Building a Bioinformatics Ontology Using OIL

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

August 1, 2001

Abstract

This paper describes the initial stages of building an ontology of bioinformatics and molecular biology. The conceptualisation is encoded using the Ontology Inference Layer (OIL), a knowledge representation language that combines the modelling style of Frame-Based systems with the expressiveness and reasoning power of Description Logics. This paper is the second of a pair in this special issue. The first described the core of the OIL language and the need to use ontologies to deliver semantic bioinformatics resources. In this paper, the early stages of building an ontology component of a bioinformatics resource querying applicationn are described. This ontology holds the information about molecular biology represented in bioinformatics resources and the bioinformatics tasks performed over these resources. It, therefore, represents the metadata of the resources the application can query. It also manages the terminologies used in constructing the query plans used to retrieve instances from those external resources. The methodology used in this task capitalises upon features of OIL described in the first paper of this special issue – The conceptualisation afforded by the Frame-Based view of OIL's syntax; the expressive power and reasoning of the logical formalism; and the ability to encode both hand-crafted, hierarchies of concepts, as well as defining concepts in terms of their properties, which can then be used to establish a classification and infer relationships not encoded by the ontologist. This ability forms the basis of the methodology described here: For each portion of the TaO, a basic frame-work of concepts is asserted by the ontologist. Then, the properties of these concepts are defined by the ontologist and the logic's reasoning power used to re-classify and infer further relationships. This cycle of elaboration and refinement is iterated on each portion of the ontology, until a satisfactory ontology has been created.

1 Introduction

The role of ontologies in bioinformatics has become prominent in the last few years [8, 10]. Much of biology works by applying prior knowledge to an unknown entity. The complex biological data stored in bioinformatics databases requires knowledge to specify and constrain values held in that database. Ontologies are also used as a mechanism for expressing and sharing community knowledge, to define common vocabularies (e.g., for database annotations), and to support intelligent querying over multiple databases [2, 10]. Ontologies and metadata describe the organisation and content of resources. Such definitions and descriptions are a requirement for resource interoperation and fusion.

The Ontology Inference Layer (OIL) [3, 11] can be used to develop and exchange ontologies. In this paper the development of an ontology using OIL is described. The ontology developed is part of the Transparent Access to Multiple Bioinformatics Information Sources (TAMBIS) [1] project, where it is used to enable biologists to ask questions over multiple external databases using a common query interface. The TAMBIS ontology (TaO) [2] is central to the TAMBIS system: it provides global

metadata over which queries can be formed, it drives the query formulation interface, it indexes the middleware wrappers of the component sources, and it supports the query rewriting process [4]. The concepts within the ontology are mapped to terms within the resources over which queries are formed. Thus, the TaO can be an exemplar of using an ontology to facilitate the interoperation and fusion of bioinformatics resources.

OIL is particularly effective as a development, delivery and exchange language for an ontology such as the Tao, which is complex and evolves with the current understanding of biology. Specifically, OIL inherits the best of both the frame and the description logic worlds. The frame-based modelling style and the range of epistemological constructs offered by OIL's syntax is comfortable and intuitive for most ontologists. Moreover, because ontologies are commonly encoded in a frame-based manner, OIL can be used to exchange ontologies between independent groups of ontologists. In addition, the clear semantics of the OIL language facilitate its use in a wide range of ontology environments. the ease of exchange is a crucial factor in re-use of ontologies.

The Description Logic-based reasoning support offered within OIL helps to manage the development of an ontology, through its delivery of a logically consistent ontology and the automatic management of the concept classification lattice. This expressivity of OIL can be used to capture bioinformatics and molecular biology domain knowledge with high-fidelity. However, OIL can be used to express a full range of conceptualisations, from hand-crafted hierarchies, to collections of concepts and their properties from which a classification can be inferred. Concept definitions can be as simple as possible yet as complex as necessary. For example, parts of the TaO are simple asserted trees of concepts, whereas other parts are very elaborate and exploit the full expressive power of the OIL language. This ability forms the core of the methodology described in this paper.

