Cathedral, Tool or Framework?
MediaWeaver As a Distributed Scholarly Workspace

Sha Xin Wei
Georgia Institute of Technology
January 2001

Introduction

The MediaWeaver system was developed at Stanford University 1993-1994 to help scholars assemble and compose collections of structured texts, photographs, videos, sounds and executable code, and so forth on heterogeneous computer networks. In that sense the MediaWeaver anticipated the World Wide Web, but it supported ways of composing media that were significantly more ample than what can be supported by the technologies that have been built on the models that underlay the HTML representation encoding and the http transport protocol. Since the MediaWeaver was explicitly designed to support scholarly work in the humanities and arts, it may be fruitful to consider its application to work in the areas of history and philosophy of science and technology. This paper describes the activities that the MediaWeaver was designed to support, the technical problems it was designed to overcome, and the architecture of the system. It describes some applications and ends with observations on the design and use of such technologies by historians and philosophers of science and technology.

Motivation

A major challenge facing designers of networked computing environments today is to fashion scholarly workspaces which are simultaneously coherent, easily reconfigurable, expressive -- small gestures go a long way, and above all, useful in everyday scholarly production. In this paper, I describe MediaWeaver, a system for composing arbitrary renderable media, applications, or mediastreams in diverse models and narrative structures. MediaWeaver was designed to support the construction of models of human systems that are both conceptually rich and data rich. It also mediates between coherent, customizable interfaces and an open set of network services, such as database engines, WWW servers, fulltext search engines, and media conversion facilities.

Our context, that of humanities computing [Thaller] significantly stretches the envelope of what is conventionally supported by networking technology, multimedia, intelligent search systems, and human-computer interface design.

Software technology is designed according to usage paradigms that run the gamut from verb-object tools (e.g. spell-checkers), templates and forms, to object-oriented frameworks that claim adherence to the principles of pattern languages as developed by Christopher Alexander [Alexander] and to peripheral and everyday computing. These technologies can be more or less coupled with specific content: they could be built around the specific texts and images that are intimately bound into a particular corpus such as the Wittgenstein electronic edition [Huitfeldt]. Alternatively, they could provide modestly scoped tools that can be used across multiple domains and authors. The polar results of these approaches can be labeled as cathedrals in the first limit and tools in the second.

Our method was to have designers/programmers share knowledge and practices intimately with the faculty and student researcher/authors who use the evolving systems. [Ehn] Indeed, MediaWeaver was conceived in the beginning as a framework to accelerate our own multimedia designers' work in creating rich complexes of media supported by relational data models. It was natural to extend the notion of designer to include authors who were experts in fields outside computer engineering, namely historians of art, historians of technology, anthropologists etc.

Figure 1. Renaissance theater history project.

SHA Xin Wei, Cathedral, Tool or Framework? MediaWeaver As a Distributed Scholarly Workspace, in Augmenting Comprehension: Digital Tools and the History of Ideas, ed. Dino Buzzetti, Giuliano Pancaldi, and Harold Short, (London: Office for Humanities Communcation, Kings College London), 2004, pp. 101-112.

