OILing the way to Machine Understandable Bioinformatics Resources

Robert Stevens, Carole Goble, Ian Horrocks and Sean Bechhofer
Department of Computer Science
University of Manchester
Oxford Road
Manchester
UK
M13 9PL
robert.stevens@cs.man.ac.uk

March 8, 2001

Abstract

The complex questions and analyses posed by biologists, as well as the diverse data resources they develop, require the fusion of evidence from different, independently developed and heterogeneous resources. The web as an enabler for interoperability has been an excellent mechanism for data publication and transportation. Successful exchange and integration of information, however, depends on a shared language for communication (a terminology) and a shared understanding of what the data means (an ontology). Without this kind of understanding, semantic heterogeneity remains a problem for both humans and machines. One means of dealing with heterogeneity in bioinformatics resources is through terminology founded upon an ontology. Bioinformatics resources tend to be rich in human readable and understandable annotation, with each resource using its own terminology. These resources are machine readable, but not machine understandable. Ontologies have a role in increasing this machine understanding, reducing the semantic heterogeneity between resources and thus promoting the flexible and reliable interoperation of bioinformatics resources. This paper describes a solution derived from the semantic web (a machine understandable WWW), the Ontology Inference Layer (OIL), as a solution for semantic bioinformatics resources. The nature of the heterogeneity problems are presented along with a description of how metadata from domain ontologies can be used to alleviate this problem. A companion paper in this issue gives an example of the development of a bioontology using OIL.

Keywords:

Ontology; OIL; semantics; interoperation; heterogeneity; understanding.

1 Introduction

In-silico experimentation conducted over large complex, distributed experimental data sets has made biology a knowledge-based discipline conducted on a global scale. The range of data now on offer reflects the diversity of the discipline and the complexity of the tasks. Each area of molecular biology generates its own data repositories in many representation forms (literature, images, video, free text annotations), many of which are not immediately computationally accessible. This does not necessarily mean that the data are not accessible by machine, but that a machine does not have the information or ability to compute with the data. Natural language annotations are meant for human readers and computational language processing has not yet reached the stage at which the meaning of

natural language annotations can be handled computationally. Added to this problem is one of terminological differences. Even when information is readily machine accessible (such as feature tables in sequence repositories), can the computation process know when terms are equivalent?

Inconveniently, the collaboration ethos of science has not extended to a co-ordinated approach to data capture, description and publishing. The various repositories commonly have different data formats, access mechanisms and user interfaces, but worse, also have different terminologies. A plethora of specialist interrogation and analysis tools exist, each typically associated with a particular database format. Although the resources have usually been developed independently, the complex questions and analyses posed by biologists cross the artificial boundaries set by these data banks. The key tasks are integration, interoperation and fusion – linking together pieces of evidence from different information sources, different experiments, different workers; comparing results, inferring and testing new hypotheses and generating new insights. The variety of information that must be combined is expanding (e.g., genomic, proteomic, transcriptomic etc) and related in complex dynamic and not well- understood ways (e. g. metabolic pathways and signal transduction).

1.1 Bioinformatics and the World Wide Web

There is no doubt that the web has proved to be an excellent mechanism for the ready publication and availability of data. It has been an effective technology for supporting the biologists' culture of co-ordinated research, and the sharing and rapid dissemination of information. The web's greatest achievement has been a set of widely established standards that guarantee a certain level of interoperability. HTTP and HTML provide a common and straightforward infrastructure for retrieving and presenting hyper-linked documents.

There is a temptation to think that exchanging information depends on sharing a common syntax. However, information exchange needs more than syntax – even when two applications use XML as their interchange format how can we be sure that they use the same vocabulary and that the words in the vocabulary mean the same thing? What is missing is the next level of interoperability – not just making data available, but understanding what the data means.

