CS AKTiveSpace: Representing Computer Science in the Semantic Web

Nigel R. Shadbolt, monica m.c. schraefel
Nicholas Gibbins, Hugh Glaser, Stephen Harris

Department of Electronics and Computer Science
University of Southampton
Southampton, United Kingdom

{nrs, mc, nmg, hg, swh}@ecs.soton.ac.uk

ABSTRACT

We present a Semantic Web application that we call CS AKTiveSpace[1]. The application exploits a wide range of semantically heterogeneous and distributed content relating to Computer Science research in the UK. This content is gathered on a continuous basis using a variety of methods including harvesting and scraping as well as adopting a range models for content acquisition. The content currently comprises around ten million RDF triples and we have developed storage, retrieval and maintenance methods to support its management. The content is mediated through an ontology constructed for the application domain and incorporates components from other published ontologies. CS AKTiveSpace supports the exploration of patterns and implications inherent in the content and exploits a variety of visualisations and multi dimensional representations. Knowledge services supported in the application include investigating communities of practice: who is working, researching or publishing with whom. This work illustrates a number of substantial challenges for the Semantic Web. These include problems of referential integrity, tractable inference and interaction support. We review our approaches to these issues and discuss relevant related work.

1. INTRODUCTION

Within the Semantic Web community, the field has reached a point where we wish to move from either language definition or the development of individual Semantic Web services towards a test of the larger Semantic Web project. We need to demonstrate the integration of both dynamic content acquisition/delivery and support services. There are numerous aspects to pulling together such a demonstrator application: how the content will be harvested and stored; there are referential integrity problems to address; queries over the space must be time efficient in returning results, and be robust in order to scale; services must communicate with each other and the content; the interaction and visualization must afford both an effective model of the aggregated content as well as a means to interrogate the content meaningfully.

CS AKTiveSpace represents our effort to consider these issues directly in the context of an integrated semantic web application. It attempts to provide an overview of current UK University based research in Computer Science. The application exploits a wide range of semantically heterogeneous and distributed content relating to Computer Science research in the UK. It provides services such as browsing topics and institutions for researchers, it can show the geographic range and extent of where a topic is researched, provides an estimation of "top" researchers in a topic and by geographic region, is able to calculate a researcher's Community of Practice. We chose this area for a number of reasons; *(i)* we had a real interest in having such a set of services, *(ii)* it is a domain that we understand, *(iii)* it is relatively accessible and easy to communicate as a domain, *(iv)* we were able to secure access to a wide range of content that was not subject to industrial embargo, *(v)* it presented real challenges of scale and scope.

The application exploits a wide range of semantically heterogeneous and distributed content relating to Computer Science research in the UK. For example, there are almost 2000 research active Computer Science faculty, there are 24,000 research projects represented, many thousands of papers, hundreds of distinct research groups.

This content is gathered on a continuous basis using a variety of methods including harvesting and scraping [17] as well as other models for content acquisition. The content currently comprises around ten million RDF triples and we have developed storage, retrieval and maintenance methods to support its management [12]. The content is mediated through an ontology [3] constructed for the application domain and incorporates components from other published ontologies [18].

CS AKTiveSpace supports the exploration of patterns and implications inherent in the content. It exploits a variety of visualisations and multi dimensional representations that are designed to make content exploration, navigation and appreciation direct and intuitive [23]. As mentioned the knowledge services supported in the application include investigating communities of practice [5] and scholarly impact [16].

This application which won the 2003 Semantic Web Challenge [1], illustrates a number of substantial issues for the Semantic Web. There are issues to do with how to best sustain an acquisition and harvesting activity. There are decisions about how best to model the harvested content; how to cope with the fact that there are bound to be large numbers of duplicate items that need to be recognised as referring to the same objects or referents; the degree to which our content presents opportunities in terms of the sorts of inferential services we can support; how we present the content so that

[1]The application is viewable on the web at `http://triplestore.aktors.org/SemanticWebChallenge/`

WWW2004, May 17–22, 2004, New York, NY USA.
ACM xxx.xxx.

inherent patterns and trends can be directly discerned must be considered; how all this information is to be maintained and sustained as a community exercise is also an issue that needs to be addressed if the Semantic Web is to become a reality.

In this paper, we present the motivating scenario for CS AKTiveSpace that has provided a context for integration. We next describe the components of CS AKTiveSpace and the processes that can be run as a result. We describe the interaction design approach we use to integrate these components at the user interface (UI) level. We review related work and discuss the particular research challenges presented by the scenario, and our results to date.

2. KNOWLEDGE ACQUISITION

2.1 The role of ontologies

Even in a distributed environment like the Semantic Web, the physical distribution of data is much less important than the semantic distribution of data brought about by the use of disparate ontologies for the same application domain. For the purposes of this application we have used a single common ontology to express the data which drives the CS AKTiveSpace. We use this common ontology, the AKT Reference Ontology [3], to mediate and guide the integration of the different data sources.

