Toward hypermedia support for information relationship management

Chao-Min Chiu

National Kaohsiung First University of Science and Technology, Taiwan

Michael Bieber

New Jersey Institute of Technology, USA

Received 6 September 1999
Revised 9 September 2000
7 November 2000

Abstract.

The overall research goal of this project is to provide hypermedia functionality to information systems (ISs) not enhanced with hypermedia with minimal or no changes to the ISs. ISs dynamically generate their contents and thus require some *mapping mechanism* to automatically map the generated contents to hypermedia constructs (nodes, links and link markers) instead of hypermedia links being hard-coded over static contents. No systematic approach exists, however, for identifying information relationships and building mapping rules to infer useful links that give users direct access to the ISs' primary functionality, give access to meta-information about IS objects, give access to relationships among information objects and enable annotation and *ad hoc* (user-declared) linking. This paper contributes a procedure for analyzing ISs and building mapping rules that supplement ISs with hypermedia support, which results in new ways to view and manage the IS's knowledge and information relationships.

Correspondence to: C-M. Chiu, Department of Information Management, National Kaohsiung First University of Science and Technology, 1 University Road, Yenchao, Kaohsiung, Taiwan. Tel: +886 7 601 1000, ext: 4117. Fax: +886 7 601 1042. E-mail: cmchiu@ccms.nkfu.edu.tw Website: http://www.nkfust.edu.tw/~cmchiu

1. Introduction

Many information systems (ISs) do not take full advantage of linking and navigation. This happens for several reasons. In part, it has not occurred to many designers and developers to incorporate hypermedia functionality (i.e. an extremely rich layer of links and other navigation, structuring and annotation functionality) [7]. Most designers and developers do not have a hypermedia mindset; they and their users have seen few examples and do not demand this functionality yet. In part, people do not have the time to reengineer existing applications, especially when migrating them to the World Wide Web (WWW) with other projects waiting in the wings. Developers also have few tools and techniques for designing and incorporating hypermedia functionality easily. Developers will not do this until it is natural to conceive and easy to implement. These reasons constitute much of the motivation for our research.

Our overall research goal is to provide hypermedia functionality to ISs not enhanced with hypermedia with minimal or no changes to the ISs. ISs dynamically generate their contents and thus require some *mapping mechanism* to automatically map the generated contents to hypermedia constructs (nodes: documents and screens; links: commands and relationships; link markers: for selecting links) instead of hypermedia links being hard-coded over static contents [10, 11].

Hypermedia can be viewed as the science of relationship management – structuring, annotating and navigating information through relationships, i.e. links [8]. What benefit do users gain from providing ISs with hypermedia support? Users may find it difficult to understand and take advantage of the myriad of interrelationships in an IS's knowledge base (data, processes, calculated results and reports). Hypermedia helps by streamlining access to, and providing rich navigational features around, related information, thereby increasing user comprehension of information and its context [4, 5]. Augmenting an IS with hypermedia support results in new ways to view and manage the IS's information relationships and knowledge, by navigating among items of interest and annotating with comments and relationships (links) [5, 6].

No systematic approach exists, however, for identifying information relationships and building mapping rules to infer useful links that give users direct access to the ISs' primary functionality, give access to meta-information about IS objects, give access to relationships among information objects, and enable annotation and *ad hoc* (user-declared) linking. This paper contributes a procedure for analyzing information systems and building mapping rules that supplement ISs with hypermedia support, which results in new ways to view and manage the IS's knowledge and information relationships.

This paper is organized as follows. In Section 2, a conceptual architecture to interface ISs with the WWW is proposed. In Section 3, some cases that do not meet the authors' criteria for integration with ISs are discussed. Section 4 presents a procedure for building mapping rules. In Section 5, there is a discussion on how to evaluate the conceptual architecture. Section 6 includes future research plans and a general discussion of the issues. Section 7 concludes with a larger view of the research.

2. System architecture

Apart from this research, no system has been identified which automatically generates links and other hypermedia functionality for general ISs. This paper contributes a framework for analyzing what such support entails. The framework identifies seven logical components that any architecture implementing this needs to incorporate. The framework also helps in evaluating existing and proposed systems in terms of their potential for automatically generating hypermedia support [10].

The proposed architecture emphasizes the integration of the WWW with ISs, providing hypermedia functionality to each IS. A novel feature about the architecture is the use of wrappers that contain mapping rules for automatically generating an extremely rich layer of links and other navigation, structuring and annotation functionality. Fig. 1 sketches the architecture with three example ISs. Different implementations may implement these logical components in different ways. Since the WWW browser and server are normal WWW components, only the functionality of other components will be described.

