1994

Issues and Obstacles with Multimedia Authoring

Fillia Makedon
Dartmouth College

Samuel A. Rebelsky
Dartmouth College

Matthew Cheyney
Dartmouth College

Charles Owen
Dartmouth College

Peter Gloor

Issues and Obstacles with Multimedia Authoring

Fillia Makedon
Samuel A. Rebelsky, Matthew Cheyney
Charles Owen and Peter Gloor*

Computer Science
The Dartmouth Experimental Visualizaton (DEV) Laboratory
The Dartmouth Institute for Advanced Graduate Studies (DAGS) in Parallel Computation
Dartmouth College
6211 Sudikoff Laboratories, Hanover, NH 03755
email: makedon@cs.dartmouth.edu *UBS, Union Bank of Switzerland

1 Introduction

Unlike traditional authoring, multimedia authoring involves making hard choices, forecasting technological evolution and adapting to software and hardware technology changes. It is, perhaps, an unstable field of endeavor for an academic to be in. Yet, it is important that academics are, in fact, part of this process. This paper discusses some of the common threads shared by three dissimilar cases of multimedia authoring which we have experimented with, that of multimedia conference proceedings, multimedia courseware development and multimedia information kiosks. We consider these applications from an academic point of view and review the benefits and pitfalls of academic development while sharing points of hard-learned wisdom. We draw on experiences from some of the projects run at the Dartmouth Experimental Visualization Laboratory (DEVlab), where we have been developing different types of multimedia applications.

A multimedia document (or application) can be anything from an electronic book to an interactive course, from an interactive slide presentation to a multimedia newspaper or an orientation tool, from an interactive auto manual to a clinical record. Multimedia authoring systems come in many forms, depending on the different applications that drive them. Applications range widely and include: office documents, conference proceedings, information kiosks, professional brochures, course materials, virtual reality museum presentations and others yet to be discovered.

1.1 Multimedia Systems and Tools

The greatest challenge in using multimedia authoring tools is that of automating the process of integration and organization. The effective integration of software programs that enable a user to “experience” a concept through several modes and media (such as text, audio, video, simulation, graphics, visualization, animation, slides, pictures, images, etc.), is still a state of art, rather than science. Multimedia authoring has not reached a stage where we can prove that learning, for example, is more effective than with traditional teaching methods.

Just as multimedia authoring systems vary, so do the users of these systems. New applications lead to the emergence of new users who can range from graphic artists to computer science graduate students learning computational biology principles, or from healthcare professionals being trained on a new machine to history students preparing presentations. One desirable feature behind all these applications is interactivity. Interactivity places the user (or learner) in control of the system; the user manipulates the media forms present through different possible modes of interaction. Interactivity may also mean that it is possible for multiple authors and or users to collaborate.

An easy classification of multimedia authoring systems can be based on three primary platforms: (1) the IBM PC compatibles (DOS, Microsoft Windows or OS/2), (2) the Macintosh, and (3) the UNIX workstation (SUN, DEC, SGI). In addition, the Power PC promises to provide an appropriate platform once it is released and gains acceptance. For an educator, the widest audience can be found in the IBM

PC platform today. In 1991, the Multimedia Personal Computer (MPC) consortium defined a minimum specification for an MPC. On the other hand, the Apple Macintosh, with its QuickTime capabilities has offered exceptional ways of integrating time-based media. UNIX workstations have the problem of dissemination to overcome still, since not all end-users own a SUN, or a DEC or an SGI.

Because of local expertise and resources, our projects at Dartmouth have been primarily designed on the Macintosh. In order to make these projects available on other platforms (typically a PC), we redevelop using related tools (e.g., ToolBook on the PC as a replacement for HyperCard on the Macintosh). Keeping in mind that disseminations is one of the main bottlenecks in making a system highly usable, we have also adopted CD-ROM technology. CD-ROM technology provides large random access storage at low cost, which makes possible the “fancy,” typically space consuming, features such as audio, motion video, and interactive animations, which drive the market for such products. Furthermore, CD-ROM devices are relatively cheap, almost absolutely conform to a single standard, and are highly popular, especially in academic circles. The reason for this popularity is portability. A CD represents a mini digital library in one's pocket. A CD also consumes little of a user's limited disk resources, which is quite important in the academic world.

