BRIEFINGS IN BIOINFORMATICS. VOL 9. NO 3. 220–231
Advance Access publication January 31, 2008
doi:10.1093/bib/bbn003

Interoperability with Moby 1.0—It's better than sharing your toothbrush!

*The BioMoby Consortium**

Submitted: 16th November 2007; Received (in revised form): 2nd January 2008

Abstract

The BioMoby project was initiated in 2001 from within the model organism database community. It aimed to standardize methodologies to facilitate information exchange and access to analytical resources, using a consensus driven approach. Six years later, the BioMoby development community is pleased to announce the release of the 1.0 version of the interoperability framework, registry Application Programming Interface and supporting Perl and Java code-bases. Together, these provide interoperable access to over 1400 bioinformatics resources worldwide through the BioMoby platform, and this number continues to grow. Here we highlight and discuss the features of BioMoby that make it distinct from other Semantic Web Service and interoperability initiatives, and that have been instrumental to its deployment and use by a wide community of bioinformatics service providers. The standard, client software, and supporting code libraries are all freely available at http://www.biomoby.org/.

Keywords: *semantic web; web services; interoperability; data integration; biomoby; schema*

INTRODUCTION

Discovery of, and easy access to, biological data and bioinformatics software is the critical bottleneck for systems biologists, resulting in missed scientific opportunities and lost productivity due to expensive and unsustainable efforts in data warehousing, or the design of *ad hoc* and transient Web-based analytical workflows. Workflow-design itself is neither trivial nor reliable for most systems biology researchers since, often, a high level of prior-knowledge and understanding of available Web-based resources is required from the biologist. Indeed, in his article 'Creating a Bioinformatics Nation' [1], Lincoln Stein suggests that it is the lack of interoperable standards that has hindered the integration of scientific datasets worldwide. Conversely, in her keynote address to the EGEE '06 conference, Carole Goble purposely misquoted Michael Ashburner [2] when she stated 'Scientists would rather share their toothbrush than their data!' These statements highlight the two somewhat opposing requirements that must be considered when designing interoperable systems for the bioinformatics domain. On one hand, the bioinformatics service provider community is composed of individuals with a wide variety of different expertise, thus any interoperability proposal must be limited in complexity and must focus on comprehensibility to non-computer-scientists; on the other hand, the functionality gained by participating in the interoperability framework must be sufficiently compelling for individual providers to be willing to openly share data that is, in some cases, personally precious. These considerations were key in establishing the technologies and practices defined by

*Full authorship: Mark D Wilkinson, Martin Senger, Edward Kawas, Richard Bruskiewich, Jerome Gouzy, Celine Noirot, Philippe Bardou, Ambrose Ng, Dirk Haase, Enrique de Andres Saiz, Dennis Wang, Frank Gibbons, Paul M.K. Gordon, Christoph W. Sensen, Jose Manuel Rodriguez Carrasco, José M. Fernández, Lixin Shen, Matthew Links, Michael Ng, Nina Opushneva, Pieter B.T. Neerincx, Jack A.M. Leunissen, Rebecca Ernst, Simon Twigger, Bjorn Usadel, Benjamin Good, Yan Wong, Lincoln Stein, William Crosby, Johan Karlsson, Romina Royo, Iván Párraga, Sergio Ramírez, Josep Lluis Gelpi, Oswaldo Trelles, David G. Pisano, Natalia Jimenez, Arnaud Kerhornou, Roman Rosset, Leire Zamacola, Joaquin Tarraga, Jaime Huerta-Cepas, Jose María Carazo, Joaquin Dopazo, Roderic Guigo, Arcadi Navarro, Modesto Orozco, Alfonso Valencia, M. Gonzalo Claros, Antonio J. Pérez, Jose Aldana, M. Mar Rojano, Raul Fernandez-Santa Cruz, Ismael Navas, Gary Schiltz, Andrew Farmer, Damian Gessler, Heiko Schoof, Andreas Groscurth.

Corresponding author. Dr Mark Wilkinson, Room 166, 1081 Burrard St. The Heart and Lung Research Institute at St. Paul's Hospital, Vancouver, BC, Canada, V6G 1Y3. Tel: +1 604 682 2344 ×62129; Fax: +1 604 806 9274; E-mail: markw@illuminae.com

BioMoby is a project within the larger Open Bioinformatics Foundation. The BioMoby Consortium consists of more than 40 participants spanning 13 nations, and participation is free and open to all.

the BioMoby project [3–5]. Now, with the release of the 1.0 version of the BioMoby Application Programming Interface (API) and supporting code-bases and end-user applications, it is useful to examine the successes and failures of the BioMoby project as it explored this question. We intend this manuscript and the Supplementary Material provided with it, to be a comprehensive and canonical description of the defining features of BioMoby, and its behaviors.

An example of the utility of BioMoby for the biologist

The semi-fictitious story below describes one example of the type of day-to-day data exploration activities undertaken by biologists. The difficulty they experience in pursuing these activities, due to the large amounts of data and the disparity between resource interfaces, provided the motivation for BioMoby's invention and development. The workflow described below does, in fact, exist and is currently being prepared for publication elsewhere.