The TaO was originally modelled in the GRAIL DL [9]. It has been migrated to OIL in order to (a) exploit OIL's high expressivity, maintaining a better fidelity with biological knowledge as it is currently perceived; (b) use reasoning support when building and evolving complex ontologies where the knowledge is dynamic and shifting; and (c) be able to deliver and exchange the TaO as a conventional frame ontology (with all subsumptions made explicit), thus making it accessible to a wider range of (legacy) applications and collaborators. The reasoning support available with OIL can be used to infer classifications not encoded by the modeller. These can be included in the OIL delivery of the ontology. This means that the ontology can be exchanged as a static and complete data structure to third parties.

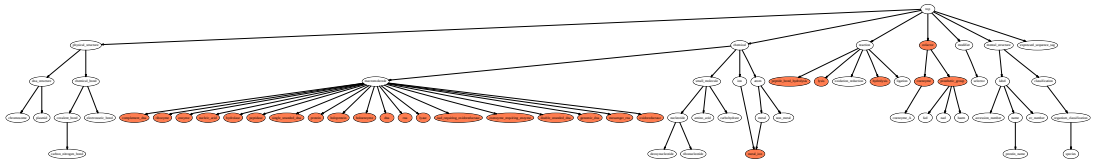
This paper is organised as follows – Section 2 describes how OIL can be used to incrementally build and manage a complex, high-fidelity ontology. Section 3 offers an extensive case study, showing how OIL has enabled this methodology to be implemented. Finally, Section 4 gives a discussion of OIL's role in developing and exchanging ontologies in bioinformatics.

2 Methodology for Ontology Development Using OIL

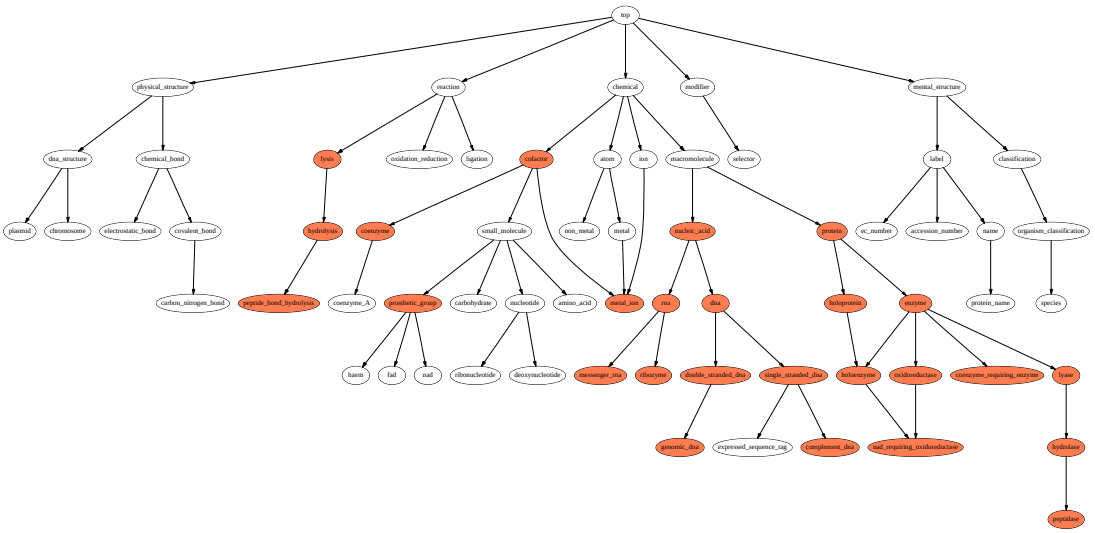
Ontology development methodologies broadly divide into those that are stage-based (e.g. TOVE [12]) and those that rely on iterative evolving prototypes (e.g. Methontology [5]). the methodology used here followed the stages of TOVE, but iterated over these cycles. In addition, there is an *elaborative* element to the methodology, that capitalises upon the reasoning power over the descriptions of concepts. Thus, the approach to developing the ontology was directly influenced by the range of expressivity that OIL affords, and the capabilities of the OIL editor OilEd [1] [6] itself, particularly its reasoning facilities. There are three principles followed in the OIL ontology development methodology:

1. Descriptive, property based modelling of concepts;
2. Incremental in situ refinement of concepts;
3. Let the reasoning take the strain of classifying concepts.

