GOHSE: Ontology Driven Linking of Biology Resources

S. K. Bechhofer R. D. Stevens P. W. Lord

School of Computer Science
Kilburn Building
University of Manchester
Oxford Road
Manchester, M13 9PL
UK
`http://cohse.man.ac.uk`

Abstract

We describe the GOHSE system, an application to support browsing of biology resources. The **C**onceptual **O**pen **H**ypermedia **Se**rvice (COHSE) system enhances web resources through the dynamic addition of hypertext links. These links are derived through the use of an ontology and associated lexicon along with a mapping from concepts to possible link targets. GOHSE applies COHSE to Bioinformatics, using the **G**ene **O**ntology (GO) as an ontology and associated keyword mappings and GO associations as link targets. The resulting demonstrator provides both glossary functionality and the possibility of building knowledge based hypertext structures linking bioinformatics resources.

Key words: Open Hypermedia, Bioinformatics, Browsing, Ontologies

1 Introduction

As a discipline, bioinformatics relies on the knowledge held within its documents (Web pages, database entries, books or articles). Biologists were early adopters of the Web and continue to use it as the primary means of delivering data, tools and knowledge to their community. Query by navigation, via links between these documents and others is still fundamental to practical bioinformatics. It is the links between biology documents that provide the utility to

Email address: `sean.bechhofer@manchester.ac.uk` (S. K. Bechhofer).

both humans and machines. Common usage of the Web involves embedding links within documents. There are, however, a number of limitations to this approach:

Hard Coding: Links are hand-crafted and hard coded in the HTML encoding of a page. An anchor is placed around the source object in the originating page and the location of the end-point is included in the link. This end-point, or target, of a link can be a page, an anchor placed within a page, or perhaps some dynamically evoked service such as a query. The link is a static, inflexible entity that is intimately bound with the source node.

Format Restrictions: Documents need to be written in a particular format (e.g. HTML or PDF) in order to support the addition of links.

Ownership: Ownership of the page is required in order to place an anchor in a page. It is, of course, possible to point to targets on other pages without ownership, but in order to insert a link source anchor, ownership is required.

Legacy resources: It can be difficult to deal with legacy material – when the view of a world changes, old pages might need to be updated with new links.

Maintenance: There is a weight of maintenance in creating and updating links in pages. This is due in part to the hard coding and ownership issues described above.

Link targets: Current Web links are restricted to point to point linking; there is only one target. Web links are essentially unary with no explicit inverse link (although browsers offer a "back" button that will take the user back to the originating point). Binary or n-ary links would allow greater flexibility in linking by offering more choice of targets for each link.

Conceptual Open Hypermedia supports the construction of hypertext link structures built using information encoded in ontologies. In this paper we describe the use of an ontology driven open hypermedia system within bioinformatics. Using this technology it is possible to separate links from document resources and consequently provide a rich, dynamically linked collection of biology oriented documents. By driving the dynamic formation of links through an ontology we are taking advantage of the common understanding provided by an ontology.

Dynamic linking services, supported by ontologies, offer a mechanism to help overcome such restrictions. The Conceptual Open Hypermedia Service (COHSE) system [1] enhances document resources through the dynamic addition of hypertext links. These links are derived through the use of an ontology and associated lexicon along with a mapping from concepts to possible link targets. We describe an application of the COHSE architecture to Bioinformatics, using the Gene Ontology (GO) [2] as an ontology and GO associations as link targets. The resulting demonstrator (referred to here as GOHSE) provides both glossary functionality and the possibility of building dynamic hypertext

structures linking bioinformatics documents. Ontology driven dynamic linking offers a vision of biology documents dynamically linked to multiple resources based on a common understanding of a domain based upon ontologies.

The following scenario describes the added value that the COHSE system can provide. A biologist is reading a Web page about cellular structure [1]. When viewed using a traditional browser, she will see static links (as inserted by the author) contained within the page. She then employs the COHSE agent (making use of the cellular component of GO) to assist browsing. Now, as well as the static links contained within the page, the lexical items within the page corresponding to GO cellular component terms or their synonyms are also highlighted as link sources – for example, the term "cytochrome c oxidase", which is a synonym of the GO term respiratory chain complex IV (sensu Eukarya) [GO:0005751]). Next to this highlighted term is an icon indicating that a definition is available. She clicks on this icon and sees a pop up definition of the term, taken from the GO. Also shown are a number of further link targets, taken from a link base, offering her a range of resources related to the term, such as `COX3_YEAST` in the SwissProt [3] database. In addition, if the number of resources found for the term is below a threshold, the agent will use the taxonomic structure of the GO cellular component ontology to find more general or more specific terms that may provide appropriate resources. Clicking upon `COX3_YEAST`, she is taken to the appropriate UniProt/SWISS-PROT entry. This resource can itself then be dynamically linked to terms etc. within the GO cellular component ontology.

In the following sections, we discuss how this scenario is realised. In Section 2 we describe the Gene Ontology and how it makes a suitable resource for the open, dynamic linking of biology document resources. We then introduce the COHSE system in Sections 3 and 4 and describe the combination with GO in Section 5. Section 6 discusses related work and places GOHSE in context. We conclude with discussion and pointers to future work.