> *Dr. Davies is an Antirrhinum (Snapdragon) researcher. He is studying a new class of mutations but has so far been unable to clone any of the loci. Moreover, he is constantly frustrated by the lack of a complete Antirrhinum genome sequence, though there are a large number of mapped mutations and ESTs. The taxonomically closest sequenced model organism is Arabidopsis, however there are no explicit links between the Arabidopsis data in The Arabidopsis Information Resource (TAIR), and the Snapdragon data in DragonDB. In an attempt to bootstrap his cloning efforts, he decides to look-up which loci from Arabidopsis have mutant phenotypes that share characteristics with his loci of interest; whether or not these have been mapped; or if there may be homologous ESTs from Snapdragon available for him to attempt a co-segregation analysis. He knows that both TAIR (in the USA) and DragonDB (in Germany) have provided many of their resources as BioMoby services, so he begins. He first asks Moby Central if DragonDB provides keyword phenotypic lookup, which it does. With a single click, he has gathered the list of loci matching his phenotypic criteria. He then asks the same question from TAIR, and with a single click has gathered all matching Arabidopsis loci. BioMoby alerts him that TAIR can provide the sequences for these loci if he wishes, and in a single click he has retrieved all of these sequences. BioMoby then alerts him that DragonDB is capable of executing a BLAST analysis on those sequences, and with a single click he sends all sequences into the BLAST service. The returning Blast reports contain a myriad of "hits", and he becomes concerned that he may need to do a large number of look-ups; however BioMoby alerts him that DragonDB provides a Blast parsing service that will extract the "hits", and he selects this option. From the resulting list of "hits", he asks BioMoby to retrieve the map locations for these sequences. In addition, he queries if any services are capable of transforming those sequence IDs into their associated locus IDs, and such a service is automatically discovered and executed for him. With this list of Antirrhinum Locus IDs resulting from the Blast search, he then requests that it be cross-referenced with the list of Antirrhinum locus IDs resulting from the keyword search. BioMoby suggests a set-intersection service available from the iCAPTURE Centre in Canada, and with a single click he has now gathered the list of loci that share both phenotypic and sequence similarity. BioMoby suggests that he might also wish to retrieve photographs of the associated mutants, and with a single click he has retrieved these images. From this complex but filtered set of data, gathered within just a few minutes, he begins a biological assessment of whether any of his genes of interest have been mapped and/or sequenced.*

Workflows such as the one described above could be constructed, utilized, visualized and executed using a wide range of BioMoby-enabled end-user applications, and it is beyond the scope of this manuscript to describe these interfaces in any detail; however a series of screenshots have been made available in the Supplementary Material Section 5 where a similar workflow is constructed and executed using the Gbrowse-Moby end-user interface. These screenshots demonstrate one way in which BioMoby could be used to suggest next-steps in an analytical pathway, and how it facilitates the automatic execution and visualization of those analytical or exploratory results.

Web services overview

The Web Services model is a framework for communication between computer applications over the World Wide Web [6]. Traditionally, they expose Web-based application interfaces in the form of a Web Services Description Language (WSDL) document [7] describing the input(s), output(s), function and location of a Web Service. The limitation of traditional Web Services lies primarily in that, while the WSDL interface definition is machine-readable, the *meaning* of the input and output, and the *intent* of the operations that are being executed to derive that output—the 'semantics' of the service—are opaque to the machine accessing it. The barriers posed by these limitations are further evidenced by a recent candidate specification for the semantic markup of WSDL documents [8]. Currently, therefore, the creation of meaningful

workflows requires manual intervention to first select appropriate services and then to accurately map the output of one service into the input of the next; automated service composition is an error-prone computational task [9–14]. At least part of the limitation results from traditional Web Services consuming and producing their data in the form of Extensible Markup Language (XML) documents [15]. Until recently [16], there have been no attempts to standardize the schemas of these XML documents in the bioinformatics domain, and thus software had to be specifically developed to access each Web Service, by individuals familiar with the Service interfaces. This software was generally task-specific, and needed to be re-written for each new analysis.

To overcome this limitation, the BioMoby framework defines an extended set of formats and conventions that allows the creation of 'Semantic Web Services'. Semantic Web Services have interfaces defined and/or annotated with terms grounded in ontologies. As such, it is possible to create software capable of utilizing the knowledge in these ontologies to support fully- or semi-automatic service discovery and workflow composition [17]. Of the three Semantic Web Services projects in widespread use—myGrid, caBIO and BioMoby (reviewed, compared and contrasted in [18, 19]), BioMoby is unique in its utilization of ontologies to define not only the biological intent and/or semantics of the data that are passed into and out of a service, but also to define the *syntax* of that data. In much the same way that the HTML standard syntax made it possible to develop generic Web browsers, standards for Web Service representation (data-types, data syntaxes and interface functional annotations) such as those developed in the BioMoby initiative are enabling the development of generic Semantic Web Service browsers [20]. Semantically-enhanced Web Services are more interoperable, easier to pipeline together, more semantically transparent, and will empower the citizens of the bioinformatics nation, allowing them to share their data more intuitively [21].

RESULTS

Stylistic conventions

Here, we represent ontological class names using **Capitalized Bold**, ontological class properties using `fixed-width font` and ontological class relationships using ***bold italics***.

Framework overview

This article describes the stable 1.0 version of the BioMoby Semantic Web Service specification; the culmination of 6 years of framework evolution and revision based on early adopter's feedback. The BioMoby interoperability framework extends and modifies the core Web Services specification by further defining:

- An end-user-extensible, ontology-based data representation syntax (Object Ontology).
- An end-user-extensible ontology of data domains (Namespace Ontology).
- An end-user-extensible ontology of Web Service operational descriptions (Service Ontology).
- A predictable Web Service message structure, including explicitly defined locations and formats for provision of metadata and cross-referencing information, as well as structured and machine-interpretable error messages.
- A Web Service registry in which all service interface definitions are represented in terms of the above ontologies, and which is able to utilize these ontologies to aid discovery of task-appropriate services.

These features work together to enable the development of generic software systems that can interact with myriad, diverse bioinformatics data and analytical tool providers. The biologist using that software requires little or no knowledge of the existence of a tool, of the kinds of resources it provides, or of the specific user interface through which it functions [20, 21]. It is worth noting that, although the three ontologies 'define' various bioinformatics concepts, that they are world-editable and constantly evolving. As such, they 'define' concepts based on the community's consensus at any given time, but are constantly adapting to new ideas, new resources and new data-types as they arise in the community.

The namespace ontology—'What data are we talking about?'

There is little consensus in the bioinformatics community around how to identify records. Often, records are simply numbered, and this requires contextualization to imbue any meaning. To assist

with this contextualization, these numeric identifiers are sometimes prefixed, for example GO:0003487 for a Gene Ontology (GO) term, or gi|163483 for a GenBank record; however this is not done consistently or reliably by all resources, nor is the separator between the prefix and the identifier consistent between different resource providers. For the biologist, this inconsistency can make it difficult to locate records in the wide variety of interfaces available to them, and can lead to problems with integrating data from multiple sources that may use different conventions for representing the same record identifier. The BioMoby Namespace Ontology is an attempt to resolve this inconsistency such that data records are unambiguously and predictably identified in the datasets returned to the biologist.