The interoperation of information requires a consistent shared understanding of the meaning of that information. The biologist's knowledge of molecular biology and bioinformatics, and their interpretation of the resources with respect to this knowledge, is essential to the task of combining resources to answer queries and perform analyses. To automate that process requires the explicit representation and use of the knowledge that is encoded in a biologist's head or embedded in bespoke integration software. This depends on elevating the status of the information resources from machine-readable to "machine-understandable". The World Wide Web Consortium (W3C) has recognised the same need for machine understanding, rather than machine readability that has been described for bioinformatics resources. Thus, the problems faced by bioinformatics in this respect are but a reflection of those faced by the web as a whole. The idea of the Semantic Web [6] is that sites and pages on the WWW have metadata describing their content. This metadata can be used by machines to understand the nature of the content of those sites and pages. The idea of a *semantic web of bioinformatics resources* is similarly to have the information in the resources defined and linked so that its meaning is explicitly interpretable by software processes rather than just being implicitly interpretable by humans. An understanding shared between machines or people requires three things: metadata, terminologies and ontologies:

Metadata – The data describing the content and meaning of the resources, for example, mark-up in flat files, database schemas, descriptions and keywords. The schemas of bio-databases are often either implicit or not easily available, so it is difficult to determine exactly the type of conceptual information captured within specific data records.

Terminologies – However, metadata is useless unless all resources use the same language. Terminologies provide shared and common vocabularies of a domain, so search engines, agents, curators, authors and users can communicate.

Ontologies – Using a common vocabulary, however, is no good unless everyone means the same thing. For example, does the word `domain` in CATH have the same meaning as the SWISS-PROT

feature table key `domain`? Ontologies provide a shared and common understanding of a domain that can be communicated across people and applications. Many resources overlap in their content, but vary considerably on the view that is taken of that content, for example, what is meant by "gene"? A comprehensive reusable reference ontology of biological concepts or terms, is a prerequisite for information integration or interoperation [16]. Thus ontologies play a major role in supporting information exchange and discovery in e-Science and e-Commerce [11].

1.2 Bioinformatics Ontologies

There have been several attempts to develop bioinformatics ontologies to exploit biological information. Ontologies are used as repositories of potentially reusable biological knowledge, as a common framework for multi- database queries [4] or as controlled vocabularies for genome annotation [21]. The success of many biological databases depends upon their fidelity to, and the clear communication of, their ontologies to prevent errors of data entry and interpretation.

The Gene Ontology (GO) [21] is a controlled vocabulary for annotating a gene product's for molecular functions, the biological processes in which it is involved and the cellular locations in which it is found. GO takes the form of phrases organised into a hierarchical classification. EcoCyc [17] has used an ontology to specify a database schema for the E. *coli*. metabolism, signal transduction etc. RiboWeb [1] uses an ontology not only to describe its data, but also to guide its users through analysis of their data. Both EcoCyc and RiboWeb are frame-based knowledge models. TAMBIS (Transparent Access to Multiple Bioinformatics Information Sources) [3] uses an ontology to allow users to query bioinformatics databases. The TAMBIS ontology uses a description logic-based model [4].

Ontologies take a variety of forms, all of which include a vocabulary of terms and some specification of the meaning of the terms. They typically have four components:

Classification scheme – concepts are organised into a generalisation/specialisation taxonomy, whose usual meaning is that a child concept 'isa' kind of the parent concept. For example, genomic-dna isa kind of dna.

Relationships – between concepts, for example genomic-dna contained-in chromosome.

Axioms – assert additional facts about the concepts in the ontology, for example rna and dna are disjoint.

Instances – for use within class descriptions, rather than supporting the production of large existential knowledge bases. As an example, we may wish to refer to Death-receptor-3 as part of a class definition though it is not a class but an instance.

Phrase based vocabularies such as GO are simply a series of hand-crafted taxonomies that have 'intuitive' semantics for the hierarchical relationships. They have the advantage of being easily accessible, but are likely to suffer from difficulties in consistency and maintenance and are error-prone, especially in the maintenance of multiple hierarchies. Frame-based systems such as EcoCyc and RiboWeb have taxonomies and relationships with clearer semantics coupled with an easily accessible and intuitive modelling style, reminiscent of an object view of the world (a frame is a class and the slots are attributes. The frame encapsulates the properties of the instances). However, the classification scheme, like phrase based vocabularies, is essentially hand-crafted, can suffer from inconsistencies and logical mistakes and is difficult to evolve. DLs differ from both forms in that concepts are described in terms of their properties (relationships and axioms) and logical reasoning is used to automatically classify the concepts based upon those descriptions. Thus the classification scheme is automatically inferred by logical inference; moreover as the properties of a concept evolve, or new concepts are described, so the classification scheme 'morphs' to accommodate the change. The well defined semantics and logical reasoning support of DLs also allow logically consistent ontologies to be maintained - if a concept expression is logically unsatisfiable in terms of the rest of the model, the reasoning support can inform the modeller of his or her mistake. Thus Description logic based ontologies have the full complement of ontology components and avoid the problems of the hand-crafted ontologies; however, they suffer from the complexity of the modelling style.

