Magpie – Towards a Semantic Web Browser

Martin Dzbor, John Domingue, and Enrico Motta

Knowledge Media Institute, The Open University, Milton Keynes, UK
{M.Dzbor,J.B.Domingue,E.Motta}@open.ac.uk

Abstract. Web browsing involves two tasks: finding the right web page and then *making sense* of its content. So far, research has focused on supporting the task of finding web resources through 'standard' information retrieval mechanisms, or semantics-enhanced search. Much less attention has been paid to the second problem. In this paper we describe Magpie, a tool which supports the interpretation of web pages. Magpie offers complementary knowledge sources, which a reader can call upon to quickly gain access to any background knowledge relevant to a web resource. Magpie automatically associates an ontology-based semantic layer to web resources, allowing relevant services to be invoked within a standard web browser. Hence, Magpie may be seen as a step towards a *semantic web browser*. The functionality of Magpie is illustrated using examples of how it has been integrated with our lab's web resources.

1 Introduction

Web browsing involves two basic tasks: (i) finding the right web page and (ii) *making sense* of its content. A lot of research has gone into supporting the task of finding web resources, either by means of 'standard' information retrieval mechanisms, or by means of semantically enhanced search [8, 14]. Less attention has been paid to the second task – supporting the *interpretation* of web pages. Annotation technologies [12, 17, 22] allow users to associate meta-information with web resources, which can then be used to facilitate their interpretation. While such technologies provide a useful way to support group-based and shared interpretation, they are nonetheless very limited; mainly because the annotation is carried out manually. In other words, the quality of the sensemaking support depends on the willingness of stakeholders to provide annotation, and their ability to provide valuable information. This is of course even more of a problem, if a formal approach to annotation is assumed, based on semantic web technology [1].

In this paper we describe Magpie, a tool supporting the interpretation of web pages. Magpie acts as a complementary knowledge source, which a user can call upon to gain instantaneous access to the background knowledge relevant to a web resource. Magpie follows a different approach from that used by the aforementioned annotation techniques: it automatically associates a semantic layer to a web resource, rather than relying on a manual annotation. This process relies on the availability of an *ontology* [7] – an explicit, declaratively specified representation of a discourse or problem matter. Ontologies are the cornerstone of the emerging semantic web: they provide conceptual interoperability, allow semantic agents to make sense of information on the web and to collaborate with other semantically aware agents. Magpie uses

D. Fensel et al. (Eds.): ISWC 2003, LNCS 2870, pp. 690–705, 2003.

ontologies in a similar way: to make it possible to associate meaning with the pieces of information found on a web page and then, on the basis of the identified meaning, to invoke the relevant services, or offer the user the appropriate functionalities.

The Magpie-mediated association between an ontology and a web resource provides an *interpretative viewpoint* or *context* over the resource in question. Indeed the overwhelming majority of web pages are created within a specific context. For example, the personal home page of a member of the Knowledge Media Institute would have normally been created within the context of that person's affiliation and organizational role. Some readers might be very familiar with such context, while others might not. In the latter case, the use of Magpie is especially beneficial, given that the context would be made explicit to the reader and context-specific functionalities will be provided. Because different readers show differing familiarity with the information shown in a web page and with the relevant background domain, they require different level of sensemaking support. Hence, the semantic layers in Magpie are designed with a specific type of user in mind.

In a seminal study of how users browse the web, Tauscher and Greenberg [21] presented the following statistics on the types of actions users may carry out:

- 58% of pages visited are revisits,
- 90% of all user actions are related to navigation,
- 30% of navigation actions are through the 'Back' button,
- less than 1% of navigation actions use a history mechanism

A fairly obvious conclusion from such statistics is that web users need support in capturing what they have seen previously. Current history mechanisms, 'Back' button aside, are of little help. Magpie is able to automatically track the items found during a browsing session using a *semantic log*. The semantic log allows *trigger services* to be subscribed to; they would be activated when a specific pattern of items has been found. One type of trigger service offered in Magpie is a *collector*, which collects items from a browsing session using an ontology-based filter. Examples of collectors are shown in the following section.

The rest of this paper is structured as follows. In the next section we give an overview of the functionality of Magpie through a scenario. Sections 3 and 4 discuss the Magpie design principles and architectural details. We then describe the different types of semantic services available in section 5. Finally, in sections 6 and 7, we review related research, and draw the conclusions from this research effort.

2 Scenario – Browsing the Web with Magpie

Imagine a journalist is writing an article on the Knowledge Media Institute (KMi) for a magazine. One of her tasks is to gather information about the important projects led by senior KMi staff. Using a web browser with a Magpie extension, she visits the home page of the lab's director Enrico Motta. After loading the page, she wants to highlight interesting concepts denoting researchers, collaborating organizations, projects, and research areas in the page. These concepts draw upon an existing ontology of academic and research organizations, which was populated by mining databases and web resources.

