

BioPAX – Biological Pathways Exchange Language

Level 1, Version 1.4 Documentation

BioPAX Recommendation Wednesday, April 27, 2005

The BioPAX data exchange format is the joint work of the BioPAX workgroup: Gary D. Bader, Erik Brauner, Michael P. Cary, Robert Goldberg, Chris Hogue, Peter Karp, Teri Klein, Joanne Luciano, Debbie Marks, Natalia Maltsev, Elizabeth Marland, Eric Neumann, Suzanne Paley, John Pick, Aviv Regev, Andrey Rzhetsky, Chris Sander, Vincent Schachter, Imran Shah, Jeremy Zucker.

Copyright © 2005 BioPAX Workgroup. All Rights Reserved

Abstract

At present, there are over 170 Internet-accessible databases that store biological pathway data. Each has its own representation conventions and data access methods. This can create problems for researchers that wish to make use of the data regardless of where and how it is stored.

BioPAX (Biological Pathway Exchange - <http://www.biopax.org>) enables the integration of these resources by defining an open file format specification for the exchange of biological pathway data. By utilizing the BioPAX format, the problem of data integration reduces to a semantic mapping between the data models of each resource and the data model defined by BioPAX. Widespread adoption of BioPAX for data exchange will increase access to and uniformity of pathway data from varied sources, thus increasing the efficiency of computational pathway research. This document describes the Level 1, version 1.4 release of the BioPAX ontology and the BioPAX format.

Scope of this document

This BioPAX documentation is targeted at bioinformaticians with an interest in biological pathway data. Those who are only interested in an overview of BioPAX are encouraged to read the introduction (section 1). It is expected that readers are familiar with one or more pathway databases and have a basic understanding of both bioinformatics and molecular and cellular biology. This background information is available in a number of textbooks^{1,2}.

This document provides an overview of the BioPAX Level 1, Version 1.4 ontology. This includes descriptions of the BioPAX ontology classes, sample use cases and best practice recommendations. This document does not provide a full definition of the BioPAX ontology, which is given by the BioPAX Level 1, Version 1.4 OWL file, located here:

<http://www.biopax.org/Downloads/Level1v1.4/biopax-level1.owl>

Ontology changes since Level 1, version 1.2

Previous version of ontology:

<http://www.biopax.org/Downloads/Level1v1.2/biopax-level1.owl>

The changes described below were made to biopax-level1.owl since the version 1.2 release. The latest version of the BioPAX Level 1 version 1.x OWL file can always be found here:

<http://www.biopax.org/release/biopax-level1.owl>

DIRECTION slot is now optional

The DIRECTION slot has been made optional. In previous versions of BioPAX, the DIRECTION slot was a non-optional property of the catalysis class. In some cases, however, it is known that an enzyme catalyzes a reaction but it is not known whether the catalysis is bidirectional or if it favors one direction over the other.

New slot: ID-VERSION

The ID-VERSION slot has been added to the xref class (and therefore also to its children: publicationXref, relationshipXref, and unificationXref). A number of biological databases track changes to database entries with ID version numbers. Since updates to database entries may alter the validity of assertions made in pathway data, such as the participation of an entity in an interaction, it is important to keep track of the ID version for which these assertions were made.

Key definitions

BioPAX workgroup: The group of volunteers designing the BioPAX ontology and format.

BioPAX ontology: The abstract representation of biological pathway concepts and their relationships developed by the BioPAX workgroup. This is also called the object model.

BioPAX format: The file format implementation of the BioPAX ontology that defines the syntax of representation for data. The BioPAX format is currently implemented only in OWL, but other implementations, such as XML Schema may be developed in the future.

OWL: Web Ontology Language. OWL can be used to both define an ontology and to store instance data that adheres to that ontology. The ontology definition can be used to validate a set of instances. An ontology may be defined within an instance data document or may be imported from an external OWL document. OWL is an XML-based language defined by the World Wide Web Consortium. See <http://www.w3.org/TR/owl-guide/>

Status of this document

This document has been reviewed by BioPAX workgroup members and interested third parties. Comments on this specification may be sent to biopax-discuss@biopax.org; archives of the comments are available by subscribing to our mailing list here:

<http://www.biopax.org/mailman/private/biopax-discuss/>.

This document and the BioPAX Level 1 OWL file will be updated over time, based on community input. The documentation for the latest version of BioPAX Level 1 version 1.x can always be found here:

<http://www.biopax.org/release/biopax-level1-documentation.pdf>

Document changes since the previous version

Previous version of this document:

<http://www.biopax.org/Downloads/Level1v1.2/biopax-level1-documentation.pdf>

- No sections of this document have been added or removed since the previous version.

Related documents

BioPAX Level 1, Version 1.4 OWL file:

<http://www.biopax.org/Downloads/Level1v1.4/biopax-level1.owl>

BioPAX Namespace

The following URI is defined to be BioPAX Level 1, version 1.x namespace:

<http://www.biopax.org/release/biopax-level1.owl#>

This namespace name (URI) will always be used to refer to the most recently released 1.x version of BioPAX; different URIs will be used for any and all other major versions of BioPAX Level 1 (e.g. versions 2.x, 3.x, etc.) See *Appendix B* for an explanation of BioPAX level and version numbers.

Table of contents

<i>BioPAX – Biological Pathways Exchange Language</i>	<i>1</i>
<i>Level 1, Version 1.4 Documentation</i>	<i>1</i>
Abstract	1
Scope of this document	1
Ontology changes since Level 1, version 1.2	2
DIRECTION slot is now optional	2
New slot: ID-VERSION	2
Key definitions	2
Status of this document	2
Document changes since the previous version	3
Related documents	3
BioPAX Namespace	3
Table of contents	4
How to Participate	6
<i>2 BioPAX Ontology Class Structure</i>	<i>8</i>
Top level classes	8
Entity (Root class of ontology)	9
Second level classes	9
Pathway	9
Interaction	10
PhysicalEntity	11
Interaction sub-classes	11
Control	11
Conversion	12
Control sub-classes	13
Catalysis	13
Modulation	14
Conversion sub-classes	14
Biochemical Reaction	14
Complex Assembly	15
Transport	16
Summary of Interaction Class Structure	17
PhysicalEntity leaf classes	18
RNA	18
Protein	18
Small Molecule	19
Complex	19
Utility classes	20
BioSource	20
ChemicalStructure	21

DataSource	21
OpenControlledVocabulary	21
PathwayStep	22
PhysicalEntityParticipant	22
Xref	22
PublicationXref	23
RelationshipXref	23
UnificationXref	24
Summary of BioPAX Class Structure	24
3 Examples	25
4 Best Practices	26
General Best Practices	26
Internal Identifiers	26
External References	28
Other recommendations	30
Best Practices: Metabolic Pathways	30
PARTICIPANTS slot	30
LEFT and RIGHT slots	30
Control	31
Catalysis	31
Conversion	31
Pathway	31
5 HOW-TO	33
Creating a knowledge-base using BioPAX and Protégé	33
Viewing Instances Graphically	34
6 Use Case Outlines	35
Data Sharing Between Databases	35
BioPAX as a knowledge-base (KB) Model	36
Pathway Data Warehouse	36
Pathway Analysis Software	36
Pathway Analysis Software Example: Molecular profiling analysis	36
Visualizing Pathway Diagrams	37
Pathway Modeling	37
Using BioPAX as metadata for SBML and CellML	37
Pathway analysis using logical inference	37
7 Glossary	39
Appendix A: Design Principles	40
Appendix B: Level and Version Numbers	41
Acknowledgements	41
References	42

1 Introduction

BioPAX (Biological Pathway Exchange) aims to facilitate the integration and exchange of data maintained in biological pathway databases. Traditionally, integrating data from a number of databases, diverse in form and content, has been a challenge in the field of Bioinformatics³. One solution is to define a mutually agreed upon file format as a standard way of representing a given type of data in a community. An example of such a standard is the DDBJ/EMBL/GenBank flat-file format, used to represent nucleic acid sequence data.

Currently, there is no file format standard broadly applicable to biological pathway data, despite the presence of this data in over 100 different internet accessible databases*. While previous work has been done to standardize specific types of pathway data, notably, the protein-protein interaction database community has developed a format called PSI-MI⁴, there is no format capable of representing all of the most frequently used types of pathway data. **The goal of the BioPAX project is to provide a data exchange format for pathway data that will represent the key elements of the data models from a wide range of popular pathway databases.** To achieve this goal, the BioPAX ontology was designed to support the data models of a number of existing pathway databases, such as [BioCyc](#)^{5,6}, [BIND](#)⁷, [WIT](#)⁸, [PATIKA](#)⁹, [Reactome](#), [aMAZE](#)¹⁰, [KEGG](#)¹¹ and others. When designing the BioPAX ontology for Level 1, the BioPAX workgroup endeavored to balance the many different representational needs of these and other biological pathway databases by adhering to design principles that promote interoperability. These design principles include flexibility, extensibility, optional encapsulation of frequently used external data, compatibility with other standards and computability (see Appendix 1: Design Principles).

