Syst. Biol. 61(4):675–689, 2012

DOI:10.1093/sysbio/sys025
Advance Access publication on February 22, 2012

NeXML: Rich, Extensible, and Verifiable Representation of Comparative Data and Metadata

RUTGER A. VOS[1,*], JAMES P. BALHOFF[2,3], JASON A. CARAVAS[4], MARK T. HOLDER[5], HILMAR LAPP[2], WAYNE P. MADDISON[6], PETER E. MIDFORD[2], ANURAG PRIYAM[7], JEET SUKUMARAN[5], XUHUA XIA[8], AND ARLIN STOLTZFUS[9]

[1]*NCB Naturalis, Leiden, the Netherlands;* [2]*National Evolutionary Synthesis Center, Durham, NC, USA;* [3]*Department of Biology, University of North Carolina, Chapel Hill, NC, USA;* [4]*Department of Biological Sciences, Wayne State University, USA;* [5]*Department of Ecology and Evolutionary Biology, University of Kansas, USA;* [6]*Departments of Zoology and Botany, and Beaty Biodiversity Museum, University of British Columbia, Canada;* [7]*Department of Mechanical Engineering, Indian Institute of Technology Kharagpur, India;* [8]*Department of Biology, University of Ottawa, Canada; and* [9]*Biochemical Science Division, National Institute of Standards and Technology, Gaithersburg, MD, USA;*
**Correspondence to be sent to: NCB Naturalis, Postbus 9517, 2300 RA, Leiden, the Netherlands; E-mail: rutger.vos@ncbnaturalis.nl.*

Received 16 May 2011; reviews returned 29 July 2011; accepted 7 February 2012
Associate Editor: Peter Foster

Abstract.—In scientific research, integration and synthesis require a common understanding of where data come from, how much they can be trusted, and what they may be used for. To make such an understanding computer-accessible requires standards for exchanging richly annotated data. The challenges of conveying reusable data are particularly acute in regard to evolutionary comparative analysis, which comprises an ever-expanding list of data types, methods, research aims, and subdisciplines. To facilitate interoperability in evolutionary comparative analysis, we present NeXML, an XML standard (inspired by the current standard, NEXUS) that supports exchange of richly annotated comparative data. NeXML defines syntax for operational taxonomic units, character-state matrices, and phylogenetic trees and networks. Documents can be validated unambiguously. Importantly, any data element can be annotated, to an arbitrary degree of richness, using a system that is both flexible and rigorous. We describe how the use of NeXML by the TreeBASE and Phenoscape projects satisfies user needs that cannot be satisfied with other available file formats. By relying on XML Schema Definition, the design of NeXML facilitates the development and deployment of software for processing, transforming, and querying documents. The adoption of NeXML for practical use is facilitated by the availability of (1) an online manual with code samples and a reference to all defined elements and attributes, (2) programming toolkits in most of the languages used commonly in evolutionary informatics, and (3) input–output support in several widely used software applications. An active, open, community-based development process enables future revision and expansion of NeXML. [Data standards; evolutionary informatics; interoperability; phyloinformatics; semantic web; syntax format.]

According to a recent whitepaper on the "New Biology" (Connelly et al. 2009), we are entering a new era in which many major advances will take place via integration and synthesis rather than decomposition and reduction. Evolutionary biologists and systematists, with their long history of integrative work, are well positioned to appreciate the challenges of synthetic science for education, career development, and technical infrastructure (Sidlauskas et al. 2010). Given that "information is the fundamental currency of the New Biology," it is important to ensure that "all researchers be able to share and access each others' information," and this puts a strong emphasis on the cyber infrastructure that supports archiving and reuse of data (Connelly et al. 2009).

Indeed, reuse of data is essential to the progressive nature of scientific inquiry, as it allows subsequent researchers to verify published results and to build on them. Except in some rare cases of high-throughput data, the primary producers of scientific data are the primary or immediate consumers of these same data, applying them to problems of scientific or technical interest. Reuse of data thus typically implicates secondary consumers, some of them using data in the same ways as the primary consumers and others carrying out synthetic and integrative science, including imaginative "repurposing" of data.

Archiving of data is a key step facilitating reuse. Whether or not evolutionary biologists are conscious of data reuse in their own field, they will be archiving thousands of phylogenetic trees and comparative data matrices in the next few years, if only because of policies to require archiving of such results (Moore et al. 2010; Rausher et al. 2010; Whitlock et al. 2010). However, while archiving policies encourage the flow of information into archives, they do not ensure that it flows back out again. For instance, although information on specimen collections had been archived and maintained by natural history museums for decades, until recently much of these data remained in metaphorical silos and were not readily integrated into a larger universe of data. Today, however, resources such as VertNet (Constable et al. 2010) aggregate data from many separate collections, to the benefit of users.

Biodiversity informatics provides useful examples of how integration may be achieved. The revolution in biodiversity informatics, with the promise to "turn the Internet into a giant global biodiversity information system" (Bisby 2000) was announced a decade ago and has achieved notable successes in data integration, often driven by the promise of addressing major issues—preserving biodiversity, remediating effects of global warming—by integrating diverse resources via occurrence data and geo-references. Behind this revolution is a scientific subculture with an emphasis on

"development and implementation of formal data exchange standards and query protocols" (Johnson 2007). Partly through a stakeholder organization called "Biodiversity Information Standards," which is more commonly known by its historic acronym of TDWG (the Taxonomic Databases Working Group), the biodiversity community has been developing standards and catalyzing success in aggregating and integrating data. The TDWG technical architecture group recommends a 3-legged strategy for interoperability (Hyam 2008), relying on:

1. Globally Unique Identifiers or GUIDs (*sensu* Page 2008) to identify the objects about which data are exchanged.
2. Concepts defined in a relevant ontology and referenced via GUIDs, typically internet addresses, that is, HTTP URIs.
3. Exchange protocols based on formats defined in XML Schema Definition (Biron and Malhotra 2004; Fallside and Walmsley 2004; Thompson et al. 2004) or other technologies.

A further aspect that is implicit in the approach taken by TDWG is to develop and utilize reporting standards that specify, not just how data and concepts are to be referenced in a computable way but what *kinds* of additional information are needed to constitute an adequate record. More generally, in the biological and biomedical research community, "minimum information" standards (Taylor et al. 2008) have played a key role in ensuring that the annotations (metadata) needed to interpret biological data are included in a record. Leebens-Mack et al. (2006) have called for a "Minimum Information About a Phylogenetic Analysis" (MIAPA) standard focused on metadata that allow secondary consumers to interpret phylogenetic results.

In considering interoperability in the context of evolutionary comparative analysis, these same 4 keys to interoperability become noticeable, whether by absence or by presence. For instance, the submission process for the TreeBASE archive (Piel et al. 2009) allows users to link OTUs with GUIDs for species concepts from external sources, making them part of a global network of resources linked by species identifiers provided by uBio (Leary et al. 2007). Sidlauskas et al. (2010) emphasize the importance of such metadata for making phylogenetic results intelligible.

In evolutionary comparative analysis, the de facto standard format for serializing data is NEXUS (Maddison et al. 1997). NEXUS was designed initially in 1987 by David Maddison and David Swofford, who sought a common format for their respective programs MacClade (Maddison and Maddison 2005) and PAUP (Swofford 2002). The subsequent popularity of NEXUS reflects its deliberate design (Maddison et al. 1997) emphasizing general interoperability concerns such as consistent representation on different computer platforms, broad scope to cover different types of data, modularity, and extensibility.

However, the promise of NEXUS for ensuring interoperability has subsequently eroded, in part because of lack of a community standards process for modification and extension and in part because of limitations of its basic design. For instance, NEXUS does not have an explicit means to link data objects with ontology concepts or GUIDs, to specify locations with geo-references, to include citations, or to convey some of the key annotations (study objectives, specimen vouchers, and methods descriptions) suggested as part of the recommended minimal information for adequately recording a phylogenetic analysis (Leebens-Mack et al. 2006). Such issues might be addressed with revisions—and some have been addressed in isolated projects—but NEXUS is not under active development as a community standard, and no revision of NEXUS subsequent to the original (Maddison et al. 1997) is considered authoritative. More importantly, fundamental weaknesses in the design of NEXUS (primarily, the lack of a formal grammar and the inability to unambiguously refer to arbitrary entities described within a file) mean that revising the original standard to meet today's requirements is not possible without also breaking backwards compatibility with current NEXUS tools, which obviates the main advantage of incremental revision.

To address the challenge of facilitating interoperability among phylogenetic resources (both data and services), some computational evolutionary biologists organized an Evolutionary Informatics (EvoInfo) Working Group, supported by the US National Evolutionary Synthesis Center (NESCent). This effort gave rise to several tangible outcomes, including an XML format for comparative data (NeXML), an ontology for comparative data analysis Ontology (CDAO; Prosdocimi et al. 2009), and a web-services standard. The results of the NeXML project are reported here.

The NeXML project is an open-source software project with a community development model, a project web site, and a mailing list. The project's primary tangible products are the NeXML schema, online documentation including a user's manual, and various implementations that support programmatic manipulation of NeXML data.

