Carnegie Mellon University
Research Showcase @ CMU

Human-Computer Interaction Institute | School of Computer Science

1993

AMORE: The advanced Mulitmedia Organizer for Requirements Elicitation

David P. Wood
Carnegie Mellon University

Michael G. Christel
Carnegie Mellon University

Scott Stevens
Carnegie Mellon University

Follow this and additional works at: http://repository.cmu.edu/hcii

This Technical Report is brought to you for free and open access by the School of Computer Science at Research Showcase @ CMU. It has been accepted for inclusion in Human-Computer Interaction Institute by an authorized administrator of Research Showcase @ CMU. For more information, please contact research-showcase@andrew.cmu.edu.

Technical Report
CMU/SEI-93-TR-12
ESC-TR-93-189

AMORE: The Advanced Multimedia Organizer for Requirements Elicitation

Michael G. Christel
David P. Wood
Scott M. Stevens

June 1993

Technical Report
CMU/SEI-93-TR-12
ESC-TR-93-189
June 1993

AMORE: The Advanced Multimedia Organizer for Requirements Elicitation

Michael G. Christel
David P. Wood
Scott M. Stevens

Software Engineering Information Modeling Project

Approved for public release.
Distribution un limited.

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

This report was prepared for the

SEI Joint Program Office
ESC/AVS
Hanscom AFB, MA 01731

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

Review and Approval

This report has been reviewed and is approved for publication.

FOR THE COMMANDER

(signature on file)

Tom Miller, Lt Col, USAF
SEI Joint Program Office

This work is sponsored by the U.S. Department of Defense.

Copyright © 1993 by Carnegie Mellon University

This document is available through Research Access, Inc., 3400 Forbes Avenue, Suite 302, Pittsburgh, PA 15213. Phone: 1-800-685-6510. FAX: (412) 682-6530.

Copies of this document are available through the National Technical Information Service (NTIS). For information on ordering, please contact NTIS directly: National Technical Information Service, U.S. Department of Commerce, Springfield, VA 22161. Phone: (703) 487-4600.

This document is also available through the Defense Technical Information Center (DTIC). DTIC provides access to and transfer of scientific and technical information for DoD personnel, DoD contractors and potential contractors, and other U.S. Government agency personnel and their contractors. To obtain a copy, please contact DTIC directly: Defense Technical Information Center, Attn: FDRA, Cameron Station, Alexandria, VA 22304-6145. Phone: (703) 274-7633.

Use of any trademarks in this report is not intended in any way to infringe on the rights of the trademark holder.

Table of Contents

1 Introduction 1
1.1 Background 1
1.2 Purpose and Organization 2

2 Selected Research Problems for Multimedia Information Systems 3
2.1 A Generalized Model of Information Management Systems 3
2.2 Roadmap of Selected Research Efforts 6

3 AMORE: An Elicitor's Assistant 9
3.1 The AMORE Concept 9
3.1.1 The Motivation for an Elicitor's Assistant 9
3.1.2 The Elicitor's Perspective of AMORE as a Modeling Tool 10
3.1.3 The Elicitor's Perspective of AMORE as a Knowledge Assistant 11
3.2 Potential Solutions to Related Research Problems 12
3.2.1 Multimedia Information Modeling and Manipulation 12
3.2.2 Rule Processing, Browsing, and Mediators 15
3.2.3 Scaling Up and Tertiary Storage 18
3.3 Status of AMORE 20

4 Conclusions 23

Appendix A Multimedia Abstraction Approaches 25
A.1 Multimedia Database Retrieval Mechanisms 26
A.2 Guides 28
A.3 Case-Based Reasoning Systems 29
A.4 Oval 31
A.5 Stratification of Multimedia 32
A.6 Tree-Maps 34
A.7 Cone Trees 36
A.8 Piano Tutor 37
A.9 A Cure for the Common Code 38

References 41

List of Figures

Figure 2-1	Generic Architecture for Information Management Systems	4
Figure 2-2	Examples of Information Management System Architectures	5
Figure 3-1	Example Data Flow Hierarchy for Organizing Requirements	11
Figure 3-2	Different Representations Associated with a Requirement	14
Figure 3-3	Elicitor Guide Discussing Brainstorming Technique	16
Figure 3-4	Use of Past Experience to Aid in New Situations	17
Figure 3-5	Architecture of AMORE Demonstration Prototype	21
Figure 3-6	Future AMORE Architecture	21

AMORE: The Advanced Multimedia Organizer for Requirements Elicitation

Abstract: The advent of larger and more complex software systems has resulted in the need to reconsider the ways in which information pertaining to those systems is stored, visualized, organized, and retrieved. The Software Engineering Information Modeling Project of the Software Engineering Institute is integrating existing and new technologies with the intent of demonstrating feasible technical directions for modeling and managing large amounts of data in multiple media formats. This paper introduces the Advanced Multimedia Organizer for Requirements Elicitation (AMORE), a system that embodies a synthesis of technologies adapted specifically for application to requirements elicitation processes and models.

1 Introduction

1.1 Background

The Software Engineering Information Modeling Project (SEIM) of the Software Engineering Institute (SEI) is developing technology for the creation and manipulation of information models useful in the conveyance of software, system, domain, engineering, and process knowledge. The directions pursued by the project are based on the following premises [Silberschatz 91]:

- The volume of information accessible by computer users is accumulating rapidly.
- The information represented is increasingly complex, with a growing tendency to include various media components, such as images, audio, video, and combinations of these types.
- Existing technologies were designed to support primarily simple information types, such as text or structured graphics. These technologies must be refined, augmented, or replaced by technologies promoting the usability and accessibility of large, multimedia object bases.

To facilitate the evaluation of proposed technological advancements in information modeling, the SEIM Project has chosen to focus initial efforts on the domain of requirements engineering. The requirements engineering domain is particularly useful for this purpose, as there is a persistent failure in the software engineering community to produce satisfactory software requirements [SEI 91], and this failure may be attributed at least in part to several factors [Christel 92c], [Wood 92], [CECOM 89]:

- It is difficult to create structured models from natural language customer descriptions.

- In the transition from raw requirements data to increasingly formal specifications, potentially important information is lost, and traceability is difficult to maintain.
- Engineering legacy from project to project is often lost.
- Policy, process, and methodological information may be available but is often difficult to access.

The SEIM Project is developing technology in support of the capture, representation, analysis, and access of requirements engineering information. The methods and techniques under development are embodied in a prototype modeling environment, the Advanced Multimedia Organizer for Requirements Elicitation (AMORE), which focuses particularly on the area of requirements elicitation.

1.2 Purpose and Organization

The purpose of this paper is to provide a concise description of current and planned functionality of, as well as the rationale behind, the AMORE requirements elicitation assistant. The paper also presents a high-level survey of available or proposed technologies for the storage, visualization, organization, and retrieval of large, diverse information bases. The intent of the survey is not to present an exhaustive assessment and evaluation of the subject technologies, but rather to paint a broad view in support of the selection of technical directions for the AMORE system.

Section 2 discusses some of the anticipated research problems in the development of next-generation systems that will be required to manage large and complex information spaces. This topic is a critical one for AMORE due to the extensive and varied nature of requirements sources. The section also presents a generalized architecture of information management systems in the context of the identified research problems.

Section 3 describes the basic concepts of the AMORE system, relates potential solutions to AMORE goals, and outlines current and future directions for AMORE in light of the issues and technologies discussed throughout the paper.

Section 4 summarizes general conclusions about the problems of multimedia abstraction and the role of the AMORE system in addressing those problems.

Appendix A presents a survey of abstraction approaches. It provides a taxonomy of features to be used as a checklist for the comparison of candidate technologies.To facilitate direct comparison, both an information management issue list and a feature comparison checklist are represented for each abstraction technology discussed in the survey. In addition, a brief description is provided for each technology, together with related reference information.

2 Selected Research Problems for Multimedia Information Systems

2.1 A Generalized Model of Information Management Systems

Typical software systems contain many thousands of requirements. These requirements are derived from many sources and in many formats, hence a tremendous amount of complex raw data comprise the source material for the requirements for a given system. AMORE, described fully in Section 3 of this paper, must have the ability to manage these vast amounts of raw source material for requirements. In addition, AMORE must manage and provide access to knowledge about the problem domains, as well as the process, methods, and tools used for requirements capture, modeling, and analysis. Because AMORE must manage a large amount of complex and diverse types of information, it is important to consider some of the ramifications of information management technologies that are on the horizon.

There are many problems that may impact the future of information management technology, some of which are summarized in the following discussion on databases [Silberschatz 91, p. 111]:

> *Next-generation database applications will have little in common with today's business data processing databases. They will involve much more data, require new capabilities including type extensions, multimedia support, complex objects, rule processing, and archival storage, and will necessitate rethinking the algorithms for almost all DBMS operations.*

The authors proceed to describe some of the major research problems facing developers of next-generation applications. Those relevant to the present paper include:

- *New kinds of data* - Applications increasingly require storage, access, and manipulation of new data types that are physically or semantically complex. For example, unlike simple string or numeric information, next-generation systems will rely on complex data types and objects, image data, audio streams, video streams, and animations.
- *Rule processing* - As information spaces become larger, more semantically complex, and more diverse in content, knowledge-based management of the data may be required to assist users in performing manipulations efficiently.
- *New concepts in data modeling* - Manipulating unstructured data, such as a sequence of sounds or images contained within audio or video clips, will require new techniques to address issues of spatial and temporal data, and accessing information based on informal or partial queries.
- *Scaling up and tertiary storage* - Not only the availability of more data, but also the nature of new data types results in compounded storage requirements. Even a short, highly-compressed video clip may require many megabytes of storage. Rapidly decreasing costs of secondary storage

devices will help to alleviate this problem, but the demand for storage already far outpaces capacity. Research is required to pursue efficient ways to store data within secondary storage capacities, and effective uses of tertiary (off-line) storage.

- Browsing and Mediators - Passive or proactive mechanisms for navigating large information spaces must be developed. Mediators, whether based on pre-defined attributes or based on semi-intelligent assistance, act between the user and the complex information space. Intelligent agents can make large, heterogeneous data available to all classes of users.

