Open Architecture Multimedia Documents

Brian R Gaines and Mildred L G Shaw

Knowledge Science Institute
University of Calgary
Calgary, Alberta, Canada T2N 1N4
{gaines, mildred}@cpsc.ucalgary.ca

Abstract: An open architecture multimedia document publication system is described which integrates a number of different representation technologies to provide a medium offering a wide spectrum of usage, from emulation of current paper publication, through electronic document delivery, multimedia inclusion of video and sound, structured hypermedia linkage, and formal knowledge representation supporting simulation and inference. The research is targeted on exploring new forms of scholarly communication, and the publication system supports collaborative document development, the authentication of disseminated material, and the citation, annotation and reuse of such material. The document publication system provides a rich word processing and page makeup environment with all the facilities normally expected, and adds multimedia, hypermedia and computational facilities incrementally and naturally, with careful attention to the usability of the human-computer interface. The result is an interactive document in which knowledge is represented in a variety of ways, some targeted on human interaction, some targeted on computational analysis, simulation and inference, and such that the document can be printed as a conventional paper or book losing the dynamic aspects of the material but retaining the visual representation.

Keywords: Multimedia information systems, publication systems, electronic books, digital journals, knowledge bases, hypermedia.

Introduction

Advances in software and hardware technology in recent years make it feasible to publish multimedia documents on a basis that is cost-competitive with printed paper, and offers greatly extended capabilities compared with conventional books and journals. It is reasonable to expect that digital publication through Internet and on CD-ROM will become very attractive in the next decade, and at some time will grow in volume to rival paper-based publication. However, paper-based media are currently the mainstream vehicles for high-status publications, and it is unlikely that revolutionary changes in publication practice will take place in the near future. Progress is most likely to be made through parallel publication of paper and electronic versions of the same material, with the paper version justified by its adherence to existing publication standards, and the electronic version justified by its improved facilities for communicating knowledge.

Commercial vendors are already beginning to offer enhanced multimedia capabilities within existing word processing and page makeup packages. However, these are currently focused on audio-visual representation facilities such as Kodak's Photo-CD [18] and Apple's QuickTime [4], and do not offer open architecture support to the very wide range of new active and interactive media that can incorporated in computer-based documents. Active pictures in documents can support animation, simulation, visual languages for programming, concept maps, semantic networks, and other interfaces to a variety of underlying applications. Multimedia document architectures offer the possibility that *any* application that interacts with the user through a window can also interact with the user through an embedded 'picture' in a document.

This generality of multimedia capabilities is both exciting and daunting. It becomes even more so when one realizes that the various applications can be cross-coupled within the document, and can be linked in a variety of interactive ways with the textual components. From this perspective multimedia documents are not just another specialist technology, but rather a highly generalized architecture for integrating a heterogeneous range of information technologies through the familiar user interface of a document.

These are not new insights. In 1945 Bush's account of *memex* gives a requirements specification for multimedia interactive documents [2]. In 1970s at Xerox PARC, Kay and Goldberg [20] brought some of the functionality of memex into being as *Dynabook*, Weyer investigated an encyclopedia dynamic book [31], and Gould and Finzer investigated educational applications [15]. Bier and Goodison [1] presented documents as an attractive form of general user interfaces at EP90, and Pasquier-Boltuck, Grossman and Collaud presented an object-oriented open-architecture implementation of the electronic book, *EBook3* [26], at ECOOP88 [25]. The major technological change in recent years supporting multimedia publication is the massive cost decline in CD-ROM production such that a single CD-ROM can be produced for $25, and production runs of a 1,000 for less than $3 each. This, coupled with the availability of powerful, low-cost lap-top computers, is making digital documents a feasible publication medium competitive in cost and usability with printed paper and offering interactive multimedia capabilities.

This paper describes the features, architecture and implementation of KWrite, an interactive multimedia document publication system, designed to support parallel paper and CD-ROM publication of scholarly books and journal papers. The system has been designed to appear to the user as a conventional word processor, but to have an open architecture enabling a wide range of interactive applications to use the document as a user interface while appearing seamlessly embedded to the user. This approach has been taken because it seems unlikely that any specific multimedia document program will ever encompass all the functionality that authors might reasonably require. What is needed is an open architecture document 'shell' supporting integration with independent applications that were written with more conventional user interfaces in mind.

System Requirements

The major motivations for the KWrite development described in the previous section lead to a number of system design and implementation requirements. It should be:

1. *Similar to standard word processors in user interface and functionality.* To minimize impediments to use it is essential that the new technology deviate as little as possible from familiar technology, and that enhanced features are incorporated seamlessly and naturally within the existing framework. Commercial word processors and page makeup systems emulate the appearance of the printed page and the keyboard of the typewriter. Acceptable multimedia systems have to build on this, and users' other experience of computer interaction. Some additional interface features will usually be necessary, but these need to be carefully designed to be coherent and consistent with the existing document interface.
2. *Capable of supporting hypertext and hypermedia linkage, color graphics and photographs, audio and video.* User expectations of interactive multimedia systems are already informed by many past developments such as *Intermedia* [33], and the availability of QuickTime in word processors. Again, acceptable multimedia systems have to build on this and provide the expected functionality.
3. *Capable of supporting active data and knowledge structures for simulation, animation and inference*. As discussed above, *any* representational form provides a multimedia extension. The forms of activity and interactivity available define a new medium as much as do its modes of audio-visual access.
4. *Open architecture such that an area in a document can be treated as a window pane by another application.* The natural junction between an interactive computer application and a document is the picture frame within a document. Conventional pictures contain tables, diagrams, mathematical and chemical formulae, photographs and so on. The picture areas interact typographically with the text so that it runs around or jumps over them in page makeup. They are 'windows' in the documents that naturally transform to become windows into another application.
5. *Open architecture such that the structure and contents of any document component may be accessed by another application.* The natural interface to supply parameters to an external application is the content of the existing document. It may be necessary to supply additional interface capabilities for particular applications but, as far as possible, the parameters required should be part of the document.
6. *Versioned such that the authenticity and integrity of the original document is preserved while being fully editable by the reader*. This is not a standard issue in multimedia publication, but it is of vital importance to the publication of scholarly documents [8]. It is essential to scholarship that documents received are precisely those published by their authors so that citations and critical commentaries by others reference a well-defined publication. However, it is also important to the effective use of electronic documents that full advantage may be taken of their active availability, for example, for annotation and personal commentary.

Another consideration in developing the multimedia document publication system was the need in CD-ROM publication for very low cost publication software. Until universal standards are available and commercially supported, systems will need to be available that can be supplied with the CD-ROM at virtually zero cost if they are not to constitute the dominant production cost.

System Architecture

KWrite is implemented as a class library in THINK C on the Apple Macintosh platform. As with *Dynabook* [20], *Intermedia* [23], *EBook3* [25], and other compound document systems [17], the object-oriented library implementation is used to enable principled software engineering to provide a very open and flexible environment that is easy to enhance and maintain. The dependence on the Macintosh is a limitation to usage on other platforms that will eventually have to be overcome, but it is currently important in giving access to the excellent typographic, graphic, sound and vision managers that the Macintosh supports. In particular, the availability of standard formats for multimedia material supported by a wide range of commercially supported editors has enabled us to provide users with much greater functionality than would have been possible if we had to support proprietary formats and provide such tools as an integral part of the publication system.

