“Wholly new forms of encyclopedias will appear,
ready-made with a mesh of associative trails running through them” – V. Bush

The Evolving mSpace Platform: Leveraging the Semantic Web on the Trail of the Memex

mc schraefel, Daniel A. Smith, Alisdair Owens, Alistair Russell, Craig Harris, Max Wilson
IAM Group, Electronics and Computer Science
University of Southampton
Southampton UK

info @ mspace.fm   http://mspace.fm

ABSTRACT
Vannevar Bush proposed the memex as a means to support building knowledge in the way he says the human brain works: by association. Achieving this vision has been a core motivation for hypertext research. In this paper, we suggest first that Bush’s memex reflects an interaction paradigm rather than system design. Second, we propose that Semantic Web promises to provide the mechanisms to enable these interaction requirements. Third, we propose the mSpace framework and architecture as a platform to deploy lightweight Semantic Web applications which foreground associative interaction. We propose this lightweight approach as a means to evaluate both interaction needs and the cost/benefits of using Semantic Web technologies to support them.

Categories and Subject Descriptors
H.5.4 [**Information Systems**]: Hypertext/Hypermedia – *architectures, navigation, user issues.*

General Terms
Design, Experimentation, Human Factors

Keywords
mSpace, hypertext, Bush, association, Semantic Web, memex

1. INTRODUCTION
There is a paradox to Web-based digital information systems: when material is digitized it has the potential to become simultaneously both more accessible and more invisible than it ever was in its non-digital form. Digitized, its bits can be transmitted effectively anywhere; digitized, it slips into the sea of zeros and ones, indistinguishable. It depends utterly on both the cues associated with it, and how those cues are represented in order for it to be retrievable.

So far, the killer app of the Web has been the search engine: it lets people search the content of the thing itself via keywords to get at that thing directly. From the list of links returned by a search engine, with its ever-improving algorithms for finding the most likely match to a query, a person can trawl through increasingly fewer dud links for the best match to their interest. In many ways, the success of the search engine has shaped our expectations of the Web. Modern browsers (a potentially anachronistic term) reflect this: the “Google box” is either the default part of the toolbar or is installed as a popular plug-in addition. But is retrieval the best we can expect from the Web? Or is it that if all we have is a search engine, everything we see is a nail?

The imagined next generation Web, the Semantic Web [4] has the potential both simply to continue the Web-as-Google paradigm, where the Semantic Web will make retrieval bigger, faster, stronger (the Semantic Web as Bionic Web). But the Semantic Web also has the potential to make take the Web closer towards Hypertext [27, 30], specifically towards the memex, Vannevar Bush’s vision of a machine that will support human-oriented knowledge building by supporting a person’s construction of associative links between one document and another.

Figure 1. View of mSpace Software Framework interface

mSpace is a project that uses Semantic Web technologies to support such knowledge building. In this paper, we discuss the motivation, design and architecture of the mSpace platform in terms of its use of Semantic Web technologies for approaching memex/hypertext goals. We first situate the discussion in a review of Bush’s memex, and then describe mSpace in terms of its interaction model, software framework and architecture. Our goal in this is to put the approach and framework before the community as one example of how hypertext research may inform the evolution of the Semantic Web, and to demonstrate as a case study in progress of some of the opportunities for hypertext research when situated in the context of the Semantic Web.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

HTí05, September 6ñ9, 2005, Salzburg, Austria.

Copyright 2005 ACM 1-59593-168-6/05/0009...$5.00.

2. BACK TO BUSH

A founding motivating vision, not just for Hypertext researchers, but for computer scientists, across disciplines [14, 34], has been Vannevar Bush's imagined memex[5]: a machine to hold *all* digital information and which will allow rapid, easy associative discovery of information and then path making from one discovered resource to another, supporting the capture of a person's annotations on those resources along the way. Some have predicted that we are close to achieving this vision [14], and indeed, the Semantic Web holds the promise of providing the technological underpinnings to enable that vision. It is then worthwhile to go back to the original description of the memex to remind ourselves of where we are with the Web relative to that vision, and where we might aim with the Semantic Web.

As noted, keyword search is incredibly practical, useful and effective. If we go back to the imaginary precursor of the Web, however, we see that Bush is critical of the limitation of such query systems. For him, being able to get at, use and extend "the record" of knowledge is critical. As such, he identifies selection, the finding of something useful, as one of the worst-served parts of the knowledge-building process. He states,

> The real heart of the matter of selection, however, goes deeper than a lag in the adoption of mechanisms by libraries, or a lack of development of devices for their use. Our ineptitude in getting at the record is largely caused by the artificiality of systems of indexing…The human mind does not work that way. It operates by association. *(All Bush quotations are to [5])*

That succinct quotation seemingly puts paid to any categorizing system, from search engines to the Semantic Web that relies on ontologies, classes and subclasses of relations. Bush proposes instead what has become a well-known concept to hypertext researchers, trail making: the ability to connect one thing to many other things, including connections to one's own annotations on a thing of interest, potentially implicitly (or explicitly) creating one's own categorization structure on top of the system's. These trails are imagined as robust and sharable. It is the making of these associative trails and their associated annotations that Bush sees as the method by which one may not only "extend the record" but makes it possible to consult it richly and effectively, beyond what is possible even in a library. As Bush states,

> A record, if it is to be useful to science, must be continuously extended, it must be stored, and above all it must be consulted…Even the modern great library is not generally consulted; it is nibbled by a few... Thus far we seem to be worse off than before - for we can enormously extend the record; yet even in its present bulk we can hardly consult it.

Bush proposes the memex as a solution to this problem.

2.1 Interaction before the "Artificial Index"

In the description of the memex, Bush founds its design on an interaction problem rather than a mechanistic one. More particularly, we might say he misdiagnoses an interaction problem as a system problem. Let us look at more of the above passage:

> Our ineptitude in getting at the record is largely caused by the artificiality of systems of indexing. When data of any sort are placed in storage, they are filed alphabetically or numerically, and information is found (when it is) by tracing it down from subclass to subclass. It can be in only one place, unless duplicates are used; one has to have rules as to which path will locate it, and the rules are cumbersome. Having found one item, moreover, one has to emerge from the system and re-enter on a new path.

What Bush describes is the way he imagines people being forced to interact with a system when the interaction mirrors the data structure: if the data is stored in classes and subclasses on a graph, then users must delve into classes and subclasses, coming up and going down as required by the graph rather than by their associative interests. Bush describes *an interaction design* as a solution to the problem, and he does so with a technique that has since become known as "scenario based design" [7].

