User Interaction Experience for Semantic Web Information

David F. Huynh, Dennis Quan, David R. Karger
MIT AI Laboratory/LCS, 200 Technology Square, Cambridge, MA 02139 USA
{dfhuynh,dquan,karger}@ai.mit.edu

ABSTRACT

The Semantic Web project [2] aims to add semantics to the existing World Wide Web. We propose an extension of the Web's user interaction experience to take advantage of the added semantics. This user experience maintains the Web's original navigation paradigm, although all URIs, not just URLs, can be used to address information objects. URIs form the superset of URLs and can name resources other than just those with retrievable contents. Information is displayed in webpage-like presentations in which each UI element is associated with the information object that the element represents. UI elements serve as proxies through which the user can manipulate information objects. Uniform support for direct manipulation complements the Web's navigation paradigm to create an information-centric environment for interacting with information. In addition, we advocate the use of small cooperative tools over large standalone applications to further promote this information-centric paradigm.

Categories and Subject Descriptors

H.5.2 [**Information Interfaces and Presentation**]: User Interfaces – *graphical user interface, interaction styles*.

General Terms

Design, Human Factors.

Keywords

Direct manipulation, information management, metadata, object oriented, RDF, Semantic Web, user experience.

1. INTRODUCTION

The wide acceptance of the World Wide Web arguably owes much to the simplicity of its user interaction experience. A lot of information can be retrieved through mnemonic URLs or labeled links. Using the Web involves mainly hopping between webpages through hyperlinks. Now, the Semantic Web project [2] seeks to improve that experience through more automation by adding semantics to the information on the Web. As this effort gains traction, we ask how this Web user interaction experience can be adapted to take advantage of the added semantics.

Several tools have been built for visualizing and editing Semantic Web data and schemas: Ontology editors such as Protégé [3] and Ont-o-mat [4] allow ontology modeling experts to enter information according to specific ontologies with a high degree of precision. Graph-based viewers show detailed interconnections within semantic networks. Finally, schema-specific user interfaces are customized for managing data in specific domains. While ontology editors and graph-based viewers are too generic to be convenient, schema-specific user interfaces are too specific to generalize and scale. We find these tools lacking to serve as the successor of the web browser.

WWW 2003, May 20-24, 2003, Budapest, Hungary.
ACM xxx.

In this paper, we propose an extension to the Web's user interaction experience that takes advantage of the semantics added by the Semantic Web project. The next three sections discuss this extension and illustrate the interaction experience through an implementation on our Haystack information management platform [5].

2. URI-BASED NAVIGATION

Our user interaction experience adopts the already familiar Web navigation paradigm for browsing through Semantic Web information. While Web contents can be addressed by URLs, Semantic Web contents can be addressed by URIs.

URIs are used by the Resource Description Framework (RDF) [1], a core Semantic Web technology, to name information objects independent of their physical storage location and binary representations. By imposing a unified naming scheme, RDF allows several unrelated schemas to be applied to a common object through its URI, and consequently opens doors for prolific sharing of data among different software applications. One application can refer to an object whose storage is managed by another application through the object's URI. Alternatively, several applications can manage different aspects of a single common object: address book software can manage the contact information of the same person whose financial statements are managed by financial software. It is this capability and others of RDF that make it suitable for modeling much of the user's information [5]. In Haystack, URIs can identify objects ranging from text documents to contact objects to audio soundtracks.

Figure 1 illustrates Haystack's resemblance to the Web browser: note the usual Back, Forward, Refresh, and Home buttons on the toolbar. Below the toolbar are the title pane and the content area. When a URI is entered into the Go address box, the content area displays a page of information about the object named by that URI. If the URI is a valid URL, the web page addressed by that URL is shown. Otherwise, Haystack puts together a page of information relevant to the URI. This process resembles the generation of dynamic web pages from data in server-side databases. In Haystack, the data can come from various places including the local machine or remote information sources.

3. SEMANTICALLY-AUGMENTED WEBPAGES

Although the pages of information put together by Haystack resemble web pages, they are actually augmented with the semantics of the information from which they have been constructed. They contain more than text and links: every UI element (e.g. text span, image) in them is bound to some underlying information object that it represents. The UI element can serve as a proxy through which the user can manipulate the underlying object. Manipulations take the form of drag and drop and invocation of context menus listing applicable operations. Figure 1 shows the context menu resulting from right clicking some piece of text representing a music album.

The binding of UI elements to information objects is implemented using the concept of *view*. A view is a way of presenting objects with certain characteristics. Given an object to present, Haystack examines its characteristics and the context in which it is to be shown and finds the appropriate view to present it. Each object may have more than one suitable view—each view is appropriate for certain contexts: a summary view is suitable in limited screen space, a full view is appropriate when all relevant details are needed, etc.

A view of one object may include the views of other relevant objects. In Figure 1, the view of the media piece named "My Heart Will Go On" includes views for the album "Titanic", the artist "Celine Dion", etc. The entire Haystack UI is constructed by nesting views within one another. Each view remembers which underlying information object it is presenting.