Because pathway data are complex and can be represented at many levels of detail, the BioPAX group is using a leveled development approach, similar to that of SBML¹². While the overall framework of the BioPAX ontology, i.e. the root class structure, has been designed with the entire pathway data space in mind, representation of specific types of pathway data are the focus of individual levels. **BioPAX Level 1 is designed to represent metabolic pathway data.** Representing other types of pathway data with BioPAX Level 1 is possible but may not be optimal. Future levels will enhance coverage for additional types of pathway data, such as signal transduction pathways and molecular interaction networks; see the BioPAX Roadmap, located here:

http://www.biopax.org/Docs/BioPAX_Roadmap.html

How to Participate

Since a data exchange format is only useful if it is widely adopted, the BioPAX project aims to promote the use of the format by as many data sources as possible. This is achieved partly through community outreach at conferences and workshops, and partly through active participation in the project by data providers and consumers.

* <http://www.cbio.mskcc.org/prl>

There are many ways to participate in BioPAX development: one can participate directly in its design, provide feedback to the BioPAX group, provide data in the BioPAX format, develop software tools that support the BioPAX format, provide sponsorship for BioPAX activities, and encourage participation by others.

BioPAX participation is currently on a volunteer basis and members have typically paid their own expenses. The US Department of Energy (DOE) has provided some funding for holding meetings and will support additional meetings in the future.

More details are available on the www.biopax.org web page.

2 BioPAX Ontology Class Structure

This section provides an overview of the BioPAX class structure. Full definitions are found in the BioPAX OWL document (<http://www.biopax.org/release/biopax-level1.owl>). Text definitions of classes are provided along with synonyms, comments and examples, where possible, to help the reader understand the definition and intended use of each class.

Additionally, classes are shown graphically using ezOWL (see HOW-TO section below). Classes are shown as boxes, with the name of the class in a pale yellow box and the slots of the class in white boxes below the class name. A white **O** denotes an inherited slot and an orange **O** denotes a slot defined in this class.

Class	
O	InheritedSlot
O	Slot

Interspersed throughout this section are diagrams generated by GKB Editor showing a subset of the BioPAX class hierarchy as an overview or summary of the classes discussed in a subsection. Bold class names indicate that the class has children that are not currently displayed.



Top level classes

The BioPAX ontology defines 4 basic concepts in the ontology: the root level **entity** class and three subclasses: **pathway**, **interaction** and **physicalEntity**.

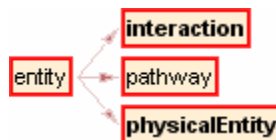









Table 1: Analogies of the root BioPAX ontology structure (first and second level classes) to other conceptual areas.			
	Linguistic	Graph representation	Pathway shorthand
Entity	Noun (Subject or Object)	Node	A, B, C
Relationship	Verb	Edge	\rightarrow , \rightarrow
Interaction	Phrase/Sentence	Either a node set by itself (member of a set relationship) or a node set connected to another node set by an edge (relationship between sets)	$A \rightarrow B$, $B \rightarrow C$
Pathway	Paragraph	Graph	$A \rightarrow B \rightarrow C$

Entity (Root class of ontology)

Definition: Any concept referred to as a discrete biological unit when describing pathways. This is the root class for all biological concepts in the ontology, which include pathways, interactions and physical entities. As the most abstract class in the ontology, instances of the entity class should be created rarely, if ever.

Synonyms: thing, object, bioentity.

entity	
	SYNONYMS
	COMMENT
	DATA-SOURCE
	SHORT-NAME
	AVAILABILITY
	NAME
	XREF

Second level classes

Pathway

Definition: An entity that consists of a set of interactions. A pathway is a series of molecular interactions and reactions, often forming a network, which biologists have found useful to group together for organizational, historic, biophysical or other reasons.

Synonyms: network

Comment: It is possible to define a pathway without specifying the interactions within the pathway. In this case, the pathway instance could consist simply of a name and could be treated as a black box. Currently this class is not subclassed, but could conceivably be subclassed in different ways into various subtypes of pathways, such as “metabolic pathway”, “signal

transduction pathway”, “gene regulatory network” or “physiological process”, “cellular process” as is done in the Gene Ontology¹³ “biological process” ontology. In fact, the entire GO biological process ontology could be included under the pathway class.

Examples: glycolysis, valine biosynthesis

pathway	
<input type="radio"/>	COMMENT
<input type="radio"/>	AVAILABILITY
<input type="radio"/>	DATA-SOURCE
<input type="radio"/>	SYNONYMS
<input type="radio"/>	SHORT-NAME
<input type="radio"/>	NAME
<input type="radio"/>	XREF
<input checked="" type="radio"/>	PATHWAY-COMPONENTS
<input checked="" type="radio"/>	ORGANISM

Interaction

Definition: An entity that defines a single biochemical interaction between two or more entities. An interaction cannot be defined without the entities it relates. Since it is a highly abstract class in the ontology, instances of the interaction class should be created rarely.

interaction	
<input type="radio"/>	SYNONYMS
<input type="radio"/>	COMMENT
<input type="radio"/>	DATA-SOURCE
<input type="radio"/>	SHORT-NAME
<input type="radio"/>	AVAILABILITY
<input type="radio"/>	NAME
<input type="radio"/>	XREF
<input checked="" type="radio"/>	PARTICIPANTS

Comment: Currently this class only has subclasses that define biochemical interactions; later levels of BioPAX may define other types of interactions.

Naming rationale: A number of names were considered for this concept, including “process”, “synthesis” and “relationship”; Interaction was chosen as it is understood by biologists in a biological context and is compatible with [PSI-MI](#).

Examples: protein-protein interaction, biochemical reaction, enzyme catalysis

PhysicalEntity

Definition: An entity that has a physical structure. This class serves as the super-class for all physical entities, although its current set of subclasses is limited to molecules. Physical entities are frequent building blocks of interactions. As a highly abstract class in the ontology, instances of the physicalEntity class should be created rarely, if ever.

Synonyms: part, interactor, object

Comment: This may be expanded to include DNA, photon, environment, cell and cellular component in later levels of BioPAX, depending on community need.

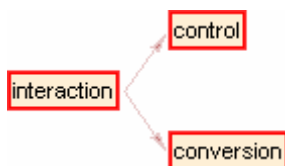
Naming rationale: It's difficult to find a name that encompasses all of the subclasses of this class without being too general. E.g. PSI-MI uses 'interactor', BIND uses 'object', BioCyc uses 'chemicals'. physicalEntity seems to be a good specialization of entity.

Examples: protein, small molecule, RNA,

physicalEntity	
①	SYNONYMS
①	COMMENT
①	DATA-SOURCE
①	SHORT-NAME
①	AVAILABILITY
①	NAME
①	XREF

Interaction sub-classes

Two terms exist under interaction: Control and conversion. In future BioPAX levels, this list may be extended to include other classes, such as genetic interactions (see the BioPAX Roadmap).



Control

Definition: An interaction in which one entity regulates, modifies, or otherwise influences another. Two types of control interactions are defined: activation and inhibition. Since this class is a superclass for specific types of control, instances of the control class should only be generated when none of its subclasses are applicable.

Synonyms: regulation, mediation

Examples: Enzyme catalysis controls a biochemical reaction, transport catalysis controls transport, a small molecule that inhibits a pathway by an unknown mechanism controls the pathway.

control	
<input type="checkbox"/>	PARTICIPANTS
<input type="checkbox"/>	SYNONYMS
<input type="checkbox"/>	COMMENT
<input type="checkbox"/>	DATA-SOURCE
<input type="checkbox"/>	SHORT-NAME
<input type="checkbox"/>	AVAILABILITY
<input type="checkbox"/>	NAME
<input type="checkbox"/>	XREF
<input checked="" type="checkbox"/>	CONTROLLED
<input checked="" type="checkbox"/>	CONTROL-TYPE
<input checked="" type="checkbox"/>	CONTROLLER

Conversion

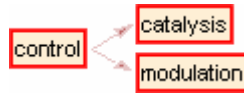
Definition: An interaction in which one or more entities is physically transformed into one or more other entities. This class is designed to represent a simple, single-step transformation. Multi-step transformations, such as the conversion of glucose to pyruvate in the glycolysis pathway, should be represented as pathways, if known. Since it is a highly abstract class in the ontology, instances of the conversion class should be created rarely, if ever.

