information extraction. Relation extraction can be viewed as the logical next step after the named entity recognition is carried out [13]. This task can be decomposed into the detection of named entities, followed by the verification of a given relation among them. For example, given extracted protein names, it should be possible to infer whether there is any interaction between two proteins. This task is accomplished by the RelationLearner web service. It uses an annotated corpus of relations to induce a model, which consequently can be applied to the test data with already detected named entities. The RelationLearner focuses on extraction of binary relations given the sentential context. Its output is a list of the named entities pairs, where the given relation holds.

The other relevant area for information extraction is detection of the collocations (or n-grams in the broader sense). This functionality is provided by the CollocationService which, given a folder with text documents, outputs the *n*-grams of the desired frequency and length.

2.2.3 Search – The Information Retrieval Module

AIDA provides components which enable the indexing of text documents in various formats, as well as the subsequent retrieval given a query, similar to popular search engines such as Google, Yahoo!, or PubMed. The Indexer and Search components are both built upon Apache Lucene, version 2.1.0 (http://lucene.apache.org). We have chosen to extend this particular open source software suite for our information retrieval components because of the long-standing history, as well as very active user/developer community. This also means that indexes or other systems based on Lucene can easily be integrated with AIDA.

Before any document set can be made searchable, it needs to be processed – a procedure known as indexing. AIDA's Indexer component takes care of the preprocessing (the conversion, tokenization, and possibly normalization) of the text of each document as well as the subsequent index generation. It is flexible and can be easily configured through a configuration file. For example, different fields can be extracted from each document type, such as title, document name, authors, or the entire contents.

The currently supported document encodings are Microsoft Word, Portable Document Format (PDF), MedLine, and plain text. The so-called DocumentHandlers which handle the actual conversion of each source file are loaded at runtime, so a handler for any other proprietary document encoding can be created and used instantly. Because Lucene is used as a basis, there is a plethora of options and/or languages available for stemming, tokenization, normalization, or stop word removal which may all be set on a per-field, per-document type, or per-index basis using the configuration file.

An index can currently be constructed using either the command-line, a SOAP web service (with the limitation of 1 document per call), or using the Taverna

plugin. Once an index is created it can be searched through, using the AIDA Search component. There are three distinct ways of interacting with an index: (i) through a SOAP web service, (ii) using AJAX tools (based on JSON objects), or (iii) through a web interface (made using the ExtJS framework[10]). All methods use and "understand" Lucene's query syntax.

The Search SOAP web service (org.vle.aid.lucene.SearcherWS) can handle two kinds of queries, which either search through a single (document) field (called "search") or through multiple fields at the same time (called "searchMFquery"). The operation named "searchContent" is a convenience method which searches the content field by default, thus eliminating one parameter.

The Search web interface uses the JSON search operation, called "searchJason." Since AJAX cannot handle SOAP messages, there is a servlet which bridges the gap between the SOAP web service and the AJAX/JSON: org.vle.aid.client.jason. Additionally, the web interface can display a thesaurus, loaded through AIDA's Storage components. This thesaurus "view" may then be used to look up terms, synonyms, broader and narrower terms, and to perform interactive query expansion. The generality of using JSON for searching is clearly demonstrated by the fact that the output of this servlet can be used directly in Web 2.0 tools, such as Yahoo! Pipes or MIT's Exhibit.

2.3 Applications of Adaptive Information Disclosure

2.3.1 Food Informatics – Adaptive Information Disclosure Collaboration

The collaboration between the Adaptive Information Disclosure subprogram (middleware layer) and the Food Science subprogram (application layer) began early in the VL-e project in 2004, with regular meetings. AID had members from two universities in Amsterdam (University of Amsterdam and the Free University) and TNO, a national research institute. The Food Science partners include both industry and academia, with members at Wageningen UR, TI Food and Nutrition (TIFN), TNO Quality of Life, Unilever, and Friesland Foods. Member organizations of the collaboration were located in diverse remote locations several hours of travel apart, so there was substantial motivation to find ways of collaborating remotely. Of course, the use of e-mail served to enhance initial communications, with an eventual mailing list and archive devoted to the collaboration. Initial efforts focused on defining the possibilities and applications of the machine learning, information retrieval, and Semantic Web to Food Science.

Multi-disciplinary collaboration is a formidable challenge, even when the collaboration is between seemingly closely related disciplines such as the groups of machine learning, information retrieval, and Semantic Web that are represented

[10] http://www.extjs.com/

within AID. Different terminologies and approaches to problem solving and software implementation lead to an intensive process of checking intentions and agreements. Of course, this process is not unlike the process that every software engineer goes through when establishing software deliverables. There is not yet an established "mainstream" terminology for verbal discourse about knowledge, despite ample history provided by the fields of philosophy, logic, and artificial intelligence. So, for example, a knowledge engineer might use the word *metadata* to refer to semantic as well as syntactic-type information about data, whereas a database engineer will typically understand *metadata* to refer to table structure and syntactic-type information.

It is generally difficult to match the problems and tasks of an application domain with the capabilities provided by a new technology. Some of the difficulty is due to the knowledge gap between middleware developers and the users from a particular application domain (we will call it the *middleware gap*). The users find it difficult to understand what the possible applications are of a piece of middleware. On the other hand, the middleware developers do not usually know enough about the target domain to explain the possible applications of their middleware to the problems of that domain. The only way to bridge the middleware gap is either for the domain experts to become experts in the middleware or for the middleware developers to become knowledgeable about the application domain. Although no one is obligated to bridge the gap, it must be done in order to arrive at practical solutions.

With the goal of the collaboration being to use Semantic Web to effectively disclose information within and between the collaborating organizations, several areas of focus were identified. Establishing the vocabularies of a given research domain is an essential first step in defining the terms of discourse and interaction. In many domains, including biomedical research, a number of vocabularies had been established before Semantic Web standards reached Recommended status but are now available in the SKOS format. In order to take advantage of such vocabularies, we provide centralized access to important vocabularies such as AGROVOC, NALT, MESH, and GEMET via the AIDA thesaurus web services. However, although some existing agricultural vocabularies were related to Food Science, the development of structured vocabularies and ontologies specific to Food Science tasks would also be necessary. Document collections and query logs were also identified that could serve as potential sources for the extraction of customized vocabularies. Related work led to a knowledge acquisition method called Rapid Ontology Construction (ROC) [14] as well as an ontology of units and measures.[11]

A few areas of Food Science research serve as application use cases for the technology developed in the collaboration. One research area is centered on the study of bitterness and the conditions (ingredients, processing, etc.) in which it arises. Bitter perception affects the enjoyment of many foods and can even serve as an indicator of toxicity. A database of bitter compounds was developed at Unilever called BitterBase and web services were developed that could use information about

[11] http://www.atoapps.nl/foodinformatics/NewsItem.asp?NID=18

a food component or molecule to predict its bitterness [15, 16]. The BitterBase web services were combined in a Taverna workflow to create a demonstration. The BitterBase web services can eventually serve as a link to knowledge about chemical compounds in other applications. Another area of interest is Food Safety. Incidents where toxic chemicals have entered the food-supply chain have resulted in huge losses for the food industry and reduced consumer confidence in food. The Early Warning System of TNO is being developed for the early detection of food safety risks in the agro-food supply chain. Having made an ontology for food, our Food Science partners were able to browse the OWL class hierarchy of the ontology after we converted it to SKOS and created a web client that accessed it via the AIDA thesaurus web services.

In interactive AIDA applications, the thesaurus services are employed to create a hierarchical browser of the terms in a structured vocabulary, allowing the user to navigate from the most general concepts down to the most specific with mouse clicks. Search is also available to find concepts whose labels match string patterns. These interactions make it possible to navigate large vocabularies, as well as the concepts of a custom-built food ontology. In the case of the ontology, rdfs:subClassOf in the ontology is mapped to skos:narrowMatch in a SKOS version of the subclass hierarchy. Such a mapping can be used to convert OWL to SKOS using a SeRQL Construct query.

An important application of vocabularies is query expansion and refinement. In principle, adding a "narrower" term to a query does not change the intention of the query. In fact, it should improve recall when we are searching for the presence of the query terms in the documents such as is done in Lucene. SKOS allows us to look up terms that are "narrower" than a given term programmatically, so that we can automatically perform query expansion based on those terms when we find our query terms in the vocabulary. For example, if our query contains the word "bread," and we find the narrower terms "ciabatta" and "bread crumbs," we can automatically add both terms to the query. Our ThesaurusRepositoryWS offers this capability in the method "getNarrowerTerms." This functionality was incorporated into the research management system called Tiffany, developed in a collaboration between TIFN and Wageningen UR and used at TIFN. The AIDA web interface that is used in TNO's Early Warning System incorporates narrower terms with a Query Builder and search of a Lucene index. The narrower terms are used in a new way: the drag of a concept to the Query Builder recursively adds all narrower terms "below" it, representing the concept with as many terms as possible within the given vocabulary. In the example shown in Fig. 2.2, the resulting search is for a long list of compound names in the index.

