The Semantics of Semantic Annotation

Sean Bechhofer[1], Leslie Carr[2], Carole Goble[1], Simon Kampa[2], and Timothy Miles-Board[2]

[1] Information Management Group
Department of Computer Science, Kilburn Building
University of Manchester, Oxford Road, Manchester M13 9PL
http://img.cs.man.ac.uk
seanb@cs.man.ac.uk
[2] Intelligence, Agents, Multimedia Group
Department of Electronics & Computer Science
University of Southampton, Highfield, Southampton SO17 1BJ
http://www.iam.ecs.soton.ac.uk

Abstract. Semantic metadata will play a significant role in the provision of the Semantic Web. Agents will need metadata that describes the content of resources in order to perform operations, such as retrieval, over those resources. In addition, if rich semantic metadata is supplied, those agents can then employ reasoning over the metadata, enhancing their processing power. Key to this approach is the provision of annotations, both through automatic and human means. The semantics of these annotations, however, in terms of the mechanisms through which they are interpreted and presented to the user, are sometimes unclear. In this paper, we identify a number of candidate interpretations of annotation, and discuss the impact these interpretations may have on Semantic Web applications.

1 Introduction

The Semantic Web (SW) vision, as articulated by Tim Berners-Lee [2], is of a Web in which resources are accessible not only to humans, but also to automated processes, e.g., automated "agents" roaming the web performing useful tasks such as improved search (in terms of precision) and resource discovery, information brokering and information filtering. The automation of tasks depends on elevating the status of the web from machine-readable to something we might call machine-understandable. The key idea is to have data on the web defined and linked in such a way that its meaning is explicitly interpretable by software processes rather than just being implicitly interpretable by humans.

To realise this vision, it will be necessary to associate *metadata* (i.e., data describing content/functionality) with web resources. One mechanism for associating such metadata is annotation. In particular, we may wish to annotate resources with *semantic* metadata that provides some indication of the content of a resource. This is a further step along the way from simple textual annotations, as the intention within the SW context is that this information will be accessible not only to humans but also to software agents. In order to do this we require languages which will support the representation of semantic metadata.

Standardisation proposals for metadata languages have already been submitted to the World Wide Web Consortium (W3C), in particular the Resource Description Framework (RDF) and RDF Schema (RDF(S)) – see [8] for a discussion of the roles of these languages and of XML/XML Schema. However, such annotations will be of limited value to automated processes unless they share a common understanding as to their meaning. Ontologies (which have already proved their usefulness in a range of application domains [28, 23, 26] can help to meet this requirement by providing a "representation of a shared conceptualisation of a particular domain" and a shared, controlled vocabulary that can be communicated across people and applications [11, 12].

In addition to the requirement for representation languages that support the sharing and exchange of semantic information between applications, we must also have a common understanding of the annotation process. Schemas to support annotation have been developed [19], but these do not explicitly provide support for this understanding. What does it mean when we make an annotation, and what are the implicit tasks that are being performed? Within the SW context, confusion reigns as to the interpretation of the annotation task. In order to support the use of automated agents (a central tenet to the Semantic Web vision), we must be *explicit* about the assumptions that we make and the context within which such annotations should be interpreted. Note that our use of the phrase "the *semantics* of semantic annotation" refers to the provision of a consistent interpretation of the task, but we do not intend to present here a formal semantics, such as that provided for languages like DAML+OIL [27].

The paper is structured as follows. We first give a brief introduction to the COHSE project and the approach being adopted there. This gives an overview of our motivation and some context for the following discussions. We then discuss annotation and provide a classification of annotation tasks along with their intended semantics. We discuss the related issue of identification and finally conclude with some remarks concerning future directions and recommendations. References to existing and related work are made throughout the paper.

2 The COHSE Project

Our interest in annotation here is within the context of the COHSE (Conceptual Open Hypermedia Service) project [3]. COHSE aims to bring together an open hypermedia architecture (in particular the Distributed Links Service [4] or DLS) with ontological services in order to provide an architecture for the Semantic Web [10].

Detailed descriptions of the COHSE system can be found in [3, 10]. Put briefly, the COHSE approach consists of a *COHSE agent* (along with supporting services such as an *ontology service* and *annotation service*) that augments documents with links based on the semantic content of those documents. The sys-

