N93-32155

AUTOMATION AND HYPERMEDIA TECHNOLOGY APPLICATIONS

Joseph H. Jupin
Edward W. Ng
Mark L. James
Jet Propulsion Laboratory
4800 Oak Grove Drive
Mail Stop 525-3660
Pasadena, CA 91109

Abstract

This paper represents a progress report on HyLite (Hypermedia Library technology): a research and development activity to produce a versatile system as part of NASA's technology thrusts in automation, information sciences and communications. HyLite can be used as a system or tool to facilitate the creation and maintenance of large distributed electronic libraries. The contents of such a library may be software components, hardware parts or designs, scientific datasets or databases, configuration management information, etc. Proliferation of computer use has made the diversity and quantity of information too large for any single user to sort process and utilize effectively. In response to this information deluge, we have created HyLite to enable the user to process relevant information into a more efficient organization for presentation, retrieval and readability. To accomplish this end, we have incorporated various AI techniques into the HyLite hypermedia engine to facilitate parameters and properties of the system. The proposed techniques include intelligent searching tools for the libraries, intelligent retrievals, and navigational assistance based on user histories. HyLite itself is based on an earlier project, the Encyclopedia of Software Components (ESC) which used hypermedia to facilitate and encourage software reuse.

1.0 INTRODUCTION

Space exploration in the 1990's is faced with an information deluge of increasing magnitude and complexities. First, advanced sensor technology has dramatically enhanced our data acquisition capability. As we consider the series of NASA's planetary probes, the Great Observatories, the Earth Observing System, the Space Station Freedom, and the international solar exploration projects, we foresee a rich endowment with large volumes and complexities of scientific and engineering data. Second, the ease of computer publishing, for better or worse, has produced exponentially increasing amounts of documentation, that may include texts, schematics, images, numbers, and even movies for scientific visualization. Third, software continues to grow in amounts and varieties, approaching a phenomenon called "software crisis" by Iwao Toda, Executive Vice President of Nippon Telegraph & Telephone Corp. in Japan. (Ref. IEEE Software, P. 14, May 1992.)

In order to to assist in the management of this information deluge, a system called HyLIte has been proposed as a means of sorting, tracking and managing many of the various pieces of information that when assembled together form the software project. HyLite also expands beyond the typical project management tool by facilitating the incorporation of hypermedia augmented with Artificial Intelligence (AI) techniques to provide even more functionality for the documentation and presentation of information to the user. This paper introduces the various AI technologies that have been selected for implementation into HyLite and the various benefits and justifications for each.

2.0 HYLITE

The Hypermedia Library Technology is a proposed hypermedia system designed for the management, access, and presentation of information in a hypermedia format. Within HyLite will be various tools that will assist the user in the design and implementation of their own hypermedia application. The user will be able to incorporate video, animation, audio and program examples as well the normal hypertext functionality normally associated with hypermedia systems. Though HyLite is still in the conceptual state, a tool which will form the backbone of the system already exists and is in use today. This tool is the Encyclopedia of Software Components (ESC) [1] , which was specifically designed for software re-engineering and reuse. However, as more applications are pursued, variations of ESC will developed that are tailored for different application domains.

