Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 1996 Proceedings

Americas Conference on Information Systems (AMCIS)

8-16-1996

What Every Information Systems Developer Should Know About Hypertext

Michael Bieber
New Jersey Institute of Technology, bieber@cis.njit.edu

What Every Information Systems Developer Should Know About Hypertext

Michael Bieber
Institute for Integrated Systems Research
New Jersey Institute of Technology
University Heights - Newark, New Jersey 07102 - USA
telephone: (201) 596-2681
facsimile: (201) 596-5777
electronic mail: bieber@cis.njit.edu
WWW URL: http://hertz.njit.edu/~bieber/bieber.html

My goal in writing this piece is to convince system developers to incorporate hypertext concepts in their design process and hypertext features in their applications. Why hypertext? Why system developers? Why discuss the design process separate from the actual product?

The concept of hypertext grows from a simple but fundamental idea: information is interrelated. The associative link, which so often characterizes hypertext, embodies this idea by connecting any two pieces of information somehow related. The sophistication-and potential-of hypertext lies in applying the idea, i.e., in identifying relationships, items to relate, and how users should access them. (Note that the term *hypermedia* nominally applies *hypertext* concepts to multiple media. Many treat the terms as synonymous.)

(1) What characterizes a useful relationship?Assume the user selects an item of interest in a computer display. Each of the following six relationships gives the user access to some aspect of that item, which he or she may not have ordinarily.

Associative Relationships

Access to the kind of domain-specific relationships one finds in a schema or application design. In a database application, this includes relationships captured in the entity-relationship diagram. Within a decision support system (DSS), one may associate models that operate within the same domain. One also may infer associations among models that share global variables or other components. Within an enterprise support system (ESS), associative relationships include the interorganizational relationships found in an organization chart.

Process Relationships

Direct access to processes and operations. In a work flow support system or ESS, process relationships represent the interconnections among workgroups, departments and software applications handling different aspects of a product or service. Process relationships give access to the next step in a work flow or procedure, subtask in a project management system or document in a series. Process relationships also include operations (e.g., menu commands), logically connecting an object to the result of operating upon it. In a DSS, for example, an explanation link connects an analysis result with an explanation of its calculation.

Structural Relationships

Access to related objects based on the application's internal structure. In a DSS these include links among a model's equations, its variables, and individual data values instantiating the variables. In a project management system, subtasks are structurally related to their parent tasks, to the portion of the timeline that they affect, and to documents produced within them. In a geographic information system (GIS) the elements of a map and their underlying data are related structurally.

Metainformation Relationships

Access to parameters and descriptive information. In a DSS, a data value's system parameters include its size, data type and security classification. Its domain parameters include its attributes. Descriptive information includes its source, owner and purpose.

Occurrence Relationships

Access to all manifestations of a given item. In a CASE tool, for example, these would connect each occurrence of an item in the requirements document, design documents, program code, documentation and system output. In a GIS, one should be able to see every occurrence of a data value or geographic object in all maps, data listings and analysis reports. Occurrence relationships would also link different versions of the same document or datum.

Annotative Relationships

Relationships declared by the user in addition to those above. While all the aforementioned relationships could be generated automatically based on the application or system structure, annotations allow the user to declare anything not inferable or automatable. Both document authors (e.g., analysts) and, security permitting, document readers (e.g., other group members, management, customers and the general public) may be able to annotate items within an application. Annotations range from comments on single items to user-declared *ad hoc* links between items.

(2) Which items should have these relationships?Taking a "user knows best" viewpoint prescribes making every object on the computer screen selectable-data values, calculation results, labels, titles, report text, entire documents, and so forth-and therefore a candidate for annotation and other relationships. Furthermore, users could access meta-level relationships: every relationship listed above could apply to its own type and to every other. For example, team members should be able to annotate each others' annotations. Users-especially developers within a hypertext-supported development environment-should be able to annotate a structural relationship or see which processes are interrelated. Similarly, they should have access to all occurrences of any relationship in the system.
Providing such information supports the goal of many organizations to empower lower-level employees with decision making capabilities. On the other hand, in certain domains developers may deem such access irrelevant or even distracting to the user's task, possibly slowing down time-critical operations. Thus, while a hypertext analysis helps developers think about an application more broadly, in the end the developer must decide which additional information is (in)appropriate.

