Ontology-based knowledge representation for bioinformatics

Robert Stevens, Carole A. Goble and Sean Bechhofer

Date received (in revised form): 28th July 2000

Robert Stevens
is a research associate whose interests include the use of ontologies in bioinformatics and the reconciliation of heterogeneities in bioinformatics resources.

Carole Goble
is a professor of computer science at the University of Manchester. Her current interests are metadata and ontologies for information integration and intelligent retrieval.

Sean Bechhofer
is a research fellow concerned with the use of description logics to support the building and application of ontologies.

Keywords: ontology, knowledge, concept, relationship, knowledge use

Abstract

Much of biology works by applying prior knowledge ('what is known') to an unknown entity, rather than the application of a set of axioms that will elicit knowledge. In addition, the complex biological data stored in bioinformatics databases often require the addition of knowledge to specify and constrain the values held in that database. One way of capturing knowledge within bioinformatics applications and databases is the use of ontologies. An ontology is the concrete form of a conceptualisation of a community's knowledge of a domain.

This paper aims to introduce the reader to the use of ontologies within bioinformatics. A description of the type of knowledge held in an ontology will be given. The paper will be illustrated throughout with examples taken from bioinformatics and molecular biology, and a survey of current biological ontologies will be presented. From this it will be seen that the use to which the ontology is put largely determines the content of the ontology. Finally, the paper will describe the process of building an ontology, introducing the reader to the techniques and methods currently in use and the open research questions in ontology development.

INTRODUCTION

Biologists need knowledge to perform their work, often using a pre-existing item of knowledge to make inferences about the item under investigation. The most common example of this within molecular biology is the use of sequence comparison to infer the function of a novel protein sequence. The reasoning is that if a sequence of unknown function is highly similar to a sequence of known function, then it is probable that the novel sequence also has that function. So, rather than using a rule, law or equation to find the function of a protein, a biologist uses the knowledge that a similar sequence has a known function to make a judgment about the function of the new sequence. This is why it is sometimes said that biology is a 'knowledge-based', rather than an 'axiom-based' discipline.[1]

Modern biologists also need knowledge for communication. Biology is a data-rich discipline, which is available as a fund of knowledge by which biologists generate further knowledge. This knowledge is stored in thousands of databases, many of which need to be used in concert during an investigation. Knowledge is vital in two respects during this process. For instance, when using more than one data store or analysis tool, a biologist needs to be sure that knowledge within one resource can be reliably compared with another. A prime example is the differing uses of the term 'gene' within the community. In one database, gene may be defined as 'the coding region of DNA'; in another as 'DNA fragment that can be transcribed and translated into a protein' and 'DNA region of biological interest with a name and that carries a genetic trait or phenotype' in a third.[2] Being able to conform to a common definition or reason about the differences between definitions, in order to reconcile databases, would be advantageous. The second need for knowledge is to define and constrain data within a resource. Biological data can be very complex; not only in the type of data stored, but in the richness and

Robert Stevens,
Department of Computer Science,
University of Manchester,
Oxford Road, Manchester
M13 9PL, UK

E-mail:
robert.stevens@cs.man.ac.uk

constraints working upon relationships between those data. When designing a database it is useful to be able to describe what values can be specified for which attributes under which conditions. This is the encapsulation of biological knowledge within database schema.

It is impossible for one biologist to deal with all the knowledge within even one subdomain of their discipline. The arrival of whole genomes and the knowledge they contain only exacerbates the situation. There is, therefore, a need for systems that can apply the domain experts' knowledge to biological data. It is not envisaged that such systems could ever perform better than human experts; however, they could play a crucial role in helping process data to the point where human experts could again apply their knowledge sensibly. This raises numerous questions, in particular regarding how knowledge can be captured to make it available and useful within computer applications.

specification

conceptualisation

Knowledge can be captured and made available to both machines and humans by an ontology. The premise for the need for ontologies within bioinformatics is the need to make knowledge available to that community and its applications. This paper will only be a brief introduction and will not be a complete guide to the philosophy, building and use of an ontology. It does, however, aim to provide the foundations for the subject area.

The next section gives the definitions of ontology and related terms. In the third section, we will describe the uses to which ontologies can be put, and then we will describe some current bioinformatics and molecular biology ontologies and how they are used. The processes of conceptualisation and specification, or building of, an ontology are described. The final section draws together the main themes of the paper and explores the future of ontologies in the bioinformatics domain.

WHAT IS AN ONTOLOGY?

Ontology is the study or concern about what kinds of things exist – what entities or 'things' there are in the universe.[3] The computer science view of ontology is somewhat narrower, where an ontology is the working model of entities and interactions, either generically (eg the Cyc ontology[4]) or in some particular domain of knowledge or practice, such as molecular biology or bioinformatics. The following definition has been given:[5]

> An ontology may take a variety of forms, but necessarily it will include a *vocabulary of terms*, and some *specification of their meaning*. This includes definitions and an indication of how concepts are inter-related which collectively impose a structure on the domain and constrain the possible interpretations of terms.

Gruber defines an ontology as 'the specification of conceptualisations, used to help programs and humans share knowledge'.[6] The *conceptualisation* is the couching of knowledge about the world in terms of entities (things, the relationships they hold and the constraints between them). The *specification* is the concrete representation of this conceptualisation. One step in this, is the encoding of the conceptualisation in a knowledge representation language. The goal is to create an agreed vocabulary and semantic structure for exchanging information about that domain. The specification and encoding of an ontology will be explored later.

The main components of an ontology are concepts, relations, instances and axioms. A *concept* represents a set or class of entities or 'things' within a domain. `Protein` is a concept within the domain of molecular biology. Concepts fall into two kinds:

- *Primitive concepts* are those that have only necessary conditions (in terms of their properties) for membership of the class. For example, a globular protein is a kind of protein with a hydrophobic core, so all globular proteins must have a hydrophobic core, but there could be

relationship

other things that have a hydrophobic core that are not globular proteins.

- *Defined concepts* are those whose description is both necessary and sufficient for a thing to be a member of the class. For example, eukaryotic cells are kinds of cells that have a nucleus. Not only does every eukaryotic cell have a nucleus, every nucleus containing cell is eukaryotic.