1.3 The Ontology Inference Layer

A prerequisite for a widespread use of ontologies is a joint standard for their description and exchange. The ontology needs to be specified in some language that can comfortably cater for the variety of forms described above. The Ontology Inference Layer (OIL) [10] is a new language proposed as a knowledge representation language for the web and web-based applications. This language originated from an effort of the bio-ontology community to develop an exchange standard, as a response to the difficulties arising from the variety of knowledge representation forms[1]. OIL, couples modelling primitives commonly used in frame-based ontologies, with the simple, clean and *well-defined semantics* of an expressive Description Logic (DL). The frame primitives facilitates tool building and ontology exchange, the DL facilitates the provision of automated reasoning services, in particular consistency and subsumption checking. Thus a modeller is offered the best of both worlds in both development and deployment of an ontology.

OIL is lightweight enough to represent the ontologies such as GO, but expressive enough to present logic-based models such as the TAMBIS Ontology. OIL also has a mapping to the syntactic and semantic encoding languages of the web. Thus OIL is a potential management device for the semantic web.

This paper introduces the OIL language and how it unites the frame-based knowledge representation world with that of the logic based knowledge representation forms. A companion paper in this issue describes the building of a bio-ontology using OIL in detail.

2 The OIL Language

The development of OIL resulted from efforts to combine the best features of frame and DL based knowledge representation systems, while at the same time maximising compatibility with emerging web standards. These standards, such as RDFS, make it easier to use ontologies consistently (consistent meaning for the elements of the ontology) across the web. The intention was to design a language that was intuitive to human users, and yet provided adequate expressive power for realistic applications.

OIL has four major characteristics:

- The familiar frame like syntax and modelling primitives (derived in part from the OKBC-lite knowledge model [9]) makes OIL less daunting to ontologists/domain experts than a DL style syntax. It facilitates a modelling style in which ontologies can start out simple (in terms of their descriptive content) and are gradually extended, both as the design itself is refined and as users become more familiar with the language's advanced features. The frame paradigm also facilitates the construction and adaptation of tools, e.g., the OntoEdit and Protégé editors and the Chimaera integration tool are all being adapted to use OIL [19, 12, 18].
- The underlying mapping to an expressive Description Logic ($\mathcal{SHIQ}$) [15] provides a well defined semantics and a clear understanding of the language's formal properties. The DL gives OIL the ability and flexibility to compose classes and slots to form new expressions using boolean connectives, unlimited nesting of class elements, transitive and inverse slots, general axioms, etc. For example, we are able to define sufficiency conditions for concepts as well as necessary conditions. Moreover, we are able to use the *automated reasoning support* to automatically compute subsumption (isa) relations between concepts, or to check the consistency and coherency of the classification and its concepts. This is highly useful when collaboratively building substantial ontologies or when ontologies are reused or merged [20]. The mapping also provides a mechanism for the provision of practical reasoning services by exploiting implemented DL systems, e.g., the FaCT system [14]. This means that an ontology expressed in OIL can be reasoned over by the FaCT reasoner (see Section 2.7).
- A *machine-readable syntactic encoding in the languages of the web* including the syntactic language (XML DTD, and an XML Schema definition) and the semantic metadata languages (an

[1] See the Bio-ontology Working Group Web site at `http://smi-web.stanford.edu/projects/bio-ontology/`

RDFSchema (RDFS) definition). RDFS [7] is a proposed mechanism for deploying metadata and ontologies are a vehicle for creating well-defined metadata. OIL is defined as an extension of RDFS, thereby making OIL ontologies (partially) accessible to any "RDFS-aware" application [8]. An ontology in OIL can be used by an agent that is not OIL aware but is RDFS aware. Thus OIL is a potential management device for the general Semantic Web.

- A *layered architecture*, avoiding the temptation to throw everything into the core language, mixing up features that cannot be- with those that can be reasoned over. Thus the limits are clear and explicit.