> The owner of the memex, let us say, *is interested in the origin and properties of the bow and arrow*. Specifically he is studying why the short Turkish bow was apparently superior to the English long bow in the skirmishes of the Crusades. He has dozens of possibly pertinent books and articles in his memex. First *he runs through an encyclopedia*, finds an interesting but sketchy article, leaves it projected. Next, *in a history*, he finds another pertinent item, and *ties the two together*. Thus he goes, building a trail of many items. Occasionally he inserts a comment of his own, either linking it into the main trail or joining it by a side trail to a particular item. When it becomes evident that the elastic properties of available materials had a great deal to do with the bow, he branches off on a side trail which takes him through textbooks on elasticity and tables of physical constants. He inserts a page of longhand analysis of his own. Thus he builds a trail of his interest through the maze of materials available to him [emphasis mine].

In the above passage, Bush describes not an architecture, but the easy affordances of his system for tying items together, joining up trails and adding annotations "through the maze of materials available." This interaction, Bush implies, is the antithesis to the bumping about in disorienting subclasses of the "artificial index." But what feeds this interaction? There are certain assumptions implicit in Bush's description, both of the person's search strategy and the mechanisms that support it. Bush describes a person who has a clear and identifiable interest: "the origin and properties of the bow and arrow." Second, the scientist has a research strategy: "he runs through an encyclopedia" to find an article, and then finds "a history." He "branches off" but can come back to the starting point and adds his own "longhand analysis." Interestingly, Bush assumes that the researcher uses more or less traditional methods to start the search, with URL-like references: "These sources [the encyclopedia and history] are referenced by the "usual scheme of indexing." Indeed, "if the user wishes to consult a certain book, he taps its code on the keyboard…Frequently-used codes are mnemonic."

In today's parlance of software engineering and human computer interaction, we'd say that Bush describes an interaction layer for his nemesis artificial index system; he does not get rid of that underlying index, but imagines at least partially, a better, *more human* way for the interaction layer to support the way the human mind works, as Bush says, by association. Early Hypertext systems from Nelson's Xanadu [24] to Englebarts Augment [10] to Intermedia [26] to Microcosm [9] recognized the value of the interaction layer. Indeed, more recent non-Web hypertext work, such as Structural Computing, is premised on the requirement of flexible structures specifically to enable data to be represented to people in multiple ways – spatially, taxonomically, textually --

depending on a person's particular context and requirements [17]. Hypertext, ultimately, is a very human thing: Bush's memex scenario starts with interaction requirements rather than system requirements. The motivation of the Semantic Web is in large part to enable better machine processing of information: organized metadata and logics applied to information sources will improve the ability to reason over data and to create knowledge by inference it from it. The approach is often modeled by what has come to be known as the "Semantic Web layer cake" of languages, protocols and provenance. There is no user /interaction layer in this representation of the next generation Web **[ref].** It is therefore this *human first* approach to software architecture and application design in Hypertext research that may be one of its most critical contributions to Web and Semantic Web research.

As a demonstration of how the human first approach of hypertext may engage with Semantic Web research to bring about the memex, we describe a software framework and architecture that lets us combine some of the "artificial index[ing]" possibilities of the Semantic Web with an interaction layer that is richer than the current link list search results that so powerfully leverages the limitations of today's Web. In mSpace, we support associative exploration and provide a variety of means for annotations of these paths. We present an overview of the interaction, describe the framework and architecture, and conclude with a discussion of methods to further generalize the approach.

3. MSPACE

The powerhouse of the Semantic Web vision is inference via *automatic* association. Through the use of metadata to describe information and with ontologies to represent the relations of those things described, the Semantic Web will be able to construct inferences over data to generate new information, as well as, Bush-like, reference existing data. For instance, the answer to How Many Cows are in Texas [2] may not exist in any one document Bush's memex-using scientist may be able to locate. So, rather than having to do the calculation manually by plowing through multiple texts, a Web Service may be able to infer that count by running calculations over associated bovine data. How these queries and their results are to be represented in the Semantic Web is an open question.

There are a variety of compelling projects in the Semantic Web application space that are looking at how to provide queries and represent results in a coherent manner. Haystack, while predating the Semantic Web, has been redeployed as a Semantic Web approach to integrate personal information via a semantic layer to break the boundaries of application limits imposed by applications specific data types [18]. In this respect of trans-application data association, it is reminiscent of Microcosm.

Another class of Semantic Web interface is the faceted browser which supports manipulations/queries on metadata associated with data. Such categorizing data can means that information can be found in a variety of sets. Flamenco [39] takes an interactive approach to facet selection and results display, where selecting facets of interest produces a result that is kept in view beside remaining available facets. Endeca.com has had particular commercial success with this approach. Longwell, a facet-browser for the SIMILE [6] digital library metadata project, and Topia [29] implemented on an art gallery's collection, both have participants select facets of interest first, then the system generates representations of data constrained by those facets. FACET itself is a project that looked at using thesauri to discover associated facets in order to discover related concepts [38]. Reflecting *relationships* (possible associations) among elements is less important in these viewers than foregrounding that the results have all the selected facets as "true." Also, in each of these approaches, there seems to be an implicit assumption that the person using the system knows enough about the data space that informed facet selection is possible. This approach is in keeping with Bush's selection by his scientist who knows what he wants to discover and largely where to look for it. mSpace starts from the premise that a person may not have this knowledge, but does have interest.

The motivation for mSpace is to improve access to information, and by improving access, help people get from where they are able to start with a domain of interest - what they may know about it - to where they wish to be in terms of what they wish to learn about a topic. We use the term "learn" in a colloquial sense of wishing to learn about a subject, rather than drawing on any particular model of learning. From this focus on *access* first, we differ slightly from Bush's motivating focus on *selection*. As noted above, Bush, whose target group for the memex is the research scientist, imagines a scenario in which the researcher knows what resources to call down to initiate his [sic] quest. In mSpace, we want to be able to support the person whose query may not yet be formulatable for comparison against an index. For instance, the scenario we have described previously [32] is that of a person who may not know much about classical music but who knows what they like when they hear it, and yet wishes to be able to access classical music. In this case, pulling down a perfect and complete index of classical music would not be able to assist the person carry out Bush's goal of *selection*. The terms would be meaningless. Our founding challenge in mSpace has been therefore first to support access especially where lexical expertise may be missing, and from there, to enable selection, association and annotation of paths of interest.

Our approach in mSpace has been to enable the exploration of an information space in multiple ways, to leverage a variety of modes a person may find useful for initiating access to such a space. From this accessibility-driven approach, we propose an architecture to support these interaction mechanisms. Our interest in embodying the architecture in the protocols and languages of the Semantic Web is that it lets us connect our approach with the *scale* of the Web's distributed information .

