Improving life sciences information retrieval using semantic web technology

Dennis Quan

Accepted: 5th April 2007; Submitted: 5th February 2007

Abstract

The ability to retrieve relevant information is at the heart of every aspect of research and development in the life sciences industry. Information is often distributed across multiple systems and recorded in a way that makes it difficult to piece together the complete picture. Differences in data formats, naming schemes and network protocols amongst information sources, both public and private, must be overcome, and user interfaces not only need to be able to tap into these diverse information sources but must also assist users in filtering out extraneous information and highlighting the key relationships hidden within an aggregated set of information. The Semantic Web community has made great strides in proposing solutions to these problems, and many efforts are underway to apply Semantic Web techniques to the problem of information retrieval in the life sciences space. This article gives an overview of the principles underlying a Semantic Web-enabled information retrieval system: creating a unified abstraction for knowledge using the RDF semantic network model; designing semantic lenses that extract contextually relevant subsets of information; and assembling semantic lenses into powerful information displays. Furthermore, concrete examples of how these principles can be applied to life science problems including a scenario involving a drug discovery dashboard prototype called BioDash are provided.

Keywords: Semantic Web; RDF; visualization; BioDash; Haystack; Semantic Web Browser; ontology; information retrieval

INTRODUCTION

Information retrieval is a long-standing branch of computer science that studies the identification of information relevant to answering a user’s query. This basic process is essential to life sciences research and development, where therapeutic advances arise from a comprehensive understanding of the interplay of the various biological processes involved in any particular ailment. The information sources being drawn upon include more traditional information retrieval targets such as the research literature as well as data sets more specific to life sciences such as microarray data and pathways.

The World Wide Web represented a huge advance in the accessibility of information retrieval technology. The Web succeeded in large part because it allows users to download information from an ever-broadening range of sources through a single tool: the Web browser. In the days before the Web, users had to jump tediously from one system to another to perform complex retrieval tasks. At present, there is a lot less system-hopping thanks to hyperlinking. Furthermore, modern Web search engines such as Google have turned the Web’s multi-billion-document corpus size—once a source of overwhelming complexity—into a means of improving the quality of retrieval through hyperlink analysis.

However, one area in which the Web is still lacking is in enabling users to consume information in aggregate. Web sites are, for the most part, rigid and one-size-fits-all peepholes into the storehouses of knowledge they expose. Most sites’ style of
navigation is best suited for navigating the information specific to that site. A more task-centric approach to information retrieval is required in order for our ability to consume information on the Web to scale with the growth of the Web itself. The notion of a traditional informatics Web site must be turned on its side to enable this, as the raw data that are needed are often buried in tables, bullet listings or even prose. The characteristics that make Web pages easily consumable for humans, i.e. context-specific page layouts and inspired uses of formatting, are the very things that inhibit machine-automated aggregation, which depends on data being laid out in a consistent, predetermined, 'boring' fashion. Differences in data formats have made true collation of information from multiple web pages hard for humans and nearly impossible for automated aggregators.

Technologies such as RSS [1] and Atom [2] have started to enable automated aggregation and represent the first step towards solving the problem. However, a much more comprehensive effort known as the Semantic Web is underway at the W3C. The idea behind the Semantic Web is to extend the World Wide Web with a standards-based means for encoding and distributing machine-processable information on the Internet [3]. The Web was premised on the idea that human-readable content must be written in a common format (HTML) and made available to Web browsers from content servers through the HTTP protocol. Analogously, the Semantic Web requires that data published to the Web should utilize the RDF format, making it easier for applications other than the data's origin to read and incorporate them. It is this requirement that enables tools to successfully merge data together from heterogeneous sources.

Once the relevant data set is brought together, a context-specific presentation should be produced, and users need to be able to customize this presentation to suit their specific needs. Presentations of aggregated information should go beyond those that are available today and show the information set as being more than just the sum of its parts. For example, when search results from a genomics database query are obtained, rather than seeing a fixed listing of matching genes, it may be more useful to see the results superimposed on a pathway display, where the matching genes are highlighted in a different color, if the user is doing the search in the context of working with a pathway. Examples of this style of presentation will be given later during the discussion of the BioDash prototype.

To better understand the improvements in information retrieval we are after, the article begins with a description of the BioDash prototype drug development dashboard. Then, the three layers of Semantic Web technology key to the vision are described: The RDF model for representing Semantic Web information is motivated and presented. Next, the notion of a semantic lens is proposed as a basic building block for context-specific presentations. Finally, we describe how one can construct powerful information displays built out of semantic lenses.