Relations describe the interactions between concepts or a concept's properties. Relations also fall into two broad kinds:

taxonomy

- *Taxonomies* that organise concepts into sub-super-concept tree structures. The most common are
  - specialisation relationships commonly known as the 'is a kind of' relationship. For example, an `Enzyme` is a kind of `Protein`, which in turn is a kind of `Macromolecule`;
  - partitive relationships describe concepts that are part of other concepts – `Protein hasComponent ModificationSite`.
- *Associative* relationships that relate concepts across tree structures. Commonly found examples include:
  - nominative relationships describe the names of concepts – `Protein hasAccessionNumber AccessionNumber` (in the context of bioinformatics) and `Gene hasName GeneName`;
  - locative relationships describe the location of one concept with respect to another – `Chromosome hasSubcellularLocation Nucleus`;
  - associative relationships that represent, for example, the functions, processes a concept has or is involved in, and other properties of the concept – `Protein hasFunction Receptor`, `Protein isAssociatedWithProcess Transcription` and `Protein hasOrganismClassification Species`;
  - Many other types of relationships exist, such as 'causative' relationships, that are described in Winston *et al.*[7] and Odell.[8]

Relations, like concepts, can be organised into taxonomies. For example, hasName can be subdivided into hasGeneName, hasProteinName and hasDiseaseName. Relations also have properties that capture further knowledge about the relationships between concepts, including, but not restricted to:

- Whether it is universally necessary that a relationship must hold on a concept. For example, when describing a protein database, we might want to say that `Protein hasAccessionNumber AccessionNumber` holds universally, ie for all proteins.
- Whether a relationship can optionally hold on a concept, for example, we might want to describe that `Enzyme hascofactor Cofactor` only describes the possibility that enzymes have a cofactor, as not all enzymes do have a cofactor.
- Whether the concept a relationship links to is restricted to certain kinds of concepts. For example, `Protein hasFunction Receptor` restricts the `hasFunction` relation to link only to concepts that are kinds of receptors. `Protein hasFunction` says that `Protein` has a function but does not restrict as to what kind of concept the function might be.

- The cardinality of the relationship. For example, a particular `AccessionNumber` is the accession number of only one `Protein`, but one `Chromosome` may have many `Genes`.

- Whether the relationship is transitive, for example if `Protein isAssociatedWithProcess Transcription` and `Transcription isAssociatedWithProcess GeneExpression` then `Protein isAssociatedWithProcess GeneExpression`. The taxonomy relations always have this property.

ontology use

Once this conceptualisation is concrete (see 'Building an ontology') an ontology has been produced. *Instances* are the 'things' represented by a concept – a human cytochrome C is an instance of the concept `Protein`. Strictly, an ontology should not contain any instances, because it is supposed to be a conceptualisation of the domain. The combination of an ontology with associated instances is what is known as a *knowledge base*. However, deciding whether something is a concept of an instance is difficult, and often depends on the application.[9] For example, `Atom` is a concept and 'potassium' is an instance of that concept. It could be argued that `Potassium` is a concept representing the different instances of potassium and its isotopes, etc. This is a well-known and open question in knowledge management research.

Finally, *axioms* are used to constrain values for classes or instances. In this sense the properties of relations are kinds of axioms. Axioms also, however, include more general rules, such as nucleic acids shorter than 20 residues are oligonucleotides.

APPLICATIONS AND TYPES OF BIO-ONTOLOGIES

A common ideal for an ontology is that it should be re-usable.[6] This ambition distinguishes an ontology from a database schema, even though both are conceptualisations. For example: a database schema is intended to satisfy only one application, but an ontology could be reused in many applications. However, an ontology is only reusable when it is to be used for the same purpose for which it was developed. Not all ontologies have the same intended purpose and may have parts that are reusable and other parts that are not. They will also vary in their coverage and level of detail.

We can divide ontology use into three:

- Domain-oriented, which are either domain specific (eg *Escherichia coli*) or domain generalisations (eg gene function or ribosomes).

- Task-oriented, which are either task specific (eg annotation analysis) or task generalisations (eg problem solving).

- Generic, which capture common high level concepts, such as `Physical`, `Abstract`, `Structure` and `Substance`. This can be especially useful when trying to reuse an ontology, as it allows concepts to be correctly or more reliably placed. It can also be important when generating or analysing natural language expressions using an ontology. Generic ontologies are also known as 'upper ontologies', 'core ontologies' or 'reference ontologies'.

Most bio-ontologies have a mixture of all three types in their ontology. A well-formed ontology will be built in a modular way using a mixture of generic domain, generic task and application ontologies. Its parts will be clearly defined so that they can be reused. A less well-formed ontology will have blurred distinctions, making reuse and modification harder. The measure of how well the dependencies in an ontology have been separated is known as its *ontological commitment*. Other measures for the quality of an ontology include its

clarity, consistency, completeness and *conciseness*.[6]

application scenarios

Ontologies are used in a wide range of application scenarios:[10]

query

- A community reference – *neutral authoring*. The knowledge is authored in a single language, and converted into a different form for use in multiple target systems. Benefits include knowledge reuse, improved maintainability and long-term knowledge retention.

annotation

- Either defining database schema or defining a common vocabulary for database annotation – *ontology as specification*. Describing a protein entry as 'mitochondrial double stranded DNA binding proteins' will ensure a common vocabulary is available for description, sharing and posing questions (see item four in list). Benefits include documentation, maintenance, reliability, sharing and knowledge reuse.

- Providing common access to information. Information must be shared but is expressed using unfamiliar vocabulary. The ontology helps to render the information intelligible by providing a shared understanding of the terms or mapping between the terms. Benefits include interoperability, and more effective use and reuse of knowledge resources.

- Ontology-based search by forming queries over databases. An ontology is used for searching an information repository. For example, when searching databases for 'mitochondrial double stranded DNA binding proteins', all and only those proteins will be found, as the exact terms for searching can be used. Whether the user of the terms can be sure of their meaning depends on how the knowledge in the ontology has been represented. For example, is it explicit that the 'mitochondrial' applies to the 'DNA' or the 'binding protein'? Queries can be refined by following relationships within the ontology, for example, following relationships to find those processes in which proteins of certain functions act and gathering the associated proteins. Moving up and down the 'is a kind of' hierarchy within the ontology can also be used to refine queries; for example, specialising 'DNA binding protein' to 'single stranded DNA binding protein' by moving down the hierarchy when the former gathered too many answers. Benefits include more effective access and hence more effective use and reuse of knowledge resources.

- Understanding database annotation and technical literature. These ontologies are designed to support natural language processing (NLP) that links domain knowledge and shows how it is related to linguistic structures such as grammar and lexicons.

Although some emerging methodologies compare the structure and role of various ontologies,[11] none compare the content of one ontology with another for a specific domain.

A SURVEY OF CURRENT BIO-ONTOLOGIES

The use of ontology within bioinformatics is relatively recent and consequently there are not a huge number of ontologies in existence. In this section, a representative sample of existing bio-ontologies will be reviewed. This survey has been restricted to those ontologies most pertinent to current trends in bioinformatics and molecular biology, rather than the wider field of biology. Biology is rich in taxonomies, such as the Enzyme Classification[12] and species taxonomies. Being taxonomies, they use only a subsumption hierarchy. The ontologies reviewed here tend to be richer in their use of relationships, but this is not to denigrate the usefulness of taxonomies to many applications. The ontologies reviewed are:

- the RiboWeb ontology;[13]
- the EcoCyc ontology;[14]
- the Schulze-Kremer ontology for molecular biology;[15]
- the Gene Ontology (GO);[16]
- the TAMBIS Ontology (TaO).[17]

The content, in terms of scope, concepts and relationships, as well as the use of each ontology will be presented. In the section on 'Building an ontology', these ontologies will be revisited, as they also illustrate the variety of ontology building styles. Table 1 summarises these bio-ontologies with respect to organisation, structure, purpose and content.

