José Encarnação, James Foley (editors):

Multimedia - System Architectures and Applications

Dagstuhl-Seminar-Report; 51
02.11.-04.11.92 (9245)

ISSN 0940-1121

Tel.: +49-6871 - 2458
Fax: +49-6871 - 5942

Das Internationale Begegnungs- und Forschungszentrum für Informatik (IBFI) ist eine gemeinnützige GmbH. Sie veranstaltet regelmäßig wissenschaftliche Seminare, welche nach Antrag der Tagungsleiter und Begutachtung durch das wissenschaftliche Direktorium mit persönlich eingeladenen Gästen durchgeführt werden.

Bezugsadresse: Geschäftsstelle Schloß Dagstuhl
Informatik, Bau 36
Universität des Saarlandes
Postfach 1150
66041 Saarbrücken
Germany
Tel.: +49 -681 - 302 - 4396
Fax: +49 -681 - 302 - 4397

MULTIMEDIA

System Architectures and Applications

Dagstuhl, Germany

November 2-4, 1992

Introduction

The Workshop on "Multimedia--System Architectures and Applications" took place in Schloß Dagstuhl, Germany, from November 2 to 4, 1992. There were 26 participants coming from 7 countries. The Workshop was based on presentations (their summaries are the contents of this small brochure) and three Working Groups. The reports of these Groups are also published here. The participants decided at the end of the Workshop that they would welcome a "follow-up workshop" in Dagstuhl in 1994. This is now being considered.

As Co-Chairs of the Workshop, we would like to thank Professor Wilhelm and all the IBFI staff in Dagstuhl for the excellent and extremely fruitful working atmosphere which was the basis for the success of the Workshop. We thank also all participants for their contributions and for all the very constructive discussions.

Darmstadt, Atlanta
March 1993

Professor José Encarnação
Professor James Foley

Using Conceptual Models in Hypermedia

Jerker Andersson
Infologics, Sollentuna, Sweden

Interactive multimedia systems and hypermedia systems are hampered by the overview problem. To help the user navigate a hyperspace, several tools have been proposed, examples are maps, history list, annotation, etc. We propose to use conceptual models as maps for navigation.

By extracting some of the semantic information and labeling each hyperspace node to match a concept and making the links match relations, we can make a 'conceptual map.' When selecting a node through any navigation tool, the corresponding concept in the map tool highlights. When following a link in the information, the distance of the jump can be assessed using the conceptual map. Also when clicking at a concept, a list of available nodes pops up for the user to select from.

Conclusion include: Users and experts disagree, the model requires consensus. It is difficult to handle such a dense view of all information. It improves on the manageability of the system, and it provides a generic map tool.

Logical Markup for Hypermedia Documents: The TRAIN System

Friedrich Augenstein, Thomas Ottmann, Jürgen Schöning
Institute for Computer Science, University of Freiburg, Freiburg, Germany

We report about the design and the implementation of a new authoring system for multimedia documents used for computer-aided instruction. The system allows a clear distinction between the tasks and the responsibility of the author, who is responsible for the content, a professional design expert and a pedagogue who is responsible for the underlying didactic principles. Our goal is to separate between the logical and the layout structure of multimedia documents, and to raise the editing process from the direct manipulation of simple primitives to a type-directed use of high-level logical commands. Expert knowledge is laid down in document type definitions (DTD) which determine the hierarchy of structured multimedia documents with logical markup, and style files which determine the presentation of these documents.

Word and Image in Multimedia

Jay Bolter, Kenneth Knoespel

School of Literature, Communication, & Culture, Georgia Institute of Technology, Atlanta, USA

We looked at several issues involving the relation of word and image in multimedia. After making brief comments about what we have called the "Rhetoric of Multimedia," I considered several historical examples regarding the development of graphics and the transformation of text in book production and printing technology. Jay Bolter then considered some of the ways text might be used in 3D books in the future.