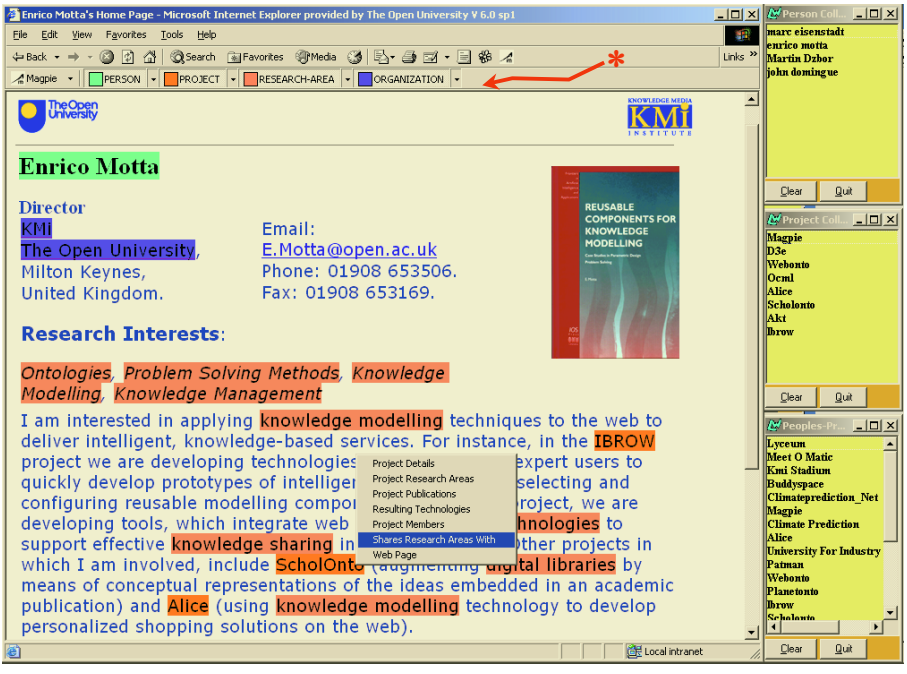

Fig. 1. Enrico Motta's home page viewed through Magpie. Known *people*, *organizations*, *projects* and *research areas* are highlighted using the Magpie toolbar (marked by '*'). On the right-hand side are three Magpie collectors – the top two log the people and projects found in the browsing session. The bottom one shows the (not explicitly mentioned) projects associated with the people found.

Fig. 1 shows the journalist's browser with the concepts of interest highlighted using the Magpie toolbar, which extends the functionality provided by Internet Explorer. As can be seen, Magpie preserves structure of the page, and only highlights the concepts upon user's request. This approach reduces the confusion, which may occur when the content and/or appearance of a web page are altered. Magpie toolbar (see close-up in Fig. 2) allows users to toggle highlighting for the specified types of entities, which were annotated in the page using an ontology-based lexicon. These types are ontology dependent – changing the ontology will modify the top-level headings displayed in the toolbar. As ontology represents an *interpretative viewpoint* we chose to leave the choice of ontology to the user. The button marked '♠' in Fig. 2 discretely hides the entire Magpie toolbar if not needed.

On the right-hand side of Fig. 1 are three Magpie *collectors*. These are automatically filled by Magpie *trigger services* as the user browses. During a browsing session, the entities found on accessed web pages are asserted into a *semantic log* knowledge base (KB). Collectors are set up to show a semantically filtered view of the semantic log. For instance, the top two collectors in Fig. 1 show the people and projects that have been recognized on the pages visited during the current browsing session. The bottom collector shows the projects associated with any people recognized during the browsing session, which were not mentioned explicitly in any page but originate from the populated domain ontology. Fig. 1 shows a number of projects the four researchers from the top-right collector are associated with.

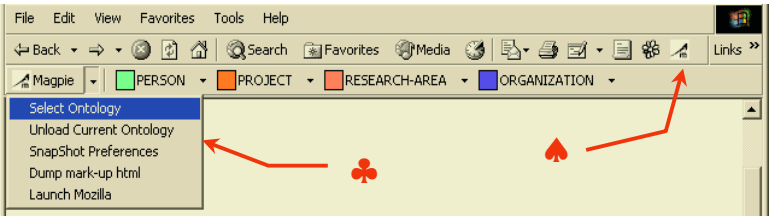

Fig. 2. Details of the Magpie toolbar showing four top-level classes that can be highlighted (‘*Person*’ through ‘*Project*’) in a page. The button marked ‘♠’ toggles the toolbar on and off; Magpie menu (marked by ‘♣’) enables the user to choose the ontology/viewpoint.

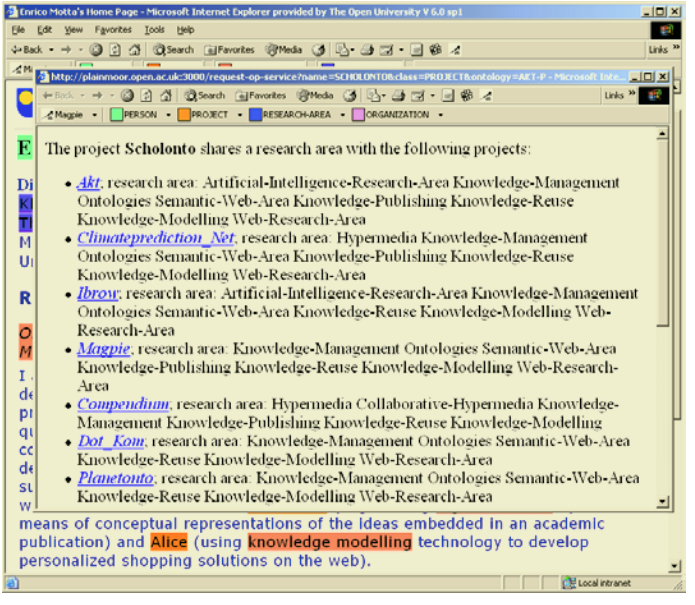

Fig. 3. Results of the ‘*Shares Research Areas With*’ semantic query invoked for the ‘*ScholOnto*’ project by the semantic menu action depicted in Fig. 1. Each bullet shows the name of a project followed by a list of overlapping research areas. Displayed answers are ordered according to the number of areas they have in common with ‘*ScholOnto*’.