2 Steps in the Production of Multimedia Documents

Before we proceed with describing three cases of multimedia authoring, we take a brief look at the tasks involved in a multimedia production effort. We can divide them as follows:

(a) ***Requirements definition:*** This task involves assessment (cost, time framework, resources available), definition of user profile, prioritization of desirable facilities included in the application and based on the preceding two factors, evaluation of dissemination platform choices and estimation of the evaluation, production and dissemination costs. This task also involves market analysis (rarely done in academic circles -- for example, how easy is it to sell a proceedings in parallel computation in a bookstore that does not have the facilities to demo it?).

(b) ***Software tool search.*** The second task is that of researching a wide range of software that matches with the platforms chosen in (a), and identifying all the specialized tools to integrate. The choice of a particular authoring system (or configuration of authoring systems and translation tools) for a particular application is critical to success. This is a complex and very time-consuming process that can evolve during production. For example, one may compromise and trade off one feature for another due to the changes in the constraints of an application or other technological changes.

(c) ***Content research, media orchestration and design integration phase.*** The material must be researched, organized, assimilated, written, and a script produced which, like a theater play, orchestrates the appearance and activation of various components and media at designated times. Since, in a hypermedia environment, the user may have alternate options of exploration, this must be taken into account. This stage involves a cyclical process of editing, evaluating, editing, expanding, evaluating, etc. The content (domain) and the users of this content are two driving forces that determine the type of user interface, the trade-off between its sophistication and ease of use, simplicity and power. For example, the built-in support facilities for sound and video or for creating interactive animations, are major factors in making the system more usable and yet more complex.

(d) ***Programming, integration, evaluation, and stepwise refinement.*** The programming of the application is done in a "build a little, test a little" mode. This task involves putting a group of programmers together (mostly students), training them, coordinating them and supervising the progress. This process is problematic in most academic environments, due to low cost student programming which takes second place to course work, etc. It is important to also have available a separate focus group of users (faculty / students) who evaluate individual components of the product and provide feedback during production.

(e) ***Product Distribution and update maintenance of the software.*** This task usually falls into the hands of the production manager (usually a faculty) and the publisher.

3 Case 1: Interactive Conference Proceedings

So, the question is, *"Why use/create multimedia proceedings?"* The first reason is that, instead of a volume of scientific papers, a simulation of the actual conference is beneficial in giving additional insight into the complex topics presented in a short scientific presentation at a conference. Preserving the presentation is (a) educational in seeing how the written material is encapsulated in a short talk; (b) educational in teaching how to give a scientific presentation; (c) a cost effective way of attending a conference (or having your students do) without leaving home. In fact, multimedia proceedings can serve as the new way of interviewing candidates for academic positions, or simply getting to know who are the key researchers in a field and how they relate the materials in their paper to their previous results.

Another important reason is interactivity: one can insert facilities for annotation,placing bookmarks, searching for a reference, being able to print a whole paper without copyingpage by page the conference proceedings, being able to synthesize new documents out of two or more papers, or even being able to create slides for a lecture. The possibilities are endless, not to mention the ability to create digital library archives out of such multimedia documents. We have incorporated many of these facilities in our interactive multimedia conference proceedings, and this process has been presented in another talk in this conference.

3.1 Experiences with Proceedings from a Parallel Computation Workshop

We have obtained the materials of our multimedia proceedings from presentations and tutorials during an annual summer institute that takes place at Dartmouth and is called DAGS, (The Dartmouth Institute for Advanced Graduate Studies in Parallel Computation). Every year, DAGS emphasizes a particular aspect of parallel computation and includes invited talks, panel discussions, as well as introductory tutorials for young researchers and presentations of new research.

The DAGS '92 proceedings were on the *Practical Implementation of Parallel Algorithms and Machines* and are published as a CD-ROM by Springer-Verlag. This CD includes both the actual papers in hypertext format and the talks in multimedia (with video and sound as well) format. The CD also provides many other features, including search mechanisms, provisions for users to annotate individual slides and sections of the paper, user-defined “paths” through the proceedings, and bookmarks that allow one to quickly return to a particular part of the proceedings. These first two features make it very easy for teachers to provide guides to the material.

What is fundamentally different about these proceedings, as opposed to, say, the CD-ROM proceedings produced for the ACM'93 multimedia conference, is the fact that we produced them after the conference was over, rather than hand them to the participants at the time. This had the advantage of the time element: we were able to re-process materials and involve the authors more actively. In fact, for DAGS'93 we plan to also include the participation of some of the organizers of the symposium as well as profiles of some of the invited speakers and the participants of the "school" that followed the symposium.