3.1 mSpace Interaction

Fundamentally, mSpace is an interaction model which exposes relationships within an information space and which provides a set of manipulations on that space to assist the exploration of those relationships. The formalism of this model has been described elsewhere [13, 21]. We focus here on the basic representation and manipulations supported by the model, and then describe the architecture developed to enable these effects.

3.1.1 mSpace Representation

An mSpace assumes there is a high-dimensional information space. It is difficult for humans to manage visualizing more than three dimensions, and indeed, a quarter of the population has difficulty managing three dimensional representations on two dimensional screens [22]. An mSpace, therefore proposes a method for managing high dimensional spaces on a two-dimensional space. This is the notion of a *slice*. A slice takes a projection through a multidimensional space, which effectively flattens the space, temporarily creating the sense of hierarchical dependencies among dimensions. While other visualizations are possible, we have been presenting this hierarchical slice as a set of columns in a spatial layout as seen in Figure 1. For instance, a slice in the Classical Music space may be Period, Composer, Arrangement, Piece. What is selected in the Period column constrains which composers are presented in the Composer column; selections in Composer constrain available Arrangements and so on.

3.1.2 mSpace manipulations

A variety of manipulations are supported on the slice: *sorting, swapping, subtraction*, and *addition*. That is, the dimensions in the slice can be rearranged and changed. If a person has no knowledge of Periods, for instance, but took piano lessons as a child, they may wish to see the world organized by Piano, rather than having to go through a space organized by composer or period and, to use Bush's image, go up and down the graph to find piano pieces. Sorting allows a person to rearrange the information in a manner that suits exploring the space in a way that makes sense to them. Likewise, a person may add new dimensions (Figure 2). From considerations of composer, a person may wish to move to an associated consideration of recordings or artists performing these recordings, to the history of recording practices at the time of a specific artist. Likewise, other dimensions may be removed. These manipulations mean that the person can construct a representation of a space, and Bush-like pull in associations on demand, which support their interests. Indeed it is possible to have multiple instances of an mSpace running on the same information in order to create multiple arrangements of space in order to compare the relationships in these different arrangements. In this way, the interface begins to support Bush's sense of selection, and exploration of association.

Figure 2. Sorting and swapping dimensions

3.1.3 Preview Cues

Beyond the representation of the domain space and the manipulations afforded on it, mSpace also provides a variety of methods to interrogate the data associated with the dimensions. While the columns present the explicit dependencies of a given slice's organization, two other attributes, Preview Cues and Info Views, support exploration by association.

Figure 3. Audio preview cues to preview an area of a domain.

A preview cue, not unlike fluid links [40] provides a gloss on an instance associated with a dimension. More particularly, however, Preview Cues are imagined as multimedia rather than textual elements. In the classical music example, for instance, one may hover a cursor over an icon next to Baroque in the Period column. This gesture will launch a set of representative audio cues from the identified period (Figure 3). The person will be able to play all or any of these cues in order to determine, in the Marshal sense of *information triage*, [20] whether this area is of interest to them or not. If it is, they can select the given period and move deeper along that path. If not, they can move onto another topic. In this way, people can preview the areas of interest at the surface of their exploration rather than, again, having to bounce up and down the graph of Bush's artificial index. We have described the rationale for Preview Cue's design, and the evaluation of their deployment elsewhere [31]. Suffice it to say for these purposes, that preview cues have been highly effective in helping people access a domain that has otherwise been experienced as inaccessible because of lack of specific domain knowledge to enable use of traditional tools like keyword search.

3.1.4 Info Views

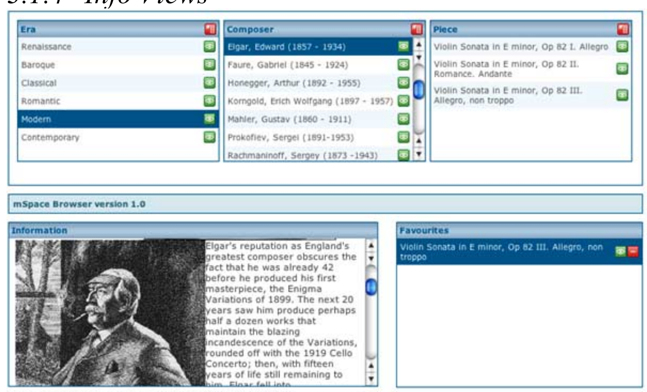

Figure 4. Info Views show information associated with a selected entity

The info view acts as a place to present more detail about a currently selected instance in a dimension. In mSpace, the selection of an entity both populates the next column of the space, and populates the info view for that instance. For instance, the selection of a composer causes the information view to open with a description of that composer (Figure 4). The info view can be a focus or a supplement. A person may only wish to know something readily revealed by categorical association: Beethoven is in the same Era as Mozart, but not the same one as Bach. We do believe, however, that having associated information available *peripherally* assists in building up a sense of context about an information space, and also helps a person become aware that more information is available in that space should they wish to pursue it. We describe in the architecture sections following how info views are populated.

3.1.5 Favorites and State

At any point, a person may select an element in the browser they wish to keep for future reference. This is stored in a persistent list. Likewise, a person may save or share the particular state (arrangement) of the browser for future reference. This recallable state acts in a manner similar to Bush's trails through a data space, showing what dimensions were associated with which part of the data. These states can be replayed chronologically as well to reconstruct how the associations evolved.

3.2 mSpace Gestalt: Related Work

As we have said elsewhere [32] Columnar, spatial views of information are not new, but despite their utility, they are not common on the Web. Further, they do not enable the full set of manipulations we describe here. The innovation of the mSpace approach may be the gestalt effect of bringing together spatial representations of multiple kinds of associated information within a single context that is highly manipulatable for user-determined explorations of an information space. The slices draw in manipulations on human-associated information in context. It is in respect to mSpace's focus on interaction, the leveraging of metadata rather than content alone, and the automatic association of new content, whether constructed by inference or resulting from a direct reference, that makes it distinct from its hypertext predecessors. For instance, Trigg's Guided Tours and Tabletops [36] work is orientated around creating and organizing card indices for information spaces supported by NoteCards: the tabletop is a visualization for arranging the digital note cards that made up the trail. The Walden's Path work focused on path construction as an overlay to available web site pages [12], not unlike Footsteps [25]. More recently, Kim *et al* created HATS (Hypertext Annotation and Trail System) that supports trails through and annotations on Web pages using WebDAV [19]. WebPath [23] reflects similar goals. In each case we see an overlay of pointers/annotations to explicit content, very memex-like.