2.1 ESC

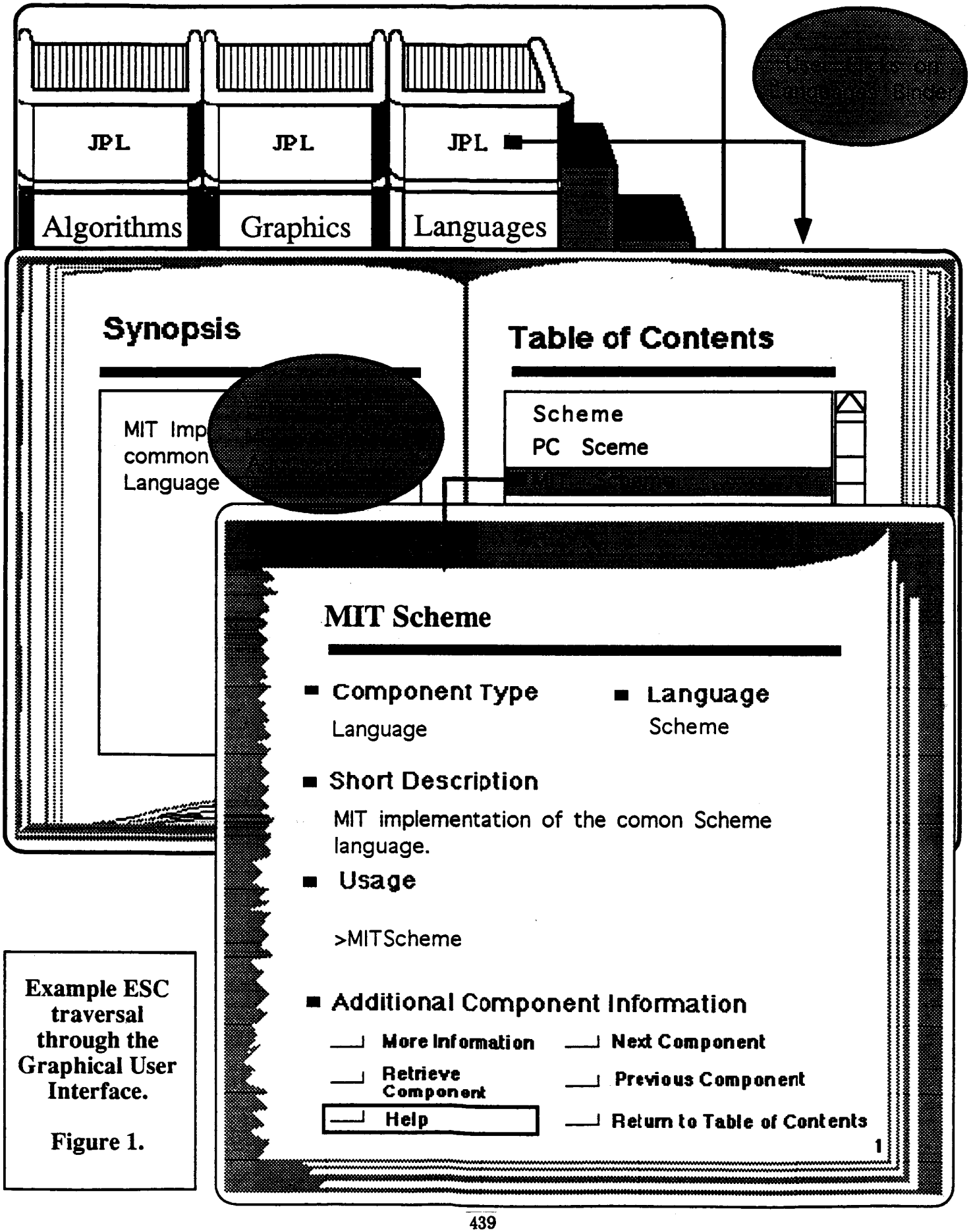
The ESC (Figure #1) is a hypermedia tool designed to encourage software reuse. To this end, many of the features of the tool have been designed to assist the user in locating, retrieving, and browsing for various software artifacts. Much like the encyclopedia metaphor for which the tool is based, a user browses a software encyclopedia looking for various pieces of information (in this case software artifacts) that will assist him in his development efforts. The user is presented with information on each component as he traverses the classification hierarchy. This information includes such items as language type, hardware and software dependencies, animations, graphical representations, and even examples of how the software component is called (calling procedure). All these items allow for the user to arrive at an intelligent decision on whether the software artifact is useful for his project. When arriving at a component that meets the user's requirements, the user merely presses a button and the software is automatically retrieved.

It should be noted that the actual software may or may not reside within a local depository. Therefore, the retrieval mechanism incorporated into the ESC accesses the Internet looking for the requested software artifact. The ESC knows where the software is located based on information within the component description which details not only the software's whereabouts, but also the necessary files needed for the complete package to run. Much of the confusion associated with software retrieval over the Internet is removed from the user's view which adds a level of ease that many users would find appealing.