Figure 2-1 Generic Architecture for Information Management Systems

These research problems surface within different layers of an information management system. Consider Figure 2-1, which shows a general architecture for such a system. The figure depicts a central core of stored information surrounded by three layers representing consumers of the information:

1. A retrieval mechanism, such as text keyword queries or visual reference queries, or access through links embedded with the data. Possible types of links include:
   - direct links, where traversal of the link always takes you to a specific database object.
   - indirect links, where invoking the link causes some search of the database to take place in order to find the destination(s) for the link. Data can be added to the database, and an old indirect link will be capable of referencing the new data.
   - intelligent links, where invoking the link causes additional knowledge (as in a rule-based expert system or a set of related cases) embedded within the information space to guide the selection of the destination data. As with indirect links, new data can be added to the database and the intelligent link will be capable of referencing the new data. In addition, if

the information manager is allowed to assume control from the user then it can invoke intelligent links based on its own objectives or perhaps the unexpressed needs of the user, resulting in an “intelligent agent” acting on behalf of the user.

2. A visualization mechanism, allowing the user to see and identify relationships between multiple objects in the information space. Representations may be as simple as a linear text list of objects, or as sophisticated as two- or three-dimensional graphical structures, and may make explicit the means of navigation, e.g., virtual movement or hyperlink buttons.

3. An application layer, which may restrict the type of retrieval and visualization being employed to fit within the context of the application.

Some simple but diverse application examples that serve to illustrate the universality of these three layers and their relationships include a jukebox, a library information system, and a flight planner (Figure 2-2).

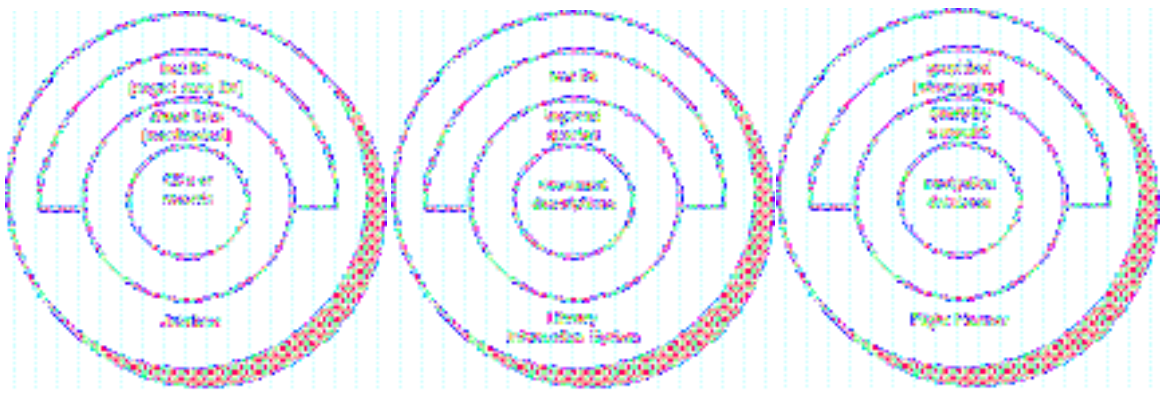


Figure 2-2 Examples of Information Management System Architectures

A jukebox provides a visual index to the user in the form of a listing of available songs. Once a song is selected by the user, the jukebox retrieves the appropriate CD or record through direct (mechanical) links.

An online library information system provides a visual index to the user in the form of listings of books, articles, or other documents. These lists are produced as a result of keyword queries into the library database.

A flight planner provides a means for a pilot to build a flight plan based on navigational data. Visual access to the available data is provided through an airway map, and the pilot is able to select particular waypoints through queries based on attributes of the desired waypoints (e.g., name, type, latitude, longitude, altitude, and so on).

There are numerous available and proposed technologies addressing the retrieval, visualization, or application of multimedia information, and a representative set is surveyed in Appendix A. The survey identifies the focus of the selected technologies (retrieval, visualization, or application) with an iconified version of the architecture diagram depicted in Figure 2-1, and also

presents a tabular listing of the retrieval and visualization mechanisms employed by each technology.

2.2 Roadmap of Selected Research Efforts

In consideration of the information management issues highlighted in Section 2.1, a number of previous research efforts in the area of complex information storage, visualization and retrieval were examined. Overviews of a selection of those research efforts considered relevant to AMORE have been provided in Appendix A of this report and are summarized below:

- *Multimedia Database Retrieval* - the retrieval of multimedia objects through keyword captions, natural language captions, visual examples, or subjective descriptions
- *Guides* - a mechanism for providing the user navigational assistance through data based on pre-defined viewpoints
- *Case-Based Reasoning* - a method of building knowledge-based systems that involves the retrieval of relevant experience and case histories from similar past situations
- *Oval* - a tailorable system for cooperative work permitting customizable information access based on objects, views, agents, and links
- *Stratification of Multimedia* - a mechanism for visualizing video data by organizing in terms of contextual chunks rather than contiguous frames
- *Tree-Maps* - a 2-dimensional method for visualizing hierarchically-structured information
- *Cone Trees* - a 3-dimensional method for visualizing hierarchically-structured information
- *Piano Tutor* - an intelligent multimedia tutoring system
- *A Cure for the Common Code* - a digital video code inspection course utilizing expert system technology

Table 1 indicates the coverage of information management system layers (see Figure 2-1) by each of the above research efforts. This table is intended as only a general guideline; the reader should consult the overviews in Appendix A or the related references for more detailed information on these research efforts.

Over time AMORE will leverage from and, where practical, incorporate techniques from the work surveyed in Appendix A. AMORE must deal with large amounts of data in various forms in support of requirements elicitation, hence multimedia abstraction approaches and information management will be increasingly relevant to the AMORE system as the system matures. The applicability of the surveyed research efforts to AMORE will be discussed in Section 3.2 and Section 3.3.

Table 1. Comparison Chart for Surveyed Approaches				
Approach	Storage	Retrieval	Visualization	Application
Multimedia Database Retrieval	✔	✔		
Guides	✔	✔	✔	
Case-Based Reasoning	✔	✔		
Oval	✔	✔	✔	
Stratification of Multimedia	✔	✔	✔	
Tree-Maps	✔		✔	
Cone Trees	✔		✔	
Piano Tutor	✔	✔	✔	✔
A Cure for the Common Code	✔	✔	✔	✔

3 AMORE: An Elicitor's Assistant

To address the need for capturing, modeling, and manipulating raw system requirements data, the SEIM Project is developing the AMORE system. A demonstration prototype has already been developed and populated with requirements from a US Army movement control system, the Highway Operations System (HOS).[1] The following subsections describe the concepts underlying the AMORE system, potential AMORE solutions to the research problems identified previously in Section 2.1, and the planned evolution of AMORE from a demonstration prototype to an operational prototype (one that will support extensive interactive population and editing capabilities) and beyond.

3.1 The AMORE Concept

3.1.1 The Motivation for an Elicitor's Assistant

Requirements for systems exist in many formats. Depending on the nature of the customer, problem domain, and extant systems, raw requirements data may arise in many ways, such as:

- Informal technical notes
- Notes from meetings
- Statements of Work
- Requests for Proposal
- Operational Concept Documents
- Interviews with customers or users
- Manuals or operational features observed in competing or preceding products
- Technological surveys

Despite the fact that requirements arise in many forms, most existing technologies for capture and representation of requirements focus on models based on textual or graphical notations. In this process, raw requirements are obscured and full traceability is lost. The AMORE concept proposes to address the deficiencies of existing requirements modeling technologies by providing the means for capture, representation, and manipulation of raw requirements data by the use of media most suited to the original formats.

There are many potential users of the AMORE system, including customers, elicitors, system and domain analysts, designers, testers, maintainers, and managers. Each user perceives the raw requirements data in a slightly different way, and, when fully realized, the AMORE system will respond to each type of user in an individualized fashion in order to most effectively assist

1. Screen images from that demonstration will be used to illustrate some of the ideas expressed here. Of course, the operational version of AMORE may look much different than these screen dumps.

in solving elicitation problems specific to each type of user. Because AMORE supports such a diversity of users, describing the system can be complex. For the purposes of this paper, AMORE is described from the perspective of the requirements elicitor.

3.1.2 The Elicitor's Perspective of AMORE as a Modeling Tool

In the process of gathering requirements for a new system, the elicitor may become inundated with a large volume of data in a wide variety of formats. To promote the likelihood of successful analysis and design of the new system, an elicitor must be able to populate AMORE's database as raw requirements data become available. For example, either dynamically during an interview with a customer or during follow-up analysis of the video transcript of the interview, the elicitor will be able to identify and store segments of the interview which support a certain requirement and associate those video segments with that requirement. AMORE must provide for the capture of information related to requirements, as well as organizing the information space in a manner that will support reasoning about the requirements by facilitating browsing, navigating, and searching through them.

The basic organizational unit of AMORE is the requirement. A requirement may comprise a large number of attributes, the simplest of which is a natural language textual description of the requirement. Other attributes, chosen to best suit the nature of the raw requirement, include various graphic, audio, and video representations of the requirement, examples from similar systems, interviews capturing the underlying rationale, design restrictions or suggestions, conflicting opinions, and so forth.

Because large-scale systems may contain many thousands of requirements, it is necessary to provide some type of structural principles for organizing multiple requirements. The AMORE concept is supportive of hierarchical organizational structures, the most common of which are leveled hierarchical data flow/control flow diagrams and object hierarchy diagrams. An example of a data flow hierarchy is shown in Figure 3-1.

For the elicitor, the selected organizational structure is the primary mode of navigation through AMORE. For example, the elicitor can move about different levels of the system hierarchy by means of "double-clicking" through parent/child relationships, similar to the type of navigation commonly found in CASE tools for analysis and design. The elicitor can modify the hierarchy, adding, moving, modifying, and deleting branches as necessary.

Requirements are located at primitive nodes in the chosen hierarchy. The elicitor may add information to any of the requirement attributes at any time to create a "story" that will be meaningful to future browsers of the requirements. For example, a designer may need a better understanding of the core motivations behind a certain requirement. By examining the attributes of that requirement in AMORE, the designer is able to examine the original rationale in its raw format, which might include video clips of a debate among various stakeholders discussing possible tradeoffs and explaining the resulting design constraints. Charts, graphics, and formal descriptions might also be available as supporting documentation.