**BIODASH**

To showcase the capabilities of the Semantic Web in the life sciences space, the W3C Healthcare and Life Sciences Interest Group has developed BioDash, a prototype of a drug development dashboard that demonstrates the principles of Semantic Web information retrieval described in this article [4]. To help motivate and concretize these principles, it is useful to explore BioDash with an example scenario that focuses on the study of the glycogen synthase kinase 3 beta (GSK3b) protein in a drug discovery context (full details can be found in previous work [5]). In this scenario, multiple forms of knowledge, including genomic, pathway, disease and single nucleotide polymorphism (SNP) data, are brought together into useful, aggregated displays through Semantic Web approaches to support the discovery process.

BioDash’s topic view, shown in Figure 1, provides the user with a way to visualize the discovery efforts underway regarding a specific gene. The view is divided into four sections known as semantic lenses: The ‘target overview’ semantic lens shows which chemical entities being considered that target GSK3b. The ‘primary disease’ and ‘alternative diseases’ semantic lenses show basic information from the Web on diseases in which GSK3b has been implicated. Finally, the ‘group members’ semantic lens shows the people involved in the effort, their roles and contact information such as what is available from a corporate directory.

Another important aspect of a gene is the biological pathways in which it participates. This information can be especially useful during drug development, since it can highlight multiple possible intervention points for modulating a key
molecular process. BioDash’s pathway view can be used to navigate pathways encoded according to the BioPAX standard [6]. Figure 2 shows the BioDash rendition of the WNT pathway, in which GSK3b participates. Unlike the topic view, which is composed of several sections intermixing public and private data, the pathway view is a full-screen graph most often showing data from a public database such as BioCyC [7]. Furthermore, the underlying pathway data is filled with tons of minute details, and the pathway view is able to filter out these details to show an easily comprehensible overview of the pathway.

It is worth noting that GSK3b is represented (and even spelled) differently in the topic view and in the pathway view. This is a common phenomenon, given that these views are based on a combination of several private and public information sources, and genes are often given different identifiers by these sources. However, this does not preclude BioDash from further aggregating the WNT pathway information with that from the topic view. If the user drags the GSK3b icon from the topic view into the pathway view, BioDash is able to take advantage of annotations on the two GSK3b objects that indicate a common Uniprot ID to merge them together for the purposes of visualization. (These annotations need not come from the original data sources but can be added in after the fact). The result is shown in Figure 3. The six chemical entities that target GSK3b are superimposed on the pathway view as blue and white squares. Furthermore, an additional insight is uncovered that existed across the two data sets but was not visible from the two independent views: the fact that one of the chemical entities targets an additional gene in the pathway.

Figure 1: BioDash topic view.
Finally, BioDash allows users to search for SNP data associated with genes. With the increased focus on pharmacogenomics, SNP data can highlight which genes may have high levels of variation in a patient population and may influence the decision of which gene to target for the development of a treatment. BioDash is able to display the SNP data in a way relevant to this purpose by inserting purple bars representing individual SNPs next to the genes in the pathway, as shown in Figure 3.

The BioDash prototype is an example of an application based on the Semantic Web architecture. It relies on the Semantic Web data model to allow it to access data from various heterogeneous data sources as if they had come from a single source. The user interface taps into this unified view of these data sources to display diagrams that show the cross-connections that exist within these data. The displays can be customized to adapt to user preferences and slight semantic disparities in the underlying data (as was seen in the topic view-pathway view merging example) and are not constrained by the way in which the underlying data are formatted (as is seen in the pathway view filtering out unnecessary levels of detail). The following sections elaborate on these aspects of Semantic Web architecture.

**RDF AS A UNIVERSAL INFORMATION ABSTRACTION**

As noted earlier, the Semantic Web information retrieval vision is founded on the idea that Web sites expose their information in a machine-consumable fashion. In the life sciences space, one finds many informatics Web sites that are built on top of relational databases. Unfortunately, simply exposing direct interfaces to these relational databases can be quite challenging. Relational databases from different vendors use distinct network protocols, varying dialects of SQL and often incompatible type systems (e.g. different representations for dates and floating-point numbers). Database servers are also

---

**Figure 2:** BioDash pathway view.
not designed to handle the scale of concurrent connections that application servers are—one reason Web sites often have distinct Web and data tiers. Perhaps most importantly, there are semantic differences in the way data are encoded in the relational tables underlying informatics Web sites. There needs to be a way to state that the ‘protein ID’ column used by one site is the same as the ‘UNIPROT’ column of another.