When expressed in terms of our ontology, the RDF data obtained from these sources are made publicly available through the hyphen.info web site (Hyphen being the name for the knowledge acquisition effort) and cached in an RDF triplestore [12] which provides the necessary query and inferential capabilities on which CS AKTiveSpace is built.

Ontology mapping is seen as a key challenge in semantic web applications [15] We have developed a method and a theory for ontology mapping and merging. The approach draws heavily on proven theoretical work but our work goes further in providing a systematic approach for ontology mapping with precise methodological steps. In particular, our method, Information-Flow based Ontology Mapping (IF-Map) [14], draws on the proven theoretical ground of Information Flow and channel theory [6], and provides a systematic and mechanised way for deploying the approach in a distributed environment to perform ontology mapping among a variety of different ontologies.

The IF-Map system formalizes mappings of ontology constructs in terms of logic infomorphisms, the fundamental ingredient of Information Flow. These are well suited for representing the bi-directional relation of types and tokens, which corresponds to concepts and instances in the ontology realm. IF-Map is focusing on instances and how these are classified against ontology concepts. This reveals the operational semantics that the ontology's community has chosen by virtue of how it uses its instances. The IF-Map algorithm makes use of this information in order to map onto related concepts from another ontology with which its concepts classify the same instances.

In this version of CS AKTiveSpace we have not included this ontology mapping capability since we have been responsible for engineering the mapping of the heterogeneous information content. In future it is likely that as we move to a push model of information provision we should provide the means to have local variants of ontologies mapping into our AKT computer science "standard reference" ontology.

2.2 Mediators

Due to the wide variety of the data sources that we use, we have found it necessary to invest a degree of effort in developing individual mediators for each of our data sources that recast these sources in terms of our ontology. These mediators range from specialized database export scripts to XML transformation tools [17] that have been trained to extract the required content from semi-structured web pages. Although these mediators are based on a common framework of code (which handles the rote work of database access, HTTP retrieval, RDF construction and the more common patterns in our ontology, such as date and time expression), they each contain specialized capabilities that are tailored to the content and nature of the individual data sources. While the bulk translation of instance data by such a mediator is straightforward, our use of these mediators has shown that the mapping of existing structured and semi-structured data at the schema/ontology level is not a task that can be effectively automated in all cases; the investment of effort in building mediators for our common ontology is reflected in the consequent perceived value of the knowledge base to which they contribute.

2.3 Exploitation of Distributed Content

The CS AKTiveSpace application requires that a range of content be available for use by the system. As it stands, some of this content already exists in suitable structured forms, while others do not. We adopt a pragmatic attitude that reflects that fact that although the content that we are gathering is the prime mover that drives the interface, we should also be tolerant of inconsistencies in that content. We use a relatively scruffy approach in which we make the immediate best use of the available data sources, perhaps in an imperfect fashion, while anticipating that we will be able to make better use of them in future. Although this comes at a cost (there is an implicit commitment to future knowledge maintenance), such early exploitation of available content is necessary to initiate a community process that should be self-sustaining in the future and so justify the investment of effort.

2.4 Models for knowledge acquisition

We employ both push and pull models of knowledge acquisition, where push and pull refer primarily to whether the publisher or consumer are responsible for translating the data into a form which is suitable for the consumer. The push model involves a data source (the publisher) choosing to express its data in terms of the ontology used by the CS AKTiveSpace. The publisher is solely responsible for the translation, so the consumer may simply retrieve the translated knowledge base without any further effort required on its part. In comparison, the pull model requires that the consumer takes a raw data source (which may be published against some other ontology, or which may only exist as a set of unannotated webpages) and construct a knowledge base from that data source.

In some ways, the pull model has advantages over the push model, in that the consumer has a much greater level of control over what information is encoded within the resulting knowledge bases and is better placed to be able to correct inconsistencies or to adapt to changes in the underlying common ontology, but this comes at the expense of a greater cost to the consumer, both in the acquisition phase of the knowledge lifecycle (when a new data source is acquired), but particularly in the maintenance phase.

We use the pull model predominantly for large, comparatively static data sources (for example, the list of countries and administrative regions given by ISO3166), and as an interim solution for high-value data sources that are of general interest to the community (for example, the Engineering and Physical Sciences Research Council's database of research funding) as a means to 'pump-prime' the system with sufficient data to encourage other members of the community to participate by offering to push their local data sources to us (the viral, rich-get-richer phenomenon that we later describe

needs to start somewhere). In the longer term, we aim to encourage the owners of the majority of these pull model sources to move to a push model of delivery.

The hyphen metadata gathered by these mediators consists of 430MB of RDF/XML files containing around 10 million RDF triples describing 800,000 instances of people, places, publications and other items of interest to the academic community.

3. MAINTAINANCE

3.1 Referential Integrity in the Semantic Web

The current development of knowledge services on the Semantic Web raises a number of issues which are not commonly encountered in existing knowledge based systems, and which pertain to the distribution of knowledge and the difficulty of obtaining agreement on a conceptualisation in a distributed environment when there is no ultimate authority. One such issue is that of coreference, which arises when more than one Uniform Resource Identifier is used to refer to a given resource, and which causes particular problems when statements from different knowledge bases are to be combined.