The Namespace Ontology (currently a simple, flat controlled vocabulary) defines all valid data 'namespaces'—the underlying source of a given data record—in the BioMoby system. Examples include KEGG_ID for KEGG records or NCBI_gi for GenBank records. There are over 300 different BioMoby Namespaces ranging from the most prominent public resources such as PubMed, to lesser known resources such as DragonDB. The Namespace Ontology is, in fact, an extension of the Cross-reference abbreviations list [22] from the Gene Ontology consortium [23]. New resources that wish to participate in the BioMoby framework simply register the namespaces they consume and/or generate in the Namespace Ontology, and any BioMoby service provider can then interpret the underlying source of data passed in that namespace.

The combination of namespace and identifier are unique to each piece of data. Since not all data is identified—for example, some data exists only transiently during the process of an analysis—use of a namespace is not always required in the BioMoby framework.

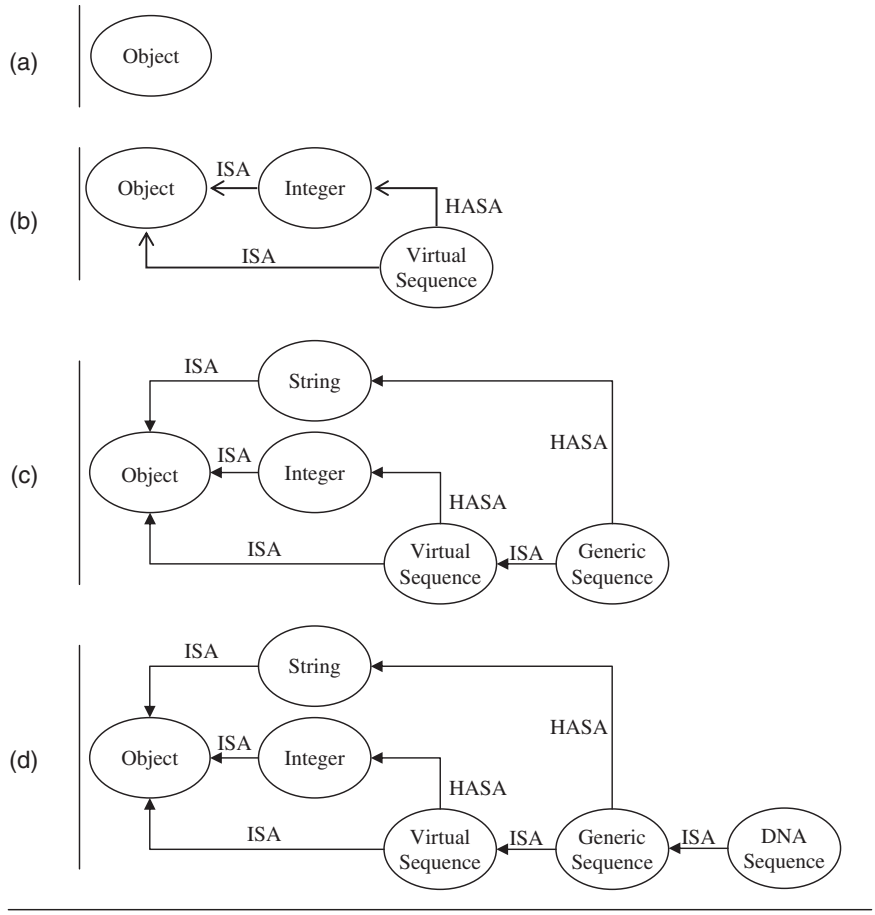
The object ontology—'How is that data represented?'

For the biologist, data is often presented to them in *ad hoc* formats, or in formats that are governed by a visual layout such as a web-page. This diversity of data formats often forces biologists to undertake copy/paste operations to extract data from their query results into their local database or spreadsheet. The purpose of the Object Ontology from the perspective of the biologist is to create a consistent, machine-readable data syntax such that the transformation of data from one common format to another, and the integration of that data, can be automated.

The Object Ontology's structure was designed to resemble that of the GO, due to the elegant simplicity of GO, and the familiarity and acceptance of it within the target community. Like GO, the Object Ontology is an asserted subclass ('***is-a'***) hierarchy, and includes two additional partite relationships ('***has-a'*** and '***has'***) representing parts in cardinality 'one', or parts in cardinality 'zero or more', respectively. The root class of the ontology—**Object**—possesses three properties—`namespace`, `id` and `articleName`—and is designed to represent record identifiers ('ID numbers') from well-known resources (e.g. GenBank, EMBL, GO, etc) in a well-defined and predictable manner. The value of the `namespace` property is a member of the Namespace Ontology, the value of the `id` property is the record-identifier within that resource, and the value of the `articleName` property indicates (as a human-readable phrase) the semantic nature of the relationship between a given class and a class that is in a ***has*** or ***has-a*** relationship to it. Figure 1A shows a small portion of the Sequence-branch of the BioMoby Object Ontology, revealing how these various components are used to construct new and more complex classes. Figure 1B shows the XML representation of specific cases ('instances') of these ontological classes. A more complete description of inheritance between classes in the Object Ontology, and derivation of the XML representation of instances of these ontological classes, is presented in Section 2 of the Supplementary Material.