2.3.2 A Metadata Management Approach to fMRI Data

The data from medical imaging experiments brings with it a fundamental problem: the system in which the data is stored can make it difficult or impossible to search

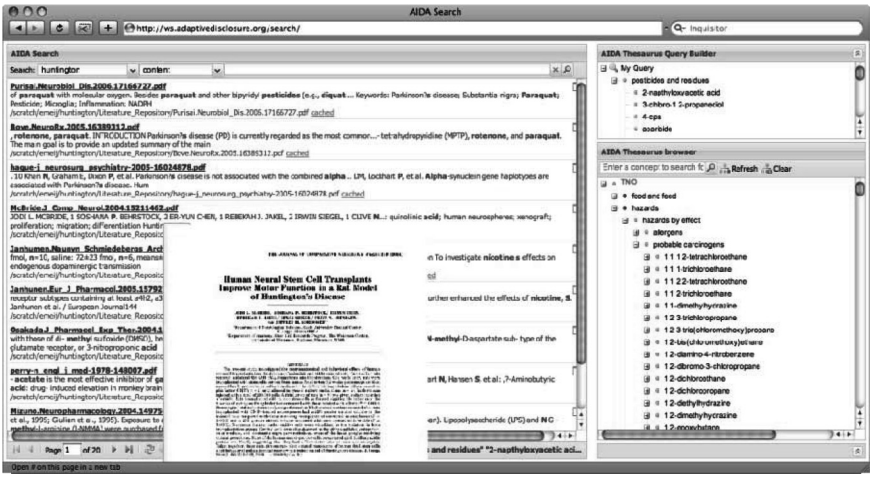

Fig. 2.2 Concept search (*upper right*) for "pesticides and residues" (dragged from *lower right*) using SKOS relations for query expansion to search Huntington's disease corpus (with open Document Viewer for one result document)

for data with a particular set of attributes. However, many types of analysis require this type of search. In medical imaging, different data sets can result from different acquisition protocols, subjects, studies, etc. Processed data can also result from different image analysis workflows and typically include intermediate and final results obtained from different algorithms or parameter settings. In this section, we describe our approach to the management of functional Magnetic Resonance Imaging (fMRI) data using a metadata management plugin for the VBrowser.

Functional MRI (fMRI)[17] enables the non-invasive study of brain activation by acquiring images while the subject is performing some physical or cognitive activity in response to controlled stimulation. The raw data consists of functional and anatomical images, stimulus data and other signals, which are submitted to a complex analysis workflow to compute Brain Activation Maps (BAMs). An ongoing study at the AMC has adopted grid technology to investigate the effect of acquisition and analysis parameters on the resulting BAMs (Virtual Lab for fMRI[12]). This study generates a large amount of data that needs to be properly annotated to facilitate retrieval for result interpretation, preparation of publications, and sharing with other researchers.

The Virtual Resource Browser (VBrowser) is a user interface that provides access to data resources on the grid [18]. The adoption of grid technology makes the VBrowser an attractive interface choice because it provides support for basic tasks such as direct data manipulation and transfer (view, delete, move, etc.) using grid protocols such as SDSC Storage Resource Broker, gridFTP, and gLite Logical

[12] www.science.uva.nl/∼silvia/vlfmri

File Catalog. Moreover, the VBrowser is extensible via plugins that implement, for example, access to grid and web services. The VBrowser is the primary front-end to the Virtual Lab for fMRI.

A set of web services from the AIDA Toolkit were incorporated as a VBrowser plugin that makes it possible to annotate and retrieve grid resources with RDF. For the fMRI application, we have created a metadata schema in OWL that is used for annotation. This allows users and programs to annotate a given fMRI image or experiment result with associated parameters and their values. Using another VBrowser plugin, the user can initiate and monitor experiments that perform large-scale fMRI analysis on the grid. In these experiments, a parameter sweep is performed across the values and the result obtained with each parameter combination is annotated with the corresponding values. At a later stage, users can retrieve results based on concepts available as knowledge resources in the AIDA plugin for VBrowser. The adoption of concepts associated with the parameters of interest in the study enhances usability by enabling the user to express queries in more familiar terms. The query results are presented as a list that can be browsed directly on the VBrowser. Query results are stored as permanent resources that can be reused, refined, and shared among researchers involved in related studies. Besides the ability to query, our metadata approach also allows for the addition of concepts and terminologies from other domains, making it possible to select images based on concepts that are more directly related to the subject of study (e.g., area of brain, type of neuron, type of activity, disease) as well as image features and image quality (Fig. 2.3).

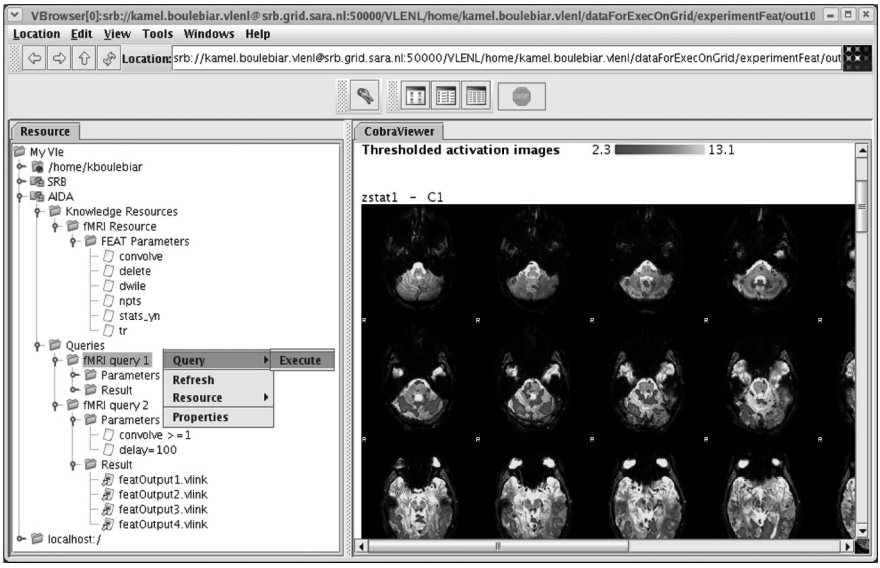

Fig. 2.3 The *right-hand* pane shows the view of a query result. The *left-hand* pane shows fMRI analysis parameters as knowledge resources that can be used in queries. Queries can be edited, saved, and executed from the Queries folder of the AIDA resource

2.3.3 Semantic Disclosure in Support of Biological Experimentation

In this section we treat two related biological cases for semantic disclosure. In the first we demonstrate the disclosure of human genome data for cases when we would like to compare different data sets to test a biological hypothesis. In the second we address the case of hypothesis formation itself that has become a formidable task for biologists. In both cases Semantic Web languages and tools help to disclose data and knowledge for use in computational experiments.

2.3.3.1 Application Case 1: Semantic Disclosure of Human Genome Data

Over the last decades it has become increasingly clear that the activity of one gene is regulated within the context of large networks of activities in the cell, including the activities of many other genes. Instead of investigating genes one by one, we are now able to perform genome-wide studies as a result of the Human Genome Project and many of its followers that provided whole genome sequences and genome-associated data such as gene expression profiles or binding locations of various DNA-binding proteins. Typically, data is stored by a data provider in a relational database and users access this data either by downloading it from the provider's web site or by interacting with the provider's web user interface. A typical example that we will use is the UCSC genome browser ([19]; http://genome.ucsc.edu), one of the largest resources of genome data. One way to analyse this data is by comparing selections of data through the visual interface of the provider. The USCS genome browser shows genome data as stacked "tracks," where a track could be a gene expression profile along a chromosome. This type of "visual integration" is of course no longer appropriate when we want to perform a more complicated analysis of several data sets, especially if we want to be able to repeat and rigorously evaluate the analysis. That requires a computational approach. The traditional approach is to download the data sets to a local database and query the tables locally through SQL queries or via specific scripts. In most of these cases, the relational models used by either the provider or the user are knowledge meager. For instance, UCSC tables typically consist of rows of at least four columns, one for the chromosome number, one for a start position on the chromosome, one for an end position, and one for a value. Any meaning beyond the name of the columns is not directly linked with the data. If we want to find out more about what the data means, its biological context or how it was created and by whom, we will have to follow the hyperlinks on the UCSC web site and read the descriptions and the papers that the web pages refer to. Consequently, knowledge that is relevant for a particular data integration experiment is known by the researcher, but not by the computer. We will not be able to use it directly for computational data integration. In this section we show how we can link data to one's own semantic model and how we can use this to achieve semantic data integration. We demonstrate how this allows us to address different data sets through our own concepts and terms. In other words, we disclose the semantics for our data integration experiments.

