RNA (2002), *8*:707–717. Cambridge University Press. Printed in the USA.

DOI: 10.1017.S1355838202028017

SYNTAX PROPOSAL

RNAML: A standard syntax for exchanging RNA information

ALLISON WAUGH,[1] PATRICK GENDRON,[2] RUSS ALTMAN,[1] JAMES W. BROWN,[3] DAVID CASE,[4] DANIEL GAUTHERET,[5] STEPHEN C. HARVEY,[6] NEOCLES LEONTIS,[7] JOHN WESTBROOK,[8] ERIC WESTHOF,[9] MICHAEL ZUKER,[10] and FRANÇOIS MAJOR[2]

[1]Stanford Medical Informatics, Stanford University Medical Center, Stanford, California 94305, USA
[2]Département d'Informatique et de Recherche Opérationnelle, Université de Montréal, Montréal, Québec H3C 3J7, Canada
[3]Department of Microbiology, North Carolina State University, Raleigh, North Carolina 27695, USA
[4]Department of Molecular Biology, The Scripps Research Institute, La Jolla, California 92037, USA
[5]Centre National de la Recherche Scientifique, Marseille 28809, France
[6]Department of Biochemistry, University of Alabama at Birmingham, Birmingham, Alabama 35294, USA
[7]Department of Chemistry, Bowling Green State University, Bowling Green, Ohio 43403, USA
[8]Department of Chemistry, Rutgers University, Piscataway, New Jersey 08855, USA
[9]IBMC-Centre National de la Recherche Scientifique, Strasbourg 67084, France
[10]Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, New York 12180, USA

ABSTRACT

Analyzing a single data set using multiple RNA informatics programs often requires a file format conversion between each pair of programs, significantly hampering productivity. To facilitate the interoperation of these programs, we propose a syntax to exchange basic RNA molecular information. This RNAML syntax allows for the storage and the exchange of information about RNA sequence and secondary and tertiary structures. The syntax permits the description of higher level information about the data including, but not restricted to, base pairs, base triples, and pseudoknots. A class-oriented approach allows us to represent data common to a given set of RNA molecules, such as a sequence alignment and a consensus secondary structure. Documentation about experiments and computations, as well as references to journals and external databases, are included in the syntax. The chief challenge in creating such a syntax was to determine the appropriate scope of usage and to ensure extensibility as new needs will arise. The syntax complies with the eXtensible Markup Language (XML) recommendations, a widely accepted standard for syntax specifications. In addition to the various generic packages that exist to read and interpret XML formats, an XML processor was developed and put in the open-source *MC-Core* library for nucleic acid and protein structure computer manipulation.

Keywords: data storage; file format; knowledge representation; XML

INTRODUCTION

An exchange format for RNA informatics

The need for informatics methods in RNA science has increased dramatically over the last few years. Large amounts of data generated by genome sequencing projects, microarray analyses, and RNA selection experiments have established a need for data storage, retrieval, and analysis methods. In addition, a number of computational capabilities have become critical for understanding RNA biology, such as sequence alignments (pairwise and multiple), secondary structure predictions, three-dimensional modeling, and the detection, classification, and description of recurrent motifs in primary, secondary, and tertiary structures. During the maturation of the RNA informatics field, individual laboratories developed many software programs that read input and write output files in formats unique to these given programs. Many of these programs can be used sequentially, and thus numerous format transformations must be applied to the data, obstructing productivity. A standard syntax for RNA information exchange that is supported by community members should in-

Reprint requests to: François Major, Département d'Informatique et de Recherche Opérationnelle, Université de Montréal, C.P. 6128, Succ. Centre-Ville, Montréal, Québec, H3C 3J7, Canada; e-mail: major@iro.umontreal.ca.

crease efficiency, opening new paths in the combination of existing RNA analysis programs.

Efforts to standardize the exchange of information in molecular biology are currently underway; however, these efforts generally are not targeted for the peculiarities of RNA science, and tend to focus on the exchange of genomic DNA information, protein sequence/structure information, or microarray data. For example, the need to specify RNA secondary structure is critical to many RNA projects, but is not the focus of genomic data representation efforts.

Existing formats do not capture details of RNA biology adequately

The primary efforts to standardize biological information exchange formats include the BIOpolymer Markup Language (BIOML) (Fenyo, 1999) project developed collaboratively by ProteoMetrics and LLC, the Bioinformatic Sequence Markup Language (BSML™) (Laurent & Biggar, 1999) developed and distributed by Visual Genomics, the Genome Annotation Markup Elements (GAME; http://www.bioxml.org/Projects/game/), and the CORBA Bio effort (Stevens & Miller, 2000).

BIOML provides an extensible framework for annotating experimental information about molecular entities, such as proteins and genes. BSML is an extensible language specification and container used to represent and display genetic sequence information. Although both BIOML and BSML include RNA information, the syntaxes are focused on gene-related elements of RNA, such as transcription start and stop sites and general base modifications or variants. Neither syntax addresses RNA structure information. GAME is focused on the programmatic needs of annotation exchange in genomics. RNA science is not a primary emphasis of this effort. The CORBA Bio effort aims to use CORBA's object orientation as a translator between diverse biological data resources.