2 GO

The Gene Ontology (GO) [2] is a collaborative effort to address the need for consistent descriptions of the major attributes of gene products in different databases [2]. Figure 1 shows a portion of the cellular component ontology from GO. Each term has an associated textual definition describing the term along with subsumption and partitive relationships with other terms in the ontology.

[1] e.g. `http://users.rcn.com/jkimball.ma.ultranet/BiologyPages/C/CellularRespiration.html`

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

Each term within GO also has associated synonyms that represent alternative, equally valid terms for the concept within the ontology. In addition, a number of mappings between other vocabularies or classification systems and GO are available [3] .

```
respiratory chain complex IV (sensu Eukarya)
 Accession: GO:0005751
 Aspect: cellular_component
 Synonyms:
          o cytochrome c oxidase
          o GO:0005752
 Definition:
          o A part of the respiratory chain, containing the 13
          polypeptide subunits of cytochrome c oxidase, including
          cytochrome a and cytochrome a3. Catalyzes the oxidation of
          reduced cytochrome c by dioxygen (O2). Found in eukaryotes.
 hierarchy
 * GO:0003673 : Gene_Ontology ( 146200 )
   o GO:0005575 : cellular_component ( 79199 )
     + GO:0005623 : cell ( 56534 )
       # GO:0005622 : intracellular ( 46101 )
       * GO:0005737 : cytoplasm ( 35977 )
         o GO:0005739 : mitochondrion ( 12311 )
          + GO:0005740 : mitochondrial membrane ( 979 )
            # GO:0005743 : mitochondrial inner membrane ( 775 )
             * GO:0005746 : mitochondrial electron transport chain ( 211 )
               o GO:0005751 : respiratory chain complex IV (sensu Eukarya) ( 54 )
       * GO:0045277 : respiratory chain complex IV ( 55 )
         o GO:0005751 : respiratory chain complex IV (sensu Eukarya) ( 54 )
      # GO:0016020 : membrane ( 13431 )
       * GO:0019866 : inner membrane ( 803 )
         o GO:0005743 : mitochondrial inner membrane ( 775 )
          + GO:0005746 : mitochondrial electron transport chain ( 211 )
            # GO:0005751 : respiratory chain complex IV (sensu Eukarya) ( 54 )
       * GO:0005740 : mitochondrial membrane ( 979 )
         o GO:0005743 : mitochondrial inner membrane ( 775 )
          + GO:0005746 : mitochondrial electron transport chain ( 211 )
            # GO:0005751 : respiratory chain complex IV (sensu Eukarya) ( 54 )
```

Fig. 1. GO Ontology fragment

Collaborating databases annotate their gene products with appropriate GO terms, providing the consistency of annotation needed for reliable querying of databases. All the entries from the 16 collaborating databases either contain GO identifiers (that map to GO terms) or their equivalent mappings to internal database keyword lists or GO synonyms. In addition, other Web pages about biology, articles in on-line databases such as Pubmed, etc. also use these GO terms, synonyms and keyword mappings. Finally, there are tools, such as the GO Amigo browser, that are also oriented towards the Gene Ontology and offer further mechanism for linking via the GO terminologies.

GO offers a huge resource of community knowledge, but no one organisation owns all the "documents" that use the GO vocabularies and definitions. The use of COHSE together with GO offers a mechanism by which any Web page or other document containing lexical items matching a GO term or any of its equivalents can be automatically linked to not only a definition of the term from GO, but also a legion of other resources, based upon GO. As a consequence, we can dynamically generate a rich, flexible web of biology resources.

[3] See `http://www.geneontology.org/GO.indices.html`

3 COHSE

Detailed descriptions of COHSE[4] can be found elsewhere [1], but we give here a brief overview of the basic approach and architecture. COHSE draws primarily on previous work in Open and Conceptual Hypermedia.

An influential early categorisation of hypertext [4] noted that links were either *extensional* (explicitly authored and stored e.g. to implement an intended navigational ordering) or *intensional* (not explicitly stored as links but derived in some manner from the content). The study (which predates the Web) recommended an increase in the support for intensional linking, based on the requirements for managing the complex interconnectedness of a scholarly literature (particularly for biblical or literary studies).

Subsequent systems (e.g. Hyperbase [5], Hyper-G [6], Devise Hypermedia [7], Microcosm [8], WWW [9]) and standards (Dexter [10], HyTime [11], HTML [12], XLink [13]) which addressed the concept of linked documents focused on links which were authored and stored (either as part of, or separately from the documents). A significant amount of effort over the subsequent years has been directed in the standards committees at the thorny issues of how to model and express extensional links in a reasonably robust and universal fashion. A well-rehearsed problem with extensional links in the context of the Web is how to store and manipulate links as first-class objects, independently of documents, while maintaining referential integrity in an uncontrolled authoring environment [14].