Looking at the annotated page our journalist can see that the ScholOnto project might be suitable for her report, and she wonders if any related projects could be included in the same section. She right-clicks the ‘ScholOnto’ term, and the *semantic services menu* visible in Fig. 1 appears. The choices in the menu depend on the class (type) of the selected entity within the selected ontology. In our case, ‘ScholOnto’ is classified as a Project, so project-related options are displayed. Selecting an option labeled ‘Shares Research Areas With’ one can explore semantic neighbourhood that comprises projects sharing one or more research areas with ScholOnto. The results in Fig. 3 (foreground window) show which research areas has each ‘neighbouring’ project in common with ScholOnto. In our case, related projects include ClimatePrediction and Ibrow. To view details of the Ibrow project, our journalist may either select the ‘Ibrow’ record in the collector, or in the window shown in Fig. 3, and

then the 'Web Page' option in the semantic menu. Selecting items in collectors brings up the same semantic services menu as if items were selected on a web page.

3 Basic Design Principles

The overall goal of this project is to support the interpretation of web documents with no a-priori mark-up through the addition of an ontology-derived semantic layer. This goal may be unfolded into a set of design principles and high-level functional requirements for providing ontology-based support for navigating and sensemaking. Each principle is listed and briefly justified below (the implementation is then detailed in sections 4 and 5):

- Magpie should extend a *standard web browser* to minimize the users' effort when learning to use the tool.
- Magpie users should not incur a significant time penalty. In contrast with most approaches to Named Entities Recognition (NER), Magpie must always provide *fast, real-time mark-up*. More precise mark-up can still be carried out in the background by a specialised NER semantic service, while the user browses. An additional mark-up would then be delivered to the user's browser *incrementally* (progressively), thus refining the simple but fast mechanisms.
- Magpie should allow users to select a particular *viewpoint* – i.e. the ontology used for mark-up and annotation should be customizable and selectable by the user[1] (the ontology may be downloaded from the web or read from a local disk).
- Magpie should *separate* the mark-up from the documents to facilitate different viewpoints (of different communities) to be layered over the same web resource.
- Magpie should *preserve the appearance* of a web page – users would quickly get confused if web pages browsed through a semantic browser did not look the same as when browsed traditionally.
- Magpie should process *any web page* – we assume the web documents are not 'pre-marked-up' by an author or librarian (e.g. using XML or RDF). However, if such a mark-up exists, Magpie should be able to make use of it.
- Magpie should provide an interface for a *simple publication of the semantic services* that can be deployed by the service authors, as well as an *easy subscription mechanism* for the service recipients/users.

4 Magpie Architecture

The architecture of Magpie is shown in Fig. 4. Magpie is essentially a bridge – a *mediator* between formal descriptions used by the ontology-based service providers and semantically unstructured HTML documents. In order to facilitate the aspect of servicing, the Magpie technology consists of a *Service Provider* component and a *Service Recipient* component. In a more traditional language of web development

[1] In our scenario, we worked with a single ontology but there is a mechanism for ontology selection built-in in the Magpie plug-in (see left-hand corner of Fig. 2).

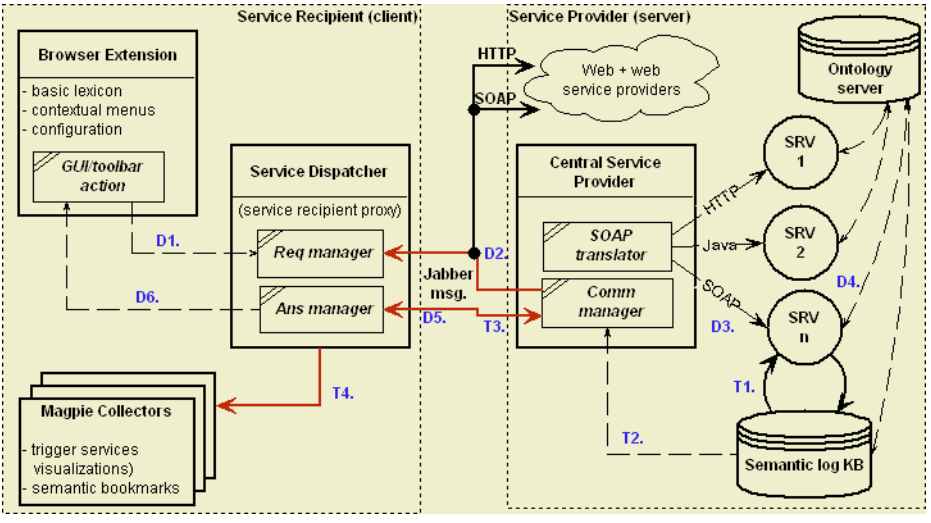

Fig. 4. Overall architecture of the Magpie framework for semantic browsing

these two components could be approximated as a server and a client, respectively. However, our notation emphasizes the fact that the document source (i.e. web server) and the source of a semantic service may not be one and the same computer. Similarly, a concept of 'client' suggests that an active role is always on the side of a browser with a server only passively serving requested data. In the context of semantic services, this is not appropriate. Mainly because (as we show below) there are circumstances where the server may become the element driving the conversation or providing the information that is semantically relevant but not explicitly requested.

In line with the web services paradigm there may be numerous providers of the same service and many different services [20]. Currently, the service provider component of Magpie is built around a suite of tools providing access to a library of knowledge models containing domain ontologies, populated KBs, semantic services and a semantic log KB. Some of the tools – e.g. a customized web server [19], which offers a library of methods to dynamically generate the appropriate content and reason about it – communicates via standard HTTP. In addition, there are tools communicating via SOAP – a popular protocol for web services [20]. In both cases, the underlying representation of the ontologies is shared.