The theme of DAGS'93 was"Parallel I/O and Databases", a very important topic in the construction of efficient parallel machines. The 1993 proceedings will be quite different from DAGS'92; it will not only be for two platforms, the Mac and the PC/Windows, but also have more features that enable someone to feel as if she were at the conference itself. For example, while the slides in the DAGS '92 proceedings were mostly static, the DAGS '93 slides will include animated pointers so as to better replicate the talk. In addition to scientific talks presenting new results, the DAGS '93 proceedings will also contain tutorials on parallel programming, a separate school activity that followed the symposium. We hope that this will expand the role of the DAGS'93 proceedings as a teaching tool as well. An abstract screen dump is shown in Figure 1.

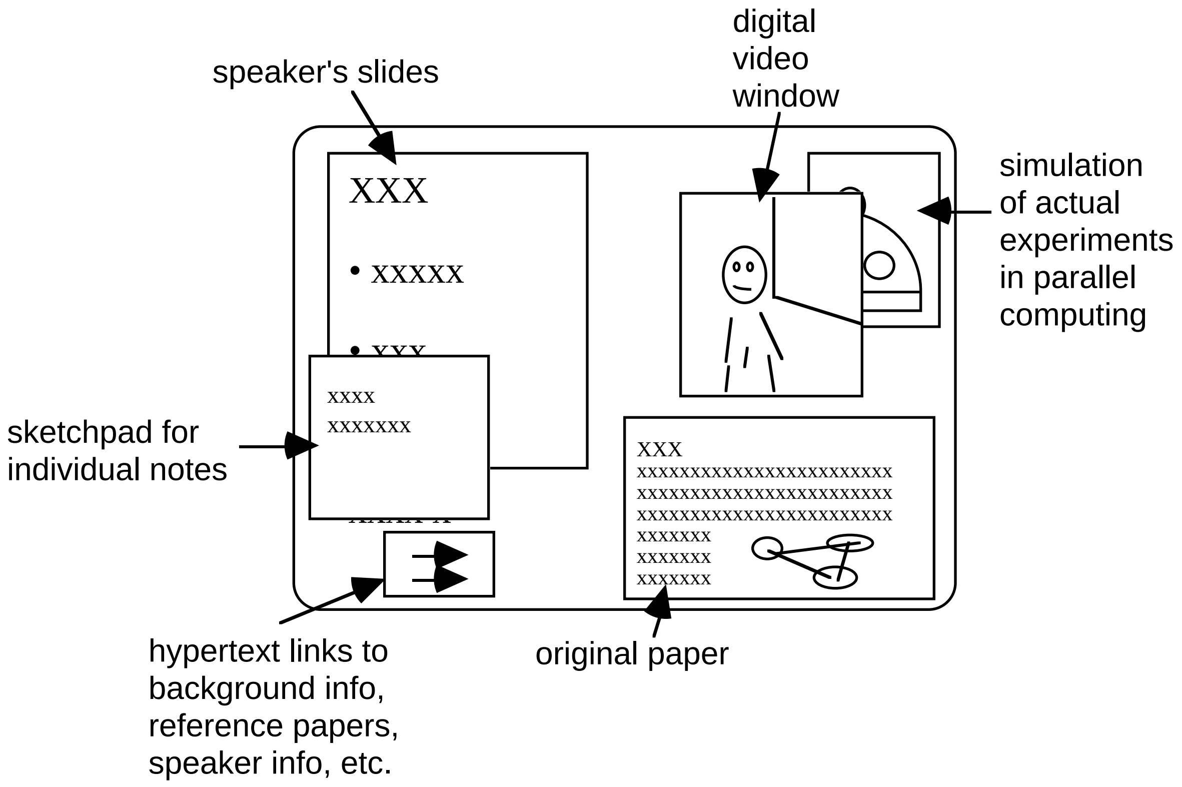
digital
video
window
speaker's slides
XXX
• xxxxx
• xxx
simulation of actual experiments in parallel computing
xxxx
xxxxxxx
sketchpad for individual notes
XXX
xxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxx
xxxxxxx
xxxxxxx
hypertext links to background info, reference papers, speaker info, etc.
original paper

Figure 1: Example of what a screen from our multimedia proceedings

3.2 New Roles of Editors, Authors and Publishers in Multimedia Publishing

We see five primary players interacting in the publishing world: authors, editors, publishers, marketers, consumers. We will concentrate on the first three. In the world of multimedia publishing, these roles have obtained new dynamics and the relationships are continuously changing. Traditionally, the biggest task has been upon the author. A new complication with multimedia publishing is that there is not just one author but many authors, including multiple secondary-authors, i.e., content-knowledgeable programmers who implement creatively the script of the primary author (or even design the script based on the materials in a traditional text). Furthermore, we do not just have the editing of text but the need to orchestrate a whole set of media other than text, based on content. In addition to the multiple media and authors, there are options for the user to read/interact/experience the system.