OIL allows the definition and description of classes (concepts), slots (relationships), individuals (instances) and axioms within an ontology. The following sections describe the basic features of the OIL language. We do not have the space to give an exhaustive account of all the features and syntax here, instead see [10]. A companion paper in this issue gives further examples of the language for a bio-ontology.

2.1 Class expressions

A key component used throughout OIL is the notion of a *class expression*. A class expression represents a concept and consists of a collection of superclasses along with an optional list of slot constraints. For example, a class called hydrolase has constraints including catalyses hydrolysis, describing one of the properties of a hydrolase to be the promotion of the reaction called hydrolysis. This is similar to other frame systems. Where OIL differs, however, is that wherever a class name can appear, a recursively defined, anonymous class expression can be used, e.g., in slot constraints and in the list of superclasses. For example, a gene has-name gene-name or part-of gene-name – indicating that a gene may be found using its name or part of its name. In addition, arbitrary boolean combinations of class expression or class names (using **and**, **or** and **not**) can also appear. For example, the class (dna or messenger-rna) describes the class whose instances are all those that are instances of either the class dna or the class messenger-rna. This is in contrast to conventional frame systems, where in general, slot constraint fillers and superclasses must be class names.

As well as being able to assert individuals as slot fillers, several types of constraints on slot fillers can be asserted (these kinds of constraint are sometimes called *facets*). These include the universal quantifier *value-type* restrictions (all fillers must be of a particular class), the existential quantifier *has-value* restrictions (there must be at least one filler of a particular class), and explicit *cardinality* restrictions (e.g., at most three fillers of a given class). For instance, it is possible to exactly describe that a G-protein coupled receptor has to have seven and only seven transmembrane regions – otherwise it is not a G-protein coupled receptor. Each constraint has a clearly defined meaning, removing the confusion present in some frame systems, where, for example, it is not always clear whether the semantics of a slot-constraint should be interpreted as a universal or existential quantification – in OIL they are clearly differentiated. Slot constraints can also be defined by enumerating the individuals (instances) or data values that every instance in the defined class must be related to, for example, slot-constraint atomic-number has-filler 19.

In Figure 1 (which uses OIL's "human readable" presentation syntax, rather than the more verbose RDFS serialisation), genomic-dna is described as a dna that contained-in only chromosome or part-of chromosome. Note that the filler for the contained-in slot is a disjunction, one of whose components is an anonymous class expression (in this case, just a single slot constraint).

2.2 Class Definitions

A class definition associates a name with a specification of whether the class is *defined* or *primitive* and an optional class expression that describes the class. If defined, the class is taken to be equivalent to the given description (necessary and sufficient conditions). If primitive, the class is taken to be an explicit subclass of the given description (necessary conditions). A class can also be described by enumerating its instances, for example (one-of Iron Sulphur) defines a class whose instances are Iron and Sulphur.

slot-def part-of
  subslot-of structural-relation
  inverse has-part
  properties transitive
class-def defined genomic-dna
  subclass-of dna
  slot-constraint contained-in
    value-type chromosome *or*
      slot-constraint part-of *has-value* chromosome

Figure 1: OIL Example 1

2.3 Slot Definitions

A slot definition gives the name of the slot and allows additional properties of the slot to be asserted, i.e. *inverses*, or whether the slot has *transitive*, *symmetric* or *functional* properties. Slots also form hierarchies, so we may also specify the names of any *superslots*. In Figure 1, it is asserted that the part-of slot is transitive, and that its inverse is the slot has-part. Thus if nuclear-chromosome is part-of the nucleus, which is itself part-of the cell then the nuclear-chromosome is part-of the cell.

Domain and range restrictions on a slot can also be specified. For example, we can constrain the relationship expresses to have both domain and range gene and protein or transfer-rna, asserting that only genes can express, protein or transfer-rna. As with class descriptions, the domain and range restrictions can be arbitrary class expressions such as anonymous class expressions or boolean combinations of class names and class expressions, again extending the expressivity of traditional frame languages. All assertions made about slots are used by the reasoner, and may induce hierarchical relationships between classes.