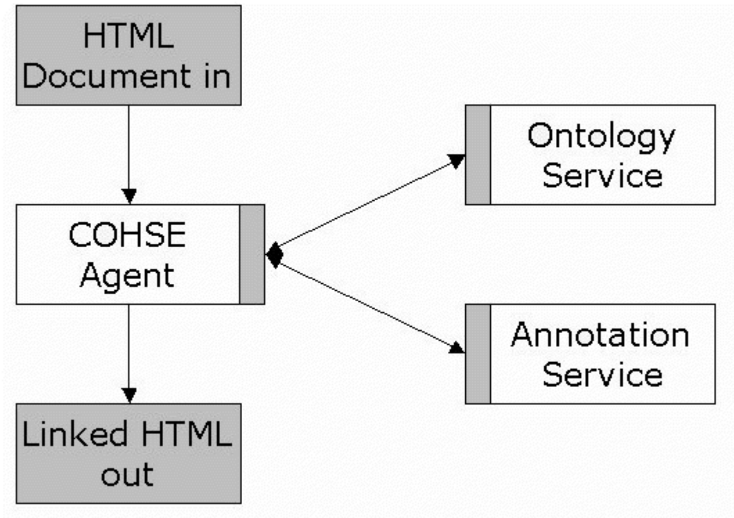

Fig. 1. COHSE Architecture

tem employs either a specialist browser (based on Mozilla[3]) or a proxy through which all http requests are routed. The words and terms that appeared in a document are used as entry points to an ontology. The relevant concepts in the ontology can then be used to determine appropriate targets for links out of the given resource. Key to the novelty of the COHSE approach is the provision of an *editorial component* within the agent. This component uses information within the ontology (such as hierarchical classification) in order to determine whether the links are suitable or to perhaps expand or cull the set of possible targets. Figure 1 shows a simplified view of the basic architecture of the system.

Figure 2 shows a page taken from Sun's Java Tutorial web site. In Figure 3, we see the same page augmented by the COHSE agent. A number of link anchors (signified by the small "L" icon) have been added to the page. One of these has been opened up, and we see a collection of possible targets which have been annotated as being "about" the particular concept selected – in this case the concept of byte.

In addition to using the words and phrases that appear in the documents, the COHSE agent can also use explicit metadata applied to the resources (rather than relying solely on mappings from words and terms). This approach relies on the ability to annotate resources with semantic metadata – where by semantic metadata we mean the explicit binding of concepts to resources rather than the use of terms and words as simple proxies for the concepts. The explicit annotations can then help guide the editorial component in its linking strategy. For example, if a passage in a web page has been annotated as being about

[3] `http://www.mozilla.org`

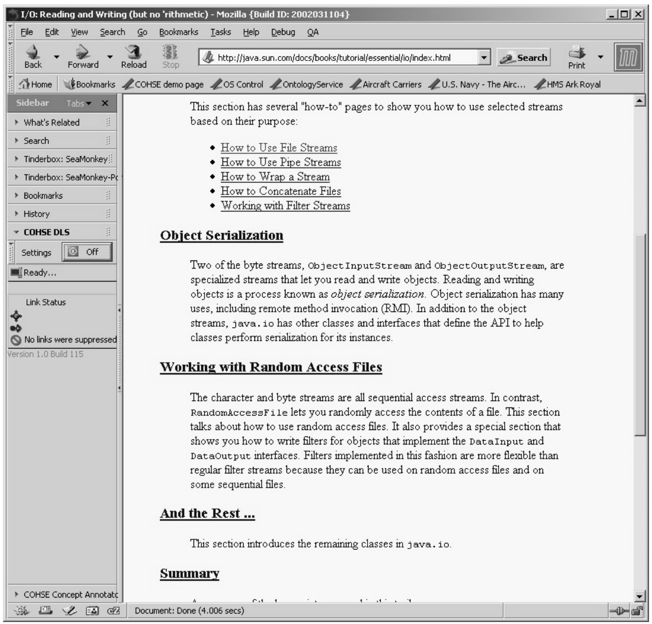

Fig. 2. Before COHSE Linking

a particular subject, say programming datatypes, the editorial component may know that there are certain terms that should be focused on within the context of that annotation (say the terms int or float) – an example of an agent using semantic information to make decisions as to its behaviour.

2.1 Produce and Consume

The situation has parallels with the underlying motivation for the use of rich languages for representing content on the Semantic Web. Languages like DAML+OIL [6] are being proposed as mechanisms which provide "machine-processable" semantic information. They provide an explicit representation of the relationships between terms and concepts which can then be used by reasoners or software agents to interpret those terms and concepts. The vision is one of providing shared conceptualizations, which then allow communities to share and exchange information unambiguously.