Recognizing the growing need for a standard syntax, a group of RNA scientists gathered at the RNA Society meeting in Madison, Wisconsin, in 1998 to outline a format for RNA data exchange that, if widely adopted, would provide a single format for representing information specific to RNA molecules. After the original meeting, this same group met in San Diego, California, in May 1999 with RNA scientists active either in database curation or algorithm design. This article is the primary report of that meeting. After addressing the problems and challenges of building a general-purpose syntax, the RNAML syntax, we established a set of design principles and chose the Extensible Markup Language 1.0 (XML, http://www.w3.org/TR/2000/REC-xml-20001006) as the data representation form. Achard et al. (2001) provide a detailed comparison of XML to other languages in the context of bioinformatics data representation.

XML is "the universal format for structured documents and data on the Web" (www.w3.org/XML). It is supported by the W3C, the World Wide Web consortium, which is the primary body for Web standards. XML sets conventions for designing text formats for data, producing unambiguous files that are extensible, supported internationally/locally, platform-independent, and easily generated and read by a computer. XML encodes a method for "marking-up" text so that the elements of the text have associated tags and attributes that provide the reader or parser information about the text. A document type definition (DTD) specifies a formal description of a particular type of document, outlining element names and types, restrictions on attribute or element values, and the hierarchical order in which elements appear in the document. All documents of a particular type must be constructed and named in a consistent and conformant manner, providing applications with advance notice of the names and structures that compose the particular document type. Thus, given an XML file, one can understand the role that each piece of data is playing by referring to the DTD. The format that was developed for standard RNA data exchange is an XML-compliant DTD. The presentation of RNAML Version 1.0 is the focus of this article.

Primary advantages of XML include its growing support, as shown by the increasing availability of tools for processing XML files in all major programming languages; the ability to extend the syntax with new concepts without breaking programs that parse older versions of the syntax; and XML's capacity to logically organize different levels of detail using nested data structures. Chief disadvantages are that XML files can appear verbose to human readers and can be relatively large. However, XML text is unambiguous and easily readable, and the benefits of computer readability and robustness seem to outweigh these disadvantages.

Common usage of a standardized exchange format

A common scenario that can be imagined for use of the RNAML format follows:

1. Upon sequencing of a RNA molecule, create a simple, brief RNAML file composed of the sequence information.
2. As the nature of RNA base modifications are ascertained experimentally, add these sequence annotations to the file.
3. Run the sequence through a secondary structure prediction server, such as *MFOLD* (Mathews et al., 1999). The server would accept the RNAML format, extract the necessary information, complete computations, and supplement the input file with additional elements that specify the molecule's predicted

secondary structure (specification of helices and single-stranded regions in the sequence).
4. Other servers, perhaps specializing in the detection of pseudoknots, add further annotation to the file.
5. Send the file to a program that performs covariation analyses of multiple alignments to predict or confirm secondary structure, as well as long-range tertiary interactions. The computational results are inserted into the file.
6. Send the file to a three-dimensional modeling program, such as *MC-Sym* (Major et al., 1991), to build fragments of the molecule's structure and assign Cartesian coordinates to parts of the molecule based on its secondary structure. The program adds these coordinates to the growing file.
7. Perform laboratory experiments to confirm or contradict all or part of the structure and add a trace of these results into the file.
8. Finally, a modeling program (or modeler) combines all the fragments into a single structural hypothesis [e.g., using Manip (Massire & Westhof, 1999)] and supplements the file with the remaining coordinates (perhaps changing existing coordinates) that are necessary to fully express the model.

By the end of this process, the file has grown in size and richness of information, and because conformance to the RNAML format has been maintained throughout, the file can be "recycled" at any time through any of the mentioned programs to recompute features or update information. With time, the file should accurately represent a summary of our accumulated knowledge on a given RNA molecule. Note that the RNAML syntax is intended to provide and exchange RNA information among programs; some specific program arguments and constraints must be dealt with separately.

Alternately to this iterative scenario, a RNAML document can be created to maintain annotations of existing structures determined by experimental methods and thus serve as a repository for storing analyses and observations made on the predetermined RNA. A RNAML document can also be used to carry only a restricted portion of information about an RNA molecule that is of concern for a particular study. Ultimately, the resulting documents can be disseminated to interested parties, all while following a standardized and familiar syntax.

MATERIALS AND METHODS

Design principles

The resources that were particularly considered while creating a syntax for RNAML include *MFOLD* (secondary structure prediction), *MC-Sym* (tertiary structures and models), and the RNA databases, such as the RNase P Database (Brown, 1999), the Ribosomal Database Project (RDP; Maidak et al., 2000), RiboWEB (Bada & Altman, 2000), the Nucleic Acid Database (NDB; Berman et al., 1992), and the Protein Data Bank (PDB; Berman et al., 2000). However, the application of the syntax should be general. Special care was taken to ensure compatibility in markup expressiveness with existing databases such as GenBank and PDB (Berman et al., 2000) so that any kind of RNA data, from genomic to three-dimensional, can be inserted into a RNAML document.

Although XML can provide a solution to the syntax issue of elaborating a new exchange format, it does not help in resolving inconsistencies in semantics. The problem of defining reliable semantics is much more difficult than defining a common syntax and requires agreement among members of the involved community. For example, a syntax might represent a helix with the start base, its partner, and the length (in base pairs) of the helix. The semantics might specify that a helix is a run of two or more antiparallel Watson–Crick base pairs with no gaps or bulges. Someone may specify a correct syntax (base-id-5p = 12, base-id-3p = 29, length = 5), which implies base pairs from nt 12 and 29 through 16 and 25. But the syntax could also be used to specify another helix (base-id-5p = 10, base-id-3p = 11, length = 6), which, though syntactically legal, is semantically incorrect. In addition, the semantics may be presumably specified but may not handle all cases. For example, one may assume RNA double helices are A-form and others A′-form.