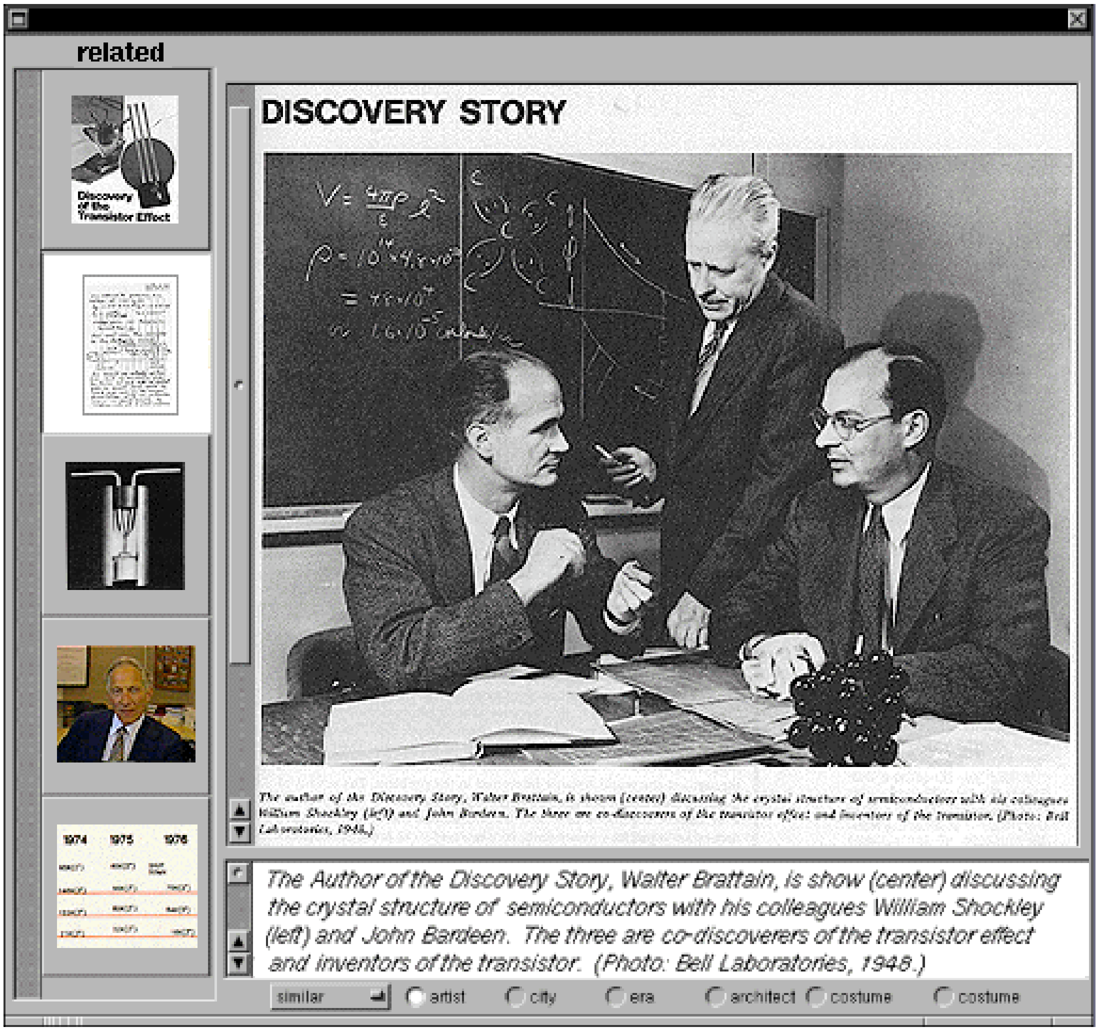

Figure 2. Siliconbase History of High Technology Project.

Figure 3. Video sprite as a proxy in experimental interface for an agent ontology.