Within COHSE (or indeed other SW systems), there are two complementary strands, with annotation *providers* enriching content and annotation *consumers* using those annotations to process, organise and present information to end users. The consumer could be a sophisticated ontological search process or portal, or alternatively document enrichment through the addition of links as used by COHSE. It is key within this relationship, however, that consumer and provider *share underlying assumptions* about the annotations. Part of this sharing is

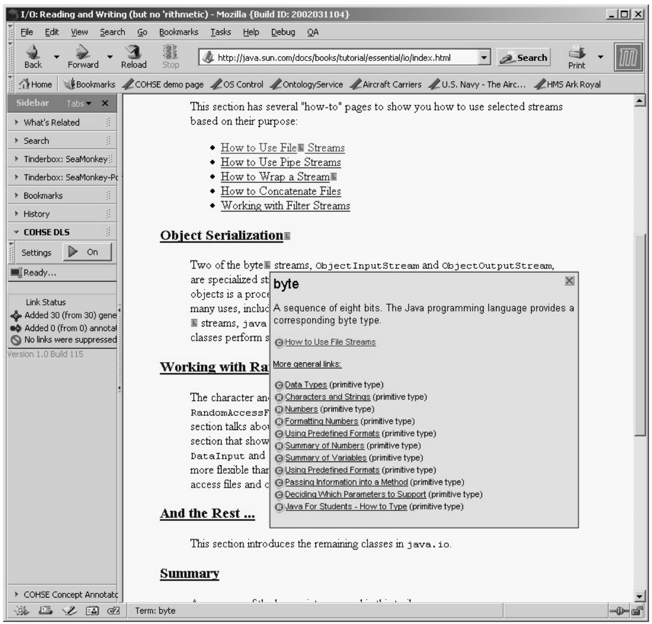

Fig. 3. After COHSE Linking

provided by the use of concept models or ontologies, part of it is provided by shared assumptions about the way these terms are to be used. If we build rich models, and then use them in a haphazard fashion, we are in some way selling ourselves short.

2.2 Linking as Annotation

The provision of dynamic linking as used by the COHSE project can be seen as a kind of annotation – in this case hypertext links are being provided rather than some textual annotation. This will be discussed in further detail below (as an example of **Link Annotation**), but we introduce the notion here as it has relevance to the description of COHSE.

Koivunen et al. [18] discuss approaches to Web annotations and categorise systems as, in the main, either **proxy-based** or browser-based. In a proxy-based approach, the annotations and document are merged by the proxy, with the browser seeing only the merged documents. In a browser-based approach, a specialist browser application will merge the annotations with the original documents while browsing. Annotations can be stored separately and provided via some annotation service (or kept within the proxy itself).

COHSE has two implementations, as either proxy or browser – the proxy having the advantage that no specialist browsing software is required, and delivery can be targeted at a number of platforms (e.g. mobile devices).

Within COHSE, the purpose of annotation is twofold: to populate a knowledge base for retrieval, and to provide anchors for links as the annotations are used to derive outward links from resources. At its simplest, then, annotation within COHSE can be seen as a mechanism that allows the user to specify possible link anchors within a document, with the anchor being associated with a conceptual description. This description will then be used to determine appropriate links at read-time. Simultaneously, the annotations are being used to provide link targets (as is the case in other, resource-discovery based, systems).

2.3 Extending Simple Annotation

COHSE's current implementation adopts a basic approach to the interpretation of annotation – an annotation simply associates a resource with a concept and no attempt is made to disambiguate the relationship between the concept and resource. This simple approach has served us well and allows us to extend and enrich the hypertext. An experiment based on Sun's Java Tutorial site[4] has been conducted and an evaluation of the resulting hypertext structure produced by the COHSE agent shows promising results[5].

A possible extension to this situation is to provide further information that describes in more detail the relationship between the resource and the annotation concept. This then has an impact in two ways:

– it can affect the way that the agent presents the link anchor;
– it can affect the way that possible link targets are found or displayed.

This leads us to a desire to classify and categorise the different ways in which this association between resource and concept could be made. The remainder of the paper proposes a number of different interpretations of the annotation process and discusses how those interpretations could affect the behaviour of systems such as COHSE.

3 Annotation

> **annotation** noun. A note by way of explanation or comment added to a text or diagram. *New Oxford Dictionary of English*

Annotation takes many forms and there are a number of what we could term "popular" ideas of annotation. Marshall [21] writes that "[annotation] has been construed in many ways: as link making, as path building, as commentary, as marking in or around existing text, as a decentering of authority, as a record of reading and interpretation, or as community memory". Here we briefly present a rough classification of annotation types. We will return to this in more detail in the later section on semantics.

[4] `http://java.sun.com/tutorial`
[5] See `http://cohse.semanticweb.org/evaluation` for preliminary findings.

Textual Annotation is the process whereby notes or commentaries are added to resources. Annotations of this kind have been used for many years in communities such as biology. For example the SWISS-PROT database [24] contains protein sequence information along with annotations describing functions, structure, domains, sites and so on. Within a database like SWISS-PROT, the annotations are first-class citizens, and are, in effect, the data. Although some use is made of controlled vocabularies such as GO [25], the hand-crafted and hand-curated annotations are primarily aimed at human readers.

This is the kind of activity supported by Annotea [17, 29, 18]. Extensions to the basic schema allow the use of richer annotation types [5] (for example commentaries can be marked as replies or gathered into threaded discussions), but a principle characteristic of this approach is that it is primarily aimed at human readers (and authors).