The logical unit supported by the system is a generalized 'document' with a highly open architecture providing links to material in a wide range of media as shown in Figure 1, and itself being accessible both to human readers and computer programs. At the top level the structure of the documents conforms very closely with those of conventional word processor and page layout documents. All the enhancements are linked at lower levels to existing features of a conventional document. This is not only a convenience in implementation but also an important contributor to the naturality of the user interface since the user's cognitive model of the document is extended rather than violated.

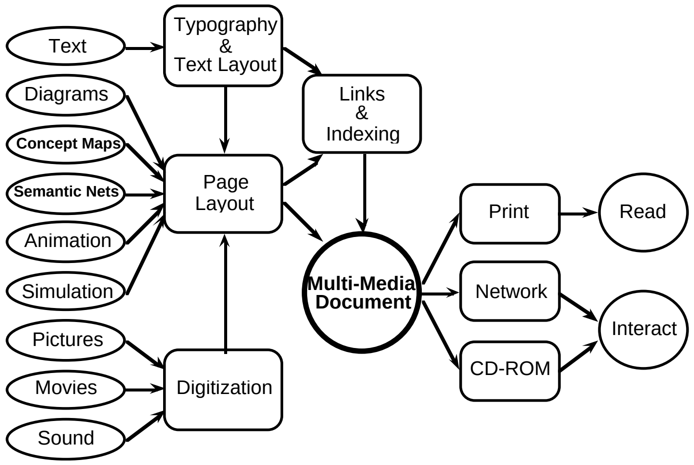


Figure 1 Multimedia document production

Figure 2 shows the internal document architecture with a flow from a top level of conventional features to a lower level of multi-media, hypertext, and application integration extensions.

In the left column, the *text component* of the document is represented very simply as a linear string of characters in Apple's international script format. No style, pagination, or other material is embedded in this basic text structure which makes it as simple as possible to search, index and analyze.

Open Architecture Document				
Text Component	Style Component	Layout Component		
		Page	Paragraph	Pane
Linear string of text in international script format No embedded material, simple to search, index and analyze	Pointers to strips of text Font size, face and style control Links to text and sound annotation Links to sections of this and other documents	Width, height, columns, border, header, footer Displayable and printable unit	Width, indentation, tabulation, line spacing, border Version derivation tracking	Width, height, position, border Links to material displayed in panes and associated applications Bit map graphics, line graphics, quicktime videos, visual knowledge structures

Figure 2 Internal document architecture

In the second column, the *style component* is represented by pointer and length structures delimiting strips of text in the text component. The attached data structures define the normal typographic enhancements of the text, such as font size, face and style. They also support links to text and sound annotation, and hypertext links to sections of the current document, to other documents, and to arbitrary data structures and programs. The hypertext functionality is modeled on that of *Intermedia* but we do not specify a special 'link marker'. Any typographic enhancement may be used to indicate an attachment. To access a link, the cursor changes from an arrow to a menu shape when it passes over text with attachments, and when the user mouses down a popup menu appears showing the available annotation and links. Attachments are automatically cut and pasted exactly as are the associated typographic enhancements. In terms of implementation, no additional functionality is necessary to keep attachments associated with particular text during editing. The additional data structures are indexed by a field in the style data structures that is otherwise treated as part of the typographic enhancements.

On the right of Figure 2, the *layout component* of the document is split into three sub-components: those associated with the *page structure* of display and printing; those associated with the *paragraph structure* of the document; and those associated with the *pane structure*, basically of embedded pictures in a conventional document.

The page sub-component is again represented by pointer and length structures delimiting strips of text in the text component. The attached data structures define the normal page layout enhancements of the text, such as margins, columns, headers and footers. The page is the basic unit of display and printing. However, since a single document is often reformatted with different page layouts for different purposes, the page is not treated as a logical unit, and there are no attached data structures.

The paragraph sub-component is also represented by pointer and length structures delimiting strips of text in the text component. The attached data structures define the normal paragraph layout enhancements of the text, such as width, indentation, line spacing and a paragraph border. They also support a unique identifier for a paragraph allowing it to be used as a logical unit for *version derivation tracking*. Documents themselves have unique identifiers and maintain a list of previous versions back to the root version of the document. Within documents, paragraphs have unique identifiers and maintain a list of previous versions back to the root version of each paragraph. This enables a document version tree to be reconstructed from a set of available documents and displayed graphically as a user interface for document access. Within a document opened in the context of previous versions, it enables a marker line to be displayed alongside a paragraph for which an alternative version is available. Mousing into this line changes the cursor to a menu symbol, and mousing down then brings up a popup menu showing the alternative versions. This part of the system conforms closely with our previous developments of group editing support systems [21].

The pane sub-component is relatively independent of the text in that it allocates rectangular areas of pages in which other material will be placed. The text is laid out to run around or hop over these areas but is not otherwise enhanced by them. The normal use of these panes is to display graphic material, pictures embedded in the document. The class library of the knowledge document publication system generalizes the usage of these panes by taking advantage of their relative independence of the text to make them available to other applications as if they were a drawing pane in an arbitrary window. This enables them to be used to support visual activity ranging from bit map and line graphics to QuickTime or laserdisc videos, through simulation and animation, to graphic editors for visual languages representing programs, concept maps, and formal knowledge structures. Mouse down clicks within a pane are reported to the associated application rather than to the document software, and hence user interaction can be supported in a completely different environment.

While the document architecture is complex with many data structures and attachments, users see only a single document file exactly as they would with a normal word processor. The internal data structures concerned with the text, typography and layout are stored in the data fork of a Macintosh file, and the attachments are stored as resources in the resource fork. In conventional use the document file is completely self-contained. Some features involve links to other files, such as hypertext links, QuickTime video files, and other application files. These links are stored in resources referencing the associated data using the Macintosh alias manager so that the files can be found automatically even if they have been moved between directories and disks, or are on another computer on the network. The storage of these aliases as standard Macintosh resources enables them to be checked by a utility that can list all the files associated with a document, or can copy them to a disk, if a user wishes to send a document to another user with the assurance that any associated files are being sent also.