Integrating Data for a Specific Hypothesis: DNA-Binding Sites of Transcription Factors and Histones

An intrinsic aspect of our approach is the focus on specific hypotheses within a biological research context. Within the broader context of investigating the relationship between how DNA is structured in the cell and transcriptional activity of genes, the question of how histone-binding sites relate to transcription factor-binding sites is of interest (Fig. 2.4). Histones are specific types of proteins that bind DNA and as such are central to packaging long DNA molecules in the nucleus of a cell. They undergo specific chemical modifications that influence their position and binding affinity for DNA, which can have an effect on transcriptional activity. "Transcription factors" are also proteins that can bind DNA, but they typically influence gene expression directly by binding specific sequences near specific genes. Many of these sequences have been identified and localized on human DNA. The interplay between histones and transcription factors is of interest and therefore we would like to be able to query DNA positions of both types of proteins. As mentioned above, we can find the appropriate data in the UCSC genome browser; the ENCODE project provided data for histone-binding tracks and transcription factor-binding tracks [20]. We can view these tracks together for visual inspection, but for any kind of genome-wide analysis this does not suffice. Below we show how we enable ourselves to perform these studies computationally by providing access to these data sets via our own semantic model. The model contains the biological concepts and relationships that we consider relevant for our biological hypothesis, where we assume a correlation exists between the binding of specific modified histones and specific transcription factors (Fig. 2.4).

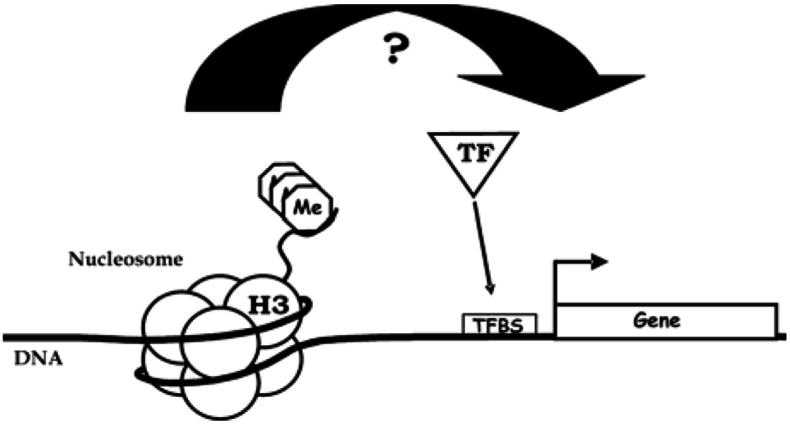

Fig. 2.4 Cartoon model representing a biological hypothesis about the relationship between positions of histones and transcription factors

"myModel" for Experimental Scientists

An important aspect of experiment biology that we would like to take into account is that disagreement among peers is a key factor for scientific progress. Ontologies often aspire to capture common agreement, but semantic models can also be used to capture the view of an individual scientist or research group. Taking this into account, we can follow these steps to integrate data:

1. Create or extend "myModel".
2. Semantic disclosure: link data sets to myModel (Fig. 2.5).
3. Perform integrative analyses using elements from myModel.
4. For new experiments return to Step 1.

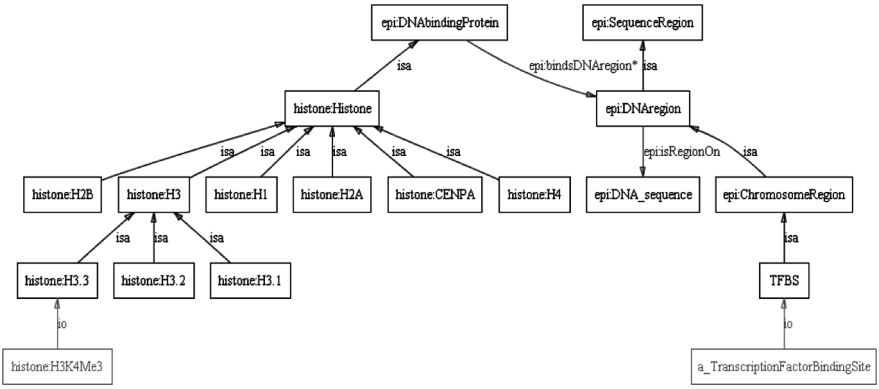

Fig. 2.5 Principle of semantic data integration. The rectangles represent classes in our OWL model: histone classes on the left and transcription factor classes on the right. The bottom two rectangles refer to data items from two different datasets (represented here as OWL instances). The histone model and transcription factor model are linked by biological properties. Therefore, linking data about histones and data about transcription factors to this model creates a biologically meaningful link between the two data sets

In some cases it may be worthwhile to use existing (de facto) standard controlled vocabularies such as the Gene Ontology, but here we follow the scenario where we require a personal model and the ability to make full use of OWL modeling possibilities. At a later stage we can map terms from more comprehensive ontologies to our purpose-built model, if needed. Initially, myModel need not be larger than the next biological analysis requires. We can exploit the extendibility of OWL for each new experiment.

Technical: From Their Data to my Model

Once we have created a semantic model in OWL, how do we link it to the data sets that we wish to query (step 2 in the previous paragraph)? In our example case we have tables of chromosomal binding locations of histones and transcription factors.

We also have the relational schema for these tables from UCSC. However, for linking with our OWL model we require data to be available in RDF or at least have an RDF interface. If the provider does not provide this, then one possible procedure is as follows:

1. Identify the data sets required for the experiment
2. Convert the provider's table schemas or column headers into a RDF Schema (theirDataModel).
3. Convert the data to RDF by linking the values in the data sets to the concepts in theirDataModel
4. Create semantic links between theirDataModel and myModel.

We expect that increasingly more providers will follow the example of UniProt (the main protein data provider) and provide a RDF interface to their data. This would allow us to skip steps 2 and 3. Otherwise, a conversion or some kind of mapping to RDF is inevitable [21–26]. An ideal situation would be if data producers (the wet laboratories), data providers, and data users would each provide their semantic models and their relations to the data (Fig. 2.6).

It is generally advisable to create models for representing the data (preserving the data supplier's naming scheme) and models to represent biological knowledge, with an explicit mapping to link them. We could have imported the raw data directly into myModel, but this would be less flexible and less robust. For instance, changes to our biological models could require an entire new conversion process for any

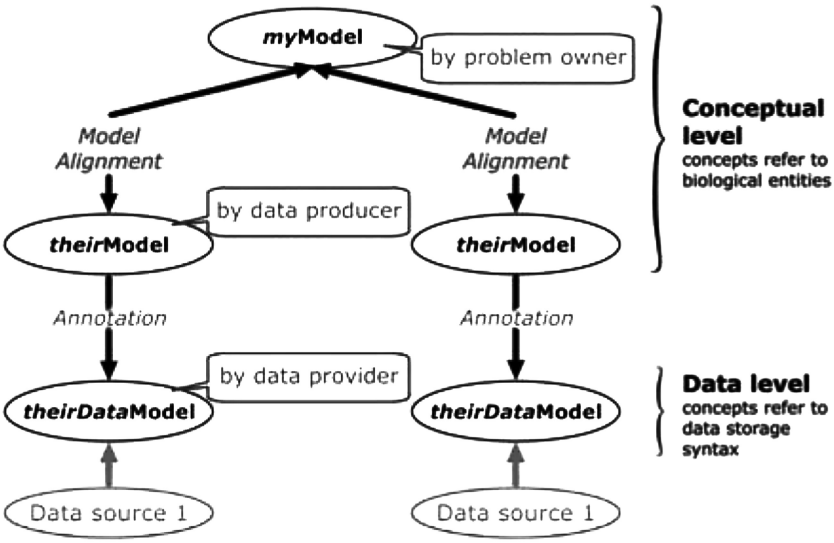

Fig. 2.6 Principle of semantic disclosure in an ideal world. All parties involved provide a semantic model conform their role with respect to the data. Data integration is achieved by aligning semantic models linked to different data sets

data that is affected, whereas we only need to change the mapping in the case of separated models.