The previous retrieval was based on the use of the graphical user interface (GUI) representation of an encyclopedia. There also exists within the ESC the ability to execute SQL-like queries on the database to retrieve software. In this case, this feature is designed for advanced users who know the specific item(s) or areas to search for. This functionality is the result of the classification scheme of the ESC (based on semantic networks and frames). A user even has the ability to rearrange the encyclopedia to reflect a specific area of interest (i.e., show volumes based only on language, or graphics, etc).

Another feature of the ESC is the ability to publish new software artifacts as they become available into the ESC so that other users can become aware of their existence. One of the main justifications for the ESC has been the ability to facilitate software reuse by making the search process easier. Therefore, it is imperative that other software artifacts have the capability of becoming a component or volume within the ESC. The publishing engine of the ESC allows a user to describe (classify) the software and submit the information for inclusion into the ESC.

One final feature of the ESC is the ability for the user to develop a user specific version of the ESC that resides on his own hardware. This feature allows the user to create volumes independently on his own hardware and organize and manage the software independently within the ESC. The ESC is a self contained software package that will allow a user to change the encyclopedia's contents to reflect their own interests. Thus, the user can make a personalized version of the ESC.



Example ESC traversal through the Graphical User Interface.

Figure 1.

Given all the functionality of the ESC, it is still important that AI techniques are part of the product so that the software will become even more responsive and flexible for the user. Not only does this benefit the user, but as the ESC is scaled up to become the HyLite system such flexibility and power will be necessary in order for the software to be developed into a fully functional hypermedia authoring/management system.

3.0 INCORPORATION OF AI INTO THE ESC

After the first prototype of the ESC had been completed, research was conducted to determine the feasibility of incorporating AI into the ESC. It was decided that AI could significantly augment the ESC in usability and flexibility. As a result, several areas have been selected for a first pass attempt at inserting different AI tools into the ESC architecture. These areas include navigation assistance for traversing the hypergraph of components, intelligent searchers and intelligent searchers (Figure #2). Each of these have been detailed further below.

- **Intelligent Retrieval**

Currently retrievals must be very specific and based upon predetermined keywords. The keywords reflect the properties that have been used to classify the software and form the connections between the differing components in constructing the classification hierarchy. Thus, the user needs to have a specific idea of the type of artifact needed. In the current system, this is accomplished by the user typing in a keyword (i.e., domain = "Applications") or selecting it from a menu of possibilities. This constraint of keyword driven retrieval needs to be eliminated and a more flexible format adopted.

In response to this problem, it has been suggested that along with the property-component network, there should exist a mechanism for the representation of relationship between properties and components. This would allow a means for intelligent retrievals to occur since the processing engine would be able to infer from the relationships. For instance, a request for Green's Theorem might yield no components from retrieval using the normal keyword search. With the relationship network, the retrieval might suggest other calculus solutions that are approximate to Green's Theorem.

- **Browsing Assistance**

There will be many sessions with the ESC for which the user does not have the context for the desired component. That is, the user is searching blindly for components or even merely casually browsing for anything interesting. To this end, it has been suggested that the creation of a database of user sessions be maintained. From this database, it would be possible for new users to quickly discover relevant components since these paths will have been identified. One possible scenario is one in which a screen pops open displaying a possible candidate at the end of the search path the user is currently navigating. The candidate represents a component that has had multiple visits from other users which translates to a high probability as the object of the user's search.

Another application is specialized sessions based on user login. For instance, if user John Doe consistently browses through the algorithms book of the ESC, then the learning mechanism of the ESC should record this information such that the algorithm book automatically opens for the user as he signs in. Not only does this translate into ease of use for the user, but shows how the system is capable of becoming more responsive to the individual users.