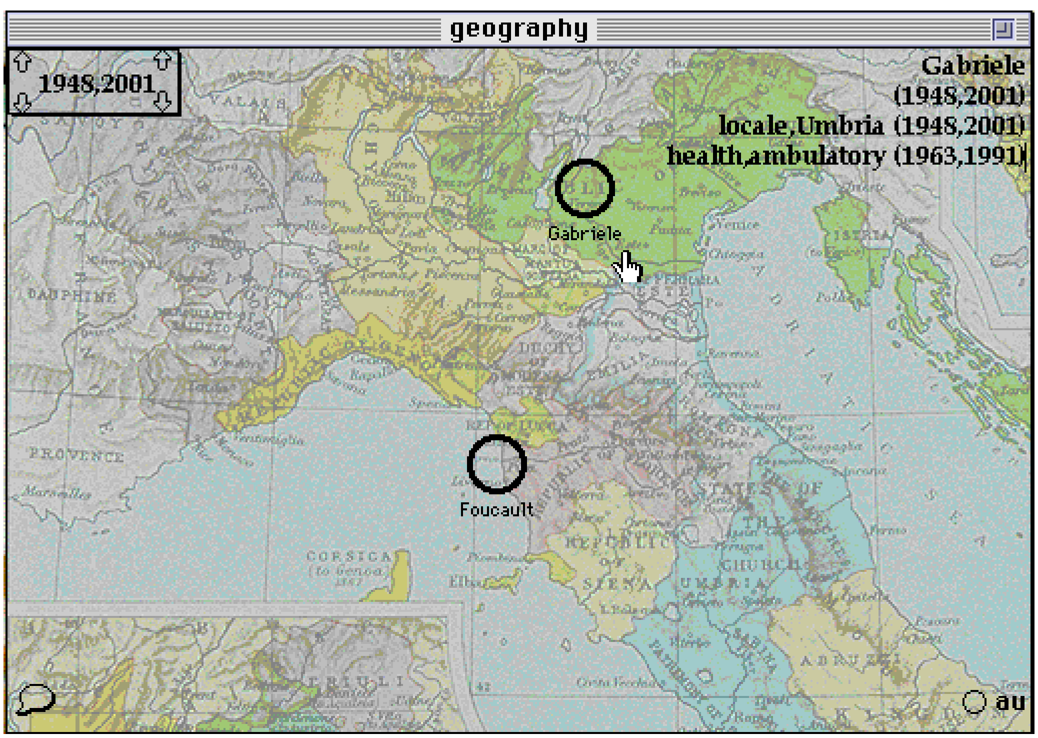

Figure 4. Geographic map as experimental interface for an agent ontology.

Genealogy

After about five years of making interactive multimedia titles, it was useful to take stock of our experimental software development group's work process to see where the bottlenecks were, and also what were the greatest limitations in the interactive titles that we produced. We discovered that:

- Media were scattered all over the network. It was becoming hard to keep inventory using ad hoc databases.
- Researchers significantly changed their conceptual models over the course of a project, so that custom data structures had to be re-written.
- User interfaces had to be constantly re-designed in concert with graphics artists, programmers and researchers, using unpredictably varied media. New interface constructs such as help sprites and custom gestures that did not fit pre-fabricated window-menu-button widgets had to be invented and implemented over and over again.
- Finished titles were often locked into a videodisc or piece of software (e.g. Supercard stack), and put out of reach of re-purposors.
- Finished titles had thin media content/ hard content boundaries -- users quickly hit the boundaries of what was recorded on a CD ROM or videodisc.
- Conceptual models were often too simplistic to be taken seriously by any but the most novice students. We wanted environments that could support research level work as well as introductory classes. (In general, software which was designed specifically for a given class or lesson was often too rigid and shallow.)
- Hypertext/media graph topologies were either navigable but too sparse to sustain a viewer's interest, or rich but too dense to be comprehended. Traditionally, hypertext links are fragile, difficult to author or manage, and hard to map.
- We could not easily support multi-author and multi-player discourse networks.
- We knew in advance that any media encoding schema and of any media technology would likely be much more short-lived than the scholarly interest in the media. Technological obsolescence had to be anticipated.

The MediaWeaver was designed to address all of these problems. Its various frameworks were designed to be used by faculty and student authors and by designers of multimedia simulations; it was designed explicitly to support members of academic disciplines outside traditional programming communities. And it had to leverage tiny application programming resources.

We started with two prototype projects in 1993-1994: a history of Renaissance (Elizabethan) theater, and a study of high technology in the Silicon Valley. The first was chosen from a pool of faculty projects, which required some management of art images and associate music or text on the network. The second presented the challenge of dealing with a significant, changing body of structured text in a complex, evolving research model. In addition, we wanted to lay the foundation for general relational modeling of human systems as such data became available in the course of the research. In both cases, we could not assume a fixed interface or conceptual model. Indeed, the only surety was change. Over the course of the next 4 years, we built more than 13 multiply-authored, multi-purpose, research and pedagogical archives ranging from a showcase of work by and about contemporary Chicana artists to a clearinghouse of information for indigenous forest-based communities.

This genealogy strongly influenced the design principles that we outline in the following section.

Design Principles and Corollaries

Make it immediately useful.

Practical necessities and principles of participatory design suggested that we let composers start working right away with their own media and compose commentaries, conduct seminars, write papers using our system, rather than wait for custom, turnkey editors and mediabases. To enable significant scholarly work, whatever we built had to exchange data transparently with commercial applications and databases, and inter-operate transparently with distributed services.

Authors were encouraged to use whatever commercial editors they already had on their personal computers (Macintosh, some Windows computers): e.g. Microsoft Word, WordPerfect, Adobe Photoshop, Adobe Premiere, Omnipage, DeskScan. MediaWeaver's framework incorporated commercial, public and custom software. Our authors worked in a heterogeneous network where UNIX and Macintosh clients shared a common network-based filesystem. This allowed authors to apply productivity tools under Macintosh, UNIX/X and UNIX/NeXTStep operating systems to a single given file. This was extremely useful, for example, when acquiring images on a personal computer, performing image touchup and sampling in batch mode on a network of UNIX workstations, and then composing compound documents on a diverse and dispersed set of personal computers.