The structure of NeXML documents is defined in XML Schema Definition Language (Biron and Malhotra 2004; Fallside and Walmsley 2004; Thompson et al. 2004), providing strict constraints for data types, cardinality of relationships, and the uniqueness of identifiers for elements. NeXML represents diverse types of comparative data by employing the character-state data model that is implicit in NEXUS (Maddison et al. 1997) and that is widely used in comparative biology (Fig. 1). NeXML represents diverse types of phylogenetic trees and networks using concepts from graph representation. All elements in a NeXML document—branches and nodes in trees, cells in a matrix, OTUs, and so on—can be identified and given annotations using a generalized system that allows for simple values as well as complex, structured information such as geo-references, taxon concepts, or character-state descriptions. Moreover, data elements can be declared as instances of a class defined in an

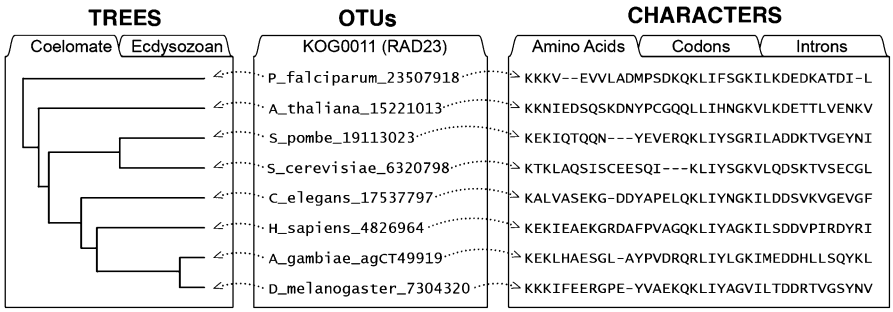

FIGURE 1. Data modeling in evolutionary informatics. Nodes in phylogenetic trees (shown left) and character-state data (right) can be conceptualized as forming a nexus at the center of which are operational taxonomic units (OTUs). Under this model, any number of trees and character-state data sets (the latter themselves following an entity–attribute–value model) are represented as data that apply to OTUs, which in principle can also be decorated with additional metadata such as taxonomy database record identifiers. This conceptualization is implicit in the NEXUS format and applications that build on it such as Mesquite (Maddison and Maddison 2011) and has been reused in NeXML. (Figure modified from Hladish et al. 2007).

ontology, making the semantics of the data themselves computable.

Because of its origin in a larger stakeholder group concerned broadly with interoperability issues and due to its development as an open community project, NeXML is well positioned to develop into a widely used community standard. Due to its extensible metadata annotation scheme, NeXML (unlike other phylogenetic data formats) has the potential to support emerging minimum information standards along the lines proposed for MIAPA (Leebens-Mack et al. 2006). NeXML-processing toolkits have been implemented for the Java, Perl, Python, JavaScript, and Ruby programming languages. In addition, NeXML is supported in end-user software such as Mesquite (Maddison and Maddison 2011) and DAMBE (Xia and Xie 2001). Full development of NeXML into a widely used community standard will require further community engagement, particularly to develop external language support for the kinds of metadata that the community considers most important.

DESIGN

NeXML was designed and developed through a community process that had its origin in the EvoInfo Working Group sponsored by the National Evolutionary Synthesis Center (NESCent) and led by R.A.V. and A.S. The working group conceived its mission in terms of facilitating interoperability in "phyloinformatics," defined broadly as the informatics dimension of evolutionary (phylogenetic) comparative analysis and took a deliberate approach to defining its scope and demands by enumerating and studying use cases, such as gene–species tree reconciliation, morphological character analysis, the construction of supertrees, and so on.

Working group members concluded that an effective interoperability strategy would depend on a "central unifying artifact," which might be a database schema, a file format, a formal ontology, or something else. Some participants conceived of this artifact as a store of domain-specific logic: they began to work on a concept glossary and later, an ontology that became the CDAO (Prosdocimi et al. 2009). Other participants turned their attention to file formats, an effort that led ultimately to NeXML. Below we describe some of the limitations of current formats that led to the decision to develop a new format, followed by design criteria for the new format, based on the analysis of use cases.

Limitations of Existing Formats and the Desirability of a New Format

For a large majority of interfaces used in phylogenetic analysis, the "Newick" (also known as "New Hampshire") string format is the standard way of representing trees. A Newick tree is a character string that organizes, within a hierarchy of nested parentheses, tokens representing nodes and their properties. The string "(chicken, (cow, (chimp, gorilla)));" includes only tokens representing the names of terminal nodes, whereas "(cow:0.21,(chimp:0.05, gorilla:0.03)primate:0.13);" includes branch lengths and an internal node name. The Newick standard is incorporated into NEXUS: the "TREES" block of a NEXUS document may contain trees represented as Newick strings. NEXUS is the most common way to represent, in a single document, trees together with associated character data (e.g., sequence alignments, matrices of morphological character states), the latter of which can be represented essentially as tabular data.

Because of the broad coverage of the NEXUS format and its widespread use, the EvoInfo working group initially considered targeting its efforts to supporting the NEXUS format rather than developing a new format. The group extensively analyzed the challenges associated with the use of NEXUS, generating a set of test files (building on Hladish et al. 2007) and producing a lengthy report (http://informatics.nescent.org/wiki/Supporting_NEXUS). Key challenges that were identified include: poorly formed files, that is, files that unambiguously violate the described

syntax (Maddison et al. 1997); a widespread practice of squeezing critical information into comments (square-bracketed strings); divergent interpretations of syntax based on ambiguity in the standard; application-specific extensions to core blocks (e.g., the "utree" command in PAUP); and incomplete implementations.

Many of these problems reflect the lack of a formally defined grammar. Without a formal grammar, there is no widely trusted validator for NEXUS documents, a situation that hampers many aspects of development but particularly the development of automated tools. For example, about 15% of NEXUS files submitted to the CIPRES portal (Miller et al. 2010) contain unrecoverable yet hard-to-diagnose file format errors (Miller M.A., personal communication). In the absence of a formal grammar written in a language supported by parsing tools, programmers who wish to support NEXUS have to write a custom parser, a heavy burden given the complexity of the standard. Indeed, this burden has been so great that, to our knowledge, no software has ever fully supported the NEXUS specification, although programs such as PAUP, MacClade, and Mesquite provide robust support for the most commonly used NEXUS blocks and commands. Until the appearance of NCL (Lewis 2003) and Bio::NEXUS (Hladish et al. 2007), software developers wishing to support NEXUS did not have access to a generalized programming library. As a result of this programming burden, most programs that utilize NEXUS represent highly incomplete implementations, and they often fail to respect NEXUS rules for such things as tokens and quoting; users of these programs then often face undocumented (therefore mysterious) restrictions on such things as the lengths of OTU names or the symbols that may appear in labels or other descriptors.

Another set of problematic issues with existing formats relates to the identifiability of elements to be referenced and annotated. NEXUS has a fixed set of elements that can be referenced (in specific contexts, in other parts of the same file), including OTUs, characters, trees, and various kinds of sets. However, use cases that integrate or combine data frequently depend on the ability to identify and differentiate other kinds of data elements. For instance, tree reconciliation (gene tree vs. species tree) and fossil calibration of phylogenetic dates both demand the ability to separately identify and annotate specific nodes of a tree, sometimes in complex ways (e.g., annotating a node with a date estimate that includes a measure of reliability and a citation of provenance). Although the NEXUS specification (Maddison et al. 1997) provides facilities for referencing some objects (such as in the "NOTES" blocks), this does not extend to other objects, among which are nodes in trees.

Facilitating interoperability not only requires fine-grained referencing of data elements, but the capacity to annotate these elements with externally defined vocabularies. The submission process for TreeBASE (Piel et al. 2009), for instance, provides a means to specify NCBI GenBank (Benson et al. 2009) accession numbers for sequences and encourages users to assign species names from controlled vocabularies (namely the NCBI taxonomy and the uBio NameBank). Yet, despite the importance of such information for the reusability of data, only a recently proposed XML format, PhyloXML (Han and Zmasek 2009), provides a specific means to assign accession numbers and vocabulary-controlled species names to nodes in the tree or molecular sequence data. Likewise, while a NEXUS file may represent complex characters using numeric states annotated with human-readable text, the ability to describe such characters in a machine-readable way, for example, using ontology terms, is increasingly important (Dahdul et al. 2010a).

The lack of a formal grammar and the lack of fine-grained referencing of elements are probably the core limitations of the NEXUS format, in that it is hard to conceive of a redesign of NEXUS that could address such issues while retaining backward compatibility. In addition to these core limitations, other weaknesses in the design of NEXUS have subsequently led to "fixes" that rely on informal conventions rather than the normative standard. For instance, the need to support the reconcile-tree use case (mentioned earlier) has led to the New Hampshire (Newick) Extended format or NHX (Zmasek and Eddy 2001), a "hot comments" scheme in which square-bracketed strings with information relevant to a node are positioned within a Newick string, after the closing parenthesis of that node. Square-bracketed strings were chosen because this is the syntax of Newick and NEXUS comments and thus should not disrupt a well-written parser. Yet, precisely because square-bracketed strings are syntactic comments, their use for storing critical information is ill advised: the NEXUS standard (which integrates Newick) does not specify that comments in an input stream must be retained in an output stream, much less retained at their original position. Thus, a valid NEXUS processor may scramble a valid NHX tree.

The challenges noted above relate to the use of inadequate representation schemes and do not reflect significant changes in foundational concepts as represented in Figure 1. However, there are some ways in which comparative analysis has broadened conceptually since the initial design of NEXUS. An example is the increasing importance of anastomosing phylogenies that represent horizontal gene transfer events or within-population recombination. These are often called "networks," as distinct from "trees" (though neither usage corresponds to the "tree" and "network" concepts from graph theory). In the Newick representation of trees, on which NEXUS currently depends, the parent of a node is represented by the innermost left and right parentheses containing it: thus a conventional Newick string cannot be used to represent multiple parentage, as there can be only one innermost pair of enclosing parentheses. This limitation led to an "eNewick" format (Cardona et al. 2008) in which hybrid nodes appear twice in the same tree, so that their multiple parentage may be represented. An alternative method (Than et al. 2008) uses multiple Newick trees to achieve the same result. Because node and branch annotations are not part of Newick, neither

scheme can indicate whether a particular parental relationship is vertical nor horizontal. This could be fixed by abandoning the restriction of following a strict Newick format, as had been proposed within the NEXUS community but doing so would destroy backward compatibility with current NEXUS-parsing tools, undercutting the benefits of revising, rather than replacing, NEXUS.