The mSpace approach facilitates connections over a richer data space than just existing web pages by foregrounding the *facets* associated with information within a page as well as re-presenting the sources themselves. This multiple interaction approach enriches the associative possibilities for a person exploring a domain. The goal again of mSpace is not only to support association, but to support *access* to material to begin to create associations; to facilitate and foreground the relations *around* the information, to provide multiple possible access points. This is an interaction paradigm, informed by usability research, that Bush's "indexophobia" did not allow: that the metadata of the index could be as important for exploration/association as the thing itself.

To explore the approach, we have developed a two phase approach: a light-weight framework to generate mSpaces now, which we have released as open source. Second, we have begun to develop a more robust architecture to support a more distributed (Semantic) Web design. The framework has allowed rapid development of a variety of mSpace applications, thereby allowing us a means to assess the strengths and weaknesses of the concepts in the model from practical use-case perspectives. The more formal architecture has evolved from the lessons being learned from the framework. The emphasis of the architecture, however, has been to explore how to best enable applying mSpace interaction to distributed heterogeneous sources, including the ability to help annotate and link sources in a memex-inspired way.

4. The mSpace Software Framework

The mSpace framework (available at mspace.sourceforge.net) is based on our experience with a large-scale Semantic Web project, CS AKTive Space [33] supports exploration of the computer science research domain in the UK. That was a purpose built application to afford some of the properties of the mSpace interaction model as an interface to a large dataspace of tens of millions of *triples* (described below). The application used a richly complex ontology and hand-crafted queries to describe the relationships between columns. While the application won the Semantic Web Challenge of 2003, it was very much a one-off, custom build.

Our hypothesis in developing the mSpace Software Framework has been that it was necessary to construct an approach that would make it possible to *sling*, in mspace parlance, an mSpace onto any domain space, and which would provide the representations and manipulations on data described above. We chose to develop the framework using Semantic Web languages and protocols because first, these are open (albeit evolving) standards which make the framework readily extensible and second, because it is deployed in the Web space, we have the potential to connect to the world's largest information resource. This access lets us begin to evaluate our approach at scale. In the following sections, therefore, we describe our approach in building our initial framework.

The framework has the following three main components: a client, a model and data storage. The client has three roles: to query the domain space, to represent the results in the interface, and to provide the manipulations on the domain as described above. The model defines the dimensions and their relationships available in the domain. The storage layer, in our case a triplestore, supports rapid returns of complex queries on the data space. To facilitate the discussion of these components, we refer to our publicly available demonstrator of the framework, the Classical Music explorer (http://demo.mspace.fm).

4.1 The Client

The client serves several purposes, first to support the interaction with the user, allowing the addition, subtraction and movement of columns, and to create a query based upon the user's selections in the interface. The selections are chained into a query that is sent to the triplestore (described below), via a web-based API that returns a document, currently in XML, that represents the results. An associated web page is inserted into the Info View by appending the URI of the selected item as the query string of a specified information-providing page. In the case of the Classical Music browser, the information-providing page performs a server-side lookup of information that is in the triplestore and returns it formatted using XHTML and CSS. The favorites functionality of the client is also purely client-side, with the saving of the URIs of the favorite items into a cookie that is then looked up by the client on the loading of the page. The client runs in-browser, using JavaScript with heavy use of the XMLHTTPRequest object to prevent page refreshes

In order to provide the functionality of trails to the user as reflected in a particular arrangement of columns, a link is provided that returns the user to the current state of the system, which can be bookmarked using the user's regular bookmarks functionality. After a selection is performed, or after an interaction that alters the layout of the columns, the link updates to reflect the currently shown columns and selected values, and can be saved as a new bookmark at any time.

4.2 The Model File Approach

The mSpace model is a segment of RDF, a manner of using a graph structure to describe data in the semantic terms of subject predicate object relations (triples) [28]. The model file uses RDF to describe both the layout of the data to be mapped to the mSpace interface, as well as how the dimensions are to be displayed and how to sort them. The use of the RDF graph enables the user-defined hierarchy that lends the mSpace interface much of its power. As described in [13] these relations are centered around the concept of a “goal” column. That is, when a query is constructed, all predicates are constructed relative to this column. The model defines *predicate lists*, which describe the path from the goal items to a specific dimension. The predicates themselves are loosely defined, so that having specific ontologies is not necessary to apply an mSpace to the data. For instance, one may use the same URI of the predicate that is in an ontology, or not.

This lack of an ontology as a requirement for a Semantic Web application is a considerable asset for getting light weight Semantic Web applications up and running. Ontologies describe relationships of entities in a way that supports inference over data to which the ontology is applied. Ontologies have also been referred to as the bottleneck of the Semantic Web [11] : there may be multiple ontologies for a given domain (how do they interoperate? do they); there may be none (now what?).

The mSpace framework reflects the approach that we can leverage RDF in our model file as a kind of implicit, light weight ontology, and then leverage existing ontologies to complement the model where and when they exist. The model has been designed so that it is generic enough to be applied to structured data that exists already, without having to redesign the data relationships in a formal ontology. This approach allows an mSpace to be applied quite cheaply. For instance, the data for the Classical Music demonstration was converted using ontologies found on the web, with our own predicates used where they did not exist already.

4.2.1 Multi-hop

One of the innovations of the mSpace framework over its CS AKTive Space predecessor is the use of predicate lists in the model file, thus enhancing the automation of deploying mSpaces. That is, when a query is constructed, all predicates are constructed relative to this column. Predicate lists address the limitations of one-hop predicates, which in CS AKTive Space required custom queries to be written to support columns that were not directly (one hop away on the graph) associated with the goal column. For example, in the Classical Music demonstrator, Piece acts as the goal column. If we wish to extract all composers in an mSpace where pieces of music form the goal column, and the composer of a piece is linked to from that piece, we might use the RDQL:

```
SELECT ?y WHERE (?x, <mspace:has-composer>,
?y) USING mspace FOR

http://mspace.ecs.soton.ac.uk/ontology/Class
icalMusic/
```

Literally, the above expression considers all pieces x, finds their composer y, and returns every unique instance of y. This is a very simple query, and already provides a degree of power in visualizing stored data.