(3) How should users access these relationships and the items they associate?The user typically selects an item on the computer screen to follow its link or chooses from a list of available links. (One may have to rank, filter or layer links if their magnitude overwhelms the user.) If a link's purpose or destination is not intuitive, its author or the system builder should label it in some way. Hypertext access includes navigational, annotation and view-oriented features. Navigation transports the user among information items. Its features include browsing (following a relationship's link), backtracking, query based on content, and query based on interrelationships. Annotation includes comments, bookmarks (hot list items) and user-declared links. View-oriented features enable navigation through local and global overviews, and along recommended paths and guided tours of interrelated items. An analyst, for example, could prepare an annotated guided tour to explain a completed analysis or other process.

Two important consequences for systems development follow from the hypertext concept. First, we can define hypertext both as the science of relationships and as a technology for information access and relationship management. (This follows from the innovative work in hypermedia design done by Tomás Isakowitz at New York University.) Second, a hypertext philosophy or vantage point emerges for thinking about applications.

The traditional design methodologies of functional decomposition and data flow analysis help us think about a system in terms of its processes and how information passes through it. An object-oriented viewpoint considers the system in terms of its components and component hierarchies, and the operations one can apply to each. A hypertext philosophy helps us think about a system in terms of the relationships among its elements and processes, focusing on how users gain access to them. A hypertext analysis could supplement the developer's standard design methodology, broadening the scope of objects and relationships a system encompasses. In addition, hypertext functionality could supplement the system's feature set.

One even could use hypertext to integrate independent applications for interrelated tasks. Consider the query "how many trucks travel on which routes between Newark and Philadelphia?" The freight databases currently available only contain commodity flows among regions in tons per item. To answer, the analyst normally would extract truck tonnage information from a series of these databases, use an analysis program to convert tonnage to numbers of trucks based on a commodity density analysis, use a second route mapping analysis program, and generate the final report for display on a word processor. A hypertext engine could manage the task for the analyst (as a process relationship), passing appropriate outputs of one application as inputs to the next. At any point, for any component, the user could see the six relationships discussed earlier, including for example, how a particular datum is processed in each of the independent applications.

Lastly, one can evaluate an information system from a hypertext viewpoint. For every object visible on the screen, can the user easily gain access to all related information? Can the user clearly "see" how the information is related and under what context? Do the relationship and context remain clear once the user views the related information? Do the users of a hypertext-enhanced system understand the system, its components and its results, and have confidence in these?

With the increased consciousness of hypertext which the World-Wide Web has brought, developers and the organizations for whom they work have never been more open to considering their computer applications in a new light. Yet few guidelines exist for assisting developers in providing access, i.e., in deciding *what* to link. In addition, the major hypertext delivery vehicle today, the World-Wide Web, provides only rudimentary support for hypertext functionality. Developers must represent all types of interrelationships and construct all advanced hypertext functionality (e.g., guided tours) from scratch using one simple anchor type and single-step forward browsing. And thus, while hypertext as a concept has made major contributions to the fields of collaboration, technical communications and literature, and while both outstanding research-oriented and commercially-available hypertext environments exist, the mainstream information systems development and software engineering fields, for the most part, have not taken advantage of hypertext concepts.

I strongly encourage systems analysts and developers to consider their applications from a hypertext vantage point. Hypertext relationships and functionality give users a greater feeling of control within an application, and a greater understanding of its domain, components and results. A hypertext analysis should play a part in the design of every application with user interaction, and hypertext access should supplement many application feature sets.

References

1) M. Bieber, "Automating Hypermedia for Decision Support," *Hypermedia* 4(2), 1992, 83-110.

2) M. Bieber, "On Integrating Hypermedia into Decision Support and Other Information Systems," *Decision Support Systems* 11, 1995, 251-267.

3) M. Bieber and C. Kacmar, "Designing Hypertext Support for Computational Applications," *Communications of the ACM* 38(8), 1995, 99-107.

4) M. Bieber and S. O. Kimbrough, "On Generalizing the Concept of Hypertext," *Management Information Systems Quarterly* 16(1), 1992, 77-93.

5) T. Isakowitz, E. Stohr and P. Balasubramanian, "RMM: A Methodology for Structuring Hypermedia Design," *Communications of the ACM* 38(8), August 1995, 34-44.