Perhaps the most important aspect of the Object Ontology is that it is end-user extensible. Any BioMoby service provider can create a new Object class by simply registering its definition in the Object Ontology in code via the Moby Central API or through a freely available graphical 'BioMoby Dashboard' application [24]. The new Object's definition includes a human-readable explanation of the purpose of the data-type, and a technical description of how this data-type relates to existing data-types in the ontology. Thus, machines receiving this novel data-type as part of a Web Service transaction need only look-up the data-type in the Object Ontology to determine its syntax. Moreover, since all sub-components of all data-types are themselves BioMoby Objects, generic re-usable software is capable of extracting and/or assembling the data

A



B

(a)
```
<Object namespace='NCBI_gi'id='111076' articleName=''/>
```

(b)
```
<VirtualSequence namespace='NCBI_gi'id='111076' articleName=''>
    <Integer namespace='' id='' articleName="Length">38</Integer>
</ VirtualSequence>
```

(c)
```
<GenericSequence namespace='NCBI_gi'id='111076'>
    <Integer namespace='' id='' articleName="Length">38</Integer>
    <String namespace='' id='' articleName="SequenceString"><![CDATA[
            ATGATGATAGATAGAGGGCCCGGCGCGCGCGCGCGC
    ]]></String>
</ GenericSequence>
```

(d)
```
<DNASequence namespace='NCBI_gi'id='111076'>
    <Integer namespace='' id='' articleName="length">38</Integer>
    <String namespace='' id=''articleName="SequenceString"><![CDATA[
            ATGATGATAGATAGAGGGCCCGGCGCGCGCGCGCGC
    ]]></String>
</ DNASequence>
```

Figure I: Sequential construction of complex objects in the BioMoby Objects Ontology, and the corresponding XML serialization of their instances. (**A**) The creation of BioMoby Object Classes starting from the root object 'Object' (a), to a VirtualSequence (b) which inherits from Object and has-a Integer (Length), to a GenericSequence (c) which inherits from VirtualSequence, and adds a String (SequenceString) through the has-a relationship, and finally a DNASequence (d) which simply inherits from and thus further specializes the GenericSequence Object semantically. (**B**) The serialization of the objects (a – d) from above. The outermost XML tag is the ontological class name. Child tags are added by the has or has-a relationships (b), or are inherited from parent classes (c). Specialization of an existing class (d) simply changes the outermost tag name.

components of any possible BioMoby object, including objects that did not exist when that software was created. The ability to create generic object parsers and assemblers significantly reduces the software's anticipated legacy problems and update/patch-cycles.

The Object Ontology currently consists of over 300 different data syntax definitions, including many of the common legacy flat-file formats, as well as novel objects that have been constructed *de novo* by participating service providers.

The service ontology—'What types of things can I do with this data?'

Biologists are faced with thousands of analytical tools both in their local applications as well as on the Web. Many of these tools are redundant and/or do very similar tasks, and it is the responsibility of the biologist to know (i) that a tool exists; (ii) what it is called in order to locate it and (iii) what kinds of operations that tool is capable of performing. The Service Ontology is an attempt to organize bioinformatics tools into a categorization system, such that tools of similar function are grouped together, and can be discovered by the biologist using a consistent naming system.

The Service Ontology is a simple, asserted subclass (***is-a***) hierarchy that defines a set of data manipulation and/or bioinformatics analysis types. These include classes such as **Retrieval** for retrieval of records from a database, **Parsing** for the extraction of information from various flat-file formats, or **Conversion** for data-type syntax changes. Sub-classing is used to define more precise types of service operation. For example, an instance of a BLAST Service may have service type **Pairwise_Sequence_Comparison**, which is a sub-class of **Analysis**. The BioMoby Service Ontology serves a purpose similar to the Bioinformatics Task Ontology from the myGrid project [25].

BioMoby Web Services—'What resources are out there?'

When interacting with analytical tools, biologists are often required to learn a new interface for each tool they wish to use, and the lack of standardization results in different layouts even for functionally-identical tools. Moreover, most Web-based tools are not amenable to bulk-uploads or automation, since the interfaces are designed specifically for human end-users. BioMoby Web Services attempt to standardize these interfaces through defining a machine-readable messaging structure that can be utilized by all analytical tools, and through utilizing the Object Ontology to define the syntax of the input and output data. Thus, the interaction between the biologist and any analytical tool exposed as a BioMoby service can be accomplished through a single, common interface.

BioMoby Services, for example a database lookup, a ClustalW alignment tool, or a BLAST report parser, are globally distributed and perform a single operation each. Input and output messages follow a well-defined message format (described in Section 3 of the SupplementaryMaterial). Services consume one or more instances of an Object within this message; they execute a single operation on that Object as described by an appropriate Service Ontology term; and finally, the output is returned to the caller as one or more instances of another Object within a well-defined output message structure. These details are registered in the BioMoby Central Service registry, along with the service endpoint (URL + service name) and a textual description of the service function for the end-user. There is library support in Java, Perl and to a more limited extent Python, to support the extraction of input data from BioMoby messages, and to construct appropriate output messages. As such, the service provider's primary concern is the business logic of their Service, with none to only modest additional code required. To be compliant with existing standards, service providers can report a wide variety of error conditions in a standardized way using the Life Sciences Analysis Engine (LSAE [26]) framework. Software designed to minimize service provider effort in setting up new Services is available for both Java (MobyServlet [27], MoSeS [28]) and Perl (MoSeS).

Unlike other successful Web Service interoperability systems [29], BioMoby services are standalone, and are not overtly designed to be inter-dependent; there is no over-arching BioMoby standard defining what types of Services can exist, what functions they must provide, or how they will interoperate. Moreover, Services are highly modular, each executing a single straightforward function (e.g. record retrieval, or file parsing), such that a service provider approaching BioMoby for the first time can have a simple compliant Service running within minutes. The service provider can therefore gradually migrate their host resources, piecemeal, into the BioMoby framework over time or re-present their

existing resources in parallel. Complex operations in BioMoby are achieved by chaining together multiple Services, or running multiple Services in parallel to extract the individual pieces of data required, and this is well-supported by existing client applications such as Taverna [17].

The BioMoby central registry—'How do I find the resource provider I want?'

'Moby Central' is a registry for BioMoby-compliant Web Services. For the biologist, it acts as a 'search engine', helping them discover all resource providers capable of executing the operation they wish to undertake. Moreover, the search can be made 'context sensitive', such that only those providers who can operate on the data that the biologist has in-hand are discovered.