It is important to offer the ability to extract data more than one link in the graph away from the goal column. Someone may wish to explore pieces by the country in which the composer was born. The country of birth of the composer is a property of the composer, however, not of the goal column piece of music. The multi-hop architecture of mSpace allows for this association between country and piece to be represented. In a structure such as that shown in Figure 5, this would allow finding pieces based on, for example, the country from which the composer originated. Support for this ‘multi hop’ data is simply a matter of chaining two or more fragments together, as shown here:

```
SELECT ?y WHERE (?x, <pred1>, ?i1), (?i1,
<pred2>, ?i2), (?i2, <pred3>, ?y)
```

Construction of a query to update a single column in mSpace can be visualized, then, as a pair of loops. The outer loop chains together the graph patterns representing the predicates/selections of all columns to the left of the column being updated, and the graph pattern representing the predicate for the updating column and the variable that will contain the required results. The inner loop constructs the graph patterns for an individual column.

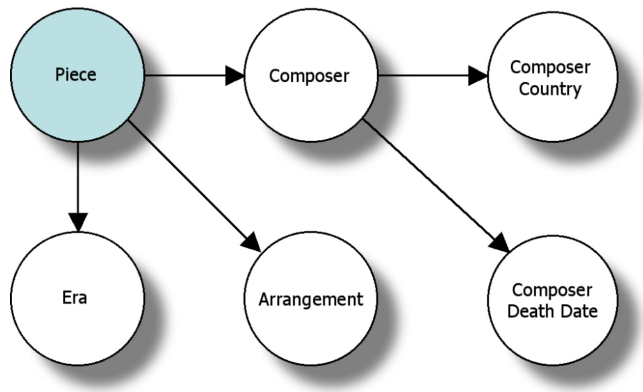

Figure 5. Multi-hop illustrated.

The following RDQL fragment shows an example of this process:

```
SELECT ?y WHERE (?x, <mspace:has-era>,
<mspace:eras/Baroque-era>), (?x,
<mspace:composed-by>, ?i1), (?i1,
<vcard:Country>, ?y) USING mspace FOR
<http://mspace.ecs.soton.ac.uk/ontology/Clas
sicalMusic/> vcard

FOR <http://www.w3.org/2001/vcard-rdf/3.0#
```

Above we see a query to extract the countries of all composers that composed pieces within the Baroque era. This is a matter of finding all pieces x that were composed in the Baroque era, and had a composer i1 who lived in a country y, and returning y. It is important to note that in providing information on how to access data, the mSpace model makes weak guarantees that certain relationships will hold. If the relationship does not hold, it will simply limit the ways in which some data can be accessed. For example, if I expect that a composer will have a birth place, I could, for example, select birth place and return information on composers that were born in that place. If the data does not exist or is encoded in an unexpected manner, that data simply will not appear as part of the returned information set.

4.3 Triplestore

The server that is queried by the client is called a triplestore, because it takes an input of many RDF files, and connects them up in to one large graph structure, using an SQL database as storage. The triplestore then allows for the graph to be queried using RDQL. The queries place constraints on the graph in order to return sections of the graph that are of interest.

The software used by mSpace is called 3store [16] and includes a web-based query interface that returns the results of the query in XML. This XML is interpreted by the client and used to populate the columns and interface. The Sourceforge download package includes instructions on how to install the 3store server and the necessary requirements for doing so.

4.4 The Classical Music Example

In the implemented mSpace Classical Music browser (http://demo.mspace.fm), the data used was taken from the ID3 tags [1] of a collection of mp3 music files. These music files were added to an Apple iTunes library, which provides the ability to export the tags of the music into an XML property list, which was then parsed and output to XML-RDF. When generating the RDF we were very aware that we should correctly utilize specified ontologies for the data, and made use of a cut-down version of the Kanzaki Music Vocabulary (http://www.kanzaki.com/ns/music), maintaining the same URIs specified in that namespace. This RDF was then asserted into a RDF 3store triplestore, which allows for RDQL queries to be made via RDQL-HTTP (a precursor to SPARQL, with similar syntax). The syntax on RDQL queries allows for straightforward injection of triple constraints on the returned data. Using 3store, we were able to query the classical music RDF using the web client, through use of the XMLHTTPRequest object directly from the client, generating the queries, from the constraints that were made in each column.

4.4.1 Applying an mSpace to other data

While there is a certain amount of RDF formatted data available on the Web, it is likely that the space that one wishes to browse using an mSpace is not formatted as RDF. A traditional data warehouse would be database backed and available to be queried using SQL. In order to apply an mSpace to this data, and to also realize the interoperability benefits of Semantic Web technologies, this data either needs to be converted to RDF, or it would need to be run against a data engine to allow this store to be queried using RDQL.

At the time of writing, the cheapest way to introduce mSpace to an existing data store is to perform SQL queries on the database such that dumps of the data can be exported, in order to be parsed (usually using a scripting language such as Perl) so that RDF that represents this data can be generated. The frequency of this operation should correlate to the frequency of changes of the data, which is likely to be tied to the type of data being modeled. For data that changes often, it is preferable to regularly update the RDF. **This** convert and store approach is similar to that taken by the AKT project to demonstrate the promise of Semantic Web technologies as well as to have a platform to explore research issues within such deployments.

We see RDF conversion and storage as an interim evil, and at approaches like the mSpace framework as viral. The framework is small and lightweight enough to begin to expose the cost/benefit value of making data available in RDF at source for the use not just of mSpace, but of any service that can add value to that data by being available to those services.

4.5 Analysis of Current Framework

In Star Trek: the Motion Picture, the main character is V'ger a planet-sized probe that has made its way to earth from the furthest reaches of space and wants to know what to do with itself. It threatens to destroy the earth if the "carbon units" do not let it communicate with its maker. It turns out that V'ger is actually the Voyager space probe, the mission of which was to collect knowledge about the universe and bring it back to earth. V'ger has fulfilled its mission: it now holds all knowledge. V'ger ends up evolving into a higher being (taking two of the Enterprise crew with it). It starts by threatening all life on earth and ends by actually communicating nothing, taking all its knowledge with it to its higher plane of existence. This story is a parable for the Semantic Web: one of its major goals is to work as a distributed resource; its current embodiments have largely been, however, V'ger like, of necessity sucking in data from heterogeneous sources into a project's single triplestore and from there, offering services on those sources.

The mSpace framework is similar: relying for core information on locally held data it then queries to populate the interface. The advantage of this approach is that it is lightweight, and can be rapidly deployed on a dataset, allowing people to explore the opportunities of using a standards based, open Semantic Web application. We released the framework on the Open Source project server Sourceforge at the end of January 2005, mspace.sourceforge.net, along with a detailed technical report [15]. At the time of writing, the report has been viewed over 1200 times (unfortunately stats are not available on number of software downloads). This approach has allowed us to gain considerable feedback from within and without the Semantic Web community. In our own department, the software is being used in a variety of projects independent of the mSpace project.