2.4 Axioms

Another area where the expressive power of OIL exceeds that of traditional frame languages/editors is in the kinds of *axiom* that can be used to assert facts about classes and their relationships. As well as standard class definitions (which are really a restricted form of a subsumption/equivalence axiom), OIL's axioms can also be used to assert the *disjointness* or *equivalence* of classes, along with *coverings*. A covering asserts that every instance of the covered class must also be an instance of at least one of the covering classes. In addition, coverings can be said to be *disjoint*, in which case every instance of the covered class must be an instance of exactly one of the covering classes. For example, disjoint rna dna asserts that the classes dna and rna have no instances in common.

Again, these axioms are not restricted to class names, but can involve arbitrary class expressions. It is useful to state explicitly that, for instance, a chemical cannot be both an element and a compound. When building an ontology, these assertions help to reveal inconsistencies and contradictions in the conceptualisation.

2.5 Individuals

Limited functionality is provided to support the introduction and description of individuals (instances)—the intention within OIL is that such individuals are for use within class descriptions, rather than supporting the production of large existential knowledge bases (it is supposed that RDF/RDFS will be used directly for this purpose). As an example, we may wish to define the class of lard-binding-protein as being all those proteins that bind Death-receptor-3, where Death-receptor-3 is not a class but an individual.

2.6 Concrete Type expressions

Concrete data-types (string and integers), along with expressions concerning concrete data-types (such as min, max or ranges) can also be used within class descriptions. However, the FaCT reasoner does not currently support reasoning over concrete data-types, and at present OIL-based applications simply ignore concrete data-type restrictions when reasoning about ontologies. The theory underlying concrete data-types is, however, well understood [2], and work is in progress to extend the FaCT reasoner with support for concrete data-types.

2.7 Reasoning Over OIL

The reasoning services offered by a Description Logic support the development and incremental maintenance of an ontology [20]. Highly optimised implementations of sound and complete tableaux subsumption algorithms for very expressive DLs such as $\mathcal{SHIQ}$ can be used in spite of the high worst-case complexity. Thus an ontology expressed using OIL can be verified using the FaCT reasoner. The key reasoning services are:

Subsumption checking between two concept descriptions, C and D, C subsumes D, when the set of individuals that are instances of D are always a subset of the individuals that are instances of C.

Classification organises a collection of concept expressions into a partial order based on the subsumption check. This provides a lattice of definitions, ranging from the general to the specific. Composed definitions have their position implicitly determined automatically. Thus classification is a dynamic process where new compositional expressions can be added to an existing hierarchy.

Concept satisfiability checks whether a concept description can never have instances because of inconsistencies or contradictions in the model.

When verification is requested, the ontology is translated into an equivalent $\mathcal{SHIQ}$ (or $\mathcal{SHF}$) knowledge base and sent to the reasoner for classification.The classified knowledge base is queried by the OIL application, checking for inconsistent classes and implicit subsumption relationships. The results are reported to the user by highlighting inconsistent classes and rearranging the class hierarchy display to reflect any changes discovered.

When verifying the ontology, a number of new subsumption relationships may be discovered (due to the class definitions in the model). Note that if the reasoning is not employed, and if the extended expressiveness and advanced features are not used, OIL will still function as a simple frame language. The modeller, however, is not forced to use reasoning support, OIL can be used to construct hierarchies of terms unadorned by descriptions of properties.

What this means to the ontologist is that (a) there is automated support for building and evolving the classification lattice and (b) the classification scheme is coherent and consistent. For example, given the model shown in Figure 2.

The class succinate-dehydrogenase is recognised as a sub-class of mitochondrial because of its definition of having a cellular location in the mitochondrion. If we introduced the class expression (enzyme and mitochondrial), as a definition of the concept mitochondrial-enzyme then succinate-dehydrogenase would be recognised as a sub-class of that concept because of its definition.

2.8 Layers of OIL

OIL has a *layered architecture*, avoiding the temptation to throw everything into the core language, mixing up features that cannot be reasoned over with those that can be. Thus the limits are clear and explicit. For example, OIL-Core includes all the mainstream primitives and is entirely computationally viable; OIL-Standard includes instances so they can be included for knowledge base exchange but reasoning over them is not supported computationally. Currently OIL does not include default

class-def defined mitochondrial
  slot-constraint cellular-location
    has-value
      mitochondrion *or*
      slot-constraint part-of
        has-value mitochondrion

class-def defined succinate-dehydrogenase
  subclass-of enzyme
  slot-constraint promotes
    value-type oxidation
  slot-constraint cellular-location
    has-value
      slot-constraint part-of
        has-value mitochondrion

Figure 2: OIL example 2

reasoning, arbitrary axioms, or second order expressivity. The idea is that these are included as layers, carefully preserving the bounds of what can be reasoned with but without excluding features that are desirable.