The RiboWeb Ontology

RiboWeb's[18,19] primary aim is to facilitate the construction of 3D models of ribosomal components and to compare the results to existing studies. The knowledge RiboWeb uses to perform these tasks is captured in four ontologies: the physical-thing ontology; the data ontology; the publication ontology and the methods ontology. The physical-thing ontology describes ribosomal components and associated 'physical things'. It has three principal conceptualisations: `Molecules`, `Molecule-Ensembles` and `Molecule-Parts`. The first describes covalently bonded molecules and includes the main biological macromolecules. Molecule-ensembles captures non-covalently bonded collections of molecules, such as enzyme complexes.

Table 1: Summary of survey of content, structure and representation of bio-ontologies

Ontology	Application scenario	Modularised?	Domain-oriented component[a]	Task-oriented component[b]	Generic component	Instances	Detail level[c]	KR[d]
RiboWeb	Database schema	✔	Ribosome components, covalently bonded molecules, biological macromolecules, regions of molecules	Experimental detail, techniques for analysing data, publication	✔	✔	High	Frames
EcoCyc	Database schema	✔	*E. coli* genes, metabolism, regulation, signal transduction and metabolic pathways	Visualisation of biochemical reactions and layout of genes with chromosome	✔	✔	High	Frames
MBO	Community reference	✔	Shallow	Shallow	✔	✘	Low	✘
GO	Controlled vocabulary for database annotation	Partially	*Drosophila*, mouse and yeast gene function, gene product function, process and cellular location and structure	✘	✘	✔	High	✘
TaO	Common access ontology-based search	Partially	Proteins, enzymes, motifs, secondary and tertiary structure, functions and processes, subcellular structure and chemicals, including cofactors. The larger model includes nucleic acid and genes	Bioinformatics search and analysis tasks	✔	✘	High*	Description Logics (DLs)

[a]Includes domain-specific components and domain generalisation components.
[b]Includes task-specific components and task generalisation components.
[c]High* – detail is high when terms are combined to give more complex concepts.
[d]The type of knowledge representation used.
'✔' and '✘' indicate presence or absence respectively.

The molecule-part ontology holds knowledge about regions of molecules that do not exist independently, but need to be talked about by biologists. These would include amino acid side chains and the 3' and 5' ends of nucleic acid molecules. The data ontology captures knowledge about experimental detail as well as data on the structure of physical-things. The methods ontology contains information about techniques for analysing data. It holds knowledge of which techniques can be applied to which data, as well as the inputs and outputs of each method.

Instances are added to RiboWeb that correspond to these concepts. For example, a publication in a peer-reviewed article describes the three-dimensional structure of the 30s ribosomal subunit. This means linked instances need to be created in the publication, data and physical-thing ontologies. A user may want to see if this structure is consistent with others captured within RiboWeb.[19] The constraints described within RiboWeb can highlight conflicts with current knowledge to the biologist.

MBO

EcoCyc

The EcoCyc Ontology

EcoCyc, like RiboWeb, uses an ontology to describe the richness and complexity of a domain and the constraints acting within that domain, to specify a database schema.[20] EcoCyc is presented to biologists using an encyclopaedia metaphor. It covers *E. coli* genes, metabolism, regulation and signal transduction, which a biologist can explore and use to visualise information.[21] The knowledge base currently describes 4,391 *E. coli* genes, 695 enzymes encoded by a subset of these genes, 904 metabolic reactions and the organisation of these reactions into 129 metabolic pathways. EcoCyc uses the classification of gene product function from Riley[22] as part of this description. Scientists can visualise the layout of genes within the *E. coli* chromosome, or of an individual biochemical reaction, or of a complete biochemical pathway (with compound structures displayed).

EcoCyc's use of an ontology to define a database schema has the advantages of its expressivity and ability to evolve quickly to account for the rapid schema changes needed for biological information.[20] The user is not aware of this use of an ontology, except that the constraints expressed in the knowledge captured mean that the complexity of the data held is captured precisely. In EcoCyc, for example, the concept of `Gene` is represented by a concept or class with various attributes, that link through to other concepts: `Polypeptide product`, `Gene name`, `synonyms` and `identifiers` used in other databases, etc. The representation system can be used to impose constraints on those concepts and instances that may appear in the places described within the system.

The Ontology for Molecular Biology

The Ontology for Molecular Biology (MBO) is an attempt to provide clarity and communication within the molecular biology database community.[2] The use of MBO would avoid 'semantic confusion', such as that which arises with the use of the concept of `Gene` (see Introduction). Schulze-Kremer claims 'By adhering to a commonly agreeable ontology, uncertainty and misunderstanding about the semantic relations between database entries from different databases can be eliminated.' This would mean that either the different databases agreed to the common MBO definition (and changed their annotations accordingly) or inferences about the differences between each databases' conceptualisation of 'gene' could be made in terms of the MBO. In either case, attempts could then be made to reconcile or interoperate between the databases.

The MBO contains concepts and relationships that are required to describe biological objects, experimental procedures and computational aspects of molecular biology.[2] It is very wide ranging and has over 1,200 nodes

representing both concepts and instances. In the conceptual part of the MBO, the primary relationship used is the 'is a kind of' relationship. The MBO has an organising, upper-level ontology. The root concept 'Being' divides into 'object' and 'event'. 'Object', for instance, is subdivided into 'physical-' and 'abstract-' object. This helps give a precise classification for lower level concepts – so, 'physics-object' is an 'abstract object' and 'DNA' a 'physical-object'. MBO defines a linkage map from GDB in the following way: 'DBObject Mapping•Object•Map LinkageMap' (the • represents the subclass relationship).

The actual biological content of the MBO is currently relatively small, ending at quite large-grained concepts such as `Protein`, `Gene` and `Chromosome`. The framework, however, exists for extending the MBO much further into the biological domain.

Gene Ontology

The Gene Ontology

The Gene Ontology (GO),[23] like the MBO, has database annotation as its main purpose. GO, however, has grown up from within a group of databases, rather than being proposed from outside. Its scope is also narrower; instead of attempting to describe the whole of molecular biology captured in the community's databases, GO seeks to capture information about the role of gene products within an organism. The classification of gene function by Riley[24] has a similar scope, but for *E. coli* only. Initially created to reflect *Drosophila* gene function via the Flybase database,[24] GO has expanded to encompass mouse and yeast databases and is expected to expand further. Its main use is as a controlled vocabulary for conceptual annotation of gene product function, process and location in databases.

GO lacks any upper-level organising ontology. GO is essentially composed of three hierarchies, representing the function of a gene product; the process in which it takes place and cellular location and structure. GO uses the 'is a kind of', 'is located in', 'has function' and 'is involved in process' relationships to describe the role of gene products. It currently has over 5,000 concepts within the knowledge base.