On the basis of such considerations, we concluded that it was not feasible to build a future-oriented standard by incrementally revising a legacy format.

Requirements

In considering the design of a new format for representing phylogenetic data, another result of the analysis of use cases is the emerging importance of metadata annotations of phylogenetic data's, ontology-one straightforward conclusion from the analysis of use cases is that a NEXUS-like capacity to combine trees and character data in a single file or data object is highly favored by users for maintaining the integrity of data. A second conclusion is that the character-state data model, depicted in Figure 1 (Hladish et al. 2007), contributes robustness and generality to representations such as NEXUS. The character-state data model is an entity–attribute–value model in which the entities are OTUs with "character" attributes that have values called "states" or character-states. This model is implicit in a wide range of applications in evolutionary analysis. Character states may represent molecular observations, for example, one of several possible nucleotides at a given site in a sequence; morphological observations, for example, one of several possible qualities of a given anatomical entity; or some other data type. However, each type of data has its own requirements for representing various types of arbitrarily complex uncertainty and polymorphism in the observations. Existing data standards can represent the simplest types of characters but lack the facility to express these with enough granularity to meet the requirements of projects to encode comparative phenotypic data (Dahdul et al. 2010a).

Another result of the analysis of use cases is the emerging importance of metadata annotations of phylogenetic data. A facility for such annotations is important for 2 reasons. First, community resources such as TreeBASE (Piel et al. 2009) produce large amounts of metadata that cannot be expressed using any of the current standards. Secondly, use cases that involve the integration and reconciliation of disparate data sets call for these data to be "joined" (that is, logically linked) on some axis. This requires a facility to identify phylogenetic data objects, whether they are OTUs linked to globally unique records in a taxonomic name bank, characters linked to a phenotype ontology, or trees linked to a tree repository. These considerations also were noted in the call for a minimum information standard for phylogenetic analyses (Leebens-Mack et al. 2006).

The analysis of use cases, as described above, thus led to the following list of requirements:

1. *Self-contained data and metadata.*—The standard must allow character data, trees, and annotations, that is, the data and metadata of a typical study, to be represented together in a single document.
2. *Granular annotation capability.*—The standard must allow all significant elements within a phylogenetic record, including those for which no current annotation standard exists (e.g., tree nodes), to be annotated flexibly and in an extensible fashion.
3. *Completely defined, versioned syntax.*—The syntax of the standard must be described by a grammar that is complete, authoritative, and evolvable. A complete and authoritative grammar is necessary to determine if a document conforms to the standard. In practice, the need for the grammar to evolve implicates a further requirement for versioning of the standard.
4. *Completely defined semantics.*—To facilitate machine interoperability, the new standard should have defined semantics. By contrast, current conventions sometimes reflect the opportunistic use of a common syntax for different meanings, for example, square-bracketed numeric values in Newick strings have been used for support values from bootstrap replicates, decay indices, posterior probabilities, and other values.
5. *Re-use of prior art; compatibility with domain-specific concepts.*—A new standard should draw as much as possible on conventional language and concepts from the domain of evolutionary comparative analysis.
6. *Machine-readable with off-the-shelf tools.*—Because the success of a standard depends so much on correct and complete implementations, a new standard should be formulated in such a way as to allow developers to leverage convenient and portable software tools to process its contents.
7. *Self-describing document metadata.*—The standard must allow any metadata needed to process the document, for example, the version of the syntax and the character encoding in which the document is expressed, to be contained within the document.
8. *Open development process.*—The standard should evolve in the open, supported by a development infrastructure that makes it possible to respond to the changing needs of a broad community.
9. *Scalability.*—The standard must be scalable such that data files are not unreasonably larger in size compared with an equivalent NEXUS document that expresses the same information content.
10. *Internationalization support.*—The standard must allow that annotations such as author names, publication titles, or character-state descriptions may be encoded in non-western character sets.

IMPLEMENTATION

Technology Choices

Structured data standards can be designed around a number of different technologies. The EvoInfo Working Group has considered flat text with a Backus–Naur form grammar, JSON, ASN.1, XML, and RDF. For each

of these formats, off-the-shelf processing tools exist for a variety of programming languages, though the extent of support varies.

A detailed consideration of these technologies is given in an Appendix. In brief, several requirements above justify the choice of XML technologies including RDF/XML. The phylogenetics community has recognized XML as a potential data syntax: for example, in *Inferring Phylogenies*, Felsenstein makes some XML syntax suggestions for tree shapes (Felsenstein 2004) that were the inspiration for some of the design choices in PhyloXML (Han and Zmasek 2009). Additionally, on a mailing list that was active around 2000 called PhyloXML (no relation to the aforementioned standard), ideas for an XML-based phylogenetic data syntax were discussed (including a suggestion by WPM to call the standard NEXXML). This discussion included a syntax proposal by Andrew Rambaut and Alexei Drummond, which proposed the use of GraphML-like syntax for trees, nodes, and edges. However, prior art has not established how an XML technology stack should be used in an evolving context. We arrived at a solution where, if information is to be treated as structured *syntax*—which is easier to process but ultimately interpreted by the "business logic" of the processing software—it will be expressed as XML. Conversely, if the *semantics* are to be explored, XML statements can be transformed (using an XSL style sheet produced by members of the EvoInfo Working Group) to an RDF/XML representation relying on CDAO, which is harder to process but interpreted with recourse to the formal semantics in CDAO.

The NeXML Schema

Based on the requirements and technology choices described above, syntax proposals for NeXML were evaluated, and processing toolkits in a variety of programming languages including Perl, Python, Java, Ruby, and JavaScript were developed at (and between) meetings of the EvoInfo working group. This approach, which allowed early identification of problems in the design of the standard itself, is ongoing.

The syntax of NeXML is formalized in a grammar constructed in XSD, which we chose because it allows a grammar to be described in great detail, including, for example, permissible symbols in character-state sequences, number formats for continuous-valued character states or branch lengths on trees, date strings for metadata, etc., thereby satisfying *requirement 3: completely defined, versioned syntax*. This NeXML schema is available from http://www.nexml.org under a Creative Commons Zero license, and its future development is subject to an open process with continuous community review, as per *requirement 8*. Currently, the NeXML website hosts a document-validation service, along with services to translate Newick and NEXUS documents into NeXML and NeXML documents into JSON and CDAO RDF/XML. Development versions of the schema and some supporting software are available from a repository (http://www.nexml.org/code/), where users and developers may sign up for a mailing list to discuss the standard.

The schema consists of multiple files (20 at time of writing) dealing with different parts of the data model; these are designed to jointly capture the data and metadata of a typical phylogenetic study in a single NeXML instance document (i.e., satisfying *requirement 1: self-contained data and metadata*). The schema is decomposed along conceptually coherent lines; for example, there is a file for each character-state data type, and new ones can be easily added. The NeXML design extensively leverages the facility to define complex types that can be subclassed, overridden, and instantiated recursively into other definitions. Super classes of type definitions are defined as abstract types that inherit from one another through extension; the final tips of the inheritance tree (the concrete instances) inherit through restriction of the allowable value space. This means that a well-written NeXML processing toolkit that is designed against the abstract superclass of the concrete instances is automatically capable of dealing with all possible such concrete instances, present, and future. This is true because in XSD, inheritance through restriction (by definition) cannot add novel features to subclasses, and it facilitates the development of processing toolkits based on code generated directly from the schema, as, for example, is done by the Phenex application (Balhoff et al. 2010).

The schema further reflects the reuse of prior art (*requirement 5*). NeXML's implicit data model follows the character-state data model (Fig. 1) that underlies software such as MacClade (Maddison and Maddison 2005), PAUP (Swofford 2002), Mesquite (Maddison and Maddison 2011), and MrBayes (Ronquist and Huelsenbeck 2003); file formats such as NEXUS (Maddison et al. 1997); and databases such as TreeBASE (Piel et al. 2009). Borrowing from NEXUS, character-state matrices are enclosed in structures named characters that comprise format declarations and matrix structures. Likewise, tree objects are contained by trees structures. In contrast, the name "taxa," which NEXUS uses for structures that contain OTUs, has been changed to otus to avoid the frequently incorrect implication that OTUs are equivalent to taxa in some formal taxonomic classification of organisms. In regard to representing trees, NeXML borrows from GraphML to use a record-oriented structure of node and edge lists.

To ensure granular identifiability of data elements (*requirement 2*), all data objects expressed as elements in NeXML can be annotated within the element and also have document-scoped identifiers so that they may be referenced from other elements. For example, every otu element has an id attribute whose value is unique within the whole document and which may be used in another element to refer to a specific *otu*. These identifiers are meant as structural aids: they have no meaning outside of the scope of a NeXML document, and they are ephemeral in round trips where documents are read, interpreted, and written out again.

Thus, a processing toolkit does not have to preserve them.

In order for annotation to be extensible, the schema allows that additional statements not specified by the core NeXML standard can be made *about* fundamental data within NeXML documents. Such statements can be decomposed into triples consisting of a subject (the fundamental datum), a predicate defined by a controlled vocabulary or ontology and an object or value. These annotations might influence the interpretation and thus the processing of whatever other data elements are contained by the triple's subject; therefore, annotations are the immediate children of the element they reference, preceding any other child elements whose processing they might influence. For example, a NeXML tree description consists of a *tree* element that contains *node* and *edge* elements: annotations that (for instance) clarify the nature of polytomies (e.g., resulting from a consensus computation) are the first child elements of the *tree* element, so that such annotations can be taken into account when processing the subsequent topology.