In the traditional world of publishing, there isn't usually a person to manage all these processes since the publisher's role has now become that of an involved "reviewer" as well as market assessor, who comments on what needs to be changed. One cannot simply "transfer" the traditional publishing roles to the multimedia publishing arena. Here, product coordination needs a so-called production manager whose task is to coordinate the author(s), the system, the programmers, and the publisher.

In terms of editors, while in traditional publishing authors provide text and editors provide layout, formatting, and other constraints, while in multimedia publishing, editors may undertake a greater responsibility in the production, depending on technical expertise. An editor must, for example, have an understanding of what it takes to present/orchestrate the multimedia data in a correct format. In fact, both editors and authors are of a new technical breed whose communication includes content and implementation.

The role of the multimedia author is now split, depending on the level of involvement; there may be a script-type of author, working with programmer-authors or programmer-editors. Authors now face new submission requirements, even if their task is to provide a script to a programmer. They must understand the system well in order to ensure successful and timely production of a given application and they may be required, for example, to provide fully formatted text to their publishers. In producing next year's proceedings, DAGS '94 (theme: Parallel Programming Environments and Problem Solving Environments), we will give each author tools to develop their talks in such a way that we can easily add it to the proceedings without extensive work on our part. For DAGS '94, we hope to convince the authors to:

- "chunk" their text, adding links and making clear the relation between their slides and the corresponding full-text paper, and between the paper and the references
- make the relation between the material in the paper and the material in the talk explicit;
- build or describe their own links to other multimedia documents in the proceedings.

In producing conference proceedings, one needs to overcome the diversity of software tools available and the resistance on the part of authors to use certain tools to create graphs or slides for their presentation. To automate the process, it is important to be always aware of new tools that emerge which allow the translation of software into each other and the integration and interaction of software from different platforms. These tools are essential for making a system portable, extensible, and disseminable.

4 Case 2: Authoring Multimedia Courseware

Unlike conference proceedings which present front-line research issues (such as, in our case, topics in parallel computing), multimedia courseware presents a different challenge: how to compact knowledge of a possibly complex nature, and present it in an aesthetic, simple, interactive way to novices that motivates learning. We will now discuss our experiences with authoring multimedia modules for a course to teach parallel computing to freshmen and sophomores, a project funded by the National Science Foundation and the New England Consortium for Undergraduate Science Education.

In this project, one of the questions has been how to effectively harness visualization and multimedia technology in order to achieve fast and pleasant learning. Our hypothesis has been that the content (syllabus) drives the design and production of such a course. Furthermore, we believe that an important criterion in the process of integration of the various materials and media is asking the question, "which is the best way to present this concept," as opposed to, "I have this video clip of x,y and this animation of x,z: how and where do I fit them into my application?" In other words, we advocate a minimalist approach which, while it aims at presentation richness, stays focused on the content to be covered and avoids "information clutter." Our aim is to make this course available widely, to computer science faculty and students nation-wide who may have diverse equipment resources.

4.1 Teaching Parallel Computing to Novices

The course is being designed to allow the student to develop fundamentally new ways of thinking, new ways of organizing and viewing real-time data and, most importantly, new ways to learn and practice computing for parallel solutions and in parallel instructional environments. Given that parallelism is harder to comprehend and teach (concurrency and asynchrony are two of the main difficulties in visualizing parallel processes), it is quite a challenge to present such materials to novices. Nonetheless, we believe that parallelism should be at the heart of the computer science curriculum and that it is the future of computing and computer science. The successful introduction to thinking in terms of parallel execution early will result in our eventually rebuilding our curriculum and providing a national model for other schools to do the same.

This course will be novel in two ways. The first novelty is the introduction of this material before algorithms and data structures have been completely introduced, as we currently conceive them in a sequential setting. Unlike courses which "add-on" two or three parallel laboratory components to an existing data structures course, we will concentrate on developing a course which is cohesive and guarantees unity. To our knowledge, this has not been done before. The second novelty of our course is the use of visualization with minimal "informational clutter", hypermedia-induced disorientation, or exposure to unnecessary details concerning parallel machine implementations and parallel languages. We believe that a complex web of information can overwhelm the novice student and it is best to keep each interactive module separate and centered on a particular topic and example. This allows the students to focus on the material at hand, and not lose themselves in an overwhelming array of information. It also allows us to gradually add new concepts as the course progresses (so that students don't accidentally skip to the last topic and become so confused that they give up).