Examples: A biochemical reaction converts substrates to products, the process of complex assembly converts single molecules to a complex, transport converts entities in one compartment to the same entities in another compartment.

conversion	
<input type="checkbox"/>	PARTICIPANTS
<input type="checkbox"/>	SYNONYMS
<input type="checkbox"/>	COMMENT
<input type="checkbox"/>	DATA-SOURCE
<input type="checkbox"/>	SHORT-NAME
<input type="checkbox"/>	AVAILABILITY
<input type="checkbox"/>	NAME
<input type="checkbox"/>	XREF
<input checked="" type="checkbox"/>	LEFT
<input checked="" type="checkbox"/>	RIGHT
<input checked="" type="checkbox"/>	SPONTANEOUS

Control sub-classes

Two types of control processes exist under the control class: catalysis and modulation.



Catalysis

Definition: A control interaction in which a physical entity (a catalyst) increases the rate of a conversion interaction by lowering its activation energy. Instances of this class describe a pairing between a catalyzing entity and a catalyzed conversion. A separate catalysis instance should be created for each different conversion that a physicalEntity may catalyze and for each different physicalEntity that may catalyze a conversion. For example, a bifunctional enzyme that catalyzes two different biochemical reactions would be linked to each of those biochemical reactions by two separate instances of the catalysis class.

Typically, each step in a metabolic pathway is either an instance of the catalysis class or a spontaneous conversion, which occurs under biological conditions without the aid of a catalyzing entity.

Synonyms: facilitation, acceleration.

Examples: The catalysis of a biochemical reaction by an enzyme, the enabling of a transport interaction by a membrane pore complex, and the facilitation of a complex assembly by a scaffold protein. Hexokinase -> (The “Glucose + ATP -> Glucose-6-phosphate +ADP” reaction). A plasma membrane Na⁺/K⁺ ATPase is an active transporter (antiport pump) using the energy of ATP to pump Na⁺ out of the cell and K⁺ in. Na⁺ from cytoplasm to extracellular space would be described in a transport instance. K⁺ from extracellular space to cytoplasm would be described in a transport instance. The ATPase pump would be stored in a catalysis instance controlling each of the above transport instances.

catalysis	
<input type="checkbox"/>	CONTROLLED
<input type="checkbox"/>	PARTICIPANTS
<input type="checkbox"/>	CONTROL-TYPE
<input type="checkbox"/>	SYNONYMS
<input type="checkbox"/>	COMMENT
<input type="checkbox"/>	DATA-SOURCE
<input type="checkbox"/>	SHORT-NAME
<input type="checkbox"/>	CONTROLLER
<input type="checkbox"/>	AVAILABILITY
<input type="checkbox"/>	NAME
<input type="checkbox"/>	XREF
<input checked="" type="checkbox"/>	COFACTOR
<input checked="" type="checkbox"/>	DIRECTION

Modulation

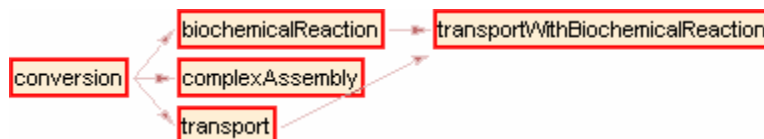
Definition: A control interaction in which a physical entity modulates a catalysis interaction. Biologically, most modulation interactions describe an interaction in which a small molecule alters the ability of an enzyme to catalyze a specific reaction. Instances of this class describe a pairing between a modulating entity and a catalysis interaction. A separate modulation instance should be created for each different catalysis that a physical entity may modulate and for each different physical entity that may modulate a catalysis instance. A typical modulation instance has a small molecule as the controller entity and a catalysis instance as the controlled entity.

Examples: Allosteric activation and competitive inhibition of an enzyme's ability to catalyze a specific reaction.

modulation	
①	CONTROLLED
①	PARTICIPANTS
①	CONTROL-TYPE
①	SYNONYMS
①	COMMENT
①	DATA-SOURCE
①	SHORT-NAME
①	CONTROLLER
①	AVAILABILITY
①	NAME
①	XREF

Conversion sub-classes

Four types of conversion processes exist under the conversion class: biochemical reaction, complex assembly, transport and transportWithBiochemicalReaction.



Biochemical Reaction

Definition: A conversion interaction in which one or more entities (substrates) undergo covalent changes to become one or more other entities (products). The substrates of biochemical reactions are defined in terms of sums of species. This is convention in biochemistry, and, in principle, all of the EC reactions should be biochemical reactions.

Examples: $\text{ATP} + \text{H}_2\text{O} = \text{ADP} + \text{P}_i$

Comment: In this example reaction, ATP is considered to be an equilibrium mixture of several species, namely ATP⁴⁻, HATP³⁻, H₂ATP²⁻, MgATP²⁻, MgHATP⁻, and Mg₂ATP. Additional species may also need to be considered if other ions (e.g. Ca²⁺) that bind ATP are present. Similar considerations apply to ADP and to inorganic phosphate (Pi). When writing biochemical reactions, it is important not to attach charges to the biochemical reactants and not to include ions such as H⁺ and Mg²⁺ in the equation. The reaction is written in the direction specified by the EC nomenclature system, if applicable, regardless of the physiological direction(s) in which the reaction proceeds.

Polymerization reactions involving large polymers whose structure is not explicitly captured should generally be represented as unbalanced reactions in which the monomer is consumed but the polymer remains unchanged, e.g. glycogen + glucose = glycogen.

biochemicalReaction	
<input type="radio"/>	LEFT
<input type="radio"/>	PARTICIPANTS
<input type="radio"/>	SYNONYMS
<input type="radio"/>	COMMENT
<input type="radio"/>	SPONTANEOUS
<input type="radio"/>	DATA-SOURCE
<input type="radio"/>	SHORT-NAME
<input type="radio"/>	AVAILABILITY
<input type="radio"/>	RIGHT
<input type="radio"/>	NAME
<input type="radio"/>	XREF
<input checked="" type="radio"/>	DELTA-S
<input checked="" type="radio"/>	EC-NUMBER
<input checked="" type="radio"/>	KEQ
<input checked="" type="radio"/>	DELTA-G
<input checked="" type="radio"/>	DELTA-H

Complex Assembly

Definition: A conversion interaction in which a set of physical entities, at least one being a macromolecule (protein or RNA), aggregate via non-covalent interactions. One of the participants of a complexAssembly must be an instance of the class complex.

Synonyms: aggregation, complexFormation

Comment: This class is also used to represent complex disassembly. The direction in which the complexAssembly occurs (toward either assembly or disassembly) is specified via either the SPONTANEOUS slot or the DIRECTION slot (in the catalysis class), depending on whether the interaction occurs spontaneously or must be catalyzed in order to occur.

Examples: Assembly of the TFB2 and TFB3 proteins into the TFIIF complex, and assembly of the ribosome through aggregation of its subunits.

The following are not examples of complex assembly: Covalent phosphorylation of a protein (this is a biochemicalReaction), the TFIIF complex itself (this is an instance of the complex class, not the complexAssembly class).

complexAssembly	
<input type="radio"/>	LEFT
<input type="radio"/>	PARTICIPANTS
<input type="radio"/>	SYNONYMS
<input type="radio"/>	COMMENT
<input type="radio"/>	DATA-SOURCE
<input type="radio"/>	SHORT-NAME
<input type="radio"/>	AVAILABILITY
<input type="radio"/>	RIGHT
<input type="radio"/>	NAME
<input type="radio"/>	XREF
<input type="radio"/>	SPONTANEOUS

Transport

Definition: A conversion interaction in which an entity (or set of entities) changes location within or with respect to the cell. A transport interaction does not include the transporter entity, even if one is required in order for the transport to occur. Instead, transporters are linked to transport interactions via the catalysis class.

Synonyms: translocation.

Comment: Transport interactions do not involve chemical changes of the participant(s). These cases are handled by the transportWithBiochemicalReaction class.

Examples: The movement of Na⁺ into the cell through an open voltage-gated channel.

transport	
<input type="radio"/>	LEFT
<input type="radio"/>	PARTICIPANTS
<input type="radio"/>	SYNONYMS
<input type="radio"/>	COMMENT
<input type="radio"/>	DATA-SOURCE
<input type="radio"/>	SHORT-NAME
<input type="radio"/>	AVAILABILITY
<input type="radio"/>	RIGHT
<input type="radio"/>	NAME
<input type="radio"/>	XREF
<input type="radio"/>	SPONTANEOUS

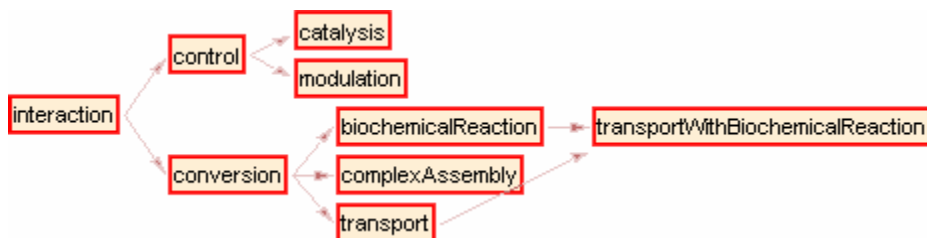
TransportWithBiochemicalReaction

Definition: A conversion interaction that is both a biochemicalReaction and a transport. In transportWithBiochemicalReaction interactions, one or more of the substrates change both their location and their physical structure.