The schema reflects additional design choices to achieve greater scalability (*requirement 9*):

1. *Declaration before referencing.*—Under the data model that NeXML follows, some fundamental data refer to others. For example, character-state sequences and tree nodes can indicate which OTU they apply to. As a general rule, whenever one element references another, the referencing element has an attribute with the name of the referenced element, and this attribute's value matches that of the referenced element's identifier. The referenced element must precede the reference in the document structure. Therefore, for example, otus elements precede trees elements and characters elements in NeXML documents, and the *trees* and *characters* elements have an attribute called *otus* whose value matches the identifier of the *otus* element to which they apply. This design principle makes it easier to implement stream-based processing toolkits because object representations can be created at the point where they are encountered in the document; when a new object is created the parser does not have to check for previous references to it.
2. *Processing hints come first.*—In some contexts, information about a data element's semantics can influence how the data element is to be processed. For example, for character-state matrices, such information includes an enumeration of the states (cell values), OTUs (rows), and characters (columns) that can be expected. If such information is provided first, processing is more straightforward to implement.

Implementations

In parallel with developing the NeXML syntax itself, members of the EvoInfo Working Group and other interested stakeholders have implemented support for NeXML in software. Because the list of such implementations is volatile, we refer the reader to the online NeXML manual (http://www.nexml.org/manual) for current information. Here, we provide several examples for illustration. An example of an API (applications programming interface) is the standalone library for Java version 5 developed by the authors; a more high-level toolkit, in the Python language, is implemented in DendroPy (Sukumaran and Holder 2010). An example of end-user software that supports NeXML is Mesquite (Maddison and Maddison 2011), an extensible graphical workbench for comparative analysis, which provides support for NeXML via an add-on package described in the NeXML manual. An example of a web resource that outputs NeXML is TreeBASE (Piel et al. 2009), which generates NeXML output as an option accessible via its web interface or via its web services interface. Both Mesquite and TreeBASE use the Java library mentioned above.

Documentation

The NeXML manual (http://www.nexml.org/manual) is a wiki with a beginner's guide and a programmer's reference to elements. The beginner's guide provides useful step-by-step instructions for getting trees and character data into the NeXML format. The programmer's reference includes code examples illustrating how to represent data and metadata.

Usage Example: TreeBASE OTU Annotations

The TreeBASE archive (Piel et al. 2009) accepts submissions of peer-reviewed studies of phylogenetics in a database that stores comparative data, trees that were inferred from these data, and metadata that records the provenance of these results. The TreeBASE web application attempts to link submitted OTU labels to known taxon names, by submitting the OTU labels to the uBio *findIT* web service after removing serial numbers or other suffixes. The uBio server then attempts to match the labels (as text strings) against records in its NameBank. The returned matching records may include additional metadata on matches between the record and other resources, such as the NCBI taxonomy. Figure 2 shows an example NeXML document that reports the results of this taxon-matching procedure.

The document in Figure 2 contains a single container of OTUs (an element called otus with the identifier otus1) with a single OTU (the otu element) that was submitted with the label "*Zenodorus* cf. *orbiculatus*." The TreeBASE web application retrieved a close match with the uBio NameBank record 3546132 for the name "*Zenodorus orbiculatus*," which in turn closely matches the NCBI taxon 393215 with the name "*Zenodorus* cf. *orbiculatus* d008." As the links between these records are all the result of string matching, the type of these links is specified using the predicate closeMatch from the SKOS

```
<nex:nexml
  xmlns:nex="http://www.nexml.org/2009"
  xmlns="http://www.nexml.org/2009"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:skos="http://www.w3.org/2004/02/skos/core#"
  xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
  generator="org.nexml.model.impl.DocumentImpl"
  version="0.9"
  xml:base="http://purl.org/phylo/treebase/phylows/">
```

A file header identifying several namespace prefixes, the document generator, the syntax version and the base URL from which relative URLs in the document are constructed.

```
  <otus id="otus1" label="Combined">
    <otu
      about="#otu3"
      id="otu3"
      label="Zenodorus cf. orbiculatus">
```

The user-submitted taxon label *Zenodorus cf. orbiculatus* inside a container "otus" element.

```
      <meta
        href="http://purl.uniprot.org/taxonomy/393215"
        id="meta8"
        rel="skos:closeMatch"
       xsi:type="nex:ResourceMeta"/>
```

A semantic annotation that links the label to an RDF representation of an NCBI taxonomy record using the *skos:closeMatch* predicate.

```
      <meta
href="http://www.ubio.org/authority/metadata.php?lsid=u
rn:lsid:ubio.org:namebank:3546132"
        id="meta6"
        rel="skos:closeMatch"
       xsi:type="nex:ResourceMeta"/>
```

A semantic annotation that links the label to an RDF representation of a uBio name bank record using the *skos:closeMatch* predicate.

```
      <meta
        href="study/TB2:S1787"
        id="meta5"
        rel="rdfs:isDefinedBy"
       xsi:type="nex:ResourceMeta"/>
```

A semantic annotation that specifies the TreeBASE study context within which the taxon label is defined.

```
    </otu>
  </otus>
</nex:nexml>
```

Closing XML tags.

FIGURE 2. NeXML syntax example: TreeBASE OTU annotations. This example shows a single container of OTUs (the *otus* element) with a single OTU (the *otu* element) that was submitted to the database with the label *Zenodorus* cf. *orbiculatus*. Matching this label to the uBio web service returned a close match with the record for *Zenodorus orbiculatus* (with the namebank identifier 3546132), which uBio describes as matching the NCBI taxonomy record for *Zenodorus* cf. *orbiculatus* d008 (with taxon identifier 393215). The normalized OTU label was defined within the context of TreeBASE study S1787.

vocabulary, as shown in the first 2 semantic annotations for the OTU. The last annotation specifies the context within which this OTU label is defined, that is, a study that was submitted to TreeBASE, using the RDFS predicate is Defined By with a stable URI (for that study) as its value, here specified as a path relative to the xml:base URI at the root of the document. Adopting NeXML has allowed TreeBASE to express (in an interoperable way) how user-supplied taxon labels are normalized against an external resource of taxon concepts.

Usage Example: Phenoscape Character States

The Phenoscape project (Dahdul et al. 2010a) links evolution to genomics using phenotype ontologies. By using the "Entity–Quality" (EQ; see Mungall et al. 2010) formalism of the OBO consortium, the Phenoscape project can combine organism-specific terms from ontologies such as the Teleost Anatomy Ontology (Dahdul et al. 2010b) with quality terms from the generic Phenotype And Trait Ontology (Gkoutos et al. 2004) to create intersections that describe character states in a specific group of organisms. The Phenex software (Balhoff et al. 2010) allows such descriptions to be attached to NeXML character states. Figure 3 shows an example code fragment of this. The format element contains enumerations of character-state definitions containing one or more state elements. The annotation clarifies that the state with identifier state0102 describes a phenotype, which is indicated by the usage of the describes Phenotype predicate from the Phenoscape vocabulary; the description itself is expressed in PhenoXML, a Phenex-specific syntax for constructing EQ statements (Balhoff et al. 2010). In this case, the entity TAO:0000127 has the quality PATO:0000462, which means that state state0102 describes the absence of the antorbital (a bone). For each subsequently defined character (char), the applicable state set is defined by referencing its identifier. Adopting NeXML has allowed the Phenoscape project to describe character states in great detail in an interoperable, machine-readable way.

DISCUSSION

In meeting the requirements for an interoperable data standard outlined here, NeXML will enable the construction of larger and more integrative phyloinformatic workflows and cyber infrastructures. As NeXML provides—by virtue of its mapping onto a formal specification of fundamental concepts in phylogenetics (the CDAO; Prosdocimi et al. 2009)—explicitly defined semantics, it will help to identify and resolve ambiguities in phylogenetic data. In addition, its facility for extensible semantic annotation will allow the inclusion of concepts from fields that intersect with phylogenetics.

Importantly, although NeXML is designed to allow for a richly annotated record of the type that could satisfy MIAPA (Leebens-Mack et al. 2006), actually

```
<format                                                                    A NeXML "format" element
  xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"                    containing character and
  xmlns:ps="http://vocab.phenoscape.org/"                                  state definitions.
  xmlns:phen="http://www.bioontologies.org/obd/schema/pheno">
  <states id="states01">                                                   A "states" element with one
    <state                                                                 state, "absent".
      id="state0102"
      about="#state0102"
      label="absent"
      symbol="1">
      <meta                                                                Annotation that links a state
        id="meta1"                                                         to an EQ phenotype using
        property="ps:describesPhenotype"                                   the describesPhenotype
        xsi:type="nex:LiteralMeta">                                        predicate.
        <phen:phenotype>                                                   An embedded block of
          <phen:phenotype_character>                                       PhenoXML data
            <phen:bearer>                                                  representing an EQ
              <phen:typeref about="TAO:0000127"/>                          phenotype with entity
            </phen:bearer>                                                 "antorbital" (TAO: 0000127)
            <phen:quality>                                                 and quality "absent" (PATO:
              <phen:typeref about="PATO:0000462"/>                         0000462)
            </phen:quality>
          </phen:phenotype_character>
        </phen:phenotype>
      </meta>                                                              Closing XML tags.
    </state>
  </states>
  <char                                                                    The character definition to
    id="char01"                                                            which the above state set
    label="Presence or absence of antorbital"                              applies.
    states="states01"/>
</format>
```

FIGURE 3. NeXML syntax example: Phenoscape character states. This code fragment shows how the Phenoscape project uses the NeXML-compatible application Phenex to annotate character states. A character, identified by "char01," is defined as able to occupy any of the states from state set "states01." Within that state set, in this instance, there is only the state "state0102." That state is annotated with an EQ statement (here expressed in a Phenex-specific XML dialect) that identifies a morphological feature called the "antorbital" and qualifies it as being absent. (In a complete NeXML document, the format element occurs within a characters element, which is preceded by a container of OTUs, i.e., an otus element, here omitted for clarity.)

satisfying a MIAPA standard would depend on external vocabularies that the community has not developed, as yet. That is, the NeXML schema has hard-coded versions of domain-specific concepts that emerged from our design and evaluation process but not hard-coded versions of annotation concepts that are poorly understood or that fall outside the domain of comparative analysis. For example, neither the evolutionary informatics community nor the science community more generally has decided on a standard for representing metadata for scientific publications. Hard-coding our own scheme into NeXML would be ill advised. Instead, the design of NeXML allows users to make references to existing vocabularies such as the Dublin Core (http://dublincore.org/) and PRISM (http://www.prismstandard.org), which are both sometimes used for metadata such as authorship and which could both be used within a NeXML document.