[1] OilEd is an graphical editor for the OIL language and may be found at urlhttp://www.ontoknowledge.org/oil



(a) Definitions pre-classification



(b) Definitions post-classification

Figure 1: Overview of the TaO, showing the distribution of defined and primitive concepts, together with the change in 'shape' of the ontology post-classification.

The modelling philosophy was to be descriptive, i.e., to model properties and allow as much as possible of the subsumption lattice to be inferred by the reasoner.

The refinement aspect of the design methodology was to first construct a basic framework of primitive foundation classes and slots, working both top down and bottom up, mainly using explicitly stated superclasses. This was a cyclic activity, with portions of the TaO being described primitively, then in the more descriptive fashion.

In each cycle, the reasoner is used to classify the ontology. The classification can then be viewed (with and without inferred subsumptions) to check the classification against the ontologist's knowledge. The OIL editor allows concepts to be found by name, so recently constructed concepts can be viewed in their context. Logically inconsistent concept expressions (those equivalent to bottom) are highlighted for easy identification of badly formed expressions.

The initial model was very "tree-like", i.e., there were very few classes with multiple superclasses. The primitive portions of the ontology were then incrementally extended and refined by adding new classes, elaborating slot fillers and constraints, and "upgrading" to defined classes wherever possible, so that class specifications became steadily more detailed and faithful to the application. This process was guided by subsumption reasoning — when elaborating or changing classes, the reasoner could be used to check consistency and to show the impact on the class hierarchy.

As each cycle of extension of concept definitions ends, the modeller is able to view the use of primitive and defined concepts across the ontology. This view 'zooms' out from the ontology, showing the lattice as dots and arcs, with the dots differentiated according to their being defined or primitive. This enables the modeller to see areas of high and low definition. Building-block concepts, that are not central to the use of the ontology, will in all likelyhood remain primitive, but it is useful to spot where definition is lacking; as definition increases the fidelity and justification for the ontology. For instance, the macromolecules within the TaO are highly defined, but the properties, used in the definition of more central concepts remain primitive.

Figures 1(a) and 1(b) illustrate this (using a subset of the complete ontology). Figure 1(a) shows the distribution of defined concepts throughout the hierarchy before classification[2]. Defined concepts are signified using a darker colour, and we can see that the hierarchy has a very flat structure. In Figure 1(b), we see the situation after classification. The defined concepts have now been organised into a subsumption hierarchy based on their definitions.

3 Case Study: the TAMBIS Ontology

Bioinformatics is the study and analysis of molecular biology – the functions and processes of the products of an organism's genes. The knowledge about molecular biology is contained within numerous data banks and analysis tools. The TAMBIS ontology (TaO) of bioinformatics therefore needs to support two domains: First, the domain of molecular biology – the chemicals and higher-order chemical structures, functions, processes, etc. within a cell and second, to reflect the nature and content of bioinformatics resources and the tasks performed over those resources.

The TaO covers the principal concepts of molecular biology and bioinformatics: macromolecules; their motifs, their structure, function, cellular location and the processes in which they act [2]. It is an ontology intended for retrieval purposes rather than hypothesis generation, so it is broad and shallow rather than deep and narrow [2].

Figure 2 shows a (greatly simplified) fragment of the TaO (using OIL's presentation syntax) that will be used to illustrate the methodology outlined in Section 2[3].

In building the TaO, a general domain framework was provided into which more detailed molecular biological and bioinformatics concepts could be fitted, during the phases of refinement and elaboration. As well as this approach, a solid conceptual foundation about chemicals and their structure and behaviour was built. Basic chemicals and their properties are used to describe the more

[2]The hierarchies are generated using OilEd's export functionality, which produces graphs for rendering by AT&T's Graphviz software

[3]The complete ontology can be found at `http://img.cs.man.ac.uk/stevens/tambis-oil.html`

class-def protein

class-def defined holoprotein
  subclass-of protein
  slot-constraint binds
    has-value prosthetic-group

class-def defined enzyme
  subclass-of protein
  slot-constraint catalyses
    has-value reaction

class-def defined holoenzyme
  subclass-of enzyme
    slot-constraint binds *has-value* prosthetic-group

class-def defined cofactor
  subclass-of (metal-ion *or* small-molecule)

disjoint metal-ion small-molecule

Figure 2: Simplified fragment of TAMBIS ontology

complex biological molecules of interest to bioinformatics (e.g., enzymes and their substrates), so this is an appropriate approach both from a straightforward content, as well as a modelling, point of view.