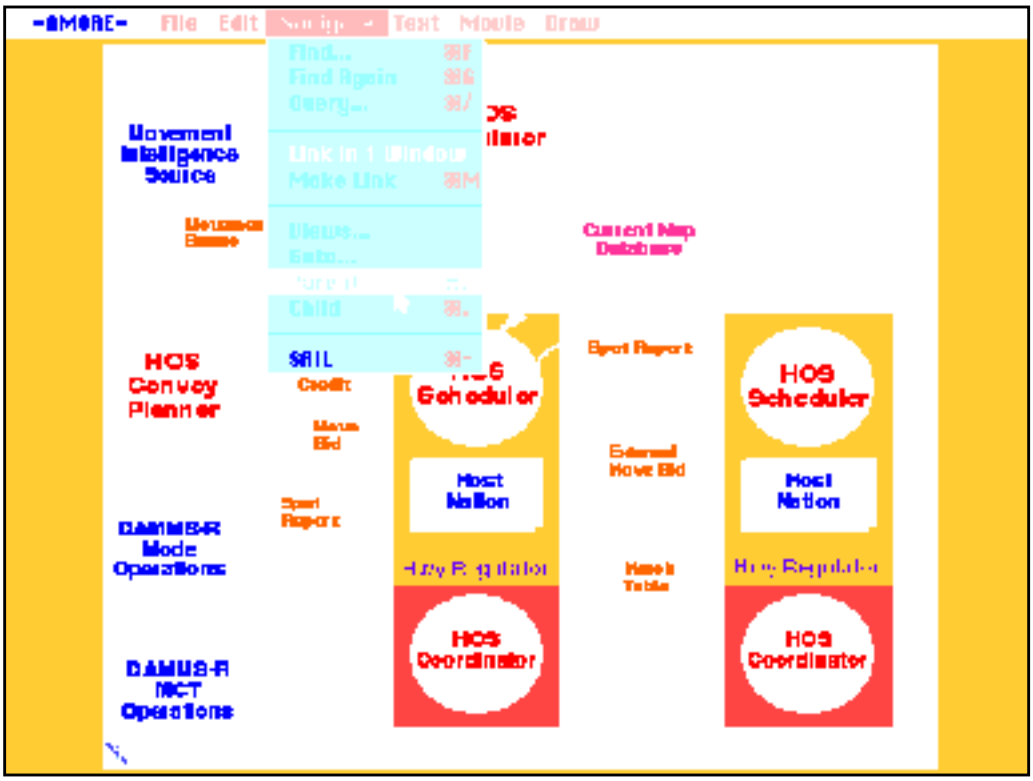


Figure 3-1 Example Data Flow Hierarchy for Organizing Requirements

3.1.3 The Elicitor's Perspective of AMORE as a Knowledge Assistant

While gathering requirements data and populating the database, the elicitor may have need of assistance. To provide as much online, instantaneous assistance as possible, AMORE includes a performance support system. The AMORE performance support system is based on the System for Access to Information and Learning (SAIL) [Hallman 93]. Although SAIL is a general-purpose system, as applied to AMORE, it provides the tools necessary to:

- Capture and preserve knowledge about the requirements elicitation process.
- Teach useful skills that the user needs to be an effective elicitor of requirements.
- Provide the means for preserving and manipulating information about the system under development, as well as preceding developments in the same domain, or other systems of relevance to the system under development.

These broad objectives of SAIL are met in a number of ways. In its simplest form, users interact with SAIL as a repository of diverse information in forms such as context-sensitive help, a data and object dictionary for the system under development, and an electronic process handbook of elicitation tools, techniques, and enterprise practices.

With AMORE, the elicitor interacts with SAIL by selecting an elementary object of interest in order to access further or related information regarding that object, or by selecting the SAIL icon. In Figure 3-2 and Figure 3-3, access to SAIL is represented by the iconic representation of a stack of books in the lower right-hand corner. The text windows (and embedded video windows) shown in Figure 3-3 and Figure 3-4 are examples of the kinds of information that can be represented within SAIL.

As SAIL grows and increasing amounts of domain and elicitation process information become available to the user, more effective information filtering will be required. Guides, better information visualization, rule-based agents, and other techniques surveyed in Appendix A may be incorporated in the future into the system to improve its performance as a knowledge assistant. The anticipated evolutionary paths of AMORE toward more intelligent and proactive user interfaces are described further in Section 3.2.

3.2 Potential Solutions to Related Research Problems

In order to accomplish the objective of modeling raw requirements data, AMORE must be capable of manipulation of multimedia objects, including informal text and graphics, formalisms, still images, audio, video, and animations. Planning for the development of an operational prototype of AMORE has led to questions regarding the storage, visualization, organization, and retrieval of multimedia objects. As discussed in Section 2.1, there are several research problems facing such next-generation applications. The first operational prototype is likely to address only some of these problem areas, prioritized in the order presented in the subsequent subsections. Some of the possible solutions have been derived from previous research efforts which were identified in Section 2.2 and are described in Appendix A.

3.2.1 Multimedia Information Modeling and Manipulation

The research problems identified in Section 2.1 included “new kinds of data” and “new concepts in data modeling.” Both of these problems point out the need for AMORE to accommodate many types of raw requirements data objects. The database underlying AMORE must permit users to navigate through two virtual infrastructures: the *system infrastructure* as embodied in the requirements organization structure, and the *information infrastructure*, as represented by the SAIL electronic handbook. As an analogy, one might think of the system infrastructure of an atlas as a series of topological maps organized by continent, while the information infrastructure might be indexed listings organized alphabetically or by geographical position. Because visualizing such infrastructures becomes more difficult as the infrastructures increase in size and complexity, future versions of AMORE may incorporate large-scale visualization technologies, such as *Cone Trees* (Section A.7), although initial versions will rely on more conventional representations, such as leveled hierarchies, linear indices, and visual links.

The primitive component of the system infrastructure is the *requirement*, while the primitive component of the information infrastructure is the *entry* [Hallman 93]. Although the user will

perceive these two primitive components differently, in implementation they may be interchangeable. For example, SAIL may view a requirement component as just another entry in its information infrastructure.

Each primitive component may comprise one or more representations from any of a wide variety of data types, as shown in Table 2.

Table 2. Examples of Data Types

Type	Format	Examples
text	free format, structured language	interview questions interview transcripts image captions dictionary definitions annotations requirement definitions
graphics	bitmap, structured	sketches drawings tables
images	digital	scanned photographs facsimile documents digitized video stills
audio	digital	audio interviews voice mail messages dictation
video	digital	group elicitation sessions existing fielded systems education/training videos

The infrastructure relating these objects will have different visual representations depending upon mode of use. For example, a requirements elicitor populating the database with requirements will view AMORE from the system infrastructure viewpoint, creating new nodes or rearranging existing nodes based on the system hierarchy that is most relevant. An elicitor who has elected to organize requirements based on a data flow hierarchy may create a new level of decomposition with several objects contained therein, as illustrated by Figure 3-1. Each object may be decomposed further, and may in addition contain multiple representations of information of the types listed in Table 2.

The AMORE user will need to filter, retrieve, and manipulate the information in the database in some way, e.g., for displaying certain text or playing back specific video. In the demonstration system (discussed in more detail in Section 3.3), interaction with the AMORE data is controlled via a window containing a menu of options pertinent to a selected component. For example, if the component is a primitive requirement, the user may have the option to examine an interview question, view the video clip of the resulting answer, view examples of other systems exhibiting similar characteristics, and so on (see Figure 3-2).

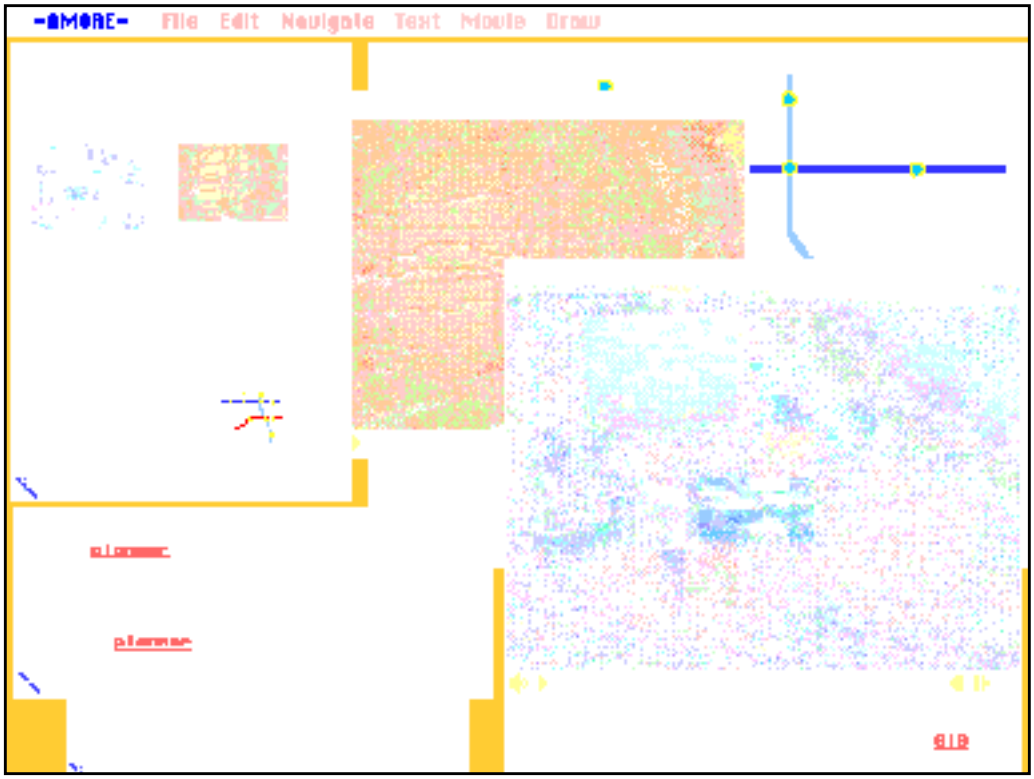


Figure 3-2 Different Representations Associated with a Requirement

The AMORE user may wish to locate information indirectly, perhaps looking for concepts similar to a selected object, but not finding an appropriate choice in a menu of options. User-directed searches through text data could be implemented with keyword queries, keyword class queries, or latent semantic indexing [Dumais 88]. Audio and video data is not so easily searched, as discussed in Section A.1 of Appendix A. A significant amount of labor would seem to be required to augment audio and video data with text descriptors so that the data could be accessed with text-based queries. However, advances in machine speech recognition may soon allow the automatic generation of fairly accurate text transcripts from audio data. The use of a speaker-independent, continuous speech recognition system like CMU's *SPHINX-II* [Huang 92], [Denton 92] could allow the labor involved in creating transcripts for multimedia data to be drastically reduced. The AMORE user could then search through audio and video data as well as text with equal ease, and natural language queries on multimedia data would become feasible.

In addition to examining the data pertaining to a given requirement object, the user may also require additional information or assistance, such as process or domain-specific information. In such instances, the user may elect to use the SAIL knowledge assistant. The user may browse through the available information as one might browse through a book, or the user might engage in computer-assisted navigation techniques, as discussed in the next section. In the demonstration prototype of AMORE, navigation of the performance support system is based in part on multimedia hyperlinks, and provides access to all parts of the requirements

model, process knowledge (including integrated courseware video clips), a data dictionary, domain-specific system knowledge and examples, and operational assistance.