(1) An IS is an application system with which users interact to perform certain tasks. ISs typically produce output content for display dynamically. IS instances are things written within an application package, such as an individual worksheet or database.
(2) An IS wrapper translates and routes messages between its IS and the WWW server. An IS wrapper also provides information to map links and nodes from the outputs of its IS. An IS wrapper must map commands selected by the user on the WWW browser to the corresponding IS commands and invoke its IS to execute them. A comprehensive IS wrapper will allow us to integrate an existing IS with few or no changes.
(3) The knowledge base stores commands for accessing various relationships on IS objects and meta-information about IS objects and relationships that cannot be accessed directly from ISs (e.g. relationships in entity-relationship diagrams).
(4) The linkbase stores user-created annotations and *ad hoc* links.
(5) The master wrapper coordinates relationship mapping and message passing among different IS application domains, thus aiding IS-to-IS integration. It provides the following functionality:
(a) decodes attributes (e.g. object type, object identification (ID) and command) that underlie a link anchor;
(b) searches the knowledge base for commands that implement various relationships from the selected IS object based on the system name and object type;
(c) maps commands to link anchors; and
(d) forms a hypertext mark-up language (HTML) document that includes the mapped link anchors and sends the document to the WWW server. Note that the master wrapper can be implemented with common gateway interface (CGI) programs, Java servlets, active server pages (ASPs), etc.

To integrate a new IS with the WWW and provide it with hypermedia support, one has to build a wrapper, declare mapping rules within it and the master wrapper and store information (e.g. commands and access information) in the knowledge base. One set of mapping rules can serve all instances of an IS.

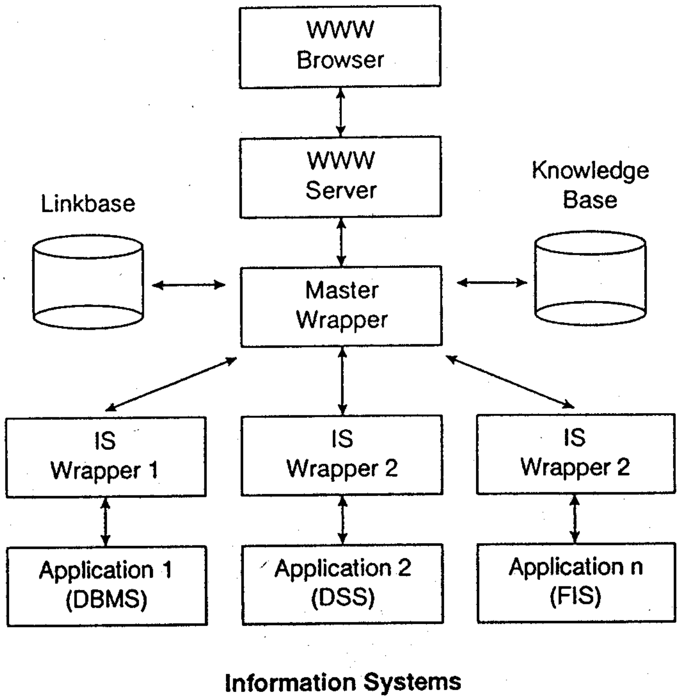

Fig. 1. A framework for integrating ISs into the WWW.

3. Supplementing ISs with hypermedia support

Many approaches exist for integrating ISs into the WWW, such as CGI, ASP and Java servlets used to interface the WWW server and external resources (e.g. databases). A WWW server to process users' inputs and dynamically generate HTML documents invokes back-end programs. No systematic approach exists, however, for dynamically supplementing an information system with hypermedia support through the WWW and giving users direct access to its interrelationships. Many ISs' resources have been made available to users through the WWW. However, those implementations do not meet the authors' criteria for integration with ISs, since they do not infer useful links that give users direct access to the ISs' primary functionality, give access to meta-information about IS objects, give access to relationships among information objects and enable annotation and *ad hoc* links. For example, the following cases do not meet the criteria.