The facility for the inclusion of concepts from external vocabularies in NeXML documents, and the potential proliferation of such, highlights a need for community-based processes for developing standards, vocabularies, and best practices. The wider science community likely will settle on vocabularies for commonly used concepts such as publication metadata. However, domain-specific concepts will require the attention of the evolutionary informatics community itself. For example, at present, many commonly used applications for phylogenetic inference (e.g., Swofford 2002; Ronquist and Huelsenbeck 2003; Drummond and Rambaut 2007) have their own syntax for describing nucleotide substitution models. This situation clearly hampers interoperability and makes it difficult to foresee how the community could attain the goal of computable descriptions of methods. The community process by which NeXML emerged may serve as an example for how such standardization might be achieved.

What are the advantages and disadvantages of NeXML relative to other syntax formats? We consider this question briefly, first with a view to long-term strategic decisions of users and developers, and second, with a view to short-term needs. We note that intelligent technology choices often are not a black-and-white choice between what works and what does not, but a choice between tools that were designed to be adequate for different purposes. NeXML was designed to be a robust, validatable, and extensible means for representing comparative data, particularly in contexts with an emphasis on automated processing or integration via unifying concepts; it was not designed to be human-readable or concise.

As a long-term strategy for interoperable representation, and assuming that issues of scalability can be worked out over time, NeXML has numerous advantages that put it ahead of other formats for many uses. We have discussed already the reasons that a

standard for syntax should have a complete, versionable grammar. The XML format used in PHYLIP (see Felsenstein 1989 for the publication describing PHYLIP; XML output was added later, in version 3.6) and the XML format used by ToLWeb (Maddison et al. 2007) do not have a schema, and thus no versionable grammar. Currently, the only such formats are XML formats, among which we may include NeXML, PhyloXML (Han and Zmasek 2009), BEAST XML (Drummond and Rambaut 2007), and OrthoXML (Schmitt et al. 2011). OrthoXML may be used to represent a hierarchy of related genes, but it is not a phylogeny format and lacks basic features for this purpose, for example, the capacity to represent branch lengths. Though BEAST XML has a rich schema for representing substitution models, it is designed as a command file to allow automated control of BEAST, not as an exchange format (Rambaut A., personal communication); accordingly, it lacks features required of an output or exchange format, which is why the BEAST program outputs NEXUS, not XML. Currently, PhyloXML and NeXML are the only viable candidates for a future exchange format in phylogenetics.

PhyloXML is well designed as a robust but minimalist format for certain uses in molecular evolution and comparative genomics. PhyloXML defines specific elements that correspond to the most common user needs and leaves aside exceptions and complications. By contrast, the design philosophy of NeXML, much like that underlying NEXUS, is to unify comparative biology by providing a framework that can accommodate nearly anything. Thus, PhyloXML uses nested XML elements to represent trees, implying classic rooted hierarchies, whereas NeXML uses a more general collection of nodes and edges, allowing for anastomosing trees. PhyloXML assumes molecular sequence data, perhaps currently the most common use case, whereas NeXML allows nonsequence data, and for this reason is favored in the Phenoscape project.

Of note is also the difference in the way PhyloXML and NeXML model the relationships between phylogenetic data objects: where NeXML follows a model of tree nodes and character-state sequences joined on OTUs, PhyloXML models both OTUs and molecular character-state sequences as attributes of tree nodes. This latter model may cause redundancies, for example in cases where a sequence alignment corresponds to many trees (e.g., as a result of an MCMC analysis) or where many alignments correspond to one tree (e.g., when simulating sequences): in the former case, the same sequence attributes would have to be repeated for each tree, whereas in the latter case, the same tree would have to be represented multiple times to be annotated with the sequences resulting from each simulation. This potential issue may be especially relevant in cases where studies with multiple trees and character-state matrices relevant to a single set of species are included in a single file, e.g., as a download from TreeBASE.

Lastly, although PhyloXML allows for a flexible set of property/value annotations on most elements, these properties are not mediated by external ontologies or vocabularies but solely by the PhyloXML schema itself. This precludes PhyloXML's annotation system from being employed to import concepts specified by other user communities, as exemplified in the TreeBASE and Phenoscape examples presented above.

In the short term, the decision to opt for a particular format may be based on more immediate practical considerations than those described above. Such decisions, when made by end users, are rarely made on the basis of the superior design of one format over another. Instead, users opt for whatever syntax formats are demanded as input or output by their favorite software. For instance, if a convenient web-based tool to assist users in constructing a MIAPA-compliant record were to output NeXML, then NeXML might be chosen by end users for archiving but only because of the convenient web-based tool. Especially, in the case of XML formats, this is an important consideration, as XML is much harder for humans to read than flat text. Recognizing this fact, we have invested significant effort in bringing NeXML compatibility to Mesquite (Maddison and Maddison 2011) and will continue to improve on this in order to provide end users with an attractive graphical user interface for exploring, modifying, and authoring NeXML documents.

This implies that, indirectly, the choice of formats is in the hands of software developers who build phylogenetic inference tools, databases, and other resources. In making such a choice, the trade-off between file size and expressiveness and the issue of interoperability come into play. NeXML is expressive but rather verbose. For example, the "treezilla" rbcL alignment (Rice et al. 1997), when converted from NEXUS to NeXML increases in size from 709 to 823 kb; the Tree of Life Web Project's XML dump of the entire tree (tolskeleton) when converted from ToLWeb's XML format to NeXML increases in size from 5.6 to 9.4 mb. Because trees are represented as node and edge tables instead of parenthetical statements, a NeXML tree file is always bigger than a NEXUS tree (and grows linearly at a higher rate than NEXUS files as more trees are added). NeXML files are larger than legacy files but often in the same order of magnitude of size.

The question of whether NeXML is the right choice is informed in part by the extent to which a consumer of data requires embedded contextual information. On the one hand, if a tool produces output as part of a larger architecture (e.g., as a node in an MPI cluster, as a computational backend for a web portal or as a step in an analysis workflow), then other components in that larger architecture are the consumers. Such consumers, because they are part of the same architecture and thus have access to the context by other means, will only need the most compact representation of the output: a Newick string, a TreeZip (Matthews et al. 2010) string or an even more compact data structure should suffice for trees; and simple tabular representations should suffice for character-state matrices.

On the other hand, if the output of a tool is consumed within an entirely different context than the one in which it was produced, for example, in an integrative analysis that combines previously published results from online repositories, metadata from the original context (e.g., species names or comparative data provenance) needs to be embedded with the data so that the secondary consumer has access to it. Such contextual information frequently cannot be expressed by legacy formats (e.g., neither the TreeBASE nor the Phenoscape examples we present could be expressed in other formats), and so NeXML may become the right choice for such use cases.

A second consideration in choosing a syntax format is that of interoperability: although NeXML is designed to facilitate interoperability of richly annotated phylogenetic data, it is not yet widely adopted. Therefore, software developers seeking to improve interoperability may find that choosing NeXML hampers this in the short term. However, with the development of the NeXML schema and the processing toolkits in a number of programming languages, a sound basis has been laid for the further adoption of the standard, and so we anticipate that this issue will resolve itself.

Resources on the web that hold phylogenetic data and metadata are key candidates for adopting NeXML. An example of this is ToLWeb (Maddison et al. 2007), which currently emits its data using a special XML format that holds the tree structure, in addition to any metadata that are encoded using special attributes and elements. Given that some of these metadata (e.g., the annotation that a node in the tree structure represents an extinct taxon) are useful outside the ToLWeb project, making them available using NeXML (with semantic annotations that reference standard vocabularies) would promote interoperability.

More generally, resources on the web that emit phyloinformatic data, such as TreeFam (Li et al. 2006), PANDIT (Whelan et al. 2006), MorphoBank (O'Leary and Kaufman 2007), PhyloTA (Sanderson et al. 2008), and TOLKIN (Beaman and Cellinese 2010) do not typically differ in the fundamental data types they emit (namely, character-state matrices, alignments, and trees, as in Fig. 1), only in the metadata that describe the context within which the data are provided. Such resources could adopt NeXML and describe the metadata in semantic annotations, in addition to the descriptive web pages that require human readers to interpret the data. This way, interoperability between those resources, and the ability to synthesize and integrate them in innovative ways, would be facilitated.