Based on the results of actually deploying such a lightweight application we have gained insights into how we can deploy mSpace as a more generic browser of mSpaces, rather than of a single mSpace at a time. Our eventual goal is to move towards mSpace as a generic Semantic Web browser. The following section describes the current architecture for supporting mSpaces which utilize and support association across distributed resources.

5. MSPACE ARCHITECTURE

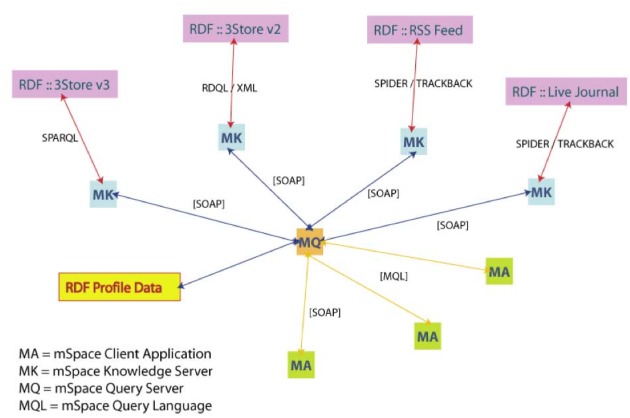

Figure 6. mSpace Architecture of MA, MQ, MK layers.

In order to allow a single mSpace interface to be able to explore disparate concepts from multiple distributed sources (to move from information about classical music to history of a selected Era, for instance), it is necessary to be able to query multiple knowledge servers. Currently this is not possible, as the client only supports the querying of a single triplestore. The architecture described below is designed to query multiple triplestores, as well as support incorporation of resources that may not yet be referenced in triplestores.

5.1 The Architecture Layers

To accomplish this**,** the proposed system takes the implemented mSpace framework and abstracts several of the internal concepts of query generation and triplestore querying in order to distribute the system, and to allow multiple heterogeneous and distributed external data sources to be browsed from one mSpace. Specifically, we have developed a three layer architecture: (1) The mSpace Application (MA), (2) the mSpace Query Service (MQ (3) the mSpace Knowledge Service (MK) First, we separate the framework's client into two distinct parts: the application layer and the query layer (MA and MQ). The triplestore is also abstracted into a generic knowledge providing server (MK).

5.1.1 The MK

The MK makes it possible and tractable to use a WWW approach to data provision: to have multiple providers of data that remain controlled separately. The approach also supports other Web features like author-linking of resources, ease of creation of new data, simple distribution and user chosen data sources. The MK approach provides a generic interface to knowledge with a protocol that maps onto any triplestore implementation.

Since we use Semantic Web protocols, the architecture needs to be able to query RDF graphs, and is designed so that any query language standard can be used, which at the time of writing is RDQL, the standard language for querying RDF. The system is ready to embrace SPARQL, a more flexible revised protocol that also allows a variety of return formats such as XML and RDF to be specified.

5.1.2 The MA / MQ interaction.

In order to allow clients to be written for multiple platforms, and to facilitate the use of mSpace querying within other applications, the concept of the client is described in the architecture as a generic mSpace Application (MA), separated from querying operations, which are translated to an mSpace Query Service (MQ). In order to: facilitate maximal compatibility among possible (Semantic) Web applications; enable applications to interact with this service easily, and to keeps the complexity of the protocol low, two specific access methods**,** Simple Object Access Protocol (SOAP) and HTTP, are used for the communication between the MA and the MQ. Hypertext Transfer Protocol (HTTP) is the core protocol for the Web and Web communication. SOAP is the open communication protocol for Web Services. Using SOAP/HTTP allows all servers to be accessible on the Web as Web Services for maximum compatibility and interoperability. The MA communicates initially with the MQ, which acts as a broker (or, effectively, a distributed linkbase), with knowledge of domains and of relevant mSpace Knowledge Services (MKs). In this structure, the MKs communicate with the MA *after* consultation with an MQ. An MK is the equivalent of the triplestore in the current mSpace implementation, and in the case of initial implementations, the MK communicates via HTTP-RDQL with a 3store triplestore. The immediate purpose of the MK, once discovered by the MQ as an appropriate repository, is therefore to construct an RDQL query using constraining triples that are specified by the MA.

5.1.3 The MQ design

The MA concept is designed such that the application can take any form that can implement the communication protocol. One of the important concepts of the architecture is the MQ, which is the first point of contact for a particularly configured client, and is the part of the system that determines which MKs to use, and therefore what the ultimate domain of the data will be. The equivalent of this to the WWW is that for a current affairs service, you might go to BBC News, who have aggregated content from their own reporters, as well as their own choice of articles from Associated Press, Reuters and more. The choice to select a particular news site such as BBC News is that of the person. Similarly, if one wanted to get information about a place to eat, they may visit a general-purpose local information supplier (such as Thomson Local) or a domain-specific restaurant guide. The same applies for the MQ model; the choice of MQ determines the content. This is equivalent to the ability to choose what links you have on your WWW site, making the information relevant, as well as preventing an information overload scenario. The key to this is that the MQ is queried by the client and as such, the client is only aware of the links provided by the MQ.

5.1.4 The MQ / MK interaction

Within a Model View Controller (MVC) approach the framework client application takes the role of both the View and the Controller. The XMLHttpRequest receives results from the triplestore that is parsed and then fed into the user interface controls. It is this stage of the process that can cause a bottleneck. As the number of results returned from the triplestore increases, so does the demand put on the client's browser. This issue is likely to become more apparent on devices that have limited processing power, such as Personal Digital Assistants (PDAs) or smart phones. With the revised architecture that separates the view (MA) module from the controller (MQ/MK), we introduce the mSpace Query Language (mQL). The mSpace Query Language is used to provide only the *necessary* information about the current column layout and item selections to the controller, and also return only the necessary formatted data to the view. This has the effect of not only reducing the processing power required by the client, but it will also reduce the data transfer between the view and controller.

The power of the MQ/MK can also be utilized to provide support for many different devices. For example Web clients can indicate that they require JavaScript array-formatted results, whereas a powerful desktop application could handle complex XML formatting. There are several other extensions that mQL supports within a client application. The information box and preview cues can be populated using mQL. The advantage of using mQL for the information box again lies in supporting multiple devices such as PDAs that have different screen requirements.