Moby Central provides a Simple Object Access Protocol (SOAP [30]) based API that allows addition of new Services to the registry, removal of Services from the registry and searching over registered Services in a variety of ways. Importantly, the registry is aware of all three BioMoby ontologies, and can thus optionally utilize the semantics embedded in these ontologies to enhance search success. For example, searching for Services that generate **b64_encoded_GIF** would, if semantic searching were enabled, also discover BioMoby Services that generated the more complex **annotated_b64_encoded_GIF** through traversal of the Object Ontology towards its leaf nodes. Similarly, and perhaps more importantly, searching for Services that consume specific, sometimes very complex data-types, for example an **annotated_FASTA** object, would also discover Services that consumed the more simplistic **FASTA** objects, or even base **Object** (i.e. a simple ID number) through traversal of the Object Ontology towards its root. Complex or provider-specific data-types therefore do not (necessarily) thwart automated discovery of Services that can consume that data, since the ontologies allows the registry (or the client) to infer the semantics of that data-type and thereby infer which Services can operate on it. Moreover, the ontologically-governed XML schema that is used to represent data in BioMoby allows service discovery based on any sub-component of a data-type. For example, a **MultipleAlignment** contains several instances of **GenericSequence**, each representing one of the aligned sequences. A generic client application can reliably decompose the **MultipleAlignment** object and use the **GenericSequence** objects in queries to Moby Central to discover Services that operate on them.

Summary

Through adoption of these extensions to traditional Web Services, it is possible to design software systems that enable bench scientists and other non-programmers to automate the discovery and connection of independent Web Services into large analytical pipelines without any task-specific tooling, nor any deep understanding of BioMoby, Web Services or any of the individual BioMoby Web Service interfaces. Generic workflow environments such as Taverna [17, 31], MOWServ [32], Remora [33], Gbrowse-Moby [20], Bluejay [34] and Seahawk [35] can (and do) suggest, and automatically connect, appropriate Web-based resources into complex pipelines without requiring any technical knowledge by the end-user. Rather, they rely on the expert knowledge of the biologist to select appropriate Services from the limited number of suggestions provided through queries to the BioMoby registry based on their stated requirements. Currently, more than 40 data and/or analytical service providers worldwide are using the BioMoby interoperability framework to provide over 1400 interoperable Services, and this number continues to grow almost daily.

DISCUSSION

BioMoby has made several key decisions, which distinguish it from other prominent Web Service frameworks in the bioinformatics domain, and result in the interoperable behaviors observed when using it in practice. Some of these decisions are part of the BioMoby specification, while others have simply arisen as a community-consensus on best practices for Web Service provision.

Closed world

The first distinguishing feature of BioMoby is that it operates in an extensible, but closed-world of data semantics. The XML Schema within a traditional WSDL document defines valid XML tags for any given service, but these are not (predictably) bound to any standard external machine-readable *interpretation*. Thus the XML tags, and the content of these tags, from one Web Service are not reliably compatible with the XML tags or content from

another arbitrarily chosen Web Service. As a result, automated pipelining of non-coordinated services is extremely difficult using traditional Web Services. In contrast, the Object, Namespace and Service Ontologies provide a common binding for all services and client software in the BioMoby framework such that a given XML tag appearing in any BioMoby message has one and only one interpretation, and this interpretation is available for automated look-up through shared ontologies. In this way, BioMoby finesses the extremely complex problem of open-world Web Service composition by defining the allowable world of data syntax and semantics via publicly extensible ontologies.

Nevertheless, it can equally be argued that operating in a closed world is artificial, unsustainable and overly-limiting. In constraining itself to its three boutique ontologies, BioMoby does not natively take advantage of the wealth of knowledge captured in third-party ontologies such as those provided by the Open Biological Ontologies (OBO) [36] consortium. Indeed, a partner project that has branched-off from the original BioMoby project is the Simple Semantic Web Architecture and Protocol (SSWAP [37]). SSWAP proposes to use an open world of data semantics, relying on third-party ontologies to define the nature and syntax of the inputs and outputs of Web resources, and defining only a minimal messaging structure within the project itself. SSWAP has shown exciting early success in achieving interoperability between a small number of participating providers. It remains to be seen, however, if the complexity of reasoning over an open-world system, and/or the potential dilution of compatibility between resources due to the increasing number of ontological possibilities, will interfere with the desired goal of straight-forward, maximal interoperability between bioinformatics Web resources.

Modularity

The second distinguishing feature is that BioMoby services tend to be extremely lightweight, highly modular and execute very fine-grained operations on incoming data. This was not a behavior mandated within the project specification; however it has become a convention among the majority of BioMoby service providers. This may be because it is more straight-forward and/or is more advantageous to do so (i.e. promoting code re-use at the level of the Service, rather than duplicating functionally similar code fragments between sets of similar but more complex Services), but it is also at least in part due to the constraints arising from the simplistic Moby ontologies.

Data-types defined in the Object Ontology tend to be quite straightforward, seldom merging more than two or three related data elements into any given Object. Contrast this with other commonly used Web Service systems such as the NCBI e-Utilities [38]. Input of a gene identifier, for example [39], to the e-Fetch Web Service returns an XML document containing a single record of 250 kb that includes 120 distinct XML tags ranging from organism and taxonomy information to detailed gene structure, cross-references and even PubMed identifiers and GO terms. While this is efficient and likely useful for software applications that have been designed specifically to utilize e-fetch data, the nonspecific 'give me everything' operation that happens within e-Fetch Web Services is difficult to semantically describe, and therefore hard to integrate using any 'generic' Web Service software. The extreme modularity of BioMoby services reduces message size in many cases, reduces computational load, simplifies service description, enables the creation of generic parsers and yet allows retrieval of arbitrarily complex data-sets through a 'Lego block' approach of combining operations of high granularity.

