BioPAX: An OWL Early Adopter Perspective

W3C Workshop on Semantic Web for Life Sciences
Position Paper

Authors: Bader GD, Cary MP, Karp P, Luciano JS, Sander C, Zucker J
BioPAX Workgroup (Bader GD, Brauner E, Cary MP, Goldberg R, Hogue C, Karp P, Klein T, Luciano JS, Maltsev N, Marks D, Marland E, Neumann E, Paley S, Pick J, Regev A, Rzhetsky A, Sander C, Schachter V, Shah I, Zucker J)
www.BioPAX.org

INTRODUCTION

The completion of the human genome gives us completeness at the molecular level allowing us to easily determine the DNA sequence for any gene of interest. For the first time, we are able to concentrate on studying how the parts of the cell fit together to form functional units. Connecting molecules to function requires knowledge representation tools that can capture key biological processes. A useful organizing concept in this context is the pathway, which relates biological molecules to each other and to a specific biological process. Many efforts are underway to map pathways. These efforts have produced almost 150 "pathway" databases that capture the molecular interactions involved in biological processes[a]. Pathway databases typically use their own non-standard data model to represent pathway data, making it a challenge to integrate data from multiple sources.

OBJECTIVES AND SCOPE

The main objective of the BioPAX initiative is to develop a data exchange format for biological pathway data that is flexible, extensible, optionally encapsulated, and compatible with other standards. The scope of BioPAX is all pathways relating to cellular and molecular biology. Initially, BioPAX will focus on metabolic, signal transduction, gene regulatory pathways and genetic interactions, as most existing data fall into one of these four categories.

USE CASES

The primary function of the BioPAX data exchange format will be to facilitate data sharing among pathway databases such as aMAZE(7), BIND(1), DIP(11), EcoCyc(5, 6), IntAct(2), KEGG(3), Reactome and WIT(8), and the scientist users of those databases. BioPAX could also facilitate the creation of a shared public repository for pathway data. The desire for such a repository was one of the driving forces behind formation of the BioPAX effort. Another intended use of the BioPAX format is to provide a standard format for future pathway databases and the software tools that must access pathway data.

STATUS

Level 1, Version 1 of BioPAX was released in July 2004. Level 2, which will extend the BioPAX format to cover molecular interaction data, is under active development and targeted for release in early 2005. The features of Level 3, which will extend BioPAX to cover signaling pathways, are currently being planned.

ONTOLOGY

[a] http://www.cbio.mskcc.org/prl/index.php

The BioPAX ontology[b] currently contains 27 classes and 46 properties defined in OWL-DL designed to capture descriptions of metabolic pathways and to be extensible to signal transduction, gene regulatory and genetic pathways. The top level class defines `entity`, which is subclassed to `pathway`, `interaction` and `physicalEntity` classes. A `pathway` instance contains instances of `interaction`, which in turn contain instances of `entity`. `Interaction` and `physicalEntity` classes are then subclassed to describe more specific concepts. This recursive design provides flexibility for representing either detailed or abstract statements about biological processes (pathways). For instance, an enzyme catalyzed biochemical reaction can be described in detail using the `catalysis` and `biochemicalReaction` classes, as would be in a metabolic pathway database, and a general interaction between a small molecule that inhibits a pathway could also be described using the `interaction` class. In this case, less molecular detail is defined, but it is still useful information, especially for drug development.

OWL has been particularly useful for the design of the BioPAX ontology because, among ontology definition languages, it is likely to be the well accepted and supported, given its use of XML and approval by the W3C. OWL-DL also provides almost all of the expressivity we need to represent metabolic pathways in biology.

POSITION

The OWL specification of BioPAX was originally developed using the GKB Ontology Editor(9) and subsequently using Protégé[c]. The BioPAX team favored OWL for its knowledge representation capabilities, despite the more mature toolsets of XML Schema and UML.

As early adopters of OWL, we have enjoyed using and learning about OWL, but we have found a number of issues that we feel will be barriers to more widespread adoption. A large part of it is simply that a lot of tools to support OWL are still immature. No doubt, many of these issues are already on the roadmap of development teams building OWL tools. We detail below some of the issues we've had in using OWL for our ontology project.