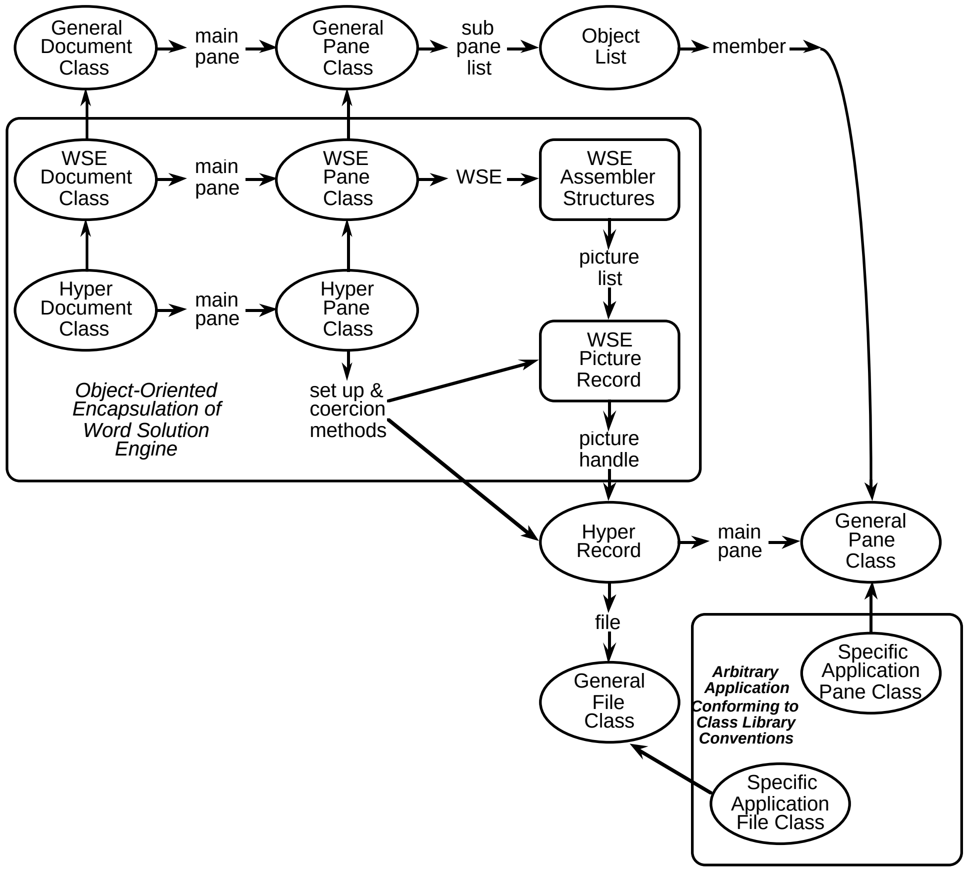


Figure 3 Making document panes available to other applications in the class library

Integration with Other Applications

The class library implementation of KWrite is designed to support panes embedded in documents such that the document processor making up pages and presenting them, and the application interacting through them are maximally decoupled. This has been very effective in enabling us to incorporate previously written applications based on extensions to the THINK C class library with virtually no modification. Figure 3 shows the relations between the class library components that implement this decoupled integration. Arrows directly between components represent class inheritance, and labeled arrows indicate instance variables or, in one case, methods.

At the top of Figure 3, the general *Document* and *Pane* classes are the abstractions through which the standard class library implements ‘documents’ as an aggregation of data files, windows for interaction, and panes for data display within the windows. Immediately below this the *Document* and *Pane* classes are subclassed to provide a wrapper for DataPak’s Word Solution Engine (WSE) which is a commercially available machine code module providing typographic and page layout functionality [3]. These are subclassed again to support our hypermedia attachments and external application integration.

WSE supports pictures embedded in documents through a record structure containing a picture handle. It supports an attached procedure called just before drawing the picture in case the programmer wishes to enhance the picture, for example with a box. When an author embeds a multimedia item in a document, the *HyperPane* class replaces the picture handle in this record structure with a pointer to a *HyperRecord* object that manages a general *File* object referencing the data for the item and a general *Pane* object located where the picture would be placed. The top level *Document* sees this *Pane* as a normal subpane of its main pane. WSE sees it as a picture location in its picture list. The external application sees the appropriate subclass of the general *File* and *Pane* objects as its normal data structures.

This implementation architecture makes it simple to offer access to any application written in THINK C’s class library through ‘pictures’ embedded in a document rather than through standard Macintosh windows. The main change we have had to make to applications is to move their dialog components into floating windows that appear when the user double clicks in a picture pane. This avoids incorporating elements in the document that are inappropriate in the printed form. Examples of such dialogs are given in the following section. It is also possible to integrate applications not written in the THINK class library by using an interapplication communication protocol that passes to them requests to update the embedded pane, and messages about user interaction with it. It may be possible to integrate some applications without modification by patching the Macintosh window manager to support this protocol.

Word Processing and Page Layout Support

As shown in Figure 4, Word Solution Engine supports the full range of features expected in a commercial word processing program, and is fast and efficient in operation, so that users who are familiar with existing programs feel no sense of loss of functionality in using KWrite. Text is entered through the keyboard, or cut and pasted, and the font size, face and style of any part of it selected can be changed through menus and command keys in the normal way.