In the School of Literature, Communication, & Culture at Georgia Tech, we are working on the development of a rhetoric of multimedia. Several questions accompany our work: 1) Can we identify a useful integrative vocabulary to define the relation or configuration of different media? 2) To what extent can we draw on traditional or classical rhetorical theory in this enterprise? 3) To what degree may such a rhetoric be informed by linguistic and semiotic theory? We would emphasize the important setting that work in multimedia creates the theory. Theory will help us create better applications and allow us to understand how the issues brought before us by electronic technologies can be sharpened by revisiting hypermedia issues within the history of communication.

The question of textural topography has been addressed repeatedly within the history of writing and reading. Text has continually been written over other texts. It is not an exaggeration to say that the capacity to write on or within texts becomes an important (and often overlooked) element of scientific change. For example, the commentaries on classical texts often become authorities themselves. These remarks were followed by a presentation and discussion of illustrations from three texts from the 16th and 17th centuries.

The texts discussed were:

- Filippo Calandri's Aritmetica (1491)
- Agostino Romelli's Diversi et artificiosi machine (1588)
- Michael Maier's Atalanta Fugieni (1618)

The illustrations show how graphic representation alters or transforms text. Such historical examples also draw attention to the status of orality in the transmission of information and notice how information becomes encoded on different levels. They also remind us that whether we look at examples of hypermedia and multi-media from the past or present we find ourselves faced with an array of sites for the articulation of a rhetorical theory.

Building (Multimedia) User Interface by Direct Composition

Martin Brenner

Corporate Research and Development, Siemens AG, Munich, Germany

When designing and building our user interface management system (UIMS) "SX/Tools", we found and implemented some principles, which will be vital for multimedia user interfaces, too:

Two of the key principles are 'Direct Composition' (1) and object orientation (2).

(1) User interfaces are built by composing UI objects out of a tool-base to a new user interface.

(2) Each object contains all functionality needed for the design process.

So, the multimedia user of the future can be empowered to use the design functionality for his own work with multimedia data.

Integration of Motion Video and Computers

Bernd Girod

Kunsthochschule für Medien, Cologne, Germany

In this talk, I briefly review the state-of-the-art in multimedia video compression. Then, I discuss the performance and suitability of the evolving ISO MPEG standard for applications foreseen. The integration of computers and motion video poses special requirements beyond highly efficient data compression. These are in the areas of scalability, support for interactive video, and editing capabilities. Scalability is desirable in three areas:

- image size scalability
- partial decodability of bit-stream
- computation-limited coding/decoding

Scalability features 2. and 3. are mandatory to realize software-only codecs which will become feasible for workstations in the second half of this decade. Interactive desktop video in addition requires "VCR features" (shuttle search, freeze frame, etc.), random access to individual frames, and short decoding delay. Unfortunately, short decoding delay and low bit-rate compression are conflicting requirements. In conclusion, the grand challenge to find a motion video representation that satisfies all the multimedia requirements remains!

Fulfilling the Requirements of Multimedia Communication

Ralf Guido Herrtwich

IBM European Networking Center, Heidelberg, Germany

The Heidelberg Transport System (HeiTS) uses the internet ST-II protocol as its communication core. ST-II provides adequate functions for multimedia communication, most notably fast packet forwarding, multiple target addressing, and bandwidth management. Some desirable additions to the protocol include graceful service degradation and feedback for rate adjustment in the nodes. For video, these functions allow dynamic scaling in the temporal and/or spacial domain of a stream. A transport protocol on top of ST-II should provide different reliability classes, as some multimedia encoding schemes are less robust to errors than others.

Fine(-grained) Synchronisation in Dynamic Documents

Wolfgang Herzner
Austrian Research Centre Seibersdorf, Seibersdorf, Austria

When adding 'dynamic' or 'continuous' contents to documents, authors need to specify temporal dependencies among the components of a document, while users have a need for interaction with the presentation of such documents. At least for selection and navigation, but also for controlling speed (of videos) or volume (of sound) or alike, and even for more complex kinds of interaction, if application areas like computer-based training and testing, entertainment, and even simulation are taken into consideration.