3.2.2 Rule Processing, Browsing, and Mediators

"Rule processing" and "browsing and mediators" were listed in Section 2.1 as two of the major issues facing the future of information management technology. Rule processing deals with knowledge-based management of the data to assist users in performing data manipulations efficiently. Browsing techniques and mediators improve navigation through complex information spaces. More powerful rules and better mediators both lead to intelligent agents which can make large amounts of heterogeneous data available to all classes of users. Thus, the distinction between rule processing and mediating blurs when considering more advanced systems, and hence they shall be considered together here.

Some of the abstraction approaches surveyed in Appendix A discuss rule-based retrieval. *Piano Tutor*, the subject of Section A.8, uses rules to select lessons which will lead to the accomplishment of a student's objective, e.g., to learn how to play a particular song. *A Cure for the Common Code*, discussed in Section A.9, uses rules to select discourses for a code inspection dialogue, leading to the accomplishment of the system's objective of providing a realistic inspection simulation in which the student has an active role. Piano Tutor's rules manage access to over 70 lessons. A Cure for the Common Code's rules manage access to thousands of dialogue components. In both cases the student is shielded from the complexity of the underlying data.

Rules can be used in AMORE for the same purpose: to hide the complexity of the underlying data, and to present to the user those items most appropriate at a given point in time. However, Piano Tutor was based upon the principles of traditional instructional design and the organization of lessons with prerequisite skills and objectives. A Cure for the Common Code was based upon the instructional philosophy of Piaget and ideas of constructivist learning, where students learn about inspections through self-guided exploration, information seeking, discovery, and decision-making in the context of an inspection simulation. Both of these systems had instructional objectives: the skills of playing a piano and participating in a code inspection, respectively. AMORE is to be more than a tutoring system, so its rules may not be based solely on an instructional framework.

AMORE is intended to be a useful tool as well as a tutor for requirements elicitation. Piano Tutor has a distinguished lesson named *Ultima* which has as prerequisites all skills that the student should have upon completion of the curriculum. There will likely be no such well-defined set of skills that users of AMORE are to possess after interacting with the system. Possible types of skills that may be important include:

- The ability to compare and contrast different techniques such as CORE, JAD, QFD, and brainstorming, as outlined in a separate report on elicitation [Christel 92c].
- The ability to use any of these techniques in practice.

- The ability to use a combination of these techniques or new techniques as they become available.
- The ability to avoid recent failures in eliciting requirements for some system and to repeat previous successes at similar activities.

It is likely that each of these concerns could be the driving force behind a set of rules for AMORE data selection and presentation, depending on the interests of the user.

One way to make the controlling influence of the user's interests explicit is through the use of the *guides* technique (Section A.2). When there is a large, amorphous set of data and multiple perspectives on using that data, as is likely with AMORE, guides can be used to visually highlight the controlling perspective. For example, with one system [Salomon 89], historical data could be accessed from a Native American, 18th century frontiersman, or modern day perspective. In another example, a French language-learning application [Adelson 92] makes use of French guides to analyze and argue with individuals in a story set in Paris. With AMORE, guides could be used to represent the perspectives of the requirements writer, end user, domain expert, system designer, customer, and other parties with an interest in the requirements under development.

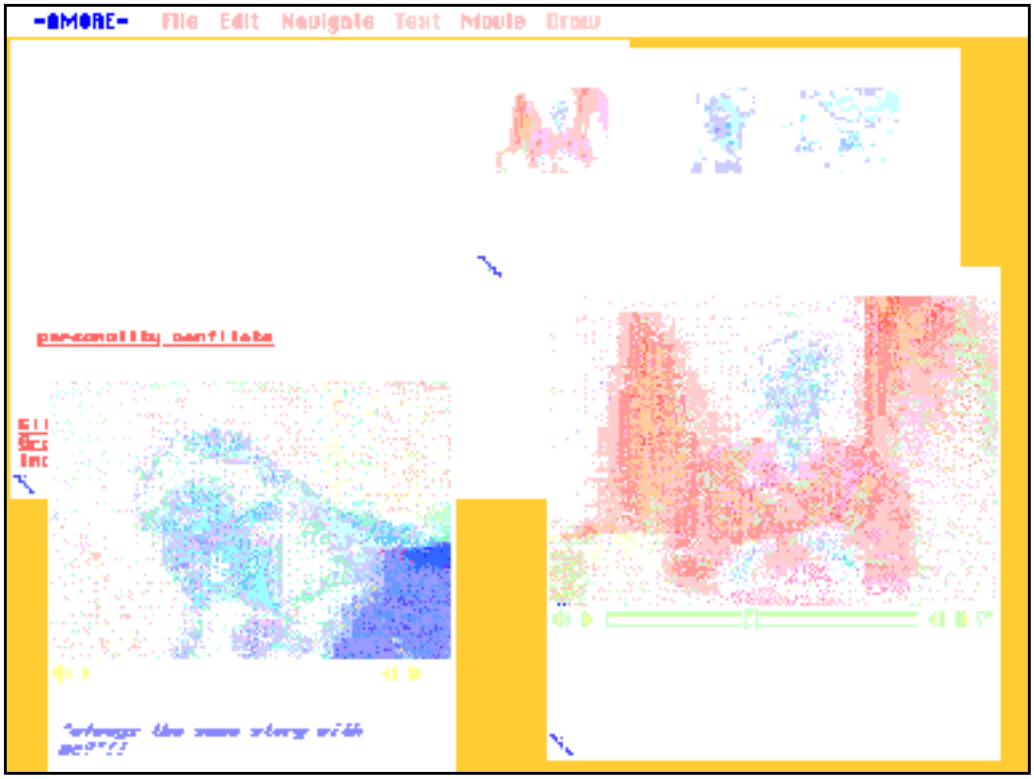


Figure 3-3 Elicitor Guide Discussing Brainstorming Technique

Guides will likely be introduced into AMORE as a way to partition the requirements elicitation database into different perspectives. The data will be labeled as belonging to a certain per-

spective or set of perspectives, and when a guide possessing that perspective is active that data will be more readily accessible. Such partitioning is a simple way to organize the data in a meaningful way to the user, assuming that the guides are meaningful to the user.

For example, consider the case of a requirements elicitor having trouble getting ideas from a group of users. He or she may call upon the advice of the elicitor guide, who then describes possible techniques to use in this situation. Figure 3-3 illustrates the elicitor guide giving information to the user. For added credibility, the guide could be a recognized expert in that particular field.

Via rules or perhaps *case-based reasoning* (Section A.3), guides could evolve toward being more like intelligent agents, assisting the user even before he or she knows help is needed. The guide could suggest a corrective plan of action even before being asked. For example, an elicitor might be repeating the same mistakes made in the past. Rather than allowing the elicitor to continue to make mistakes, a guide could suggest a solution which worked in similar circumstances before. For example, a particularly good summary that brought a group to closure on System A might be saved and then retrieved by an intelligent guide when a similar unresolved dispute is taking place for System B (see Figure 3-4).

Figure 3-4 Use of Past Experience to Aid in New Situations

Being able to save past successes and failures and make those experiences available to all users is one of the strengths of a case-based reasoning system [Kolodner 92b]. The case library acts as a sort of corporate memory and mitigates the effects of employee turnover, which is often cited as a root cause of many elicitation problems [SEI 91].

Intelligent, proactive agents are more difficult to implement than simply using guides to partition the data into different perspectives. Intelligent agents rely on adequate indexing of the data for retrieval and appropriate use of vocabulary for describing entries and requirements. Because of these issues, outlined further in Section A.3 of Appendix A, AMORE guides will first be used to present options to the user, rather than being used to present the definitive answer or approach to the user's situation or need.

With early versions of AMORE, final decision-making and information filtering will remain with the user. The guides will be just another resource available to the user, who remains the catalyst. The user must explicitly invoke the guides to witness what they have to say. As AMORE evolves, more intelligent and proactive capabilities may be added. Perhaps this will be done in a manner similar to *Oval* (Section A.4), but with multimedia objects receiving extra attention. As with Oval, agents can become building blocks for AMORE, with default agents or guides present in the system to help naive users, and with experienced users having the ability to create new agents tailored to their specific needs.

3.2.3 Scaling Up and Tertiary Storage

Information spaces for the next-generation systems [Silberschatz 91] will be huge by comparison to most current systems, and AMORE is no exception. Capture of raw requirements, particularly with digital video, audio, and images, requires vast amounts of storage space. For example, without compression, a one minute, full-screen, full-motion, full-color[1] video clip requires approximately 1800 megabytes (MB) of storage [Lu 93]. Not only does such a requirement far exceed the storage capacity of most currently available hard drives, it also would exceed the data transfer rate of a typical hard drive by about 50 times and that of standard CD-ROM drives by as much as 200 times. Given that today's typical storage devices cannot handle even a one minute video clip, storage of many hours or days of video data is a crucial hurdle for next-generation applications to overcome.

Software capture algorithms, such as those used by the popular QuickTime™ standard of Apple Computer, address the data transfer limitations by selectively restricting one or more of the resolution, frame rate, or color of the captured video. For mid-range Macintosh® computers, a one minute video clip with sound can be restricted to about 625 KB per second (KBps) by reducing the resolution to 1/16 image size (160 by 120), using 16 bit color, and capturing 12 frames per second (fps). The resulting clip would take 37.5 MB of storage space. For faster Macintosh computers, 30 fps can usually be recorded given the same image and color con-

1. For the purposes of this paper, we consider full-size to be a 640 x 480 pixel screen image, full-motion playback to be 30 frames per second, and full-color to be 24-bit.

straints, resulting in a one minute motion video clip without motion artifacts, but one that takes 94 MB of storage space [Lu 93].

Compression algorithms can be used to reduce the vast storage requirements of digital video. With typical software-provided compression under QuickTime, one minute video clips can be reduced to under 5 MB, given the same restrictions of the previous paragraph (1/16 image size, 16 bit color, and 12 fps). Such compression techniques do not preserve the video data in its exact original form, but rather perform approximations on the data which allow the high compression ratio. Such approximations may introduce visual artifacts when the compressed data is played back, depending on the nature of the video and the employed compression algorithm.

If the video capture and compression are both done simultaneously by software, the storage space required by video can be reduced significantly. However, when the CPU has to compress as well as capture video bits to the hard disk at the same time, the capture suffers significantly, with video frames that should have been captured often being ignored completely because the compression throws off the timing of the capture. One solution to this dilemma is to set up a very large buffer area hundreds of megabytes in size and to use that area to capture the video without any compression. Then, after the video is already captured and stored software compression is applied to reduce its size. Such techniques will generally require many minutes to compress each minute of video (i.e., real-time compression is not possible).