For the purpose of RNAML, we strived to avoid semantically contentious issues while taking the first step of specifying a syntax. The syntax allows for the definition of helices, bulges, loops, and other concepts and abstractions from RNA structure, but does not attempt to constrain their meaning. Indeed, the precise definition of secondary structure evolved as the subject of significant debate and is not even agreed upon in the RNA community.

The hierarchical (nested) nature of XML mimics the inherent hierarchical nature of biological molecules (a molecule is composed of bases, which are composed of atoms). Thus, XML provides a valuable method for logically structuring knowledge about RNA molecules into a single text document (see Fig. 1). An RNAML document can represent one RNA molecule as a sequence, possibly annotated, along with a set of structures that describe the RNA under various conditions or modeling experiments. RNA molecules that are evolutionarily related may further be enclosed in a class element that expresses their relationship by incorporating such elements as a consensus structure and sequence alignments. Multiple tentative alignments from multiple authors can also be represented in a single RNAML file. Furthermore, intermolecular interactions can be represented using the syntax, as many different RNAs or RNA classes can be included in a single document.

Existing standards

RNAML compatibility with existing standards is critical. The most central standards used in RNAML include the International Union of Pure and Applied Chemistry (IUPAC) lettering (International Union of Pure and Applied Chemistry & International Union of Biochemistry and Molecular Biology, 1992), base modifications by McCloskey et al. (Limbach et al., 1994), PDB ATOM records, and base pair nomenclatures (Leontis & Westhof, 2001).

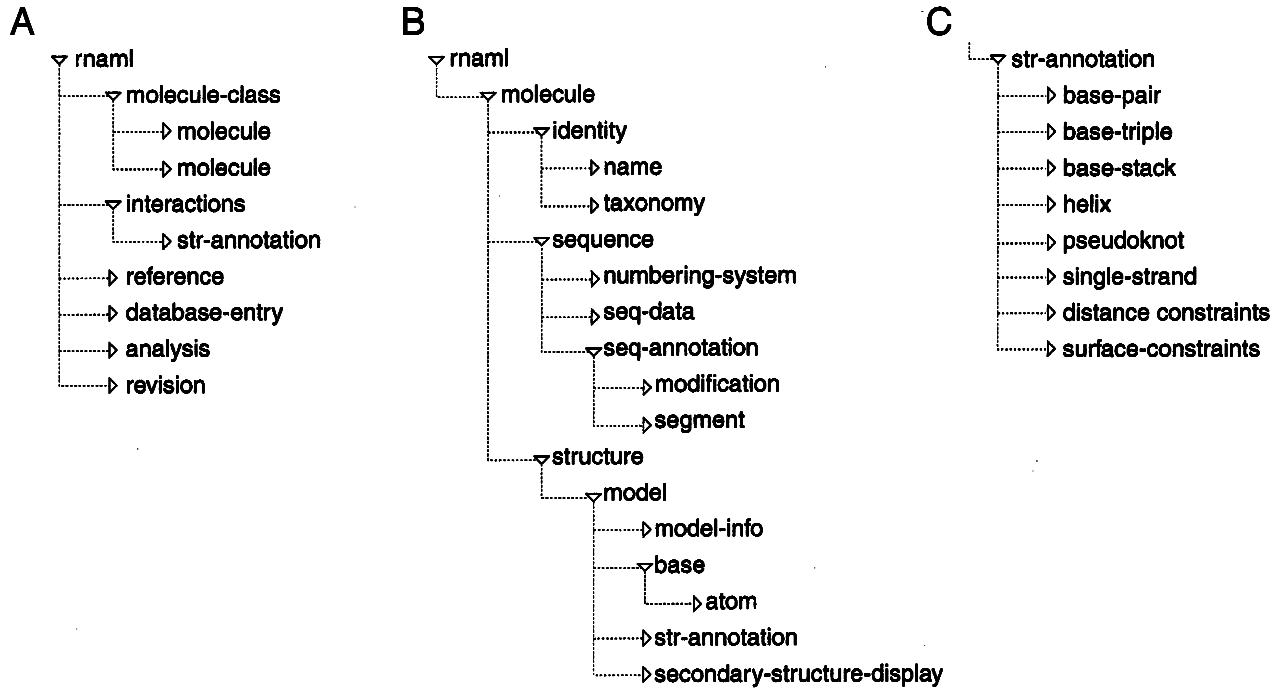

FIGURE 1. Schematic view of some markup elements defined in the RNAML syntax. **A:** The basic markup elements of the RNAML format that appear under the top level `rnaml` element. **B:** Elements contained in the description of a `molecule` element. **C:** Elements contained in the annotation of tertiary structures.

The IUPAC nomenclature for specifying bases and ambiguous bases was adopted. Two additional characters are acceptable. The character "." is used to indicate fuzziness about a base or its presence. The character "-" is used to fill the gaps in an alignment. McCloskey and coworkers have compiled a list of posttranslationally modified RNA bases and present it on a public web site (http://www-medlib.med.utah.edu/RNAmods/). The symbols for these modified bases were adopted for use in the sequence annotation records of RNAML. The characters in superscript can be included in the ASCII representation of the base symbols, but of course not as superscripts.