Therefore, a synchronisation model is presented, which allows to specify complex temporal and spatial-temporal relationships, to consider a wide range of user-interactions, as well as hard/software-related delays (e.g., transmission of remotely stored content).

This approach regards documents (or applications, resp.) as acyclic, directed graphs of content objects. For these objects, so-called presentation units (p-units) can be defined by the author, which describe their behaviour at presentation time. P-units interact with each other by sending messages and emitting signals when certain operations are completed. Several classes of p-units model different default behaviour (e.g., 'loop'), while conditions like 'stop object A two seconds after both B and C have been started' can be specified for describing additional conditions. Furthermore, p-units possess attributes controlling characteristics like size, default playing speed, or even logical target display/device.

Finally, the user's interaction with a document/application is modelled by means of input objects. Several classes of input objects provide for different kinds of input (e.g., button, slider, object pick), while different modes allow to gather input values in different ways, e.g., on explicit user request, or by sampling.

A prototype implementation is currently carried out at Seibersdorf.

The Modelling of Images for Communication: From the Image Signal to the Image Document

Georg Rainer Hofmann
Fraunhofer Institute for Computer Graphics, Darmstadt, Germany

The field of "image communication" is evolving from three different areas: telecommunication engineering, electronical engineering, and computer science.

In this situation, where different disciplines of sciences intersect, the terms (a.o.)

> image model, image format, image signal, image data structure, video/moving picture, image document, images for multimedia

are used in different contexts and contradictory meanings, even ambiguous ways.

It has been shown that the variety of image types can be arranged in a 'pipeline' of eight stages:

Oriented towards realtime display and communications:

(A) optical image signal
(B) electronical analogous image signal
(C) digital image signal
(D) raw image data bit stream
(E) pixel array (motion picture)

Oriented towards document structures:

(E') pixel array (still picture)
(F) generic image data format
(G) image contents for document architectures

(H) image within interpersonnel messages

Image rendition processes must--at least partially--cover the different stages (H) through (A), until an optical (visible) presentation of the image data has been reached.

Advanced User Interfaces for Multimedia Applications

Wolfgang Hübner

Computer Graphics Center, Darmstadt, Germany

Multimedia increases the bandwidth of communication between computers and users. Multimedia output technology like 3D graphics, video, audio exploit human perceptual power. Multimedia input technology allows multidimensional input in several channels in parallel. But the widely used interaction paradigms like desk-top metaphors or windowing are not capable to integrate advanced user interfaces into multimedia applications properly, because they are based on 2D raster devices and mouse-oriented pointing devices. Multimedia and 3D graphics have led to the need of developing a new generation of interaction techniques and advanced user interface paradigms based on the new media available.

The underlying platform to develop systematically advanced user interfaces for multimedia applications is a classification of the levels of interaction that was derived from traditional 2D interaction being mapped onto multimedia interaction.

Integrated Digital Video Architecture

Takahiko Kamae

NTT Human Interface Labs, Kanagawa, Japan

My talk consists of three parts: broadband ISDN, network casting through B-ISDN, and the future integrated digital video architecture.

B-ISDN has three important keywords: multimedia capabilities based on ATM, fiber-to-the-home, and the infrastructure in the 21st century. Fiber-to-the-home will make sufficient bitrates available even to the private sector; sufficient enough to transmit a high-definition TV signal. Network casting takes advantage of this abundant bitrates. Various video programs and multimedia information can be distributed through B-ISDN. Network casting will become as important as broadcasting to distribute information to the general public.

The integration of broadcast video, telecommunication video, and packaged video will be a key issue. I propose the integrated digital video architecture. The integrated digital video is fully based on digital technologies including receiving terminals and videocameras.