Analyzing Semantically Integrated Data

By semantically disclosing data we achieve a situation where we can address separate data sets via biological concepts and relations in our own model. We conclude by an example query (in "pseudo-SeRQL") that retrieves those regions of DNA where both histones and transcription factors bind (see [22–24] for other examples). The "domain of comparison" in this experiment is "ChromosomeRegion." Note that the query does not need to contain direct references to the UCSC data sets.

```
SELECT histone, transcriptionfactor, chrom1, tStart1, tEnd1
FROM {histone_region} myModel:Chromosome_identifier {chrom1};
      myModel:hasStartLocation {tStart1};
      myModel:hasEndLocation {tEnd1};
      myModelExperiment:hasMeasurementValue {score1},
    {histone} myModel:bindsDNAregion {histone_region};
      rdfs:type {myModel:Histone} rdfs:type {owl:Class},
    {tf_region} myModel:Chromosome_identifier {chrom2};
      myModel:hasStartLocation {tStart2};
      myModel:hasEndLocation {tEnd2};
      myModelExperiment:hasMeasurementValue {score2},
    {transcriptionfactor} myModel:bindsDNAregion {tf_region};
      rdfs:type {myModel:TranscriptionFactor} rdfs:type
    {owl:Class},
WHERE chrom1 = chrom2 AND (tStart1 <= tEnd2 AND tEnd1 >=
tStart2)
```

The above query is sometimes called a *Stand-off Join* or *Interval Join*, where the boundaries of two intervals are compared in order to see if there is overlap. This join is frequently necessary when scanning for voice annotations (e.g., reviewer's comments) over a film in a multimedia database. We note that this type of query is challenging for any non-optimized database and certainly also for the semantic repositories we tested some years ago [25]. Performance was generally very poor for all the databases that we tried, including relational databases. However, the XML database MonetDB [27] which is optimized for precisely this type of stand-off join performed better by orders of magnitude. We conclude that considerable performance gains are to be expected considering that semantic repositories are still immature in this respect.

2.3.3.2 Application Case 2: Semantic Disclosure of Biological Knowledge Trapped in Literature

A common way to study intracellular mechanisms in biology is via cartoon models that represent a particular hypothesis. A typical example is a cartoon that represents

a hypothesis about the compaction of DNA, a key factor for sustained regulation of gene expression (top left in Fig. 2.8; see also [28]). Hypotheses can contain many different types of information: sequences, biophysical entities such as proteins and lipids, 3 D structural information, biochemical reactions. They are the basis for each experiment in the laboratory. Given that laboratory experiments are expensive in terms of money and effort, hypothesis generation is an area of interest for bioinformatics and e-Science. Forming a good hypothesis requires the integration of increasingly large amounts of resources. There are over a thousand public databases with experimentally derived data available to biologists and over 17 million biomedical publications are available via the prime knowledge resource for most medical and molecular biologists, Entrez PubMed.[13] PubMed gives access to the National Library of Medicine's public digital library MedLine and several other resources. It is increasingly challenging to ensure that all potentially relevant facts are considered while forming a hypothesis. Support for retrieving relevant information is therefore a general requirement. This leads to the question of how to disclose this information such that it becomes a resource for computer-aided hypothesis generation. Preferably, this process is under the control of a biologist as much as possible, considering that no one else has a better understanding of the end goal: a better biological hypothesis. This presents a problem, because automated information extraction is generally not the area of expertise of a biologist. In this section we show how an e-Science approach based on the application of (AIDA) Web Services, Workflow, and Semantic Web technology enables application scientists to exploit the expertise of scientists from various disciplines for building a machine readable knowledge base as a resource for hypothesis generation.

2.3.3.3 An e-Science Approach for Extracting Knowledge from Text

In order to demonstrate our approach, we will discuss an application in which we would like to extend a hypothesis about condensation and decondensation of chromatin. This is an important determinant of gene expression, because the effects can be sustained over generations of proliferating cells. In particular we would like to investigate putative relationships between the protein "Histone deacetylase 1" (HDAC1), involved in condensation, and other proteins. A traditional scenario would be to query PubMed and browse through the documents it retrieves (over 300 for the query "HDAC1 and Chromatin"). We obtain a "feel" for what is important and read a selection of papers for more in depth information. However, this selection would probably be biased. Extracting the proteins related to HDAC1 without human bias would be at least a highly laborious task. We also have to consider that subsequent experiments based on alternative hypotheses will require new searches that preferably extend our previously obtained information. Therefore, we would like to address this problem computationally.

[13] http://www.ncbi.nlm.nih.gov/pubmed/

Our objectives are to

1. Extract specific knowledge from literature, proteins in the case of our example
2. Examine *all* relevant papers or at least papers selected without subjective bias
3. Store the results in a structured way such that they fit our biological hypothesis and can be re-examined and extended
4. Enable biologists or bioinformaticians to design their own knowledge extraction "experiments"

In line with experimental science, we regard the whole knowledge extraction procedure as a "computational experiment", analogous to a wet laboratory experiment. Such an experiment requires an insightful and re-executable design of which the results are structured enough to allow us to retrace evidence. In the wet laboratory analogy we would use a laboratory journal for the latter.

We can achieve objectives 1–3 by implementing a basic text mining procedure [2, 3] as follows:

(1) Retrieve appropriate documents from Medline (information retrieval)
(2) Extract protein names from their abstracts (information extraction)
(3) Store the results for later inspection. The results have to be linked to our hypothetical model and we need to store the evidence that led to these results. Evidence (provenance) in this case is, for instance, the documents from which protein–protein relationships were derived and the computational resources that were used.

Taking another look at these basic steps we see the desire for a multidisciplinary approach. Step 1 is one of information retrieval, step 2 can be done using machine learning techniques, and for step 3 Semantic Web formats and tools can be used. Each of these steps relates to a distinct scientific discipline. Instead of one bioinformatician reinventing many wheels, an e-Science approach leverages expertise from disparate fields for an application. We achieve this when scientists in these fields produce Web Services as part of their activities and make them publicly available. We call this "collaboration by Web Services." In our case, PhD students in three research groups produced the Web Services and infrastructure that we need for our application (see Section 2.2).

Making use of a service-oriented approach also allows us to meet objective 4. AIDA Web Services and others can be strung together with a tool such as Taverna[14] [29] to form an executable workflow that performs the knowledge extraction procedure (Fig. 2.7). The workflow reflects the basic steps of the text mining procedure and its design can be stored and reused. For instance, we store our workflows on myExperiment.org,[15] a "web2.0" site for computational scientists to store and share

[14] http://www.mygrid.org.uk/tools/taverna/

[15] http://www.myExperiment.org

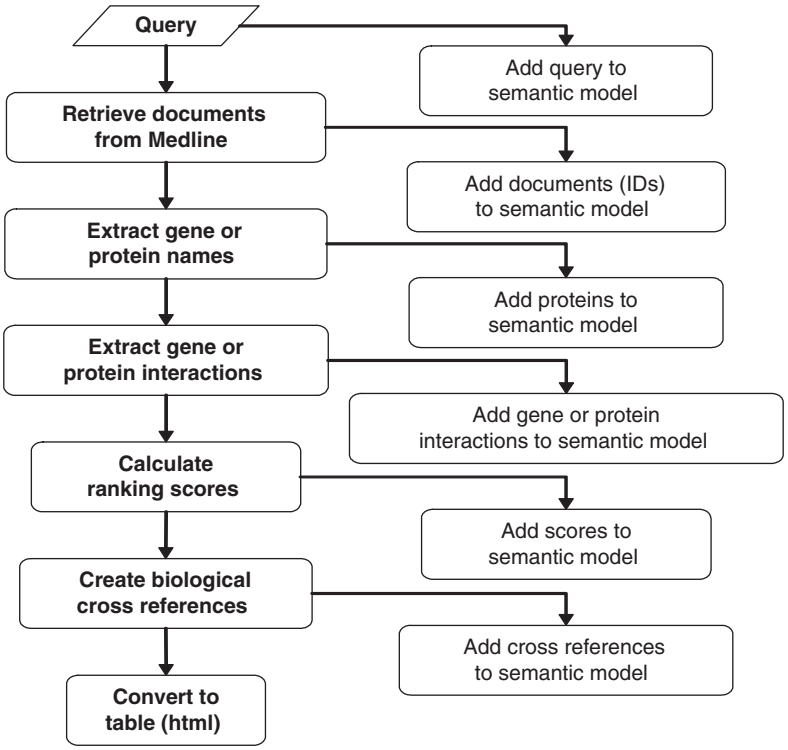

Fig. 2.7 Workflow for extracting proteins from literature (*left*) and store them in a knowledge base (*right*). We added steps to provide a likelihood score, cross-references to some popular biological databases, and tabulated results

(publish) computational artifacts such as workflows. The final step of the procedure, storing results, uses the Web Ontology Language (OWL) and the Resource Description Framework (RDF) to represent the knowledge that we want to extend with the text mining results. The next paragraph explains the knowledge modeling step in more detail.