Examples: In the PEP-dependent phosphotransferase system, transportation of sugar into an E. coli cell is accompanied by the sugar's phosphorylation as it crosses the plasma membrane. Also, active transporters that use ATP as an energy source fall under this category, even if the only covalent change is the hydrolysis of ATP to ADP.

transportWithBiochemicalReaction	
①	LEFT
①	PARTICIPANTS
①	DELTA-S
①	EC-NUMBER
①	SYNONYMS
①	COMMENT
①	SPONTANEOUS
①	KEQ
①	DELTA-G
①	DATA-SOURCE
①	SHORT-NAME
①	AVAILABILITY
①	DELTA-H
①	RIGHT
①	NAME
①	XREF

Summary of Interaction Class Structure



PhysicalEntity leaf classes



RNA

Definition: A physical entity consisting of a sequence of ribonucleotide monophosphates; a ribonucleic acid

Examples: messengerRNA, microRNA, ribosomalRNA. A specific example is the let-7 microRNA.

rna	
<input type="checkbox"/>	SYNONYMS
<input type="checkbox"/>	COMMENT
<input type="checkbox"/>	DATA-SOURCE
<input type="checkbox"/>	SHORT-NAME
<input type="checkbox"/>	AVAILABILITY
<input type="checkbox"/>	NAME
<input type="checkbox"/>	XREF
<input checked="" type="checkbox"/>	ORGANISM
<input checked="" type="checkbox"/>	SEQUENCE

Protein

Definition: A physical entity consisting of a sequence of amino-acids; a protein monomer; a single polypeptide chain.

Examples: The epidermal growth factor receptor (EGFR) protein.

protein	
<input type="checkbox"/>	SYNONYMS
<input type="checkbox"/>	COMMENT
<input type="checkbox"/>	DATA-SOURCE
<input type="checkbox"/>	SHORT-NAME
<input type="checkbox"/>	AVAILABILITY
<input type="checkbox"/>	NAME
<input type="checkbox"/>	XREF
<input checked="" type="checkbox"/>	ORGANISM
<input checked="" type="checkbox"/>	SEQUENCE

Small Molecule

Definition: Any bioactive molecule that is not a peptide, protein, or RNA. Generally these are non-polymeric, but complex carbohydrates and DNA are not explicitly modeled as classes in this version of the ontology, thus are forced into this class.

Comment: There is a known lack of adequate small molecule databases to cross-reference from this class.

Examples: glucose, penicillin

smallMolecule	
<input type="checkbox"/>	SYNONYMS
<input type="checkbox"/>	COMMENT
<input type="checkbox"/>	DATA-SOURCE
<input type="checkbox"/>	SHORT-NAME
<input type="checkbox"/>	AVAILABILITY
<input type="checkbox"/>	NAME
<input type="checkbox"/>	XREF
<input checked="" type="checkbox"/>	CHEMICAL-FORMULA
<input checked="" type="checkbox"/>	STRUCTURE
<input checked="" type="checkbox"/>	MOLECULAR-WEIGHT

Complex

Definition: A physical entity whose structure is comprised of other physical entities bound to each other non-covalently, at least one of which is a macromolecule (protein or RNA).

Complexes must be stable enough to function as a biological unit; in general, the temporary association of an enzyme with its substrate(s) should not be considered or represented as a complex. A complex is the physical product of an interaction (complexAssembly), thus is not an interaction itself.

Comment: Complexes can be defined recursively to describe smaller complexes within larger complexes, e.g., a participant in a complex can itself be a complex.

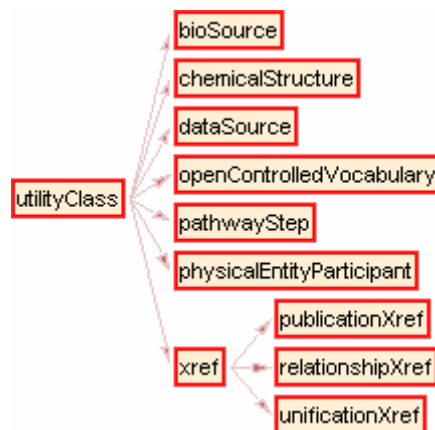
The boundaries on the size of complexes described by this class are not defined here, although elements of the cell as large and dynamic as, e.g., a mitochondrion would typically not be described using this class (later versions of this ontology may include a cellularComponent class to represent these). The strength of binding of the components is also not described.

Examples: Ribosome, RNA polymerase II. Other examples of this class include complexes of multiple protein monomers and complexes of proteins and small molecules.

complex	
0	SYNONYMS
0	COMMENT
0	DATA-SOURCE
0	SHORT-NAME
0	AVAILABILITY
0	NAME
0	XREF
0	COMPONENTS
0	ORGANISM

Utility classes

A number of slots in the ontology accept instances of utility classes as values. In essence, these utility classes provide a custom data type when a simple data type, such as a string or an integer, is insufficient.



BioSource

Definition: A utility class that defines the biological source of a protein or RNA. Other entities are either considered source-neutral (e.g. small molecules) or their biological source can be deduced from their constituents.

Examples: human, mouse liver tissue, and HeLa cells.

bioSource	
0	COMMENT
0	CELLTYPE
0	TISSUE
0	NAME
0	TAXON-XREF

ChemicalStructure

Definition: A utility class that defines a small molecule structure. An instance of this class can also define additional information about a small molecule, such as its chemical formula, names, and synonyms. This information is stored in the slot STRUCTURE-DATA, in one of two formats: the CML format¹⁴ (see URL www.xml-cml.org) or the SMILES format¹⁵ (see URL www.daylight.com/dayhtml/smiles/). The STRUCTURE-FORMAT slot specifies which format used is used.

Examples: The following SMILES string, which describes the structure of glucose-6-phosphate:

`'C(OP(=O)(O)O)[CH]1([CH](O)[CH](O)[CH](O)[CH](O)O1)'`.

chemicalStructure	
0	COMMENT
0	STRUCTURE-FORMAT
0	STRUCTURE-DATA

DataSource




Definition: A description of the source of this data. Currently, this class only contains a free text description, but may be made more structured in future levels.

Examples: A database or person name.

dataSource	
0	COMMENT
0	NAME




OpenControlledVocabulary

Definition: A utility class used to import terms from external controlled vocabularies (CVs) into the ontology. To support consistency and compatibility, open, freely available CVs should be used whenever possible, such as the Gene Ontology (GO)¹³. A repository for open biological CVs has been created by the OBO project (<http://obo.sourceforge.net/>).

openControlledVocabulary	
	COMMENT
	XREF
	TERM





PathwayStep

Definition: A utility class that describes the order in which interactions occur in a pathway. The interactions that take place at a pathway step are listed and an ordering relationship between pathway steps by pointing to the next pathwayStep(s) in the pathway is given. For example, a metabolic pathway may contain a pathway step composed of one biochemical reaction (BR1) and one catalysis (CAT1) instance, where CAT1 describes the catalysis of BR1.

pathwayStep	
	COMMENT
	STEP-INTERACTIONS
	NEXT-STEP

PhysicalEntityParticipant

Definition: A utility class that describes any additional special characteristics of a physical entity required in order for it to participate in an interaction. In the current ontology, these include stoichiometric coefficient and cellular location. For example, in the interaction describing the transport of L-arginine into the cytoplasm in E. coli, the LEFT slot in the interaction would be filled with an instance of physicalEntityParticipant that specified the location of L-arginine as periplasm and the stoichiometric coefficient as one.

physicalEntityParticipant	
	COMMENT
	CELLULAR-LOCATION
	STOICHIOMETRIC-COEFFICIENT
	PHYSICAL-ENTITY

Xref

Definition: A utility class that defines a reference between an instance of a class in this ontology to an object in an external resource.

xref	
0	COMMENT
0	DB-VERSION
0	ID
0	DB

PublicationXref

Definition: An xref that defines a reference to a publication such as a book, journal article, web page, or software manual. The reference may or may not be in a database, although references to PubMed are preferred when possible. The publication should make a direct reference to the instance it is attached to.

Examples: PubMed:10234245

publicationXref	
0	DB-VERSION
0	COMMENT
0	ID
0	DB
0	TITLE
0	SOURCE
0	URL
0	YEAR
0	AUTHORS

RelationshipXref

Definition: An xref that defines a reference to an entity in an external resource that does not have the same biological identity as the referring entity.