OWL (Web Ontology Language)

1. **Tool Support**

   The complexity of OWL necessitates substantial tool support to make it easy to use.

   a. Language support: Other languages, in addition to Java must be supported. Many bioinformatics labs, even large ones, only use Perl or Python. Perl OWL parsing tools must be supported for widespread adoption in the life sciences.

   b. Utility Software:

      i. XML-like tools:

      Tools similar to those available for XML Schema are needed. Before these tools are implemented, very robust automatic conversion to XML Schema would be useful.

      1. XML Spy like editing, visualization
      2. Automatic object to relational mapping (e.g. Apache OJB)
      3. Automatic object code generation in multiple languages, but at least Java, combined with OWL I/O functionality that is specific to an OWL

[b] http://www.biopax.org/release/biopax-level1.owl
[c] http://protege.stanford.edu

defined ontology without the instance data. (Similar to Sun's JAXB or Apache's Castor).
4. XML Schema-like data validators (Validator Light vs. Validator Full) for users who do not use reasoning technology. E.g. validation of strings and validation of domain constraints on properties.

ii. `diff` functionality: Easily finding differences between versions of an OWL file is helpful when building an ontology. Changes to an OWL file made by multiple authors as well as by different versions of ontology editing tools are important to quickly find, but are currently difficult because OWL files do not maintain order of definitions from version to version. Currently we find differences between versions by converting the OWL file to n-triple format using Protégé, loading the output into a spreadsheet, sorting and manually searching for differences, which is time-consuming and error-prone. An alternative to creating a diff tool is to use an ontology development tool that can record the update transactions on the ontology, and report those transactions to an interested user, as is possible with Ocelot(4).

c. Better documentation
   i. Better automatic documentation generation (like that of GKB Editor or XML Spy)
   ii. Expanded tutorials on use of tools for OWL similar in scope to those books/tutorials available for open source Java programming e.g. Wiley's *Open Source Java Programming* (ISBN: 0471463620).
d. Clear best practices
   i. Namespace: Namespace options are not clearly defined. When should a new namespace be defined? Who should define it in the following contexts?
      1. BioPAX documents created by databases for the purpose of distributing their data
      2. BioPAX documents created by databases via a web service in response to a user query
      3. BioPAX documents created by users for posting on their website
   ii. RDF ID: How should RDF IDs be chosen by users in the above situations? The idea of using globally unique RDF IDs currently imposes too much overhead on our users. We currently recommend that RDF IDs are only unique within a document, don't have to be globally unique and can change from document to document. Should LSIDs be used directly as RDF IDs or are better placed as data elements in a property value? We realize this doesn't completely support the semantic web, but we feel that we are forced to do things this way until the technology matures. Later someone can build an agent for BioPAX documents to convert our external references that we store as data to RDF IDs and synonym tables in the context of a semantic web application.
   iii. When and how to use various OWL and RDFS specific tags (e.g. ontology description, version information)
e. More examples. We were unable to find examples of the use of important OWL constructs e.g.

i. Use of `oneOf` in a property restriction. Additionally, we were unable to decide if this OWL construct is OWL DL or OWL Full, since different OWL validators disagreed.
ii. Validator consistency: validators disagree sometimes on OWL Full vs. OWL DL and report different errors and warnings.
iii. Expanded tutorials considering reasoners and instance data. A list of recommended/best of breed/approved tools would be nice.

2. Easier use of specific features in OWL

a. Controlled vocabularies (CVs), either flat, hierarchical or DAG-like (directed acyclic graph), like the Gene Ontology (GO)(10), are commonly used in Bioinformatics applications. Depending on the size of the CV and its complexity as well as that of the containing ontology, different CV representations make sense. Flat controlled vocabularies map well to `oneOf` lists of string literals in OWL. DAG-like ontologies like GO partly map to the value partition design pattern described in the highly useful Protégé OWL tutorial[d]. The problem with the value partition for ontologies like BioPAX, which have few classes and many CV terms, is that it creates high overhead, since many classes are created compared to the number of classes in the main ontology. These classes are never expected to have properties, so why make classes? Classes require that users create instances upon use, which is more difficult than using literals. It would be helpful to have a construct that made use of literals in a hierarchical way to address this usability issue of this particular use case. BioPAX contains a small list of terms in the `CONTROL-TYPE` property of the `control` class that propagates to the `modulation` subclass, which is an example of this.