GO defines a fine level of conceptual detail: double stranded DNA binding proteins; transcription factors; cytosolic chaperones; muscle motor protein; learning and memory; blood coagulation; male genital morphogenesis; ventral pattern formation; and many pathways, transport and signal transduction systems. GO uses multiple inheritance in the 'is a kind of' hierarchy in forming some of the concepts and there is some use of an 'is part of' relationship. Many of the relationships held by concepts, however, remain implicit in GO, eg the concept 'succinate (cytosol) to fumarate (mitochondrion) transporter' implicitly holds properties about location and orientation in the mitochondrial membrane, etc.

TAMBIS

The TAMBIS Ontology

TAMBIS (Transparent Access to Multiple Bioinformatics Information Sources) uses an ontology to enable biologists to ask questions over multiple external databases using a common query interface.[1] The TAMBIS Ontology (TaO)[25] describes a wide range of bioinformatics tasks and resources, and has a central role within the TAMBIS system.

An interesting difference between the TaO and some of the other ontologies reviewed here is that the TaO does not contain any instances. The TaO only contains knowledge about bioinformatics and molecular biology concepts and their relationships – the instances they represent still reside in the external databases. As concepts represent collections of instances, a concept can act as a question. The concept `Receptor Protein` represents the instances of proteins with a receptor function and gathering these instances is answering that question.

dynamic ontologies

The TaO is a dynamic ontology, – it can grow without the need for either

conceptualising or encoding new knowledge. In contrast, the other ontologies described, are static – developers must intervene and encode new conceptualisation to form new concepts. The TaO uses rules within the ontology to govern what concepts can be joined to another concept via relationships, to form new concepts.[25] Thus the TaO places great emphasis on relations. A user can form a complex, multisource query, using relationships, in the following manner. Starting with the concept `Protein`, the TaO is consulted as to which relationships can be used to join `Protein` to other concepts. Among many, the following two are offered: `ishomologous to Protein` and `hasAccessionNumber AccessionNumber`. Initially, the original `Protein` is extended to give a new concept `Protein isHomologous to Protein`; then the second 'protein' is extended with `hasAccessionNumber AccessionNumber`. The resulting concept (Protein homologue of Protein with Accession Number) describes proteins homologous to protein with a particular accession number. This concept can be used as a source-independent query containing no information on how to answer such a query. The rest of the TAMBIS system takes this conceptual query and processes it to an executable program against the external sources.[26]

The TaO is available in two forms – a small model that concentrates on proteins and a larger-scale model that includes nucleic acids. The small TaO, with 250 concepts and 60 relationships, describes proteins and enzymes, as well as their motifs, secondary and tertiary structure, functions and processes. There is also supporting material on subcellular structure and chemicals, including cofactors. Motifs extend to detail such as the principal modification sites; function and process to broad classifications such as `Hormone` and `Receptor`, and `Apoptosis` and `Lactation`; structure extends to detail such as gross architecture, eg `SevenPropellor`. Important relationships include `is component of`, `has name`, `has function` and `is homologous to`, as well as many more. The larger model, with 1,500 concepts, broadens these parts to include concepts pertinent to nucleic acid, its children and genes.

Summary

The first important message from this brief survey is that ontologies are being used within the community to provide knowledge input to databases and applications. The second is that all these ontologies are very different and specific to their intended use. TaO is an ontology of bioinformatics tasks and so contains such concepts as `AccessionNumber` and `ProteinId` , which are not part of the world of molecular biology. The TaO could not be substituted for EcoCyc's ontology. GO is an ontology of gene product function and RiboWeb represents knowledge of ribosomal subunit structure, data and methodologies. As GO is used for database annotation, it holds a fine level of detail whereas the TaO is quite shallow, but precision is gained during query formulation by joining concepts together. Even if one ontology could be developed, individual applications would use only a subset, leading to a requirement of highly modular ontologies with minimised dependencies and assumptions between them. That ontology use influences the content and nature of the knowledge captured within an ontology is not a contradiction of the knowledge-holding ability of ontologies. Not only does the purpose determine the scope and granularity to which the same knowledge is represented in different ontologies, but conceptualisations may differ without one being incorrect. For example, TaO describes that DNA may be translated to protein. This is wrong in molecular biological terms, but is a feature of bioinformatics – so conceptualisations of the same domain may differ. Sometimes a constraint is necessary for an application and sometimes it is not needed for

fitness for purpose

life cycle

another, this simply changes what knowledge is captured or how it is captured, it does not change the knowledge itself.

BUILDING AN ONTOLOGY

Although there is some collective experience in developing and using ontologies, there is no field of ontological engineering comparable to knowledge engineering. In particular, there are no standardised methodologies for building ontologies. Such a methodology would include a set of stages that occur when building ontologies, guidelines and principles to assist in the different stages, and an ontology life cycle which indicates the relationships among stages.[27] The most well-known ontology construction guidelines were developed by Gruber[6] to encourage the development of more reusable ontologies. Recently, there has been increased effort in trying to develop a comprehensive ontology methodology (eg Fernandez *et al.*,[28] Gruninger and Fox[29] and Uschold and Gruninger[27]). A survey is given in Jones *et al.*[30]

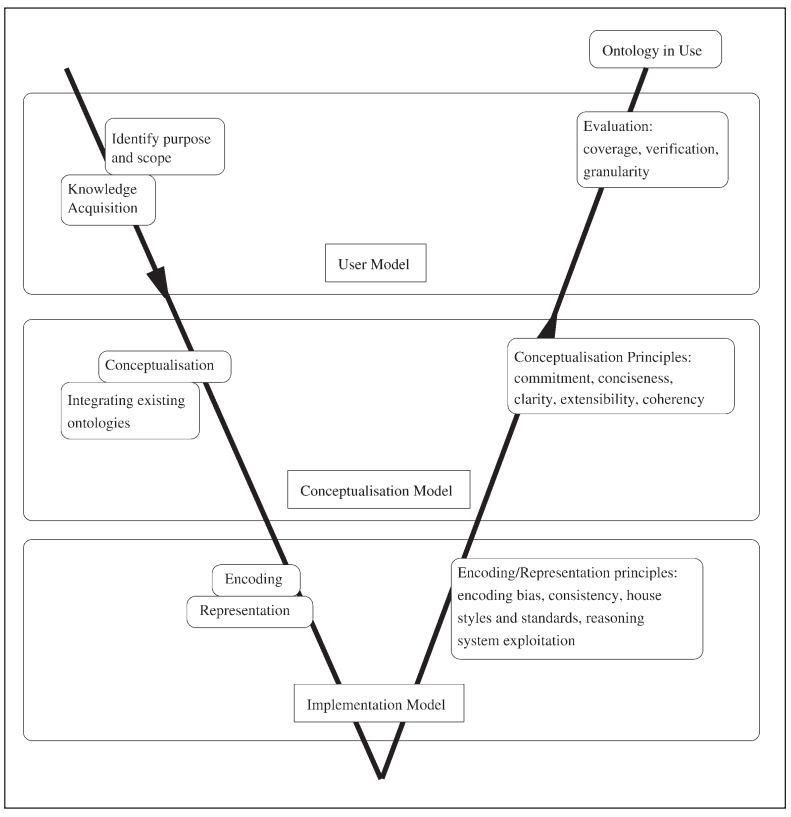

Figure 1: The V-model inspired methodology for building ontologies

The development life cycle

Methodologies broadly divide into those that are stage-based (eg TOVE[27]) and those that rely on iterative evolving prototypes (eg Methontology[31]). These are in fact complementary techniques. Most distinguish between an informal stage, where the ontology is sketched out using either natural language descriptions or some diagram technique, and a formal stage where the ontology is encoded in a formal knowledge representation language, which is machine computable. As an ontology should ideally be communicated to people and unambiguously interpreted by software, the informal representation helps the former and the formal the latter.