Magpie accepts ontologies represented in RDF(S) [2], DAML+OIL [5], and OCML [16] (the latter being the internal representation for reasoning). In the future, we intend to include ontologies represented in OWL [18]. The services (cf. Fig. 1 and Fig. 3), are defined in one of the *Services* modules (SRV*) of the central Magpie service provider, and detailed in section 5.1. The purpose of the ontology of services is to abstract from the actual implementations, and enable the association of the semantic services with the object/entities on a web page. The *Semantic log KB* – one of the unique services in the Magpie suite, is used by the Magpie trigger services, which are described in section 5.2.

In order to use the services meaningfully, as well as to associate them with the appropriate terms, ontologies need to be regularly populated and updated. In our case, a set of techniques ('populators') was used to populate the domain ontology from

heterogeneous data stored in web-accessible RDF documents, mined from organizational databases, or extracted from the standard web pages. The ontology population process is beyond the scope of this paper; for more information see [6].

4.1 Magpie Service Recipient Components

As shown in Fig. 4, the Magpie 'client'-side suite consists of a Magpie Browser Extension (of Internet Explorer), Magpie Collectors, and a Magpie Service Dispatcher. The primary purpose of the *Magpie Browser Extension* is to sit in the browser, and control the entire interaction with Magpie facilities. Specifically, it contains the user interface components, which visualize entities found in a web page, and enables users to interact with the semantic services through a simple contextual menu. The *Magpie Service Dispatcher* acts as a dedicated semantic proxy for the user's web browser. Its main task is to manage the communication between the menu in the Browser Extension and the appropriate dispatcher of the service providers. The Magpie Dispatcher delivers all the user's requests and the provider's responses in a uniform way in a form of customized XML messages.

4.1.1 Magpie Browser Extension (Plug-In)

As mentioned earlier, the Browser Extension is embedded in the user's web browser, and it is responsible for managing the interaction between the user and the semantically enriched browser. The plug-in is implemented in an unobtrusive way as a dynamically loaded library (DLL) that is registered to Internet Explorer. When the user wishes to turn the semantic capabilities and services on, s/he only activates the Magpie toolbar that forms an external GUI to the plug-in by pushing a single button.

The browser extension contains a small HTML parser that is able to highlight the entities from a particular ontology. A pre-condition of a successful parsing *within* the Magpie browser extension is the definition or download of an ontology-derived lexicon from an appropriate service provider. Although not as accurate as many existing NER techniques, simple lexicon-based parsing is extremely fast, and thus satisfies our low time overhead design constraint (see section 3).

In our case, the lexicon entries are generated overnight from the instances within the ontological knowledge base, which is populated from various sources (e.g. databases). We use several simple linguistic rules, such as the recognition of abbreviations or people's initials. Also, ontology specific transformation rules can be applied. For example, the classes in our ontology have `pretty-name` and `variant-names` slots, which can be included in the lexicon, and consequently, used to recognize the concepts of interest on a web page. Simple rules can then derive for example, variants "J. Bloggs" or "Bloggs, J." from a `pretty-name` "Joe Bloggs". The specific rules applicable to our scenario from section 2 use the AKT reference ontology[2].

We are investigating how other NER mechanisms can be incorporated into the parser to enhance its precision. To this extent, the web services paradigm seems to be a plausible way forward. Instead of adding more complex NER techniques to the browser extension, we suggest leaving the browser plug-in thin, and implementing the

[2] More information available at http://www.aktors.org.

advanced NER algorithms as (semantic web) services available upon a user's request. More on this topic follows in the next section.

The HTML annotation process in Magpie comprises a number of simple steps. When an entity of interest is recognized in the web page, the HTML annotator (a part of the visual interface), annotates it using <SPAN…> tags, and links it with a relevant ontological instance/class within the chosen ontology. This procedure creates essentially *a semantic layer* over the original document. The original content remains untouched; only the interesting concepts and the corresponding text on the page are highlighted.

This approach to visualizing the semantic layers means that users remain in control of what types of entities are visible at any time. We argue that this improves navigation through the content, and avoids the greatest danger of various link-recommending systems (see discussion e.g. in [15]) – overwhelming the users with too much information. When the recognized entities are highlighted, the Dispatcher also passes them to a *Semantic Log Updater*, which is one of our central semantic services. This service asserts the entities as facts into the *Semantic Log KB*. The purpose of semantic logging is addressed later in section 5.2, and more information about the delivery implementation is described in the next section.

The Magpie Browser Extension incorporates three user interface components, which enable users to interact with the semantic services. We mentioned the *Visual Component* for highlighting the matched entities found in the page. The second component features the *Semantic Services Menu* (shown in Fig. 1), which is created on the fly in a close interaction with the *Dispatcher* and the *Services* module of the Magpie service provider. The last component of the Service Recipient suite takes care of the Magpie *trigger services* – for instance, various *Collectors*, *Summarizers* and *Visualizers*. Trigger services and the individual components are described in section 5.2. We should note at this point that unlike the contextual menu-based services, the user does not explicitly request the trigger services. They are typically pushed from the service provider to reflect some recognized pattern in the recognized entities. The Magpie Browser Extension GUI in a form of a Magpie toolbar is shown in Fig. 2.