3 Discussion

The issues to be addressed by the biology e-Science community are a version of issues to be addressed by the whole web community – specifically, how to move the web from one where information is machine readable by humans to one where information is machine processable by intelligent services such as information brokers, search agents and information filters. OIL has been developed as an international collaboration as part of an effort to realise the W3C vision of a semantic web, through the explicit representation of their meaning [6]. A similar ontology language called DAML has been developed as part of the DARPA Agent Markup Language initiative [13]. The OIL and DAML communities have come together in a joint language committee to produce a universal, scalable yet technically sound language for describing knowledge on the web. Thus the DAML and OIL languages have been merged under the name DAML+OIL[2].

OIL is lightweight enough to represent the simple taxonomies or frame-like ontologies such as GO without obstruction. It is also expressive enough to present logic-based models such as the TAMBIS Ontology, and all points in between. Some parts of the ontology can be simple, others complex; moreover, the language offers the ontologist an *evolutionary development path* whereby they can progressively introduce more expressive constructs as required. A companion paper in this issue describes the initial stages of building a bio-ontology by this method. Although OIL is a powerful language it isn't necessary to use all the expressive power.

An OIL ontology can be deployed in a variety of forms: From an active software component, with an API offering services to an application, to a series of static representations including XML schema and RDFS. Moreover, OIL is an effective way of exchanging and publishing ontologies without sacrificing expressivity or forcing ontologists to abandon their preferred modelling formalism.

Current OIL activity is focused on:

Example applications Manchester has already produced a version of the Gene Ontology in OIL and has developed the second generation TAMBIS ontology in OIL – a description of our experiences of the latter are described in a companion paper in this issue. Other examples of OIL ontologies can be found at `http://www.ontoknowledge.org/oil/`

[2] `http://www.daml.org`

Tools The FaCT reasoner is available with a logic sufficient to reason over OIL, an implementation efficient enough to make reasoning empirically tractable and a CORBA IDL with a clean API [5]. Other tools under development include OIL editors (OILEd at Manchester[3], OntoEdit [19] and an extension of Protégé 2000 [12], and the adaptation of ontology integration tools, such as Chimaera [18].

WWW metadata language standardisation The OIL effort has been major contributor to DARPA in their attempt to define an ontology language. In addition, the W3C have been developing an extension to RDF to include logical inferencing; OIL's RDFS mapping gives RDFS a semantics that is otherwise absent. At the time of writing, W3C have announced a Semantic Web activity where OIL takes a central role.

However, many issues are unresolved. OIL has limitations – for example it has no default reasoning or general axioms. In particular, for the bioinformatics community three major issues remain:

1. The number of bioinformatics resources is large, so solutions to describing resources should scale up;
2. Those resources are dynamic and the resources change without notification. Resources fall out of use and others emerge. The relationships between resources change as well as their relevance or popularity. Thus, as the resources change, so does their metadata;
3. Scientific knowledge is in a state of flux, and so as this changes so do the ontologies that reflect it.

Consequently, demonstrating the practical deployment of OIL in a changing and large-scale environment is our chief challenge. this paper's companion in this issue describes the early construction of an ontology that covers several bioinformatics resources.

For bioinformatics to realise the full potential of the web and its own resources requires open resources, publishing, agreeing and sharing metadata, controlled vocabularies and ontologies for defining that metadata, and fundamental developments in the underlying infrastructure of the 'Semantic Web' [6]. A prerequisite for a widespread use of ontologies is a joint standard for their description and exchange. OIL is a promising proposal. the problems of interoperating and fusing bioinformatics data are but an example of the same problems that face the development of a machine understandable web. Ontologies are gaining in popularity within the bioinformatics community as a way of describing resources and data and the consequent problems of sharing ontologies led to the birth of OIL. Although OIL has ambitions to be a universal knowledge mechanism for the web, it is pleasing to note that its original inspiration and early development was by the bioinformatics community[4].

Acknowledgements: Robert Stevens is supported by BBSRC/EPSRC grant 4/B1012090 and Sean Bechhofer is supported by EPSRC grant GR/M75426.