Factor, factor, factor.

The architecture reflected a separation between (1) persistent storage in the filesystem (e.g. ASCII or AIFF blob bytes) and in databases (eg. blob metadata in Sybase tables); (2) model (e.g. hypermedia topological structure, bibliography); and (3) presentation/interaction (eg. WWW/Mosaic document, Hypercard simulation, custom disposable applications). By decoupling models from media, we sidestepped the question of data ownership and allowed complex research models to be constructed on existing corpora or proxy media.[end note 1]

Since MediaWeaver stores topological information in databases, it can generate HTML documents on the fly rather than keep source media in HTML files -- a simple version of dynamic documents [Weitzman]. This factorization gave us the option of interposing even more expressive and nuanced means of forming constellations media or mediastreams on the fly.

Maintain user interface metaphor neutrality.

We allowed multiple views on shared media, which means that rather than building a single interface application or layout protocol (a la HTML forms), we provided an application programmer interface (API) supporting multiple, concurrent, and most importantly, reconfigurable interfaces. The MediaWeaver did not assume that views must look like word-processors. Word-processor-like document viewers like Microsoft Word or Mosaic present essentially a unidimensional rebus, a stream of generalized characters, some of which are ordinary letters, some of which are raisins of media like an embedded graphic. In general, a simulation can have quite a different structure, such as a map, timeline, multi-track score, vivarium, video-based virtual and mixed reality, soundspace etc. MediaWeaver user interface kits did not assume fixed interface format conventions such as documents, windows, chunks, or links. But the MediaWeaver did deliver documents as a special case. In this way ordinary word-processor documents could be composed and catalogued in indigenous formats rather than some canonical universal format such as SGML.

Broadcast rather than publish.

MediaWeaver was designed to deliver information over networks, rather than in detached forms such as CD ROM. The distribution model for CD ROMs and videodiscs was in a sense a natural relic of the traditional publishing model that required a physical commodity in order to function. From the point of view of a university library, most if not all of the same problems encountered in acquiring preserving, cataloguing and circulating paper books or journals recur in dealing with CD ROMs and videodiscs. Some of these library issues are even thornier with newer formats like streaming video. Fine-grained network distribution of software, even of single computing objects, offers quite a different paradigm which may be more akin to a broadcast model than to the publishing model. A broadcast model provides us the flexibility we need to support live research projects in which the primary source media as well as the secondary literature and even the conceptual models are in flux. In any case, MediaWeaver's factorization allowed us to build templates by which we could download a subset of a project's model and data content as desired. In this way, we could print a standalone version of simulations analogous to T. Gieryn's Cornell Biotechnology Lab or G. Crane's Perseus simply by downloading data and models from the network into local storage.

Even more interesting are the new models of editorial work now made possible by online mediabases. MediaWeaver provided a scheme in which progressively more formal or public compositions could arise organically from flexible, personal or project-specific research collections. For example, collections of source material could be acquired and edited according to research agenda. This demand-driven model efficiently allocated human and system attention. New scholarly articles or pedagogical presentations could be made in situ and catalogued back into the mediabase. For example, the SiliconBase seminar's reader was an entirely online hypermedia structure, which was modifiable at any moment by the instructors. Lectures could be composed, presented in conferences, and revised online. Over time, as articles were reviewed and rewritten, they were given more public status by progressively relaxing their access permissions. Such research reports became a virtual professional journal with the addition of a suitable editorial board and digital signatures.

Design issues such as the social conventions around periodicity and cost recovery mechanisms would be interesting to investigate using such a framework.

Maintain model neutrality.

To allow multiple conceptualizations requires that authors be able to build rapidly several models over the same media. This came from a practical need to reconcile the very different time-scales involved in designing provisional research schema of annotations and associations vs. designing a MARC-quality archival description of the same set of media. Again, by factorization and abstraction MediaWeaver allowed very different communities to work with media, represented when necessary by proxies, using their own models. Specifically, we had to continually accommodate the work of librarians concerned with conservation and universal classification as well as the needs of the research historian concerned with rapid, unobstructed access to content, and the construction of particular narratives.