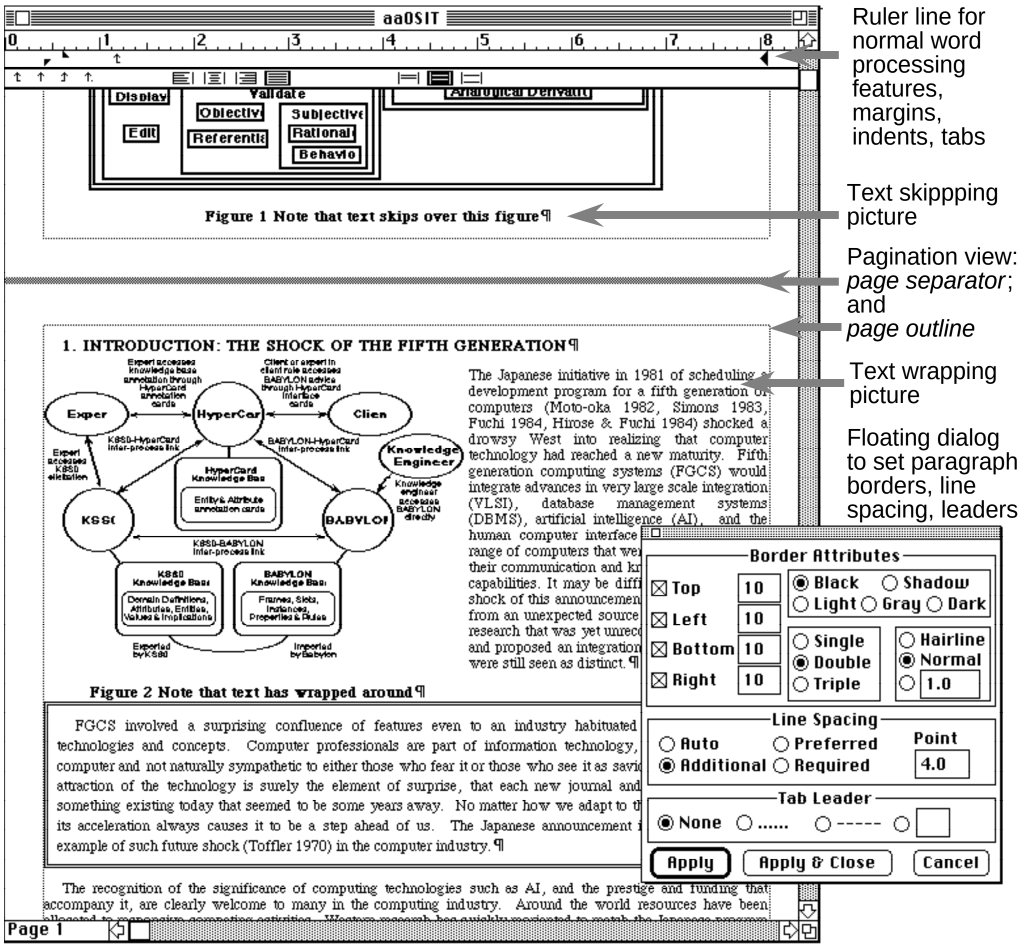

aaOSIT
Display
Validate
Edit
Objectiv
Subjective
Referentia
Rational
Behavio
Analytical Derivati
Exper
HyperCar
Clien
Knowledge Engineer
KSS0
BABYLON
Expert accesses knowledge base annotation through HyperCard annotation cards
Client or expert in client role accesses BABYLON advice through HyperCard interface cards
KSS0-HyperCard inter-process link
BABYLON-HyperCard inter-process link
Expert accesses KSS0 elicitation
Knowledge engineer accesses BABYLON directly
HyperCard Knowledge Bas
Entity & Attribute annotation cards
KSS0-BABYLON inter-process link
KSS0 Knowledge Bas
Domain Definitions, Attributes, Entities, Values & Implications
BABYLON Knowledge Bas
Frames, Slots, Instances, Properties & Rules
Exported by KSS0
Imported by Babylon
Page 1
Ruler line for normal word processing features, margins, indents, tabs
Text skippping picture
Pagination view: *page separator*; and *page outline*
Text wrapping picture
Floating dialog to set paragraph borders, line spacing, leaders

Figure 1 Note that text skips over this figure¶

1. INTRODUCTION: THE SHOCK OF THE FIFTH GENERATION¶

The Japanese initiative in 1981 of scheduling a development program for a fifth generation of computers (Moto-oka 1982, Simons 1983, Fuchi 1984, Hirose & Fuchi 1984) shocked a drowsy West into realizing that computer technology had reached a new maturity. Fifth generation computing systems (FGCS) would integrate advances in very large scale integration (VLSI), database management systems (DBMS), artificial intelligence (AI), and the human computer interface
range of computers that we
their communication and kn
capabilities. It may be diffi
shock of this announcemen
from an unexpected source
research that was yet unrec
and proposed an integration
were still seen as distinct. ¶

Figure 2 Note that text has wrapped around¶

FGCS involved a surprising confluence of features even to an industry habituated
technologies and concepts. Computer professionals are part of information technology,
computer and not naturally sympathetic to either those who fear it or those who see it as savi
attraction of the technology is surely the element of surprise, that each new journal and
something existing today that seemed to be some years away. No matter how we adapt to t
its acceleration always causes it to be a step ahead of us. The Japanese announcement i
example of such future shock (Toffler 1970) in the computer industry. ¶

The recognition of the significance of computing technologies such as AI, and the prestige and funding that accompany it, are clearly welcome to many in the computing industry. Around the world resources have been

Border Attributes

[✓ Top] 10 [✓ Left] 10 [✓ Bottom] 10 [✓ Right] 10

[✓ Black] Shadow Light Gray Dark

Single [✓ Double] Triple

Hairline [✓ Normal] 1.0

Line Spacing

Auto Preferred [✓ Additional] Required Point 4.0

Tab Leader

[✓ None] ...... -----

Apply | Apply & Close | Cancel

Figure 4 Word processing and page layout features

At the top of Figure 4 is a ruler line allowing paragraph formatting to be changed, tab positions to be entered, and so on. Pictures are pasted into the text from the clipboard, or placed as files from the menu. They can be dragged to any position and resized through ‘handles’ that appear when they are selected. Text may be set to run around or hop over pictures, and the margins around pictures between them and the text may be adjusted.

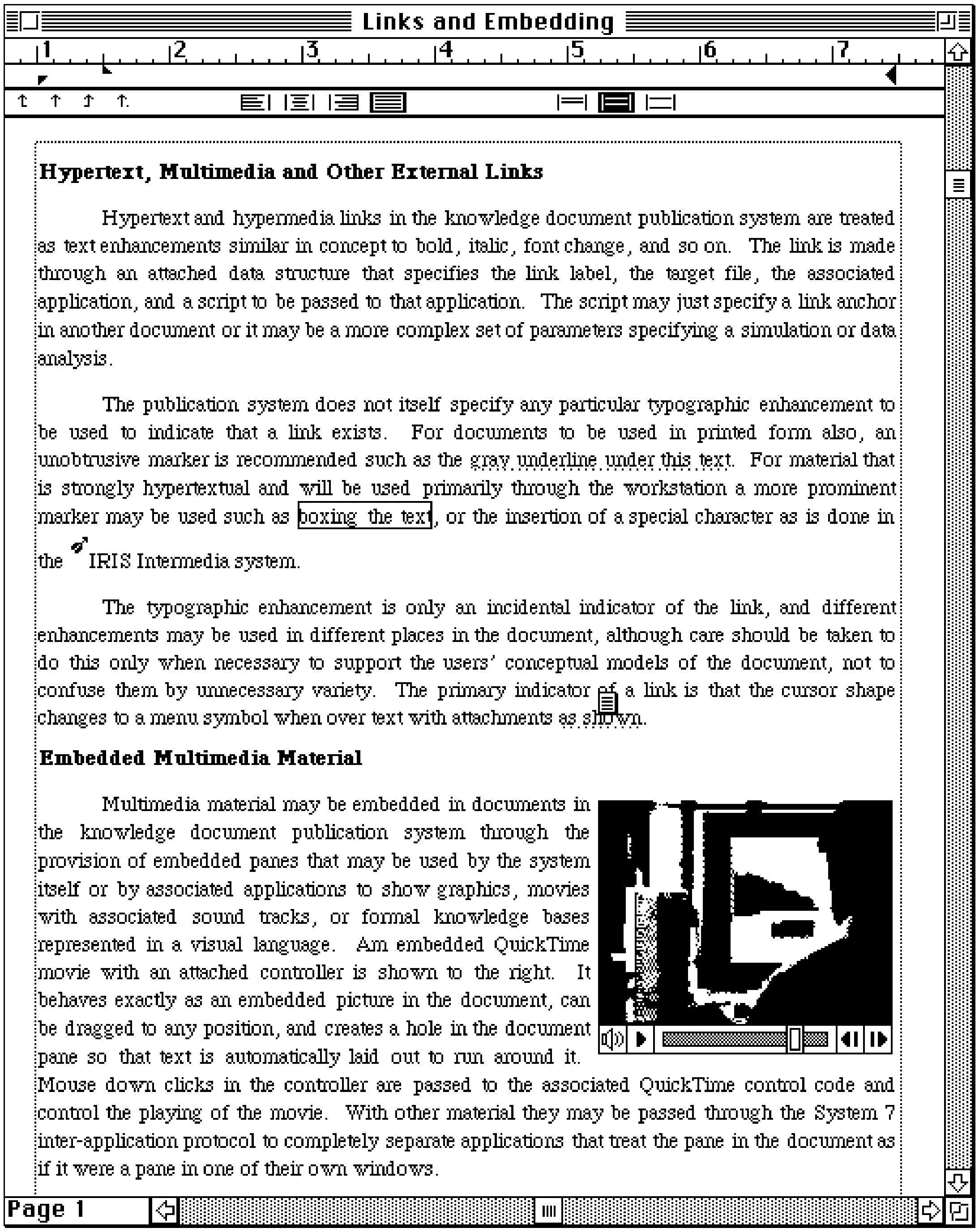
At the lower right in Figure 4 is a floating dialog that the user has brought up through a menu in order to adjust paragraph and picture margins, borders and so on. Such floating dialogs are a major feature of the user interface to KWrite, supporting not only the typography and page layout, but also hypertext linking, visual language knowledge structures editing, and so on. They are movable and non-modal, float in their own layer above document windows, and are hidden if another application is active. They are important in allowing the complex functionality of the publishing system to be made accessible without overwhelming users.