Systems such as the Distributed Links Service (DLS) [4] or 3rd Voice [20], allow the addition of links to arbitrary documents (including those in control of a third party). This **Link Annotation** extends the textual annotation notion, where here the content of the annotation is given, not by some text, but by a link destination (and possibly associated behaviour). Again, link annotation can be seen to be an activity primarily targeted at human readers.

Finally, we can consider what might be called **Semantic Annotation**, where the content of the annotation consists of some rich semantic information[6]. This idea of semantic annotation has been pursued in both the Ontobroker [7] and SHOE [16] projects and more recently in COHSE [3]. In both Ontobroker and SHOE, specialised markup was inserted into web pages – this markup contained semantic information drawn from an ontology providing richer descriptions of resource content. In COHSE, a more open annotation framework following the DLS philosophy is in use, allowing the decoration of arbitrary resources without the necessity to control the original document. Semantic Annotation is targeted not only at human readers of resources, but also at software agents – this does bring with it the requirement that relationships are explicitly represented. The use of semantic information taken from well defined ontologies will allow agents to make decisions based on those resource descriptions (for example the COHSE editorial component as described above).

Returning to Marshall [21], a number of different axes or dimensions are identified that reflect the forms of annotation. Included in these are a notion of *formal* vs. *informal*. Informal includes personal notes written in the margin while reading an article. Formal is deemed to be metadata following structural standards and assigned values using conventional naming authorities. The use of semantic annotation, drawing on conceptual models represented using well-defined knowledge representation languages can be seen to sit at the extreme end of the formal spectrum, perhaps even more so than Marshall's original intension of formality. Also of interest is the identification of *explicit* vs. *tacit* annotation. According to Marshall, many personal annotations are tacit – they

[6] Of course Textual Annotations can also contain rich semantic information, but in general this is not accessible to machine-processing.

are telegraphic and incomplete and rely on contextual information for their interpretation. For example, a bookmark, highlighted sentence or the annotation "No!" are examples of tacit annotation as we need extra information about the annotator, or the history of the annotation process in order to interpret them. An explicit annotation will carry sufficient information for its interpretation. As Marshall says, the dimension of explicit vs. tacit is crucially related to intelligibility – in the context of the provision of markup intended for software agents or processes the requirement of explicitness is particularly strong as such agents will not possess the real world knowledge, reading history, cultural background and so on, of human readers.

4 Semantics of Annotation

Here we present a classification of possible uses of annotation. This can be seen as a classification of the possible semantics of the annotation relationship (where here we use the term semantics in a loose fashion). For the purposes of this discussion, we consider the following situation. A web page with the URL `U` is being viewed and a region of the document corresponding to an XPointer expression `X` has been selected. This is to be annotated with a concept expression `C`.

What does it now mean to annotate resource `U#X` with concept `C`? Table 1 lists a number of what we might call use cases regarding this action. For each class described in the table, we discuss the ideas in more detail using simple concrete examples to illustrate the differences.

Note that the distinction between these different annotation types introduced in Section 3 can become blurred. For example, semantic annotation (e.g. the association of a resource fragment with a machine-processable concept description as discussed here) may result in the addition of a link if the resource is viewed using the COHSE agent. The **Type** column of Table 1 gives an idea of the annotation type in terms of Section 3.

Decoration is the Annotea view of the world, where annotations are seen as commentaries on resources. In the simple annotation scheme used by Annotea, the body of a resource is a chunk of HTML, which simply provides the (textual) content of the annotation. Other approaches (such as COHSE [1]) may extend this schema, however, to provide annotations of other types.

Linking (or possibly *Transclusion*, to borrow Ted Nelson's term) provides a simple COHSE view – annotations are simply a mechanism that provides link anchors. If the content of that annotation happens to be a complex conceptual description that then enables a client agent to support "better" linking, then all to the good.

Instance Identification makes a strong assertion about the resource `U#X`, i.e. that it is an instance of a particular class. For example, the resource `http://www.w3.org/TR/xptr` is a `CandidateRecommendation` of the W3C. The situation here is clear in part, because in this case, the object about which the assertion is being made (the XPointer Recommendation) is clearly accessed

Name	**Usage**	**Type**
Decoration	When the user views `U`, the concept `C` will decorate the resource fragment referred to by `U#X`.	link/textual
Linking	When the user views `U`, links about `C` will appear with the source anchor being the fragment `U#X`.	link
Instance Identification	We are making an assertion that there is some individual x in the world, such that x is an instance of the concept `C`, and the url `U#X` identifies x.	kb population
Instance Reference	We are making an assertion that there is some individual x in the world, such that x is an instance of the concept `C`, and the url `U#X` in some way refers to x.	kb population
Aboutness	The resource fragment `U#X` is "about" `C`.	textual
Pertinence	For any x such that x is an instance of `C`, the information in the resource fragment `U#X` is pertinent to x.	textual

Table 1. Possible Uses of Annotation. `U` is a URL, `X` is an XPointer expression and `C` is a concept

by the given URI. Dereferencing the URI provides exactly the object that the assertion is about.