Consequently, instead of binding to one particular database, MediaWeaver used a data access framework that allowed connections to any of several standard types of relational database engines over the net, including Sybase and Oracle. MediaWeaver provided an object-oriented abstraction so that its clients did not have to deal with dialects of relational databases. Clients could store arbitrary objects like bitmaps or serialized Objective-C objects as meta-data via MediaWeaver's object-oriented database access framework. In practice, media were kept as source media in ordinary distributed filesystems, and meta-data -- annotations, references, links, abstracts, etc. -- were kept in relational databases. Conceptually, meta-data were regularized pieces of information that commented on data or media objects and provided ways for humans and automated search engines to organize and retrieve those objects. In practice, meta-data were also characterized by being one to several orders of magnitude smaller than the data they described.

Expect evolution.

Perhaps the key to making an scholarly workspace worth using is to
ensure that intellectual content survives across change in technology. This is an institutional commitment as well as a technological issue. Aside from the obvious requirement of a modularized architecture whose components may be replaced without breaking service, the following principles guided our work:

Assume no single data representation.

We did not need to spend resources to converting media systematically to a single format like HTML or SGML. This was one of the most labor-saving operational features of MediaWeaver. By making no assumption about the internal structure of a media entity (a blob), and not even requiring that a media entity existed as bytes in a filesystem, MediaWeaver allowed authors to compose with any computable or renderable medium whatsoever. This way, MediaWeaver could accommodate currently unknown data types and operations. Moreover, this way MediaWeaver could deal with opaque or pre-recorded media (e.g. TIFF, MPEG, AIFF, TeX, Renderman), performable scripts (eg. NS scorefiles, Mathematica notebooks, Applescripts), executables (e.g. a UNIX tool, Hypercard stack, Photoshop plug-in), and data streams (eg. live video channel) with equal ease /difficulty.

How is this feasible? The general principle here is to

Focus on space of transforms more than the base space.

Converting all the authors' source media into some standard structure (such as SGML) was neither cost effective nor strategic in our context because of the diversity of the material (some conversions would lose too much information), the large labor cost (editorial, programmer, administrative), and the constantly changing substance. Moreover, no document structure presented itself as a universal, permanent (on the scale of decades) format that could capture all the semantic and conceptual structures we had in hand. Therefore, I decided it was wiser to build a filter service that

MediaWeaver core objects as well as clients could invoke on foreign platforms.

Assume nothing about the internal structure of a media entity.

A media entity may be a programmatically generated stream of data, a file of any renderable data type, an executable, or may even exist only as a virtual object in a meta-data record. This allows authors to work with proxy objects even when, for legal or technical reasons, primary media are not available. Conversely, multiple versions of a logical media entity can be tracked. The front end application, which could be a custom piece of software or a WWW browser, rather than a centralized logic, decided how to interpret multiple versions of a blob. For example, a movie clip could exist in MPEG as well as Macintosh QuickTime proprietary format. The front end would ask for a version renderable on the local client computer, but authors always dealt only with a single logical entity.