Hypermedia and Multimedia Support

Using the facilities already described, the support of hypermedia linkages to other documents and applications, and the support of embedded multimedia material in the knowledge document publishing system is very straightforward. Figure 5 illustrates both capabilities in use through part of a tutorial document describing them. Any form of typographic enhancement, such as the gray underline, boxing and special symbol shown, may be used to indicate the existence of a link. When the user mouses over such a link the cursor changes to a menu symbol as shown at the center left in Figure 5, and mousing down brings up a popup menu that may be used to access the linked material.

Multimedia items embedded in the document are supported equally simply using the independent sub-panes of the document already described. Figure 5 also shows a QuickTime movie embedded in a document such that the video and sound may be played through the attached controller. When the document is printed only the image shown will appear but this is usually a valuable reminder of the contents of the full movie. Similarly the typographic enhancements associated with hypermedia links are printed as a reminder that there was associated material.

The protocols associated with the linkage structure and the embedded panes are extremely flexible and open architecture allowing for new facilities to be added, such as simulation and data analysis, through relatively independent applications that receive messages from the document including the availability of a pane for their text or graphic output. Currently only applications written in the THINK C class library can take full advantage of this capability, but the support of such open systems integration features by independently developed applications may be expected to become increasingly common in the future. For example, the scripting program Frontier [32] expects independent applications to support an additional menu under its control, and Microsoft’s Object Linking and Embedding protocols (OLE) [16] are already being used by other vendors to integrate their applications with applications such as Word.



Hypertext, Multimedia and Other External Links

Hypertext and hypermedia links in the knowledge document publication system are treated as text enhancements similar in concept to bold, italic, font change, and so on. The link is made through an attached data structure that specifies the link label, the target file, the associated application, and a script to be passed to that application. The script may just specify a link anchor in another document or it may be a more complex set of parameters specifying a simulation or data analysis.

The publication system does not itself specify any particular typographic enhancement to be used to indicate that a link exists. For documents to be used in printed form also, an unobtrusive marker is recommended such as the gray underline under this text. For material that is strongly hypertextual and will be used primarily through the workstation a more prominent marker may be used such as boxing the text, or the insertion of a special character as is done in the IRIS Intermedia system.

The typographic enhancement is only an incidental indicator of the link, and different enhancements may be used in different places in the document, although care should be taken to do this only when necessary to support the users' conceptual models of the document, not to confuse them by unnecessary variety. The primary indicator of a link is that the cursor shape changes to a menu symbol when over text with attachments as shown.

Embedded Multimedia Material

Multimedia material may be embedded in documents in the knowledge document publication system through the provision of embedded panes that may be used by the system itself or by associated applications to show graphics, movies with associated sound tracks, or formal knowledge bases represented in a visual language. Am embedded QuickTime movie with an attached controller is shown to the right. It behaves exactly as an embedded picture in the document, can be dragged to any position, and creates a hole in the document pane so that text is automatically laid out to run around it. Mouse down clicks in the controller are passed to the associated QuickTime control code and control the playing of the movie. With other material they may be passed through the System 7 inter-application protocol to completely separate applications that treat the pane in the document as if it were a pane in one of their own windows.

Figure 5 Hypermedia and embedded multimedia support

Concept Map Support

Concept maps have a long history of use in education and management as a means of representing and investigating knowledge structures [12, 24]. KWrite provides tools for concept map design supporting nodes of different shapes and colors, and linkage from nodes to scriptable document actions, such as playing QuickTime movies. One project using these features is a continuation of some previous media studies undertaken with the KSI by Dr Joan Vickers, Head of the Neuro Motor Control Laboratory in the Faculty of Physical Education at the University of Calgary. Vickers has developed an approach to sports coaching based on knowledge structures for various sports expressed as hierarchical concept maps. Knowledge structures for specific sports have been developed by senior international coaches for those sports and have been used as the basis for an extensive book series on sports coaching [28].

Our original project took place in the mid-1980s as these knowledge structures were being created and involved the production of a number of laserdiscs incorporating video material illustrating athletes of various levels of skill in actual situations performing activities illustrating the knowledge structures. These discs were accessed through programs graphically representing the knowledge structures and managing the lesson sequences, drill and practice in class situations [13, 14].