For annotations of class **Instance Reference**, the situation is less clear. The resource `http://www.mcfc.co.uk/player.asp?PLAYER=1191` is about Shaun Goater the Manchester City football (soccer) player. We could annotate this resource with the concept `Footballer`, but the intended interpretation here is that there is an object in the world (Shaun Goater) that is an instance of `Footballer` and which is referred to or referenced by the given URI rather than a statement that the URI is an instance of the concept `Footballer`. A human reader seeing such an annotation would implicitly assume that the assertion was being made about the subject of the page (e.g. Shaun Goater), as the idea of a web page being a `Footballer` is nonsensical – to make this inference, however, requires background and world knowledge.

This distinction between **Instance Identification** and **Instance Reference** and the mechanisms that may be used to support the difference is discussed in the later section on Identification.

Aboutness gives a rather loose notion of annotation. In contrast to **Instance Identification** and **Instance Reference**, there is no assertion of the existence of a specific instance of the concept `C`. Instead there is a loose association of the resource with the concept. As an example of this, the page `http://www.nczooeletrack.org/` is about `Elephants`. It does not discuss a particular elephant, nor does it contain information that applies about the class of elephants (see discussion of **Pertinence** below).

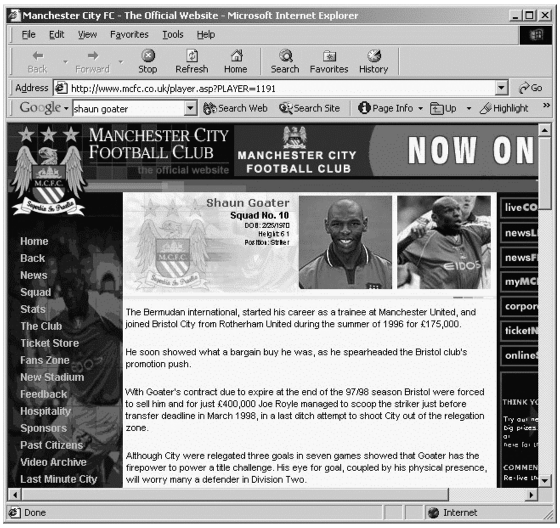

Fig. 4. Page about Shaun Goater

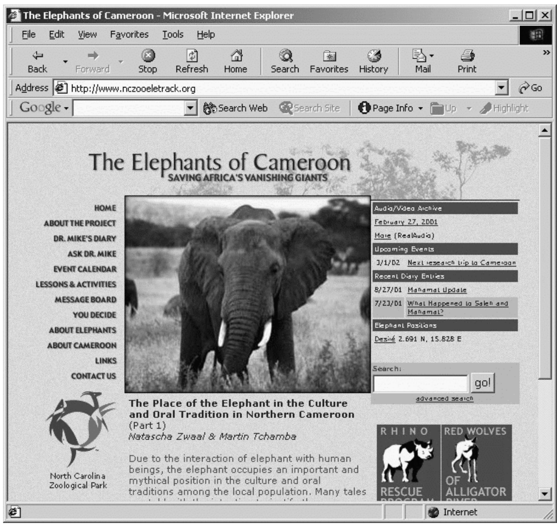

Fig. 5. Page about Elephants

```
<rdf:Description rdf:about="http://www.w3.org/TR/xptr/">
 <rdf:type rdf:resource=
 "http://cohse.semanticweb.org/ontologies/docs#W3C_Candidate_Recommendation"/>
</rdf:Description>
```

Fig. 6. XPointer spec is a Candidate Recommendation

Pertinence gives what we might call a kind of weak ontological extension. It allows us to make assertions about the classes and concepts within the ontology without actually explicitly enocoding or embedding that information within the ontology. Of course this means that the information may not be readily available to reasoning agents, but it may be that the information is not appropriate for a reasoner. For example, the EPSRC (Engineering and Physical Sciences Research Council) web site may have pages which contain useful information relating to `Researchers` such as employment opportunities, pay scales and the like. This is not necessarily information we would wish to model directly within the ontology, but is, in the main, of interest to researchers. An annotation of such a resource could be considered to be in the **Pertinence** class.

Aboutness and **Pertinence** could be considered as examples of textual annotation as introduced above – although the content of the annotation may have some richer structure, the annotation is essentially a note or commentary on the resource. We can consider **Decoration** and **Linking** as enabling mechanisms for the construction of hypertexts, in other words link annotation (although **Decoration** is also a kind of commentary mechanism). In contrast, **Instance Identification** and **Instance Reference** are about knowledge base construction, i.e. the population of an ontology or conceptual schema with instances and do not correspond directly with link or textual annotation (although the information could ultimately be used to generate links).