Architecture

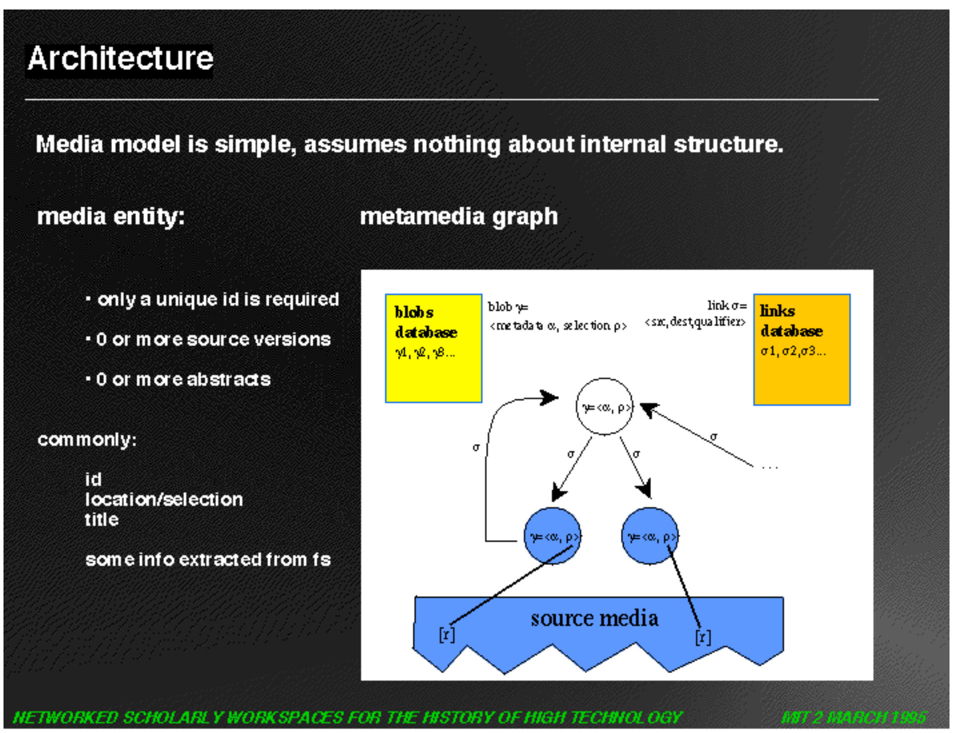

Figure 5. Media Model.

SHA Xin Wei, Cathedral, Tool or Framework? MediaWeaver As a Distributed Scholarly Workspace, in Augmenting Comprehension: Digital Tools and the History of Ideas, ed. Dino Buzzetti, Giuliano Pancaldi, and Harold Short, (London: Office for Humanities Communcation, Kings College London), 2004, pp. 101-112.

The basic media object model is described in Figure 5. A media entity, or blob, has a unique tag, zero or more source versions, and zero or more attributes/metadata fields/abstracts. Typically, the logical media entity is associated to some data ("source media") in persistent storage, but this is not required. By allowing virtual blobs that refer to no source media, we can construct compound structures quite naturally. MediaWeaver has four framework levels (Figure 6):

- A set of user interface kits (Macintosh, NS, WWW),
- A set of mediation class libraries (TCP-IP, WWW, general Service Object Manager, NS DBKit),
- A set of services (mentioned above), and
- Persistent storage (AFS, AppleShare, Sybase).

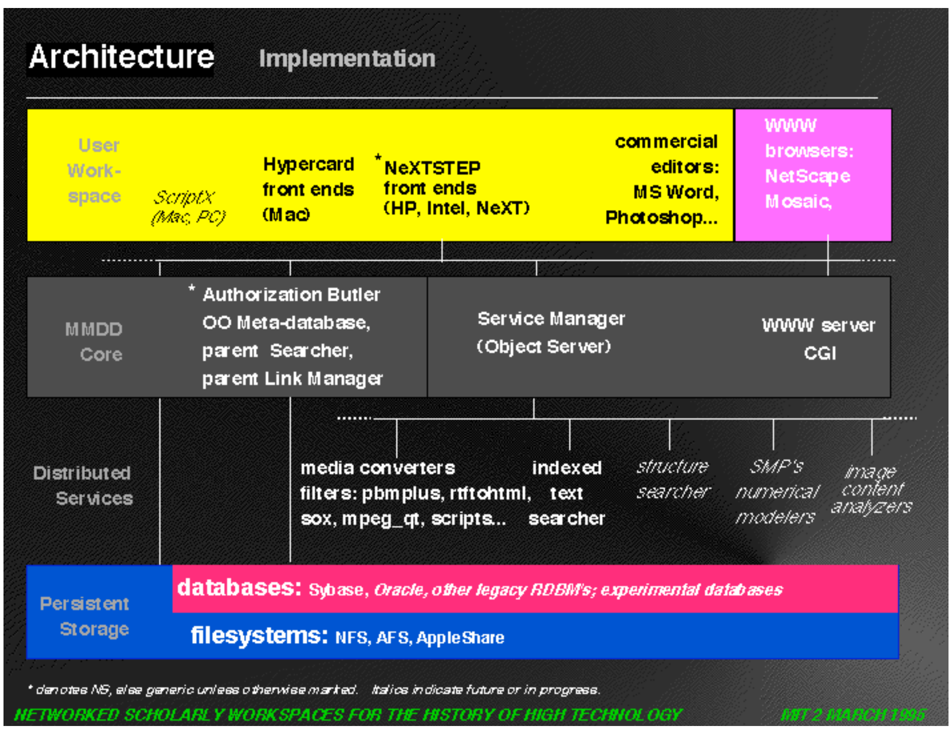

Figure 6. Architecture.