Many types of metadata are of interest for such integrative projects. For example, OTUs can be annotated to specify taxonomic identifiers (as in the TreeBASE example) or geographic coordinates or polygons of occurrence data; molecular sequence data can be annotated with database accession numbers or GO (Ashburner et al. 2000) term identifiers; morphological and molecular data can be annotated with specimen identifiers; phylogenetic trees and networks can be annotated with biological events such as extinctions, speciations, gene duplications, lateral gene transfers, hybridizations, and so on.

The great challenge in the integration of phyloinformatic data is to interpret them and to draw conclusions from them. From a technological point of view, this issue might be addressed in part by the development of environments within which such data can be explored. Web-based tree visualization tools such as PhyloWidget (Jordan and Piel 2008), Archeopteryx (Han and Zmasek 2009), jsPhyloSVG (Smits and Ouverney 2010), and PhyloBox (http://phylobox.appspot.com/); environments for the exploration of comparative data and trees such as GenGIS (Parks et al. 2009), Nexplorer (Gopalan et al. 2006), Mesquite (Maddison and Maddison 2011), and Phenex (Balhoff et al. 2010); as well as more generic tools (e.g., semantic web browsers and map overlays) could be modified to facilitate the exploration of semantic metadata in combination with fundamental phylogenetic data. Such technological innovations would be aided by changes in our disciplinary culture: a greater willingness to participate in efforts to develop standards and to identify best practices and a greater willingness to adopt common standards for their long-term benefits, even when it is not the most convenient short-term choice.

Funding

R.A.V. received support from the CIPRES project (NSF #EF-03314953 to W.P.M.), the FP7 Marie Curie Programme (Call FP7-PEOPLE-IEF-2008—Proposal No. 237046) and, for the NeXML implementation in TreeBASE, the pPOD project (NSF IIS 0629846); P.E.M. and J.S. received support from CIPRES (NSF #EF-0331495, #EF-0715370); M.T.H. was supported by NSF (DEB-ATOL-0732920); X.X. received support from NSERC (Canada) Discovery and RTI grants; W.P.M. received support from an NSERC (Canada) Discovery grant; J.C. received support from a Google Summer of Code 2007 grant; A.P. received support from a Google Summer of Code 2010 grant.

Acknowledgments

Support for the work reported here came from many sources. The authors especially acknowledge NESCent (NSF #EF-0423641) for supporting an Evolutionary Informatics Working Group (R.A.V. and A.S., Co-PIs) that, during its 4 meetings from 2006 to 2009, sustained much work on NeXML. R.A.V., M.T.H., W.P.M., P.E.M., and J.S. participated in the earliest design discussions, subsequently joined by X.X. and H.L. J.P.B., H.L., and R.A.V. participated in the design and implementation of the metadata annotation schema. J.P.B. and R.A.V. participated in implementation of the Java API. X.X. implemented it in DAMBE. A.P. and R.A.V. implemented the Ruby API for NeXML and wrote the "Programmer's guide" section of the manual. R.A.V. and J.C. implemented the Perl API for NeXML. P.E.M. participated

in the design, testing, and implementation of the Java API and several derivative applications. J.S. and M.T.H. wrote the Python implementation in DendroPy and participated in design discussions. A.S. organized the analysis of use cases, wrote the "how to" section of the manual, and nagged the other authors. R.A.V., H.L., and A.S. drafted the manuscript and, along with W.P.M., P.E.M., and X.X., revised it. The authors are grateful to Sam Donnelly and David L. Swofford for their contributions to design and coding; to Andrew Rambaut, Jamie Whitacre, and Christian Zmasek for input on the file format comparison; and to Karen Cranston, Peter Foster, David Maddison, Mark Miller, Rod Page, William H. Piel, and an anonymous reviewer for comments on the manuscript. The identification of specific commercial software products is for the purpose of specifying a protocol and does not imply a recommendation or endorsement by the National Institute of Standards and Technology.

REFERENCES

Adida B., Birbeck M., McCarron S., Pemberton S. 2008. RDFa in XHTML: Syntax and Processing. Available from: http://www.w3.org/TR/rdfa-syntax/. [Wed Sep 21, 2011].

Ashburner M., Ball C.A., Blake J.A., Botstein D., Butler H., Cherry J.M., Davis A.P., Dolinski K., Dwight S.S., Eppig J.T., Harris M.A., Hill D.P., Issel-Tarver L., Kasarskis A., Lewis S., Matese J.C., Richardson J.E., Ringwald M., Rubin G.M., Sherlock G. 2000. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. Nat. Genet. 25:25–29.

Balhoff J.P., Dahdul W.M., Kothari C.R., Lapp H., Lundberg J.G., Mabee P., Midford P.E., Westerfield M., Vision T.J. 2010. Phenex: ontological annotation of phenotypic diversity. PLoS One 5:e10500.

Beaman R.S., Cellinese N. 2010. The tree of life knowledge and information network. Available from: http://www.tolkin.org. [Wed Sep 21, 2011].

Beckett D. 2004. RDF/XML syntax specification (revised). W3C Recommendation. Available from: http://www.w3.org/TR/REC-rdf-syntax/. [Wed Sep 28, 2011].

Benson D.A., Karsch-Mizrachi I., Lipman D.J., Ostell J., Sayers E.W. 2009. GenBank. Nucleic Acids Res. 37:D26–31.

Biron P.V., Malhotra A. 2004. XML schema part 2: datatypes second edition. W3C Recommendation. Available from: http://www.w3.org/TR/xmlschema-2/. [Wed Sep 21, 2011].

Bisby F.A. 2000. The quiet revolution: biodiversity informatics and the internet. Science. 289:2309–2312.

Brandes U., Eiglsperger M., Herman I., Himsolt M., Marshall M.S. 2002. GraphML progress report: structural layer proposal. In: Mutzel P., Jünger M., Leipert S., editors. Proceedings of the 9th International Symposium on Graph Drawing (GD 2001); 2001 Sep. 23–26; Berlin/Heidelberg: Springer. p. 109–112.

Cardona G., Rossello F., Valiente G. 2008. Extended Newick: it is time for a standard representation of phylogenetic networks. BMC Bioinformatics 9:532.

Connelly T., Sharp P., Ausiello D., Bronner-Fraser M., Burke I., Eisen J., Janetos A., Karp R., Kim P., Lauffenburger D., Lidstrom M., Lim W., McFall-Ngai M., Meyerowitz E., Yamamoto K. 2009. A new biology for the 21st century: ensuring the United States leads the coming biology revolution. Washington (DC): National Academies Press. p. 112.

Constable H., Guralnick R., Wieczorek J., Spencer C., Peterson A.T. 2010. VertNet: a new model for biodiversity data sharing. PLoS Biol. 8:e1000309.

Dahdul W.M., Balhoff J.P., Engeman J., Grande T., Hilton E.J., Kothari C., Lapp H., Lundberg J.G., Midford P.E., Vision T.J., Westerfield M., Mabee P.M. 2010a. Evolutionary characters, phenotypes and ontologies: curating data from the systematic biology literature. PLoS One 5:e10708.

Dahdul W.M., Lundberg J.G., Midford P.E., Balhoff J.P., Lapp H., Vision T.J., Haendel M.A., Westerfield M., Mabee P.M. 2010b. The teleost anatomy ontology: anatomical representation for the genomics age. Syst. Biol. 59:369–383.

Drummond A.J., Rambaut A. 2007. BEAST: Bayesian evolutionary analysis by sampling trees. BMC Evol. Biol. 7:214.

Fallside D.C., Walmsley P. 2004. XML schema part 0: primer second edition. W3C recommendation. Available from: http://www.w3.org/TR/xmlschema-0/. [Thu Sep 22, 2011].

Felsenstein J. 1989. PHYLIP—phylogeny inference package (version 3.2). Cladistics 5:164–166.

Felsenstein J. 2004. Inferring phylogenies. Sunderland (MA): Sinauer.

Gkoutos G.V., Green E.C.J., Mallon A.-M., Hancock J.M., Davidson D. 2004. Using ontologies to describe mouse phenotypes. Genome Biol. 6:R8.

Gopalan V., Qiu W.-G., Chen M.Z., Stoltzfus A. 2006. Nexplorer: phylogeny-based exploration of sequence family data. Bioinformatics 22:120–121.

Han M.V., Zmasek C.M. 2009. phyloXML: XML for evolutionary biology and comparative genomics. BMC Bioinformatics 10:356.

Hladish T., Gopalan V., Liang C., Qiu W., Yang P., Stoltzfus A. 2007. Bio::NEXUS: a Perl API for the NEXUS format for comparative biological data. BMC Bioinformatics 8:191.

Hyam R. 2008. TDWG technical roadmap 2008. Available from: http://www.tdwg.org/fileadmin/subgroups/tag/TAG_Roadmap_2008.pdf. [Wed Sep 21, 2011].

Johnson N.F. 2007. Biodiversity informatics. Annu. Rev. Entomol. 52:421–438.

Jordan G.E., Piel W.H. 2008. PhyloWidget: web-based visualizations for the tree of life. Bioinformatics 24:1641–1642.

Leary P.R., Remsen D.P., Norton C.N., Patterson D.J., Sarkar I.N. 2007. uBioRSS: tracking taxonomic literature using RSS. Bioinformatics 23:1434–1436.

Leebens-Mack J., Vision T.J., Brenner E., Bowers J.E., Cannon S., Clement M.J., Cunningham C.W., dePamphilis C., deSalle R., Doyle J.J., Eisen J.A., Gu X., Harshman J., Jansen R.K., Kellogg E.A., Koonin E.V., Mishler B.D., Philippe H., Pires J.C., Qiu Y.L., Rhee S.Y., Sjolander K., Soltis D.E., Soltis P.S., Stevenson D.W., Wall K., Warnow T., Zmasek C. 2006. Taking the first steps towards a standard for reporting on phylogenies: Minimum Information About a Phylogenetic Analysis (MIAPA). Omics 10: 231–237.