Although intensional links avoid this problem, they depend on an understanding of the document content. If this is not achieved by the "author", then there must be some computational entity (be it a subroutine, DLL, Web Service or Intelligent Agent) capable of (i) recognising a potential link anchor and (ii) working out what to link it to. This recognition software can be a part of the system (and configured to "know" the appropriate requirements of the application) or independent of the system which must perform a retro-fit of the suggested links. In the Web, an example of the former case is a script on a company's Web server which links the names of that company's products to their Web pages. An example of the latter case is a general language service which annotates technical terms with explanatory notes; only those explanations relevant to the specific user should be maintained ([4] suggests a dictionary or part-of speech linker that would affect every word in the document).

The Microcosm system [8] (and its Web-based successor DLS [15]) implemented linking semantics which were a combination of extensional and intensional linking. An entry in a linkbase (extensional) declared that occurrences

[4] `http://cohse.semanticweb.org`

of a particular term may be linked to a particular destination but devolved the responsibility of recognising where the term occurred and under what circumstances the link should be activated to the runtime browsing environment (intensional). This split paradigm relied on both a skilled link author (to make appropriate generic links to lead into their documents and out to other resources) and on a skilled hypertext integrator (to partition the various links appropriately between modular sets of linkbases which can be automatically activated for the right users when they visit the right pages). Experience shows that failure in the authoring task results in a familiar paucity of links whereas failure in the system administrator's task results in users' being overwhelmed by inappropriate selections of links. We refer to this second case as the "prolific but ignorant linker".

The problems of extensional linking are well-known and widely experienced: the significant authoring effort required to create and maintain links may not be forthcoming, even in an environment such as scientific research where longevity of access to publications is a key requirement [16]. In order to decrease the reliance on authoring effort in a linking environment, we need to employ more sophisticated content recognition software. We want to be able to recognise the kinds of things that are mentioned in the text, and understand the relationship between those things and other things which we may have links to, which may feature in glossaries, dictionaries or subject-specific portals.

The promise of ontology-driven linking is the potential to use an agreed common collection of significant concepts, expressed as an agreed vocabulary in a given natural language, modelled together with agreed inter-relationships. In fact, the key objective is to reuse a model which has already been constructed for other knowledge management purposes. (In other words, to get improved linking functionality "for free".) Not only could the lexicons of the ontology identify the natural language terms which signify a significant concept, but the explicit relationships (and carefully quantified semantics) could help to overcome the problem of the 'prolific but ignorant linker' by providing an understanding of what can be linked, to dovetail in with the linking application's in-built knowledge of what kinds of things should be linked (subject key terms, foreign terms, named entities, general concepts) to achieve a particular goal (glossary explanation, tutorial support, related information). In tandem with explicit annotation data, this provides a rich framework for building linked structures.

The Distributed Link Service (DLS) [15], developed by the University of Southampton is a service that adopts an open hypermedia approach, and provides dynamic linking of documents. Links are taken from a link base, and can be either *specific*, where the source of the link is given by addressing a particular fragment of a resource, or *generic*, where the source is given by

some selection, e.g. a word or phrase. Documents and linkbases are dynamically brought together by the DLS, which then adds appropriate links to documents.

COHSE extends the DLS with *ontological services*, providing information relating to an ontology. These services include mappings between concepts and lexical labels (synonyms). For example, GO tells us that cytochrome c oxidase is a synonym of respiratory chain complex IV (sensu Eukarya) (See Figure 1). In this way, the terms and their synonyms in the ontology form the means by which the DLS finds lexical items within a document from which links can be made (providing *generic* links). The services also provide information about relationships, such as sub- and super-classes – here respiratory chain complex IV (sensu Eukarya) is a sub-class of respiratory chain complex IV. The use of an ontology helps to bridge gaps [17] between the terms used in example web pages (e.g. in this case cytochrome c oxidase), and those used to index other bioinformatics resources, such as the more specialised or generalised GO terms. We can loosen the restriction of linking only to Eukaryote complexes, and consider also linking to all complexes.

COHSE thus extends the notion of generic linking – the key point being that the ontology provides the link service with more opportunities for identifying link sources. As the ontology contains the terms that inform the DLS about the lexical items that may become links, there is no longer a need to own the page in order to make the link from the source to the target – this is taken care of by the DLS. Furthermore, the effort in providing the source of links moves from the document author to the creator(s) of the ontologies that are used by COHSE. In this case, these are the creators of GO – an ontology supported by a wide community of biologists and database curators that form a consensus in the ontological representation of domain understanding.

The system is implemented as a *COHSE agent*, along with two supporting services: the *Ontology Service* (OS) and *Resource Manager* (RM). The agent augments documents with links based on the semantic content of those documents. The Ontology Service delivers ontological information (as introduced above) in a dynamic fashion [18] to the DLS. The Resource Manager associates concepts with resources and provides mechanisms for querying those associations. Figure 2 shows a simplified view of the basic architecture of the system.