Algorithm visualization plays an important role in the authoring of multimedia courseware for computer science. One problem is, however, that algorithm visualization is quite different from traditional, scientific visualization. While most fields are grounded in real objects, Computer science algorithms involve abstractions which lack concrete representation; learning how the visual objects correspond to the abstract objects of an algorithm represents an additional concept for the novice student.

Course production planning involves, therefore, a choice of what to include and how to present it effectively, in a way that motivates a parallel solution. Secondly, it involves a careful review and prototype development of a shell and a graphical user interface, which should be consistent throughout the course, easy to use, cost effective, easy to port to other environments, easy to extend, and easy to maintain. Since we have divorced the programming skills that the author-user needs to have from the skills that the end-user needs to have to experience the course, neither student nor teacher need to know the authoring programing language to use the course.

As a primarily laboratory-based course, it will involve the students in parallel programming using interactive visualization tools and integrated multimedia facilities. Multimedia tools will be used for cooperative work, student-teacher communication, homework production, course management, solution dissemination and faculty presentation. The course will be composed of eleven multimedia modules to represent ten weeks of a laboratory-based course. The first module provides a self-contained interactive guide for the user to practice with and to “preview” the remaining modules.

We have decided to de-emphasize the mechanical aspects of parallel computation, (e.g., programming implementation details, interconnection protocols, parallel performance issues such as load balancing), and put more emphasis on the intellectual (algorithmic) impact of parallel computation. Unlike courses which present topics in the form of a series of “smart” projects, we aim to produce a novel and integrated intellectual view of data structures. We achieve this by concentrating on the concepts of parallel computing, rather than on a series of parallel languages and systems that, like a black box, produce an answer fast but provide no understanding of the underlying principles. We visualize the data to show how processors are applied to data, thus focusing on the algorithms.

5 Kiosks—Interactive Multimedia Brochures

One of our initial forays into the world of multimedia authoring has been the development of an interactive guide (kiosk) to computer science at Dartmouth. This kiosk includes descriptions of the computer science courses offered at Dartmouth (text); short statements by the faculty members that describe their research interests and reasons for being at Dartmouth (audio and video); a map of Sudikoff Laboratory, our new research center (graphics); and a variety of other materials in many formats. The kiosk includes links between the various sections so that one may, for example, jump from a course to the professor teaching the course, to the map of the building with the professor's office highlighted.

The kiosk project required us, more than the other two projects, to examine the management aspects of the multimedia authoring experience. We had to decide which materials to include; hire undergraduates to work on the project; build a timetable; obtain or construct the need materials; convert them to digital format; decide upon an overall interface to provide so that novices can easily navigate through the various types of information; coordinate material collection (e.g., videotaping of the faculty) and put it all together. Note that the kiosk is not a static application: because the department does not remain stable (courses are added and removed, teaching assignments change, faculty members arrive and leave), the kiosk needs to be updated regularly. This feature is less obvious in the two previous applications described above.

Because of our success with the Computer Science kiosk, we have since been involved in the development of other kiosks, ranging from a guide to computing services at Dartmouth to an orientation tool for new employees at Dartmouth, to a counselor-kiosk for premedical students. We recently began to develop WisKit, an interactive tool to attract women and minorities to the field of Computer Science.

With the help of six first year students funded by Dartmouth's Women in Science Program (WISP) and the New England Consortium for Undergraduate Science Education (NECUSE), we are experimenting with the development of this very different kiosk: (a) the students have had no previous expertise in multimedia authoring; (b) while the information included in the other Kiosks could be described and obtained in advance of the authoring, the WisKit requires much more gradual development; we learn more about the material as we progress (in part, because the young women developing the WisKit discover and suggest it), and therefore find new things to incorporate. This project suggests a "build a little, test a little" development paradigm in multimedia publishing. A group of three senior upper classmen are assisting with the training of the first year students. and we are investigating a network interface to other kiosks within Dartmouth.

6 Commonalities and Lessons Learned

While proceedings, brochures, and teaching materials fall into quite different domains, we found that the preparation of interactive multimedia documents for each of these domains shared many problems and solutions. We employed a similar process going from basic idea to final product in each situation. All have a short expected delivery time and therefore require authoring tools that allow us to meet that schedule.