Atomic coordinates are included in RNAML similarly to existing formats such as NDB and PDB, which are built upon the macromolecular crystallographic interchange format (mmCIF). RNAML was carefully elaborated to provide compatibility with the fields of PDB ATOM records. All known and predicted base-pairing types can be described in RNAML using the unambiguous and descriptive nomenclature of Leontis and Westhof (2001). Note that most secondary and tertiary annotations of RNAML can be computed from a tertiary structure by using the *MC-Annotate* program (Gendron et al., 2001).

RESULTS

RNAML version 1.0 was created, and the DTD for it is shown in Appendix A. Examples, the DTD file, and documentation on the syntax are available at the project's web site (http://www-lbit.iro.umontreal.ca/rnaml/).

Many RNAML elements are optional. A user can create a file that makes extensive use of the markup capabilities or can report simple sequence or structure information with little additional information. The syntax allows one to put comments, both within and outside of the XML markup. The parser must pass internal comments to the application programs that could then properly treat the information.

Following the markup-oriented approach of XML, all data in an RNAML document is inserted as parsable CDATA (#PCDATA) and markups are used to classify the data within the hierarchy. In this context, no validation of the data, using the DTD, is made. For example, even though the IUPAC nomenclature and the McCloskey's modifications are accepted as standard, the content of the corresponding XML elements is not restricted. This may be seen as a problem, but, in fact, is a task that must be dealt with at another level, such as during data parsing.

RNAML documents must specify which version of the DTD is to be used to validate their content, eliminating possible confusion when several versions exist. Requiring explicit documentation of the version is also particularly useful if one wants to write "static" parsers, as compared to relying on generic XML parsers.

An RNAML document can describe one or more RNA molecules, intermolecular RNA interactions, and reference information under the top-level markup `rnaml` (see Fig. 1A). Each RNA molecule is characterized using

the identity element, which specifies the name of the molecule and the taxonomic information about the species, strain, and other categories of this molecule within the Linnaean system. A full description of the sequence and structure may also be contained in the molecule element (see Fig. 1B for a schematic view of some elements).

Sequence

The sequence element may contain one or more numbering systems followed by the actual sequence data and possible annotations. The length attribute is required as a forward indication for parsers of the number of bases, *N*, that the seq-data element contains. The seq-data should contain a string of IUPAC symbols of the specified length. Space and end-of-line characters are ignored. The seq-data element is optional.

The numbering-system elements establish multiple numbering and index systems over a single sequence, which allows an author to map his base numbering method to other numbering systems. For example, a tRNA molecule may have its "natural" numbering system, from x to $x + N - 1$, but may also be numbered according to the canonical numbering system from 1 to *N*. This would create two numbering systems in the document, which can easily be mapped to each other by a program. A numbering system is composed of one numbering-table, that is a sequence of *N* base identifiers, ids, not necessarily consecutive, or of numbering-range elements. Each numbering-range specifies the start and end of consecutive base positions in a portion of the whole sequence. Numbering tables are useful to depict base insertions in a sequence, that is, bases that all possess the same numeric position and are distinguished by insertion symbols. Numbering ranges are most useful to lessen the size of documents that contain large RNAs using natural numberings.

Sequence annotation (seq-annotation) elements indicate base modifications and segment identifications. Base modifications are specified by the base position and the standard base modification type. Segment identifications are specified by a segment name and its location in the sequence.

Consecutive bases in a sequence description are covalently connected unless otherwise indicated. Gaps should be represented using the appropriate symbol, and cut sequences expressed with as many sequence elements as there are subsequences.

Base identifiers in an RNAML document must contain not only the position in the sequence, but also a reference to a molecule and a model to ensure uniqueness. In a sense, the molecule element is equivalent to the chain id in the PDB format. Insertions made to natural molecules are indicated using a letter in the position element, following the base number.

Structure

Because an RNA structure is a dynamic system that varies in shape and function depending on the environmental and experimental conditions, the structure element of an RNA molecule description is comprised of a set of models that represents the RNA in different states. Models can also represent alternate hypotheses of a RNA structure in a given state, obtained by modeling, for instance.

Each model element possesses information about the method used in its determination as well as the resolution of the model when available. A set of base elements provides information on the base type, alternate location and insertion tag, and atomic description of each base. Atomic descriptions include Cartesian coordinates for each atom named using standard PDB symbols. Additional information about each atom, such its occupancy, temperature factor, segment id, element, and charge, is included for compatibility with PDB ATOM fields.

An RNA structure model may be described in terms of higher level components to produce a symbolic annotation of the geometric organization of the molecule. An annotation (str-annotation) may include base conformations, base pairs, base stacking, base triples, helices, single-stranded regions, pseudoknots, and other motifs (see Fig. 1C). A base conformation supplies information regarding its pucker mode, orientation around the glycosyl angle, and torsion angles.

Base pair description is extensive and serves to express bases that interact through hydrogen bonds. An author can describe the interaction of the base edges, the orientation of the glycosidic bond and local strand, and the overall base pair geometry. Base stacking elements can be used to specify bases that are stabilized by the superimposition of their nitrogen rings. Base pairing and stacking have 5′ and 3′ base identifiers. A base triple is specified by three base identifiers.