Under the assumption that editors, browsers, search engines, filters, abstractors, and high-level, object-oriented, inter-operable user environments could be added incrementally and in parallel, we invested more of our energy into the service mediation, plus abstract classes which captured the semantics of search, annotation, and association. Over time in the course of supporting different projects, MediaWeaver integrated many complementary tools. For details, see [Sha].

Implications for Historians of Science and Technology

The primary materials of a historian of technology and science can be far more varied than texts, of course. They can include diagrams, hand-written notebooks, illustrated manuscripts, wooden models, mechanical instruments, and for contemporary science, pieces of executable software, databases and video. To be worthy of the name, a technology of media management and annotation should accommodate physical and digital artifacts as well as texts. Databases as generalizations of bibliographies constitute a symbolic interface between the realm of digital representations or artifacts and the realm of material media (paper or celluloid-based media) and physical artifacts. Digital proxies can provide the handles by which a scholar can author narratives richly documented by the full range of archival matter.

However, it is not only the representational forms but the modes of work that are changing given the availability of systems like the MediaWeaver, the World Wide Web, and streaming video. Institutional standards for what constitutes professional work in the historical and philosophical disciplines are changing as scholars adopt digital media technologies and computer-augmented methodologies such as ontology modeling using object-oriented knowledge representations, or live performance with video, sound and web-search. The problem at hand is to evaluate what techniques yield new insight into scholarly issues of importance to the discipline. This disciplinary problem is as thorny as it is reflexive: scholarly concerns in history and philosophy of science, and the norms by which their investigation are conducted will inevitably evolve at the same time that scholars present demands to the technologists. This follows from observing the same processes of exploration, frustration, making demands, and accommodation that scholars in other disciplines undergo as they incorporate computer technologies in their disciplinary practice.

Going to larger social scale, let me comment on G. Pancaldi's observation about the shortage of modest tools for the historical or philosophical scholar. Presented with what he called the very impressive cathedrals of scholarly-technical work such as the Wittgenstein electronic edition (Claus Huitfeldt, University of Bergen) and the Thomas Mann electronic edition (Lothar Rostek, GMD-IPSI, Darmstadt), Pancaldi questioned who would define, design and build the small tools of everyday craft that a working historian of technology like himself could use. Could the software market help define the tools? Would the commercial vendors build what scholars need?

Pancaldi's questions raise many issues that conventionally lie outside the design envelope of computer tools for scholarly research. However, since the MediaWeaver was designed with the benefit of some experience with these very issues in university-based academic software development, I venture a few remarks in response.

Technology of Performance vs. Technology of Representation

As Manfred Thaller pointed out in the symposium in Bologna, September 2000, much of the humanities computing discussion around the publication of the Text Encoding Initiative (TEI) concerned questions of encoding standards for manuscripts. This, in fact is a special case of a broader movement around the same period of time, of a multi-national movement among computer professionals working in multimedia to establish standards for representing all sorts of digital media (HyTime, JPEG, MPEG, to name a few of the more well-known examples). This focus on normalized formal representations of media indicated, to my mind, a hypertrophied drive among academic software developers to try to make their software intercompatible. By itself this is a healthy technical goal, but as Thaller claimed, this also diverted a lot of time and energy from the construction, adaptation and refinement of software tools themselves. All these efforts sprang from a fixation on representation as opposed to performance. [end note 2]

Hollywood Model of Developing Research Software

The post WWII model of big science left an echo in the form of what one could call the Hollywood model of developing academic research software. One example is the NSF sponsored

Geometry Center which was associated with the national center for super-computing (NCSA) in Minneapolis. As supercomputing centers began to lose business and perceived relevance in the 1980's with the adoption of personal computers by physicists and other scientists, they looked for other projects that would require their resources. (This was later formalized as a call for " Grand Challenge" computing projects.) Eminent mathematicians, Fred Almgren from Princeton and William Thurston (Fields Medalist in topology and geometry of hyperbolic manifolds), were given very large resources over a decade to construct software tools for their research, which had to be cast in the mold of a scientific program. For the most part, however, the Geometry Center's technical staff ignored the supercomputer itself and built custom visualization tools that ran on high-end Silicon Graphics workstations.