Examples: A link between a gene G in a BioPAX data collection, and the protein product P of that gene in an external database. This is not a unification xref because G and P are different biological entities (one is a gene and one is a protein). Another example is a relationship xref for a protein that refers to the Gene Ontology biological process, e.g. 'immune response,' that the protein is involved in.

relationshipXref	
0	DB-VERSION
0	COMMENT
0	ID
0	DB
0	RELATIONSHIP-TYPE

UnificationXref

Definition: A unification defines a reference to an entity in an external resource that has the same biological identity as the referring entity¹⁶. For example, if one wished to link from a database record, C, describing a chemical compound in a BioPAX data collection to a record, C', describing the same chemical compound in an external database, one would use a unification xref since records C and C' describe the same biological identity. Generally, unification xrefs should be used whenever possible, although there are cases where they might not be useful, such as application to application data exchange.

Comment: Unification xrefs in physical entities are essential for data integration, but are less important in interactions. This is because unification xrefs on the physical entities in an interaction can be used to compute the equivalence of two interactions of the same type.

An xref in a protein pointing to a gene, e.g. in the LocusLink database¹⁷, would not be a unification xref since the two entities do not have the same biological identity (one is a protein, the other is a gene). Instead, this link should be a captured as a relationship xref¹⁶.

Examples: An xref in a protein instance pointing to an entry in the Swiss-Prot database, and an xref in an RNA instance pointing to the corresponding RNA sequence in the RefSeq database.

unificationXref	
0	DB-VERSION
0	COMMENT
0	ID
0	DB

Summary of BioPAX Class Structure



3 Examples

A number of examples of pathways in the BioPAX format are available for download here:

<http://cvs.sourceforge.net/viewcvs.py/biopax/biopax/examples/>

As new examples are developed, they will be posted on this site.

4 Best Practices

While the BioPAX ontology is very structured and imposes logical constraints so that data encoded make sense for the use cases envisioned, a few parts of the ontology have the potential for encoding data in multiple ways. This section recommends best practices in the use of the ontology for data exchange between groups. It is expected that major data providers follow these recommendations to ensure compatibility of their data with other BioPAX data.

Users of BioPAX who are not exchanging data between groups, e.g. using BioPAX as an internal data model for their software, might find alternate representations to the ones recommended here more useful for their purposes.

General Best Practices

General best practices relate to all uses of BioPAX.

Internal Identifiers

Internal identifiers (IDs) relate elements within a BioPAX document to each other. For example, if a number of proteins are defined, a protein complex can be constructed by simply pointing to the existing proteins using their unique internal IDs. Internal identifiers are expected to be unique within a single document. This is a major feature of OWL documents - instances are defined once and simply pointed to as necessary.

RDF ID

In an OWL document, such as BioPAX, each instance of a class will have an RDF ID. This comes from the Resource Descriptor Framework standard (<http://www.w3.org/RDF/>). These IDs must be unique and are used to reference class instances within a document. An RDF ID exists within a namespace, which can be explicitly appended before the RDF ID. If not explicit, the RDF ID exists in the default namespace of the document. Like anchors in HTML, a pointer to an RDF ID defined elsewhere in the document is denoted with a hash mark (“#”) in front of the RDF ID.

Example

```
<protein rdf:ID="protein76">  
  <XREF rdf:resource="#xref1146"/>  
</protein>
```

It is recommended that RDF IDs do not encode any semantics and should be either unique (within the file) positive integers (encoded as strings) or should be composed of the class name followed by a unique positive integer (e.g. “protein76”). Some applications that use OWL, such as Protégé and some examples of OWL from the main OWL website, use human readable names for the RDF IDs. As long as these names are unique, a BioPAX document will be valid, but the

use of human readable names as RDF IDs might encourage people to rely on information stored in them and is thus not recommended, since RDF IDs may not persist after certain data processing operations, such as integrating data from two separate BioPAX files.

Please note that in the Protégé tool, the RDF ID of an instance is referred to as its Name. This should not be confused with the BioPAX NAME (all letters capitalized) slot, which is meant to provide the human readable name for biological entities (Figure 1). Protégé can be configured to display the value of the NAME slot (or another field value) instead of the RDF ID. Use the Display Slot pull-down menu in the Individuals tab to select the slot to display.

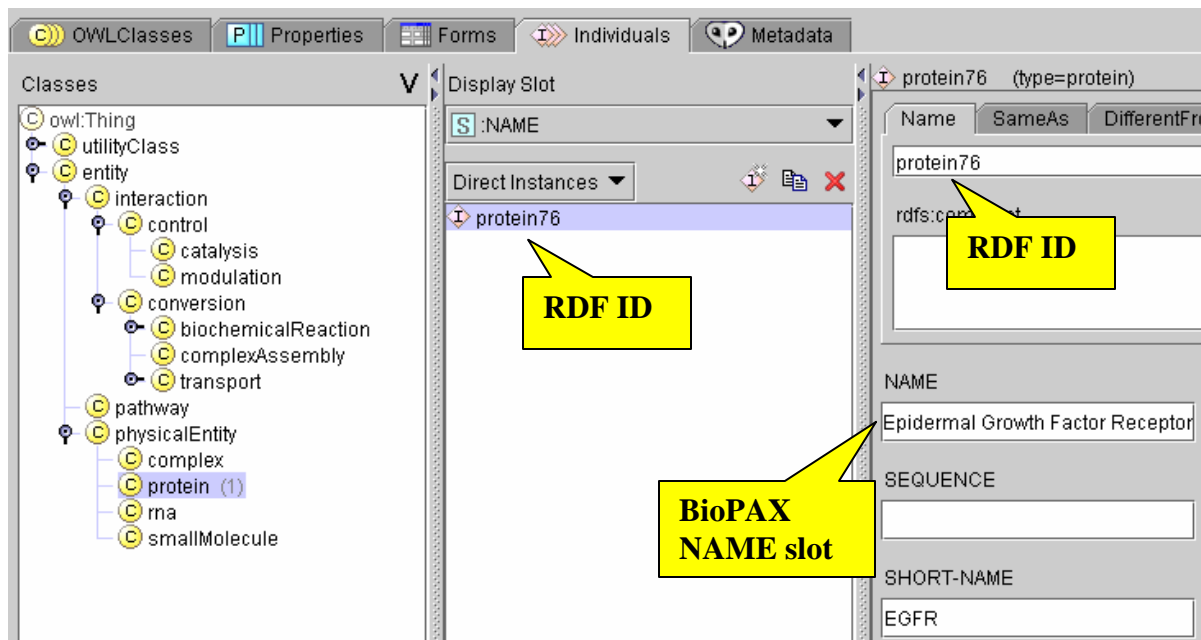


Figure 1: The difference between name and RDF ID shown in Protégé.

Document namespace

OWL XML documents require a default namespace. The creator of the BioPAX document should create a namespace and encode it in the BioPAX document. The namespace and the RDF ID may be used together to reference instances in a document from an external document (explicit use of namespace). This reference mechanism is part of the basis of the planned Semantic Web (<http://www.w3.org/2001/sw/>). If a BioPAX document is going to be on the Semantic Web, it should have a unique namespace. Since there is no namespace naming authority, it is not possible to guarantee unique namespaces across the internet, but following these recommendations will reduce the chances of naming collisions.

Technically, any string without spaces is allowed (see [namespace rules](#)) as a namespace. Operationally, a URL (or more generally a URI) should be used. This does not have to be a 'real' URL that resolves to a web page, but it should be related to the organization of the creator and a registered domain name owned by the organization is useful to include e.g. "http://ecocyc.org/ontology/biopax/#".

Use of the `xmlns` and `xml:base` attributes to specify the namespace for any BioPAX documents created is recommended. The BioPAX ontology definition should be imported and the BioPAX namespace should be defined using the 'bp' string e.g.

`xmlns:bp=http://www.biopax.org/release/biopax-level1.owl`, so that XML elements in the file appear like this `<bp:pathway></bp:pathway>`.

A typical header of an OWL XML document that uses the BioPAX ontology will look like this:

```
<?xml version="1.0"?>
<rdf:RDF
  xmlns="http://www.myorganization.org/ontology#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:xs=http://www.w3.org/2001/XMLSchema#
  xmlns:bp="http://www.biopax.org/release/biopax-level1.owl#"
  xml:base="http://www.myorganization.org/ontology#"
>
<owl:Ontology rdf:about="">
  <owl:imports rdf:resource="http://www.biopax.org/release/biopax-level1.owl"/>
</owl:Ontology>
```

Where <http://www.myorganization.org/ontology#> defines the namespace for this document.

OWL XML documents that mix BioPAX definitions with those from other ontologies or extend BioPAX will have different ways of using namespaces, but those are not dealt with here. Extensions to the BioPAX ontology are not expected to be compatible with tools written specifically to support the BioPAX ontology.