Figures 1 and 2 represents a skeletal methodology and life cycle for building ontologies, inspired by the software engineering V-process model.[32] The left side of the V charts the processes in building an ontology and the right side charts the guidelines, principles and evaluation used to 'quality assure' the ontology. The life cycle of the overall process is depicted in Figure 2.

The stages in the V-process model and life cycle are as follows:

- **Identify purpose and scope:** developing a requirements specification for the ontology by identifying the intended scope and purpose of the ontology. A well-characterised requirements specification is important to the design, evaluation and reuse of an ontology. It can be seen from the section on 'A survey of current bio-ontologies' that the use to which an ontology is put has a great effect on the content and style of that ontology.

- **Knowledge acquisition:** the process of acquiring domain knowledge from which the ontology will be built. Sources span the complete range of knowledge holders: specialist biologists;

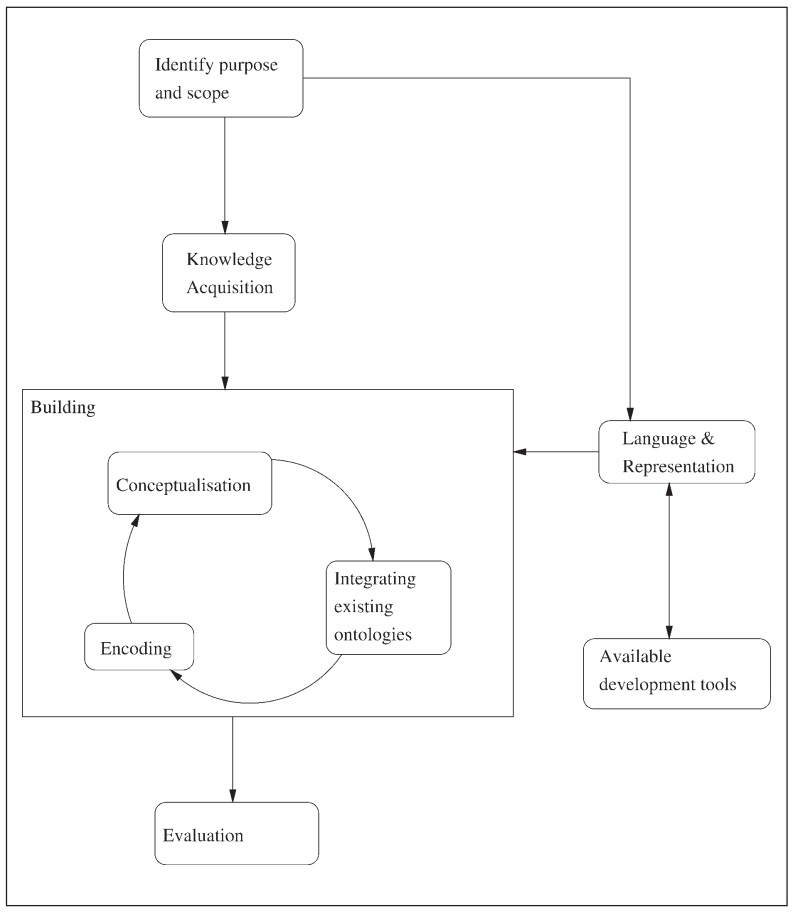

Figure 2: The ontology building life cycle

database metadata; standard textbooks; research papers; and other ontologies. Motivating scenarios are collected and *informal competency questions* formed[27] – these are informal questions that the ontology must be able to answer and will be used to check that the ontology is fit for purpose. The EcoCyc and RiboWeb ontologies had the bulk of their knowledge gathered from the research literature on *E. coli* metabolism and ribosomal structure respectively. In the former case this was a huge volume of material, which took many years to process. The TaO, being built to query databases, extracted a large part of its knowledge from database documentation. In addition, standard texts also contributed to the knowledge of core molecular biology.

- **Conceptualisation:** identifying the key concepts that exist in the domain, their properties and the relationships that hold between them; identifying natural language terms to refer to such concepts, relations and attributes; structuring domain knowledge into explicit conceptual models. This is touched upon in the section 'What is an ontology?', where the concepts and relationships describing the domain are captured. The ontology is usually described using some informal terminology. Gruber[6] suggests writing lists of the concepts to be contained within the ontology and exploring other ontologies to reuse all or part of their conceptualisations and terminologies. At this stage it is important to bear the results of the first step, that of requirements gathering, in mind.

- **Integrating:** use or specialise an existing ontology; a task frequently hindered by the inadequate documentation of existing ontologies, notably their implicit assumptions. Using a generic ontology, such as MBO, or Rector *et al.*[33] and Sowa[34] gives a deeper definition of the concepts in the chosen domain.

- **Encoding:** representing the conceptualisation in some formal language, eg frames, object models or logic. This includes the creation of formal competency questions in terms of the terminological specification language chosen (usually first order logic). The representation of ontologies is explored further below.

- **Documentation:** informal and formal complete definitions, assumptions and examples are essential to promote the appropriate use and reuse of an ontology. Documentation is important for defining, more expansively than is possible within the ontology, the exact meaning of terms within the ontology.

- **Evaluation:** determining the appropriateness of an ontology for its intended application. Evaluation is done

pragmatically, by assessing the competency of the ontology to satisfy the requirements of its application, including determining the consistency, completeness and conciseness of an ontology.[31] Conciseness implies an absence of redundancy in the definitions of an ontology and an appropriate granularity. For example, an ontology that modelled protein molecules at the atomic resolution when the amino acid level would suffice would not be considered concise.

Knowledge representation languages

For ontologies to be used within an application, the ontology must be specified, ie delivered using some concrete representation. The encoding stage, described above, is key to this specification. A variety of languages can be used for encoding or representation of conceptual models, with varying characteristics in terms of their expressiveness, ease of use and computational complexity. The field of knowledge representation (KR) has, of course, long been a focal point of research in the artificial intelligence community[35] – here we simply outline some of the KR languages that have been used for ontologies in bioinformatics (see Table 1).