The agent first contacts the OS to obtain a collection of relevant lexical items. As documents are processed, the agent then looks for these items. Any that are found in the documents provide potential link sources. For each source, a link is then added that includes:

(1) The concept to which the lexical item resolves (in the case of GOHSE

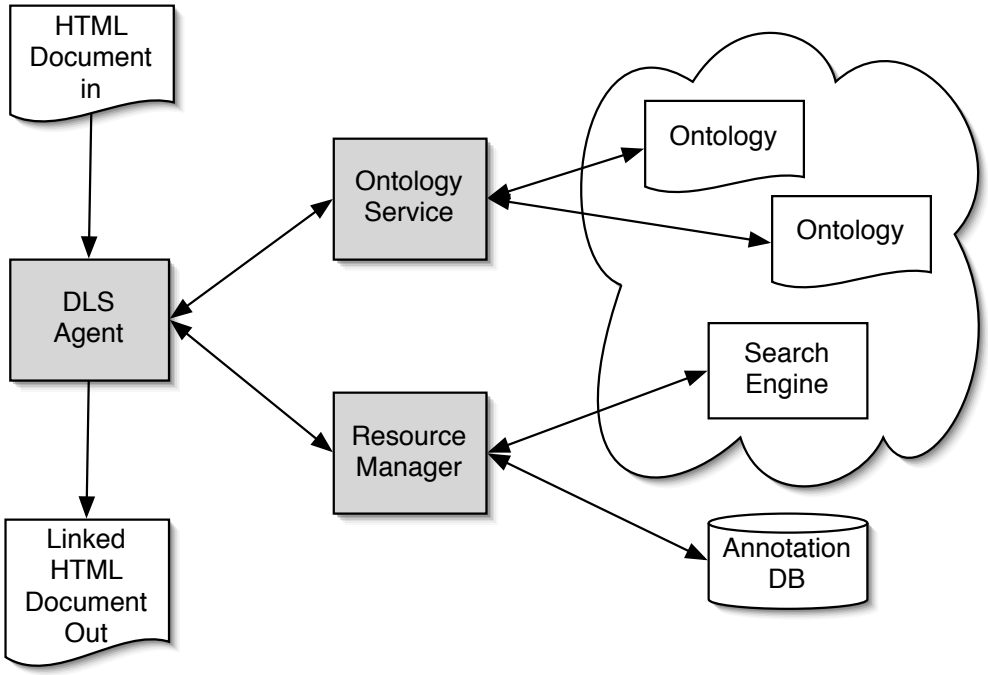

Fig. 2. COHSE Logical Architecture

this will be the GO term).
(2) A description of the term.
(3) A collection of link targets associated with that term.

Items 1 and 2 are supplied by the OS. The targets in 3 are supplied via calls to the RM. The concepts in the ontology are used to determine appropriate targets for links out of the given document. Within COHSE, the RM plays two roles. It maps resources or documents to concepts (via explicit annotations that have been made), and maps concepts to documents. It is this latter functionality that allows us to provide potential link targets once a link source has been found. For the concept to document mapping, we can either rely on a reversal of the explicit document to concept mapping, or provide targets through the use of external resources. For example, the RM provides potential link targets through queries to the GO database. Once a link source and associated GO term has been identified, we can query the GO database for proteins that have been annotated with that term, in the UniProt/SWISS-PROT database. Given the identifiers for those proteins, we can then produce URLs allowing browsing of those proteins via UniProt's web front end. In addition, we can provide links to the AmiGO or MGI GO browsers. Alternative mechanisms that we have explored for target retrieval include using external search engines (such as Google or the Amazon catalogue) with the query being based on keywords associated with a concept.

Central to the COHSE agent is the provision of an *editorial component* within the agent. This component uses information within the ontology (such as

hierarchical classification) in order to either determine whether the generated links are suitable or to expand or cull the set of possible targets. The procedure used to make this decision is currently rather simple. The user sets a threshold for the number of links that they would like to be presented for each concept. If the number of links returned by the Resource Manager is less than this threshold, then the agent will query the Ontology Service to retrieve links for broader or narrower terms.

4 Implementation

Figure 2 shows the logical architecture of the COHSE systems. The COHSE Agent takes documents/pages in and produces pages enhanced with links. To do this it relies on services providing information about ontologies and resources. Both OS and RM are presented to the COHSE agent using simple CGI interfaces. This allows the system to make use of existing protocols and results in a relatively lightweight, loosely-coupled, and open architecture.

Different physical architectures are possible. In particular, we can choose to provide the agent either at the server side, as a proxy or at the client side. Each of these approaches has pros and cons.

Server side Here, the agent is incorporated into the server, and rewrites or processes documents before they are dispatched to the client. The advantages here are that this places no requirement on the client side (other than that of having a standard web browser). It is also possible to batch process sites before delivery – in fact this approach can be seen primarily as support for hypertext authoring. The downside here is that only one site is processed, and that site is necessarily in our control.