While the semantics of the URIs used as names by the Semantic Web require that identical names must refer to the same entity, they do not require the unique names assumption. In the absence of any coordination between data sources, it is reasonable to assume that in most cases the unique names assumption holds within the domain of a single data source, but that it does not hold across the union of several data sources. Two data sources on the Semantic Web may contain statements that refer to the same resource, but by different URIs. When constructing a system like the CS AKTiveSpace which uses knowledge from different sources, we must be able to integrate the contents of these sources by mapping the URIs used by one source onto the URIs used by another.

In addition, Semantic Web resources are not necessarily digital artifacts like web pages which can be retrieved by resolving the URI which names that resource; URIs can also denote resources which are physical entities, such as people or organizations. Since URIs need not denote digital objects, it is not possible to resolve them and determine if two URIs are coreferent by comparing the objects that they denote. In fact, it is current common practice for Semantic Web agents to treat URIs as opaque symbols and to ignore any retrieval semantics that might otherwise be associated with them (for example, the structure of an HTTP URI gives the necessary means for an agent to be able to retrieve the object indicated by that URI).

Unidentified coreferent entities present a problem for Semantic Web applications since they partition the information space in such a way as to reduce the recall of queries made in that space. Even if a SW application is able to use heterogeneous data sources by mediating them through a common ontology, it cannot be properly considered to have integrated those data sources if unnecessary coreferences exist for a resource that is described in more than one data source.

Within the context of a community resource like the CS AKTiveSpace, the most appropriate – and the simplest – solution to this issue is predominantly social in nature. The emerging knowledge base that is being collaboratively built by the community may be viewed as a gazetteer or name authority containing the agreed names that should be used to refer to entities in the application domain. It is here that the careful selection of data sources in the initial knowledge acquisition phase is important, since the resources that they describe are likely to be referred to in many other sources.

We have complemented this social process where necessary by the use of heuristic techniques for identifying coreferent entities [4] and then coalescing them (through the use of `owl:sameAs` assertions). By keeping the equality between entities as explicit assertions, rather than by making it implicit by rewriting gathered information to use canonical URIs for objects, we provide a means to roll back coreference resolution (the process of identification is defeasible)

In addition to the purely automatic techniques, we have also developed a coreference editor which allows a user to vet the potential coreferent entities in a semi-automatic manner. The entities are identified using the same heuristic techniques mentioned above, but the user is presented with the information about both entities and asked to judge whether the assertion of equality is correct or not.

4. SERVICES

The core of the CS AKTiveSpace system is a series of – mostly HTTP – services that collaborate to provide the knowledge capabilities required by the user interface.

Currently the services that comprise CS AKTiveSpace are manually connected, using hard coded URLs to refer to the servies. We would prefer a system in which the services are bound dynamically using service discovery techniques, to reduce brittleness and make the system less dependent on specific instances of services. The service level characterisation of this system, describing the interactions we would like to support using srevice discoverey has be discussed in an AKT Project technical report[11], and will be covered in the future work section.

4.1 The Services

3store

The ability to store and query the rapidly changing hyphen.info data lead to the requirement for an RDF triplestore capable of storing RDF data on the order of 25 million triples (to allow for planned expansion), and updating around 10% of that daily, in a reasonable time.

To meet these requirements the RDF data is stored in a 3store server. 3store is a Free Software (GNU General Public Licence [9]) RDF triplestore that stores RDF triples in a SQL database, using an ontology independent schema for flexibility, scalability and efficiency reasons. The architecture is similar to the ones used in HYWIBAS [19] and SOPHIA [2], though with a greater emphasis on query execution speed and with an underlying RDF representation replacing the frame logic representation [12].

The server offers an RDQL [13] service that is used by the user interface and many of the other services. Typical RDQL queries issued by the interface are answered in a few milliseconds (depending on network latency), and RDF metadata files can be asserted at a rate of around 1000 triples/second on commodity 80x86 Linux servers. This allows the interface to query and the KB to be updated interactively.

3store browser

In addition to its query service, 3store also provides a browser interface which enables users to ‘drill down’ and examine the entities identified by the CS AKTiveSpace explorer in greater depth. Although this interface is ontology- and application-neutral, it has a number of features which give users the means to use the information presented in CS AKTiveSpace.

The browser can explicitly indicate the origins of the statements which have been made about the entity being browsed (alternatively, the context in which those statements were made), which provide provenance information about the statements (where they

came from, how they were obtained, when they were gathered). The browser also provides application-level support for equivalence assertions between entities; if different sources refer to an entity by different names (URIs), and this correspondence between names is known, the browser presents a unified view across the different instances of that entity.

This 'one size fits all' approach is not appropriate for all applications, however, so 3store also provides facilities for the (application -specific) dynamic generation of webpages based on the instantiation of a webpage template by the results of a query.

Geographic Visualizer

This service was developed specifically for this application and consists of a map with which the user may interact. It provides two distinct subsidiary services to the application as a whole.