4.1.2 Magpie Service Dispatcher

The role of the Service Dispatcher is to handle all interactions between the user with the Magpie-enabled browser and the respective Magpie service providers. This approach is an alternative to the GET/POST requests available in the standard HTTP. Magpie still supports HTTP requests, but a growing number of services are available in formats that do not lend themselves for a seamless integration into a semantic web browser. Separating interface and communication gives us several advantages.

The clean separation enables us to implement interface so that it can abstract the user's actions and automatically generate appropriate XML form-based communication request. The XML forms needed for the communication are delivered together with the lexicon for a particular ontology upon the user's request. The form 'filling' occurs automatically – the user's right click generates a Semantic Services Menu consisting of the available services. Selection of one option in this menu basically 'fills in' the associated XML form with the data the user clicked on. Once filled in, the form is passed to the *Service Dispatcher* that delivers it on the user's behalf to the appropriate service provider (or its Dispatcher) for processing.

The second advantage becomes obvious when we take into account that it may take a while to process any particular request. Thus, instead of keeping a stream open to the specific service provider, the Magpie Dispatcher will wait until the provider is ready. In other words, the division enables us to implement the service request/response communication in an *asynchronous* way. This facilitates scalability, flexibility and enables more customized interaction.

Third, the Magpie Dispatcher has a direct access to the information pushed by the *trigger* services that the user subscribed to, because everything pushed towards the user passes through the dispatcher. Here, it is more practical to re-direct pushed information into a suitable window managed by the dispatcher rather than directly to the browser or browser plug-in. The main benefit is in lifting burden with managing the windows, connections and requests/responses from the browser, and enabling a service provider to address the user (this *bi-directional communication* is not possible with standard HTTP).

Finally, the Magpie approach is centered on the specific viewpoints onto the web facilitated by different ontologies. We envisage that users would select a particular ontology depending on their current task from a set of 'subscribed' ontologies. In this way, the interface will reflect the user's current needs and context. However, this is not everything. Because the Dispatcher is loosely connected to the web browser it can be easily used by other information accessing applications (e.g. word processors, eMail clients or instant messengers). Hence, the services could be made available in other environments in addition to the web browsers. For example, we have successfully implemented Magpie for Microsoft Word XP based on this architecture.

The flexibility of Magpie semantic services in respect to both, the selected ontology and the target user environment, is a strong and unique feature of our Magpie technology. It means that the content of a particular resource may be *re-interpreted* in the context of a new ontology (i.e. re-parsed and re-annotated). Consequently, different semantic services may become available to shed light on the particular aspects of the document. Presently, our main constraint in respect to the ontology flexibility is that at any time, one ontology is actively used for the annotation. This should avoid most of the consistency issues between multiple knowledge sources.

5 Semantic Services

In section 4, we presented the conceptual architecture of Magpie, and showed how a semantic layer is created, displayed and activated. The main benefits of using Magpie however are generated from the ability to deploy semantic services on top of the semantic layer. These services are provided to the user as a physically independent layer over a particular HTML document. Magpie distinguishes between two types of semantic services, each having a specific user interaction model. In Fig. 4, the two types of user interaction are shown using process paths labeled '**D***' and '**T***'.

According to the communication model in Fig. 4, the parser inside the browser identifies entities from a chosen ontological lexicon in the raw web page. Discovered entities are annotated, and using a sequence of arrows '**D1**' → '**D3**' recorded in semantic log. Meanwhile, the annotated page is displayed in the web browser (see also section 4.1.1). The path labeled with '**D***' represents services activated on a

user's request; i.e. a user explicitly selects an entity s/he is interested in, and by a right mouse click invokes a contextual *Services menu*. This layer of *on-demand* semantic services is described in section 5.1. The initial point of the *on-demand* interaction ('**D1**') is always in the browser. Alternatively, semantic services may be based on patterns or *footprints* of the entities that co-occur in a particular document or browsing session (path with labels '**T***'). In section 5.2, we refer to this log-based functionality as *trigger services*, and these services originate always at the service provider's side.

5.1 On-Demand Semantic Services

Semantic services are enabled by clicking on the 'Magpie' button in the browser's main toolbar (see marker '♠' in Fig. 2). This action displays a dedicated toolbar, where a particular ontology can be selected, as shown in Fig. 2. Once the semantic services are activated, the contextual (right-click) menu of a web browser is overridden by an *on-demand services menu* whenever the mouse hovers over a recognized entity. The 'on-demand services' menu is also context-dependent as could be expected; however, in this case, we are dealing with a semantic context defined by the membership of a particular entity to a particular ontological class. The class membership of the entities is contained in the ontology or a lexicon generated from ontology.

In addition to domain ontologies that can be selected by the user (see marker '♣' in Fig. 2) to facilitate a viewpoint of a particular community, Magpie uses a special *Services* ontology. This ontology is part of the 'Ontology server' module (see top-right hand corner in Fig. 4), and it formally defines what operations/services can be performed for particular class(es) of entities, and the semantics of each operation. The semantic services are defined and published in line with standards of the emerging web services technology [20]. Thus, different groups of users may see different services to suit their knowledge or expertise. Generally speaking, the services may be published or revoked by their authors, and/or brokered to achieve richer interaction. In the scenario of using Magpie as a semantic portal for organizational research, the services were defined for the individual ontological classes directly in the 'Ontology server' without any specific publication or brokering mechanism. The services for the class *Project* are shown in the Semantic Menu displayed in the center of Fig. 1.