External References

External references (Xrefs) relate elements within a BioPAX document to external data. Xrefs are more than just identifiers, as they contain the name of the data source the identifiers are part of. Xrefs are NOT related to RDF IDs. They exist to uniquely point to a record in an external data source (e.g. a database). Examples are pointers from a protein instance in BioPAX to a record in a database describing the protein. The creator of the BioPAX document likely does not have control over all external references.

Unification xrefs

Abundant use of unification xrefs, where possible, is highly recommended, especially in `physicalEntity` instances. These xrefs allow a user to understand that two independent instances from different BioPAX documents are actually the same entity (as long as they share one or more unification xrefs).

When exporting data from a database with primary keys, those keys should generally be encoded as unification xrefs. For example, if a database contains biochemical reactions with IDs for both the reactions and the small molecules that participate in those reactions, unification xrefs containing these IDs should appear in the corresponding BioPAX instances generated by the database. In general, the original data record from which an instance was generated should be pointed to via a unification xref. The exception to this rule occurs when the native class of the data is not completely synonymous with the BioPAX class to which it is mapped. In these cases, the resulting BioPAX instances should point back to the original data records via relationship xrefs.

Caution: Complications with unification xrefs can arise when the database that is being pointed to contains redundant information or contains more than one type of record. If a database contains redundant information, such as GenBank or Chemical Abstracts Service (CAS), it is possible to reference the same physical entity in the same database, but use IDs of different redundant records. In this case, unification xrefs can not be guaranteed to be useful in determining if two physical entities are the same across multiple BioPAX documents. Also, if a database contains different types of records, such as mRNA and protein records in GenBank or chemical structures with and without R groups in CAS, then it may be impossible to determine the type of record referenced, which may lead to unification xrefs that point to molecules of a different type than the referencing physical entity. Care in creating unification xrefs should be taken when linking to these types of databases.

Publication xrefs

Publication xrefs should make use of PubMed IDs wherever possible. The DB slot of an xref to an entry in PubMed should use the string “PubMed” and not “MEDLINE”.

Use of xrefs within specific classes

Taxonomy references in the TAXON-REF slot of the bioSource class should be stored as a unification xrefs unless the species is not in an existing DB, in which case it should be stored as a general ‘xref’ instance.

References to an external controlled vocabulary term within the OpenControlledVocabulary class should use a unification xref where possible (e.g. GO:0005737)

Inside an xref

Within any xref, database names (in the DB slot) should be from a controlled vocabulary of database names to avoid data integration problems that arise when different people use different spellings of database names. The PSI-MI has a database name controlled vocabulary under development. If this is not possible to use, be careful to use the database name exactly as spelled on the database website (e.g. Use “Swiss-Prot” instead of swissprot, SWP or other frequent spellings).

Database accession numbers that contain version information should keep the version information with the accession number in the ID field e.g. “CAA61361.1”

Other recommendations

The COMMENT slot should be used instead of the OWL documentation elements (rdfs:comment) for instances. The Documentation text entry box in Protégé (which writes documentation as an rdfs:comment) should never be used for instances.

The AVAILABILITY should be a copyright statement, if necessary.

The DATA-SOURCE slot, present in all BioPAX entities, should be used to describe the source of the data. This is meant to be used by databases that export their data to the BioPAX format or by systems that are integrating data from multiple sources. The granularity of use (specifying the data source in more or fewer places) is up to the user.

The NAME slot, present in all BioPAX entities, is a string that should be treated as case insensitive. Names should be all lower case or capitalized, but not all UPPER CASE.

Best Practices: Metabolic Pathways

The main use of BioPAX level 1 is representation of metabolic pathways. **These best practice recommendations relate to this use and should be followed in addition to, not instead of, the best practices listed above.** It is possible to use BioPAX level 1 to represent other types of pathway information, but this document does not address those uses. The BioPAX workgroup will make note of how BioPAX level 1 is being used to help in the design of future levels.

PARTICIPANTS slot

The PARTICIPANTS slot of the interaction class (and subclasses) should be the union of whatever is in the LEFT, RIGHT slots of conversions or the CONTROLLER, CONTROLLED slots of control interactions. This slot is optional and is present mainly to enable representation of general relationships in the interactions class.

LEFT and RIGHT slots

The direction in which a conversion proceeds (either from left-to-right or from right-to-left) is specified by the catalyzing interaction (via the DIRECTION slot) or, in the case of spontaneous conversions, by the SPONTANEOUS slot. Therefore, the substrates and products of a conversion may be placed in either the LEFT or the RIGHT slots. However, in order to ease data integration it is preferable that users adhere to the same conventions for the contents of these slots. We therefore recommend the following, in order of precedence:

- If the conversion has an Enzyme Commission (EC) number or a Transport Commission (TC) number, store the participants in the LEFT and RIGHT slots such that they mirror the EC/TC reaction.
- For complex assemblies, store the subunits in the LEFT slot and the complex in the RIGHT slot.
- For transports, store the outermost participants (relative to the interior of the cell or organelle) in the LEFT slot and the innermost participants in the RIGHT slot.
- If none of the above are applicable, store the participants from left-to-right in the order that the conversion occurs or is suspected to occur in the pathway.

Control

In general, the targets of control processes (i.e. occupants of the CONTROLLED slot) should be interactions. Conceptually, interactions should be thought of as behaviors that physical entities may exhibit and it is these behaviors, rather than the physical entities themselves, that should be controlled or modified.

For example, a kinase activating a protein is a frequent event, especially in signaling pathways. Instead of capturing this information as one physical entity activating (via an instance of the control class) another, however, this information should be captured as the kinase catalyzing (via an instance of the catalysis class) a reaction in which the protein is phosphorylated.

Catalysis

Generally, the enzyme catalyzing a conversion is known and the use of this class is obvious. In the cases where a reaction is known to occur but the enzyme is not known, a catalysis instance should be created without a controller specified (i.e. the CONTROLLER slot should remain empty).

Conversion

The spontaneity of a conversion is expressed with the SPONTANEOUS slot. If the conversion is not spontaneous, or if the spontaneity is not known, the SPONTANEOUS slot should be left empty.

Pathway

A pathway is defined using interactions and pathwayStep instances. For most metabolic pathways, the contents of the PATHWAY-COMPONENTS slot should be a set of pathwayStep instances, one for each step of the pathway. In rare cases, such as if a pathway has not been completely elucidated, the specific order in which some interactions occur in a pathway may not be known. These interactions may be stored directly in the PATHWAY-COMPONENTS slot without being wrapped inside pathwayStep instances. The PATHWAY-COMPONENTS slot may also be left completely empty, in which case the pathway would simply have a name and could be treated as a black box. This use is valid according to the BioPAX ontology, but in

general creating black-box pathways with the Level 1 BioPAX ontology is not encouraged as the intent of Level 1 is to represent pathways with a high degree of detail.

Each pathwayStep instance should contain at most one conversion, typically one catalysis, and any number of modulation instances. Exception to this rule are cases in which a conversion is known to be catalyzed by multiple enzymes. In these cases, each separate catalysis instance should be included in the pathwayStep (providing each occurs within the context of the pathway).

A pathway step should not be listed in the NEXT-STEP slot of another pathwayStep if the intersection of the entities in the participants slots of their interactions is empty. Typically, at least one product of the conversion in each preceding pathwayStep should participate either as a CONTROLLER or as a substrate to the conversion interaction of a pathwayStep.

5 HOW-TO

Creating a knowledge-base using BioPAX and Protégé

Protégé is an open-source ontology and knowledge-base editor from Stanford University. It can be used to view and edit the BioPAX ontology and to create a database of instances of BioPAX classes. The ezOWL plugin allows graphical viewing and editing of an ontology, but does not graphically show instances.

To use BioPAX in Protégé, it is necessary to download three components:

1. Protégé from <http://protege.stanford.edu/>
Downloading the current stable release and not the beta release is recommended.
2. The Protégé OWL Plugin from <http://protege.stanford.edu/plugins/owl/index.html>
This allows Protégé to understand the OWL format.
3. The Protégé ezOWL Plugin from <http://iweb.etri.re.kr/ezowl/>
Used to view and edit an OWL ontology graphically.

Follow the instructions for installing these applications provided on their respective web pages.

After installing the above software, create a new OWL Files project in Protégé. To load the BioPAX ontology, one may either: a) import the BioPAX OWL file from the web (recommended), or b) load the ontology from a local copy of the BioPAX OWL file.