Another solution is to use hardware compression. Hardware compression boards produce fast, efficient video compression without taxing the CPU, allowing the video to be captured and compressed simultaneously in real time. Hardware support for compression and presentation, such as that used by Intel's Digital Video Interactive (DVI®) technology, can work within the data transfer rate and storage limitations mentioned earlier while relaxing the constraints on the video being captured, the time needed for compression, and the requirements of the host platform. A mid-range Macintosh computer with DVI technology boards can capture a one minute, 1/16 image size, 8 bit color, 12 fps clip that takes less than 1.5 MB of storage. This video clip has the same apparent quality as a 5 MB clip created with software-only compression under QuickTime (see earlier example), but takes less space in the end and also can be captured and compressed without the need for a 37.5 MB disk buffer area. The same DVI hardware setup could capture a one minute, 1/4 image size, 8 bit color, 30 fps clip in 9 MB of storage at 150 KBps. These limits permit access to higher resolution, full-motion video clips even with a relatively slow standard CD-ROM transfer rate.

Equally important, DVI technology can provide real-time image scaling, permitting a 1/4 image size, full-motion video to be presented as a full-screen, full-motion video. By providing hardware scaling, a 1/4 resolution video can be presented full-screen with an apparent resolution similar to that of a VHS home videotape, a resolution that is adequate for many applications.

In summary, software-only solutions to compression and image manipulation can now overcome transfer rate and storage limitations and compress and store digital video, with some constraints on resolution, color, and frame rate. Hardware solutions can ease these con-

straints to allow full-motion video capture and playback even from CD-ROMs. Hardware solutions also offer the following benefits:

- real-time simultaneous capture and compression of incoming video, rather than a two-step process of capture first to a large buffer and then compress
- real-time image manipulation without critical performance penalties, e.g., scaling of motion video to full-screen while preserving frame rate
- the digital data from hardware-assisted compression is smaller in size and of comparable or better quality than video which is first captured and then compressed via software-only solutions

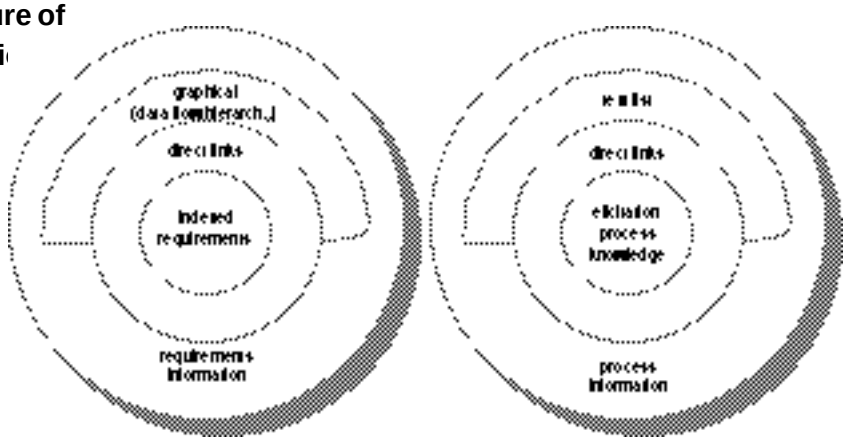
While hardware compression does hold the promise of providing access to full-motion, high-resolution video within the constraints of today's popular personal computer platforms using standard hard drive and CD-ROM storage devices, modeling of very large systems and information spaces with many hours of video clips will require access to more storage than can be provided on a single storage device (a single CD-ROM can hold about 72 minutes of 1/4 image size, full-motion DVI video). Future versions of AMORE will need to examine uses of multi-disk ("jukebox") devices and networked storage techniques to address concerns of scale-up. Meanwhile, the initial operational version of AMORE will assume a single-workstation configuration with access to one or more high-capacity hard drives and CD-ROM units.

3.3 Status of AMORE

The existing demonstration prototype of the AMORE system has been developed for an 80386-based workstation running under MS-DOS. The system is supported by a first generation, 7-board DVI technology audio/video capture and playback configuration, a 500 MB hard drive unit, a CD-ROM unit, and 4 MB of main memory. The system has been populated with requirements objects (including digital video and audio clips) gathered during elicitation sessions for the U.S. Army Highway Operations System (HOS). The SAIL subsystem is populated with elicitation process knowledge, including supporting video training clips from the SEI Continuing Education Series, dictionaries of HOS components and data, and process knowledge based on Total Quality Management objectives. System navigation is achieved through hard-wired hyperlinks with limited interactive object population capabilities. The architectures of this demonstration system are depicted in Figure 3-5.

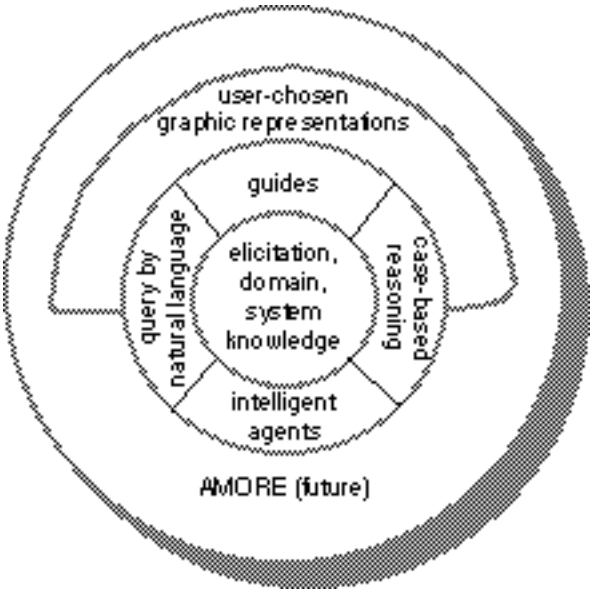
The SEIM Project is presently developing a fully operational ("Version 0") prototype of AMORE. Version 0 will be hosted on a Macintosh-based platform and will support QuickTime video in addition to DVI video formats. Version 0 will be oriented toward a single user configuration and will support interactive capture of digital audio and video, in addition to text and graphics editing. Organizational support for requirements will be available in the form of leveled data flow or object diagrams with an integrated data dictionary. The SAIL subsystem will be fully integrated and will support information retrieval based on casual browsing as well as direct links and query-based searching. The SAIL repository will incorporate requirements elicitation process knowledge based on an electronic handbook concept and will have the capability of being populated with domain-specific and system-specific information.

Figure 3-5 Architecture of AMORE Demonstrati Prototype



Future versions of AMORE (see Figure 3-6) will support user-chosen organizational representations, perhaps in a manner similar to Oval or other tailorable systems. Requirement object templates will be fully configurable by the populator of the AMORE database. SAIL will evolve toward wider support of varied domain and system models, supported by user-selectable graphical visualization mechanisms. AMORE will emphasize growth toward increasingly intelligent performance support for information retrieval. Likely directions include implementations of guides, case-based reasoning, and pattern-recognizing intelligent agents. An anticipated integration with CMU's *SPHINX-II* [Huang 92], [Denton 92] or similar automated transcription systems will increase the likelihood of technological solutions to natural language queries of image, audio, and video data, de-emphasizing the labor-intensive aspects of transcription.

Figure 3-6 Future AMORE Architecture



4 Conclusions

Requirements elicitation is a crucial system development activity which is often plagued with problems of traceability, inadequate conflict detection and resolution, poor communication between elicitation stakeholders, and the loss of important process and system information. A multimedia environment can help address these shortcomings by capturing and organizing the information generated during requirements elicitation. This paper has outlined the underlying concepts and growth plan for one such environment: AMORE, the Advanced Multimedia Organizer for Requirements Elicitation.

The intent of AMORE is to provide the technological support necessary for the efficient capture and organization of raw system requirements. Elicitors will use AMORE as a platform for storing requirements in as close to their natural forms as possible to maximize traceability and to promote understanding of original intentions and motivations. Aside from storing the requirements, AMORE provides the tools necessary to organize them as the first step toward a formalization of the requirements into specifications and designs. The purpose of this organizational support is not to provide yet another data flow CASE tool, but to provide the basic environment necessary to facilitate navigation and browsing of vast quantities of raw data, increasing the likelihood of early detection and correction of conflicts and omissions in requirements.

The intent of the SAIL subsystem of AMORE is to provide automated assistance to elicitors, specifiers, designers, implementors, testers, managers, customers, and anyone else having a need to examine and manipulate requirements. In its simplest form, SAIL can be thought of as an integrated electronic handbook, a repository of information at the fingertips of the AMORE user. As SAIL evolves, it will become an increasingly proactive performance support system, providing interactive guidance and personalized access to process, system, domain, and other information contained within the repository.

AMORE has only been sketched out here in terms of its current implementation, ongoing enhancements, and potential future directions. Forthcoming versions of AMORE will very likely incorporate many of the techniques identified within this paper (and detailed further in Appendix A). As work evolves on the environment, not only will requirements elicitation issues such as traceability, completeness, consistency, and ambiguity be addressed, but the more general issues of capturing, organizing, filtering, and retrieving multimedia information will be addressed as well.

Appendix A Multimedia Abstraction Approaches

High speed networks and the convenience of computer storage in comparison to paper files are putting more and more information at the fingertips of the professional, information which will be increasingly composed of audio and video as well as text. Technology is needed that will empower the user to contribute to these volumes of information, to move through them with understanding, and to find what is desired, without the constant need for expert human assistance. This appendix lists a few of the techniques and applications that concentrate on helping the user effectively access information, analyze it, and synthesize the results.

Section 2.1 of this paper presented an architecture model that will be used to identify the focus area of surveyed abstraction approaches. In addition, a more detailed summary of each technology is provided in a second table. The second table is based on a checklist created by Loeb [Loeb 92] to assist in the classification of information filtering systems. We have adapted this checklist to more closely reflect the concerns of the SEIM Project and to address some of the next-generation concerns discussed previously. The attributes listed in this table for each technology are described below.

1. *User Types* - This attribute describes the viewpoint of the potential users as either casual or proactive. The proactive user will take significant direct action in navigating, browsing, and retrieving objects from the information space. The casual user will tend toward a reliance on some form of mediation to assist in personalized information manipulation.

2. *Information Type* - This attribute describes the nature of the information that is modeled as either static or growing. Further distinction may be made with regard to the data itself, or the descriptors by which that data is accessed. For example, “static data, growing descriptors” would indicate that the data itself may not be modified during a typical session, but the user may provide new links from one object to another in support of personalized access.