Similarly to parsing and annotation, the 'on-demand services' menu is generated on the fly. When a right click occurs, it is handled by the Magpie Browser Extension, and through the Magpie dispatcher the Magpie server proxy may be asked for the updated list of services that are available for a particular entity. This list may be cached in the browser to avoid delays. The request uses the information about class membership of a particular entity that was created while annotating the content (see section 4.1.1). If there are any applicable semantic services available, Magpie displays them in a menu, and lets the user choose what s/he is interested in. A selection of an option leads to a request to the Magpie dispatcher to contact the appropriate service provider and perform the requested reasoning. The knowledge-level reasoning facilitated by a particular service provider gives the requested context for a particular entity. This is delivered back to the web browser to be annotated and displayed. An example of a response is visible as a new browser window in the foreground of Fig. 3.

Hence, Magpie facilitates two complementary methods for web browsing. First, it implements syntactic browsing through the `<A HREF=...>` anchors inserted into a document by its author. A document accessed via anchors is parsed, annotated, and displayed with a Magpie toolbar to support semantically enriched user interaction (as described in section 4). The second browsing method follows the customized semantic anchors created during the automatic annotation, and the applicable, dynamically generated semantic services. While the first method gives access to physically linked content, the second method makes available the semantic context of a particular entity. The two methods are visually differentiated to minimize confusion, and provide complementary functionality. Fig. 1 shows a sample semantic services menu for term 'ScholOnto' (which according to the ontology, belongs to the 'Project' class). The semantic context corresponding to the user's request for similar projects is displayed in Fig. 3 *as if* the user followed a navigational link, and in this case contains a list of ontologically related and ordered projects.

5.2 Trigger Semantic Services

User-requested (on-demand) semantic services are one technique for interacting with the relevant background knowledge. A number of researchers stress the importance of active or push services, which we describe next in the context of Magpie. A background to this kind of services is discussed e.g. in [6]. The main feature distinguishing active services from the user-requested ones is that they tend to "look over the user's shoulder", gather facts, and present conclusions.

Such services are depicted in Fig. 4 by the interaction path containing labels '**T***' starting on the right hand side. As can be seen, a pre-condition for having active services is to keep *history logs* of browsing, particularly a log of the recognized entities. The label 'browsing history' is more than appropriate because a log accumulates findings not only from the current web page, but also from previously visited pages in the same browsing session.

The process of semantic logging runs in parallel with the web page annotation. While an annotated web page is displayed in a browser, the recognized entities are sent to the Magpie server component responsible for semantic log maintenance. The logged data are asserted as *facts* into a 'working' KB. Several *watchers* monitor and respond to patterns in the asserted facts. When the relevant assertions have been made for a particular watcher, a semantic service response is *triggered*, and applicable information delivered to the Magpie dispatcher on the client's side that in turn displays it in a dedicated window next to the user's web browser. This interaction is *asynchronous* – the service provider starts the communication, contacts the user's dispatcher, and pushes potentially relevant information.

A few examples of the results of a trigger service firing are shown on the right-hand side of Fig. 1 ('People', 'Projects' and 'People's Projects' collector windows). Definition of watchers underlying some trigger services has been published in [6], and we shall not repeat it here. However, generally speaking, when a web page is viewed in Magpie, one of the service providers responsible for the maintenance of the Semantic Log asserts `found-item` facts, for each of the lexical entities found, into the Semantic Log KB (see Fig. 4). The watcher, which may be implemented as an independent service provider, say "*SRV n*" in Fig. 4, triggers if a *Person* is found in

the log, and s/he is a member of a *Project*, which is not yet present in the log. Once trigger fires, the *Project* and the URL of the page it relates to, are collected. Future work will enable Magpie users to create watchers using a direct publication interface, as well as subscribe to the watchers/triggers of interest.

The information deliverable in this way may range from simple collections of relevant items to sophisticated guidance on browsing or browsing history visualization. Since the service provider taps into a knowledge base constructed potentially from the logs of community members, the guidance or history visualization may draw on community knowledge and behaviors. This type of setup may seem surprising in our scenario presented earlier because a journalist is clearly *not a member* of KMi community. Does it make sense to send her community-relevant information?

Our view is yes – this setup may be seen as a journalist (an external agent) adopting the viewpoint of a specific community to *interpret* and *make sense* of a given web resource *from the perspective* of that community. Thus, a formal membership of a particular community and the utilization of their ontological viewpoints are two different roles, each of us can be involved in. Since a trigger service can be (in principle) selected and subscribed to, there is nothing wrong in tapping to the knowledge of a community of which the user is *not a formal member*. On the contrary, this enables him or her to see the document in its 'native' context.

This is clearly beneficial, especially if we follow Tauscher and Greenberg's argument [21] that 58% of all visits to web documents are to sites visited previously, but that history mechanisms are used infrequently. The large number of re-occurring visits calls for a sophisticated approach to the management of browsing histories. Indeed, one of design recommendations from their study was that bookmarks should have a *meaningful* representation. History management based on the semantics of the visited pages, and implemented by a *triggered* semantic layer may help to alleviate issues with the syntactic and linear (access time ordered) methods.

Although the design goal for our two types of services is the same – to provide users with additional knowledge to support the interpretation of web pages and to assist in information gathering – the underlying paradigms are different. The 'on-demand' services are invoked by a specific user request. Goal-driven reasoning from the user's query leads to a response, which is typically presented as a new web page. The trigger service is invoked when a watcher matches a pattern within the semantic log. The pattern is equivalent to data-driven reasoning, results of which are displayed by a change in the interface. In Fig. 1, trigger services amend one of the three collectors.