Database Management for Multimedia Applications

Klaus Meyer-Wegener

Friedrich-Alexander-Universität, Erlangen-Nürnberg, Germany

New kinds of input devices (scanners, cameras, sound boards, etc.) produce new types of data objects. These must often be held on special storage devices due to their size, e.g., on optical storage or on VCRs. Later, they are used to feed certain kinds of output devices (raster displays, sound boards, video boards).

The volume of the data will increase significantly in the future and, thus, their management will reach a new quality. DBMS technology can be extended and then applied to this task. However, it should be realized that DBMS are built to provide storage and retrieval, not more and not less. Neither complex processing (e.g., image analysis) nor lengthy editing should be done inside a DBMS. Instead, analysis software and editors should cooperate with the DBMS, i.e., obtain data from it and file their results to it.

Even though the task of a DBMS is restricted to storage and retrieval, providing this for multimedia data bears some challenges. Device and format independence, representation of relationships, and content addressability must be offered. This is not sufficiently done by current systems and proposals. A number of research issues can be derived from that.

A Modeling/Programming Framework for Large Media-Integrated Applications

Max Mühlhäuser
University of Karlsruhe, Karlsruhe, Germany

Multimedia base technology evolves rapidly. Most multimedia (MM) applications, however, are rather self-contained off-the-shelf tools for specific tasks. This is in contrast to major trends in SW technology:

- organizations try to INTEGRATE their application SW (CIM, office automation, 'integrated enterprise')
- organizations want the application SW to be CUSTOMIZED to their environment, not vice versa.

Therefore, the key message here is a request to move from isolated 'niche' MM applications to MEDIA-INTEGRATED applications, especially in the context of large cooperative 'enterprise-integrating' SW. However, current SW technology (in particular, software engineering support) is very inappropriate for such large media-integrated applications. The request for more appropriate SW technology breaks down to three basic requirements:

1. Sophisticated base services: In terms of 'traditional SW', this should include distributed long-living threads ('workflows'), object migration, and a complete computational model. In terms of MM, it should include a generic MM object model, network-transparent MM object handling, generic support for CSCW objects ('team objects'), their handling and control in a cooperative scenario, and generic support for hypermedia navigation.

2. Encompassing model and framework: We advocate the "object category" approach here: Major "kinds of objects" (media objects, team objects, workflow objects, 'small message' objects, etc., see item 1. above) are made available as pre-programmed objects and templates, and can be integrated in a large media-integrated application by combining

a visual programming (click-select, instanciate, customize; cf. to interface builders) and a traditional (alphanumerical) programming paradigm, so that ONE SW engineering framework is used for all (conventional/CSCW/MM) aspects during resign AND programming.

3. Higher levels of abstraction: in order to protect the investment in SW development when moving to different environments, media used, and policies (about security, auditing,...), abstractions of different kinds are of great importance, such as:

   - target network abstraction (e.g., based on the distributed object-oriented approach);
   - roles which abstract from individual users, base objects, etc;
   - HCI abstractions which hide the paradigm and media used in the UI;
   - policy abstractions, used to decouple operational aspects (such as auditing, fault tolerance, etc.) from the 'mainstream' SW;
   - interaction abstraction (details of multiparty communication).

Speeding up the Process of Multimedia Application Development

Meinrad Niemöller, Ulrike Harke
Corporate Research and Development, Siemens AG, Munich, Germany

The target market has a strong impact on how multimedia applications are built and also on which tools are used. Applications in professional environments call for a close integration of multimedia with traditional data processing. Our approach is to extend a powerful user interface design system called SX/Tools towards a multimedia application development environment. We have built an exemplary multimedia application called MOSeS--Multimedia Option Selector for Car Sales Support. The goals of the project MOSeS are to show the potential of an integration of conventional data processing and multimedia, and also to help to identify the bottlenecks in the process of application building. Based on these experiences we discuss current activities in our labs to improve and to extend SX/Tools to speed up the production of multimedia applications.