3. *Feedback* - This attribute describes whether or not the technology incorporates feedback from the user in making decisions about navigational paths. Are a user’s actions tracked to develop a user model affecting the way subsequent information is retrieved and visualized? Such a model may include complexities such as system determinations on whether the user is naive or experienced, or may simply be based on the explicit selections of the user to utilize a particular viewpoint or filter in accessing data.

4. *Media Composition* - This attribute describes the types of information objects that the technology has been postulated to support. Examples might include text, bitmaps, structured graphics, audio, analog video, digital video, animations, simulations, or constructs with formal semantics.

5. *Platform* - This attribute describes the platform on which the technology has been demonstrated. This should not be taken as an indication that implementation is possible on only that platform.

6. *Information Attributes* - This attribute describes the ways in which the objects contained within the information space can be identified for access.

A.1 Multimedia Database Retrieval Mechanisms

	Retrieval Mechanism	Visualization Mechanism
Retrieval	Query by keyword caption, Query by visual example, Query by natural language caption, Query by subjective description; Direct and indirect links	n/a

Retrieval of multimedia data (e.g., sound, image, and video) from a database is more difficult than text retrieval because the information contained in such data is unstructured, yet it embodies complex semantics. For example, much of the information relevant to a video clip of a meeting may be contained in the emotions and body language of the participants.

Several solutions to the problem of multimedia database retrieval have been posed [Keim 91], [Kato 91], [Hirata 92], [Cesarini 92], which can be summarized as:

- keyword caption
- natural language caption
- visual example
- subjective descriptions

Keyword caption queries, commonly used for text retrieval, require a substantial investment on the part of the database populator to examine each object and condense its content into a set of keywords. Such an approach is simple to implement, but labor intensive. Further, the keyword approach will tend not to capture the rich semantics contained within the data object.

Natural language caption [Keim 91] is proposed as a solution to some of the difficulties inherent in keyword captions. Artificial intelligence techniques are used to parse the populator-provided caption written in a natural language subset. Matching is accomplished in concert with a pre-defined domain-specific knowledge base to improve matching reliability by using a weighted partial-matching scheme. Like the keyword caption approach, the natural language caption approach will tend to be labor intensive due to the need for abstraction of unstructured multimedia objects by the populator of the database.

Query by visual example [Hirata 92] enables retrieval of visual objects through the use of visual queries. Pattern recognition algorithms are presented which enable the database management system to compare images in the database and return a selection of similar images to the user. The query image may be a hand-drawn sketch, a photograph, or a photocopy. Each image in the database has a counterpart in an abstract pictorial index, against which the query image is compared. Hirata describes experimental results that showed roughly a 95% accuracy rate among the best five matches for a given query based on rough sketches, from a database of 205 distinct images. Good results were also reported when using other types of input images. At present, the storage costs and computational costs (for pattern matching) limit the approach to small resolution images, on the order of 64 x 64 pixels.

Query by subjective descriptions [Kato 91] is similar to query by visual example, but it relies on a *sense retrieval* function. Here, the user need not provide a sketch or other visual input, but rather provides a subjective textual description of that image. For example, one might request images that are “clear, bright, clean.” Matching is based on a “personal index”, under the assumption that different user audiences will view the images from different perspectives. The advantage to this approach over simple keyword approaches is that the populator need not painstakingly enter keywords for each entry. Rather, the system will examine each image and choose which personal indices to use based on user profiles.

User Types	Info Type	Feedback	Media Composition	Platform	Information Attributes
n/a	n/a	yes	focus is on graphic or photographic still images	n/a	textual captions (indices), image pattern recognition

A.2 Guides

	Retrieval Mechanism	Visualization Mechanism
Retrieval, Visualization	guides, direct hyperlinks	hypertext keywords, iconic guide representations

In browsing a simple database, the user supplies a series of related keywords as part of a query operation. The database management system returns to the user a list of all database entries that match the query. The emergence of massive, multimedia databases can make simple queries problematic in that the volume of matches may be overwhelming unless the query is very specific. Salomon, *et al*, describe a method for assisting in the navigation of large multimedia databases by allowing the user to select an identity known as a *guide* [Salomon 89], [Oren 90]. A guide is an iconic representation of a particular viewpoint that the user of the database might be expected to take. Each guide has associated attributes that represent the guide's viewpoint and provide the basis for a more focused search.

For example, a database on early American history may contain thousands of multimedia artifacts relating to geographic, ethnic, religious, political, scientific, social, or other historical information. A simple database search on the topic of warfare might result in hundreds or thousands of artifacts, most of which may not be useful for the user's purposes or intents. The user may wish to focus on warfare from the point of view of a modern weapons manufacturer, an 18th century frontiersman, the Chairman of the Joint Chiefs of Staff, or a member of a native tribe. The user may select a particular guide as a representative; this viewpoint then becomes an integral part of subsequent database navigation. Guides have been used in similar ways in other instructional applications, e.g., a French language tutor [Adelson 92].

A browsing mechanism called a *tour* provides casual observers with a cohesive audio-visual presentation of a topical theme based on the viewpoint of a selected guide. The user may passively view the tour or stop it at any time to follow related threads through the use of hyperlinks which are provided throughout the tour.

The use of guides has been informally demonstrated with students, teachers, and administrators of elementary and secondary schools, with largely positive results, although tours were not as effective as hoped. The authors of the technique speculate that greater interactivity and higher video resolution would make tours more effective.

User Types	Info Type	Feedback	Media Composition	Platform	Information Attributes
proactive and casual	static	yes	text graphic images audio video animation	Mac II 1M CD-ROM Hypercard Videodisc Player	descriptors organized into classes based on guides

A.3 Case-Based Reasoning Systems

	Retrieval Mechanism	Visualization Mechanism
Retrieval	indices, intelligent links	n/a

Case-based reasoning (CBR) is a method of building knowledge-based systems that involves the retrieval of relevant experience and case histories from similar past situations. It is a different way of constructing solutions than that of rule-based reasoning, which relies on chains of deductive reasoning. The basic premise of the case-based technique is to solve a problem by referring to previously solved problems, by searching for similar past cases and adapting them [Kolodner 92a].

The data in a multimedia database could be organized as cases. Then CBR could be used to answer such questions as “If I know nothing about one-on-one interviewing, but have often performed in group elicitation techniques and am an experienced systems analyst, what should I read or view to learn more, and what should I do to have the best chance of a good solo interview?” If the multimedia database is used to capture engineering legacy, i.e., record past successes to emulate and failures to avoid, then cases are particularly applicable for allowing this old experience to be recorded in a way that it can be retrieved to solve new problems. Advantages of CBR include the following [Kolodner 92a]:

- CBR allows the reasoner to propose solutions to problems quickly, avoiding the time necessary to derive those answers from scratch.
- CBR allows a reasoner to propose solutions in domains that may not be understood completely; assumptions and predictions are made based on what worked in the past.
- CBR gives a reasoner a means of evaluating solutions when no algorithmic method is available for evaluation.
- Cases are particularly useful for use in interpreting open-ended and ill-defined concepts.

CBR is central to four teaching programs built at the Institute for the Learning Sciences [Schank 91]. These programs are based on the following plan:

1. Present an interesting situation for a student to try out.
2. Allow the student to fail.
3. Have a story ready to tell that relates to what the student did wrong.
4. Tell the story when consulted.

The hardest parts of CBR include the indexing of data, the use of correct vocabulary, and the selection of a good framework in which to describe your cases. The third step in the plan above is heavily reliant on a good indexing scheme. CBR works well for animal or zoo games and encyclopedias, because the framework for such a domain is well researched, developed, and accepted (family-genus-species). Likewise, CBR works well for finding solutions which are essentially lists, e.g., a menu for a party, because the vocabulary for the cases then easily defaults to the fields which compose the list (main dish, side dishes, appetizer, etc.). For multimedia databases without a default selection for a hierarchical framework and lacking a common vocabulary, such as software engineering, the indexing problem can be a major inhibitor to the organization of the data as cases. A practical concern is that the matching and retrieval of indexed cases must be fast if interactive applications are to incorporate CBR.

Case-based reasoning systems can learn from their past experiences if feedback is given as to the correctness of its proposed solutions. Such feedback allows the reasoning system to repair its information model. Of course, if the cases in a multimedia database are allowed to be updated, then configuration management issues become important, for any replacement data may alter dependencies based on the old data.

Kolodner distinguishes between two extremes of CBR systems [Kolodner 92a, p. 32]:

> *Fully automated systems are those that solve problems completely by themselves and have some means of interacting with the world to receive feedback on their decisions. Retrieval-only systems... act to augment a person's memory, providing cases for the person to consider that the person might not be aware of himself, but the person will be responsible for the hard decisions.*

CBR could be incrementally introduced into a multimedia database, beginning with retrieval-only systems and perhaps evolving into fully automated systems.

User Types	Info Type	Feedback	Media Composition	Platform	Information Attributes
casual and proactive	growing data; growing descriptors	yes	all types, with video having recognized benefits for representing stories/cases	Mac II & videodisc jukebox	cases organized by some indexing scheme

A.4 Oval

	Retrieval Mechanism	Visualization Mechanism
Retrieval, Visualization	user-customizable views and event-driven agents; direct and indirect links (including those created by the user)	prespecified set of display formats, including text tables, graphic networks, matrices and calendars, shown in overlapping Macintosh windows

Oval is a tailorable system for cooperative work that was built at MIT under the direction of Thomas Malone [Malone 92]. The name “Oval” is an acronym for the four key building blocks of the system: objects, views, agents, and links. These four building blocks provide an elementary “tailoring language” for user interfaces to information management applications, and the tailorability provided by Oval allows a variety of applications to be constructed within the same system. For example, Oval can be used to construct cooperative work tools such as gIBIS, Coordinator, Notes, and Information Lens [Malone 92].

Proactive users may define agents and customize views so that information access and presentation are tailored to their needs. Casual users may utilize preexisting agents and views for information filtering until they are comfortable enough with or face the need to define filtering rules themselves. Earlier research on Information Lens, a predecessor of Oval, showed that even people without significant computer experience are able to create and use rules effectively [Mackay 89]. While Oval allows users to define new agents and rules, create new object types, add fields to existing object types, and insert new links, Oval does not allow new display formats to be defined from scratch by the user for visualizing information. Rather, the user can select views for objects and collections of objects from a prespecified set of display formats and then specify parameters for a given view (such as which fields to show).