Since the 1990's however, the university-industry economy no longer could sustain the construction of custom tools for research. Under the current economy of commercial software development, the cost of developing a robust tool of any significance appropriate to a research question is much too expensive in capital and expertise for the narrowness of its market. Even for tools that were developed, such as natural language analysis software or SGML-based editors, the costs associated with technical support and distribution sufficed to stop their dissemination and maintenance across institutions. As Thurston said, what we need are not big factories (cathedrals), but hand tools. [end note 3]

Process vs. Tool

Perhaps we gain purchase on the problem of producing appropriate scholarly technology if we focus on process of scholarly work rather than a tool based on some representation or encoding.

Here is a concrete suggestion. Adapting the tools that Rostek developed for the Thomas Mann knowledge base, a program in history of science and technology could teach students to go through an exercise in building an ontology around a historical complex, sorting out historical agents -- individuals and institutions, relations among these agents, and transcribing that complex into an object-oriented knowledge model. One should think of Rostek's system not as a complete analytic model but as a note-taking device, not as an complete explanatory simulation but as a notational system that a researcher can use to express certain aspects of a historical complex.

To return to Pancaldi's metaphor, the experiment would be to see not how a department can acquire and incorporate readymade cathedrals, but how apprentice historians of science and technology can incorporate and even devise object-oriented computer-based knowledge modeling tools in their work. Let me clearly state that I do not claim any explanatory power for such ontology models. And the danger with any modeling technique of course is always the tendency to elevate the method to an explanation. But in this respect using an agent model system is not different from using, say a semiotic framework. A. Scotti & M. Beretta's web-base is an example of where historians of science are creating digital representations of their artifacts and texts in a coherent media space. Such work constitutes a second transition in the use of digital technology, with online fully streamed digital media as a close echo. The next major transition, presaged by T. Lenoir's SiliconBase and videogame projects [end note 4], could be the development not only of these new modes of representation but the adaptation and incorporation of technologies of performance from scientific simulation, visualization and even gaming to the essential craft of textual analysis.

End Notes

[1] We have in mind notions such as using relational grammars to define meta-layouts for user-interfaces. Examples include WRI's Mathematica 2.3, and work by Weitzman and Wittenburg[Weitzman].

[2] This emphasis is in fact not confined to academic software, but is a hallmark of much computer technology. This design approach stems from the logico-linguistic influences on computer science.

[3] "The trouble is that our tools are not as portable, and not as easy to learn to use as they could be and should be. It's like we have factories to make, and big department stores to sell big powerful industrial drill-presses which you require considerable training to actually use, and which are useful only for special purposes. We ourselves most of the time use hand drills, which we and our potential customers acquire on the black market. If they become proficient with hand tools, it might make sense for them to acquire and use the machines. But we'd all be better off if we spent more effort designing mathematical tools for human use." (William Thurston, mathedu mail group, 1998.)

[4] See these three research projects: Sloan Project on the History of Contemporary Science: Bioinformatics, http://www.stanford.edu/dept/HPS/TimLenoir/bioinformatics.html; History of the Mouse, http://sloan.stanford.edu/mousesite; and How They Got Game: The History of Interactive Simulations and Videogames, http://www.stanford.edu/dept/HPS/VideoGameProposal .

Bibliography

Christopher Alexander, Sara Ishikawa, Murray Silverstein. A Pattern Language. Oxford University Press, 1977.

Pelle Ehn. "Towards a Philosophical Foundation for Skill-Based Participatory Design." Usability: Turning Technologies into Tools, 116-132 in P. Adler and T. Winograd (eds.). Oxford University Press, 1992.

Tim Lenoir, Sha Xin Wei. "Networked Scholarly Workspaces for History of High Technology." Talk at MIT, March 1995, online document available at http://lummi.stanford.edu/Media2/pix/www/MIT/slides. 1994.

Sha Xin Wei, "Metamedia Distributed Databases." Online document available at http://lummi.Stanford.EDU/Media2/ASD/ASD_Homepage/Multimedia.html. 1994.

____, "MediaWeaver -- A Distributed Media Authoring System for Networked Scholarly Workspaces," Special issue on Multimedia Tools, Multimedia Systems Journal. 1996.

Manfred Thaller. mthalle@gwdg.de. Max-Planck Institute for History, Goettingen, Germany. "What is 'source oriented data processing'; What is a 'historical computer science'?" In Historical Informatics: an Essential Tool for Historians? 1994.

Louis Weitzman and Kent Wittenburg. "Automatic Presentation of Multimedia Documents Using Relational Grammars." ACM Multimedia Proceedings. 1994.