Modeling for Biological Knowledge Extraction in OWL

Our general approach toward modeling hypotheses for computational experiments is to start with a "proto-ontology" that represents a minimal amount of knowledge appropriate to the problem at hand. In our example case the model should represent at least proteins and artifacts related to the experiment itself to enable us to express, for instance, that a protein was discovered in a particular document from Medline. The purpose of our workflow is to extract knowledge from text and populate this model with individual knowledge instances (e.g., a particular protein). We should consider that our observations from text mining are just pieces of text until

we have interpreted them and converted them into a machine readable form accordingly. For instance, we will interpret the term "p53" found in a particular abstract as an instance with label "p53" of the class "Protein," and its collocation with the term "HDAC1" as a (putative) biological relationship with the protein labeled "HDAC1." Obviously, "collocation" in text does not necessarily mean collocation in the biological sense. To prevent conflation of the biological view and observational views we create four distinct OWL models and one to map between these models (Fig. 2.8):

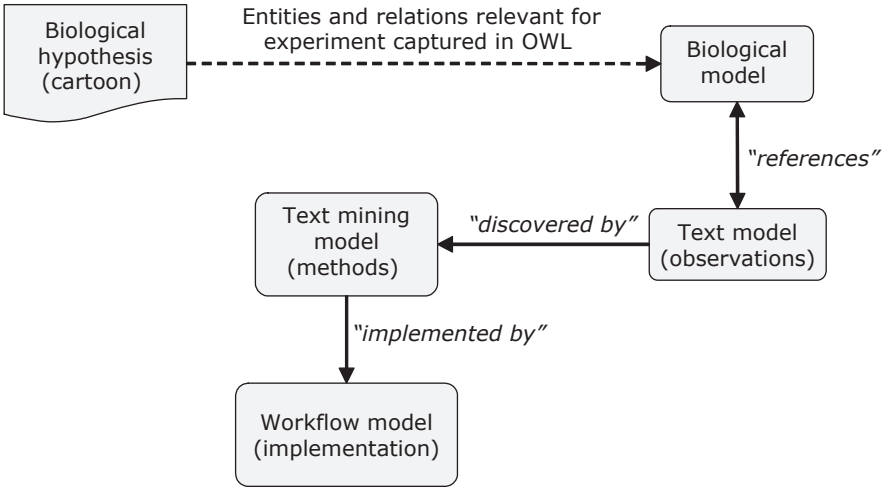

Fig. 2.8 Overview of models and their interrelationships. *Arrows* represent relationships between instances of classes between the models. These relationships are defined in a separate mapping model

1. *Biological model* The biological model is the primary model representing our biological hypothesis. It contains classes such as "Protein," "Interaction" and "Biological model." The model is not extensive, because part of our approach is that we do not define more classes than are necessary for our experiment. OWL allows us to extend the model later as needed by new experiments. We follow the concept of the biologist's cartoon model in that we do not necessarily try to represent real entities or relationships, but hypothetical models of them. Instances in this model are interpretations of certain observations, in our case of text mining results. The evidence for these interpretations is important, but it is not explicitly within the scope of this model.
2. *Text model* The text model contains classes such as "Document," "term," and "interaction assertion." Instances are the concrete results of the knowledge extraction procedure. We can directly inspect documents or pieces of text, in contrast to instances of the biological model such as proteins or DNA. Creating an instance in the document model leads to creating an instance in the biological model based on the assumption that if a protein name is found collocated

with another protein name we assume that the referred-to proteins participate in a biologically meaningful relationship.

3. *Text mining model* The text mining model represents the knowledge extraction process itself. It contains classes for information retrieval, information extraction, and the text mining process as a whole. In principle, these processes could be implemented in different ways. Therefore we created a separate model, in our case a workflow model. These models are linked by "implementation" relationships.
4. *Workflow model* The workflow model represents the computational artifacts that are used to implement the text mining procedure. Example instances are (references to) the AIDA Web Services and runs of these services. Following the properties of these instances we can retrace a particular run of the workflow.
5. *Mapping model* While we have a clear framework for representing our biological hypothesis, text, text mining, and workflow, we also need a way to relate the instances in these models. Therefore we created an additional mapping model that defines the reference properties between the models.

In summary, we have created proto-ontologies that separate the different views associated with a text mining experiment. We can create instances in these models and the relationships between the instances in these views (Fig. 2.9). This allows us to trace the experimental evidence for creating the instances in the biological model.

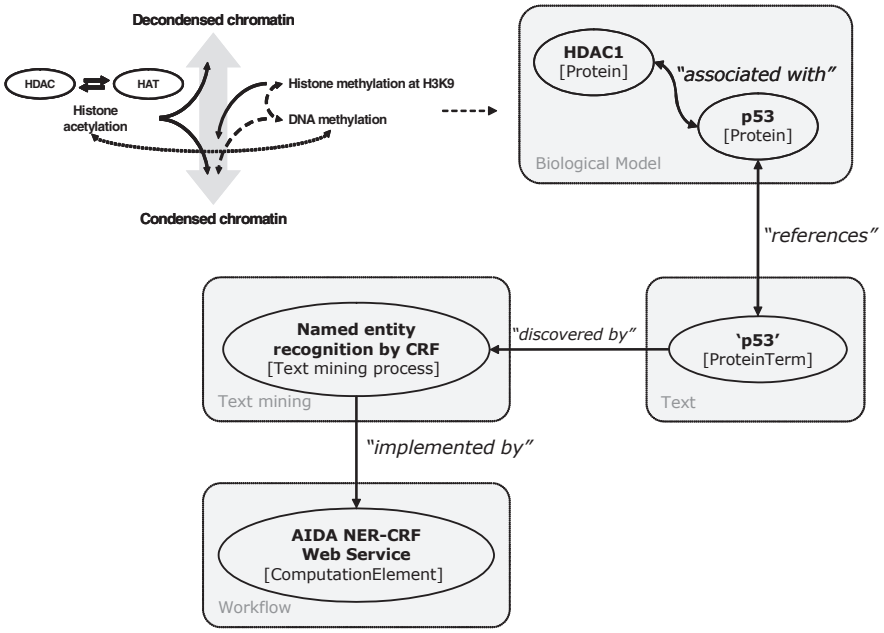

Fig. 2.9 Examples of instances and their relationships between the views associated with a text mining experiment

In our case of text mining, evidence is modeled by the document, text mining, and workflow models. A different type of computational experiment may require other models to represent evidence and new mappings.

We note at this point that in our example case the distinction between proteins and genes presents a problem. They are biologically distinct, but are typically referred to by the same name. There are simple typographic rules to distinguish between gene and protein names (e.g., *ftsQ* versus FtsQ), but these are not always adhered to, are lost in digital copies, or are overlooked by text miners. Therefore, text mining does not make this distinction generally. In our case, we chose to map the text mining results, instances of protein (or gene) names to instances of proteins by default. Alternatively, we could have defined a (biologically awkward) class "gene or protein" in the biological model and map protein (or gene) names in the text model to instances of that class.

A Repository for Storing and Retrieving Biological Knowledge

For our knowledge extraction experiment, we now have a workflow and proto-ontologies for structuring the results of the workflow. For storing this knowledge we use Sesame, a freely available RDF repository. We can add the proto-ontologies and let the workflow populate these ontologies with instances from the text mining procedure (see right side of Fig. 2.7). One of the conveniences of this approach is that by the relatively straightforward action of adding an instance with certain properties, the instance becomes linked with any additional knowledge that was previously added to the repository. For instance, when we add NF-KappaB to the repository and its relationship with a hypothesis about HDAC1, we find that NF-KappaB is also related to a hypothesis about nutrients and chromatin that was used in a previous experiment. Another practical convenience in comparison to relational databases is that referential integrity or preventing redundancy is largely handled by Sesame's built-in RDFS reasoner.[16] With AIDA services we can query and alter the content of a Sesame repository not only from within a workflow but also in a client such as VBrowser, a general purpose resource browser that has been extended by an AIDA plugin. In addition, we can manipulate semantic content by using the Sesame workbench user interface or the Sesame API.

Finally, OWL data can be used over the Internet to create a Semantic Web, because each node and edge of the underlying RDF are referred to by a Universal Resources Identifier (URI). This enables exploitation of powerful features, such as virtual integration of distributed models and data. However, to make OWL models truly part of the Semantic Web, we have to make sure that the URIs resolve properly. A simple way to do this is to ensure that the models are also stored as OWL files on a publicly accessible URL that corresponds to the base of the URIs used in the semantic models, in our case http://rdf.adaptivedisclosure.org/. It is also possible

[16] Sesame supports RDF reasoning for RDF-Schema repositories, not for RDF repositories.

to configure Virtuoso[17] to expose repository contents to URL access in the Linked Open Data tradition. Transparent access to the content of semantic repositories by means of URLs is currently subject to active research and development.