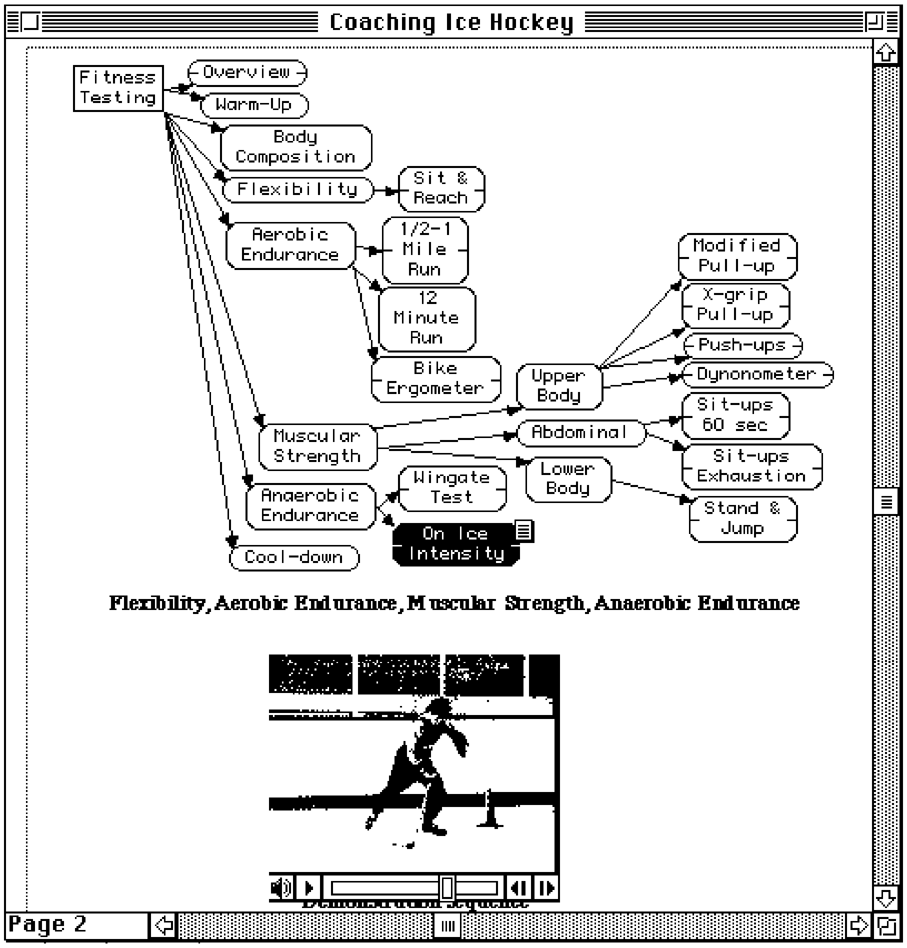


Figure 6 Hypermedia and embedded multimedia support

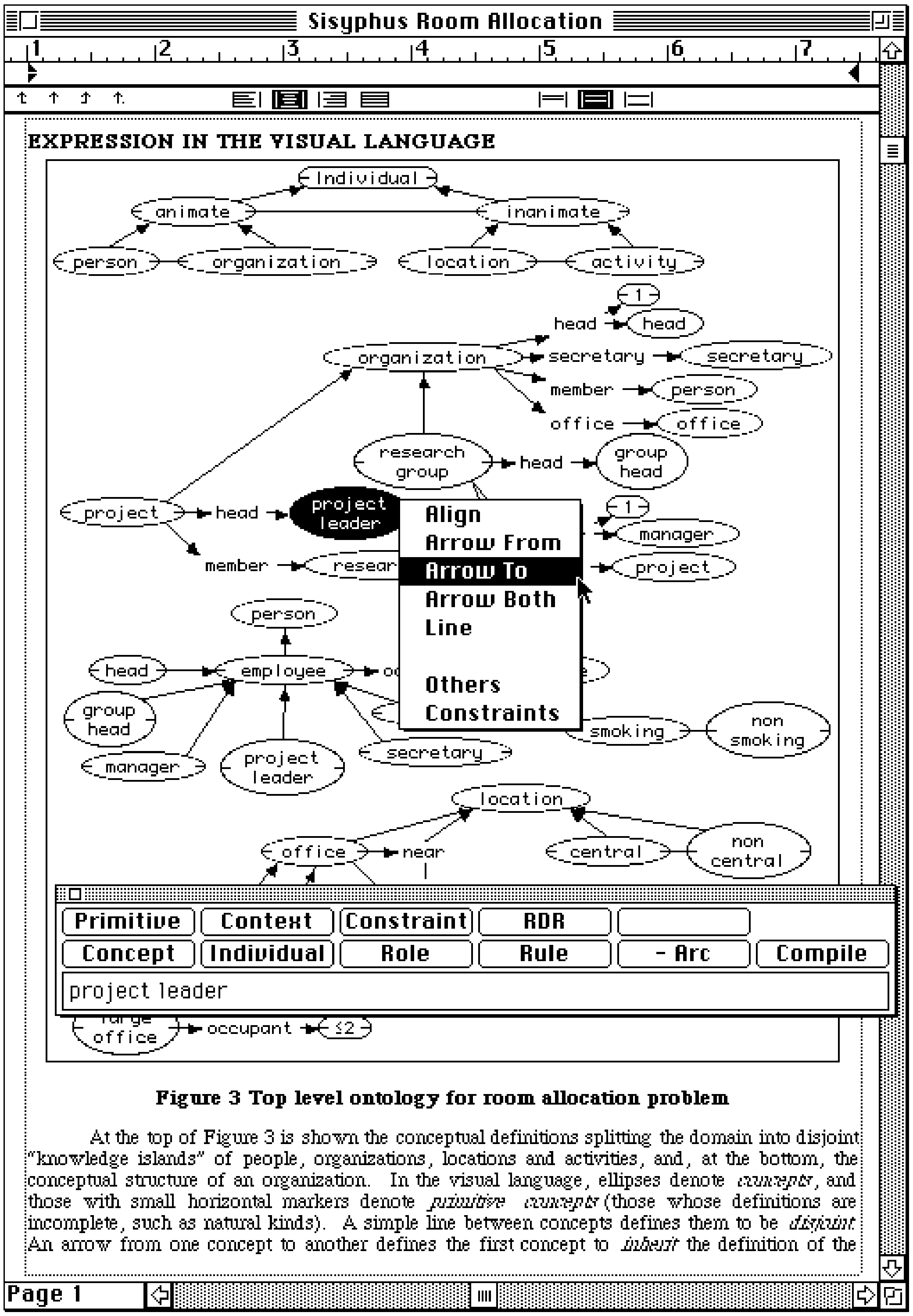
One outcome of these studies was a detailed comparison of laserdiscs and books as media through which to present the coaching material [29], and as the new multimedia document technology became available it was obviously interesting to investigate it as a third option in which the book and laserdisc material could be combined. We have digitized some of the video material and incorporated it together with text and active knowledge structures in documents that emulate sections of the existing books and laserdisc material. Figure 6 shows a screen dump of a page being accessed. At the top is the relevant part of the overall knowledge structure as a concept map. The popup menu accessible at the right of each node activates the relevant section of the QuickTime movie below the knowledge structure.

Many interesting questions are being raised by this project. For example, in a sports coaching context are the quality limitations of current QuickTime replay without special hardware support significant? In particular, do the irregularities of timing defeat the object of showing a movement sequence? In book form sequences are shown as a composite photograph of successive body and limb positions. One can achieve this effect by stepping through the QuickTime sequence, and from this perspective the jerky playback is not necessarily a major fault. However, only empirical evaluation can determine the acceptability of the quality of the currently available technology in this application.

The coaching material is of interest not only because of the availability of comparisons with both paper and laserdisc material, but also because its presentation is driven by the knowledge structures presented as hierarchical concept maps. Such structures are closely related to the conceptual structures for documents proposed as a way of managing multimedia documents [22], and their acquisition is also related to the elicitation of semantic networks as a basis for structuring hypermedia documents [19]. Thus, the form of the coaching material detached from its content is representative of some of the possibilities for new document architectures that continue to support parallel publication in paper and digital form yet have greater structure than conventional books and papers. It will be interesting to see whether interest arises in scholarly publications in which knowledge structures and argument forms are more explicitly formulated in this way.

Knowledge Base Support

Another system that was developed to demonstrate integration with other applications provides access to the knowledge base facilities that comprise the knowledge support system, KSSn [6]. The knowledge support systems research has been part of a program of development of knowledge acquisition methodologies and tools with major emphasis on these being usable by end-users such as knowledgeable experts [11, 27]. This has been achieved primarily through the provision of simple, natural, attractive and easily learnt graphic interfaces to text analysis, repertory grid, inductive modeling, semantic network, and related tools [10]. The independent embedded panes of the document publication system now allow all of these interfaces to be offered as embedded features of a document rather than as separate applications operating through their own windows.