To import from the web:

- 1) On the Metadata tab (on the main screen next to the Individuals tab) click on the + symbol near where it says "Namespace Prefixes". This will add a new namespace to the project. The new namespace must be manually changed from "http://www.domain2.com#" to "http://www.biopax.org/biopax-level1.owl#" and check the "Imported" checkbox. The prefix should be changed from "p1" to "biopax".
- 2) Save the project as an OWL file, then reload it (via the "Project→Import..." menu item).
- 3) Upon reloading, the BioPAX ontology will be visible (grayed-out) in the classes menu.
- 4) Use the Individuals tab to create instances.
- 5) To use ezOWL, check the "ezOWLtab" checkbox in the dialog box that appears under the "Project->Configure..." menu item. This allows viewing of the BioPAX ontology structure in the normal Protégé tab and in the ezOWL tab.

Note: This method of importing BioPAX into Protégé prevents inadvertently made changes to the imported BioPAX classes; changing the ontology is not recommended if the instance data are meant to be shared.

To import from a local copy of the BioPAX OWL file:

- 1) Load the BioPAX OWL file via the "Project → Import..." menu item. In the resulting dialog box, select "OWL Files" and browse to the BioPAX OWL file on the local

computer disk drive by clicking on the + symbol next to “OWL file name” and press “OK”. Protégé will load BioPAX.

- 2) Upon loading, the BioPAX ontology will be visible (not grayed-out) in the classes menu.
- 3) Use the Individuals tab to create instances.
- 4) To use ezOWL, check the “ezOWLtab” checkbox in the dialog box that appears under the “Project->Configure...” menu item. This allows viewing of the BioPAX ontology structure in the normal Protégé tab and in the ezOWL tab.

Note: This method of importing BioPAX into Protégé does not prevent inadvertently made changes to the imported BioPAX classes; changing the ontology is not recommended if the instance data are meant to be shared.

Protégé can be used as a full-fledged customizable database and data entry system, although it requires programming effort. For example, Genome KnowledgeBase (<http://www.genomeknowledge.org>) uses Protégé as its backend system. If used this way, it may be desirable to modify the BioPAX ontology and create inverse slots for convenience. These slots should be removed in shared data files in order to make them compliant with the BioPAX standard.

Viewing Instances Graphically

RICE is a tool that can be used to graphically view the relationships between instances created with Protégé (<http://www.b1g-systems.com/ronald/rice/>). It requires the use of RACER, an inference engine (<http://www.cs.concordia.ca/~haarslev/racer/>). At the time of this writing, the graph features of RICE functioned properly in a Linux environment but not in Windows; other operating systems were not tested.

The RACER server must be active in order for RICE to interpret the BioPAX OWL ontology. To start RICE, use the command "java -jar rice.jar". Use the “File” menu to open a BioPAX instance file. Select the instance(s) to view and click the “ShowGraph” button. A pop-up graph should appear showing the instance(s) and the relationships to other instances (see Figure 1).

6 Use Case Outlines

These use-cases were taken into account during the design of BioPAX. Other use-cases may be suggested via the biopax-discuss@biopax.org mailing list.

Data Sharing Between Databases

One of the primary intended functions of the BioPAX format is to facilitate data exchange between existing biological pathway databases. In order for this to happen, databases must develop the ability to write-to and read-from the BioPAX format. Typically, this will require the creation of in-house software. While a number of freely available software packages may help make this task easier (e.g. Jena, an open source Java API for RDF; see <http://jena.sourceforge.net/index.html>), development of data translation software may nonetheless require a fair amount of programming time for each individual database.

The typical data transaction, i.e. passing a set of data from one database to another, will consist of a number of steps. These steps will vary depending on the particular situation, but in general they should consist of the following:

- 1) Convert a set of data into the BioPAX format. This step involves mapping the native data model to the BioPAX data model (i.e. the BioPAX ontology) and then creating a BioPAX OWL file that contains instances of the mapped classes. This step will almost always require developing software to perform the mapping.
- 2) Transfer the BioPAX file. There are many mechanisms by which this could be accomplished, e.g. the data provider could make the file available for download from an ftp or http server.
- 3) Convert the BioPAX file into the native format of the receiving database (the reverse of step 1). Again, this will likely require new software to perform the data conversion.
- 4) Merge data sets and remove redundancies. Often, many instances in a BioPAX file may already exist in the target database (Note: these are only detectable if the redundant instances share one or more unification x-refs). These instances should be merged with the existing data (if they contain additional information not present in the database) or removed from the data set being imported (if not) to prevent redundant entries from being created. Also, any pointers to such instances must be redirected to the existing database objects.

As more datasets become available in the BioPAX format, software utilities will be developed (by the BioPAX group and others) to ease data sharing. For example, a utility to integrate the data from two different BioPAX files would be useful. With such a utility, users could integrate new BioPAX data with their own by first outputting their data into BioPAX format, then running the utility to combine it with the new data, then translating the combined data set back into their own format. Thus, the need for system-specific data integration software (step 4 above) would be reduced.

BioPAX as a knowledge-base (KB) Model

The BioPAX ontology is readily usable as the data model for a pathway knowledge-base (KB) using a tool like Protégé (<http://protege.stanford.edu>). Building a new KB with the BioPAX ontology would save time and resources since it would eliminate the need to create a data schema from scratch and it would reduce the translation requirement for exporting and importing data to/from the BioPAX format (some custom semantic mapping and ID mapping might still be required to import data from another database).

Of course, some users may wish to extend the BioPAX ontology to suit their own needs. For example, many KBs use “inverse slots” – slots that are the reciprocal of other relationship slots – in order to speed up queries and facilitate browsing. Since such slots provide redundant information, they were left out of the BioPAX ontology. See the HOW-TO section for more information on creating a BioPAX KB.

Pathway Data Warehouse

The initial motivation for creating the BioPAX standard was that it was seen as a logical first step toward creating a central public repository for biological pathway data, a resource strongly desired by many members of the pathway community. If many databases provide access to their data in the BioPAX format, it should be relatively simple to aggregate this data in a central repository. With the completion of BioPAX level 1, many members of the BioPAX Group will resume work toward creating this resource.

Pathway Analysis Software

Another intended function of BioPAX is to speed development time of software that makes use of pathway data. Currently, in order for pathway software to access pathway data from multiple sources it must either be programmed to understand each different format, or the data from each source must be translated into a format that the software understands. This can require significant development time and as a consequence most pathway software is run on only a few datasets, limiting its utility.

The presence of a standard format for pathway data should alleviate this problem. With the lower barrier to data access, pathway software will be much easier to develop and apply. Also, additional software that might not be practical without an agreed upon standard, e.g. a sophisticated pathway visualization tool, may be more likely to be developed if BioPAX becomes widely adopted.

Pathway Analysis Software Example: Molecular profiling analysis

Genomics and proteomics technologies, such as gene expression microarrays and mass spectrometers, are being used to generate large datasets of molecules present at a specific place and time in an organism (molecular profiling), among other types of data. Molecular profiling experiments are often compared across two or more conditions (e.g. normal tissue and cancerous tissue). The result of this comparison is often a large list of genes that are differentially present in

the tissue of interest. It is interesting and useful to analyze these lists of genes in the context of pathways. For instance, one could look for pathways that are statistically over-represented in the list of differentially expressed genes. The result is a list of pathways that are active or inactive in the condition of interest compared to a control. The list of pathways is often much shorter than the list of input genes, thus is easier to comprehend. BioPAX documents describing pathways could be supported by tools that perform pathway-based analysis.

Visualizing Pathway Diagrams

Pathway diagrams are useful for examining pathway data. A number of formats are available for these images, but only few available viewing tools link components in the image to underlying data. A mapping of BioPAX to a symbol library for pathway diagrams (such as Kohn maps - <http://discover.nci.nih.gov/kohnk/symbols.html>) could be the basis for a general BioPAX pathway diagram tool.

Pathway Modeling

Mathematical modeling to understand the dynamics of a pathway system is a frequent use of pathway information. Qualitative modeling requires information about components in the pathway and their connections, as well as some qualitative knowledge of rates (e.g. fast, slow) and concentrations of the components (e.g. high, medium, low). Quantitative modeling additionally requires such things as measured rate constants, stoichiometry and initial concentrations in order to quantitatively predict pathway behavior. Many tools are available for this type of modeling, and the SBML (<http://sbml.org>) and CellML (<http://www.cellml.org>) standards are available to describe the models, which many tools support. While BioPAX does not contain enough information to describe a pathway model as well as SBML and CellML, there are two envisioned use cases:

Using BioPAX as metadata for SBML and CellML

SBML and CellML, as model representation languages, focus on representing the structure, parameters and mathematical description of a pathway model. BioPAX focuses on molecule and interaction classification schemes and database cross-referencing for pathway components. BioPAX and SBML or CellML could be linked together when a user wants both a full model description and information about types of pathway components and database links. A hybrid XML document containing BioPAX and SBML or CellML elements that are tied together using the CellML metadata standards could be created that fills this need.