(1) Most WWW database applications handle queries and generate HTML documents from query results without mapping query results to useful links that allow users to view and manage the contents of the database management systems (DBMSs), navigate among related items of interest or annotate with comments and links.
(2) WWW search engines use simple mapping mechanisms to map a query result to a dynamically generated Web page with links based on the 'URL address' and 'page title' for each hit.
(3) The Web interface definition language (WIDL) [2] is an application of the extensible mark-up language (XML) that provides interfaces and services to automate the process of information access to remote applications and systems. WIDL services are like CGI scripts or other back-end Web server programs. A service takes input parameters, performs some processing and then returns a dynamically generated HTML, XML or text document to the Web browser for display. However, the WIDL does not infer useful links that allow users to access information relationships among, and meta-information of, IS objects, navigate among items of interest, annotate with comments and create *ad hoc* links.
(4) Data mining is a process of uncovering relationships, patterns and trends among huge volumes of data and then transforming them to valuable information that can leverage business intelligence and improve the process of making decisions [17]. Data mining tools include IBM's Intelligent Miner, NetGenesis's NetAnalysis, SAS's Enterprise Miner and Oracle's Darwin. The mapping mechanism and data mining proposed here share the common goal of extracting useful information relationships. Data mining uncovers information relationships from the data warehouse. Again, the mapping mechanism proposed here discovers information relationships over IS objects. Data mining tools provide mechanisms for visualizing analyzed results. Unlike the mapping mechanism proposed here, however, they do not incorporate rich hypermedia functionality that provides an extremely rich layer of links and other navigation, structuring and annotation functionality.

Only two systematic approaches exist for reengineering applications for the WWW: Bieber and Yoo's five-step Relationship-Navigation Analysis (RNA) [8] and the nine-step Relationship-Navigation-Rule Analysis (RNRA) in this paper. RNA has five steps: stakeholder analysis, element analysis, relationship and meta-information analysis, navigation analysis and relationship and feasibility analysis [8]. The RNA is a two-stage approach. First, the software engineer performs an RNA analysis, analyzing the application specifically in terms of its intra- and inter-relationships. Second, a dynamic hypermedia engine (DHE) automatically generates links for each of these relationships and meta-information items at run-time, as well as sophisticated hypermedia navigation techniques not often found on the WWW (e.g. guided tours, overviews and structural query [7]) on top of these links.

The major difference between RNA and these RNRA approaches is that the transition from determining the relationships to implementing the mapping rules is clearly defined. RNA supplements this mapping mechanism. Combining RNA and the mapping rule approach proposed here forms the RNRA technique for engineering applications for the WWW.

4. Relationship-Navigation-Rule Analysis

In this section will be discussed the process of analyzing ISs and building mapping rules. These rules would reside in, and become invoked by, the master wrapper or IS wrapper. When providing large ISs with hypermedia support, a facility that automatically infers useful links from dynamically generated information will be helpful. Mapping rules are such rules that convert dynamically generated information to hypermedia constructs (nodes, links and anchors). Mapping rules make extensive use of three features that Halasz [15] identified among outstanding issues in hypermedia research over ten years ago and which still have not been addressed in many hypermedia or WWW applications:

- creating and manipulating virtual structures of hypermedia components;
- computing over the knowledge base during link traversal; and
- tailoring the hypermedia network [3].

Mapping rules infer useful links that enable users to view and manage objects accessed, handled and generated by ISs. The process of building mapping rules is here divided into nine steps.

Step 1. Identify users

The approach begins with an identification of the system's possible users. ISs converting to Web interfaces might have a much broader range of users than systems without Web interfaces. Knowing the audience of a system helps the developer broadly to determine the entire range of important objects, meta-information, commands and relationships. User analysis will help the developer to focus on specific areas during the following steps.

Step 2. Analyze tasks

This step is to analyze what tasks users will want to perform within ISs, after which the system developer can identify IS objects, commands performed on objects, objects' meta-information and relationships among IS objects associated with these tasks. This step also needs to analyze which functionalities of the IS are appropriate for a Web interface. To integrate with multiple systems, the developer also should understand and identify the tasks users will want to perform among systems. Task analysis provides a basis for steps 3, 4 and 5.

Step 3. Identify objects

This step is to identify all the potential objects in which users might be interested. Objects include all items displayed in Web pages (graphs, labels, data value, etc), as well as objects of the system's internal structure. In a DBMS, objects of the system's internal structure include databases, tables, fields and records. Note that the developer should identify the kinds of objects instead of individual instances.

Step 4. Identify objects' meta-information

Three of the relationships in Bieber and Yoo's RNA generic relationship taxonomy [8, 24] concern relationships about the information object itself: characteristic, descriptive and occurrence relationships. In this paper, a relationship is defined as the association between two or more information objects within or among applications, thus these three relationships are defined as an object's meta-information, instead of relationships:

- *characteristic meta-information*: attributes, parameters, metadata and other background information of the item of interest;
- *descriptive meta-information*: definitions, illustrations, explanations and other descriptive information of the item of interest;
- *occurrence meta-information*: multiple instances/views/uses/transformations of the same object in different parts of a system.