Modularity of services has another more important consequence in simplifying service discovery. An operation performed by a given BioMoby service must be unambiguously described by a *single term* from the Service Ontology (e.g. **Parsing**). This severe restriction forces service providers to make their services highly granular. At this level of granularity, the semantics of a service become nearly transparent, with the intent of making automated discovery of appropriate or desired services easier and more accurate. In practice, however, the Service Ontology is hopelessly insufficient to adequately describe many even straightforward services built within the BioMoby framework. A Web Service can seldom be described in a single word or phrase, and thus many service providers put the full semantics of their service functionality into the human readable service name and description. This creates a barrier to fully-automated service pipeline composition; nevertheless, despite being the most obvious weaknesses of the BioMoby system, users (bioinformaticians and biologists) seem comfortable choosing an analytical strategy from the limited set of sensible possibilities

Table I: Comparison of the features of the BioMoby Object Ontology versus that of an OWL ontology and their respective instances

Feature	OWL/RDF	BioMoby objects	W3C XML schema
Declared classes	Yes	Yes	Sort of
Classes have class properties	Yes	Yes	Sort of
Classes have literal properties	Yes	Yes	Sort of
Extensible class definitions	Yes	Yes	Sort of
Heritable class definitions	Yes	Yes	No
Reasoning over instances based on asserted ontological subclasses	Yes	Yes	No
Reasoning over instances based on instance properties	Yes	No	No

A comparison with XML Schema is also included to show the gains achieved by moving towards ontologically-based data structures.

presented to them through more general registry queries, rather than having the precise choice made for them by the system itself.

Extensibility

The fact that the closed-world of BioMoby ontologies is end-user extensible was, we believe, critical to its adoption by a community that embraces open-world behaviors. While BioMoby encourages consensus on data models, it does not dictate them; end-users are allowed to construct, register and use alternative data models as they see fit (though they limit their interoperability with other resources by doing so).

Giving end-users the ability to define their own data-types, service-types, and namespaces was considered a risky approach in the early days of the project, particularly since ontology-development has historically been undertaken by a curation team of domain experts [40]. However, in the past 5 years, the Object ontology has only required significant curation twice, and one of these was to accommodate a change in the core BioMoby API. No 'organized' curation has been done on either the Namespace or Service ontologies. As such, the BioMoby ontologies, while imperfect, have required no centralized investment of time or money, and are largely self-curating through an open and public API. The open model takes advantage of the 'passive altruism' of a collaborative community of providers acting on their shared desire to enhance interoperability for their own individual purposes.

BioMoby versus peer semantic and schema technologies

The development of BioMoby was influenced significantly by the growth in popularity of Web Services, WSDL and XML Schema in 2001/2002, but developed independently of, and was largely uninfluenced by, the emergent Semantic Web activity taking place within the World Wide Web Consortium (W3C) between 2001 and 2004. Thus, it is interesting to compare the semantic aspects of various peer technologies, and examine where the benefits and/or limitations of each approach might lie.

BioMoby versus W3C XML Schema

The W3C XML Schema (XSD) specification describes the structure of an XML document, but not its intent or its semantics. In this sense, as described in Table 1, XSD are class-like, but are not grounded in a semantic definition of what that a class 'means', and thus are semantically opaque to software applications. Other limitations of XSD are that, while XSD are modular (in that it is possible to refer to an external or third-party XSD fragment from within a local Schema document) XSD are not natively extensible; inheritance is not part of the XSD specification, and one must use non-standard extensions [41] to describe the semantic relationship between embedded schema fragments. Projects such as HOBIT [16] are attempting to add more semantics into XSD by providing a universal grounding for a curated set of XSD such that both the intent and the syntax are shared by all consumers and providers. Unfortunately, this is occurring through manual curation, and is limited only to the 'outer-most' element of the XML document; embedded XML tags, while representing real-world identifiable data-types, are not included in this semantic annotation, and thus cannot be utilized in a generic way to construct novel inputs to downstream Web Services, as can be achieved by 'decomposing' BioMoby Objects. BioMoby achieves schema specification through its Object Ontology; however, the sub-classes and embedded classes within these schema are

grounded in the same Ontology and the semantic relationships between embedded classes are indicated by the `articleName` property. Thus many of the limitations, in particular the lack of heritability and the semantic opacity of XML Schema-based data definitions, are overcome within the BioMoby data typing framework.

BioMoby versus OWL/RDF

In early 2004, the World Wide Web Consortium (W3C) formally announced the two core Semantic Web technologies: Web Ontology Language (OWL [42]) and Resource Description Framework (RDF [43]). OWL is an abstract language for defining classes and their properties. Many OWL ontologies are 'decidable', meaning (simplistically) that a Description Logic (DL) reasoner (FaCT++ [44], RACER [45], Pellet [46]) can computationally infer both the internal consistency of the ontology as well as computationally classify instance data to be members of a particular OWL class(es). RDF is an abstract language for describing resources as subject-predicate-object graphs. Resources described in RDF can be grounded as instances of an OWL class definition either by direct assertion or by computationally inferred DL-based classification. OWL ontologies can be represented in RDF, and RDF has a defined serialization into XML (called RDF-XML), which are among the most common representation formats for both OWL and RDF on the Semantic Web.

Although the early releases of BioMoby preceded the W3C's formal announcement of OWL/RDF by several years, BioMoby nevertheless exhibits many of the behaviors that are expected from the emergent Semantic Web, such as automated discovery of appropriate resources, interoperability between them, and the ability to automatically compose and decompose data types in novel and meaningful combinations. In fact, the BioMoby framework resembles the OWL/RDF duo of Semantic Web technologies in several key respects, and these are detailed in Table 1.

The 'semantics' of ontologically-based systems arise in three ways. The first is through the human-readable definition of the class, which can be used as grounding for all software that utilizes that ontology. In this aspect, OWL, BioMoby, and many of the other ontologies in the bioinformatics domain (e.g. most of the OBO ontologies) are identical in that all three allow for human-readable class definitions. The second way of adding semantic meaning into an ontology is through asserting subclass (***is-a***) relationships. Again, OWL, BioMoby and most other bioinformatics ontologies share this level of complexity. The final level of semantics comes from the explicit elaboration of the properties that define a given class. While many of the most commonly used bioinformatics ontologies do not take this final step of semantics, BioMoby does; however to maintain simplicity and robustness within the Web Service use-case, property definitions in BioMoby are managed differently than those in OWL/RDF, which leads to both positive outcomes as well as limitations. Though a comprehensive discussion of this topic is provided in Section 4 of the Supplementary Material, it can be summarized in the observation that BioMoby achieves its interoperability largely through agreement between humans, rather than through machine-interpretation. In BioMoby, individuals (people) agree on (i) the meaning/intent of a particular class/concept, and (ii) the syntax by which that shared concept will be represented. Therein the system achieves its *semantic* behaviors. There is little, if any, computational reasoning over the semantics of BioMoby messages, and it seems that for an important subset of existing bioinformatics problems, machine interpretation is simply not required. So long as all service providers output a FASTA file in a **FASTA** Object, another service provider can safely interpret that an incoming **FASTA** Object contains a FASTA file, and ensure that their software parses it as such. In essence, the semantics of BioMoby resides in the brains of the service providers themselves. BioMoby thus behaves much like a human language, where the spelling of words and structure of a sentence is sufficient to communicate between two individuals since the meaning of those words and structures is commonly held between them.