Practical Issues in Developing Hypermedia Systems for Training and Education

G. Scott Owen
Hypermedia and Visualization Laboratory, Georgia State University, Atlanta, USA

The development of hypermedia systems for training and education exhibits many similarities to general software development projects but with some extra problems and issues. We at the HVL have had several externally funded projects to develop multimedia systems for training or education. There are a number of real world issues to consider. These include the cost of the delivery system, especially for educational programs. One must also consider and plan for the rapid technology changes that are constantly occuring.

The Next Generation of Distributed Multimedia System Architectures

Ralf Steinmetz
IBM European Networking Center, Heidelberg, Germany

Distributed multimedia system structures evolved from "hybrid" to "unified" approaches. "Unified" systems allow for full digital data processing of the various kinds of media.

Today's unified systems are built on various platforms according to the respective operating and windowing system paradigms.

The next generation will comprise the same interaction method and application user interface on various platforms. In the HeiProjects at the ENC, we architected a cross-platform structure which has already been implemented in a first version. It allows for an easier development of networked applications due to the re-use of software design and code. The implementation covers AIX- and OS/2-based systems.

Cooperative Multimedia on Heterogeneous Platforms or: How to Apply Telemedia

Bernhard Tritsch, Christoph Hornung
Fraunhofer Institute for Computer Graphics, Darmstadt, Germany

The development of a distributed and heterogeneous generic multimedia platform presents many issues. The hardware, software, and network platforms are heterogeneous, thus, device-independence and interoperability play significant roles there. The primary objective of the presented research effort is to introduce and to specify a generic architecture, including communication mechanisms and multimedia features. This architecture provides functionalities in the areas of transfer, processing, storage, presentation, and interaction. Moreover, cooperation and timing control paradigms must be introduced, allowing realtime communication, cooperative work, and remote access. This is described by the term telemedia (telecommunicating multiple media). Thus, this framework integrates support for cooperative work on multimedia documents and multimedia group conferencing features.

Group I Report: Authoring Systems

Group I Members: G. Scott Owen (Chair), Victoria Burrill (Rapporteur), Jerker Andersson, Friedrich Augenstein, Jay Bolter, Wolfgang Hübner, Wolfgang Herzner, Kenneth Knoespel, Kaisa Väänänen, and Frank Weller

Introduction

The objective of this group was to discuss the User's and Author's views of multimedia (MM). Authors are the Information Providers, responsible for the content, creation and maintenance of the information base. Authors use the tools provided by the developers. Sometimes the author is the user, and the user the author, i.e., in some applications, the author and user are the same person; in others they are different. The author is the person developing the MM applications (NOT the person developing the toolkit). The user *uses* the MM system.

There are different types of author: (1) Producers generate the text, pictures, video etc; (2) Integrators define the underlying hyper structure. These two types of author may be one and the same person, or they may be different. A third type of author is the Trail Blazer who does not change the data but adds to it. But these are all very fuzzy and blurred categories.

The first thing we did was to restructure the set of keywords into the following categories (in priority order):

- Application areas
- User interface
- Information structure and change
- Authoring systems

In each major category we prioritized the set of keywords, added and deleted them as we felt necessary, and then discussed them. We left Authoring Systems to the last not because this is the least important topic but because the discussion of the first three topics would mandate the requirements for the Authoring Systems.

Application Areas

The first topic discussed was Application areas. We did not discuss these extensively but listed and categorized them by the role of author versus user and other relevant issues, e.g., systems for computer literates versus "walk up and use" naive users. In some systems there is overlap between the roles of author and user in that an author might create the original system and the users might be able to modify the system, either by adding annotations or by changing the web of hyperlinks. In other systems this would not be true.

Are there really any application domains? There are lots of MM systems but who really uses them? Does this depend solely on who has the money to spend on them? There should be a huge market. Maybe people don't really understand MM? Once they see one good application to which they can personally relate and/or use then they'll believe in it. The problem is finding that application. What do we think the best/most important MM applications are?