Helix elements serve to define runs of antiparallel base-paired strands of RNA. The precise semantic of helices is not defined here, but it is generally accepted that a helix should be formed of at least two consecutive Watson–Crick or wobble G-U base pairs that stack on each other. According to this definition, bulged bases interrupt helices. Helices are specified with a start base, its base pair partner, and the length (in base pairs) of the helix. Pseudoknot elements describe interleaved strands of RNA that form distinct helices, and, hence, references to two helices define a pseudoknot.

Experimental, theoretical, or statistical constraints on the mean distance between two atoms in three-dimensional space and the solvent accessibility of a base can be expressed. There is no assumption that the three-dimensional coordinates provided in the model markup will satisfy the constraints expressed in these records, but the records allow information about

target distances (and their uncertainty) to be communicated. Distance constraints may come from NMR, crosslinking or labeling experiments, and so forth.

A model containing only an RNA secondary structure can be depicted simply with an empty atomic description and filled base-pair annotations. A `secondary-structure-display` element can be used to store the two-dimensional coordinates of bases in a secondary structure diagram, such as those produced by *MFOLD*. This allows a user to encapsulate folding results in an RNAML document.

Structure relationships

If multiple interacting RNA molecules are depicted, the author can document the interactions between these molecules in an `interaction` element. Distances, base pairing, helices, stacking information, and other relations between two molecules can be expressed. This element is making use of the same structural annotations as intramolecular interactions. Consequently, base references must contain the originating molecule.

Multiple RNAs

Information relative to a given set of RNA molecules can be represented in RNAML by using the `molecule-class` element. One use of this element is to group occurrences of the same functional RNA found in multiple organisms. Molecules contained in a class can either be included in the `molecule-class` element or referred to using the `molecule-id` element. A class may define a standard numbering system that is common to all molecules based on an alignment of their sequences. One or many alignments may be specified for a single class. A `consensus-molecule` element allows one to represent the common structural components of the molecules in the class. The common numbering system can be found in this `consensus-molecule` element.

Alignments are encoded using `seq-data` elements by allowing the common symbols "." and "-." The task of establishing the correspondence between the various numbering systems and the alignment data is left to programs that interpret RNAML documents.

It should be noted that a given RNA molecule should not be represented more than once in an RNAML document. The proposed organization is sufficiently flexible to express the information within a single molecule element.

Documentation

In addition to the specific elements that form the structure of an RNAML document, general attributes can be used to comment on each element. These attributes refer to actual elements located at the end of a document. Journal articles, database entries, and analysis results may be included. Articles allows one to give a formal reference to information found in the file. Database entries provide a mechanism by which element content may refer to external data found in other databases. A program that reads an RNAML document with such comments may access the referred database, either directly from the information placed in the document or by connecting to the appropriate database server. Finally, analysis elements allow one to give additional details about the methods, programs, and versions that were used to produce data placed in the file.

DISCUSSION

Other efforts to develop a standard syntax for areas within biology exist, most of which do not confer sufficient details for RNA databases and algorithms. Thus, the RNAML syntax has been designed with two purposes in mind. First, we want a syntax that is both immediately adoptable by application builders in RNA science and capable of allowing one to represent a large amount of application-independent information in an unambiguous and reusable syntax. Second, we would like to ensure that the syntax can easily be expanded as new types of information become available. We propose to set up an ad hoc committee for standards to which suggestions of syntax changes could be submitted and from which approvals could be received efficiently.

Because of XML's extensibility, growing support, and nested levels of complexity, we believe that a standard specified by an XML DTD has the greatest chance of success. Furthermore, because RNAML is XML compliant, users can automate the translation from RNAML to other bioinformatic XML-compliant DTDs using the extensible well-supported stylesheet language (XSL). XSL is a transformation and formatting language that, given a set of user-specified rules, can be used to rewrite the content of one XML document using a different set of XML tags, generating text as needed. The transformation rules are specified in an XSL stylesheet. Because the stylesheet complies with the XML syntax, the basic syntax is familiar to the users, and enables an XML processor to check for basic syntactic errors (http://www.w3.org/Style/XSL).

Because DTDs are designed to specify the structure of a text document, XML has no mechanisms for constraining the text content. Thus, a DTD cannot be used to specify numeric ranges of a text value, or to define limitations or checks on the text content. Only the markups can be controlled by the DTD. For instance, the RNAML DTD cannot enforce the use of IUPAC letters in sequence descriptions. Rather, the software application that uses the data must perform its own error checking to enforce this constraint. XML Schema tech-

nology may be useful in adding constraints of this type in the future (http://www.w3.org/XML/Schema).

To encourage the use of RNAML, we made available a web page that contains the DTD, a documentation, and examples (http://www-lbit.iro.umontreal.ca/rnaml/). The present authors will implement it and systematically use it in their program developments. In addition, the RNase P Database serves its documents in RNAML, and the *MFold* server and the *MC-Sym* modeling software will read and produce RNAML documents in the future. The nucleic acid database (NDB; Berman et al., 1992) is also prepared to archive and distribute RNAML for experimental as well as model structures.

ACKNOWLEDGMENTS

We thank the San Diego Supercomputer Center for hosting the May, 1995 meeting. R.B.A. is supported by National Institutes of Health Grant LM06244 and National Science Foundation Grant DBI-9600637. F.M. thanks the Canadian Institutes of Health Research for grant support.