CONCLUSIONS

BioMoby has been running with open, public participation for participation for more than 5 years, and its continued adoption by new bioinformatics resources worldwide is testament to its simplicity and successful use by third-party providers. We believe the experiences of the BioMoby development community offer significant insight into successful approaches to Web Services interoperability platforms and best-practices in service provision on the emergent Semantic Web. As Web

Services and Semantic Web Services increasingly become the architecture for bioinformatics, we believe that BioMoby and BioMoby-like frameworks will have a significant role to play in this future.

MATERIALS AND METHODS

The BioMoby ontologies are available as RDF/OWL documents and/or can be queried through the BioMoby Central API. The Moby Central API is implemented as a Perl SOAP service, and the ontologies and service information is stored and fetched from a MySQL database. Support libraries for clients and service providers are available in Perl, Java, and to a limited extent in Python. All code is available under the Perl Artistic License, via the BioMoby project homepage.

Key Points

- BioMoby's interoperability is achieved through a novel type of XML schema which is derived from an ontology.
- The BioMoby ontologies were constructed through an open, community-driven process and are accessible for update through an open API.
- Although it does not use the World Wide Web Consortium's Semantic Web technologies, BioMoby exhibits many behaviors predicted for the Semantic Web.
- Tooling for BioMoby is now sufficiently rich that both providers and consumers of BioMoby services are well-supported.

SUPPLEMENTARY MATERIAL

Supplementary Material is available at *Briefings in Bioinformatics* Online.

Acknowledgements

The BioMoby project was established through an award from Genome Prairie, and has continued with the support of Genome Alberta and Genome Canada, a not-for-profit corporation leading Canada's national strategy on genomics. Significant code development was undertaken in collaboration with the myGrid project, and we would like to acknowledge the myGrid team, in particular: the director of myGrid, Carole Goble; the Taverna lead, Tom Oinn; Pinar Alper; Duncan Hull; Chris Wroe; Robert Stevens and Phil Lord. The large community of BioMoby service providers are thanked for their patience and tolerance, especially during the transitional days when the API was changing—hopefully you will never have to re-code your services again! Funding was provided by Genome Prairie and Genome Alberta 'A Bioinformatics Platform for Genome Canada'; Canadian Institutes for Health Research; The Natural Sciences and Engineering Research Council of Canada; The Heart and Stroke Foundation for BC and Yukon; The EPSRC through the myGrid (GR/R67743/01, EP/C536444/1, EP/D044324/1, GR/T17457/01) e-Science projects; The Spanish National Institute for Bioinformatics (INB) through Fundación Genoma España; The Generation Challenge Programme (GCP; http://www.generationcp.org) of the Consultative Group for International Agricultural Research.

References

1. Stein L. Creating a bionformatics nation. *Nature* 2002;**417**: 119–20.
2. Pearson, H. Biology's Name Game. *Nature* 2001;**411**:631–2.
3. Wilkinson MD, Links M. BioMOBY: an open-source biological web services proposal. *Brief Bioinform* 2002;**3**: 331–41.
4. Wilkinson MD, Gessler D, Farmer A, *et al*. The BioMOBY project explores open-source, simple, extensible protocols for enabling biological database interoperability. *Proc Virt Conf Genom and Bioinf* 2003;**3**: 16–26.
5. Wilkinson M. BioMOBY: The MOBY-S platform for interoperable data service provision. In: Richard P. Grant (ed.). *Computational Genomics*. Wymondham: Horizon Bioscience, 2004.
6. The World Wide Web Consortium. *Web Services Activity*. http://www.w3.org/2002/ws (14 March 2007, date last accesed).
7. The World Wide Web Consortium. *Web Services Description Language (WSDL)* 1.1. http://www.w3.org/TR/wsdl (14 March 2007, date last accessed).
8. The World Wide Web Consortium. *Semantic Annotations for Web Services Description Language Working Group*. http://www.w3.org/2002/ws/sawsdl (14 March 2007, date last accessed).
9. Narayanan S, McIlraith SA. Simulation, verification and automated composition of web services. *Proceedings of the 11th international conference on World Wide Web 2002*; doi/10.1145/511446.511457.
10. Wu D, Sirin E, Hendler J, *et al*. Automatic Web Services Composition Using SHOP2. In: *Proceedings of the World Wide Web Conference 2003*. http://www2003.org/cdrom/papers/poster/p226/p226-wu.html (14 March 2007, date last accessed).
11. Korhonen J, Pajunen L, Puustjarvi J. Automatic composition of Web service workflows using a semantic agent. In: *Proceedings IEEE/WIC International Conference on Web Intelligence 13–17 October 2003*, pp. 566–9. doi/10.1109/wi.2003.1241269.
12. Gekas J, Fasli M. *Automatic Web Service Composition Using Web Connectivity Analysis Techniques. W3C Workshop on Frameworks for Semantics in Web Services 2005 Position Paper, 2005*; http://www.w3.org/2005/04/FSWS/Submissions/39/web_service_composition.pdf (14 March 2007, date last accessed).
13. Carman MJ, Knoblock CA. *Inducing Source Descriptions for Automated Web Service Composition. 2005*. http://www.isi.edu/integration/papers/carman05-wswkshp.pdf (14 March 2007, date last accessed).