Lewis P.O. 2003. NCL: a C++ class library for interpreting data files in NEXUS format. Bioinformatics 19:2330–2331.

Li H., Coghlan A., Ruan J., Coin L.J., Hériché J.-K., Osmotherly L., Li R., Liu T., Zhang Z., Bolund L., Wong G.K.-S., Zheng W., Dehal P., Wang J., Durbin R. 2006. TreeFam: a curated database of phylogenetic trees of animal gene families. Nucleic Acids Res. 34:D572–D580.

Maddison D.R., Maddison W.P. 2005. MacClade 4: analysis of phylogeny and character evolution. Version 4.08a. Available from: http://macclade.org/. [Thu Sep 22, 2011].

Maddison D.R., Schulz K.-S., Maddison W.P. 2007. The tree of life web project. Zootaxa 1668:19–40.

Maddison D.R., Swofford D.L., Maddison W.P. 1997. NEXUS: an extensible file format for systematic information. Syst. Biol. 46: 590–621.

Maddison W.P., Maddison D.R. 2011. Mesquite: a modular system for evolutionary analysis. Version 2.74. Available from: http://mesquiteproject.org. [Wed Sep 21, 2011].

Matthews S.J., Sul S.-J., Williams T.L. 2010. A Novel Approach for Compressing Phylogenetic Trees . In: Borodovsky, M., Gogarten J., Przytycka T., Rajasekaran S., editors. Bioinformatics research and applications. Berlin (Germany): Springer. p. 113–124.

McEntire R., Karp P., Abernethy N., Benton D., Helt G., DeJongh M., Kent R., Kosky A., Lewis S., Hodnett D., Neumann E., Olken F., Pathak D., Tarczy-Hornoch P., Toldo L., Topaloglou T. 2000. An evaluation of ontology exchange languages for bioinformatics. Proc. Int. Conf. Intell. Syst. Mol. Biol. 8:239–250.

Miller M.A., Pfeiffer W., Schwarz T. 2010. Creating the CIPRES Science Gateway for inference of large phylogenetic trees. Gateway Computing Environments Workshop (GCE). 2010. p. 1–8.
Moore A.J., McPeek M.A., Rausher M.D., Rieseberg L., Whitlock M.C. 2010. The need for archiving data in evolutionary biology. J. Evol. Biol. 23:659–660.
Mungall C.J., Gkoutos G.V., Smith C.L., Haendel M.A., Lewis S.E., Ashburner M. 2010. Integrating phenotype ontologies across multiple species. Genome Biol. 11:R2.
O'Leary M.A., Kaufman S.G. 2007 MorphoBank 2.5: web application for morphological phylogenetics and taxonomy. Available from: http://www.morphobank.org. [Wed Sep 21, 2011].
Page R.D.M. 2008. Biodiversity informatics: the challenge of linking data and the role of shared identifiers. Brief. Bioinform. 9: 345–354.
Parks D.H., Porter M., Churcher S., Wang S., Blouin C., Whalley J., Brooks S., Beiko R.G. 2009. GenGIS: a geospatial information system for genomic data. Genome Res. 19:1896–1904.
Piel W.H., Chan L., Dominus M.J., Ruan J., Vos R.A., Tannen V. 2009. TreeBASE v. 2: a database of phylogenetic knowledge. London: e-BioSphere.
Prosdocimi F., Chisham B., Pontelli E., Thompson J.D., Stoltzfus A. 2009. Initial implementation of a comparative data analysis ontology. Evol. Bioinform. Online 5:47–66.
Rausher M.D., McPeek M.A., Moore A.J., Rieseberg L., Whitlock M.C. 2010. Data archiving. Evolution 64:603–604.
Rice K.A., Donoghue M.J., Olmstead R.G. 1997. Analyzing large data sets: rbcL 500 revisited. Syst. Biol. 46:554–563.
Ronquist F., Huelsenbeck J.P. 2003. MrBayes 3: Bayesian phylogenetic inference under mixed models. Bioinformatics 19:1572–1574.
Sanderson M.J., Boss D., Chen D., Cranston K.A., Wehe A. 2008. The PhyLoTA browser: processing GenBank for molecular phylogenetics research. Syst. Biol. 57:335–346.
Schmitt T., Messina D.N., Schreiber F., Sonnhammer E.L. 2011. SeqXML and OrthoXML: standards for sequence and orthology information. Brief. Bioinform. 12:485–488.
Sidlauskas B., Ganapathy G., Hazkani-Covo E., Jenkins K.P., Lapp H., McCall L.W., Price S., Scherle R., Spaeth P.A., Kidd D.M. 2010. Linking big: the continuing promise of evolutionary synthesis. Evolution 64:871–880.
Smits S.A., Ouverney C.C. 2010. jsPhyloSVG: a javascript library for visualizing interactive and vector-based phylogenetic trees on the web. PLoS One 5:e12267.
Stoesser G., Sterk P., Tuli M.A., Stoehr P.J., Cameron G.N. 1997. The EMBL nucleotide sequence database. Nucleic Acids Res. 25: 7–14.
Sukumaran J., Holder M.T. 2010. DendroPy: a Python libary for phylogenetic computing. Bioinformatics 26:1569–1571.
Swofford D.L. 2002. PAUP*: phylogenetic analysis using parsimony (and other methods), 4.0 beta. Sunderland (MA): Sinauer.
Taylor C.F., Field D., Sansone S.A., Aerts J., Apweiler R., Ashburner M., Ball C.A., Binz P.A., Bogue M., Booth T., Brazma A., Brinkman R.R., Michael Clark A., Deutsch E.W., Fiehn O., Fostel J., Ghazal P., Gibson F., Gray T., Grimes G., Hancock J.M., Hardy N.W., Hermjakob H., Julian R.K. Jr., Kane M., Kettner C., Kinsinger C., Kolker E., Kuiper M., Le Novere N., Leebens-Mack J., Lewis S.E., Lord P., Mallon A.M., Marthandan N., Masuya H., McNally R., Mehrle A., Morrison N., Orchard S., Quackenbush J., Reecy J.M., Robertson D.G., Rocca-Serra P., Rodriguez H., Rosenfelder H., Santoyo-Lopez J., Scheuermann R.H., Schober D., Smith B., Snape J., Stoeckert C.J. Jr., Tipton K., Sterk P., Untergasser A., Vandesompele J., Wiemann S. 2008. Promoting coherent minimum reporting guidelines for biological and biomedical investigations: the MIBBI project. Nat. Biotechnol 26:889–896.
Than C., Ruths D., Nakhleh L. 2008. PhyloNet: a software package for analyzing and reconstructing reticulate evolutionary relationships. BMC Bioinformatics 9:322.
Thompson H.S., Beech D., Maloney M., Mendelsohn N. 2004 XML schema part 1: structures second edition. W3C Recommendation. Available from: http://www.w3.org/TR/xmlschema-1/. [Wed Sep 21, 2011].
Whelan S., de Bakker P.I.W., Quevillon E., Rodriguez N., Goldman N. 2006. PANDIT: an evolution-centric database of protein and associated nucleotide domains with inferred trees. Nucleic Acids Res. 34: D327–D331.
Whitlock M.C., McPeek M.A., Rausher M.D., Rieseberg L., Moore A.J. 2010. Data archiving. Am. Nat. 175: 145–146.
Xia X., Xie Z. 2001. DAMBE: software package for data analysis in molecular biology and evolution. J. Hered. 92: 371–373.
Zmasek C.M., Eddy S.R. 2001. ATV: display and manipulation of annotated phylogenetic trees. Bioinformatics 17: 383–384.

APPENDIX 1

Abbreviations

ASN.1: Abstract Syntax Notation One (ASN.1) is an object representation language well suited to highly structured data (see McEntire et al. 2000). ASN.1 is used internally at NCBI.

CDAO: the Comparative Data Analysis Ontology (Prosdocimi et al. 2009; http://www.evolutionaryontology.org/cdao), an ontology developed in Web Ontology Language (OWL) to formalize the concepts and relations used in evolutionary comparative analysis, such as phylogenetic trees, OTUs, and character-state data.

DNS: Domain Name System, a hierarchical distributed naming system for resources connected to the Internet. DNS is used to translate human-readable names (e.g., nexml.org) into globally unique, numerical addresses used by networking equipment. Such human-readable domain names are often part of GUID schemes such as LSIDs and HTTP URIs.

EvoInfo: The Evolutionary Informatics Working Group supported by NESCent from 2006 to 2009 (http://evoinfo.nescent.org) spawned a variety of projects, including NeXML, CDAO, and PhyloWS.

GraphML: a file format for graphs (Brandes et al. 2002). It consists of a language core describing the structural properties of graphs and an extension mechanism to add application-specific data.

GUID: Globally Unique Identifier, an identifier, that is, a string of text, intended to identify one and only one object (e.g., a concept, a species, a publication). Different schemes have been devised for this, among which are LSIDs, DOIs, and HTTP URIs. A characteristic shared by a number of GUID schemes is that they are frequently a combination of a (sometimes DNS-based) "naming authority" part and a local identifier that is managed by the naming authority.

HTTP: HyperText Transfer Protocol, the data transfer protocol used on the World Wide Web. HTTP can be used as a technology upon which GUID schemes can be built because it, in turn, builds on a scheme for uniquely identifying addresses (DNS) and because it defines a mechanism for resolving those addresses and returning content, such that information about an object that

is identified using an HTTP-based GUID can be looked up.