1. The map may be used as an output display. If the results of a query have a geographical basis, the locations of the returned elements (for example, universities and research institutions) are marked on the map.

2. The map may be used as an input controller, in which queries may be constrained by selecting regions on the map.

These behaviours and information affordances are discussed further in section 5 below.

Armadillo

Armadillo [8] is a service for on-the-fly, user-determined, directed knowledge acquisition from web pages, which can be used to opportunistically expand the knowledge base.

The service uses triplestore queries to determine what knowledge is already asserted regarding the instance to be populated, and to help resolve referential integrity issues [4]. This pre-existing knowledge is then used to inform natural language searches (over a variety of web sources) that extract further knowledge and assert it into the triplestore. In this way, future requests for information about the instance in question will potentially return more information.

Ontocopi

Ontocopi is a Community of Practice [5] analysis service that identifies implicit communities by finding sets of instances associated with a selected instance in a knowledge base. Through the user interface, a user may request more detailed information on an individual, including their community of practice.

Ontocopi uses Ontological Network Analysis (ONA) to discover connections between the objects that the ontology only implicitly represents. For example, the tool can discover that two people have similar patterns of interaction, work with similar people, go to the same conferences, or publish in the same journals.

5. INTERACTION DESIGN

The main interaction paradigms of the web are browsing and searching. Both approaches restrict users to exploring discrete instances of information (i.e. Web pages) without a strong semantic sense of their association. Search results which produce pages of links create an implicit association among the pages, insofar as the returned pages contain the words given, but such an association can be distinct from a person's context informing the choice of those terms. While some projects have attempted to derive the semantic relevance of discrete search results, at least sufficiently to be able to group them into derived categories after the fact [24],

Figure 1: CS AKTiveSpace UI featuring multicolumn queries and geographical visualizer and constraint controller

the unstructured nature of the Web makes exploring relationships among pages, or the information components within pages difficult to determine. The value of being able to explore a domain rather than only query it, however, is well known [21]. Each approach affords complementary but distinct approaches to knowledge building. While search results focus on retrieving a specified set of instances, exploration affords discovery of the context of information and the relations among various points of contact of an instance within the rest of the domain. With the Semantic Web's, we have the opportunity to leverage its ontological underpinnings in order to expose and make explorable the relations within a represented domain space. In CS AKTiveSpace, we have foregrounded exploration as the core approach for interogating the computer science domain.

One of the major disincentives when dealing with complex semantic information systems is dealing with lengthy, compound query representations. One of the advantages of the CAS UI is that there is a systematic mapping between the direct manipulation of the interface and the dispatching of complex RDQL queries. Indeed, our approach in CS AKTiveSpace has been to create a UI that affords real-time, direct manipulation [7] of a set of meaningful visual queries [20] of a privileged set of attributes in the domain represented by our ontology. Each query answer is then situated within the context of the results. Indeed maintaining context is a critical aspect of exploration: to explore information within the context of its discovery. In the remainder of this section, we describe how exploration is enabled in the CS AKTiveSpace UI.

5.1 Domain Exploration

5.1.1 Selection and Detail View

The top half of the CS AKTiveSpace UI features a set of columns. The labels on the columns represent a set of queries against the ontology. In the example given below, the columns represent < Region | Reserach Area | Reserchers >. The selection in the column to the left acts as a constraint on the results of the column to the right. When the user makes a selection of one (or more) of the instances in a column, the Detail View beneath the columns provides further information about the selected instance. The combination of instance context, provided by seeing a selected instance in the

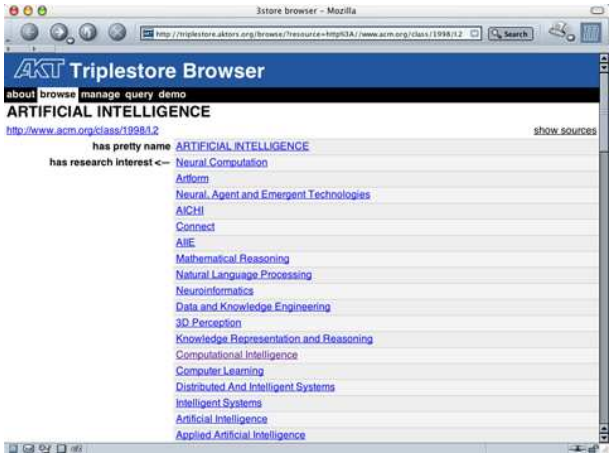

Figure 2: The 3store browser showing the rendering of an RDF instance for a researcher