Step 5. Identify relationships among IS objects

Identifying explicit and implicit relationships for system objects forces developers to consider what information users are interested in and then to build mapping rules to access this information. Sometimes, meaningful relationships cannot be accessed directly from ISs, so developers have to declare those relationships and store them in the knowledge base.

Yoo and Bieber [23, 24] identify the following types of relationships for system objects. Each gives the user easy access to some aspect of an object:

- *configuration/aggregation relationship*: connects a part to other parts or a whole functionally or structurally;
- *membership/grouping relationship*: connects a member of a collection to other members or to a whole collection;
- *classification relationship*: connects an item of interest to its instance or class;
- *equivalence relationship*: connects instances of the exact same object to a given item;
- *similar/dissimilar relationship*: connects all items that share some positive or negative degree of similarity;
- *ordering relationship*: provides access to objects sequentially related to the object of interest;
- *activity relationship*: deals with relationships that exist among elements that are involved in some kind of activity;
- *intentional relationship*: connects an item of interest to the goals, arguments, issues, decisions, opinions and comments associated with the item;
- *influence relationship*: provides access to the item over which the item of interest has some kind of influence;
- *socio-organizational relationship*: connects an item of interest to the position, authority, alliance, role and communication associated with the item in a social setting or organizational structure;
- *temporal relationship*: provides access to items temporally related to the item of interest;
- *spatial relationship*: provides access to objects spatially related to the item of interest.

Step 6. Identify commands

Commands underlying the <A> tags give users direct access to various meta-information or relationships on IS objects. After identifying the implementation commands, one can then build mapping rules. The displayed labels for each command may be different from actual system commands. The IS wrapper should pass the actual system commands to its IS.

Step 7. Build knowledge base

Developers have to create a knowledge base that stores commands for accessing various relationships on IS objects. The knowledge base can also store meta-information about objects and relationships that cannot be accessed directly from ISs (e.g. relationships in entity-relationship diagrams).

Step 8. Build mapping rules

This step is to choose an appropriate algorithm and then use it to build mapping rules. Algorithms commonly applied to machine learning include neural networks, decision trees, clustering, inductive logic programming, symbolic rule-learning algorithms and Bayesian learning algorithms [17].

The main purpose of mapping rules is to infer useful links from the output dynamically generated by an IS. These rules would reside in, and become invoked by, the master wrapper or IS wrapper. Note that one set of mapping rules can serve all instances of an IS. Two types of links are identified: object links and command links. Command links contain commands for operating on IS objects (see Fig. 2). Object links are links that represent IS objects (see Fig. 3). Each is explained by using a relational DBMS as the target IS.

Command links

When a user selects a link representing an IS object, mapping rules will infer commands for operating upon the selected object based on the system name and object type. For this process, mapping rules should provide the following functions:

(1) search the knowledge database for commands accessing various relationships on the selected IS object;
(2) map commands to links;
(3) form an HTML document that includes mapped links and send the document to the Web server.

For example, when a user selects an object link representing a database table, mapping rules will execute the aforementioned functions and create an HTML document. Fig. 2 lists some commands. In reality, the system would present all commands (or a filtered subset).

In this example HTML document:

(1) the object ID 'Books,Customers' means the 'Customers' table of the 'Books' database;
(2) the IS wrapper is an ASP application called 'MasterWrapper'. Note that the link anchor has a 'Command' attribute with the value 'list_fields'. This type of link anchor is referred to as a 'command link anchor';
(3) the 'System' parameter is used to discriminate among different ISs.

```
<HTML><BODY>
.....
<A href="http://163.18.21.246/fis/MasterWrapper.asp?System=DBMS&ID=Books,Customers&
Type=Table&Command="list_fields">list fields</A>
<A href="http://163.18.21.246/fis/MasterWrapper.asp?System=DBMS&ID=Books,Customers&
Type=Table&Command="show_comments">show comments</A>
.....
</BODY></HTML>
```

Fig. 2. Some command links for operating on a database table.

Object links

When a user selects a link representing a command, mapping rules will execute the command and then infer links from the output generated by the IS. For this process, mapping rules should provide the following functions:

(1) map commands to actual IS commands;
(2) send actual commands and other parameters to the IS;
(3) receive display output from the IS;
(4) infer links from the output generated by the IS;
(5) create the HTML document with inferred links and send the document to the Web server.