Results

The result of running the workflow is a knowledge base filled with instances of biological concepts, relationships between those instances, and links to instances that can tell us why the instances were created. The instances were classified according to our own proto-ontologies. We can examine the results in search of unexpected findings or we can trail the evidence for certain findings, for instance, by examining the documents in which some protein name was found. An interesting possibility is to explore relationships between the results of one or more computational experiments that added knowledge to the knowledge base. There are a number of ways to explore the knowledge base. We can load the models and instances in Protégé to use its browsing and reasoning features or we can use RDF query languages such as SeRQL and SPARQL. We can also access the models and new instances with our own web interface or Taverna plugin (see Fig. 2.10). It is also possible to perform

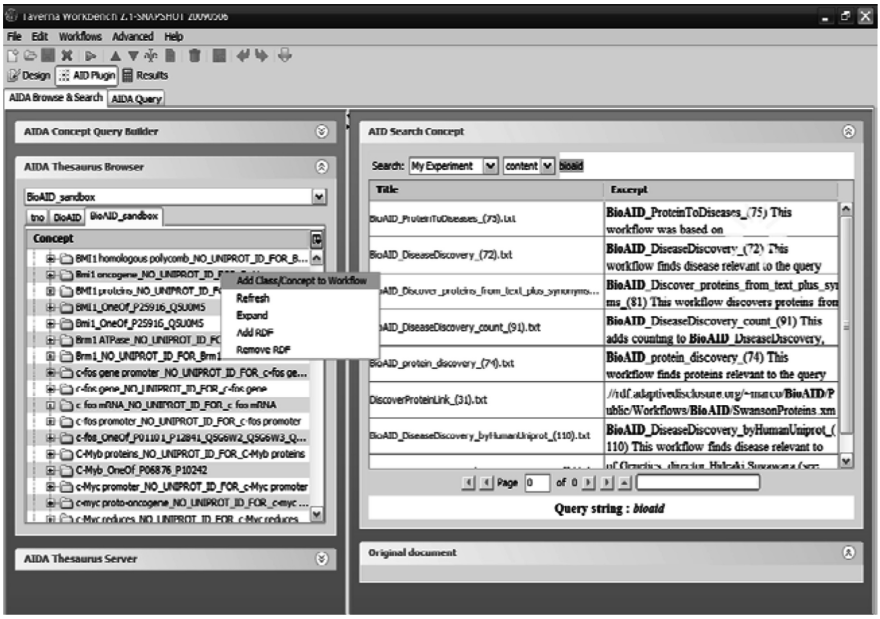

Fig. 2.10 The AIDA Taverna plugin makes it possible to browse and search the results of the workflow without leaving Taverna

[17] http://virtuoso.openlinksw.com/

another computational experiment by means of a workflow using the knowledge base. To illustrate the principle we provide three examples of RDF queries[18]:

1. This query retrieves instances of biological hypothesis models and their partial representation by a user's search query. The query tries to match "{node} *edge* {node}" patterns in the RDF graph. The prefixes refer to our proto-ontologies and standard models such as owl: and rdf:

```
SELECT model, label(query)
FROM {model} rdf:type {bio:BiologicalModel} rdf:type {owl:Class},
{representation} map:partially_represents {model},
{representation} meth:has_query {query}
```

Output:

Instance of biological model	query partially representing the model
http://rdf.adaptivedisclosure.org/owl/BioAID/myModel/Enriched-ontology/BioAID_Instances.owl#BioModel_HDAC1_AND_chromatin	"HDAC1 AND chromatin"

2. The following query retrieves proteins that are shared between two subsequent runs of the workflow with different biological models (hypotheses) as input. We consider an input query of a workflow as a (partial) representation of the hypothesis.

```
SELECT label(comment), label(query1), label(query2)
FROM {protein_instance} rdf:type {bio:Protein} rdf:type {owl:Class},
{protein_instance} rdfs:comment {comment};
      bio:isModelComponentOf {model1};
      bio:isModelComponentOf {model2},
{representation1} map:partially_represents {model1};
      meth:has_query {query1},
{representation2} map:partially_represents {model2};
      meth:has_query {query2}
WHERE model1 = inst:BioModel_HDAC1_AND_chromatin AND
   model1 != model2
```

[18]The examples here are a simplified version of SeRQL; for complete SeRQL examples including namespaces, see http://www.adaptivedisclosure.org/aida/workflows/bioaid-serql-query-examples

```
Output:
```

Protein	Query for model 1	Query for model 2
"protein referred to by as NF-kappaB and UniProt ID: P19838"	"HDAC1 AND chromatin"	"(Nutrician OR food) AND (chromatin OR epigenetics) AND (protein OR proteins)"
"protein referred to by as p21 and UniProt ID: P38936"	"HDAC1 AND chromatin"	"(Nutrician OR food) AND (chromatin OR epigenetics) AND (protein OR proteins)"
"protein referred to by as Bax and UniProt ID: P97436"	"HDAC1 AND chromatin"	"(Nutrician OR food) AND (chromatin OR epigenetics) AND (protein OR proteins)"

3. Finally, a query that retrieves a "trail to evidence" for a protein instance. It retrieves the process by which the name of the protein was found, the service by which the process was implemented, and its creator, the document from MedLine, that is the input for the service and contains the discovered protein name and the time when the service was run. In this case we depict the query as a graph emphasizing that RDF queries are in principle graph patterns that match patterns in the knowledge base (Fig. 2.11).

Overall, using the December 2008 version of our MedLine index the workflow created 257 protein instances linked to our biological model of HDAC1 and chromatin. They were discovered through 489 protein terms found in 276 documents and we could recover by what process, web service, and workflow these were discovered, and when. The discrepancy between the number of protein instances and protein names is because proteins are referred to by various synonyms. As our knowledge base grows with each experiment (not necessarily text mining), we can perform increasingly interesting queries in search of novel relations with respect to our nascent hypothesis. We can store these queries and share them as "canned queries". We are curious to see which queries will turn out to be biologically most revealing.

Semantic Data Integration by Knowledge Extraction Workflows

The text mining workflow above, although created for building a knowledge base to support hypothesis generation, can also be seen as a data integration or data annotation workflow. If we consider documents contained in MedLine as data elements, then we have annotated these data elements by linking them to proteins and several semantic relationships in our proto-ontologies. The semantic annotation is provided by mining the text that is associated with the data. We can now query these data elements via our own concepts and instances (which we may have linked to de facto standard ontologies for extra points of reference), as we could after semantically annotating human genome data from the UCSC genome browser (see elsewhere in this chapter and [25, 26]). One of the advantages of combining workflow and

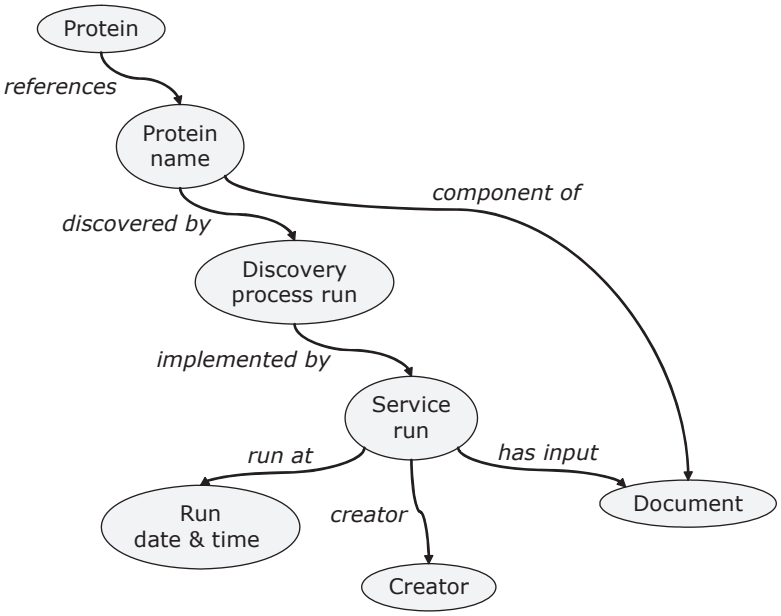

Fig. 2.11 Graph representing a RDF query to retrieve the evidence for a biological instance

Output for Protein identified by "P19838".
Protein: "Protein referred to by as NF-kappaB and UniProt ID: P19838"
Discovered by: "Named entity recognition trained by conditional random fields (CRF) on protein names"
Implemented by: "AIDA CRF Named Entity Recognition service"
Created by: "Sophia Katrenko (University of Amsterdam)"
Input/Component container: Document with PubMed ID 17540846[19]
Timestamp: "2008-11-18T03:29:30+01:00"

Semantic Web technologies in this way is that we can dynamically create a semantic "data warehouse" according to our own needs and wishes. This pattern could be applied whenever data is coupled to some form of free text.