context of other associated instances, along with detail about that selected instance, provides users with a fast visual means to interogate relations among data. The representation of these queries visually means that they can be reformulated with new data quickly, simply by selecting different instances within a column. The detail view along with the instance selection context of the surrounding instances in the column view also means that users do not have to leave the context of the CS AKTiveSpace window in order to gain at least two kinds of information about a particular instance: breadth and depth. Context in the column view provides a persistent view of alternate instances which match the given criteria for the current set of date presented. The proximity of one instance to another may provide new information about that part of the domain which might spark a new query in itself, to explore in more detail, for instance, how the two instances are related. This awareness can expand the breadth of understanding of a domain. Context is represented persistently via the spatial layout of the multicolumn view. Persistence means that users can focus their energies on knowledge building with the information discovered, rather than on splitting their attention, and hence increasing cognitive load, between remembering which instance was next to another while focusing on the detail of a current selection. Similarly, the detail view provides specific information for the selected instance, enhancing the depth of knowledge about the instance, beyond the label value provided in the column view. The CS AKTiveSpace UI which includes both columns and detail view represents a way of supporting both focus and context [21] by supporting both users contextual exploration of the domain through rapid selections of instances within columns, and users' focus on an instance, *within* the domain context of that selection.

Within the Detail View area as well, we have embedded a "Browse" button. By selecting browse at any point, one is taken to the sources from the triplestore which inform the results presented in the detail view.

With this view into the triplestore sources, users are free to interogate the provenance of any of the data represented in the main UI: the browse view has an option to view sources of the triples. This view presents the source URL of a given triple as well as the date it was harvested.

For the most part, the Browse Sources has been provided as a way to debug the data represented by those of us working on the CS AKTiveSpace project. The tool is also, however, an effective way for any user to check the integrety of the data and to explore the assumptions informing the representation's construction. This gives us a mechanism to better support users' correction of data. By making these views available, they can for instance, tell us that the URL that has given a particular set of address info is incorrect or out of date. Likewise, they can point us to alternative sources that our harvesting methods, described above, may not have yet discovered.

5.1.2 Inter-Column Sorting

The selected set of columns can be though of as representing one particular slice through the domain's multidimensional space. The default view in CS AKTiveSpace is < Research Area | Geographical Region | Person >. The columns, as described above, reflect a hierarchy where the column to the right is constrained by the column on the left. Thus, the selection of a research area in the first column constrains which geographical areas will be highlighted: only those locations which have departments with research being carried out in the selected topic. If a user is interested in all the research being carried out in southern England, with this configuration of the UI, the person would need to select each topic in the first column and perform an inspection to see which univeristies that result are listed in the south. It would be far simpler to be able to rearrange the columns to support geographical region first, and then get the list of research areas next. Interestingly, few category-based Web indexes, from Yahoo to Amazon support this simple mechanism to allow a user to reorient a hierarchy based on their locus of interest. Beyond supporting more effective structuring of a query, this kind of inter-column sorting allows the user to interogate the data at any time from mulitple perspectives in order to see how the information changes when recontextualized.

Inter-column sorting is distinct from ordering a set database results, for instance, which have various fields in them. For instance, when selecting southern England, a list of universites with CS departments in that region in populated into the Detail View. These results could be potentially ordered in a variety of ways: by region, alphabetically, numerically by phone number and so on. The result set from the initial query remains the same. By reorganizing hierarchies, the search result sets will almost always change, the results of which can be, as we suggest above, informative in itself.

5.1.3 Column Filters: Query Refinements

Part of the goal of CS AKTiveSpace is to make transparent the operations which produce a given result. In the researcher field, for example, a number of parameters can be selected to determine not only who, but how many who's show up in that column. The fist parameter is labelled "top". Users can select to view the top 5, 10, 15 or just all researchers matching a given criteria from the previous columns.

The second parameter, RAE or Grant Total, represents how "topness" is currently calculated: either by ranking in a national review assessment exercise, the criteria for which are well known in the modeled community of UK researchers, or by funding total from the national granting council for the sciences.

Neither of these metrics is perfect, and many may argue that neither in itself is a holistic measure of top-ness. However, rather than hide the parameters of the query that support topness, we expose them. This way, users can make their own assessments of the rankings. Indeed, the comparison between which researchers appear depending on which metric is selected is revealing in itself.

5.2 Example Walk-Through

In order to contextualize the above interaction description, we

present a brief walkthrough of the interaction pictured of CS AKTiveSpace in figure 1, above.

In the state pictured, the user has re-sorted the column views from the default arrangement of < Research Area | Region | Researcher > to < Region | Research Area | Researcher >. The Region column shows that a 200 mile reticule has been selected in the map view, and the reticule has been dragged over part of the map representing southern England. From the list of research areas that shows up in the Research Area column, artificial intelligence has been selected. With the constraints on Researcher set as Top 5, determined by Grant Total, the list of 5 people in AI with the greatest grant totals are represented. One of the researcher names has then been selected. The Detail View shows that that researcher is the current selection: it is populated with information about the researcher, including full name, contact information, URL and list of significant papers on the left two-thirds of the view. In the right remaining third, the list of the researcher's community of practice is generated automatically by the CoP service described in section 4.1 above. Finally, at the bottom of the CS AKTiveSpace window the user can activate the Armadilo service by clicking the "Run Armadilo" service button. As described in section 4.1 above, Armadilo will search for aditional information about this reseasrcher to supplement the information already present in the triplestore. Browsing the sources for the selected researcher will show which sources have been found via an Armadilo session.