6. Near Future: Annotation and Association

Some of the feedback on the mSpace approach so far has been to suggest that rather than constructing a set of mSpace models, or collections of predefined dimensions, we construct a general mSpace browser which can integrate/associate any dimensions from anywhere in the Web. This vision seems very much to reflect a facetted browser approach where dimensions of interest are selected first, and then the space is organized accordingly. The Sculpture project [35] is using the mSpace approach over an ontology in just this way. Our concern in terms of access rather than selection first suggests that the model approach has value both for motivation for seeding a particular information space of interest, and for creating a means to engage with a domain that may otherwise be inaccessible: how would someone who knows nothing about classical music be able to make use of facet selection first before exploration? Bush envisioned people who would delight in making trails for others through complex information spaces. We imagine mSpace domain modeling as a similar practice, and mSpace domain models similar, if richer, than a specific trail. They provide a kind of rapidly manipulatable organization to help support a person in determining where they wish to branch off, pull in new dimensions. Critical in this model making and trail blazing is the discoverability of associate-able knowledge stores. The following sections describe our designs for supporting development of one's own annotations which can feed into an mSpace, and a proposed mechanism for connecting models to each other at dimensional pivot points.

6.1 Annotation and Publication

In order to make mSpace knowledge sources (MKs) discoverable for mSpace models, we plan to utilize the rapid publishing and notification mechanisms of the Web Log (Blog) trackback [37] mechanism. A trackback URL will be set up for each RDF file, enabling the MK and the author's publication software to link to each other, so that when a user updates the RDF, the trackback is performed so that the MK updates its data graph with the updated RDF file. With this approach, the scalability and inter-graph relationship benefits that a triplestore provides are preserved, while allowing for updated-on-change distributed RDF data to be brought together and queried concurrently.

The *trackback* concept is based in Blogging software architecture [3] that supports subscription-based notification of content changes. Subscription means that deciding the relevancy of data to a particular MK is up to the provider of that aggregator (a primary function of the MK). Taking another example from the blog world, some blog hosting services (such as LiveJournal.com) allow user communities to be set up. A community of users could therefore contribute the data they consider to be important in one MK, in a way that is similar to the way people contribute to blogs. Currently a user posts a blog entry to the community, and may include links (HTML anchors) to off-site pages that they deem to be noteworthy. A blog-like engine that hooked into an mSpace service could allow that link to be to a new piece of RDF that then is hooked into the MK, with the trackback to update the store when the external site updates their RDF. The blog trackback model gives us both the means to, memex-like, associate an annotation directly with an instance in an mspace, and makes new resources and dimensions publishable and discoverable.

There is growing interest in Web Logs in the Semantic Web Community [8].We also see considerable potential in the Blog space to further enhance association and annotation with mSpaces. Blogs already output to RDF. Blogs also make use of user-defined categories to organize blog entries. By linking with blogging software, as one example, we can make it easy to discover and associate ontology descriptions with categories for mSpace model discovery. In this respect, we leverage existing Web practice, enhanced via the strong linking properties of the Semantic Web approach, to enable the memex's.

6.2 Intersecting mSpaces

The above architecture defines how multiple knowledge sources can be associated with a dimension in an mSpace model. In order to support Bush's ready travel by association from one related idea to another, the mSpace approach requires a mechanism to support connecting one model with another. For instance, in the classical music space, a person may wish to move seamlessly from Beethoven's musical works to the history of events around any of those works. There are at least two ways to support these connections: pre-authored associations of domains at the model level, and user-determined association at the interface level.

In the current mSpace framework, each mSpace knowledge domain is defined using an mSpace model that at its highest level defines the column contents in relation to a fixed goal column. The semantic model describing the domains makes it possible to link mSpaces by comparing the `rdf:type` of the content of their columns. Authors of mSpace models will be able to add links from their mSpace to external mSpaces in a similar manner to the way they use traditional hyperlinks to link their web pages to other existing web pages. Additionally an MA client can be written to support both recommendations of associate-able mSpaces and specific searches for the discovery of available mSpaces. Both MAs and model generators can send out requests (via MQ/MK) to all known mSpaces in search of map-able dimensions. Our near term plans are to develop tools to make model authoring, content seeding and trackback feasible for rapid deployment. We will incorporate these as extensions to the existing framework so that systems already deployed will be able to take advantage of these extensions.

7. Conclusion and Further Future

We have presented an overview of the mSpace interaction, current framework and architecture. We have presented the approach as a way to demonstrate the connective tissues between fundamental hypertext issues and the Semantic Web. We have grounded these connections between the Semantic Web and hypertext through Bush's founding vision, the memex and have suggested how facetted browsing approaches facilitated by the Semantic Web's capture of metadata may enrich that vision, going beyond associations of one thing to another to support exploration of contexts about one thing with contexts of another.

In the mSpace framework we have a lightweight approach that supports associative exploration and trail-making interaction over lightly-structured data. By making ontologies initially optional, we have made it feasible to do rapid prototyping *at Web scale* of hypertext concepts (such as link typing or distributed link bases) which these new protocols afford but have not yet been fully explored. With lessons learned from applications developed with the framework, we have developed a more robust architecture to enable the use of distributed heterogeneous data stores, and support automatic discovery of these resources to be associated with an mSpace. The contribution of our framework approach is to provide a practical platform for hypertext exploration that takes advantage of Semantic Web protocols which let us support in the wild of the Web, Bush's sense of the way the human mind works, through human-made association.

8. ACKNOWLEDGMENTS

Thank you to Kieron O'Hara, David de Roure, Dave Millard, Les Carr and Helen Ashman. mSpace is sponsored by the Advanced Knowledge Technologies Project (aktors.org). Special thanks to Wendy Hall and Nigel Shadbolt for their support of the project.

9. REFERENCES

1. *ID3v2 made easy*, id3.org, http://www.id3.org/easy.html.
2. Search for Tomorrow *The Washington Post*, BizReport, 2004.
3. *Six Apart: Developer Documentation: TrackBack Technical Specification*, Six Apart, 2002-2004, http://www.sixapart.com/pronet/docs/trackback_spec.
4. Berners-Lee, T., Hendler, J. and Lasilia, O. The Semantic Web *Scientific American*, Scientific American, 2001.
5. Bush, V. As we may think. *interactions*, *3* (2). 35-46.