The OntoMat tool [14] supports annotation corresponding to **Instance Identification**. Instances of concepts are introduced and have fillers for their relationships harvested from information appearing on the web page being annotated. In the current version, annotations are not anchored to particular resource fragments, but are instead stored as markup within the web page being annotated. The new instances have generated identifiers which are based on the URI of the page being annotated. This could be extended to use external storage of the annotations (for example using an annotation service or RDF repository) along with an XPointer mechanism.

5 Instance vs. Aboutness and Identification

A key question to address when we consider annotation is that of *instance-of* vs. *aboutness*. RDF has a built in property `rdf:type` that allows us to make assertions about individual resources. For example, take the RDF statement shown in Figure 6.

This says that the resource `http://www.w3.org/TR/xptr/` is an instance of the class `W3C_CandidateRecommendation`. RDF is well set up to deal with such assertions. However, there may often be situations where we want to make an assertion that a particular resource is *about* a particular concept (in terms of its content), rather than saying it is an instance of it. This relates to annotations of kind **Instance Reference** as discussed above.

The instance vs. aboutness issue is closely related to the problem of **identification** of objects within the Semantic Web. In a fully-fledged implementation of the SW, we would expect to be able to make assertions not only about web resources, but also about objects, for example being able to assert information about Sean Bechhofer the person, not just about the URIs `http://www.cs.man.ac.uk/~seanb` or `mailto:seanb@cs.man.ac.uk`. In order to do this we need mechanisms that allow us to refer to objects that may not directly have an explicitly dereferenceable URI. Mechanisms such as **tdb** [22] or existential quantification over DAML+OIL `uniqueProperties` [15] have been proposed which allow us to refer to "the thing described by x" or "the thing with property x".

These mechanisms will then allow us to support interpretations such as **Instance Reference** as discussed above. For instance, in our example of Shaun Goater, we can now say that `tdb:20011030:http://www.mcfc.co.uk/player.asp?PLAYER=1191` has `rdf:type Footballer`, in other words the thing described by the given URI (i.e. Shaun Goater) is a footballer.

Note that we should not confuse RDF's **rdf:about** attribute with "aboutness" as discussed here. Within RDF, **rdf:about** is really a syntactic mechanism that relates a resource to RDF statements concerning it, rather than describing the content of some resource.

The CREAM framework [13] distinguishes different roles that correspond to the treatment of an annotation. **Quotation** copies an excerpt from a resource (such as the string "Shaun Goater"). This is a rather loose association, similar to **Instance Reference** as described above – the copied string is referring to some object in the world. **Reference** allows the metadata to use a pointer to a resource fragment – the example given uses a pointer to a particular place at `http://www.whitehouse.gov` in order to refer to the current U.S. president. If the actual president changes, the metadata will continue to refer to "the president". In this example, this is again an **Instance Reference** as the URL is not the president, but is a reference to the president.

The Annotea [19] schema[7] contains properties which link the Annotation to the resource which it is annotating – `annotates` refers to the enclosing URI and `context` provides the precise location, say using XPointer [9]. The schema also contains a `body` property which provides a link to the body of the annotation. The schema, however, remains agnostic as to the exact semantics of the annotation (in terms of our classification above). All that the annotation asserts is that the selected resource has an annotation which consists of the selected concept.

---

[7] `http://www.w3.org/2000/10/annotation-ns`

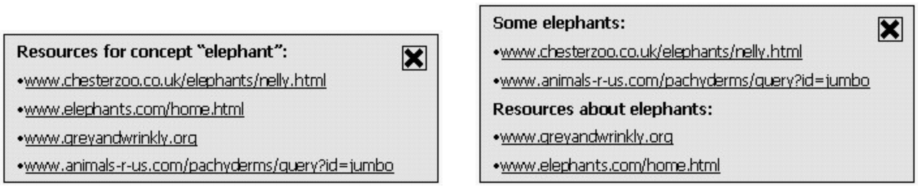

Fig. 7. Result Presentation depending on annotation type

There is no direct instance-of assertion and it is up to the application using the annotations to decide on the appropriate interpretation.

This is weak, and we suggest that extensions to the schema are needed in order to record and represent what the intended semantics of the annotation are. As an example of this approach, the COHSE annotations employed an extension of the W3C Annotea schema, with a property `http://cohse.semanticweb.org/annotation-ns#concept` being used to indicate that the content of the annotation is a concept. This property is a specialisation of the `http://www.w3.org/2000/10/annotation-ns#body` property from the Annotea schema. In addition, approaches such as **tdb** [15] give us the machinery to represent the differing interpretations of annotation, The **tdb** namespace provides [22] "...a space which is useful for describing entities, concepts, abstractions, and other items which are not themselves network accessible resources, but have been at some point described by network accessible resources. The "tdb" namespace designates the "thing described by" a resource at a given URI at the given time."