The final state of the UI shows both the context of discovery for the current selected information, as well as a look at some of the detailed information available in the triplestore and associated with the researcher.

5.3 CS AKTiveSpace extensions

5.3.1 Swapping, Addition, Subtraction

The current CS AKTiveSpace application is a first instantiation of a semantic web-based user interface that exploits the properties of the domain ontology in order to support context/exploration of the domain "out of the box". The interaction properties of spatial layout, inter-column sorting and detail view are based on the mSpace interaction model [22] to support flexible exploration of a domain. There are interactions within the mSpace model that we have yet to implement, most particularly, column-swapping, and column addition/subtraction. These attributes and their formal representations are described in detail in [10], but in brief, column swapping allows a user to replace a given column (effectively a class expression) with another column not already present in the current slice. One may wish to replace Researcher in the above CS AKTiveSpace examples with Projects. Similarly, with column addition/subtraction one can expand or contract the slice through the domain underconsideration. For instance, one may wish to expand the current CS AKTiveSpace slice with Projects to follow Researchers.

5.3.2 Second Level Zoom

The Detail view in the current CS AKTiveSpace application supports what may be called a first level zoom in on the current selection: the selected label is expanded to reveal more detail about it. We are looking at how to support second level zoom, in other words, to increase the detail about a current view in the detail view area. For instance, selecting the name of the researcher in the detail view might open up another view of the researcher which includes more information available in the triple store, such as list of research projects and their descriptions, graduate students under the researcher's supervision. Ideally, this second level zoom could produce a CV on demand, where it parameters could be tweaked to generate appropriate versions for distinct contexts.

5.4 CS AKTiveSpace summary

CS AKTiveSpace represents a light-weight UI for exploring an ontology-modelled domain. By leveraging the semantic affordances of the domain as spatially presented visual queries, it disguises the complexity of the sources, services and queries running real-time beneath the interface. Hiding the complexity of the processes beneath a clear UI is exactly what we designed the CS AKTiveSpace application to do: our target users who wish to explore both the relations and details of the state of research in computer science in the UK do not necessarily want to know how many heterogeneous sources have gone into making up the result before them: they want the information, and based on that information to build up actionable knowledge.

6. RELATED WORK

A number of applications on the Semantic Web (and its precursors) have informed the development of the CS AKTiveSpace and its ancillary technologies:

6.1 KA^2 and Ontobroker

The KA^2 Knowledge Acquisition Community Ontology was an early example of a ontology-driven website designed to support a community. The content collected for the communal knowledge base was gathered almost entirely by the push model; members of the community were encouraged to annotate their web pages with information that would be collected and collated to form the resulting knowledge base. From the point of view of the knowledge acquisition process, the most important difference between KA^2 and CS AKTiveSpace lies in the granularity of data sources. In KA^2, the use of annotated web pages as data sources yields a large number of small sources that must be maintained individually, although Staab et al do note that this is a time consuming and costly task. The KA^2 portal adopts a task-neutral approach to their interface, as opposed to the task-specific domain exploration that we espouse in the CS AKTiveSpace; in the basic KA^2 interface, queries are permitted on six axes (classes of object) with constraints from up to three axes applied conjunctively to the query.

While relatively successful as a proof of concept, the main failing of KA^2 as a community endeavour was one of scale. The section of the knowledge acquisition community that took part was arguably too small for the effort to achieve critical mass and become self-sustaining, because the projected reward for participation (namely having consistent information to which one would not normally have access) was outweighed by the cost of participation in the short to medium term. In our development of CS AKTiveSpace, we have tried to harness economies of scale and progress past this initial hurdle by choosing a broader subject base (so that there is a greater likelihood that users will encounter information of which they were not previously aware, thus a greater reward), and by investing in the construction of an initial high-value knowledge base that provides an immediate reward to users and an incentive for further engagement.

6.2 SHOE

As with KA^2, SHOE relies on users annotating their web pages with relevant semantic markup that is harvested and aggregated in a communal knowledge base, although tools are provided for knowledge extraction from specific sources (e.g. CiteSeer). As a family of technologies rather than an application for a specific domain, SHOE supports the use of multiple ontologies, which is one of its

particular strengths, but does not use the same knowledge representation formalism as current Semantic Web languages. The primary interface to the knowledge bases constructed from SHOE markup is the Semantic Search query tool, which although ontology-neutral, requires the user to specify the necessary characteristics (property values) of the objects to be returned by the search (the existence of more domain- and task-specific interfaces is mentioned, but not elaborated upon). In comparison, the CS AKTiveSpace interface described in this paper aims to provide a user with a more flexible mode of interaction that eliminates the need to explicitly state the constraints on their query in this way.

6.3 Ontoportal

The Ontoportal project is an ontological hypermedia system that provides a navigable interface to a knowledge base expressed in a given ontology. The data used in an Ontoportal system is typically (but not necessarily) human-authored, and is presented as a browser interface. However, the content behind this hypermedia interface is static; the underlying infrastructure does not support query processing or inference, which results in a fixed page structure that provides only a single visualisation of the knowledge within. Our approach with CS AKTiveSpace has been to provide task-specific visualisations of our gathered content, leveraged by the use of an RDF triplestore with suitable query and inferential capabilities.