- **Intelligent Spelling Correction**

Spelling errors are a natural phenomena that occur when people interact with a computer. The ESC should be able to correct for common types of spelling mistakes in the context in which they occur. That is if a request is made to retrieve an algorithm for sorting vectors and the word vectors is spelled wrong, then the algorithm should only look for words that are data types that can be sorted. The problem is difficult due to the fact there will be hundreds of thousands of words known by the system. As a result, a spelling correction algorithm cannot iterate over the entire set in a feasible amount of time. In order to delimit the size of the lexicon to be searched, the spelling corrector needs to take into account knowledge about the objects and actions to be initiated. This knowledge if based upon a combination of the classification scheme of the components and an analysis of how actions can be applied to the components (certain items can be sorted, while other items cannot).

- **Overly Specific Requests**

Another problem that will be frequently encountered is an overly specific retrieval request. The user may have a specific idea of what is needed and make an appropriate request. The ESC should be able to process this request not only as a single unit but as sub-sections also. The resulting information will provide the user with more alternatives rather than returning an empty retrieval. That is, if a request is made to retrieve an algorithm that performs a heap sort of two-dimensional arrays and the only algorithm existing within the ESC can sort one-dimensional arrays, then that algorithm should be retrieved with an appropriate message from the ESC. The message should state why the alternate component was retrieved relative to the stated request.

After these tools have been implemented, work is to continue to further research alternate AI technologies for further expansion of the ESC/HyLite's capabilities.

3. CONCLUSION

It is felt that as HyLite and more specifically the ESC gain wider usage, the use of AI tools will significantly facilitate the ease of use of the program. These tools will significantly enhance the retrieval mechanism as well as assist the user in searching for the desired software artifacts. Additionally, the user will gain assistance in the navigation of the large database associated with the ESC. As the ESC transitions over to HyLite, the hypermedia authoring and development system will greatly benefit from AI, also. AI has the potential for incorporation into many differing aspects of the tool ranging from user interface assistance to intelligent file and memory management. Further research is needed to evaluate other potential uses, but initial attempts indicate that the inclusion of AI technology into hypermedia and software engineering tools will significantly enhance the capabilities and ease-of-use of each.

4. ACKNOWLEDGMENTS

We would like to acknowledge Sheldon Shen for his input in this area from a database perspective.