We must ensure, however, that tools provide adequate support for users during the process of annotation.For example, we may expect to be offered different options corresponding to the class or category of annotation being made. This can be seen as a requirement for *explicitness* in the process. We cannot make assumptions about what the intended semantics of the annotation should be.

6 Application Behaviour

What might the effects of the different uses be on the behaviour of applications? We use COHSE as an example of an application making use of semantic annotations in the following discussion, although this topic is relevant to many other SW applications.

Consider the distinction between Instance Identification, Instance Reference and Aboutness as introduced above. The first states that a resource `R` is an instance of concept `C`, the second states that `R` refers to some instance of `C` and the third states that the resource `R` is about the concept `C` (where aboutness is itself a rather loose notion). If this information is included in the annotation, we can make use of it in the following possible ways.

The information can affect what happens when the agent tries to find link targets for the concept `C`, when `C` has been identified as the concept associated with a source anchor. The classification of the annotations may help the application in organising and presenting the links to the user. For example, rather than simply displaying a list of targets, the targets could be grouped according to whether they are deemed to be actual instances of the concept, or simply "about" the concept. Figure 7 gives an example of what the popup link menu might look like in the COHSE application with the left hand side showing the bare list and the right hand the reorganised list.

In our Java tutorial example[8], an example of where this behaviour could help the user would be if we are dealing with a concept such as Java Servlet Engine. The user may want to find out more information about Java Servlet Engines (for example API documentation or an overview of what a Servlet Engine is). In this case pages described as being about the concept may be useful. Alternatively, the user may actually want to go and get a Java Servlet Engine, in which case an Instance Reference or Identification annotation will be of more relevance. The issue here is very much concerned with how information can be organised and presented to the user.

The scenario described above could, of course, hold true of any resource discovery agent – for example this extra information could be of benefit for search engines in ranking and presenting information.

If the annotation has been used to derive a source anchor for a link, this may then affect the way that possible link targets are found. If the user is looking at a resource `R` which has been annotated as being about some concept `C`, a sensible option for the agent would be to present links with targets which are instances of `C`. Alternatively, if the resource being viewed is described a being an instance of `C`, then it may be more appropriate to display resources about `C` first (providing me with some more context) rather than other instances of the the concept. Of course, such behaviour is strongly application dependent, and may also depend on factors such as user preferences. However, the presence of the extra information associated with the annotation allows the agent to make more informed choices about the way that results are presented to the user.

7 Concluding Remarks

Semantic metadata is set to play a major part in the implementation of the Semantic Web and annotation will be a primary mechanism for supplying the metadata which will then be used by agents as they retrieve information. In this paper we have presented a number of different interpretations for the process of semantic annotation. Current annotation mechanisms do not support this distinction, or if they do it is in an implicit rather than explicit fashion. Extensions to existing annotation schemas (such as Annotea) can provide some support, but must be done in an agreed fashion to ensure a shared understanding.

[8] `http://cohse.semanticweb.org/evaluation`

It is clear that even without an agreement on the precise interpretation, annotation information can be of use to applications, as is demonstrated by the current COHSE system. Without an agreement on the underlying assumptions behind the use of semantic annotation, however, software agents within the SW will be unable to perform their tasks in a truly consistent fashion. The consistent interpretation of notions such as Instance Identification and Aboutness will help SW applications to present and use information in ways that will further benefit users. Key to the provision of workable semantic annotation is a need for *explicitness*. We require explicitness of **context** to allow us to determine how to interpret the conceptual content of the annotations. In addition, we require that the intended **semantics** of the annotation be made explicit in order that agents which use the annotations can process and interpret them consistently.

8 Acknowledgements

This work was supported by EPSRC Grant GR/M75426. The authors would like to thank Bernard Horan of Sun Microsystems for his comments on the paper.