Received February 14, 2002; returned for revision February 28, 2002; revised manuscript received March 21, 2002

REFERENCES

Achard F, Vaysseix G, Barillot E. 2001. XML, bioinformatics and data integration. *Bioinformatics 17*:115–125.

Bada M, Altman R. 2000. Computational modeling of structured experimental data. *Methods Enzymol 317*:470–491.

Berman H, Olson W, Beveridge D, Westbrook J, Gelbin A, Demeny T, Hsieh S-H, Srinivasan A, Schneider B. 1992. The nucleic acid database: A comprehensive relational database of three-dimensional structures of nucleic acids. *Biophys J 63*:751–759.

Berman H, Westbrook J, Feng Z, Gilliland G, Bhat T, Weissig H, Shindyalov I, Bourne P. 2000. The Protein Data Bank. *Nucleic Acids Res 28*:235–242.

Brown JW. 1999. The ribonuclease P database. *Nucleic Acids Res 27*:314.

Fenyo D. 1999. The biopolymer markup language. *Bioinformatics 15*:339–340.

Gendron P, Lemieux S, Major F. 2001. Quantitative analysis of nucleic acid three-dimensional structures. *J Mol Biol 308*:919–936.

International Union of Pure and Applied Chemistry & International Union of Biochemistry and Molecular Biology. 1992. *Biochemical Nomenclature and Related Documents*. Chapel Hill, NC: Portland Press.

Laurent SS, Biggar RJ. 1999. *Inside XML DTDs: Scientific and Technical*. Berkeley, CA: McGraw-Hill.

Leontis N, Westhof E. 2001. Geometric nomenclature and classification of RNA base pairs. *RNA 7*:499–512.

Limbach P, Crain P, McCloskey J. 1994. Summary: The modified nucleosides of RNA. *Nucleic Acids Res 22*:2183–2196.

Maidak BL, Cole JR, Lilburn TG, Parker CT, Saxman PR, Stredwick JM, Garrity GM, Li B, Olson GJ, Pramanik S, Schmidt TM, Tiedje JM. 2000. The RDP (ribosomal database project) continues. *Nucleic Acids Res 28*:173–174.

Major F, Turcotte M, Gautheret D, Lapalme G, Fillion E, Cedergren R. 1991. The combination of symbolic and numerical computation for three-dimensional modeling of RNA. *Science 253*:1255–1260.

Massire C, Westhof E. 1999. MANIP: An interactive tool for modelling RNA. *J Mol Graphics 16*:197–205, 255–257.

Mathews D, Sabina J, Zuker M, Turner D. 1999. Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. *J Mol Biol 288*:911–940.

Stevens R, Miller C. 2000. Wrapping and interoperating bioinformatics resources using corba. *Brief Bioinformat 1*:9–21.