Pathway analysis using logical inference

One advantage of representing BioPAX pathway data in OWL format is the availability of logical inferencing tools that support OWL. These tools are useful for analyzing pathways. For example, given a metabolic network model for an organism in BioPAX format, a known minimal nutrient media for that organism and the set of compounds essential for growth under one set of living conditions, then a transitive closure computation of the minimal nutrient set can be used to

verify if the metabolic network model of the organism is sufficient to explain growth. If any essential compound is not reachable through the network from the minimal nutrient list, then the network model is incomplete.

7 Glossary

Some of the following definitions may only relate to BioPAX, thus may not be general.

Biological pathway: A working definition of a pathway is a series of molecular interactions and reactions (or other biological relationships), often forming a network. For molecular pathways, the start and end points are often defined by observation of a detectable phenotype after stimulation or perturbation, such as observing gene expression after stimulating the cell with a peptide growth hormone.

Class: Used in knowledge representation to represent a category of things. A specific member of a class is called an instance.

Data exchange format: Any data format, usually electronic, used to exchange data.

GKB Editor: Generic Knowledge Base Editor. A software tool to build an ontology and manage instances of classes defined in that ontology. <http://www.ai.sri.com/~gkb/>

Instance: An instance or particular member of a class. Known as 'individual' in OWL.

Ontology: A system for describing knowledge, a conceptualization of a domain of interest usually made up of any or all of the following: concepts (classes), relations, attributes, constraints, objects, values. <http://www-ksl.stanford.edu/kst/what-is-an-ontology.html>

OWL: Web ontology language, a proposed W3C standard, is an extension of RDF to support ontologies. It provides semantics for classes and subclasses, instances, and relationships. <http://www.w3.org/TR/owl-features/>

Protégé: Protégé ontology and knowledge base editor. A software tool to build an ontology and manage instances of classes defined in that ontology. <http://protege.stanford.edu/>

RDF: Resource Description Framework, a proposed W3C standard, allows description of basic relationships between objects (subject-predicate-object semantics). <http://www.w3.org/TR/rdf-primer/>

Slot: A 'field' or 'member' of a data structure. Known as a 'property' in OWL.

Appendix A: Design Principles

Flexible: Biological pathway data are organized and represented in various ways depending on the type of data and its intended use. BioPAX must support the most frequently used representations to be widely accepted. Of course, there is a trade-off that must be considered: increased flexibility may increase data integration overhead. For example, the issue of semantic mapping between different representation styles must be dealt with when users wish to integrate BioPAX data sets that use different representations. Therefore, BioPAX should strike a reasonable balance between flexibility and rigidity by allowing multiple preferred representations and providing best practice recommendations to encourage consistent data representation.

Extensible: Biological pathway data are available in various forms and at varied levels of detail. BioPAX aims to initially support the most frequently used types of pathway data and levels of detail and to progressively broaden support for additional pathway data types and finer detail through a leveled approach. The class structure of BioPAX was designed to be extensible for this reason. Many parts of the BioPAX ontology, such as internal controlled vocabularies and many of the intermediate level classes, will be extended in future BioPAX levels. All efforts will be made to keep future levels backwards compatible.

Encapsulation: Pathway data depends on many primary databases of physical entities (e.g. proteins, small molecules, etc.). Many pathway data sets reference physical entities using database identifiers. Because of the varied nature of the physical entity databases, resolving these identifiers in a general way can be difficult, especially for the naïve user. Frequently used data about the physical entities (e.g. sequence for proteins, structure for small molecules) is optionally present (encapsulated) in the BioPAX format for convenience.

Compatible: BioPAX uses existing standards for encoding biological pathway information to avoid “re-inventing the wheel”. Specifically, pointers to the Gene Ontology (GO), and instances of Chemical Markup Language (CML) and the SMILES format are used in various slots in the ontology. Also, compatibility with other pathway standards, such as SBML, CellML, and PSI-MI has influenced the design of many BioPAX features.

Computable: BioPAX stores data in a format that supports many different types of computational analysis. Values are strongly typed and the class structure is clearly defined. A wide range of computational tasks, from simple reading and parsing of a BioPAX file to logical inference based on the data, are supported. The OWL version of the BioPAX ontology complies with the OWL-DL sublanguage and is thus interpretable by description logic software such as RACER (<http://www.sts.tu-harburg.de/~r.f.moeller/racer/>) and should be compatible with future software built to work with the Semantic Web.

Appendix B: Level and Version Numbers

BioPAX level numbers indicate the relative scope of the ontology. BioPAX Level 1 focuses on metabolic pathway data; subsequent levels will expand this scope to include other types of data such as molecular binding interactions and signal transduction pathways. BioPAX level numbers are always whole numbers (e.g. Level 1, version 1.0).

In addition to the level numbers, BioPAX version numbers indicate the relative stage of development of each level. Version numbers are a composite of two or three individual integers: the major version number, the minor version number, and, optionally on beta versions, a revision number. All three numbers are separated by decimal points to form the composite version number (e.g. Level 1, version 1.1.1). The major version number appears before the first decimal point and is only incremented when an update is likely to affect existing data. Releases in which the major version is 0 are early draft releases of their respective levels (e.g. Level 1, version 0.5).

The minor version appears after the first decimal point and is incremented when an update is unlikely to affect data that conforms to the prior version. Odd minor version numbers indicate beta versions, while even minor version numbers indicate release versions.

The revision number is optional and may only appear on beta versions (i.e. where the minor version is an odd number). It appears after the second decimal point and is incremented with each new revision of the beta version.

For example, the first non-draft release of every level is version 1.0 (major version 1, minor version 0). If this version would need to be updated, the first beta version of the update would be called version 1.1. Subsequent revisions of this beta version would be called versions 1.1.1, 1.1.2, 1.1.3, etc. When no further revisions were needed, a release version of the update would be created called version 1.2 if the revisions would not affect data that complied with version 1.0, or version 2.0 if they would.

All versions of BioPAX are available in the following directory on the BioPAX website:
<http://www.biopax.org/Downloads/>

The most recent major versions of each level of BioPAX are always available in this directory:
<http://www.biopax.org/release/>

Acknowledgements

The BioPAX workgroup thanks members of the community who have contributed to this work through discussions: Melissa Cline, Autumn Cuellar, Emek Demir and the PATIKA group, Andrew Finney, Ken Fukuda, Matt Halstead, Mike Hucka, Stan Letovsky, Peter Murray-Rust, the PSI-MI workgroup and others who have contributed through involvement in the biopax-discuss list, seminar participation and birds of a feather (BOF) sessions at conferences.

References

1. Baxevanis, A. D. & Ouellette, B. F. F. *Bioinformatics : a practical guide to the analysis of genes and proteins* (Wiley-Interscience, New York, 2001).
2. Alberts, B. *Molecular biology of the cell* (Garland Science, New York, 2002).
3. Stein, L. Creating a bioinformatics nation. **417**, 119-120 (2002).
4. Hermjakob, H. et al. The HUPO PSI's molecular interaction format--a community standard for the representation of protein interaction data. *Nat Biotechnol* **22**, 177-83 (2004).
5. Karp, P. D. et al. The EcoCyc Database. *Nucleic Acids Res.* **30**, 56-58 (2002).
6. Krieger, C. J. et al. MetaCyc: a multiorganism database of metabolic pathways and enzymes. *Nucleic Acids Res* **32 Database issue**, D438-42 (2004).
7. Bader, G. D., Betel, D. & Hogue, C. W. BIND: the Biomolecular Interaction Network Database. *Nucleic Acids Res.* **31**, 248-250 (2003).
8. Overbeek, R. et al. WIT: integrated system for high-throughput genome sequence analysis and metabolic reconstruction. **28**, 123-125 (2000).
9. Demir, E. et al. PATIKA: an integrated visual environment for collaborative construction and analysis of cellular pathways. **18**, 996-1003 (2002).
10. Lemer, C. et al. The aMAZE LightBench: a web interface to a relational database of cellular processes. *Nucleic Acids Res* **32 Database issue**, D443-8 (2004).
11. Kanehisa, M., Goto, S., Kawashima, S., Okuno, Y. & Hattori, M. The KEGG resource for deciphering the genome. *Nucleic Acids Res* **32 Database issue**, D277-80 (2004).
12. Hucka, M. et al. The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics* **19**, 524-31 (2003).
13. The_Gene_Ontology_Consortium. Gene ontology: tool for the unification of biology. **25**, 25-29 (2000).
14. Murray-Rust, P. & Rzepa, H. S. Chemical markup, XML, and the World Wide Web. 4. CML schema. *J Chem Inf Comput Sci* **43**, 757-72 (2003).
15. Weininger, D. SMILES, a Chemical Language and Information System. **28**, 31-36 (1988).
16. Karp, P. D. Database links are a foundation for interoperability. *Trends Biotechnol* **14**, 273-9 (1996).
17. Wheeler, D. L. et al. Database resources of the National Center for Biotechnology. *Nucleic Acids Res.* **31**, 28-33 (2003).