6.4 S-CREAM

S-CREAM is a framework for creating metadata by applying annotations to web pages. S-CREAM avoids the maintenance bottleneck associated with semantic annotation by employing natural language information extraction techniques to semi-automate the annotation of documents. The tool used for this, Amilcare, produces a discourse representation for a document identifying the referents within the document that is then transformed into a set of annotations made against a given ontology. While the design of S-CREAM does automate much of the process of document annotation, the definition of the mapping from the structures in the discourse representation of the document to the ontological structures in the intended annotation is carried out manually, as are the equivalent mapping rules in our mediators.

6.5 GIS Systems

There are a number of Geographic Information System (GIS) based visual query systems that are superficially similar to CS AKTiveSpace, with geographic range expression, there are however a number of fundamental differences, GIS database systems are often capable of resolving sophisticated geo-spatial queries based on topological expressions, while the CS AKTiveSpace back-end has only basic numerical comparison support, as this is not a development focus. Conversely, GIS databases do not have the inferential or semantic representation capabilities of a knowledge base, as they are mainly concerned with conventional database retrieval. Equally their visual user interfaces are generally designed for query construction, rather than domain exploration. Recent developments in this area include an integration of some knowledge base features, and ontologies have been used for some time for data integration.

7. CONCLUSIONS AND FUTURE WORK

It is our impression following the development of this interface that the inferential capability of RDF, combined with the structure provided by the ontology allows for comparatively simple queries to perform tasks that would require substantially more complex queries in an RDBMS system.

The CS AKTiveSpace is offering a range of services, significant integration, and flexible interaction. We are dealing with millions of triples and yet it is still not enough. Despite our successful harvesting of the web and exploitation of various organizations' RDF, we still cannot reflect the richness of the queries our system can support because we do not always have enough content.

We have a strong proof of concept, but to get to the point where CS AKTiveSpace becomes a meaningful application we need the content that we cannot get without participation. We need individuals and/or organizations to publish RDF content either directly against our ontology or else against an ontology we can translate. Participation cannot usually be enforced. Users have to see very strong benefits for the effort of publishing their content against our ontology.

We have started to see that happen within the UK CS community, and more recently, the eScience community. Interestingly, it is not simply the power of the back end tools that has provoked these enquiries and requests to apply CS AKTiveSpace to other domains; it has been the user interaction. There is an immediate appeal to the fact that patterns and gestalts, particular and general content exploration can be so rapidly effected.

Many of us will have been frustrated by the fact that so often individuals and organizations, our managers and colleagues ask us for the same basic content in different formats and configurations. One of the attractions of mediating content through shared ontologies is that re-presentation and repurposing of content becomes so much easier. A huge collective effort is made in the UK every few years to collect content about the state of research in Higher Education. This so-called Research Assessment Exercise is important since the content collected goes on to form the basis of rankings of Department to particular grades. These grades are direct drivers on the funding departments receive. The clerical effort of collecting and make available this content is many hundreds of person years. Much of this effort is the equivalent of the semantic clerk – someone who has to work out how to integrate the various content into the formats required for the particular exercise. These sorts of requirement can provide an opportunity to convince communities and the organizations demanding the content that that for a little investment in Semantic Web engineering and ontology construction they could dramatically reduce the cost of repeated content collection.

One other issue, behind the content and behind the UI is service integration. One of the ways we have been dealing with the issue in AKT is, for the time being, ignore service discovery and focus on service integration with AKT partner technologies. Partners have worked to integrate their services with the central triplestore. What this means is that CS AKTiveSpace users have a variety of services they can invoke that we have evaluated as relevant to working with CS AKTiveSpace. Through the process of working through these integration problems we plan to have something to report about how these specific integration efforts can be generalized towards a loose protocol so that integration of dynamically discovered/requested services can be supported.

In the future we hope to include more component services. All the services have to be able to do is interrogate the triplestore and be able to take advantage of the ontological structures that provide our common models for content integration. We already have web services that can classify our research papers against our research area ontology. We are able to invoke a natural language learning service that is able to detect ontologically significant parts of texts and abstracts – titles, topics, authors and general term entity recognition. We are incorporating a dynamic link service and intend to provide a range of e-scholar services such as detecting parts of the research area ontology that are rich in publications with various

types of impact factor. One ambition is that in future RAE's the panels will not just have standard bibliometric evidence, number of citations and impact factors of journals. With a system like CS AKTiveSpace we will be able to engage in semiometrics – looking at a whole range of measures and content implicit in the Conventional Web and meta content explicit in the Semantic Web to build much richer views as to the real research activity underway in our discipline.