14. Thakkar S, Ambite L, Knoblock A. Composing, optimizing, and executing plans for bioinformatics web services. *VLDB J* 2005;**14**:330–53. doi/10.1007/s00778-005-0158-4.
15. The World Wide Web Consortium. *Extensible Markup Language*. Available http://www.w3.org/XML (14 March 2007, date last accessed).
16. Seibel PN, Krüger J, Hartmeier S, *et al*. XML schemas for common bioinformatic data types and their application in workflow systems. *BMC Bioinformatics* 2006;**7**:490. doi/10.1186/1471-2105-7-490.
17. Kawas E, Senger M, Wilkinson M. BioMoby extensions to the Taverna workflow management and enactment software. *BMC Bioinformatics* 2006;**7**:523. doi/10.1186/1471-2105-7-523.
18. Lord P, Bechhofer S, Wilkinson MD, *et al*. Applying semantic web services to bioinformatics: experiences gained, lessons learnt. *Lect Notes Comput Sci* 2004;**3298**:350–64.
19. Good B, Wilkinson MD. The life sciences semantic web is full of creeps! *Brief Bioinform* 2006;**7**:275–86.
20. Wilkinson MD. Gbrowse Moby: a web-based browser for BioMOBY Services. *SCFBM* 2006;**1**:4. doi/10.1186/1751-0473-1-4.
21. Wilkinson MD, Schoof H, Ernst R, *et al*. BioMOBY successfully integrates distributed heterogeneous bioinformatics Web Services. The PlaNet exemplar case. *Plant Physiol* 2002;**138**:5–17. doi/10.1104/pp.104.059170.
22. The Gene Ontology Consortium. *GO Database Abbreviations*. http://geneontology.org/cgi-bin/xrefs.cgi (14 March 2007, date last accessed).
23. Ashburner M, Ball CA, Blake JA, *et al*. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nat Genet* 2000;**25**:25–9. doi/10.1038/75556.
24. Senger M. The BioMoby Dashboard. Available at http://biomoby.open-bio.org/CVS_CONTENT/moby-live/Java/docs/Dashboard.html (14 March 2007, date last accessed).
25. Stevens RD, Robinson AJ, Goble CA. myGrid: personalised bioinformatics on the information grid. *Bioinformatics* 2003;**19**(Suppl 1):i302–4.
26. Object Management Group. *Life Sciences Analysis Engine Specification*. http://www.omg.org/technology/documents/formal/lsae.htm (14 March 2007, date last accessed).
27. Gordon PMK, Trinh Q, Sensen CW. Semantic web service provision: a realistic framework for bioinformatics programmers. *Bioinformatics* 2007;**23**:1178–80.
28. Senger M, Kawas E. *MoSeS – Code Generators*. http://biomoby.open-bio.org/CVS_CONTENT/moby-live/Java/docs/Moses-generators.html (14 March 2007, date last accessed).
29. Phillips J, Chilukuri R, Fragoso G, *et al*. The caCORE software development kit: streamlining construction of interoperable biomedical information services. *BMC Med Inform Decis Mak* 2006;**6**:2. doi/10.1186/1472-6947-6-2.
30. The World Wide Web Consortium. *SOAP 1.2 Specification*. http://www.w3.org/TR/soap (14 March 2007, date last accessed).
31. Hull D, Wolstencroft K, Stevens R, *et al*. Taverna: a tool for building and running workflows of services. *Nucleic Acids Res* 2006;**34**:W729–32. doi/10.1093/nar/gkl320.
32. Navas-Delgado I, Rojano-Munoz Mdel M, Ramirez S, *et al*. Intelligent client for integrating bioinformatics services. *Bioinformatics* 2006;**22**:106–11. doi/10.1093/bioinformatics/bti740.
33. Carrere S, Gouzy J. REMORA: a pilot in the ocean of BioMoby web-services. *Bioinformatics* 2006;**22**:900–1. doi/10.1093/bioinformatics/btl001.
34. Turinsky AL, Ah-Seng AC, Gordon PM, *et al*. Bioinformatics visualization and integration with open standards: the Bluejay genomic browser. *In Silico Biol* 2005;**5**:187–98.
35. Gordon PMK, Sensen CW. Seahawk: moving beyond HTML in web-based bioinformatics analysis. *BMC Bioinformatics* 2007;**8**:208.
36. Open Biomedical Ontologies Consortium. http://obo.sourceforge.net (14 March 2007, date last accessed).
37. Simple Semantic Web Architecture and Protocol. http://sswap.info. (14 March 2007, date last accessed).
38. Wheeler DL, Barrett T, Benson DA, *et al*. Database resources of the national center for biotechnology information. *Nucleic Acids Res* 2006;**34**:D173–80. doi/10.1093/nar/gkj158.
39. NCBI e-Utilities Function Call. http://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?db=gene&mode=xml&id=640 (14 March 2007, date last accessed).
40. Good BM, Tranfield EM, Tan PC, *et al*. Fast, cheap and out of control: a zero curation model for ontology development. In: *Proceedings of the Pacific Symposium on Biocomputing* 2006;128–39.
41. Wang G, Liu M. Extending XL Schema with nonmonotonic inheritence. *Lect Notes Comput Science* 2003;**2814**:402–7. doi/10.1007/b13245.
42. World Wide Web Consortium. *Web Ontology Language (OWL)*. Available http://www.w3.org/2004/OWL (14 March 2007, date last accessed).
43. World Wide Web Consortium. *Resource Description Framework (RDF)*. http://www.w3.org/RDF (14 March 2007, date last accessed).
44. Tsarkov D, Horrocks I. FaCT++ Description Logic Reasoner: System Description. In: *Proceedings of the International Joint Conference on Automated Reasoning (IJCAR 2006)*. 2006.
45. Haarslev V, Moller R. Racer: a core inference engine for the semantic web. In: *2nd International Workshop on Evaluation of Ontology-based Tools (EON-2003)*, Sanibel Island, FL, 2003.
46. Sirin E, Parsia B, Grau BC, *et al*. *Pellet: a practical OWL-DL reasoner. 2006*. http://www.mindswap.org/papers/PelletJWS.pdf (9 November 2007, date last accessed).