Furthermore, a lot of data significant to life sciences research remain unstructured because it is too hard to impose a structure on them. The difficulties stem from a variety of causes, ranging from lack of clarity in how to model the data to the inconvenience of the administrative overhead of setting up a relational database. As a result, users end up storing their information in much less reusable formats such as Microsoft PowerPoint slide decks and Microsoft Excel spreadsheets, resulting in what Neumann has termed ‘information cul-de-sacs’ [8].

The Semantic Web introduced a data model called the Resource Description Framework (RDF) in an effort to address these issues. As with most elements of Web architecture, new, higher-level functionality tends to be built upon existing, proven infrastructure. Predating RDF is XML, which represents one of the first attempts by the Web community to address issues of data portability [9]. The key idea behind RDF is that by introducing some syntactical simplifications on XML, a number of important capabilities are enabled: (i) personal or domain-specific annotations, classifications, and other forms of knowledge can be added to any application’s data without interfering with its normal function; (ii) information retrieval is made easier, because RDF-enabled Web browsers and search engines can index and extract classification metadata from any RDF file; (iii) arbitrary RDF data files, containing pieces of knowledge from multiple applications, can be easily merged to form a larger whole (information integration); (iv) automated, rules-based processing is possible using off-the-shelf RDF inference engines [10].

The Semantic Web also requires objects that are described by RDF data files, such as gene sequences, research papers or 3D structures, be referred to by universal names in accordance with the Universal Resource Identifier (URI) standard [11], an extension of the original URL system (e.g. http://www.w3.org/). A universal naming scheme simplifies the processing of data from a variety of sources, because the application does not need to have

Figure 3: Target and SNP data incorporated into the pathway view.
specific, hard-coded support for each naming scheme. This allows cross-referencing between data sources to be done implicitly using URIs. Efforts such as the Life Sciences Identifier project [12] are looking at ways to map standard informatics naming schemes into the URI model.

**Statements: the quantum unit of RDF**

The second constraint is that RDF data files are decomposable into fundamental units of information called statements (or triples). A statement is composed of three parts: a subject, a predicate and an object. Here are some examples of statements:

- GSK3b is-type Protein
- GSK3b has-name ‘Glycogen Synthase Kinase 3 beta’
- GSK3b interacts-with betaCatenin

Each of these elements are specified by their URIs (the object could also be a string value), eliminating ambiguity for machine processing. The last statement from the above set could be recorded as follows:

  `<urn:lsid:uniprot.org:uniprot:GSK3b>
   <http://www.w3.org/2005/04/swls/biodash#interactsWith>

By breaking data files into quantum units, applications that ‘see’ RDF statements they do not ‘understand’ can safely ignore them (this is a consequence of the concept of logical monotonicity). Formats that are not based on a quantum unit, such as standard XML formats, have blurred boundaries between units of information, so the complete structure must be understood in advance. Consequently, RDF data can be combined by simple concatenation, with metadata included and additional properties appended, without any need to re-program applications that already read these files.

As is the case with human languages, ‘understanding’ is based on an agreement between the party that utters a statement and the party reading a statement that terms, such as ‘interacts-with’, will have some fixed meaning. At the heart of ‘understanding’ an RDF statement is interpreting the name specified in the predicate slot of the statement. Terms such as ‘interacts-with’ are called RDF properties and are defined within RDF data files known as ontologies, which are designed by domain experts to specify the RDF properties of importance in a given domain. Examples of RDF-based ontologies include the aforementioned BioPAX standard for encoding biological pathway information [6] and the Gene Ontology (GO) [13]. Tools such as Protégé are commonly used to design ontologies for the Semantic Web; more information about ontologies can be found in related work [14].

**Developing applications that use RDF**

A number of Open Source toolkits exist that enable developers to easily read, write and query RDF data sources. The Jena framework from HP [10] is one of the most widely used of such frameworks and provides a Java-based API for managing RDF data. Joseki, also from HP and based on Jena, is an RDF server that enables clients to retrieve and store RDF data over the network [36]. IBM provides a complete Semantic Web application platform called SLRP [15] as well as Open Source LSID libraries for various programming languages [16]. In the commercial space, Oracle offers RDF support in their relational database product [17].