Proxy In a proxy solution, the agent is provided through a web proxy, sitting between the server and client. Again, this provides the advantage that there are no specific requirements made on the client side in terms of software or installation. In addition, third party pages can now be processed. Disadvantages here are that the proxy implementation needs to be able to deal with parsing of the documents – in the context of HTML on the Web, this can prove to be a non-trivial task as many pages are ill-formatted and do not provide "valid" HTML. Tools such as Tidy[5] can be a help here. Additionally, the use of a proxy can introduce processing delays – the entire document may need to be processed before delivery to the client. Introducing long delays will impact on user satisfaction.

Client Side Finally, we can consider providing the functionality at the client side, through a browser plugin. This has the significant disadvantage of

[5] `http://tidy.sourceforge.net/`

requiring specific browsing software – plugins need to be written for a particular platform (e.g. Mozilla or Internet Explorer). On the plus side, client side plugins are relatively easy to prototype, can support customization and are able to use the parsing infrastructure and document handling provided by the browser implementation. Delayed processing (adding links once the page is loaded and displayed) is also possible.

We have experimented with both client side and proxy implementations of the system[6]. – as yet server side is unimplemented. The GOHSE demo described here was provided using a proxy. This choice was made primarily to make it easier to demonstrate to interested users. Use of a client side plugin requires the user to download and install software on their machine, and restricts us to a single platform (in our case, Mozilla). With a proxy implementation, (almost) any browser on (almost) any platform can be used – this is a particularly desirable characteristic when trying to fit within a community delivering information using existing Web infrastructure (such as the bioinformatics community). With a simple proxy, the user has to make adjustments to the set up of their browser, however, which may dissuade the casual user from trying the system. In addition, if the user is behind a firewall that requires the use of a proxy, this can cause problems.

To overcome these difficulties, we made use of a forwarding or rewriting proxy. The forwarding proxy takes a URL, retrieves the contents of that URL via the COHSE proxy, and then returns the results to the user. Rather than having to adjust browser settings, the user simply appends the URL of their required page to the URL of the forwarding proxy. The forwarding proxy ensures that URLs referred to in links in the retrieved page are themselves rewritten to pass through the forwarding proxy again. Note that the forwarding proxy is not specific to COHSE or GOHSE. Figure 3 shows the interactions between the client, server, proxies and services.

A Browser sends request of the form
`http://gohsehost/proxy/http://whatever/page.html`
to the forwarding proxy.
B Forwarding proxy sends request for
`http://whatever/page.html`
to COHSE proxy.
C COHSE proxy sends request for
`http://whatever/page.html`
to the Web server. An HTML page is returned.
D The page is processed by the proxy. Links are added where appropriate and the amended page is returned to the forwarding proxy.
E Forwarding proxy rewrites any links in the resulting page. Targets of the

[6] See `http://cohse.man.ac.uk/` for details of available software.

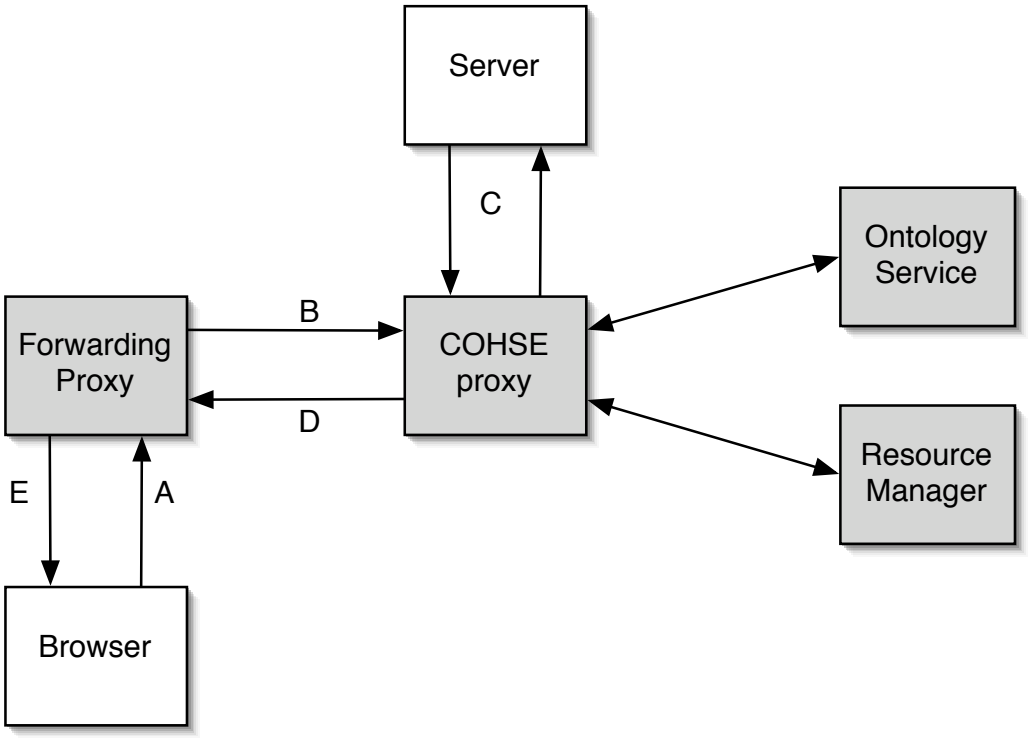

Fig. 3. Physical Architecture

form
`http://somewhere/otherpage.html`
are rewritten to
`http://gohsehost/proxy/http://somewhere/otherpage.html`
and the resulting page returned to the browser.