This foundation involved the description of the different kinds of chemicals (ions, atoms and molecules etc.); their structure, reactions, function and processes in which they act. This general foundation was then used to give the subsequent detailed description of the salient molecular biological concepts that form the bottom-up placement of defined concepts.

3.1 Necessary and Sufficient Conditions

The various kinds of chemical are initially described as children of the concept chemical. These primitive classes are then elaborated and defined in terms of their necessary and sufficient properties. A property is *necessary* when the concept must hold that property, but that holding that property is not enough in itself to know the type of that concept. When holding that property is enough to decide upon the kind of concept, then the property is a *sufficiency* condition [11]. Deciding upon sufficiency conditions is not an easy task, particularly in general domain concepts of biology, where exceptions to sufficency conditions can often be found [7]. The OIL editor OilEd allows easy toggling of class descriptions from *primitive* to *defined*, thus easing the process of describing properties that are necessary or both necessary and sufficiency conditions.

These elaborated, defined concepts can subsequently be used in another cycle of definition of further chemical concepts. These building-block concepts include:

atom The building block of all chemicals. An atom's behaviour is defined by the number of protons it contains, i.e., its atomic number. The atomic number is *sufficient* to describe an atom. Therefore, atom is defined as:

class-def defined atom
  subclass-of chemical
  slot-constraint atomic-number
    cardinality 1
    value-type integer
    has-value (*min* 1)

So, atoms may only have one atomic number, which must be an integer greater than or equal to

1. The concepts metal-atom, nonmetal-atom and metalloid-atom are defined to be atoms with the physicochemicalproperty of either metal nonmetal or metalloid respectively.

The concept of carbon has been defined as a kind of atom with atomic number six and the physicochemical property of non-metal. This description of the concept carbon enables it to be automatically placed as a kind of nonmetal-atom. Several other, biologically relevant, atom types have been included in the TaO.

ion An ion is simply a chemical with an electrical charge. It is defined as:

class-def defined ion
  subclass-of chemical
  slot-constraint has-charge
    has-value (*not* 0)

The slot constraint describes that an ion must have an electrical charge and it can only be an electrical charge. It also describes that the value for this charge can be a positive or negative number, but not zero. It would be possible to capture chemical reality further by specifying a minimum cardinality of one – that is, a chemical must have at least one charge to be an ion, but may have more than one charge (a molecule could, for instance, contain both a positive and negative charge).

The chemical ion has two asserted children: cation and anion. Defining cation as a chemical with charge greater-than 0 enables the classifier to place it correctly as a kind of ion. An equivalence axiom can be used to state that cation is a synonym of positive-ion.

Now, divalent-cation (a chemical with two positive charges) can be defined by adding further properties to this slot constraint: That the filler for has-charge is equal 2, that is, has positive two charges on the chemical.

element An element is a kind of chemical containing only one kind of atom. OIL has the expressive power to constrain the slot atom-type to be equal to only one. Adding the slot constraint atom-type with the value one to any chemical would cause that chemical to be classified as an element.

compound A compound is a chemical containing more than one kind of atom. The slot constraint used for element (above) is altered so that the constraint indicates that at least two kinds of atom must be present in this kind of chemical.

molecule A molecule is a kind of chemical containing atoms linked by covalent bonds. The concept covalent-bond was described as a kind of chemical-structure and used to fill the slot contains-bond, with the *has-value* restriction. So, there must be a covalent bond present for it to be classed as a molecule, but other kinds of bond may be present – exactly capturing what we understand of basic chemicals.

Two principal features of the ontology development arise from this chemical core:

1. The need for a framework of primitive concepts, such as metal and properties such as has-charge. These can be used to develop the core of defined concepts at the centre of the TaO. Primitive concepts, as well as those such as chemical itself, are placed within a simple upper level ontology containing physical, mental, substance, structure, function and process. These are extended by their obvious conjunctive forms, e.g., physical-structure. This crude upper level is simply used to organise the rest of the ontology. So, a chemical is a physical-substance, but a protein-name is a kind of mental-structure.