APPENDIX

Following is the document type definition (DTD) for RNAML version 1.

```
<?xml version="1.0" encoding="iso-8859-1"?>

<!ELEMENT rnaml (molecule | molecule-class | interactions | reference |
                 database-entry | analysis | revision)*>
<!ATTLIST rnaml
          version (1.0)   #REQUIRED
   comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT molecule (identity?, sequence?, structure?)>
<!ATTLIST molecule
          id             ID          #REQUIRED
   type          (rna | dna)  "rna"
   comment        CDATA       #IMPLIED
          reference-ids  IDREFS       #IMPLIED
   analysis-ids   IDREFS       #IMPLIED
          database-ids   IDREFS       #IMPLIED>

<!ELEMENT identity (name, taxonomy?)>
<!ATTLIST identity
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT name (#PCDATA)>              <!-- type: string -->

<!ELEMENT sequence (numbering-system*, seq-data?, seq-annotation?)>
<!ATTLIST sequence
          length    CDATA          #IMPLIED
    circular  (true | false) #IMPLIED
          comment        CDATA     #IMPLIED
          reference-ids  IDREFS    #IMPLIED
   analysis-ids   IDREFS    #IMPLIED
          database-ids   IDREFS    #IMPLIED>

<!ELEMENT numbering-system ((numbering-range)* | numbering-table)>
<!ATTLIST numbering-system
          id             ID     #REQUIRED
          used-in-file  (true | false) "false"
   comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT numbering-range (start, (end | length))>
<!ATTLIST numbering-range
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT numbering-table (#PCDATA)> <!-- type:space delimited integers -->
<!ATTLIST numbering-table
          length    CDATA   #REQUIRED
   comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT seq-data (#PCDATA)>                          <!-- type: string -->
<!ATTLIST seq-data
   comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT seq-annotation (modification | segment)*>
<!ATTLIST seq-annotation
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT modification  (base-id, modified-type)>
<!ATTLIST modification
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT modified-type (#PCDATA)>                     <!-- type: string -->

<!ELEMENT segment  (seg-name?, base-id-5p, (base-id-3p | length))>
<!ATTLIST segment
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT seg-name (#PCDATA)>                          <!-- type: string -->

<!ELEMENT structure (model)*>
<!ATTLIST structure
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT model (model-info?, base*, str-annotation?, secondary-structure-
display?)>
<!ATTLIST model
          id  ID  #IMPLIED
   comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT model-info (method?, resolution?, free-energy*)>
<!ATTLIST model-info
          comment        CDATA   #IMPLIED
```

```
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT method      (#PCDATA)>                    <!-- type: string -->
<!ATTLIST method
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT resolution  (#PCDATA)>     <!-- type: float, unit: angstroms -->
<!ATTLIST resolution
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT free-energy (#PCDATA)>     <!-- type: float, unit: kcal/mole -->
<!ATTLIST free-energy
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT base      (position, base-type?, alt-loc?, insertion?, atom*)>
<!ATTLIST base
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT base-type (#PCDATA)>                    <!-- type: string -->
<!ELEMENT alt-loc   (#PCDATA)>                    <!-- type: char   -->
<!ELEMENT insertion (#PCDATA)>                    <!-- type: char   -->

<!ELEMENT atom (atom-type, coordinates?, occupancy?,
                temp-factor?, seg-id?, element?, charge?)>
<!ATTLIST atom
          serial         CDATA   #IMPLIED
   comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT atom-type   (#PCDATA)>                  <!-- type: string -->
<!ELEMENT coordinates (#PCDATA)>     <!-- three space delimited floats -->
<!ELEMENT occupancy   (#PCDATA)>                  <!-- type: float  -->
<!ELEMENT temp-factor (#PCDATA)>                  <!-- type: float  -->
<!ELEMENT seg-id      (#PCDATA)>                  <!-- type: string -->
<!ELEMENT element     (#PCDATA)>                  <!-- type: string -->
<!ELEMENT charge      (#PCDATA)>                  <!-- type: string -->

<!ELEMENT str-annotation ((base-conformation | base-pair |
                           base-triple | base-stack | helix |
      pseudoknot |
      single-strand | distance-constraint |
      surface-constraint)*)>
<!ATTLIST str-annotation
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT base-conformation (base-id, pucker?, glycosyl?,
                             base-torsion-angles?)>
<!ATTLIST base-conformation
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT pucker   (#PCDATA)>                     <!-- type: string -->
<!ELEMENT glycosyl (#PCDATA)>       <!-- type: string -->

<!ELEMENT base-torsion-angles (alpha?, beta?, gamma?, delta?, epsilon?,
                               zeta?, chi?, nu0?, nu1?, nu2?, nu3?, nu4?)>
<!ATTLIST base-torsion-angles
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT alpha   (#PCDATA)>          <!-- type: float, unit: degrees -->
<!ELEMENT beta    (#PCDATA)>          <!-- type: float, unit: degrees -->
<!ELEMENT gamma   (#PCDATA)>          <!-- type: float, unit: degrees -->
<!ELEMENT delta   (#PCDATA)>          <!-- type: float, unit: degrees -->
<!ELEMENT epsilon (#PCDATA)>          <!-- type: float, unit: degrees -->
<!ELEMENT zeta    (#PCDATA)>          <!-- type: float, unit: degrees -->
<!ELEMENT chi     (#PCDATA)>          <!-- type: float, unit: degrees -->
<!ELEMENT nu0     (#PCDATA)>          <!-- type: float, unit: degrees -->
<!ELEMENT nu1     (#PCDATA)>          <!-- type: float, unit: degrees -->
<!ELEMENT nu2     (#PCDATA)>          <!-- type: float, unit: degrees -->
<!ELEMENT nu3     (#PCDATA)>          <!-- type: float, unit: degrees -->
<!ELEMENT nu4     (#PCDATA)>          <!-- type: float, unit: degrees -->

<!ELEMENT base-pair (base-id-5p, base-id-3p, edge-5p?, edge-3p?,
                     bond-orientation?, strand-orientation?)>
<!ATTLIST base-pair
   comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>
<!ELEMENT base-id-5p         (base-id)>           <!-- type: string -->
<!ELEMENT base-id-3p         (base-id)>           <!-- type: string -->
<!ELEMENT edge-5p            (#PCDATA)>           <!-- type: string -->
<!ELEMENT edge-3p            (#PCDATA)>           <!-- type: string -->
<!ELEMENT bond-orientation   (#PCDATA)>           <!-- type: string -->