b. Aliases: It would be helpful to have aliases on properties so that they can be named differently at different levels of the class hierarchy. This issue crops up numerous times in biology. For instance, as classes become more specific down the class hierarchy, names of properties should become more specific as well, since biology usually has generally accepted names for these more specific properties. Specifically, in BioPAX, a `Control` class is present with a subclass called `Conversion`. In `Control`, the property that describes the concept of a controller is called `CONTROLLER`. A biologically recognized name for this in the `Conversion` subclass is `ENZYME`, but we have to maintain `CONTROLLER`. There should be an aliasing feature that is natural to use in these cases. `Rdfs:label` may be the right tool, but it is not clear and current tools don't seem to make good use of it.

LSID (Life Science Identifier)

BioPAX supports the use of `LSID`s as external references for biological entities such as proteins in addition to the database external references commonly used.

1. LSID requires the following services to be useful

a. A set of universal and up-to-date web services for LSID resolution and synonym matching for all existing databases, much like DNS servers.

b. Support for synonyms: Realization that there are many valid synonyms for many biological concepts and availability of a web service that, given an LSID, will provide a list of known synonymous LSIDs. This would solve a major problem in bioinformatics and would drive

[d] http://www.co-ode.org/resources/

adoption of LSIDs. This information could be made available on the semantic web using the OWL `sameAs` element in various OWL documents.

As early adopters, we wish to help make the semantic web technologies more useful in the life sciences community by relaying our experiences, reporting bugs, and suggesting features.

REFERENCES

1. **Bader, G. D., D. Betel, and C. W. Hogue.** 2003. BIND: the Biomolecular Interaction Network Database. Nucleic Acids Res. **31:**248-250.
2. **Hermjakob, H., L. Montecchi-Palazzi, C. Lewington, S. Mudali, S. Kerrien, S. Orchard, M. Vingron, B. Roechert, P. Roepstorff, A. Valencia, H. Margalit, J. Armstrong, A. Bairoch, G. Cesareni, D. Sherman, and R. Apweiler.** 2004. IntAct: an open source molecular interaction database. Nucleic Acids Res **32:**D452-5.
3. **Kanehisa, M., S. Goto, S. Kawashima, Y. Okuno, and M. Hattori.** 2004. The KEGG resource for deciphering the genome. Nucleic Acids Res **32 Database issue:**D277-80.
4. **Karp, P. D., V. K. Chaudhri, and S. M. Paley.** 1999. A collaborative environment for authoring large knowledge bases. Journal of Intelligent Information Systems **13:**155--94.
5. **Karp, P. D., M. Riley, S. M. Paley, and A. Pellegrini-Toole.** 2002. The MetaCyc Database. **30:**59-61.
6. **Karp, P. D., M. Riley, M. Saier, I. T. Paulsen, J. Collado-Vides, S. M. Paley, A. Pellegrini-Toole, C. Bonavides, and S. Gama-Castro.** 2002. The EcoCyc Database. Nucleic Acids Res. **30:**56-58.
7. **Lemer, C., E. Antezana, F. Couche, F. Fays, X. Santolaria, R. Janky, Y. Deville, J. Richelle, and S. J. Wodak.** 2004. The aMAZE LightBench: a web interface to a relational database of cellular processes. Nucleic Acids Res **32 Database issue:**D443-8.
8. **Overbeek, R., N. Larsen, G. D. Pusch, M. D'Souza, E. Selkov, Jr, N. Kyrpides, M. Fonstein, N. Maltsev, and E. Selkov.** 2000. WIT: integrated system for high-throughput genome sequence analysis and metabolic reconstruction. **28:**123-125.
9. **Paley, S. M., J. D. Lowrance, and P. D. Karp.** 1997. A Generic Knowledge-Base Browser and Editor, Proceedings of the 1997 National Conference on Artificial Intelligence, Providence, RI.
10. **The_Gene_Ontology_Consortium.** 2000. Gene ontology: tool for the unification of biology. **25:**25-29.
11. **Xenarios, I., L. Salwinski, X. J. Duan, P. Higney, S. M. Kim, and D. Eisenberg.** 2002. DIP, the Database of Interacting Proteins: a research tool for studying cellular networks of protein interactions. Nucleic Acids Res. **30:**303-305.