Major considerations in the choice of representation are the expressivity of the encoding language, the rigour of an encoding and the semantics of a language:

expressivity

- The *expressivity* of an encoding language is a measure of the range of constructs that can be formally, flexibly, explicitly and accurately used to describe the components of an ontology as set out in the section on 'What is an ontology?'. For example, first order logic is very expressive. However, there is a trade-off between expressivity (what you can say) and complexity (whether the language is computable in real time).

- The *rigour* of an encoding is a measure of the satisfiability and consistency of the representation within the ontology. A model is satisfiable if none of the statements within contradict each other (eg an `Enzyme` is a `protein which catalyses Reaction` and `Protein which notCatalyses Reaction` is contradictory). Consistency within an ontology is a matter of encoding or conceptualising the knowledge in the same manner throughout the ontology. The rigour of an ontology's representational scheme should be maintained by the systematic enforcement of mechanisms using the ontology, which ensures the uniform and universal interpretation of the ontology. Rigour can be maintained computationally via logic-based systems or by the skill of the human encoder. Obviously, in the latter case, mistakes are more easily made and confidence in reuse of the ontology by other developers would be reduced.

semantics

- The *semantics* of a language refers to the fact that it is unambiguously what the language means. For example, the language construct 'A subconcept-of B' – does this mean that all the instances of A are also instances of B, or parts of B, or special kinds of B? Just because two languages use the same syntax does not mean they intend the same meaning. Clearly defined and well-understood semantics are essential if the ontology is to be used within the bioinformatics community for exchange of information. The definition of a general exchange language for ontologies is the subject of much current effort in the ontology research community.[36]

Languages currently used for specifying bio-ontologies fall into three kinds: vocabularies defined using natural language; object-based knowledge representation languages such as frames and UML (unified modelling language), and languages based on predicates

frames

expressed in logic such as description logics (DLs).

Vocabularies support the creation of purely hand-crafted ontologies with simple tree-like inheritance structures. The Gene Ontology, for example, has a hierarchical structure that is asserted – the position of each concept and its relation with others in the ontology is completely determined by the modeller or ontologist. Each entry or concept in the GO has a name, an identifier and other optional pieces of information such as synonyms, references to external databases and so on.

Although this provides great flexibility, the lack of any structure in the representation can lead to difficulties with maintenance or preserving consistency, and there are usually no formally defined semantics. The single inheritance provided by a tree structure (each concept has only one parent in the is-a hierarchy) can also prove limiting. Maintaining multiple inheritance hierarchies, however, is an arduous task – the hand-crafting of single inheritance hierarchies is a difficult enough exercise.

logic

A *frame-based system* provides greater structure. These systems are based on the notion of frames or classes which represent collections of instances (the concepts of the ontology). Each frame has an associated collection of slots or attributes which can be filled by values or other frames. In particular, frames can have a kind-of slot which allows the assertion of a frame taxonomy. This hierarchy can then be used for inheritance of slots, allowing a sparse representation. As well as frames representing concepts, a frame-based representation may also contain instance frames, which represent particular instances.

Frame-based systems have been used extensively in the KR world, particularly for applications in natural language processing. The most well-known frame system is Ontolingua.[37] Both EcoCyc and RiboWeb use a frame representation. EcoCyc has a frame, among others, called 'Gene', representing the concept `Gene`. This frame has slots describing relationships to other concepts, such as `Polypeptide product`, `gene name`, `synonyms` and so on. Frames are popular because frame-based modelling is similar to object-based modelling and is intuitive for many users.

The semantics of frame systems are defined by the OKBC standard,[38] although this is a little unclear in places. For example, it is not always clear how to interpret an assertion that a slot is filled with a particular value. Does this mean that all instances of the frame must have this particular attribute taking this value? Or does the value represent possible fillers for the slot for each instance? For example, we might want to say that the frame `Gene` has a slot saying 'all genes must have a `GeneName`', but it is only a possibility that `Genes` 'have a `Polypeptide Product`' (some, after all, produce `tRNAs`).

An alternative to frames is logic, notably DLs.[39,40] DLs describe knowledge in terms of concepts and relations that are used to automatically derive classification taxonomies. A major characteristic of a DL is that concepts are defined in terms of descriptions using other roles and concepts. For instance, in the TaO, the concept `Enzyme` was not simply asserted by the ontologist. Instead, a composite concept was made from `Protein` and `Reaction`, joined with the relation 'catalyses' – to make the concept `Protein which catalyses Reaction`. Thus someone viewing the ontology can see a definition for the concept `Enzyme` and the DL reasoner can automatically classify `Enzyme` as a kind of `Protein`. In this way, the model is built up from small pieces in a descriptive way, rather than through the assertion of hierarchies. The DL supplies a number of reasoning services which allow the construction of classification hierarchies and the checking of consistency of these descriptions. These reasoning services can then be made available to applications that wish to

make use of the knowledge represented in the ontology.[41]

Frames generally provide quite a rich set of language constructs but impose very restrictive constraints on how they can be combined or used to define a class. They only support the definition of primitive concepts, and the kind of taxonomy must be hand-crafted. DLs have a more limited set of language constructs, but allow primitives to be combined to create defined concepts (as described in the section on 'What is an ontology?'). The taxonomy for these defined concepts is automatically established by the logic reasoning system of the DL.

The drawback, however, is that as languages become more and more expressive, the computational complexity of reasoning increases. Recent results,[42] however, show that efficient and practical implementations of expressive languages are feasible, despite their theoretical complexity. The TaO is represented using one such DL formalism. Early implementations of TaO made use of the DL GRAIL[43] – the TaO is now represented using FaCT,[42] one of the new breed of DL implementations.

reasoning

As DLs have clear semantics, it is possible to use all of the knowledge encapsulated in the ontology to reason whether it is consistent and complete. This is not possible with simple representations such as GO – the only relationship available for exploitation is the is-a hierarchy. On the other hand, many DL implementations do not have reasoning over instances.

In fact DLs and frames are not that far apart – DLs are a logical reformulation of frames. The OIL (ontology inference layer) knowledge interchange language unifies both into one language, defined using RDF.[36] This turns out to have the simplicity of frames combined with the reasoning services of a DL.

ontology tools

Tools for ontology development

Tools are essential to aid the ontologist in constructing an ontology, and merging multiple ontologies. Such conceptual models are often complex, multidimensional graphs that are difficult to manage. The DL GRAIL has associated tools to shield the ontologist from the logical formalism. An intermediate 'template' form is used to represent the conceptualisation, from which the encoding can be generated.[44] These tools also usually contain mechanisms for visualising and checking the resulting model – over and above the logical means for checking the satisfiability of the specified model. The MBO also has an editor for creating and visualising the object-based encoding used in that ontology.[2] The frame-based system used by EcoCyc also has the GKB editor for handling the conceptualisation and encoding in frame-based representations.[45] Such tools are really essential for maintaining complex ontologies that are necessary for capturing knowledge within the biology domain. Other tools support the collaborative development of ontologies over the web (eg WebOnto[46]). A survey of tools can be found in Duineveld *et al*.[47]

DISCUSSION

This briefing has introduced the need and use of ontology within the bioinformatics community. The need for ontologies arises from the need to be able to cope with the size and complexity of biological knowledge and data. Ontologies enable knowledge to be used within systems for communication, specification and other processing tasks (see the section on 'Applications and types of bio-ontologies').