5 Wrapping GO

For the purposes of the GOHSE demonstration, the cellular component hierarchy of GO provides the ontology, while link targets are derived from GO annotations. The GO terms and annotations are supplied as a single (mySQL) database containing a number of tables. Some hold information about the taxonomic structure of the Gene Ontology and the language terms attached to GO identifiers. Others contain information describing annotation of proteins in external databases with GO terms. In order to expose the information in a suitable format for our system, we have provided a simple wrapper around the database that exports the GO taxonomy (and associated lexical mappings) as an OWL [19] ontology – the format expected by the Ontology Service.

Each GO term is translated to an OWL class (identified by its GO identifier), and the GO hierarchy links are translated to rdfs:subClassOf properties.

Descriptions of terms are attached to classes using rdfs:comment annotation properties and the terms themselves are associated using rdf:label properties.

We are aware that this naive translation of the GO taxonomic structure may not be an ideal treatment of GO [20], but for the purposes of this demonstrator, it suffices.

The OWL translation is loaded into the COHSE OS. Resource retrieval in the RM is implemented by returning UniProt/SWISS-PROT GO associations, taken from the GO database. For any given GO term, the RM returns URIs providing access to a number of potential targets:

- The AmiGO browser focused on the term.
- The Gene Ontology Browser focused on the term
- Any UniProt/SWISS-PROT entries known to be annotated with the term.

The AmiGo and Gene Ontology browser URIs are "hard-wired" into the system – we know for any GO term what the URI will be [7] The UniProt/SWISS-PROT entries are obtained through queries on GO annotations – again this is achieved through a wrapping of the GO database. In this way, we achieve a loose coupling of the demonstrator with GO – changes in the underlying database will not require any re-engineering of the application. It is also relatively easy for us to plugin alternative mechanisms for target selection (such as a bespoke or generic search engine).

6 Related Work

A closely related application is Magpie [21,22]. Magpie is "intended to support interpretation and information gathering" [22], rather than the hypertext linking ambitions of COHSE, although there are clearly overlaps between the approaches. When pages are loaded, the Magpie plugin attempts to detect the occurrence of entities within the page. As with COHSE (see Section 3) the lexical items to look for are generated from the ontology. When entities are found, they are highlighted, and the user is offered services pertaining to the identified entity. Magpie is implemented as a browser plugin (see Section 4 for a discussion of architectural choices).

AmiGO [8] provides a web-based browser for GO terms and their associations, exposing the content of the GO database. Users can search for particular GO terms (for example, those that include the word "oxidase". Once located and

[7] For example, the AmiGo browser focused on term GO:0005829 is given by a URI: `http://www.godatabase.org/cgi-bin/amigo/go.cgi?query=0005829`.
[8] `http://godatabase.org/`

selected, the term and its place in the GO hierarchy is shown. Annotations using that term can then also be displayed, via a query to the GO database. This is offering similar functionality to the retrieval provided by GOHSE. However, AmiGO's emphasis is on supporting focused browsing for particular terms and associations. GOHSE is not in itself a search engine, but is a mechanism for presentation of information that may be derived from search engines. Thus in GOHSE, the terms to display are driven by the content of a document being browsed. In addition, although the GOHSE targets are currently being retrieved through queries to the GO database (as in AmiGO), this is a loose coupling, and alternative mechanisms for finding targets can easily be added, as could alternative ontologies.

COHSE uses simple lexical matching in order to find link sources. There are other systems that analyse web pages in order to perform more sophisticated Named Entity Recognition – for example MnM [23] or KIM [24]. These are intended to support the task of knowledge base population rather than annotation or linking of pages. Although COHSE makes use of language technology, this is not a key aspect of our research. Third party systems may be used to identify the potential link sources, but we can see this as a external service that is provided for COHSE to use (see Section 7).

7 Discussion

We have described an application of the COHSE infrastructure to support ontology driven browsing of biology document resources on the Web. In particular, the *dynamic* nature of the linking process helps to alleviate some of the problems with traditional Web linking, which can be static, restricted and inflexible.

Linking is based upon a conceptual model provided by an ontology, where the definitions and structure of the ontology, together with the lexical labels drive the consistency of link provision and dynamic aspects of the linking. The Gene Ontology has already been developed and encapsulates a great deal of shared knowledge about important concepts in the domain. Although this was not necessarily the purpose for which GO was designed, by using GO within this application, we are able to access a wealth of domain knowledge and gain added value "for free". The ontological resource that drives the linking of documents has already been created and its existence independent of the documents it links means that linking is consistent between documents (for a given version of the ontology).