2. The ability to rapidly extend this chemicals core to another layer of defined chemical concepts, all of which used the previously defined concepts. This exemplifies the iterative, cyclic nature of the ontology development – a primitive frame-work is defined, classified and then used to extend the original portion of the ontology, either with further primitive, asserted concepts or by further definition.

Other descriptions include metal-ion, a kind of metal with an electrical charge. this was defined to be the conjunction of metal and ion:

class-def defined metal-ion
  subclass-of metal, ion
class-def defined divalent-cation
  subclass-of chemical
  slot-constraint has-charge
    has-value (*equal* 2)

Similarly, a concept divalent-zinc-cation can be defined as:

class-def defined divalent-zinc-cation
  subclass-of zinc
    slot-constraint has-charge
      has-value (*equal* 2)

So, the expressivity of OIL allows the properties that define what it is necessary and sufficient to be various kinds of chemical to be described, but the frame-like view allows relatively easy modelling of the knowledge.

These descriptions of chemicals can be reinforced with the use of axioms. It is not possible to be both an element and a compound, so these two concepts are described as disjoint. This means that if a concept were to be defined with properties of both an element and a compound, it would be found to be inconsistent by the reasoner during the checking phase of each development cycle. Such strict definitions help maintain the consistency and biological thoroughness of the ontology.

An organic-molecular-compound is a molecular compound that contains at least one carbon atom. This, however, is not sufficient to define an organic molecular compound. Carbon dioxide (CO_2) is a molecular compound containing carbon, but is not organic. Thus the property of containing carbon is only a *necessary* condition for being an organic molecular compound. Again, the ability to be exact with concept descriptions allows the ontology to match chemical and biological knowledge closely and prevent conceptualisations being made that contradict domain knowledge.

Bioinformatics is mainly concerned with organic macromolecular-compounds. Thus, organic molecular compound was split into the biologically useful distinctions of macromolecular-compound and small-molecular-compound. the distinction is one of size and a protein, for example, of over 100 kiloDaltons is usually said to be a macromolecule. Unfortunately the boundary is more complex, a smaller molecule can still be "macro", depending on its context. For this reason, sufficiency conditions were not used in the definition. Useful small organic molecules were simply asserted as primitive concepts underneath small-organic-molecular-compound. These include nucleotide, amino-acid and others useful in describing the properties of biological concepts.

3.2 Reasoning and Classification

OIL not only allows precise descriptions of a concept's propeties, but can use these descriptions to infer a classification. The reasoning support can automatically infer the classification from the descriptions of ion etc shown above. Divalent-cation is automatically placed as a kind of cation and divalent-zinc-cation is automatically classified as a kind of zinc and divalent-cation.

For the purposes of the TaO, macromolecular-compounds are polymers of small-organic-molecular-compounds and are defined as such. Thus, protein is defined as a polymer of amino-acid; nucleic-acid as a polymer of nucleotide and polysaccaride as a polymer of saccaride. Whenever a concept is defined to be one of these kinds of polymer, the reasoner will use this property to classify that concept correctly under the appropriate macromolecule.

A macromolecule can only be a polymer of one kind of small molecule, so the *value-type* restriction is used in the slot constraint. It is only possible to be one of these molecules, so the *disjoint* axiom is used on these macromolecules. This means that a concept defined to have the properties of more than one macromolecule will be logically inconsistent with the ontology. Thus, the reasoning support available for the OIL language helps to maintain the correctness of the ontology.

As most of bioinformatics concentrates on the analysis and description of nucleic acids and proteins, much of the TaO's description concentrates in this area. DNA and RNA are both nucleic acids formed from different kinds of nucleotide.

Describing DNA *slot-constraint* polymer-of *value-type has-value* deoxy-nucleotide, allows the classifier to correctly place it as a kind of nucleic-acid and capture that DNA can only be a polymer of the deoxy- form of a nucleotide and some of the nucleotide have to be present. Similarly, any macromolecule defined as a polymer of the ribo- form of the nucleotide is classified correctly as RNA.