There follows an example in which the command accesses a structural relationship. Mapping rules will execute the five aforementioned functions and send the resulting HTML document to the Web server. Fig. 3 lists some fields.

Note that the link anchor has a 'Command' attribute the value of which is 'No'. This indicates that it does not provide a command. This type of link anchor is referred to as an 'object link anchor'.

Step 9. Identify possible navigational tools

Hypermedia researchers have developed several hypermedia features to help users to easily navigate information and reduce cognitive overhead and disorientation (i.e. becoming 'lost in the hyperspace' [13]). This support includes backtracking [19], history lists [21], guided tours [20], overview diagrams [12, 20], paths [20], structure-based query [15, 16], timestamps [18], footprints [18], fisheye viewer [14], annotation [1, 9], bi-directional links [22], multi-destination ('n-ary') links [22], etc.

Developers should decide which navigational tools the system should provide based on users' characteristics, the way that users perform tasks with systems, the IS's features and available Web technologies.

5. Evaluation

This paper proposes a system architecture and contributes a procedure for analyzing ISs and building mapping rules, which form the basis of the authors' future research. There are four indicators for evaluating whether a system satisfies the original design goal of hypermedia support for information relationship management:

(1) inference capability of the mapping rules for finding useful information relationships;
(2) availability of the navigational tools for helping users easily navigate information and adding comments;

```
<HTML><BODY>
.....
<A href="http://163.18.21.246/fis/MasterWrapper.asp?System=DBMS&ID=Books,Customers,Name&
Type=Field&Command=No">Name</A>
.....
<A href="http://163.18.21.246/fis/MasterWrapper.asp?System=DBMS&ID=Books,Customers,Name&
Type=Field&Command=No">Address</A>
.....
</BODY></HTML>
```

Fig. 3. Some object links for representing database fields.

(3) user-friendliness of the user interface (Web page) for accessing original IS functionality;
(4) accessibility of meaningful meta-information about IS objects.

Proposed here are four ways to evaluate the conceptual system architecture: cross-platform, extensibility, interoperability and integration capability. Cross-platform support concerns whether system components can run on heterogeneous operating systems. Extensibility concerns whether new components and data formats can be added to the conceptual architecture and whether new functionality can be added to components. The master wrapper can provide multiple hypermedia services, including navigation, annotation, inference, etc. It is suggested that a component-based approach should be used to implement each hypermedia service. Interoperability concerns whether hypermedia service components within the master wrapper can communicate with each other effectively and completely.

The main strength of this conceptual system architecture is the proposal for using wrappers to integrate with ISs with minimal or no changes to the ISs and automatically provide hypermedia support. Integration capability concerns the communication capability of the master wrapper with the linkbase, knowledge base and IS wrappers and of an IS wrapper with its IS application. Do all modules and components access and share data and functionality as intended? Many approaches exist for building the master wrapper and IS wrappers to integrate ISs into the WWW, such as CGI, Java servlets, ASPs, hypertext preprocessor (HP) and Java server pages (JSPs). There are also various protocols to support system components to communicate with each other, including common object request broker architecture (CORBA), distributed common object model (DCOM), distributed computing environment (DCE), transfer control protocol/Internet protocol (TCP/IP) sockets, Apple Events, remote procedure call (RPC), object linking and embedding (OLE), OLE automation, etc. A good way to evaluate the system architecture is to evaluate the stability, performance, security, portability and backward capability of the server-side programming approach used to build wrappers and whether wrappers support multiple protocols for integrating multiple IS applications.

6. Future work

There are three directions for the future research of the authors:

(1) extending the mapping rules with the newly emerging Web standards for hypermedia linking will be explored: XML linking language (XLink) and XML pointer language (XPointer). Here, the authors would contribute as pioneers in exploring how these can support the kind of dynamic hypermedia support that this system provides;
(2) the types, level and detail of metadata that can be generated automatically and actually be useful will be determined;
(3) there is an intention to develop extensive guidelines for stages 3, 6, 8 and 9 of the RNRA approach: identify objects, identify commands, build mapping rules and identify possible navigational tools. These guidelines will enable designers and developers to implement an integrative architecture quickly.