Additionally, since most of the world’s data are not yet expressed in RDF form, it is important that adaptation capabilities exist to bridge today’s data sources into RDF. Much of the world’s data are stored in relational form, and technologies exist today that can programmatically make this information appear as a set of RDF statements even though the information remains stored in a relational database (and not in XML) [18]. The core model of the RDF specification—that information can be represented in terms of statements—is independent of the notion that RDF statements can be recorded in an XML format. Query languages, such as SPARQL, which is being standardized by the W3C [19], provide an abstraction for working with information sources that can expose the information contained within in terms of RDF statements. By basing RDF data access code on SPARQL queries, applications can be agnostic to whether RDF data are expressed as XML or are made virtually available from an existing relational database.

Relational databases also play another critical role in supporting the construction of RDF-based applications. A consequence of the fact that one can surface relational tables as RDF is that relational databases make for a natural technology for implementing RDF data stores. Decades of research into query optimization become immediately applicable,
and queries across data sources—a key motivation for the use of RDF—can be achieved through database federation, whereby the database query engine distributes the query across various data sources in an intelligent fashion. Performance can be tuned using well-understood techniques such as indexing. Furthermore, the major relational database products on the market were designed with scalability in mind. As a result, most of the major RDF toolkits mentioned earlier use relational databases as their underlying data storage mechanism for holding sizeable data sets.

Another important source of data to be considered lies buried in unstructured text on Web pages, and this information can also be incorporated into an RDF model. Frameworks such as IBM’s Unstructured Information Management Architecture (UIMA) [20] provide an Open Source-based platform for the development of text analytics tools such as named entity detectors, natural language translation and question answering systems. UIMA also enables tools to annotate natural language corpora with concepts (e.g. proper nouns, classes, etc.) from Semantic Web ontologies [21].

Haystack, upon which BioDash is based, uses a combination of the techniques described in this section to make RDF available to users. Like a UNIX system, Haystack allows users to ‘mount’ RDF data sources backed by relational stores, RDF/XML files, the LSID protocol and even file systems. The user interface, which is described in the following sections, is designed to aggregate information from these various ‘mounted’ RDF sources.

**CAPTURING CONTEXTUALLY RELEVANT INFORMATION SUBSETS WITH SEMANTIC LENSES**

In the last section we discussed a framework for structuring information to make it susceptible to aggregation. We turn our attention to the problem of presenting aggregated information to the user. The naïve approach to aggregation is to simply take all available information and put it onto one page. This approach may work for small information spaces, but for most life science problems, the naïve approach readily leads to information overload, since much of this information is bound to be extraneous to the task at hand. The key to eliminating extraneous information—and hence addressing information overload—is to make use of knowledge of the task at hand.

A technique that has been explored extensively in the literature is the notion of semantic lenses [22]. Just as an optical lens allows one to focus in on or magnify one particular subset of a picture, a semantic lens is designed to highlight a specific subset of the properties of an object. Examples of semantic lenses include ‘physical properties’ (to highlight properties like boiling point and melting point, for example) and ‘target overview’ (graphically depicting the chemical entities that target the protein in question). The semantic lens concept attacks the problem of task-specific information filtration head on by grouping together pieces of information that are relevant to a specific task. As a result, semantic lenses, like ontologies, require knowledge of the domain to create, since domain practitioners are in the best position to know what is extraneous and what is not. However, once created by domain experts, semantic lenses are easily and readily reusable, and creating useful visualizations, as discussed in the next section, is then reduced to the task of putting together combinations of relevant semantic lenses.

A more precise definition of a semantic lens is a function that takes an object and returns a subset of the available information that is useful in some context. The way this subset extraction procedure is defined can be visual or formulaic in nature, or both. A strictly visual definition is defined like a JavaServer Page [23] in that the logic for extracting relevant data and presenting that data may be tightly coupled into a single script. A strictly formulaic definition tends to be more declaratively specified and is often defined in terms of a parameterized RDF query written in a language such as SPARQL. For example, the Group Members semantic lens of Figure 1 is defined as follows:

Find all statements whose subject is [target of the lens—in this case, the project] and whose predicate is <http://www.w3.org/2005/04/swls/ls-ont#team>, and return the objects of these statements.

A hybrid definition may involve both a declarative RDF query and a hint to the system on how to present the data (e.g. as a table, a list, a graph, etc.). Furthermore, more complex semantic lenses can be built up by grouping simpler semantic lenses together to form group lenses or other composite forms.