2.3.3.4 Conclusion

In this section we have explored mechanisms for semantic disclosure of human genome data and knowledge enclosed in literature, both in the context of supporting hypothesis-driven experimentation. In the first part of this section we could demonstrate the principle of being able to query (experiment with) data in terms of our own semantic model. Perhaps more importantly, it showed an elegant way to integrate different data sets using Semantic Web formats and tools. Extracting knowledge

[19]http://www.ncbi.nlm.nih.gov/sites/entrez?cmd=Retrieve&db=PubMed&list_uids=17540846

from literature is a more challenging task requiring expertise from different fields of science. We showed that by using a workflow we can combine the expertise of several scientists for disclosing information trapped in literature and created a biological knowledge base that can be explored by semantic queries in search of biological hypotheses. The pattern of storing (interpretations of) results from a workflow into a knowledge base can be used repeatedly to build an increasingly rich resource for elucidating biological phenomena and may also be applied for automated data integration

2.4 Discussion

A number of technologies are implemented in AIDA, each with particular uses and advantages. For example, Sesame RDF repositories enable storage and retrieval of knowledge with Semantic Web technology, creating support for significant portions of the semantic stack. Lucene provides document indexing and retrieval, enabling us to search through document collections from within applications. The ability to create a personalized index using AIDA enables customized document management and search on personal document collections using the same interface as that used for larger public collections such as MedLine. The machine learning techniques that have been applied within AIDA provide application builders with entity recognition for bio-entities such as proteins, as well as relation extraction ("proteinA *interacts with* proteinB") within text. The main bottleneck in training the statistical models that comprise our machine learning components is annotated training sets. This bottleneck points to the conundrum of how to produce annotated training sets without tedious manual annotation (impractical). Ideally, annotation would be performed automatically by employing the very same statistical model that one would like to train. This points to a natural progression of functionality: once you can search (for knowledge resources such as vocabulary terms), you can proceed to annotate, and once you have annotation, you can proceed to (machine) learn. This is where the knowledge resource tooling of AIDA can come in handy: annotators can use AIDA to search a number of repositories for the knowledge resources that they then use to annotate a given word or phrase. The resulting annotations could then be used to train a statistical model, as required for this approach to machine learning.

Although each technology is useful on its own, it is the combination of the aforementioned technologies that make AIDA truly useful. For example, in a well-established approach to text mining, the services that have been created from machine learning algorithms can be used to extract information from documents that have been retrieved from a Lucene index with information retrieval techniques such as query expansion (i.e., adding related terms to the query that increase recall). In a new approach to resource management, a selection of indexes (both personalized and public) can be searched with terms from vocabularies and ontologies that are made available from the RDF repository interface. This makes it possible to create a concept-based query of a personal resource collection, in which it is possible

to switch to other terminologies. Alternative terminologies and ontologies not only provide possible terms for queries but also for annotation.

It is the programmatic access to term-concept mappings (in our case, we make use of labels that are part of the RDF – more elaborate schemes are possible), as well as the ability to create them by annotation that lies at the core of personalized search. The ability to access vocabulary and ontological terms for both search and annotation enables the interface to be customized to the user. When mappings from concepts to multiple query languages have been made, a customizable user interface will become available for the browsing, management, search, and annotation of a wide range of data types and documents. Furthermore, the current machine learning web services could eventually be supplemented with other entity recognizers and types of relation extraction and even other forms of pattern recognition, including those from the field of image processing. This would make it possible to apply the same architecture in order to link concepts to other types of data, such as images, through the features pertinent to that domain that allow us to classify to certain concepts (i.e., tumor).

A few formidable challenges for the Semantic Web remain: end users would like to pose questions without having to know special query languages or know technical specifics such as where the information can be found. Technical solutions to this challenge exist, but will require much more work in the areas of query federation and repository annotation. Once the foundations have been laid, questions that are posed in the terms of (RDF) vocabularies can be translated to the form of a SPARQL query, where the query is automatically decomposed into subqueries that are dispatched to the data sources that contain the relevant data, the answers are assembled and presented to the user in a single application. Such a one-stop-shop will require sophisticated query federation, repository annotation, and software engineering. Some initial work in this area has been reported by the HCLS Interest Group [30].

Another important challenge for Semantic Web is to make it easier for people to use knowledge bases. When a user has access to a knowledge base, even if she is very knowledgeable about SPARQL, she will have to issue a SPARQL query in order to find out what can be matched as a subject, predicate, or object in subsequent queries. The process of discovering which items are available and which properties or predicates refer to them is a tedious and error-prone excercise. An interface that allows users to start from keywords and find the closest related terms in the knowledge base will lower this barrier considerably.

At the core of semantic e-Science is semantic disclosure. In order to enable computational experiments for biology research to be conducted in terms of concepts, with transparent access to workflow and grid resources such as data, knowledge models, and (web)services, we must make the disclosure of semantics and data provenance an essential part of experimental data production. It is our hope that this will convert databases and repositories from “data graveyards” into re-useable data pools, adding value to the data that then serves as evidence linked to the assertions in knowledge models.

How far are we in reaching the goals of semantic e-Science? A new era of data sharing has quietly begun. Enough organizations have opened up their data and

API's, with the XML data format becoming standard practice, that data exchange has become a trivial exercise. Service-Oriented Architectures (SOA) have also made it possible to share expertise in the form of programs as well as their components [31], such as the Taverna (and other) workflows that are shared on myExperiment [32]. BioCatalogue [33] will enable the community that employs these services to look up and report the quality and behavior of the services, as well as share information about how to use them. However, in general, knowledge sharing is still quite exceptional. Although it is simple enough to look up the syntactic type of a given piece of data (i.e., Integer, Float), there is generally no provision for semantic types (i.e., Chromosome Number, Score). This Do-It-Yourself (DIY) approach to semantics means that the handling of semantics is left up to the application developer who is building on the data and services. Until information systems provide the facilities to supply the semantics of a given piece of data, developers will continue to code assumptions about semantics into their programs and applications. Semantic support is especially important for workflow systems, where computational experiments can produce new knowledge but must store the new knowledge with labels that indicate its origins. Without integrated support for semantics in workflow systems, individual users must be motivated and knowledgeable enough to come up with their own ad hoc systems for disclosing metadata and knowledge from workflow components. In this respect we look forward to integrating the semantic approaches described in this chapter with the RDF-based provenance that is being developed for new versions of Taverna [34]. We look further toward an e-Science environment where semantics is just as much a fixed component as data and services, and knowledge about data and services, can be used to distribute jobs across grid nodes and partition data accordingly.

There are distinct benefits to collaborative research provided by the technologies used to build the e-Science applications described in this chapter. The combination of platform independent technologies such as SOAP, WSDL, Java, and Ajax in SOA-based applications has profoundly enhanced our ability to collaborate with colleagues. Typical problems that require communications overhead such as obtaining the latest version of code and compiling have been solved by simple web-based access to web services. We were able to take advantage of remote collaboration via web services in several ways. In the collaboration with Food Scientists, food vocabularies could be served to all interested parties and the latest version of the web interface that made use of those vocabularies via web services could be remotely evaluated by partners and testers without requiring any extra steps. A synonym server was made available to us as a web service by a partner in Erasmus University in Rotterdam and the service was effortlessly incorporated in our text mining workflow. The services available from AIDA have been updated without change to the API, allowing legacy applications to continue functioning.

How does our progress in semantic disclosure by the AIDA toolkit and others relate to high-performance computing, such as enabled by a grid? Web services have not yet been integrated with grid services in such a way that it is straightforward to combine them in an application (as is now possible with web services in many programs such as Taverna, Galaxy). At the time of writing, there is an artificial division

between grid and the Web largely due to data transport and security differences. For example, the default data transport for web services is SOAP, a protocol that wasn't designed for large data volumes or high throughput. Also, there is apparently no simple way yet to proxy grid credentials in order to provide web services access to grid services. The problem of data transport serves as an objection to web services for bioinformatics practitioners who are already processing large data in their programs. We expect that both the data transport and the security technical barriers to web and grid integration will be addressed in the short term. We are already applying one possible approach that has been implemented as a library that provides alternative data transport protocols through proxy. This approach has allowed us to distribute Lucene indexing over DAS[20] cluster nodes, which we expect will eventually speed up our nightly Medline indexing process and other large indexing jobs.

Despite the technical hurdles still to overcome, one of the most prominent challenges for e-Science is not of a technical nature but is related to the gaps in culture between the many fields involved in this multidisciplinary discipline, in particular the gap between application sciences and computer science. We think that it is important that application scientists try new developments in computer science at an early stage, providing feedback from real-life practice while application scientists can be the first to reap the benefits of a new approach. In our experience, however, a computer scientist's proof-of-concept is often not practically usable by scientists from the application domain. The theory might be proven academically by the computer scientist, but not implemented far enough to judge whether it is useful in practice, i.e., in application to a particular domain. In our view, one of the aims of e-Science is to overcome this gap. We believe that it is helpful to anticipate enough software engineers to create the necessary proof-of-concept implementations during the budgeting and planning stage of e-Science projects. We advocate their explicit addition to e-Science projects by making the analogy to wet laboratories, where scientists are typically supported by laboratory assistants who are trained at putting theory to practice. In addition, we have experienced that many computer scientists tend to apply the paradigm of "separation of concerns" to multidisciplinary collaborations, preferring to remain "domain agnostic" with the intent of providing only generic solutions. In our experience, this is often counter-productive. Without substantial understanding of each other's domains there is a risk that the mutual benefit is small or even negative. In fact, it is impossible to demonstrate the benefits of a new technology to a particular domain without carefully fitting it to an appropriate problem or use case. However, when these social factors of e-Science are taken into account, we are convinced that we will see some highly needed breakthroughs in, for instance, life science and health care.