References

1. S. Bechhofer, I. Horrocks, C. Goble, and R. Stevens. OilEd: a Reason-able Ontology Editor for the Semantic Web. In *Proceedings of KI2001, Joint German/Austrian conference on Artificial Intelligence*, Vienna, September 2001.
2. T. Berners-Lee. *Weaving the Web*. Orion Business Books, 1999.
3. L. Carr, S. Bechhofer, C. A. Goble, , and W. Hall. Conceptual Linking: Ontology-based Open Hypermedia. In *Proceedings of WWW10, Tenth World Wide Web Conference*, Hong Kong, May 2001.
4. L. Carr, D. De Roure, W. Hall, , and G. Hill. The Distributed Link Service: A Tool for Publishers, Authors and Readers. *World Wide Web Journal*, 1(1):647–656, 1995.
5. P. Cross, L. Miller, and S. Palmer. Using RDF to Annotate the (Semantic) Web. In *K-Cap Workshop on Knowledge Markup and Semantic Annotation*, Victoria,B.C., Canada, October 2001.
6. DAML+OIL. `http://www.daml.org/language`.
7. S. Decker, M. Erdmann, D. Fensel, , and R. Studer. Ontobroker: Ontology Based Access to Distributed and Semi-Structured Information. In R. Meersman, Z. Tari, , and S. Stevens, editors, *Semantic Issues in Multimedia Systems. Proceedings of DS-8*, pages 351–369. Kluwer Academic Publishers, 1999.
8. S. Decker, F. van Harmelen, J. Broekstra, M. Erdmann, D. Fensel, M. Klein I. Horrocks, , and S. Melnik. The Semantic Web — on the Respective Roles of XML and RDF. *IEEE Internet Computing*, 2000.
9. S. DeRose, E. Maler, and R. Daniel Jr. XML Pointer Language (XPointer) Version 1.0. W3C Candidate Recommendation. `http://www.w3.org/TR/xptr/`, September 2001.
10. C. A. Goble, S. Bechhofer, L. Carr, D. De Roure, , and W. Hall. Conceptual Open Hypermedia = The Semantic Web? In *SemWeb2001 The Second International Workshop on the Semantic Web*, Hong Kong, May 2001.

11. T. R. Gruber. Towards principles for the design of ontologies used for knowledge sharing. In *Proceedings of International Workshop on Formal Ontology*, 1993.
12. N. Guarino. Formal Ontology and Information Systems. In *Proceedings of FOIS-98.*, 1998.
13. S. Handschuh and S. Staab. Authoring and Annotation of Web Pages in CREAM. In *Proceedings of WWW2002, Eleventh World Wide Web Conference*, Honolulu, Hawaii, May 2002.
14. S. Handschuh, S. Staab, and A. Maedche. CREAM - Creating relational metadata with a component-based, ontology-driven annotation framework. In *Proceedings of K-CAP 2001, First International Conference on Knowledge Capture*, Victoria, B.C. Canada, October 2001.
15. S. Hawke. How We Identify Things (on the Semantic Web)? `http://www.w3.org/2001/03/identification-problem/`.
16. J. Heflin, J. Hendler, , and S. Luke. SHOE: A Knowledge Representation Language for Internet Applications. Technical Report CS-TR-4078 (UMIACS TR-99-71), Department of Computer Science, University of Maryland, 1999.
17. J. Kahan, M.-R. Koivunen, E. Prud'Hommeaux, and R. Swick. Annotea: An Open RDF Infrastructure for Shared Web Annotations. In *Proceedings of the Tenth International World Wide Web Conference*, Hong Kong, May 2001.
18. M.-R. Koivunen, D. Brickley, J. Kahan, E. Prud'Hommeaux, , and R. R. Swick. The W3C Collaborative Web Annotation Project ... or how to have fun while building an RDF infrastructure. `http://www.w3.org/2000/02/collaboration/annotation/papers/annotationinf%rastructure`, May 2000.
19. M.-R. Koivunen and Ralph Swick. Metadata Based Annotation Infrastructure offers Flexibility and Extensibility for Collaborative Applications and Beyond. . In *K-Cap Workshop on Knowledge Markup and Semantic Annotation*, Victoria, B.C., Canada, October 2001.
20. M. Margolis and D. Resnick. Third Voice: Vox Populi Vox Dei? *First Monday*, 4(10), 1999. `http://www.firstmonday.dk/issues/issue4_10/margolis/`.
21. C.C. Marshall. Towards an ecology of hypertext annotation. In *Proceedings of HT98, ACM Conference on Hypertext*, pages 40–49, Pittsburgh PA, USA, 1998.
22. L. Masinter. "duri" and "tdb": URN Namespaces based on dated URIs. IETF Internet-Draft, `http://larry.masinter.net/duri.html`, April 2002.
23. D. L. McGuinness. Ontological issues for knowledge-enhanced search. In *Proceedings of FOIS-98*, 1998.
24. SWISS-PROT Annotated Protein Sequence Database. `http://www.expasy.org`.
25. The Gene Ontology Consortium. Gene Ontology: a tool for the unification of biology. *Nature Genetics*, 25:25–29, 2000.
26. M. Uschold and M. Grüninger. Ontologies: Principles, methods and applications. *Knowledge Engineering Review*, 11(2):93–136, 1996.
27. F. van Harmelen, P.F. Patel-Schneider, and I. Horrocks. A Model-Theoretic Semantics for DAML+OIL. `http://www.daml.org/2001/03/model-theoretic-semantics.html`.
28. G. van Heijst, A. Schreiber, , and B. Wielinga. Using explicit ontologies in KBS development. *International Journal of Human-Computer Studies*, 46(2/3):183–292, 1997.
29. World Wide Web Consortium. Annotea Project. `http://www.w3.org/2001/Annotea`.