Technically speaking, the concept of an RDF property (i.e. a name commonly used in the predicate slot of an RDF statement) could also be
defined in similar terms as the ones we used for defining semantic lenses above. One could imagine creating a property called ‘physical properties’ whose value was a string with the boiling point and melting point together. The distinction between semantic lenses and RDF properties lies in the purpose: The design of an ontology (which includes the definitions of various RDF properties) is driven by a desire to fix a specific, consistent way to represent the information in a specific domain for automated processing. An ontology for family trees would likely either have a property named ‘parent’ or a property named ‘offspring’, but not both, since they are redundant. However, when rendering information to the user, it may be more useful or more intuitive to display people as having parents even if the ontology had fixed the style of representation such that offspring are recorded. With semantic lenses, an additional layer of abstraction is made available to allow for new user-facing representations to be conceived of without needing to change the underlying ontological representation.

Semantic lenses are described in RDF, and there exist a number of semantic lens ontologies. Two popular ontologies are the Haystack ontology [22] and the Fresnel ontology [24]. Haystack’s is more complex, as it contains an embedded HTML-like description language for laying out custom graphical elements, whereas Fresnel, which is based on a simplified version of Haystack’s ontology, is more streamlined, uses Cascading Style Sheets [25] for presentation customizations (e.g. fonts and colors), and is supported across multiple systems such as Longwell [26] and Piggy Bank [27] from the W3C SIMILE project [28]. The BioDash demonstration is built on Haystack, but examples of the BioDash data set being rendered by Piggy Bank are available [29].

Because semantic lenses are described in RDF, they inherit many of the benefits of RDF. RDF descriptions are easily shared between systems and can be included with the ontologies with which they are associated (remember that because of the statement model, merging a set of semantic lens definitions with an ontology data file is straightforward). Users can customize semantic lenses with RDF editing tools such as Haystack and Protégé [14]; in contrast, visualization components written in a traditional programming language such as Java are far harder for end users to customize. Users can also share semantic lenses and the customizations thereof with each other. It is not hard to imagine semantic lens galleries and exchange sites becoming available in the future, especially if a unified semantic lens ontology is standardized.

**ASSEMBLING SEMANTIC LENSES INTO POWERFUL INFORMATION DISPLAYS**

In this section we build on the concepts of the RDF data model and semantic lenses to discuss the overall construction of a Semantic Web-based information retrieval system. The usual information retrieval workflow consists of a query phase, in which a user specifies constraints on the information that is desired, and a display phase, in which the results of a query are presented to the user. These two phases are repeated until the user’s search is fulfilled. There are a number of techniques for enabling the user to enter a query. Keyword-directed search is one of the most popular, commonly employed by search engines. Systems such as TAP [30] have been developed that pair traditional keyword-directed search with a mechanism for locating relevant RDF-encoded data. Natural language question answering has been applied to the Semantic Web space, enabling users to phrase RDF data requests in plain English [31]. There are also life science informatics-specific search interfaces such as BLAST [32].

A degenerate example of a query interface is retrieval based on an identifier, perhaps the most direct query specification method of all. On the Web, when a user knows the URL for a page of interest, he or she can enter that directly into his or her Web browser and be taken to the page. On the Semantic Web, an analogous concept exists in Semantic Web browsers [22]. Users can enter URIs for objects of interest and be shown information on that object. While it is less likely users will remember the URI for a protein than the URL for the *New York Times* homepage, identifier-based retrieval is also at work, under the covers, when users click on hyperlinks—a concept common to both Web browsers and Semantic Web browsers. Hyperlinking is important because it gives the user the ability to initiate a new but related query with a single click.

Once the user’s query is established, the display phase takes over, in which information that is relevant to the query is shown in a way that
facilitates the user’s information search. The richness of the RDF representation makes it possible to display a plethora of related information, but as asserted earlier, concepts such as semantic lenses are essential to keeping information overload to a minimum. Semantic lenses make for natural building blocks for context-specific information displays, as was seen in the BioDash demonstration.

Because semantic lenses are often designed to be context-specific, there needs to be a mechanism for capturing the context of the user. Haystack employs a system in which each type of object can have a set of views associated with it [22]. Views can be used to group semantic lenses together in different styles, such as overviews and graphs, and to target different audiences, such as chemists, biologists and project leads. The user is then free to change the view of an object from within the information retrieval application.