<!ELEMENT strand-orientation (#PCDATA)>           <!-- type: string -->

<!ELEMENT base-triple ((base-pair | base-pair-id),
                       (base-pair | base-pair-id),
        (base-pair | base-pair-id))>
<!ATTLIST base-triple
```

```
   comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT base-stack (base-id, base-id)>
<!ATTLIST base-stack
   comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT helix (base-id-5p, base-id-3p, length)>
<!ATTLIST helix
          id ID  #IMPLIED
   comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT pseudoknot (helix-id, helix-id)>
<!ATTLIST pseudoknot
   comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT helix-id EMPTY>
<!ATTLIST helix-id
          ref            IDREF   #REQUIRED
   comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT single-strand (segment)>
<!ATTLIST single-strand
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT distance-constraint (base-id, atom-type,
                               base-id, atom-type,
          mean, range?, weight?)>
<!ATTLIST distance-constraint
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>
<!ELEMENT mean   (#PCDATA)>          <!-- type: float, unit: angstroms -->
<!ELEMENT range  (#PCDATA)>          <!-- type: float, unit: angstroms -->
<!ELEMENT weight (#PCDATA)>                              <!-- type: float -->

<!ELEMENT surface-constraint (base-id, atom-type, surface-value?)>
<!ATTLIST surface-constraint
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>
<!ELEMENT surface-value (#PCDATA)>                    <!-- type: float -->

<!ELEMENT secondary-structure-display (ss-base-coord)*>
<!ATTLIST secondary-structure-display comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT ss-base-coord (base-id, coordinates)>
<!ATTLIST ss-base-coord
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT molecule-class (identity?, (molecule | molecule-id |
                          numbering-system |
                          consensus-molecule | alignment)*)>
<!ATTLIST molecule-class
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT consensus-molecule (alignment-id, molecule)>
<!ATTLIST consensus-molecule
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT alignment (ali-sequence*)>
<!ATTLIST alignment
          id   ID   #REQUIRED
   comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT ali-sequence (molecule-id, seq-data)>
<!ATTLIST ali-sequence
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT interactions (str-annotation)*>
<!ATTLIST interactions
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT taxonomy (domain?, kingdom?, phylum?, class?,
```

```
                order?, family?, genus?, species?, strain?)>
<!ATTLIST taxonomy
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>
<!ELEMENT domain  (#PCDATA)>                          <!-- type: string -->
<!ELEMENT kingdom (#PCDATA)>                          <!-- type: string -->
<!ELEMENT phylum  (#PCDATA)>                          <!-- type: string -->
<!ELEMENT class   (#PCDATA)>                          <!-- type: string -->
<!ELEMENT order   (#PCDATA)>                          <!-- type: string -->
<!ELEMENT family  (#PCDATA)>                          <!-- type: string -->
<!ELEMENT genus   (#PCDATA)>                          <!-- type: string -->
<!ELEMENT species (#PCDATA)>                          <!-- type: string -->
<!ELEMENT strain  (#PCDATA)>                          <!-- type: string -->

<!ELEMENT reference (author | title | journal |
                     date | pubmed-id |
      book-title | editor | volume |
      issue | pages | path)*>
<!ATTLIST reference
   id       ID     #REQUIRED
   comment  CDATA  #IMPLIED>

<!ELEMENT author     (person)>
<!ELEMENT title      (#PCDATA)>                       <!-- type: string -->
<!ELEMENT journal    (#PCDATA)>                       <!-- type: string -->
<!ELEMENT pubmed-id  (#PCDATA)>                       <!-- type: string -->
<!ELEMENT book-title (#PCDATA)>                       <!-- type: string -->
<!ELEMENT editor     (#PCDATA)>                       <!-- type: string -->
<!ELEMENT volume     (#PCDATA)>                     <!-- type: integer -->
<!ELEMENT issue      (#PCDATA)>                     <!-- type: integer -->
<!ELEMENT pages      (#PCDATA)>     <!-- type: comma delimited integers-->

<!ELEMENT database-entry (database, entry, path?)>
<!ATTLIST database-entry
          id       ID     #REQUIRED
   comment  CDATA  #IMPLIED
   reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>
<!ELEMENT database (#PCDATA)>                         <!-- type: string -->
<!ELEMENT entry    (#PCDATA)>                         <!-- type: string -->

<!ELEMENT analysis (program | date | author | reference)*>
<!ATTLIST analysis
          id      ID    #REQUIRED
   comment        CDATA   #IMPLIED
   reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT revision (#PCDATA | date)*>
<!ATTLIST revision
          comment        CDATA   #IMPLIED
          reference-ids  IDREFS  #IMPLIED
   analysis-ids   IDREFS  #IMPLIED
          database-ids   IDREFS  #IMPLIED>

<!ELEMENT url  (#PCDATA)>                             <!-- type: string -->
<!ELEMENT file (#PCDATA)>                             <!-- type: string -->

<!ELEMENT path (url | file)>

<!ELEMENT person (first-name, last-name, affiliation*)>
<!ELEMENT first-name   (#PCDATA)>                     <!-- type: string -->
<!ELEMENT last-name    (#PCDATA)>       <!-- type: string -->
<!ELEMENT affiliation (#PCDATA)>                      <!-- type: string -->

<!ELEMENT program (prog-name | prog-version | reference)*>
<!ELEMENT prog-name    (#PCDATA)>                     <!-- type: string -->
<!ELEMENT prog-version (#PCDATA)>       <!-- type: string -->

<!ELEMENT date (day?, month?, year)>
<!ELEMENT day   (#PCDATA)>                            <!-- type: integer -->
<!ELEMENT month (#PCDATA)>       <!-- type: integer -->
<!ELEMENT year  (#PCDATA)>       <!-- type: integer -->

<!ELEMENT position (#PCDATA)>       <!-- type: integer -->
<!ELEMENT start  (#PCDATA)>                           <!-- type: integer -->
<!ELEMENT end    (#PCDATA)>       <!-- type: integer -->
<!ELEMENT length (#PCDATA)>       <!-- type: integer -->

<!ELEMENT base-id (molecule-id?, model-id?, position)>

<!ELEMENT molecule-id EMPTY>
<!ATTLIST molecule-id
          ref  IDREF  #REQUIRED>

<!ELEMENT model-id EMPTY>
<!ATTLIST model-id
          ref  IDREF  #IMPLIED>

<!ELEMENT base-pair-id EMPTY>
<!ATTLIST base-pair-id
          ref  IDREF  #REQUIRED>

<!ELEMENT alignment-id EMPTY>
<!ATTLIST alignment-id
          ref  IDREF  #REQUIRED>
```