6 Related Work

One of the inspirations for Magpie was the COHSE system [3]. COHSE combines an Open Hypermedia System with an ontology server into a framework for ontological linking – an ontology-derived lexicon is used to add links to arbitrary web pages. The links are added either by proxy server or by an augmented Mozilla™ browser. The distinctions between Magpie and COHSE are due to their differing design goals. The design goals for COHSE were (i) to separate web links from the web pages and (ii) to make these links conceptual (i.e. potentially generated from ontology). The goal for

Magpie is to support interpretation and information gathering. Magpie's interface enables ontological differences to be highlighted, and the services provided are dependent on the class of entity found. Magpie also offers *trigger* services via semantic logs. Neither type of Magpie service is meant to replace traditional links; they act as an auxiliary knowledge source available at the user's fingertips.

In the last few years, a number of tools have emerged that support the annotation of web pages. A classic example is the Amaya HTML editor, which implements the Annotea infrastructure [12]. Annotea facilitates the RDF-based mark-up of documents as they are created. The authors or viewers may add various meta-statements to a document, which are separate from the document itself and are accessible to collaborating teams via a centralized annotation server. The annotation in this sense centers on attaching additional information to a chunk of content on an arbitrary web page. This feature of Annotea makes it a powerful tool for the joint authoring of documents where a small group of collaborating agents share a common goal. However, the same feature may make it more difficult to facilitate a similar form of annotation sharing in 'open' user communities. In these cases, there is no guarantee that a freely articulated annotation would convey the same meaning to the different users.

Another difference of the Annotea framework as compared to Magpie is the source of annotations. Annotea assumes that at least one author (human) is willing to invest additional effort into making a page semantically richer. Magpie is more liberal and assumes a reader subscribes to a particular domain ontology, which is then used to provide relevant background knowledge. It may be argued that ontology creation takes even more effort than manual document mark-up. This is true; however, an ontology is a domain model, a shared viewpoint that can be re-used for different purposes, not solely for the annotation of a single document. Thus, the effort spent on designing a shared ontology is greater in the short term but in the longer term, it is a more cost-effective way of recording a shared point of view. Moreover, ontologies are increasingly available for several domains, so in many cases, no development effort is actually required.

A similar approach to annotating documents can be found in other research projects. The CREAM-based Ont-O-Mat/Annotizer [9] is a tool similar to MnM [22], which integrates ontologies and information extraction tools. As with MnM, Amilcare [4] provides information extraction support, and ontologies are represented in DAML+OIL. Annotations in this framework are very close to those advocated in this paper. Any ontological instance, attribute or relation known in a particular ontology may be an annotation hook. A key feature of this tool is its use of *discourse representations* to structure the relatively flat output of Amilcare according to the chosen ontology, thus facilitating ontology population.

The CREAM research team show an important feature of ontology-based annotation and document enrichment. Namely, any annotating tool must be aware of already existing (i.e. recognized) entities and their relationships; otherwise harm can be done with redundancies and multiple definitions. CREAM's annotation inferences resemble our trigger services produced by a data-driven reasoning. On the other hand, our 'on-demand' services smoothly and seamlessly address the issue identified above – the awareness of the existing relationships and the actual context of ontological instances.

The SHOE project [11] proposed an extension to HTML to allow the specification of ontological information within common HTML-based documents. In addition to

the inclusion of a semantically rich, ontological knowledge, SHOE tried to make these inclusions re-usable and understandable 'throughout the web'. An editor was developed to support the annotation. As with the tools mentioned above, and unlike our Magpie framework, SHOE relies on the offline mark-up of web documents. Once that is accomplished, the enriched documents are published, and dedicated tools may use the contextual knowledge (e.g. Exposé web crawler [10]).

7 Concluding Remarks

Reducing the information overload caused by the growing web is often cited as the premise for work on supporting the retrieval of relevant documents. But finding relevant documents is only half of the story. Their interpretation involves a reader in understanding the surrounding context, in which the document was created. In order to gain the full understanding, a reader will require knowledge of the specific terms mentioned and the implicit relationships contained both within the document and between the document and other external knowledge sources. Magpie addresses this issue by capturing context within an ontology, which then is used to enrich web documents with a semantic layer. Semantic services expose relevant segments of the ontology according to the user's needs. The choice of ontological viewpoint for interpreting a particular web page drives the interpretation bottom-up – by the user rather than domain expert or knowledge engineer.

Magpie users browse the web in a standard way with negligible differences in the user experience. Magpie achieves this by extending standard web browsers with standard mark-up languages, without altering the layout of the web page and imposing any significant time overhead. The key principle is that the user controls to what extent semantic browsing comes to the fore. The Magpie toolbar enables concepts to be made visible according to their ontological category, and the Magpie infrastructure enables arbitrary semantic actions to be triggered by patterns of items found within a semantic log. Trigger services also allow certain tasks to be delegated. In the scenario we showed how discovered entities could be used for a later inspection. However, Magpie allows more complex trigger services to be implemented. For example, the Magpie proxy may automatically parse web pages linked to the current page, thus allowing reconnaissance services similar to those found in Letizia [13] to be set up.

Attention as opposed to information is now widely acknowledged to be the scarce resource in the Internet age. Consequently, tools that can leverage semantic resources to take some of the burden of the interpretation task from the human reader are going to be of enormous use. We believe that Magpie is a step towards achieving this goal.