References

[1] R. Altman, M. Bada, X.J. Chai, M. Whirl Carillo, R.O. Chen, and N.F. Abernethy. RiboWeb: An Ontology-Based System for Collaborative Molecular Biology. *IEEE Intelligent Systems*, 14(5):68–76, 1999.

[2] F. Baader and P. Hanschke. A scheme for integrating concrete domains into concept languages. In *Proc. of IJCAI-91*, pages 452–457, 1991.

[3] P.G. Baker, A. Brass, S. Bechhofer, C. Goble, N. Paton, and R. Stevens. TAMBIS: Transparent Access to Multiple Bioinformatics Information Sources. An Overview. In *Proceedings of the Sixth International Conference on Intelligent Systems for Molecular Biology (ISMB'98)*, pages 25–34, Menlow Park, California, June 28-July 1 1998. AAAI Press.

[3] `http://img.cs.man.ac.uk/oil`

[4] See the Bio-ontology Working Group Web site at `http://smi-web.stanford.edu/projects/bio-ontology/`

[4] P.G. Baker, C.A. Goble, S. Bechhofer, N.W. Paton, R. Stevens, and A Brass. An Ontology for Bioinformatics Applications. *Bioinformatics*, 15(6):510–520, 1999.

[5] S. Bechhofer, I. Horrocks, P. F. Patel-Schneider, and S. Tessaris. A Proposal for a Description Logic Interface. In *Proceedings of DL'99, International Workshop on Description Logics*, pages 33–36, 1999.

[6] T. Berners-Lee. *Weaving the Web*. Orion Business Books, 1999.

[7] D. Brickley and V.R. Guha. Resource description framework schema specification 1.0. W3C Candidate Recommendation, 2000. `http://www.w3.org/TR/rdf-schema`.

[8] J. Broekstra, M. Klein, S. Decker, D. Fensel, F. van Harmelen, and I. Horrocks. Enabling knowledge representation on the Web by Extending RDF Schema. In *Proceedings of WWW10, the 10th World Wide Web conference, Hong Kong*, May 2001.

[9] V. K. Chaudhri *et al.* OKBC: A programmatic foundation for knowledge base interoperability. In *Proc. of AAAI-98*, 1998.

[10] D. Fensel *et al.* OIL in a nutshell. In *Proc. of EKAW-2000*, LNAI, 2000.

[11] C.A. Goble. Supporting Web based Biology with Ontologies . In *Proceedings of IEEE EMBS International Conference on Information Technology Applications in Biomedicine (ITAB 2000)*, pages 384–389, 2000.

[12] W. E. Grosso *et al.* Knowledge modeling at the millennium (the design and evolution of protégé-2000). In *Proc. of KAW99*, 1999.

[13] J. Hendler and D. L. McGuinness. The DARPA agent markup language. *IEEE Intelligent Systems*, jan 2001.

[14] I. Horrocks. Benchmark analysis with fact. In *Proc. TABLEAUX 2000*, pages 62–66, 2000.

[15] I. Horrocks, U. Sattler, and S. Tobies. Practical reasoning for expressive description logics. In *Proc. of LPAR'99*, pages 161–180, 1999.

[16] P. Karp. A strategy for database interoperation. *Journal of Computational Biology*, 2(4):573–586, 1995.

[17] P.D. Karp, M. Riley, M. Saier, I.T. Paulsen, S.M. Paley, and A. Pellegrini-Toole. The EcoCyc and MetaCyc Databases. *Nucleic Acids Research*, 28:56–59, 2000.

[18] D. L. McGuinness, R. Fikes, J. Rice, and S. Wilder. An environment for merging and testing large ontologies. In *Proc. of KR-00*, 2000.

[19] S. Staab and A. Maedche. Ontology engineering beyond the modeling of concepts and relations. In *Proc. of the ECAI'2000 Workshop on Application of Ontologies and Problem-Solving Methods*, 2000.

[20] R. Stevens, C. A. Goble, and S. Bechhofer. Ontology-based knowledge representation for bioinformatics. *Briefings in Bioinformatics*, 2001.

[21] The Gene Ontology Consortium. Gene Ontology: Tool for the Unification of Biology. *Nature Genetics*, 25:25–29, 2000.