Several bio-ontologies have already been used within the community. Those reviewed in 'A survey of current bio-ontologies' demonstrate a wide range of scopes and granularities. Most have common core features of molecular biology, such as `Gene`, `Protein` and related `biologicalFunction` and `BiologicalProcess`, but differ widely in both the content and articulation of their knowledge. This is primarily due to

the wide range of tasks to which the ontologies are put. Both RiboWeb and EcoCyc use part of their ontology to define the structure and content of their databases, but as the databases are as different as ribosomal subunit structure and *E. coli* metabolism, the ontologies are also necessarily different. Even the common areas, such as macromolecule, differ widely between ontologies, but without any of the ontologies being incorrect.

Bio-ontologies are currently being used for communication of knowledge, as well as database schema definition, query formulation and annotation. When the use of conceptual annotation grows we can expect to see a concomitant change in database retrieval. This will become much more precise and complete than is currently possible with natural language-based annotations. Annotation by ontologies should also allow the relationships describing functions, process and components, etc, of retrieved entries to be explored with ease.

There are a number of open issues to be addressed in the use of ontology within the bioinformatics community:

- **Knowledge-based reasoning.** This briefing started with a description of how biology research is often driven by the use of knowledge, especially by determination of function by sequence similarity. Only RiboWeb, of the ontologies described, approaches this kind of use. It can be expected that the use of ontology to assist in analysis will grow further. This will be made easier by the conceptual annotation of the primary databases – a collection of similar sequences returned by a search could be clustered within an ontology of protein function and features. Such clustering should be able to help with the analysis of similarity search results and other bioinformatics analyses.

- **Reuse *v.* specific.** Currently there is little reuse of bio-ontologies – this is partly because of difficulties in the diversity of their representational form, the explicitness of their semantics and the range of applications they address. OIL moves us further forward to a common representational language. As the number of bio-ontologies increases, it will be interesting to see whether there is a growth in the reuse of ontology. The use of ontology in annotation could drive this process, as well as that of ontology in analysis. An open issue in ontology reuse is the evolution of the source ontology once it has been reused in another ontology. If the original ontology changes, should the changes be reflected where it is reused and how would this evolution be managed?

- **Tools and libraries.** The frame-based Protégé ontology development tool[48] is currently being adapted to represent ontologies in OIL, so that we can build and deliver frame-based ontologies while gaining from the reasoning services offered by a DL. This may be less important with small local ontologies designed by one expert, but becomes important for large, collaboratively developed ontologies that are intended to be reused and shared. Libraries of ontologies, such as those held by WebOnto and Ontolingua, must be developed if reuse is to be promoted.

- **Methodologies for constructing ontologies.** The process of building an ontology, as described in the section on 'Knowledge representation languages', is a high-cost process. The reality is that the construction of ontologies is an art rather than a science. Methodologies (supported by tools) are essential to: help the developer spot a concept; to modularise their ontologies; to avoid problems such as over-elaboration (when should I stop elaborating the ontology); to ensure relevance (when is a concept relevant for an application?) and to verify the ontology for its fitness of purpose and its reusability (if any).

If the application genuinely needs an ontology and that ontology will be long lived, then the investment may well be worthwhile. Like many technologies, in a discipline such as bioinformatics, it is the community effort that is important in making the use of that technology productive.

Acknowledgements
Robert Stevens is supported by a grant from the BBSRC/EPSRC under the bioinformatics initiative (34/BIO12090); Sean Bechhofer is supported by a grant from the EPSRC under the DIM initiative (GR/M/75426).

References

1. Baker, P. G., Brass, A., Bechhofer, S. *et al.* (1998), 'TAMBIS: Transparent access to multiple bioinformatics information sources. An overview', in 'Proceedings of the 6th International Conference on Intelligent Systems for Molecular Biology', 28 June–1 July, AAAI Press, Menlo Park, CA, pp. 25–34.
2. Schulze-Kremer, S. (1998), 'Ontologies for molecular biology', in 'Proceedings of the Third Pacific Symposium on Biocomputing', AAAI Press, Menlo Park, CA, pp. 693–704.
3. Blackburn, S. (1996), 'The Oxford Dictionary of Philosophy', Oxford University Press, Oxford.
4. Lenat, D. B. (1995), 'Cyc: A large-scale investment in knowledge infrastructure', *Commun. ACM*, Vol. 38(11), pp. 32–38.
5. Uschold, M., King, M., Moralee, S. and Zorgios, Y. (1998), 'The enterprise ontology', *Knowledge Eng. Rev.*, Vol. 13(1) (special issue on 'Putting ontologies to use'), pp. 31–89.
6. Gruber, T. R. (1993), 'Towards principles for the design of ontologies used for knowledge sharing', in Guarino, R. P. N., Ed., 'International Workshop on Formal Ontology, Padova, Italy, 1993'. Available as technical report KSL-93-04, Knowledge Systems Laboratory, Stanford University. ftp.ksl.ftanford.edu/pub/KSL_Reports/KSL-983-04.ps
7. Winston, M., Chaffin, R. and Herrmann, D. (1987), 'A taxonomy of part–whole relations', *Cognitive Sci.*, Vol. 11, pp. 417–444.
8. Odell, J. J. (1998), 'Six Different Kinds of Aggregation', Cambridge University Press, Cambridge, pp. 139–149.
9. Brachman, R. J., McGuinness, D. L., Patel-Schneider, P. F. *et al.* (1991), 'Living with Classic: When and how to use a KL-ONE-like language', in Sowa, J., Ed., 'Principles of Semantic Networks: Explorations in the Representation of Knowledge', Morgan Kaufmann, pp. 401–456.
10. Jasper, R. and Uschold, M. (1999), 'A framework for understanding and classifying ontology applications', in 'Twelfth Workshop on Knowledge Acquisition Modeling and Management KAW'99'. Published online.
11. Guarino, N. and Welty, C. (2000), 'Identity, unity, and individuality: Towards a formal toolkit for ontological analysis', in Horn, W., Ed., 'Proceedings of ECAI-2000: The European Conference on Artificial Intelligence', Aug., Amsterdam IOS Press.
12. International Union of Biochemistry (1984), 'Enzyme Nomenclature 1984: Recommendations of the Nomenclature Committee of the International Union of Biochemistry on the Nomenclature and Classification of Enzyme-catalyzed Reactions', Academic Press (for the International Union of Biochemistry), Orlando, Florida.
13. http://smi-web.stanford.edu/projects/helix/riboweb.html
14. http://ecocyc.PangeaSystems.com/ecocyc/ecocyc.html
15. http://igd.rz-berlin.mpg.de/~www/oe/mbo.html
16. http://genome-www.stanford.edu/GO/
17. http://img.cs.man.ac.uk/tambis
18. Chen, R. O., Felciano, R. and Altman, R. B. (1997), 'RiboWeb: Linking structural computations to a knowledge base of published experimental data', in 'Proceedings of the 5th International Conference on Intelligent Systems for Molecular Biology', AAAI Press, Menlo Park, CA, pp. 84–87.
19. Altman, R., Bada, M., Chai, M. *et al.* (1999), 'RiboWeb: An ontology-based system for collaborative molecular biology', *IEEE Intelligent Systems*, Vol. 14(5), pp. 68–76.
20. Karp, P. and Paley, S. (1996), 'Integrated access to metabolic and genomic data', *J. Comput. Biol.*, Vol. 3(1), pp. 191–212.
21. Karp, P., Riley, M., Paley, S. *et al.* (1999), 'EcoCyc: Electronic encyclopedia of *E. coli* genes and metabolism', *Nucleic Acids Res.*, Vol. 27(1), pp. 55–58.
22. Riley, M. (1993), 'Functions of the gene products of *Escherichia coli*', *Microbiol. Rev.*, Vol. 57, pp. 862–952.
23. The Gene Ontology Consortium (2000), 'Gene ontology: Tool for the unification of biology', *Nature Genetics*, Vol. 25, pp. 25–29.