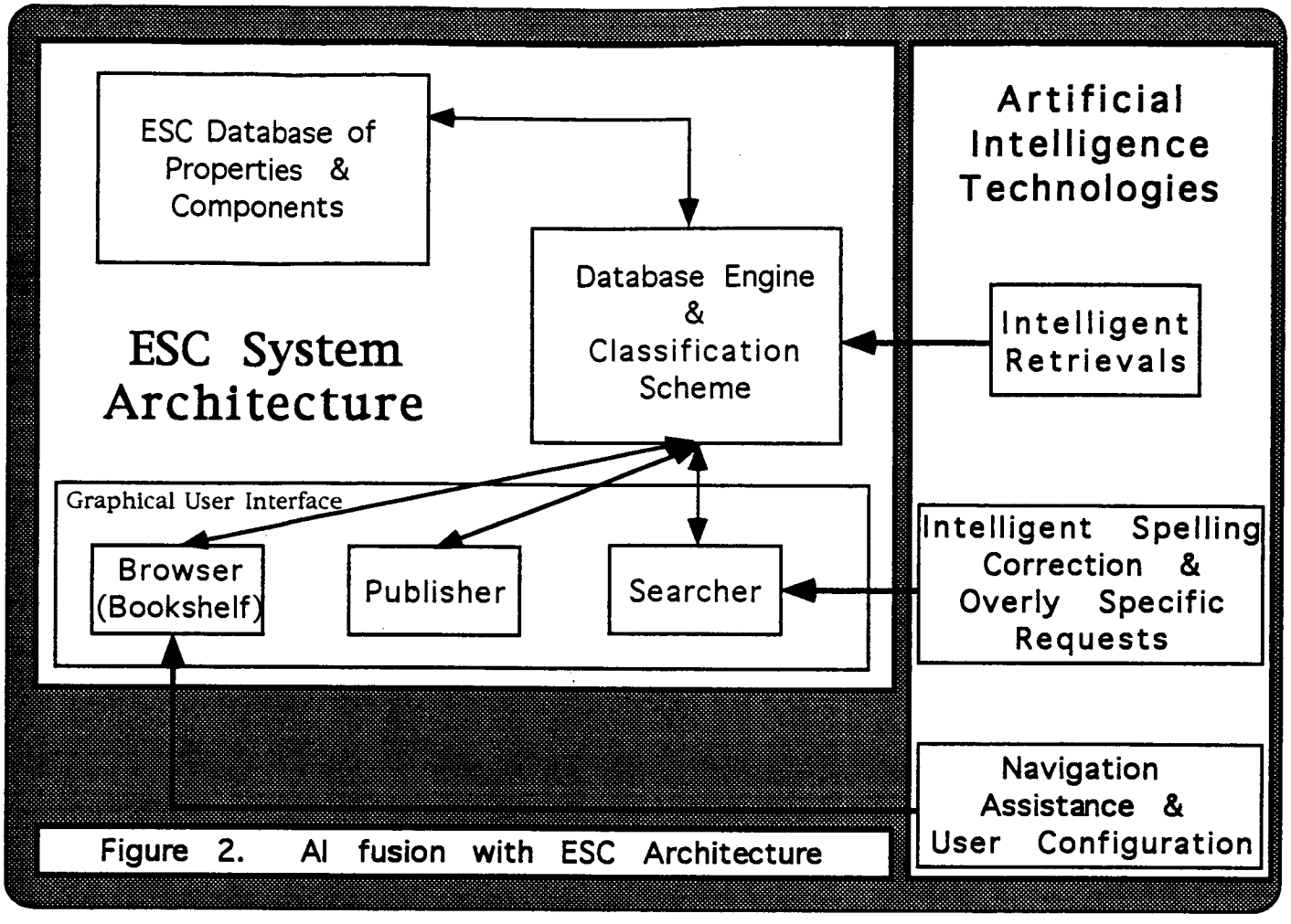


Figure 2. AI fusion with ESC Architecture

5. REFERENCES & BIBLIOGRAPHIES

1. Beckman, B., Jupin, J., Snyder, W. V., Shen, S., Warren, L. V., Boyd, B., Tausworthe, R.C., "The ESC: A Hypermedia Encyclopedia of Reusable Software Components", Information Systems Prototyping and Evaluation Quarterly Report, July 1991, JPL Publication D-8770.

2. Bertsche, S.V. Hypermedia/time-based document (HyTime) and standard music description language (SMDL) user needs and functional specification, X3V1.8M/SD-6.

3. Bullard, L., Price, H., Schoolfield, B., Harlow, R., Dominque, D., Know, E. and Laffin, M. Beyond the Book Metaphor: Concepts for Designing and Implementing Integrated Product Development Environments and Digital Technical Information Services. Obtain from Bruce Schoolfield, GE Aircraft Engines, One Neumann Way MD F126, Cincinnati, OH 45215-6301.

4. Coombs, J.H., Rinear, A.H. and DeRose, S.J. Markup systems and the future of scholarly text processing. Commun, ACM 30, 11 (Nov., 1987).

5. Dykiel, R. Technical requirements for ODA/Hypermedia. ESPRIT project. PODA2, Bull S. A. X3V1.8M/91-2.

6. Englebart, D.C. Knowledge-domain interoperability and an open hyperdocument system. In CSCW 90 Proceedings of the ACM. Available from Bootstrap Project, Stanford University. Doc. (AUGMENT, 132082).

7. Goldfarb, C.F., and Newcomb, S.R. ANSI Project X3.749-D, Hypermedia/Time-based Document Structuring Language (HyTime). X3V1.8M/SD-7 (now ISO/IEC DIS 10744).

8. Markey, B.D. Multimedia and hypermedia standardization, a report from the ad hoc Study Group on Multimedia Standardization to the ISO/IEC's JTC1 TAG. X3V1.8M/90-73.

9. Oak Ridge National Laboratory. Hypertext bibliograph, Apr. 1990. X3V1.8M/90-57.

10. Samuel, R.L., III, Ed. User requirements for hypermedia. X3V1.8M/90-44.