User Types	Info Type	Feedback	Media Composition	Platform	Information Attributes
proactive and casual	growing data; growing descriptors	yes	“radical tailorability” supposedly allows Oval to manage all types of info; no special predefined support for multimedia objects	networked Apple Macintoshes (Oval is implemented in Macintosh Common Lisp)	object fields (descriptors), links

A.5 Stratification of Multimedia

	Retrieval Mechanism	Visualization Mechanism
Retrieval, Visualization	iconic direct links, keyword classes used for indirect links	stratagraph (graphical display of strata)

The stratification of multimedia [Davenport 91], [Smith 91], is a means by which video can be segmented into contextual chunks rather than contiguous frames. It is based on the following assumptions:

- video is randomly accessible; any subset of a video data item can be played back and combined with other video segments
- content can be represented in layers, or strata
- any small piece of video may be part of a number of strata; a piece of video's content is derived by examining the union of all the contextual descriptions that are associated with it
- for a machine to select and sequence video footage, the content of the footage must be represented in machine-readable form
- we need descriptions that present us with longer or shorter elements according to the detail our query requires; descriptions of content must be structured at the shot level to maximize the potential for browsing and sequence assembly

Information in a video shot is described in terms of cinematic primitives, including: perspective, camera, sound, content, and context [Davenport 91]. Ambient sound can be used to provide contextual clues as to when one shot ends and another begins [Smith 91].

Describing video with keywords can be problematic; different video clips may be described with completely different terminology so that no comparisons can be made [Smith 91]. The authors of the technique solve this problem by using a set of primitive descriptors, or keyword classes. However, other sources note that keyword classes are expert-labor intensive to create and are not very successful in overcoming the problems of polysemy (more than one meaning for a single word) and synonymy (many ways of referring to the same concept) [Dumais 88], [Salton 83]. Latent semantic indexing [Dumais 88] could improve the access to video shots described by cinematic primitives.

As presented in the cited articles, the stratification of multimedia focuses on documenting a fixed set of video data, such as that found on a videodisc [Davenport 91, [Smith 91]. The doc-

umentation, or strata, may accumulate over time, better describing the set of data from a variety of viewpoints.

User Types	Info Type	Feedback	Media Composition	Platform	Information Attributes
casual	static data; growing descriptors	no	analog audio and video (digital possible but not illustrated in the cited articles)	Unix workstation, X/Motif, Parallax video card, analog video source	video segment descriptors organized into classes (including cinematic primitives)

A.6 Tree-Maps

	Retrieval Mechanism	Visualization Mechanism
Visualization	retrieval mechanism set by visualization technique: selection of rectangular bounding boxes, direct links	rectangular bounding boxes with color attributes

A Tree-Maps is a method for the visualization of hierarchically structured information [Johnson 91]. It is based on the following assumptions:

- a large quantity of the world's information is hierarchically structured (such as manuals, library cataloging, directory structures)
- people can grasp the content of a picture much faster than they can scan and understand text
- efficient use of space is essential for the presentation of large information structures
- displaying the entire information structure at once allows users to move rapidly to any location in the space

Hierarchical information structures contain two kinds of information: "structural (organization) information associated with the hierarchy, and content information associated with each node" [Johnson 91, p. 284]. Tree-Maps are able to depict both the structure and content of the hierarchy via rectangular bounding boxes filling all of the designated display space. The area of the bounding box is proportional to a user-selected property associated with content, such as the size of a file. The display location and orientation (tall vs. wide) of the bounding box indicates structure, such as the location of a file within a directory tree. In addition to bounding box area, color and sound could be used to represent two other properties of the nodes in the hierarchy [Shneiderman 93]. Tree-Maps contrast with traditional static methods of displaying hierarchical information, which generally make either poor use of screen space or hide vast quantities of information from users.

Limitations of Tree-Maps include the following:

- They are best suited to hierarchies in which the content of the leaf nodes and the structure of the hierarchy are of primary importance, and the content information associated with the internal nodes is largely derived from their children.
- If there are a number of content properties of interest to the user, then either a series of Tree-Maps will need to be presented, or some of the content will need to be hidden in popup windows and presented only upon user selection. This weakens the assumption that Tree-Maps can display the entire information structure. The number and variety of domain properties that can be statically coded in the drawing of the tree are limited.

- The display resolution imposes limits on the size of the hierarchy that can be presented. This limit is on the order of a thousand objects.
- If there is a great range of node weights, i.e., the user-selected property controlling the area of the bounding box, then the display will be dominated by those bounding boxes with a high node weight, and all the boxes with relatively small node weights will often become completely black regions because there will only be enough screen space to draw their borders. It could be argued that since the assignment of node weights is user controlled, presumably the user is most interested in the nodes with greatest weights and the nodes with the smallest weights are of least interest. However, the Tree-Map would no longer be displaying the entire information structure: the only way to access the nodes with the small weights would be to magnify those areas.
- Some commonality ought to exist among the nodes so that the user can select a property for which each node can be assigned a given weight. For example, with computer directories each file has a file size which can be used as this common property, but with a collection of disparate technical reports the selection of a meaningful weighting function may be more difficult.

For a certain class of hierarchical structures, Tree-Maps provide a way to organize and present the information in a single display space for easier user comprehension and retrieval. This class would have most of the important information in the leaf nodes and hierarchical structure, concentrated on a small set of properties with a limited range of values. It is proactive in the sense that the user defines which properties are to be used for the node weights, and hence the user's selection of properties determines the presentation of the bounding boxes.

User Types	Info Type	Feedback	Media Composition	Platform	Information Attributes
proactive	static data; growing descriptors	yes	graphic front-end to set of hierarchical data (which may be multi-media data)	Mac II	visual descriptors (rectangular bounding boxes) of hierarchical data

A.7 Cone Trees

	Retrieval Mechanism	Visualization Mechanism
Visualization	retrieval mechanism set by visualization technique: 3-D model manipulation, assisted by cognitive co-processor; direct links	3-D conic representation rendered in a 2-D workspace

Cone trees [Robertson 91] are one mechanism for information representation used by the *Information Visualizer* [Card 91], [Clarkson 91]. The Information Visualizer is an information modeling and retrieval system developed at the Xerox Palo Alto Research Center. The basic premises of the Information Visualizer are

- that information that is readily available visually is far less expensive in terms of the physical and cognitive activities required for access than information that is out of view, and
- even large-screen workstations suffer from lack of sufficient screen real estate to depict large and diverse information spaces.

The Visualizer makes use of 3-D rendering effects on 2-D screens, in combination with smooth, interactive animation in an attempt to shift cognitive processing load to the human perceptual system. The intent of the Visualizer approach is to provide greater visual access to large information spaces within the confines of conventional workstation hardware.

Cone trees are a representation of hierarchical tree information structures. The parent node is the apex of a cone, while children nodes are evenly distributed about the base of the cone. These children, in turn, are at the apex of their own cones. The entire structure appears to the user to be three-dimensional, resulting in perceptual recognition of information that is partially hidden “behind” other information. The user may navigate the structure through rotation of the cone tree, as well as zooming toward and away from it. To enhance the user’s perception of the cone tree as a real, physical entity, an interactive animation technique is employed to ensure that rotations of the structure achieve a consistent speed that is neither so slow as to be obtrusive, nor so fast as to cause perceptual disorientation. In addition to browsing, facilities are provided for dynamic reconfiguration of the hierarchy and for template-based searching.

An experimental model of a 650-node organizational chart was easily accommodated on a conventional-size workstation display using a cone tree representation, while an equivalent 2-D depiction was estimated to require three feet of screen space [Robertson 91]. Cone trees have been used experimentally to model as many as 10,000 nodes, although practical maximums of 1,000 total nodes, 10 levels of hierarchy, and not more than 30 branches for a given node have been suggested.

User Types	Info Type	Feedback	Media Composition	Platform	Information Attributes
proactive	static	yes	graphic	Silicon Graphics Iris	tree family relationships

A.8 Piano Tutor

	Retrieval Mechanism	Visualization Mechanism
Application, Retrieval, Visualization	skills (lessons chosen to meet student's needs based on student's skill set and student's goal); intelligent links	music notes graphically updated synchronously with live music; analog video and digital audio examples; Macintosh windows

The Piano Tutor is an intelligent tutoring system developed at Carnegie Mellon University for teaching first year piano playing. The system verifies the starting time, pitch, dynamic level, and duration of each note the student plays. It then compares the note with a model performance programmed in the system and provides automatic error checking and correction to the student. Piano Tutor runs on a Macintosh II computer with an electronic keyboard or synthesizer, a musical instrument digital interface (MIDI), and a videodisc player.

Piano Tutor employs real-time score following technology to allow the synchronization of computer-synthesized music with a live performance by the student. Piano Tutor is also based on instructional design techniques which allow the system to present lessons to the student that are directed to the student's needs [Capell 93].

The system is composed of numerous lessons, each with prerequisites, objectives, a presentation, and an evaluation. This representation of the Piano Tutor's curriculum allows the curriculum to be extensively analyzed by the computer. Analysis might prove that in order for the student to learn a particular skill s, he or she must first take lesson l. Analysis can identify:

- useless skills, which do not help the student advance in the curriculum;
- patently unobtainable skills, which are not taught by any lesson;
- equivalent skills, which are interchangeable in the curriculum design; and
- a short lesson path toward accomplishing a student's goal.

User Types	Info Type	Feedback	Media Composition	Platform	Information Attributes
proactive and casual	static data plus student's input	yes	primarily digital audio; analog video	Apple Macintosh II, MIDI, videodisc	lessons comprised of prerequisite skills, objectives (taught skills), presentation, and evaluation

A.9 A Cure for the Common Code

	Retrieval Mechanism	Visualization Mechanism
Application, Retrieval, Visualization	intelligent links (expert system controlling dialogue for code inspection simulation)	Virtual world, including a code inspection simulation with 3 computer-controlled reviewers

“A Cure for the Common Code” is the world’s first AI-based all-digital video simulation. Developed at the SEI, this course teaches code inspections, which are formal software technical reviews involving four (or more) participants. “A Cure for the Common Code” places the student in a virtual world, a fictitious software development company named Ultimex, where the student learns about and actively participates in code inspections. Other references present the design of the course [Stevens 89] and a summary of students’ experiences with the course [Christel 92b].

This course needs to manage thousands of dialogue components which are pieced together by an expert system to create the code inspection simulation. These dialogue components, or discourses, total over ten hours of digital audio and ninety minutes of digital video. They are stored locally on a hard disk and CD-ROM. Based on an indexing scheme that richly describes the multimedia data, the expert system locates a relevant discourse for the current state of the inspection. This indexing allows the discourses to be used as building blocks so that inspection conversations can be constructed dynamically.