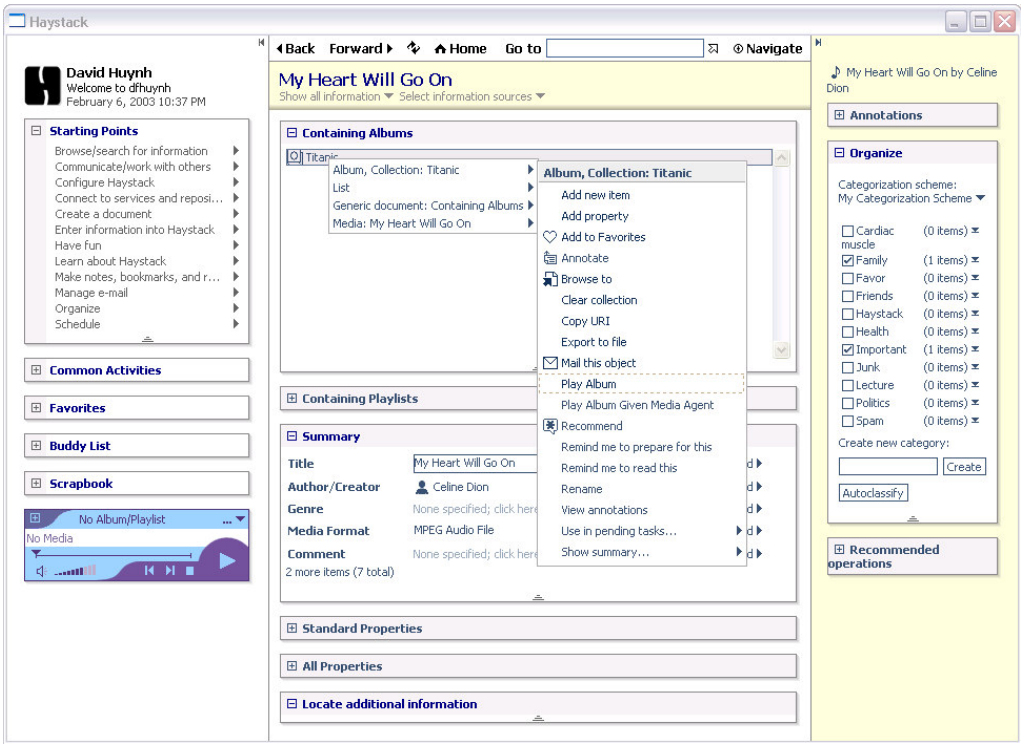

Figure 1. Haystack screenshot

When the user initiates a UI action (e.g. right click, drag), Haystack can systematically enumerate all views that enclose the screen location of the action and trace back to their associated information objects: the user wants to interact with one of these objects. Figure 1 shows Haystack suggesting four different objects upon the user right-clicking on the text "Titanic". The best candidate, the album, is listed first while the most unlikely candidate, the media piece, is listed last.

Uniform support for direct manipulation (e.g. context menus, drag and drop) complements the URI-based navigation paradigm to create an information-centric environment for interaction with information. Information can be called upon by URIs or links, and they can be operated on in place. There is no need to explicitly open an application in order to perform an operation. Rather, operations are invoked directly on each object through its context menu; invocation of an operation could lead to a custom UI through which the operation can be completed.

4. SMALL, COOPERATIVE TOOLS

To promote information-centric interactions, we recommend against monolithic standalone applications with prepackaged sets of features. Instead, we advocate the use of small tools that can be used together easily to accomplish complex tasks. Each tool should be designed to work on all information having characteristics with which the tool is applicable. With proper wiring, context menus can automatically list tools (or operations) applicable to right-clicked objects. For instance, Figure 1 shows the "Play Album" operation for the album object.

Certain tools are generic and can work on all types of information. For instance, using Haystack's "Organize" tool shown on the right hand side of Figure 1, any object can be classified into zero or more categories. In contrast, most of today's applications that allow organization provide their own implementations of folder hierarchies. These various implementations are slightly different, forcing the user to adapt to them. In addition, information from various applications cannot be organized together in a common, unified hierarchy. Haystack breaks down barriers between applications to allow information to be managed together in more sensible ways.

5. ACKNOWLEDGMENTS

This work was supported by the MIT-NTT collaboration, the MIT Oxygen project, and IBM.

6. REFERENCES

[1] Resource Description Framework (RDF) Model and Syntax Specification. http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/.

[2] Berners-Lee, T., Hendler, J., and Lassila, O. "The Semantic Web." Scientific American, May 2001.

[3] Eriksson, H., Fergerson, R., Shahar, Y., and Musen, M. Automatic Generation of Ontology Editors. In *Proceedings of the 12th Banff Knowledge Acquisition Workshop, Banff, Alberta, Canada*, 1999.

[4] Handschuh, S., Staab, S., and Maedche, A. CREAM—Creating relational metadata with a component-based ontology-driven annotation framework. K-CAP '01.

[5] Huynh, D., Karger, D., and Quan, D. (2002). "Haystack: A Platform for Creating, Organizing and Visualizing In-formation Using RDF." Semantic Web Workshop, The Eleventh World Wide Web Conference 2002 (WWW2002). http://haystack.lcs.mit.edu/papers/sww02.pdf.