JSON: JavaScript Object Notation (http://www.json.org), a lightweight open standard for representing structured data originally based on the syntax for data structures of the JavaScript programming language. XML, which is more verbose, can be translated to JSON, allowing for more concise transmissions of NeXML data in situations where bandwidth is at a premium, for example, inside a web browser window.

LSID: Life Science Identifier, a means to identify a piece of biological data using an URN scheme (see URI, below) comprised of an authority, a namespace, an object identifier, and an optional version number. HTTP URI serves the same function and is more widely used and supported.

MIAPA: Minimum Information for a Phylogenetic Analysis, a draft proposal for a MIBBI (Minimum Information for Biological and Biomedical Investigations) standard, specifying the key information for authors to include in a phylogenetic record in order to facilitate the reuse of the phylogenetic data and validation of phylogenetic results.

NCBI: the National Center for Biotechnology Information (http://ncbi.nlm.nih.gov), part of the United States National Library of Medicine, a branch of the National Institutes of Health. NCBI provides access to biomedical and genomic information. In particular, its databases of DNA sequence data (GenBank, http://www.ncbi.nlm.nih.gov/genbank/) and its taxonomy (http://www.ncbi.nlm.nih.gov/Taxonomy/taxonomyhome.html) are relevant to the comparative biology community. Due to NCBI's sheer size and longevity, many of the technology choices it made (e.g., for its sequence and taxon identifiers, its sequence file formats) have become de facto standards.

OBO: Open Biological and Biomedical Ontologies (http://www.obofoundry.org/), a consortium of developers of science-based ontologies with the goal of creating a suite of orthogonal, interoperable reference ontologies in the biomedical domain.

OWL: Web Ontology Language, a knowledge representation metalanguage for authoring the formal semantics of ontologies commonly serialized as RDF/XML.

RDF: Resource Description Framework (Beckett 2004), consisting of a set of W3C specifications for conceptually describing objects (e.g., by their attributes) and the relationship among the objects (e.g., by the changes of their attributes in response to changes in the attributes of other objects). One of the applications of RDF is in the development of database schemas.

RDFa: Resource Description Framework in attributes, which extends XHTML and other XML formats to allow data described in RDF to be rendered into well-formed XML documents. RDFa therefore bridges RDF to the XML-based web and database world.

RDFS: RDF Schema, a semantic extension of RDF that defines a set of classes and properties using the RDF language. These classes and properties provide basic elements for the description of RDF vocabularies or ontologies.

RDF/XML: RDF serialized as XML.

SKOS: Simple Knowledge Organization System (http://www.w3.org/2004/02/skos/core#), which is a family of formal languages designed for representation of knowledge in the form of trees and networks in specific ontologies (This representation is often achieved through RDF and RDF-schema). SKOS, together with the publication of the SKOS-organized data as web documents and the computational infrastructure for automating the processing of such web documents, makes up the semantic web.

uBio: the Universal Biological Indexer and Organizer, http://www.ubio.org (see Leary et al. 2007). uBio records canonical names, vernacular names, synonyms and homonyms for biological taxa in its NameBank database, and anchors these recorded names on a number of widely used taxonomies, including the NCBI taxonomy. uBio also provides a number of web services, including ones that query its NameBank for occurrences of provided names (the *findIT* service).

URI: Uniform Resource Identifier, which can take 2 forms, the uniform resource name (URN) and uniform resource locator (URL). A digital object identifier (DOI) is an example of URN, for example, a journal paper can have a URN as doi:10.1093/molbev/msr005 and a URL as http://mbe.oxfordjournals.org/content/early/2011/01/07/molbev.msr005. URN and URL are analogous to a person's name and his street address where he can be found.

W3C: the World Wide Web Consortium, a standards body that published "recommendations" that formally describe technologies used on the world wide web, including, for our purposes, OWL, RDF, RDFa, RDFS, RDF/XML, SKOS, XHTML, XML, XPath, XQuery, XSD, and XSLT.

XHTML: Extensible HyperText Markup Language, an XML-based, stricter version of HTML, the markup language in which pages on the World Wide Web are authored.

XML: Extensible Markup Language (XML), a metalanguage consisting of a set of rules for encoding data in machine-readable form in user-defined, customized domain languages, of which NeXML is an example.

XPath: the XML Path Language, which is a query language for selecting nodes from an XML document which is represented by a hierarchical multi-furcating tree. The query language

facilitates the tree traversal by allowing the selection of specific nodes in the tree through a variety of criteria. It is used in XML parsers and other software programs that process XML documents.

XQuery: a query and functional programming language that is intended to achieve the ultimate objective of seamlessly integrating the web and the database, that is, when both are based on XML and therefore can be accessed and processed in the same way. XPath is a component of XQuery.

XSD: XML Schema Definition, a language for describing the syntax and grammar of an XML-based domain language such as NeXML (see Biron and Malhotra 2004; Fallside and Walmsley 2004; Thompson et al. 2004; for the formal W3C recommendations).

XSLT: Extensible Stylesheet Language Transformations, which can take an XML document and convert it either into another XML document or a non-XML document containing either the same or a subset of the information in the original XML document. It does this by applying transformation templates on XPath expressions that select patterns in a source XML document. For example, a mitochondrial genomic sequence stored in the XML format in GenBank can be rendered by XSLT to other sequence format (e.g., FASTA or HTML for web display) or to another XML file containing a subset of information (e.g., containing only coding sequences in the genome).

APPENDIX 2

Technology Choices

The specific choice to implement a file format standard using XML, with RDF extensions, is based on the following considerations.

JSON is a popular, lightweight serialization format that would satisfy the scalability requirement (*requirement 9*); however, facilities for syntax validation (*req. 3*), annotation (*req. 2*), and processing using off-the-shelf tools (*req. 6*) were in their infancy at the time of this project. Character encoding of JSON documents can be specified in an HTTP header as it is being downloaded (as is the case for any text format), but this information is typically lost once the document is stored locally on disk (which contravenes *req. 7*).

Flat text is a mature solution, easily readable, and is amenable to validation if designed from the start around a formally defined grammar. Whether off-the-shelf tools can process flat text (*req. 6*) depends on how it is structured. For example, a lot of information can be easily extracted from EMBL data files (Stoesser et al. 1997) with the standard UNIX text tools ("grep" and related tools) because lines in such files are prefixed with a 2-letter code that identifies which aspect of the data is described in that line. On the other hand, NEXUS is harder to process that way as the meaning of any particular word or number used in a NEXUS file depends on its location within a context of nested "blocks." Flat text is very flexible, but this also means that no obvious single solution exist for metadata annotation systems (*req. 2*) or for the validation of types (e.g., number formats, *req. 3*). For flat text, there is no standard way of self-describing a file's character encoding and grammar (*req. 7*).

ASN.1 (see McEntire et al. 2000) is a standard and flexible notation for describing data structures to be encoded, transmitted, and decoded across telecommunications and computer networks. The extent to which data that is described in ASN.1 is scalable (*req. 9*) or supports internationalization (*req. 10*) depends on the encoding that is used for data transmission: some encodings are highly concise and binary (e.g., "BER" encoding), whereas others are textual (e.g., "XER" encoding). ASN.1 documents do not self-describe the syntax or encoding in which they are written (*req. 7*). Indeed, most use cases for ASN.1 are for the ephemeral transmission of data, not for storage, which obviates the need for self-description, as the participants in a transmission agree upon these metadata. Parsers for specific ASN.1 implementations (e.g., GenBank records in ASN.1) can be generated from their specification for a number of programming languages, however, no specification-agnostic off-the-shelf tools for processing or querying any ASN.1 document exist (*req. 6*).

RDF is conceptually attractive for expressing metadata annotations: RDF uses a model that captures semantic data as "triples" that link fundamental data to other objects via "predicates" (*req. 2*) in a way that is designed to resolve conceptual ambiguities (*req. 4*). However, its flexibility in the way it is serialized (e.g., as RDF/XML, Turtle or N3) and in the way relationships between entities can be expressed (e.g., as nested RDF/XML tags or ID references) make it difficult to process (*req. 6*) and potentially verbose (*req. 9*). Best practices for RDF dictate that relationships between entities are identified by querying an RDF graph, that is, by exploring the semantics of the data, not by treating the RDF as a structured syntax, which complicates the workflow for processing, querying, and transforming RDF data (*req. 6*).

XML and related technologies such as XML Schema (a technology that can be used to satisfy *req. 3*), XPath, XQuery, and XSLT are very well suited for defining strict syntax for structured data. XML is probably the most commonly used solution for this kind of problem, and a rich ecosystem of tools to validate, process, query, and transform XML documents exists for a variety of programming languages (*req. 6*). XML documents can self-describe the grammar and character encoding in which they are written (*req. 7* and 10) by, respectively, specifying namespaces to which tokens in the document are bound and the schema against which those tokens in that namespace are validated and by including a processing instruction at the top of the document that specifies the character encoding (e.g., Unicode). The utility of XML for addressing data exchange obstacles in biological research, including for phylogenetics, has been recognized before (Felsenstein 2004). XML applications have been developed for the Tree of Life web project (Maddison et al. 2007), for the annotation of gene evolution and other events on phylogenies (Han and Zmasek 2009), and for the BEAST software for Bayesian MCMC analysis of molecular sequences (Drummond and Rambaut 2007). In addition, the recent advent of RDFa (Adida et al. 2008) allows data objects to be annotated with the expressiveness of RDF (*req. 2*) but in a syntax that must conform to XML schema, that is, with more predictable structure.