Acknowledgments This work was carried out in the context of the Virtual Laboratory for e-Science project (http://www.vl-e.nl). This project is supported by a BSIK grant from the Dutch Ministry of Education, Culture, and Science (OC&W) and is part of the ICT innovation program of the Ministry of Economic Affairs (EZ). Special thanks go to Bob Herzberger, who made the VL-e

[20] http://www.cs.vu.nl/das3/

project a reality and to Pieter Adriaans for creating and leading AID. We also thank Edgar Meij, Sophia Katrenko, Willem van Hage, Kostas Krommydas, Machiel Jansen, Marten de Rijke, Guus Schreiber, and Frank van Harmelen. Our VL-e Food Informatics partners: Jeen Broekstra, Fred van de Brug, Chide Groenouwe, Lars Hulzebos, Nicole Koenderink, Dirk Out, Hans Peters, Hajo Rijgersberg, Jan Top. Other VL-e colleagues: Piter de Boer, Silvia Olabarriaga, Adam Belloum, Spiros Koulouzis, Kasper van den Berg, Kamel Boulebiar, Tristan Glatard. Martijn Schuemie, Barend Mons, Erik van Mulligen (Erasmus University and Knew Co.). Simone Louisse for careful reading of this document. Thanks to Alan Ruttenberg and Jonathan Rees of Science Commons for supplying the Huntington's corpus. We appreciate the support of many colleagues at NBIC, theW3C HCLS IG, myGrid, myExperiment, and OMII-UK.

References

1. Wang, X., Gorlitsky, R., Almeida, J.S.: From XML to RDF: How semantic web technologies will change the design of 'omic' standards. Nature Biotechnology 23 (2005) 1099–1103
2. Allemang, D., Hendler, J.: Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL Morgan Kaufmann (2008)
3. Stein, L.D.: Towards a cyberinfrastructure for the biological sciences: Progress, visions and challenges. Nature Reviews 9 (2008) 678–688
4. Galperin, M.Y.: The molecular biology database collection: 2008 update. Nucleic Acids Research 36 (2008) D2–D4
5. Ruttenberg, A., Clark, T., Bug, W., Samwald, M., Bodenreider, O., Chen, H., Doherty, D., Forsberg, K., Gao, Y., Kashyap, V., Kinoshita, J., Luciano, J., Marshall, M.S., Ogbuji, C., Rees, J., Stephens, S., Wong, G.T., Wu, E., Zaccagnini, D., Hongsermeier, T., Neumann, E., Herman, I., Cheung, K.H.: Advancing translational research with the semantic web. BMC Bioinformatics 8(3) (2007) S2
6. Marshall, M.S., Prud'hommeaux, E.: A Prototype Knowledge Base for the Life Sciences (W3C Interest Group Note). 2008 (2008)
7. Samwald, M., Cheung, K.: Experiences with the conversion of SenseLab databases to RDF/OWL (W3C Interest Group Note). Vol. 2008 (2008)
8. Ruttenberg, A., Rees, J., Samwald, M., Marshall, M.S.: Life sciences on the semantic web: The neurocommons and beyond. Briefings in Bioinformatics 10 (2009) 193–204
9. Broekstra, J., Kampman, A., van Harmelen, F.: Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. The Semantic Web – ISWC 2002: First International Semantic Web Conference, Vol. 2342/2002. Springer, Berlin, Heidelberg, Sardinia, Italy (2002) 54
10. LingPipe 4.0.0. http://alias-i.com/lingpipe (accessed October 1, 2008)
11. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, San Francisco (2005)
12. Katrenko, S., Adriaans, P.: Using Semi-Supervised Techniques to Detect Gene Mentions. Second BioCreative Challenge Workshop (2007)
13. Katrenko, S., Adriaans, P.: Learning Relations from Biomedical Corpora Using Dependency Trees. KDECB (Knowledge Discovery and Emergent Complexity in BioInformatics), Vol. 4366 (2006)
14. Koenderink, N.J.J.P., Top, J.L., van Vliet, L.J.: Expert-based ontology construction: A case-study in horticulture. In: Proceedings of the 5th TAKMA Workshop at the DEXA Conference (2005) 383–387
15. Rodgers, S., Busch, J., Peters, H., Christ-Hazelhof, E.: Building a tree of knowledge: Analysis of bitter molecules. Chemical Senses 30 (2005) 547–557
16. Rodgers, S., Glen, R.C., Bender, A.: Characterizing bitterness: Identification of key structural features and development of a classification model. Journal of Chemical Information and Modeling 46 (2006) 569–576

17. Smith, S.M.: Overview of fMRI analysis. The British Journal of Radiology 77(2) (2004) S167–S175
18. Olabarriaga, S.D., Boer, P.T.d., Maheshwari, K., Belloum, A., Snel, J.G., Nederveen, A.J., Bouwhuis, M.: Virtual lab for fMRI: bridging the usability gap. In: Proceedings of the 2nd IEEE International Conference on e-Science and Grid Computing, Amsterdam, Netherlands IEEE Computer Society, Los Alamitos, CA (2006)
19. Kent, W.J., Sugnet, C.W., Furey, T.S., Roskin, K.M., Pringle, T.H., Zahler, A.M., Haussler, D.: The human genome browser at UCSC. Genome Res 12 (2002) 996–1006
20. Thomas, D.J., Rosenbloom, K.R., Clawson, H., Hinrichs, A.S., Trumbower, H., Raney, B.J., Karolchik, D., Barber, G.P., Harte, R.A., Hillman-Jackson, J., Kuhn, R.M., Rhead, B.L., Smith, K.E., Thakkapallayil, A., Zweig, A.S., Haussler, D., Kent, W.J.: The ENCODE project at UC Santa Cruz. Nucleic Acids Research 35 (2007) D663–667
21. Belleau, F., Nolin, M.A., Tourigny, N., Rigault, P., Morissette, J.: Bio2RDF: Towards a mashup to build bioinformatics knowledge systems. Journal of Biomedical Informatics 41(5) (2008) 706–716
22. Cheung, K.H., Yip, K.Y., Smith, A., Deknikker, R., Masiar, A., Gerstein, M.: YeastHub: a semantic web use case for integrating data in the life sciences domain. Bioinformatics 21(1) (2005) i85–i96
23. Dhanapalan, L., Chen, J.Y.: A case study of integrating protein interaction data using semantic web technology. International Journal of Bioinformatics Research and Application 3 (2007) 286–302
24. Lam, H.Y., Marenco, L., Shepherd, G.M., Miller, P.L., Cheung, K.H.: Using web ontology language to integrate heterogeneous databases in the neurosciences. AMIA Annual Symposium Proceedings, Washington, DC (2006) 464–468
25. Marshall, M., Post, L., Roos, M., Breit, T.: Using semantic web tools to integrate experimental measurement data on our own terms. On the move to meaningful internet systems 2006: OTM 2006 Workshops (2006) 679–688
26. Post, L.J., Roos, M., Marshall, M.S., van Driel, R., Breit, T.M.: A semantic web approach applied to integrative bioinformatics experimentation: A biological use case with genomics data. Bioinformatics 23 (2007) 3080–3087
27. Boncz, P.A., Kersten, M.L., Manegold, S.: Breaking the memory wall in MonetDB. Commun. ACM 51 (2008) 77–85
28. Verschure, P.J.: Chromosome organization and gene control: It is difficult to see the picture when you are inside the frame. Journal of Cellular Biochemistry 99 (2006) 23–34
29. Hull, D., Wolstencroft, K., Stevens, R., Goble, C., Pocock, M.R., Li, P., Oinn, T.: Taverna: a tool for building and running workflows of services. Nucleic Acids Research 34 (2006) W729–W732
30. Cheung, K.-H., Frost, H.R., Marshall, M.S., Prud'hommeaux, E., Samwald, M., Zhao, J., Paschke, A.: A journey to semantic web query federation in life sciences. BMC Bioinformatics 10 (2009) S10
31. Miyazaki, S., Sugawara, H., Ikeo, K., Gojobori, T., Tateno, Y.: DDBJ in the stream of various biological data. Nucleic Acids Research 32 (2004) D31–34
32. De Roure, D., Goble, C., Stevens, R.: The design and realisation of the myexperiment virtual research environment for social sharing of workflows. Future Generation Computer Systems (2008) 2009 May, 25(5)
33. Goble, C., De Roure, D.: Curating scientific web services and workflows. Educause Review 43 (2008)
34. Missier, P., Belhajjame, K., Zhao, J., Goble, C.: Data lineage model for Taverna workflows with lightweight annotation requirements. IPAW'08, Salt Lake City, Utah (2008)

http://www.springer.com/978-1-4419-5902-7

Semantic e-Science
(Eds.)H. Chen; Y. Wang; K.-H. Cheung
2010, VII, 352 p. 127 illus., Softcover
ISBN: 978-1-4419-5902-7