Proceedings are expected to disseminate current information and should be made available as soon after the conference is held as is possible. Delays can occur from inappropriate materials (e.g., due to faulty videotaping; back-up videotaping person is recommended), digitizing, editing, scanning, OCR, and formatting. Delays can also occur due to the features one chooses to include (for example, we felt that we should include an extensive network of links and, while, creating links does not take an inordinate amount of time, determining links does).

Another delay may occur due to unexpected editing needs. For example, we expected to be able to use the audio and scanned slides with few changes. The audio required significant editing (removing noise and interjections, amplification, and more). Although the slides were readable at full size (approximately 640x480 pixels), when reduced to a size compatible with CD bandwidth requirements (280x312 pixels), they were virtually unreadable and, in most cases, had to be partially or completely retyped. While we have tried to get around some of these problems in the DAGS '93 proceedings by requiring authors to submit materials in electronic form, we have found that many of these problems still recur even when all the files are available in electronic form because it is difficult (and often impossible) to convert from one format to another. Thus, the solution to automating the production still remains unsolved.

One major area of difficulty is the lack of standardized cross-platform tools. While we appreciate the differences between the Unix, PC, and Macintosh environments, it should be possible to develop presentations that are available in all three, or at least the two mass market personal computer environments, without a complex and manually assisted translation process. Though presentation tools exist that provide this basic capability, such a product does not currently exist in the authoring world.

Another issue of concern is the lack of sophisticated text manipulation capabilities in most multimedia software systems. Yes, motion video and sound are impressive, and powerful graphics help represent information more compactly, but both the course project and the conference proceedings require delivery of a large amount of text. The proceedings add powerful presentations to the basic backbone of the presented papers. However, we have found basic text capabilities such as the presentation of highly formatted documents, search engines, automatic keyword indexing, and cross references between documents to be sorely lacking. Hence, we have to build such tools either entirely manually or using scripting languages. This is surprising since so many multimedia presentations contain large amounts of text (encyclopedias, book and document collections).

It has been hard to evaluate the results in the different applications described. One difficulty is due to the fact that the criteria used are not standard and change with the application, the domain, and the type of user who is doing the evaluation. "Learning effectiveness" is a very hard thing to assess and it remains an open questionas to how multimedia-based teaching compares to traditional methods in this respect. The constraints of the university environment had a major impact on the way we developed these documents and our success in doing so. Since a great deal depends on available funding for the student programmers and since these projects are very time consuming, it is extremely hard for a single faculty to develop an application as described above without a large amount of faculty release time and/or student assistants. One possibility is that the university will view this development as infrastructure, rather than as writing a book, and provide facilities for linking to other such resources on campus electronically. This is already happening, to an extent, at Dartmouth.

Among the lessons we have learned is that many of our problems could have been avoided if we had begun with a better idea of what the end product should be like and what form the sources should take to ease development. Furthermore, understanding better our limitations on time and programmer resources, would have helped us make more conservative decisions. Finally, to meet deadline and quality measures, as in all projects, it is essential to keep tight control on the project and to meet with the project team regularly (preferably weekly).

Bibliography

1. *Parallel Computation: Practical Implementation of Algorithms and Machines.*, editors: J. Matthews, F. Makedon, P. Gloor, P. Metaxas and D. Johnson. Telos/Springer-Verlag, Santa Clara, CA, August 1993.
2. "VideoScheme: A Research, Authoring and Teaching Tool for Multimedia", J. Matthews, F. Makedon and P. Gloor, Proceedings of EdMedia 1994.
3. "Conference on a Disk: An Experiment in Hypermedia Publishing", Fillia Makedon, P. Metaxas, J. Matthews, P. Gloor, D. B. Johnson, Matthew Cheyney, Proc. of EdMedia '94.
4. "VideoScheme: A Programmable Video Editing System for Automation and Media Recognition", F. Makedon, J. Matthews and P. Gloor, 1993 Proc. of the ACM Multimedia Conf.,Anaheim, CA 8/1-8/6/93.
5. "Building a Digital Library with Extensible Indexing and Interface: A Discussion of Research in Progress", D.B. Johnson and F. Makedon, Digital Libraries Workshop, Rutgers Univ., 5/94.
6. "A Digital Library Proposal for Support of Multimedia in a Campus Educational Environment", Fillia Makedon and Charles Owen, EdMedia '94 Poster Presentation.