24. The FlyBase Consortium (1999), 'The FlyBase database of *Drosophila* Genome Projects and Community Literature', *Nucleic Acids Res.*, Vol. 27(1), pp. 85–88. http://flybase.bio.indiana.edu/

25. Baker, P. G., Goble, C. A., Bechhofer, S. *et al.* (1999), 'An ontology for bioinformatics applications', *Bioinformatics*, Vol. 15(6), pp. 510–520.

26. Paton, N. W., Stevens, R. D., Baker, P. G. *et al.* (1999), 'Query processing in the TAMBIS Bioinformatics Source Integration System', in Ozsoyoglo, Z. M. *et al.*, Eds, 'Proceedings of 11th International Conference on Scientific and Statistical Database Management (SSDBM)', IEEE Press, pp. 138–147.

27. Uschold, M. and Gruninger, M. (1996), 'Ontologies: Principles, methods and applications', *Knowledge Eng. Rev.*, Vol. 11(2).

28. Fernandez, M., Gomez-Perez, A. and Juristo, N. (1997), 'METHODONTOLOGY: From ontological art to ontological engineering', in 'Spring Symposium Series', pp. 33–40.

29. Gruninger, M. and Fox, M. S. (1995), 'Methodology for the design and evaluation of ontologies', in 'IJCAI Workshop on Basic Ontological Issues in Knowledge Sharing', published online.

30. Jones, D. M., Bench-Capon, T. J. M. and Visser, P. R. S. (1998), 'Methodologies for ontology development', in 'Proc. ITi and KNOWS Conference of the 15th IFIP World Computer Congress', Chapman-Hall, pp. 62–75.

31. Gomez-Perez, A. (1994), 'Some ideas and examples to evaluate ontologies', Technical Report KSL-94-65, Knowledge Systems Laboratory, Stanford.

32. Ould, M. A. (1990), 'Strategies for Software Engineering: The Management of Risk and Quality', John Wiley, Chichester.

33. Rector, A. L., Rogers, J. E. and Pole, P. (1996), 'The Galen high level ontology', *Studies Health Technol. Informatics*, Vol. 34, pp. 174–178.

34. Sowa, J. (1995), 'Top-level ontological categories', *Int. J. Human–Computer Studies*, Vol. 43(5/6), pp. 669–686.

35. Duce, D. A. and Ringland, G. A. (1988), 'Approaches to Knowledge Representation: An Introduction', John Wiley, Chichester.

36. Horrocks, I., Fensel, D., Broekstra, J. *et al.*, 'The ontology interchange language oil: The grease between ontologies'. http://www.cs.vu.nl/~dieter/oil

37. Farquhar, A., Fikes, R. and Rice, J. P. (1997). 'The ontolingua server: A tool for collaborative ontology construction', *J. Human–Computer Studies*, Vol. 46, pp. 707–728.

38. Chaudhri, V. K., Farquhar, A., Fikes, R. *et al.* (1998), 'OKBC: A programmatic foundation for knowledge base interoperability', in 'Proceedings of 15th National Conference on At (AAAI-98) and the 10th Conference on Innovative Applications of AI (IAAI-98)', AAAI Press, Menlo Park, CA, pp. 600–607.

39. Borgida, A. (1995), 'Description logics in data management', *IEEE Trans. Knowledge Data Eng.*, Vol. 7(5), pp. 671–782.

40. Woods, W. A. and Schmolze, J. G. (1992), 'The KL-ONE family', *Computers Math. Applic.*, Vol. 23(2–5), pp. 133–177.

41. Bechhofer, S. and Goble, C. A. (1999), 'Delivering terminological services', *AI*IA Notizie, Periodico dell'Associazione Italiana per l'intelligenza Artificiale*, Vol. 12(1), March.

42. Horrocks, I. (1998), 'Using an expressive description logic: FaCT or Fiction?' in Cohn, A. G., Schubert, L. K. and Shapiro, S. C., Eds, 'Principles of Knowledge Representation and Reasoning: Proceedings of the Sixth International Conference (KR'98)', Morgan Kaufmann Publishers, San Francisco, CA.

43. Rector, A. L., Bechhofer, S. K., Goble, C. A. *et al.* (1996), 'The GRAIL concept modelling language for medical terminology', *Artificial Intelligence Med.*, Vol. 9, pp. 139–171.

44. Rogers, J. E., Solomon, W. D., Rector, A. L. *et al.* (1997), 'Rubrics to dissections to GRAIL to classifications', in 'Medical Informatics Europe '97', Vol. 43, IOS Press, pp. 241–245.

45. Paley, S. M., Lowrance, J. D. and Karp, P. D. (1997), 'A generic knowledge-base browser and editor', in 'Proceedings of the 1997 National Conference on Artificial Intelligence'.

46. Domingue, J. (1998), 'Tadzebao and WebOnto: Discussing, browsing, and editing ontologies on the Web', in '11th Knowledge Acquisition for Knowledge-Based Systems Workshop'.

47. Duineveld, A. J., Stoter, R., Weiden, M. R. *et al.* (1999), 'Wondertools? A comparative study of ontological engineering tools', in 'Twelfth Workshop on Knowledge Acquisition, Modeling and Management'.

48. Grosso, W. E., Eriksson, H., Fergerson, R. W. *et al.* (1999), 'Knowledge modeling at the millennium (the design and evolution of Protégé-2000)', Technical Report SMI-1999-0801, Stanford Medical Informatics (SMI), Stanford University School of Medicine, 1999. http://www-smi.stanford.edu/pubs/SMI_Reports/SMI-1999-0801.pdf