Overviews

One straightforward way to produce a view is to show an overview comprised of multiple semantic lenses. The BioDash topic view and the Haystack screen that is shown when viewing a gene record from GenBank, shown in Figure 4, are good examples of overviews. Both examples show a variety of semantic lenses being used, ranging from simple listings (such as the Pubmed semantic lens) and semantic lens groups (such as the Sequence Summary semantic lens) to graphical lenses (such as the BioDash target overview lens) and custom visualization-based lenses (such as the Sequence lens, which depicts the gene sequence as a series of colored bars).

Graphs

A very different usage of semantic lenses is to construct specialized graphical views. Graphs are commonly used on the Semantic Web to visualize RDF data, since a set of RDF statements can be assembled into a graph by turning statements’ subjects and objects into nodes and predicates into edges connecting the nodes. The key strength of a graphical view is that it emphasizes the relationships between objects. However, this style of visualization
can become burdensome if the graph becomes too busy. Semantic lenses are useful for constructing graphical views because they can be employed to filter out relationships that are not relevant to the task at hand.

The BioDash pathway view provides a good example of the power of semantic lenses. The BioPAX ontology defines pathways as being composed of reaction step objects, and reaction step objects have left-hand-side reactant properties and right-hand-side product properties. The semantic lens used by the BioDash pathway view collapses these intricate structural details into a more straightforward connection between proteins that are involved in the same reaction step. Here, the underlying formulaic relationship defined by the semantic lens is used to project a graph. The WNT pathway shown in Figure 2, if shown without this semantic lens, would have several times the number of nodes and edges visible on the current diagram.

Semantic Web browsers such as Haystack can be used to easily create semantic lens-driven graphical diagrams such as the one seen in BioDash. Haystack allows the user to alter the choice of semantic lenses being used to render a graph on the fly, so radically different presentations can be created on demand for a given data set. (Semantic lenses can also be used to control the display of the objects shown as individual nodes on the graph; this idea is explored in the next subsection). Other examples of semantic lens-driven graphical tools include IsaViz [33], which can use Fresnel lens specifications to render graphs, and the e-science experimental provenance record visualization tool used by the myGrid Project [34], which is based on Haystack.

Collections
An area under exploration where semantic lenses play a useful role is in the problem of looking at multiple objects at once. A common scenario is going through a list of results from a search engine such as Entrez Gene [35]. Individual results are commonly displayed in a table, with some small subset of fields (e.g. gene ID, aliases, chromosome/location) being displayed for each result. To see more detail for any given item, the user can click on the associated hyperlink. However, if the user is determining the relevance of a search result by a field shown only in the detailed view and not by one of the fields shown in the summary, it becomes troublesome to locate the relevant items of interest, as the user must constantly flip back and forth between the search results page and items’ detail pages.

In Haystack, the search results page is an example of a collection view, in which multiple objects are shown, and the list of fields being displayed in each individual result item summary is exposed to the user as a list of semantic lenses. The detailed page that is shown when a user clicks on a hyperlink is an overview, also composed of semantic lenses. To address the above scenario, we are developing prototypes that allow the user to drag a semantic lens from the overview onto the search results collection view, to indicate that an additional semantic lens should be shown in the item summaries.

This principle of applying semantic lenses across displays of multiple objects at once extends beyond traditional report-style listing pages. For example, a data set with a strong temporal component, such as a listing of relevant publications and patents over time, can be shown as a timeline. The entries on the timeline need not be restricted to showing a fixed set of properties such as publication date and title; rather, as with graphical views and listing views, giving the user control over the set of semantic lenses being applied can greatly improve the user’s chances of finding relevant information.

SUMMARY
In this article we discussed how the Semantic Web’s RDF model is a useful abstraction for supporting the distribution and aggregation of information over the Internet and how semantic lenses can enable information retrieval systems to highlight context-relevant information in order to reduce information overload. These two ideas can be used in concert to construct powerful information displays, such as the ones seen in the BioDash prototype. Tools and ontologies exist today—many in Open Source—that can be used to create such displays, and standardization and best practices definition efforts, such as those ongoing in the W3C Simile project and the W3C Healthcare and Life Sciences Interest Group, should promote further adoption of these techniques for improving information retrieval in the life sciences space.
References

19. SPARQL Query Language for RDF. http://www.w3.org/TR/rdf-sparql-query/.