Our current effort is focused on deploying the Magpie suite of tools within the climate*prediction*.net project. Using the scheme that was successfully deployed in the SETI@home project, the idea of climate*prediction*.net is to exploit the idle time on PCs to run multiple versions of the UK Met Office climate model. Running large numbers of perturbed climate models (the project aims to collect 2M users) will overcome uncertainties present in the modeling (and hence prediction) process. During their participation in the project, the users would run climate models on their computers for several months. Magpie will be used for the purposes of interacting with and making sense of highly complex analyses of climate data that will be

produced from running a statistical ensemble of perturbed climate models. Magpie will also enable lay members of the public to explore the rich scientific resources that exist in the domain of climatology and climate prediction. Thus, it is hoped that the semantic browsing capabilities of Magpie will serve as an *enabling technology* for the increased public understanding of science.

Acknowledgments. The Magpie effort is supported by the climate*prediction*.net and the Advanced Knowledge Technologies (AKT) projects. Climate*prediction*.net is sponsored by the UK Natural Environment Research Council and UK Department of Trade e-Science Initiative, and involves Oxford University, CLRC Rutherford Appleton Labs and The Open University. AKT is an Interdisciplinary Research Collaboration (IRC) sponsored by the UK Engineering and Physical Sciences Research Council by grant no. GR/N15764/01. The AKT IRC comprises the Universities of Aberdeen, Edinburgh, Sheffield, Southampton and The Open University.

References

1. Berners-Lee, T., Hendler, J., and Lassila, O., *The Semantic Web.* Scientific American, 2001. **279**(5): p. 34–43.
2. Brickley, D. and Guha, R., *Resource Description Framework (RDF) Schema Specification*. 2000, World Wide Web Consortium. (URL: http://www.w3.org/TR/2000/CR-rdf-schema-20000327).
3. Carr, L., Bechhofer, S., Goble, C., *et al. Conceptual Linking: Ontology-based Open Hypermedia*. In *Proc of the 10th Intl. WWW Conference*. 2001. Hong-Kong
4. Ciravegna, F. *Adaptive Information Extraction from Text by Rule Induction and Generalisation*. In *Proc. of the 17th Intl. Joint Conference on AI*. 2001. Washington, USA.
5. DAML.org, *Reference description of the DAML+OIL ontology mark-up language*. 2001, (URL: http://www.DAML.org/2001/03/reference.html).
6. Domingue, J., Dzbor, M., and Motta, E., *Semantic Layering with Magpie*, In *Handbook on Ontologies in Information Systems*, (Staab, S. and Studer, R., Editors). 2003, Springer Verlag.
7. Gruber, T.R., *A Translation approach to Portable Ontology Specifications.* Knowledge Acquisition, 1993. **5**(2): p. 199–221.
8. Guarino, N., Masolo, C., and Vetere, G., *OntoSeek: Content-Based Access to the Web.* IEEE Intelligent Systems, 1999. **14**(3): p. 70–80.
9. Handschuh, S., Staab, S., and Maedche, A. *CREAM – Creating relational metadata with a component-based, ontology driven annotation framework*. In *Proc. of the International Semantic Web Working Symposium*. 2001. California, USA.
10. Heflin, J. and Hendler, J., *A Portrait of the Semantic Web in Action.* IEEE Intelligent Systems, 2001. **16**(2): p. 54–59.
11. Heflin, J., Hendler, J., and Luke, S. *Reading Between the Lines: Using SHOE to Discover Implicit Knowledge from the Web*. In *Proc. of the AAAI Workshop on AI and Information Integration*. 1998.
12. Kahan, J., Koivunen, M.-R., Prud'Hommeaux, E., *et al. Annotea: An Open RDF Infrastructure for Shared Web Annotations*. In *Proc of the 10th Intl. WWW Conference*. 2001. Hong-Kong.
13. Lieberman, H., Fry, C., and Weitzman, L., *Exploring the web with reconnaissance Agents.* Communications of the ACM, 2001. **44**(8): p. 69–75.

14. McGuinness, D.L. *Ontological Issues for Knowledge-Enhanced Search.* In *Proc. of the Formal Ontology in Information Systems*. 1998.
15. Middleton, S., DeRoure, D., and Shadbolt, N. *Capturing knowledge of user preferences: Ontologies in recommender systems.* In *Proc. of the ACM K-CAP'01 Conference*. 2001. Victoria, Canada: ACM Press.
16. Motta, E., *Reusable Components for Knowledge Modelling*. Frontiers in AI and Applications. 1997, The Netherlands: IOS Press.
17. Ovsiannikov, I.A., Arbib, M.A., and Mcneill, T.H., *Annotation Technology.* International Journal of Human-Computer Studies, 1999. **50**(4): p. 329–362.
18. Patel-Schneider, P.F., Horrocks, I., and van Harmelen, F., *OWL Web Ontology Language 1.0 Abstract Syntax*. 2002, (URL http://www.w3.org/TR/owl-absyn/).
19. Riva, A. and Ramoni, M., *LispWeb: A Specialised HTTP Server for Distributed AI Applications.* Computer Networks and ISDN Systems, 1996. **28**(7–11): p. 953–961.
20. Sadiq, W. and Kumar, S., *Web Service Description (Usage Scenarios)*. 2002, World Wide Web Consortium, (URL: http://www.w3.org/TR/2002/WD-ws-desc-usecases-20020604).
21. Tauscher, L. and Greenberg, S., *How People Revisit Web Pages: Empirical Findings and Implications for the Design of History Systems.* International Journal of Human Computer Studies, 2001. **47**(1): p. 97–138.
22. Vargas-Vera, M., Motta, E., Domingue, J., *et al. MnM: Ontology Driven Semi-automatic and Automatic Support for Semantic Markup*. In *Proc. of the 13th European Knowledge Acquisition Workshop (EKAW)*. 2002. Spain.