8. ACKNOWLEDGMENTS

This work was supported by the Advanced Knowledge Technologies (AKT) Interdisciplinary Research Collaboration (IRC). The AKT IRC is sponsored by the UK Engineering and Physical Sciences Research Council under grant number GR/N15764/01 and comprises the Universities of Aberdeen, Edinburgh, Sheffield, Southampton and the Open University.

9. REFERENCES

[1] *Semantic Web Challenge*, `http://challenge.semanticweb.org/`, 2003.

[2] N.F. Abernethy and R.B. Altman, *Sophia: Providing basic knowledge services with a common DBMS*, Proceedings of the 5th International Workshop on Knowledge Represenation Meets Databases (KRDB '98): Innovative Application Programming and Query Interfaces (1998).

[3] *The AKT Reference Ontology*, `http://www.aktors.org/publications/ontology/`, 2002.

[4] Harith Alani, Srinandan Dasmahapatra, Nicholas Gibbins, Hugh Glaser, Steve Harris, Yannis Kalfoglou, Kieron O'Hara, and Nigel Shadbolt, *Managing reference: Ensuring referential integrity of ontologies for the semantic web*, Proceedings 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW'02), 2002, pp. 317–334.

[5] Harith Alani, Srinandan Dasmahapatra, Kieron O'Hara, and Nigel Shadbolt, *Identifying communities of practice through ontology network analysis*, IEEE Intelligent Systems **18(2)** (2003), 18–25, `http://eprints.aktors.org/archive/00000172/`.

[6] J. Barwise and J. Seligman, *Information flow: The logic of distributed systems*, 1997.

[7] R. Chimera and Shneiderman, *An exploratory evaluation of three interfaces for browsing large hierarchical tables of contents*, ACM Transactions on Information Systems (TOIS) (1994).

[8] Alexiei Dingli, Fabio Ciravegna, and Yorick Wilks, *Automatic semantic annotation using unsupervised information extraction and integration*, Proceedings of SemAnnot 2003 Workshop, 2003.

[9] Free Software Foundation, *GNU General Public License*, `http://www.gnu.org/licenses/gpl.txt`, 1991.

[10] Nicholas Gibbins, Stephen Harris, and monica schraefel, *Applying mspace interfaces to the semantic web*, preprint: `http://triplestore.aktors.org/tmp/www2004-mspace-model.pdf`, 2003.

[11] Stephen Harris, Nicholas Gibbins, Hugh Glaser, and monica schraefel, *Requirements for an integrated semantic web services application*, Tech. report, AKT Project, November 2003, ECSTR-IAM03-006.

[12] Steve Harris and Nicholas Gibbins, *3store: Efficient bulk RDF storage*, Proceedings of the 1st International Workshop on Practical and Scalable Semantic Systems (PSSS'03), 2003, `http://eprints.aktors.org/archive/00000273/`, pp. 1–20.

[13] Hewlett-Packard Labs, *RDQL - RDF data query language*, `http://www.hpl.hp.com/semweb/rdql.htm`, 2003.

[14] Y. Kalfoglou and M. Schorlemmer, *If-map: An ontology-mapping method based on information-flow theory*, Journal of Data Semantics (2003).

[15] ———, *Ontology mapping: the state of the art*, The Knowledge Engineering Review **18(2)** (2003).

[16] Simon Kampa, *Who are the experts? e-scholars in the semantic web*, PhD Thesis. University of Southampton (2002).

[17] Thomas Leonard and Hugh Glaser, *Large scale acquisition and maintenance from the web without source access*, Workshop 4, Knowledge Markup and Semantic Annotation, K-CAP 2001, Oct 2001, pp. 97–101.

[18] Ian Niles and Adam Pease, *Towards a standard upper ontology*, 2001, `http://citeseer.nj.nec.com/niles01towards.html`.

[19] M.C. Norrie, U. Reimer, P. Lippuner, M. Rhys, and H.J. Schek, *Frames, objects and relations: Three semantic levels for knowledge base systems*, Reasoning About Structured Objects: Knowledge Representation Meets Databases. In 1st Workshop KRDB'94 (1994), 20–22.

[20] C. Plaisant, B. Shneiderman, K. Doan, and T Bruns, *Interface and data architecture for query preview in networked information systems*, ACM Transactions on Information Systems (TOIS) **17(3)** (1999), 320–341.

[21] D. Schaffer, Z. Zuo, S. Greenberg, L. Bartram, J. Dill, S. Dubs, Roseman, and N. Navigating M., *Navigating hierarchically clustered networks through fisheye and full-zoom methods*, ACM TOCHI (1996), 162–188.

[22] m.c. schraefel, M. Karam, and S. Zhao, *mSpace: Interaction design for user-determined, adaptable domain exploration in hypermedia*, 2003, pp. 217–235.

[23] monica schraefel, Maria Karam, and Shengdong Zhao, *mspace: interaction design for user-determined, adaptable domain exploration in hypermedia*, Proceedings of the AH2003 Workshop on Adaptive Hypermedia and Adaptive Web-Based Systems, June 2003, pp. 217–229.

[24] Vivisimo, *the vivisimo document clustering engine*, `http://www.vivisimo.com/`, 2002.