The various different kinds of DNA and RNA are distinguished by their function, single- or double-stranded form and/or cellular location. Again, as before, other parts of the TaO are used to describe these properties of biological concepts. For example, genomic-dna is dna that is found on a nuclear chromosome, chloroplast chromosome, or mitochondrial chromosome. The slot constraint uses *or* in the filler class expression to describe this:

slot-constraint part-of
  cardinality 1
  value-type
    (nuclear-chromosome *or*
    mitochondrial-chromosome *or*
    chloroplast-chromosome)).

A subsequent description of chloroplast-dna as being part-of chloroplast-chromosome will enable the reasoner to classify it as being a *kind of* genomic-dna.

3.3 Incremental Refinement

The use of reasoning plays a vital role in the refinement and elaboration phases of the methodology. Concepts are progressively described by adding properties in the form of slot-constraints. the reasoner can then be used to reveal the new classification lattice inferred from those descriptions. In the initial description of kinds of protein, holoprotein, enzyme and holoenzyme were originally primitive classes, with no slot constraints, and an explicitly asserted class hierarchy: holoprotein and enzyme were subclasses of protein, and holoenzyme was a subclass of enzyme.

During the extension and refinement phase, the properties of the various classes were described in more detail: it was asserted that a holoprotein binds a prosthetic-group, that an enzyme catalyses a reaction, and that a holoenzyme binds a prosthetic-group. Several of the classes were also upgraded to being *defined* when their description constituted both necessary and sufficient conditions for class membership, e.g., a protein is a holoprotein if and only if it binds a prosthetic-group.

Enzyme was removed from the superclass list and replaced with protein; then holoenzyme's properties were described in more detail using slot constraints—in particular, it was asserted that a holoenzyme catalyses a reaction and binds a prosthetic-group. This allows the reasoner to infer not only the subclass relationship w.r.t. enzyme, but also additional subclass relationships w.r.t. holoprotein, and in particular that holoenzyme is a subclass of holoprotein. This latter relationship could have been missed if the ontology had been hand crafted.

The extension and refinement phase also included the addition of axioms asserting disjointness, equality and covering, further enhancing the accuracy of the model. Referring again to Figure 2, our biologist initially asserted that cofactor was a subclass of both metal-ion and small-molecule (a common confusion over the semantics of 'and' and 'or') rather than being either a metal-ion or a small-molecule. Subsequently, when it was asserted that metal-ion and small-molecule are disjoint, the

reasoner inferred that cofactor was logically inconsistent, and the mistake was rectified. Modelling mistakes such as these litter bioontologies crafted by hand and reasoning support can reveal such errors and thus lead to refinement of the ontology.

3.4 Slot Hierarchy

The slot hierarchy in an OIL ontology can be used by the reasoner to infer the subsumption hierarchy. For example, there are two kinds of cofactor – coenzyme and prosthetic-group. A coenzyme can be either a small molecule or metal ion and binds loosely to a protein. A prosthetic group, on the other hand, is a kind of cofactor that binds tightly to a protein, but can only be a small molecule. Again, OIL is expressive enough to capture these distinctions accurately.

class-def defined prosthetic-group
  subclass-of cofactor *and* (*not* metal-ion)
  slot-constraint binds-tightly
    has-value protein

The slot hierarchy was also used to induce the classification of types of enzyme. For example, reaction (used in the definition of enzyme) has a child lysis. Lysis is the breaking of a covalent bond and hydrolysis is breaking of a covalent bond with water. These two reactions are defined using the following slot definitions:

slot-def lysis-of
  domain reaction
  range covalent-bond
slot-def hydrolysis-of
  subslot-of lysis-of
class-def defined lysis
  subclass-of reaction
  slot-constraint lysis-of
    has-value covalent-bond
    value-type covalent-bond
class-def defined hydrolysis
  subclass-of reaction
  slot-constraint hydrolysis-of
    has-value covalent-bond
    value-type covalent-bond
class-def defined lyase
  subclass-of protein
  slot-constraint catalyses
    has-value lysis
    value-type lysis
class-def defined hydrolase
  subclass-of protein
  slot-constraint catalyses
    has-value hydrolysis
    value-type hydrolysis

A lyase is a protein that catalyses lysis. A hydrolase is a protein that catalyses hydrolysis. As the slot hierarchy describes hydrolysis-of being a subslot of lysis-of, hydrolysis is a child of lysis and consequently, hydrolase is a child of lyase. The lysis-of slot has the range covalent-bond. Various types of covalent bond can be used to further extend the conceptualisation. A peptide-bond is a kind of

carbon-nitrogen-bond, and defining peptidase to catalyse the reaction of hydrolysis of the peptide-bond automatically classifies it as a kind of hydrolase, due to the reasoning over the range of the slot definition.