The use of GO in this context illustrates a key benefits of the Semantic Web approach: a computationally amenable representation of the content and facil-

ities of documents and services [25]. GO provides an encoding of some domain knowledge (concept synonyms and taxonomy) in a *machine processable* fashion. By making this information available to applications, we are able to use this to support the presentation and browsing of resources. In many cases, this benefit may be of an unanticipated nature – the Gene Ontology was not built in order to assist in the production of hypertext links, but nonetheless it has proved useful in that context. The use of a standard representation language such as OWL facilitates this reuse – although there is much focus within the Semantic Web community on ontology languages with rich expressiveness, the importance of standardization should not be over-looked. For a number of use cases (such as here, where we are currently making most use of basic taxonomic information along with lexical mappings) it is the *existence* of an agreed language and format that helps, rather than the choice of language operators. Note also that the approach used in COHSE is generic – we are not bound to the use of the Gene Ontology, but could use any other ontology appropriate for the domain, for example MGED [26].

We note that the separation of the maintenance of link data from the underlying content enables us to manage the task of updating databases (or at least their web representations) as new knowledge becomes available. For example, while UniProt/SWISS-PROT links directly to the Gene Ontology, its predecessor, SWISS-PROT, did not. Using GOHSE, we can synthesize these links before the underlying data source provides them. Similarly, by extending GOHSE to recognise UniProt/SWISS-PROT identifiers, we can link between free text resources, such as PubMed, GO concepts, and the underlying protein data sources. This feature of open hypermedia systems in general, and GOHSE in particular, is of particular relevance to biological data where cross-linking is known to be fragile [27]. As a discipline, bioinformatics relies on access to knowledge held in its databases. A system such as COHSE, augmented by ontologies such as GO, provide a knowledge model to drive the linking of diverse, distributed resources according to that knowledge.

It is clear that one of the reasons that this approach works here is because we have a well-defined domain of interest, a community and (to a certain extent) agreement on the important terms and concepts within that community. This is where we believe Semantic Web technology will have its initial successes – within well-defined communities.

In the current implementation, the identification of potential link sources is done in a rather naive fashion – effectively through a straight lexical match on the names of terms provided by GO (plus some synonyms provided by additional keyword mappings). In the GOHSE setting, this produces reasonable results (from the technical point of view), largely because (as discussed above), GO tries to use terms commonly used in the domain and these provide a clear set of lexical items to act as potential link sources. We can, however,

encounter problems when, for example, formatting information is included in the source, making the identification of lexical items harder. This is clearly an area where the use of Human Language Technology may bring benefits. We are investigating the use of the GATE [28] framework in order to gain access to more effective text processing components. These will provide the DLS greater flexibility in its use of the terms provided by the OS.

Once concepts have been identified within the page, navigation of the ontology is driven by the COHSE Agent rather than the user. The agent decides if sufficient link targets are available, and whether or not to traverse the hierarchy to obtain more candidates (see Section 3. It may be more profitable to allow the user to explicitly navigate or explore the ontology at this point, rather than relying on the agent. However, there are then questions as to how one exposes the ontological structure to the user. In the demonstrator described here, we are not making explicit use of other relationships in the ontology (such as partonomy which is represented in GO).

In a similar vein, COHSE is clearly a system in which *personalization* can play a part – different users will want to use different ontologies or annotation collections. The current architecture is rather inflexible in this respect, and we are investigating support for more effective personalization. Although the use of the forwarding proxy makes it relatively easy to use the system and makes little demand on users (no download or installation is necessary and browser settings need not be altered), the forwarding proxy makes it hard for us to support basic customization – for example allowing users to select the ontologies to use for term recognition or the sources for target selection. To that end, we are investigating a reimplementation of the basic system as a portlet [9]. Portlets allow delivery of applications via a portal – the portal is then responsible for maintaining user account information such as preferences.

Finally, we make the observation that GOHSE does not attempt to provide us with any *new* knowledge – it simply allows us to organise and present what is already there. Nor is it, at present, a particularly sophisticated implementation and improvements can certainly be made. It does, however, allow us to link together diverse biology resources, including those not in our control, in a consistent fashion based upon a community understanding of the domain.

[9] Most likely making use of the JSR168 Recommendation: `http://jcp.org/aboutJava/communityprocess/final/jsr168/`

Acknowledgments

Phil Lord was supported by the myGrid EPSRC E-science pilot (EPSRC GR/R67743). The original COHSE system was developed in collaboration with the University of Southampton and enhancements have made with support from Sun Microsystems Ltd. The authors would like to thank John Kimball for permission to use his pages in our examples.

References

[1] L. Carr, S. Bechhofer, C. Goble, W. Hall, Conceptual Linking: Ontology-based Open Hypermedia., in: WWW10, Tenth World Wide Web Conference, 2001.

[2] The Gene Ontology Consortium, Gene Ontology: a tool for the unification of biology, Nature Genetics 25 (2000) 25–29.

[3] SWISS-PROT Annotated Protein Sequence Database., `http://www.expasy.org`.

[4] S. DeRose, Expanding the Notion of Links., in: Proceedings of the Second Annual ACM Conference on Hypertext, 1989, pp. 249–257.