EXPRESSION IN THE VISUAL LANGUAGE

Figure 3 Top level ontology for room allocation problem

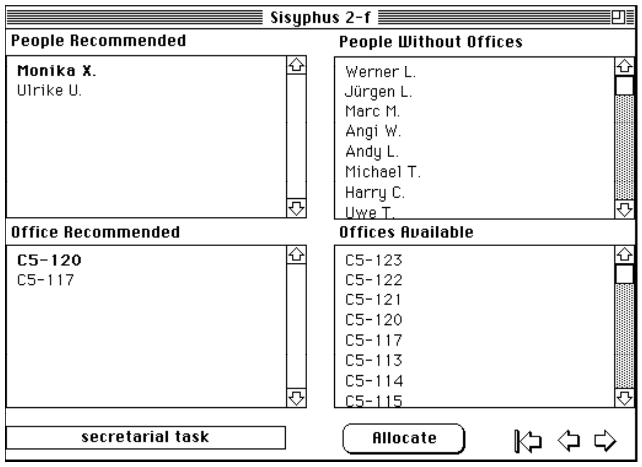
At the top of Figure 3 is shown the conceptual definitions splitting the domain into disjoint "knowledge islands" of people, organizations, locations and activities, and, at the bottom, the conceptual structure of an organization. In the visual language, ellipses denote *concepts*, and those with small horizontal markers denote *primitive concepts* (those whose definitions are incomplete, such as natural kinds). A simple line between concepts defines them to be *disjoint*. An arrow from one concept to another defines the first concept to *inherit* the definition of the

Figure 7 Formal knowledge base in a visual language embedded in a document

Figure 7 shows a semantic network in a formal visual language for KL-ONE-style term subsumption knowledge representation [7] embedded in a document. It normally appears as if it were a line graphic but it is in fact a fully active, editable knowledge structure that can be used for inference as part of a knowledge base. Editing controls are provided through floating dialogs and popup menus similar to those already shown. When the user double clicks in the pane a floating dialog appears as shown towards the bottom of Figure 7. This enables existing visual objects to be edited and new ones to be created. When the user mouses over the right edge of a knowledge object the cursor changes to a menu symbol and mousing down produces a popup menu as shown in the center of Figure 7. This enables visual objects to be linked and aligned, and gives access to other functionality such as object definitions, usage elsewhere and hypertext links from objects if they exist.

The visual language allows knowledge structures with the full functionality of those common in frame-based or object-oriented expert system shells to be entered, and these knowledge structures may be exported to such shells or run deductively in the knowledge representation server, KRS [6], which is accessible from the knowledge document publication system. Equally importantly, the knowledge document itself acts as a knowledge base to KRS and can be interrogated by problem-solving programs external to the publication system. From this perspective, the knowledge document is an annotated, structured, humanly readable and editable knowledge base that should be much easier to maintain than a conventional knowledge base. It is enabling us to explore a possibility we posed some time ago, to design "a MYCIN development within a 'library and information science' ethos in which medical text books and journal articles are seen as the paramount vehicles for knowledge" [5]. We can now develop knowledge bases in which formal knowledge structures 'shadow' informal ones, and processes of annotation and explanation are simply and naturally introduced.

The active knowledge document shown in Figure 7 is one describing a solution to the room allocation problem derived from an ESPRIT project [30]. Figure 8 shows a HyperCard stack being used as a user interface to interrogate the knowledge document of Figure 7 as part of a problem solving sequence in allocating rooms based on conceptual frameworks, rules and facts embedded within the document. The HyperCard stack sends messages to the knowledge document publication system using the Macintosh System 7 inter-application protocol and receives back the formal knowledge base components in a form that can be run deductively in KRS. Editing the knowledge structures in the document can change the concepts, rules and facts, and thus change the solutions developed for the problem.


Sisyphus 2-f

People Recommended: Monika X., Ulrike U.

People Without Offices: Werner L., Jürgen L., Marc M., Angi W., Andy L., Michael T., Harry C., Uwe T.

Office Recommended: C5-120, C5-117

Offices Available: C5-123, C5-122, C5-121, C5-120, C5-117, C5-113, C5-114, C5-115

secretarial task

Figure 8 Interrogating the knowledge base embedded in the document from another application

The full example was published in a book of papers from the British Computer Society Expert Systems Conference in December 1992 with the abstract:

"This paper is written in a document production tool that appears to a user as a word processor but also acts as an expert system shell with frame and rule representations supporting deductive inference. The electronic version of the document is active, providing typographic text and page layout facilities, versioning, hypermedia sound and movies, hypertext links, and knowledge structures represented in a visual language. It can be read as a hypermedia document and also interrogated as a knowledge-based system for problem-solving. The paper version of the document, which you are now reading, is produced by printing the electronic version. It loses its active functionality but continues to act as a record of the knowledge in the document. The overall technology has been developed as an alternative approach to the dissemination of knowledge bases. It also provides a different interface to knowledge-based systems that emulates document interfaces with which many users are already familiar." [9]

The electronic version of the paper was made available through anonymous ftp and as a CD-ROM. Thus we demonstrated parallel publication of the paper version as a camera-ready copy book chapter, and the electronic version, identical in appearance, as a full working demonstration of the problem solution. What is particularly significant is that there were *no* hidden data structures. The semantic networks in the paper were the complete knowledge structures and operated directly in the inference engine to solve the problem. Thus, the document was also the knowledge base.

Conclusions

An interactive multimedia document publication system has been described which integrates a number of different representation technologies to provide a medium offering a wide spectrum of usage, from emulation of current paper publication, through electronic document delivery, multimedia inclusion of video and sound, structured hypermedia linkage, and formal knowledge representation supporting simulation and computational inference. The system is targeted on exploring new forms of scholarly communication, and the knowledge document publication system specifically supports collaborative document development, the authentication of disseminated material, and the citation, annotation and reuse of such material.

A central hypothesis in the research reported in this paper is that the scholarly utilization of new technologies for formal representation of knowledge in operational form will be most rapid and effective if it can be offered as a natural evolution of existing publication media. Hence the publication system provides a rich word processing and page makeup environment with all the facilities normally expected, and adds multimedia, hypermedia and computational facilities incrementally and naturally, with careful attention to the usability of the human-computer interaction involved.

The product from the system is an active document in which knowledge is represented in a variety of ways, some targeted on human interaction, some targeted on computational analysis, simulation and inference, and such that the document can be printed as a conventional paper or book losing the dynamic aspects of the material but retaining the visual representation. The implementation of the system has an open architecture allowing document functionality to be enhanced by links with other applications. The system is robust and easy to use, and compares favorably with existing word processing and page makeup systems. Only time and experience with diverse user communities will show which of the new features are important to the creative and communication processes of scholarship.