A final aspect of the use of slots that can capture domain knowledge with great fidelity is the use of slot properties. A slot can be described as transitive, functional or symmetric. The slot has-publication-year has the property of being functional, which means, for instance, that an article can have only one publication year. The slot homologous-to has the property of being symmetric. This means that the inverse slot is the same as the defined slot. Thus, protein homologous-to protein, is the same as protein homologous-to protein.

The TaO contains a rich partonomy. Building the partonomy is facilitated by the assignment of the *transitive* property to the part-of slot definition. The cellular structures, in particular, use this part-of slot and its transitive property. For instance, nuclear-chromosome is part-of the nucleus, which itself is part-of the cell. Thus, a nuclear-chromosome is part-of the cell.

4 Summary

Two papers in this special issue have introduced the Ontology Inference Layer (OIL). In the first paper, the need to describe the semantics of the diverse bioinformatics resources via terminologies and metadata managed by ontologies was introduced. OIL is a language designed for this purpose – originally being designed for managing the provision of metadata for the World Wide Web. OIL unites the modelling style of Frame-Based systems, with the expressive power and reasoning support of Description Logics.

This, the second OIL paper of the special issue, has shown how these features can be used to describe a bioinformatics based ontology. The TaO acts as a metadata resource for a collection of resources, over which it executes queries. The ontology also manages the terminologies used to generate arguments in the query plans used to retrieve instances from these resources. OIL can be used to form hand-crafted ontologies as well as those using concept definitions and reasoning support to classify concepts and infer relationships not encoded by the ontologist. The reasoning support can also check the logical consistency of an ontology, in turn helping to ensure consistency with the domain being modelled.

These features were used in the methodology for re-building the TaO: A core of hand-crafted concepts are progressively refined and elaborated in terms of their properties. The reasoning support can then be used to classify the concepts, check the logical consistency and infer additional relationships. The ontology can then be checked against domain knowledge and concept definitions altered for any concepts that are not satisfiable. This cycle is iterated over new portions of the ontology, using previously defined and checked concepts to help build new concepts. Thus the frame-based view with the expressive power of OIL can help capture domain knowledge with high-fidelity – as we have aimed to demonstrate with the examples taken from the TaO. Together with the reasoning support, OIL offers a powerful ontology development forum, with the additional facility for ontology exchange.

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

References

[1] 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.

[2] 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.

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

[4] C. Goble *et al.* Transparent access to multiple bioinformatics information sources. *IBM Systems Journal*, 40(2), 2001.

[5] A. Gomez-Perez. Some Ideas and Examples to Evaluate Ontologies. Technical Report Technical Report KSL-94-65, Knowledge Systems Laboratory , Stanford, 1994.

[6] Ian Horrocks, Sean Bechhofer, Carole Goble, and Robert Stevens. OilEd: a Reason- able Ontology Editor for the Semantic Web. Submitted to IJCAI '01, Seventeenth International Joint Conference on Artificial Intelligence, 2001.

[7] D.M. Jones, P.R.S. Visser, and R.C. Paton. Addressing Biological Complexity to Enable Knowledge Sharing. In *AAAI'98 Workshop on Knowledge Sharing Across Biological and Medical Knowledge-based Systems*, 1998.

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

[9] A. Rector *et al.* The GRAIL concept modelling language for medical terminology. *Artificial Intelligence in Medicine*, 9:139–171, 1997.

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

[11] Robert Stevens, Carole Goble, Ian Horrocks, and Sean Bechhofer. OILing the way to Machine Understandable Bioinformatics Resources. Submitted to special issue of IEEE Information Technology in Biomedicine, 2001.

[12] M. Uschold and M. Gruninger. Ontologies: Principles, Methods and Applications. *Knowledge Engineering Review*, 11(2):93–113, June 1996.