[5] J. Schnase, J. Leggett, D. Hicks, P. Nürnberg, J. A. Sánchez, Design and Implementation of the HB1 Hyperbase Management System, Electronic Publishing: Origination, Dissemination and Design 6 (2) (1993) 35–63.

[6] K. Andrews, F. Kappe, H. Maurer, Hyper-G: Towards the Next Generation of Network Information Technology., Journal of Universal Computer Science.

[7] K. Grønbæk, J. A. Hem, O. L. Madsen, L. Sloth, Designing Dexter-based cooperative hypermedia systems., in: Proceedings of ACM Hypertext 93, 1993, pp. 25–38.

[8] H. Davis, I. H. W. Hall, G. Hill, R. Wilkins, Towards an Integrated Information Environment with Open Hypermedia Systems, in: ECHT '92, Proceedings of the Fourth ACM Conference on Hypertext, ACM Press, Milan, Italy, 1992, pp. 181–190.

[9] T. Berners-Lee, R. Cailliau, J.-F. Groff., The World-Wide Web., Computer Networks and ISDN Systems 25 (4-5) (1992) 454–459.

[10] F. Halasz, M. Schwartz, The Dexter Hypertext Reference Model, in: Proceedings of the Hypertext Standardization Workshop by National Institute of Science and Technology (NIST), 1990, reprinted in Communications of ACM, Vol. 37, No. 2, 30-39, February 1994.

[11] International Standards Organisation, Hypermedia/time-based structuring language (hytime), ISO/IEC Standard 10744 (1992).

[12] D. Ragget, A. Le Hors, I. Jacobs, Html 4.01 specification, W3C Recommendation (24 December 1999).

[13] S. DeRose, E. Maler, D. Orchard, Xml linking language (xlink) version 1.0, W3C Recommendation (27 June 2001).

[14] H. Davis, Referential integrity of links in open hypermedia systems., in: Proceedings of the Ninth ACM Conference on Hypertext and Hypermedia, Hypertext '98, Pittsburgh, Pennsylvania, 1998, pp. 207–216.

[15] L. Carr, D. D. Roure, W. Hall, , G. Hill, The Distributed Link Service: A Tool for Publishers, Authors and Readers., World Wide Web Journal 1 (1) (1995) 647–656.

[16] S. Lawrence, M. David, G. Pennock, R. Flake, F. Krovetz, E. Coetzee, A. Finn, A. Kruger, C. Giles, Persistence of Web References in Scientific Research, IEE Computer 34 (2) (2001) 26–31.

[17] M. J. Bates, Indexing and Access for Digital Libraries and the Internet: Human, Database and Domain Factors, JASIS 49 (13) (1998) 1185–1205.

[18] S. Bechhofer, C. Goble, Delivering Terminological Services, AI*IA Notizie, Periodico dell'Associazione Italiana per l'intelligenza Artificiale. 12 (1).

[19] D. L. McGuinness, F. van Harmelen, OWL Web Ontology Language Overview, W3C Recommendation, World Wide Web Consortium, `http://www.w3.org/TR/owl-features/` (2004).
URL `\url{http://www.w3.org/TR/owl-features/}`

[20] R. Stevens, S. Bechhofer, U. Sattler, P. Lord, Reconciling the Semantics of DAG and OWL Ontology Representations, in: Workshop on The Formal Architecture of the Gene Ontology, Leipzig, 2004.

[21] M. Dzbor, J. Domingue, E. Motta, Magpie – Towards a Semantic Web Browser, in: Fensel et al. [29].

[22] M. Dzbor, J. Domingue, E. Motta, Opening Up Magpie via Semantic Services, in: Proceedings of WWW2004, Thirteenth International ACM World Wide Web Conference, 2004.

[23] M. Vargas-Vera, E. Motta, J. Domingue, M. Lanzoni, A. Stutt, F. Ciravegna, MnM: Ontology Driven Semi-Automatic and Automatic Support for Semantic Markup, in: Proceedings of the 13th International Conference on Knowledge Engineering and Management (EKAW 2002), Springer Verlag, 2002.

[24] B. Popov, A. Kiryakov, A. Kirilov, D. Manov, D. Ognyanoff, M. Goranov, KIM – Semantic Annotation Platform, in: Fensel et al. [29], pp. 834–849.

[25] J. Hendler, Science and The Semantic Web, Science (2003) 24.

[26] C. Stoeckert, H. Parkinson, The MGED Ontology: A framework for describing functional genomics experiments, Comparative and Functional Genomics 4 (1) (2002) 127–132.

[27] J. D. Wren, 404 not found: the stability and persistence of URLs published in MEDLINE, Bioinformatics 20 (5) (2004) 668–672.
URL http://bioinformatics.oupjournals.org/cgi/content/abstr%act/20/5/668

[28] H. Cunningham, D. Maynard, K. Bontchev, V. Tablan, GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications., in: Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02), Philadelphia, 2002.

[29] D. Fensel, K. Sycara, J. Mylopoulos (Eds.), Proceedings of the 2nd International Semantic Web Conference, ISWC2003, Vol. 2870 of Lecture Notes in Computer Science, Springer, 2003.