In conclusion, all the technologies necessary for the development of new forms of electronic publication are now available, and it is not difficult to provide support for multimedia documents incorporating hypermedia links, formal knowledge structures, and access to a variety of applications. We now have to use our creative imaginations to take advantage of these systems to enhance the creation and dissemination of knowledge.

Acknowledgments

Financial assistance for this work has been made available by the Natural Sciences and Engineering Research Council of Canada. The research reported here would not have been possible without access to the word processing software developed by Gary Crandall of DataPak. We are grateful to Joan Vickers for the use of her research material on knowledge structures in coaching.

References

[1] E.A. Bier and A. Goodisman, "Documents as user interfaces," in *EP90: Proceedings of the International Conference on Electronic Publishing, Document Manipulation & Typography,* R. Furuta, Editor. Cambridge University Press: Cambridge, UK. p. 249-262, 1990.

[2] V. Bush, "As we may think," *Atlantic Monthly*, vol. 176, pp. 101-108, 1945.

[3] G. Crandall, *Word Solution Engine Programmer's Manual*, Vancouver, Washington: DataPak. 1990.

[4] D.L. Drucker and M.D. Murie, *QuickTime Handbook*, Carmel, Indiana: Hayden. 1992.

[5] B.R. Gaines, "Knowledge support systems," *Knowledge-Based Systems*, vol. 3, no. 3 pp. 192-203, 1990.

[6] B.R. Gaines, "Empirical investigations of knowledge representation servers: Design issues and applications experience with KRS," *ACM SIGART Bulletin*, vol. 2, no. 3 pp. 45-56, 1991.

[7] B.R. Gaines, "An interactive visual language for term subsumption visual languages," in *IJCAI'91: Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*. Morgan Kaufmann: San Mateo, California. p. 817-823, 1991.

[8] B.R. Gaines, "An agenda for digital journals: the socio-technical infrastructure of knowledge dissemination," *Journal of Organizational Computing*, vol. 3, no. 2 pp. to appear, 1993.

[9] B.R. Gaines and M.L.G. Shaw, "Documents as expert systems," in *Research and Development in Expert Systems IX. Proceedings of British Computer Society Expert Systems Conference,* M.A. Bramer and R.W. Milne, Editor. Cambridge University Press: Cambridge, UK. p. 331-349, 1992.

[10] B.R. Gaines and M.L.G. Shaw, "Integrated knowledge acquisition architectures," *Journal for Intelligent Information Systems*, vol. 1, no. 1 pp. 9-34, 1992.

[11] B.R. Gaines and M.L.G. Shaw, "Eliciting knowledge and transferring it effectively to a knowledge-based systems," *IEEE Transactions on Knowledge and Data Engineering*, vol. 5, no. 1 pp. 4-14, 1993.

[12] B.R. Gaines and M.L.G. Shaw, "Supporting the creativity cycle through visual languages," in *AAAI Spring Symposium: AI and Creativity*. AAAI: Menlo Park, California. p. 155-162, 1993.

[13] B.R. Gaines and J.N. Vickers, "Design considerations for hypermedia systems," *Microcomputers for Information Management*, vol. 5, no. 1 pp. 1-27, 1988.

[14] B.R. Gaines and J.N. Vickers, "Hypermedia design," in *Proceedings of RIAO'88 Conference on User-Oriented Content-Based Text and Image Handling*. MIT: Cambridge, Massachusetts. p. 14-23, 1988.

[15] L. Gould and W. Finzer, "A study of TRIP: A computer system for animating time-rate-distance problems," in *Computers in Education,* R. Lewis and D. Tagg, Editor. North-Holland: Dordrecht. p. 359-366, 1981.

[16] M. Heller, "Future documents," *Byte*, vol. 16, no. 5 pp. 126-135, 1991.

[17] W. Herzner and E. Hocevar, "CDAM - Compound Document Access and Management. An object-oriented approach," in *Multimedia Systems, Interaction and Applications,* L. Kjelldahl, Editor. Springer: Berlin. p. 17-36, 1992.

[18] B. Hurter and A. Stone, "Kodak Photo CD System," *Photographic*, vol. (September), pp. 15-24, 1992.

[19] D.H. Jonassen, "Semantic net elicitation: tools for structuring hypertext," in *Hypertext: State of the Art,* R. McAleese and C. Green, Editor. Intellect: Oxford, UK. p. 142-152, 1990.

[20] A. Kay and A. Goldberg, "Personal dynamic media," *Computer*, vol. 10, no. 3 pp. 31-41, 1977.

[21] N. Malcolm and B.R. Gaines, "A minimalist approach to the development of a word processor supporting group writing activities," in *COCS'91: Proceedings of Conference on Organizational Computing Systems*. ACM Press: p. 147-152, 1991.

[22] C. Meghini, F. Rabitti, and C. Thanos, "Conceptual modeling of multimedia documents," *IEEE Computer*, vol. 24, no. 10 pp. 23-30, 1991.

[23] N. Meyrowitz, "Intermedia: The architecture and construction of an object-oriented bypermedia system and applications framework," in *OOPSLA'86 Conference Proceedings,* N. Meyrowitz, Editor. ACM: New York. p. 186-201, 1986.

[24] J.D. Novak and D.B. Gowin, *Learning How To Learn*, New York: Cambridge University Press. 1984.

[25] J. Pasquier-Boltuck, E. Grossman, and G. Collaud, "Prototyping an interactive electronic book system using an object-oriented approach," in *ECOOP'88 European Conference on Object-Oriented Programming Proceedings,* S. Gjessing and K. Nygaard, Editor. Springer: Berlin. p. 177-190, 1988.

[26] J. Savoy, "The electronic book Ebook3," *International Journal Man-Machine Studies*, vol. 30, pp. 505-523, 1989.

[27] M.L.G. Shaw and B.R. Gaines, "KITTEN: Knowledge initiation and transfer tools for experts and novices," *International Journal of Man-Machine Studies*, vol. 27, no. 3 pp. 251-280, 1987.

[28] J.N. Vickers, *Instructional Design for Teaching Physical Activities: A Knowledge Structures Approach*, Champaign, Illinois: Human Kinetics. 1990.

[29] J.N. Vickers and B.R. Gaines, "A comparison of books and hypermedia for knowledge-based sports coaching," *Microcomputers for Information Management*, vol. 5, no. 1 pp. 29-44, 1988.

[30] A. Voß, W. Karbach, U. Drouven, D. Lorek, and R. Schuckey, "Operationalization of a synthetic problem," in *ESPRIT Basic Research Project P3178 REFLECT Task I.2.1 Report*. GMD: Bonn, Germany. 1990.

[31] S.A. Weyer, *Searching for Information in a Dynamic Book*, Palo Alto, California: Xerox PARC. 1982.

[32] D. Winer, *UserLand Frontier User Guide*, Palo Alto, California: UserLand Software. 1992.

[33] N. Yankelovich, N. Meyrowitz, and A. van Dam, "Reading and writing the electronic book," *Computer*, vol. 18, no. 10 pp. 15-30, 1985.