7. Conclusion

The WWW provides the infrastructure for supplementing ISs with hypermedia support. The authors believe that integrating hypermedia-supported business ISs on the WWW should constitute a major thrust for the WWW research. It will go a long way toward making complex applications more understandable. When reengineering ISs for the WWW or developing new Web applications, dynamic relationship mapping could be an effective way to add additional hypermedia links. This will facilitate adding useful hypermedia functionality to new WWW applications (especially ISs) to view and manage their knowledge and information relationships. It is hoped that this paper will call people's attention to this opportunity.

Acknowledgements

We gratefully acknowledge support for this research by the United Parcel Service, the NASA JOVE faculty fellowship program, by the New Jersey Center for Multimedia Research, by the National Center for Transportation and Industrial Productivity at the New Jersey Institute of Technology (NJIT), by the New Jersey Department of Transportation, by the New Jersey Commission of Science and Technology and by Rutgers University.

References

[1] R. Akscyn, D. McCracken and E. Yoder, KMS: a distributed hypermedia system for managing knowledge in

organizations, *Communications of the ACM* 31 (1988) 820–835.

[2] C. Allen, WIDL: application integration with XML, *World Wide Web Journal* 2 (1997) [Special issue on XML].

[3] M. Bieber, Automating hypermedia for decision support, *Hypermedia* 4 (1992) 83–110.

[4] M. Bieber and C. Kacmar, Designing hypertext support for computational applications, *Communications of the ACM* 38 (1995) 99–107.

[5] M. Bieber, H. Oinas-Kukkonen and V. Balasubramanian, Hypertext functionality, *ACM Computing Surveys* (forthcoming).

[6] M. Bieber and F. Vitali, Toward support for hypermedia on the World Wide Web, *IEEE Computer* 30 (1997) 62–70.

[7] M. Bieber, F. Vitali, H. Ashman, V. Balasubramanian and H. Oinas-Kukkonen, Fourth generation hypermedia: some missing links for the World Wide Web, *International Journal of Human-Computer Studies* 47 (1997) 31–65.

[8] M. Bieber and J. Yoo, Hypermedia: a design philosophy, *ACM Computing Surveys* (forthcoming).

[9] T. Catlin, P. Bush and N. Yankelovich, InterNote: extending a hypermedia framework to support annotative collaboration. In: *Proceedings of Hypertext '89* (1989), pp. 365–378.

[10] C.M. Chiu, Reengineering information systems with XML, *Information Systems Management* 17 (2000) 40–55.

[11] C.M. Chiu and M. Bieber, A dynamically mapped open hypermedia system framework for integrating information systems, *Information and Software Technology* (forthcoming).

[12] G.H. Collier, Thoth-II: hypertext with explicit semantics. In: *Proceedings of Hypertext '87* (1987), pp. 269–290.

[13] J. Conklin, Hypertext: an introduction and survey, *IEEE Computer* 20 (1987) 17–41.

[14] G.W. Furnas, Generalized fisheye views. In: *Proceedings of the ACM CHI '86 Conference on Human Factors in Computing Systems* (1986), pp. 16–23.

[15] F. Halasz, Reflection on NoteCards: seven issues for the next generation of hypermedia systems, *Communications of the ACM* 31 (1988) 836–855.

[16] Y.K. Lee, S-J. Yoo, K. Yoon and P.B. Berra, Querying structured hyperdocuments. In: *Proceedings of the Twenty-Ninth Annual Hawaii International Conference on System Sciences* (1996), pp. 155–164.

[17] T.M. Mitchell, Machine learning and data mining, *Communications of the ACM* 42 (1999) 30–36.

[18] J. Nielsen, The art of navigating through hypertext, *Communications of the ACM* 33 (1990) 296–310.

[19] J. Rosenberg, The structure of hypertext activity. In: *Proceedings of Hypertext '96* (1996), pp. 22–30.

[20] R. Trigg, Guided tours and tabletops: tools for communicating in a hypertext environment, *ACM Transactions on Office Information Systems* 6 (1988) 398–414.

[21] K. Utting and N. Yankelovich, Context and orientation in hypermedia networks, *ACM Transactions on Information Systems* 7 (1989) 58–84.

[22] N. Yankelovich, B.J. Haan, N.K. Meyrowitz and S.M. Drucker, Intermedia: the concept and the construction of a seamless information environment, *IEEE Computer* 21 (1988) 81–96.

[23] J. Yoo and M. Bieber, Towards a relationship navigation analysis. In: *Proceedings of the 33rd Hawaii International Conference on System Sciences* (2000).

[24] J. Yoo and M. Bieber, Finding linking opportunities through relationship-based analysis. In: *Hypertext '00 Proceedings* (2000), pp. 181–190.