6. Butler, M.H., Gilbert, J., Seaborne, A. and Smathers, K. Data conversion, extraction and record linkage using XML and RDF tools in Project SIMILE. HP Labs, 18, HPL-2004-147, 20040831

.
7. Carroll, M.J. *Making Use: Scenario-Based Design of Human-Computer Interactions*. MIT Press, Cambridge, MA, 2000.
8. Cayzer, S. Semantic blogging and decentralized knowledge management. *Commun. ACM*, *47* (12). 47-52.
9. Davis, H., Hall, Wendy, Heath, Ian, Hill, Gary and Wilkins, Robert, Towards an Integrated Information Environment with Open Hypermedia Systems. In *ECHT '92: Proceedings of the Fourth ACM Conference on Hypertext*, (Milan, 1992), 181-190.
10. Engelbart, D. and English, W., A Research Centre for Augmenting Human Intellect. In *Fall Joint Computer Conference*, (San Francisco, CA, 1968), AFIPS, 395-410.
11. Fensel, D. Ontologies: Their Glory and the new bottlenecks they create. *OntoWeb Workshop.Semantic Web Project Proposal.*, Vrije Universiteit Amsterdam (the Netherlands), 2000.
12. Furuta, R., III, F.M.S., Marshall, C.C., Brenner, D. and Hsieh, H.-w. *Hypertext paths and the World-Wide Web: experiences with Walden's Paths*. ACM Press, Southampton, United Kingdom, 1997.
13. Gibbins, N., Harris, S., Dix, A. and schraefel, m.c. Applying mSpace interfaces to the Semantic Web - working paper. IAM Group, University of Southampton, Southampton, 2004, Working Paper,
14. Gray, J. What Next? A Dozen Information-Technology Research Goals, ACM Turing Award Lecture. MSR, 1999, MS-TR-99-50.
15. Harris, C., Owens, A., Russel, A. and Smith, D.A. mSpace: Exploring The Semantic Web. A Technical Report in Support of the mSpace software framework. IAM Group, University of Southampton, Southampton, 2004, 98, Technical Report,
16. Harris, S.a.G., N., 3store: Efficient Bulk RDF Storage. In *1st International Workshop on Practical and Scalable Semantic Systems*, (Sanibel Island, Florida., 2003), 1-15.
17. Hicks, D.L. and Wiil, U.K. Searching for revolution in structural computing. *Journal of Network and Computer Applications (JNCA)*, *27* (1). 27-45.
18. Karger, D.R. and Quan, D. *Haystack: a user interface for creating, browsing, and organizing arbitrary semistructured information*. ACM Press, Vienna, Austria, 2004.
19. Kim, S., Slater, M. and Whitehead, J., WebDAV-based Hypertext Annotation and Trail System. In *Fifteenth ACM Conference on Hypertext and Hypermedia*, (2004).
20. Marshall, C.C.S., F.M., Coombs, J.H., VIKI: Spatial Hypertext Supporting Emergent Structure. In *ACM European Conf. on Hypermedia Technology*, (1994), ACM Press, 13–23.
21. McGuffin, M.J. and schraefel, m.c., A comparison of hyperstructures: zzstructures, mSpaces, and polyarchies. *Fifteenth ACM conference on Hypertext & hypermedia*, (Santa Cruz, CA, USA, 2004), ACM Press, 153-162.
22. Modjeska, D. Hierarchical Data Visualization In Desktop Virtual Reality. Department of Computer Science, Univeristy of Toronto, Toronto, PhD Dissertation, 2000.
23. Moody, P. WebPath: Sharable Personalized Guided Web Tours. IBM Watson Research Center, Cambridge, MA, LotusTechnical Report, no: 98-09.

.
24. Nelson, T.H. *Computer Lib/Dream Machines*. Microsoft Press, 1987., Redmond, WA, 1982.
25. Nicol, D., Smeaton, C. and Slater, A.F. *Footsteps: trail-blazing the Web*. Elsevier North-Holland, Inc., Darmstadt, Germany, 1995.
26. Norman, M. *Intermedia: The architecture and construction of an object-oriented hypemedia system and applications framework*. ACM Press, Portland, Oregon, United States, 1986.
27. Ossenbruggen, J.v., Hardman, L. and Rutledge, L. Hypermedia and the Semantic Web: A Research Agenda. *Journal of Digital Information (JoDI)*, *3* (1).
28. Rusher, J. *Triple Store*, 2001, http://www.w3.org/2001/sw/Europe/events/20031113-storage/positions/rusher.html.
29. Rutledge, L., Alberink, M., Brussee, R., Pokraev, S., Dieten, W.v. and Veenstra, M. *Finding the story: broader applicability of semantics and discourse for hypermedia generation*. ACM Press, Nottingham, UK, 2003.
30. schraefel, m.c., Carr, L., Roure, D.D. and Hall, W. You've got hypertext *Journal of Digital Information (JoDI)*, 2004.
31. schraefel, m.c., Karam, M. and Zhao, S., Listen to the Music: Audio Preview Cues for Exploration of Online Music. In *Interact 2003 - Bringing the Bits Together*, (Zurich, Switzerland., 2003).
32. schraefel, m.c., Karam, M. and Zhao, S. mSpace: interaction design for user-determined, adaptable domain exploration In hypermedia *AH2003*
Workshop on Adaptive Hypermedia and Adaptive Web-Based Systems, Nottingham, UK, 2003.
33. Shadbolt, N.R., Gibbins, N., Glaser, H., Harris, S. and schraefel, m.c. CS AKTive Space or how we stopped worrying and learned to love the Semantic Web. *IEEE Intelligent Systems*. 41-47.
34. Simpson, R., Renear, A., Mylonas, E. and Dam, A.v. 50 years after "As we may think": the Brown/MIT Vannevar Bush symposium. *interactions*, *3* (2). 47-67.
35. Sinclair, P.A. *Sculpteur mSpace Ontology Browser*, Scupteur (http://www.sculpteurweb.org/), 2004-2005, http://www.ecs.soton.ac.uk/~pass/sculpteur/prototype.html.
36. Trigg, R.H. Guided tours and tabletops: tools for communicating in a hypertext environment. *ACM Trans. Inf. Syst.*, *6* (4). 398-414.
37. Trott, M. and Trott, B. *A Beginner's Guide to TrackBack*, MoveableType.org, 2001-2004, http://en.wikipedia.org/wiki/Trackback.
38. Tudhope, D., Binding, C., Blocks, D. and Cunliffe, D. *Compound descriptors in context: a matching function for classifications and thesauri*. ACM Press, Portland, Oregon, USA, 2002.
39. Yee, K.-P., Swearingen, K., Li, K. and Hearst, M., Faceted metadata for image search and browsing. *CHI conference on Human factors in computing systems*, (Ft. Lauderdale, Florida, USA, 2003), ACM Press, 401-408.
40. Zellweger, P.T., Regli, S.H., Mackinlay, J.D. and Chang, B.-W., The Impact of Fluid Documents on Reading and Browsing: An Observational Study. In *CHI 2000*, (The Hague, Amsterdam, 2000), 249 - 256.