As with the stratification of multimedia (Section A.5), the descriptors for video segments indicate not only content but also directorial information, such as context and camera angle. However, “A Cure for the Common Code” takes full advantage of the flexibility of digital video to use such descriptors for dynamic scene creation and behavioral modeling [Christel 92a]. The user is presented and interacts with a situation dynamically constructed from the video segments, a code inspection, rather than being shown the video segment descriptors as layers of strata or in any other similarly passive representation [Stevens 92]. The situation is created based on the student’s contributions as well as on the current behavioral state of three other inspectors modeled by the system. Personality factors for these inspectors, including talkativeness, defensiveness, aggressiveness, and wit, are modified during the course of the inspection according to the portion of the expert system dealing with simulating inspections and discourse selection.

Another portion of the expert system is devoted to managing presentation. Digital video scenes are composed dynamically to match the current state of the inspection and to give the user appropriate feedback, utilizing directorial knowledge stored in the expert system. For example, if the user has been dominating the conversation, he or she may be presented with scenes of the other reviewers which were recorded with the camera pointing down at these

other reviewers. The student's perspective will then be that of looking down at the other inspection participants, which enhances the apparent dominance of the student in the inspection.

User Types	Info Type	Feedback	Media Composition	Platform	Information Attributes
casual	static data plus student's input	yes	digital video; digital audio, graphics, text	Intel 80386 machine, Digital Video Interactive technology 7 board set, CD-ROM	text descriptors of multimedia dialogue components (such as speaker, topic, tone); directorial information

References

[CECOM 89] Communications-Electronics Command. *Proceedings of the Requirements Engineering and Rapid Prototyping Workshop.* Fort Monmouth, NJ: Center for Software Engineering, U.S. Army Communications-Electronics Command, November 14-16, 1989.

[SEI 91] Software Engineering Institute Requirements Engineering Project. *Requirements Engineering and Analysis Workshop Proceedings* (CMU/SEI-91-TR-30, ADA250415). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, December 1991.

[Adelson 92] Adelson, Beth. "Evocative Agents and Multi-Media Interface Design," pp. 351-356. *Proceedings of the CHI '92 Conference on Human Factors in Computing Systems (Monterey, CA)*. New York: ACM, May 3-7, 1992.

[Capell 93] Capell, Peter; & Dannenberg, Roger B. "Instructional Design and Intelligent Tutoring: Theory and the Precision of Design." To appear in the *Journal of Artificial Intelligence in Education 4*, 1993.

[Card 91] Card, Stuart K.; Robertson, George G.; & Mackinlay, Jock D. "The Information Visualizer, An Information Workspace," pp. 181-188. *Proceedings of the CHI '91 Conference on Human Factors in Computing Systems (New Orleans, LA).* New York: ACM, April 28-May 2, 1991.

[Cesarini 92] Cesarini, F.; Mazzoni, A.; & Soda, G. "Modelling and Managing Linked Multimedia Dossiers," pp. 219-224. *Proceedings of the Tenth IASTED International Conference (Innsbruck, Austria)*. Zurich: Acta Press, February 10-12, 1992.

[Christel 92a] Christel, Michael G.; & Stevens, Scott M. "Rule Base and Digital Video Technologies Applied to Training Simulations." *SEI Technical Review '92*. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University 1992.

[Christel 92b] Christel, Michael G. "Experiences with an Interactive Video Code Inspection Laboratory," pp. 395-411. *Lecture Notes in Computer Science 640: Software Engineering Education SEI Conference Proceedings (San Diego, CA).* Berlin: Springer-Verlag, October 5-7, 1992.

[Christel 92c] Christel, Michael G.; & Kang, Kyo C. *Issues in Requirements Elicitation* (CMU/SEI-92-TR-12, ADA258932). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, September 1992.

[Clarkson 91] Clarkson, Mark A. “An Easier Interface.” *Byte 17*,2 (February 1991): 277-282.

[Davenport 91] Davenport, Glorianna; Smith, Thomas Aguierre; & Pincever, Natalio. “Cinematic Primitives for Multimedia.” *IEEE Computer Graphics and Applications 11*, 4 (July 1991): 67-74.

[Denton 92] Denton, Jutta S.; & Taylor, C. Roy, eds. *Final Report on Speech Recognition Research* (Research Paper CMU-CS-92-157). Pittsburgh, PA: Carnegie Mellon University School of Computer Science, June 1992.

[Dumais 88] Dumais, Susan T.; Furnas, George W.; & Landauer, Thomas K. “Using Latent Semantic Analysis to Improve Access to Textual Information,” pp. 281-285. *Proceedings of the CHI ‘88 Conference on Human Factors in Computing Systems (Washington, D.C.)*. New York: ACM, May 15-19, 1988.

[Hallman 93] Hallman, Harvey K. *A System for Access to Information and Learning* (SEI SEIM Project Report). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, January 1993.

[Hirata 92] Hirata, Kyoji; & Kato, Toshikazu. “Query by Visual Example,” 56-71. *Proceedings of the EDTB ‘92 Third International Conference on Extending Database Technology (Vienna, Austria)*. Berlin: Springer-Verlag, March 23-27, 1992.

[Huang 92] Huang, Xuedong; Alleva, Fileno; Hon, Hsiao-Wuen; Hwang, Mei-Yuh; & Rosenfeld, Ronald. *SPHINX-II Speech Recognition System: An Overview* (Research Paper CMU-CS-92-112). Pittsburgh, PA: Carnegie Mellon University School of Computer Science, January, 1992.

[Johnson 91] Johnson, Brian; & Shneiderman, Ben. “Tree-Maps: A Space-Filling Approach to the Visualization of Hierarchical Information Structures,” pp. 284-291. *Proceedings of the IEEE 1991 Visualization Conference (San Diego, CA).* New York: IEEE Computer Society Press, October 22-25, 1991.

[Kato 91] Kato, Toshikazu; Kurita, Takio; Shimogaki, Hiroyuki; Mizutori, Tetsuya; & Fujimura, Koreaki. “Cognitive View Mechanism for Multime-

dia Database System," pp. 179-186. *Proceedings of the First International Workshop on Interoperability in Multidatabase Systems (Kyoto, Japan).* Los Alamitos, CA: IEEE Computer Society Press, April 7-9, 1991.

[Keim 91] Keim, Daniel A.; Kim, Kyung-Chang; & Lum, Vincent. "A Friendly and Intelligent Approach to Data Retrieval in a Multimedia DBMS," pp. 102-111. *Proceedings of the DEXA 91 International Conference (Berlin, Germany).* Wien, Austria: Springer-Verlag, August 21-23, 1991.

[Kolodner 92a] Kolodner, Janet L. "An Introduction to Case-Based Reasoning." *Artificial Intelligence Review 6*, 1 (1992): 3-34.

[Kolodner 92b] Kolodner, Janet; & Mark, William. "Guest Editors' Introduction: Case-Based Reasoning." *IEEE Expert 7*, 5 (October 1992): 5-6.

[Loeb 92] Loeb, Shoshana. "Architecting Personalized Delivery of Multimedia Information." *Communications of the ACM 35*, 12 (December 1992): 39-47.

[Lu 93] Lu, Cary. "The Full-Motion Macintosh." *MacWorld 10* (January 1993): 140-147.

[Mackay 89] Mackay, Wendy E.; Malone, Thomas W.; Crowston, Kevin; Rao, Ramana; Rosenblitt, David; & Card, Stuart K. "How Do Experienced Information Lens Users Use Rules?", pp. 211-216. *Proceedings of the CHI '89 Conference on Human Factors in Computing Systems (Austin, TX)*. New York: ACM, April 30-May 4, 1989.

[Malone 92] Malone, Thomas W.; Lai, Kum-Yew; & Fry, Christopher. "Experiments with Oval: A Radically Tailorable Tool for Cooperative Work," pp. 289-297. *Proceedings of the CSCW '92 Conference on Computer-Supported Cooperative Work ((Toronto, Canada).* New York: ACM, October 31-November 4, 1992.

[Oren 90] Oren, T.; Salomon, G.; Kreitman, K.; & Don, A. "Guides: Characterizing the Interface," pp. 367-381. *The Art of Human-Computer Interface Design* (B. Laurel, ed.). Reading, MA: Addison-Wesley, 1990.

[Robertson 91] Robertson, George G.; Mackinlay, Jock D.; & Card, Stuart K. "Cone Trees: Animated 3D Visualizations of Hierarchical Information," pp. 189-194. *Proceedings of the CHI '91 Conference on Human Factors in Computing Systems (New Orleans, LA).* New York: ACM, April 28-May 2, 1991.

[Salomon 89] Salomon, Gitta; Oren, Tim; & Kreitman, Kristee. “Using Guides to Explore Multimedia Databases,” pp. 3-12. *Proceedings of the Twenty-Second Annual Hawaii International Conference on System Sciences Vol. IV: Emerging Technologies and Applications Track*. Washington, DC: IEEE Computer Society Press, January 3-6, 1989.

[Salton 83] Salton, G.; & McGill, M.J. *Introduction to Modern Information Retrieval.* New York: McGraw-Hill, 1983.

[Schank 91] Schank, Roger C. *Case-Based Teaching: Four Experiences in Educational Software Design* (Technical Report #7). Evanston, IL: Northwestern University Institute for the Learning Sciences, January, 1991.

[Shneiderman 93] Shneiderman, B. “Beyond Intelligent Machines: Just Do It!” ***I****EEE Software 10*, 1 (January 1993): 100-103.

[Silberschatz 91] Silberschatz, Avi; Stonebraker, Michael; & Ullman, Jeff. “Database Systems: Achievements and Opportunities.” *Communications of the ACM 34*, 10 (October 1991): 110-120.

[Smith 91] Smith, Thomas G. Aguierre; & Pincever, Natalio C. “Parsing Movies in Context,” pp. 157-168. *Proceedings of the Summer 1991 USENIX Conference (Nashville, TN)*. Berkeley, CA: USENIX Association, June 10-14, 1991.

[Stevens 89] Stevens, Scott M. “Intelligent Interactive Video Simulation of a Code Inspection.” *Communications of the ACM 32*, 7 (July 1989): 832-843.

[Stevens 92] Stevens, Scott M. “Next Generation Network and Operating System Requirements for Continuous Time Media,” 197-208. *Network and Operating System Support for Digital Audio and Video*. New York: Springer-Verlag, 1992.

[Wood 92] Wood, David P.; & Kang, Kyo C., *A Classification and Bibliography of Software Prototyping* (CMU/SEI-92-TR-13, ADA258466). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, October, 1992.

Apple and Macintosh are registered trademarks of Apple Computer, Inc. QuickTime is a trademark of Apple Computer, Inc. DVI is a registered trademark of Intel Corp. All other trademarks are property of their respective owners.