The first category was "Information Providers," which are systems that provide general types of information, as opposed to systems, e.g., training systems, which are focused on a narrow topic. The first example of this type of system is a Kiosk. We defined a Kiosk as a simple information provider with a very easy to use interface and extremely simple selection interaction method, though not necessarily of a simple internal information structure. These are oriented towards use by the general public, i.e., a "walk up and use" type system. A very simple example might be a museum kiosk and more complex examples would be a system telling tourists how to get around a city (as in ShareME) or college students how to get around a university campus. In a kiosk the author is distinct from the user, i.e., the user does not modify the system in any way.

A kiosk system is one where (a) the user and author are totally separate; (b) has a relatively simple interface (that is probably physically easy to use); (c) is an information provider for the general public (this is a sweeping statement); (d) a system for which the answer to "What is this system for?" should be blatantly obvious; (e) the information is not modifiable by the users (but it may not necessarily be a static structure).

The word "kiosk" is a general term, subcategorized by tourist information systems, encyclopedias etc. There was some discussion of cultural differences over this word; in the USA a kiosk is a system that dispenses information in a public place, with very limited functionality and very limited purpose; in Germany a kiosk is thought of as being a more general information provider.

A more complex information provider system would be an encyclopedia with a large set of knowledge and thus, a more complex hyperspace. In an encyclopedia an author creates the system and users may annotate and/or modify the web of links. This could include both the addition and deletion of links.

Other application areas include the following:

- Training
- Education
- Entertainment
- On-line electronic manuals

The first three areas require more complex information such as time dependence; interaction is less generic. For these systems an author creates the system and users may or may not modify the system depending upon the application.

Another application area is in Multimedia Monitoring, Control, and Decision Support Systems, such as a medical database or industrial plant monitoring. These systems are categorized by being very dynamic and having realtime changes. For decision support systems the users typically put in the structure and information for the system. For process monitoring and control systems an author will create the view and the information may be automatically updated for the user (control room operator).

Another application is Multimedia Computer Supported Cooperative Work (CSCW) with issues of communication and information content and sharing. A final category is Multimedia Simulation.

User Interface Issues

Under User Interface we grouped the topics (in priority order):

1) User Interface (UI) metaphors
2) Navigation
3) Dimensionality (2D and 3D, VR)

4) Interactive methods
5) Personalization

We then discussed each of the above topics.

User Interface (UI) Metaphors

A metaphor is the representation of one type of object by an analogy with another type of object. In our usage it is the representation of some aspect of the user interface by an analogy to a familiar object. Metaphors can be subclassed into physical, social, abstract and others. Several types of UI metaphors were listed, the first of which was Visual Representation. Next was Spatial Representation. We decided that there were two different types of Spatial Representation, the first were metaphors which used physical or realworld things such as a desktop or trash can. The second type was abstract, for example, a graph or a tree. There were also control metaphors. An example of this would be a button for opening a text window for scrolling. Another example would be a representation of a VCR or cassette player for video or animation and audio.

Metaphors cost something. If you use one you are "buying into" the metaphor it represents together with lots of assumptions and implications about it. There are connotations as well as denotations. Metaphors encapsulate a story. When you adopt a metaphor you take the good parts as well as the problems so that certain expectations may be associated with the metaphor that the system might not meet.

If you are building a MM system, what metaphors do you need? Some authoring systems use a score (temporal layout), others a stage and cast. Should author and user metaphors be the same? For example the Microsoft-AVI system gives the author the choice of whether or not to give control to the user, i.e., the user can be presented with just a video clip and no control or the full AVI control window.

How ubiquitous is the VCR control metaphor? It also applies to audio cassettes. But what happens when DAT tapes come along? What is the metaphor for accessing the tape frame by frame? Text is editable, but video is not--this is a crucial difference. Without semantics you cannot edit video. With current metaphors for video there is no easy way to allow the user to access a frame by content. The ZGDV/RAL MOVie project is addressing this problem.

There are several traditional disciplines upon which we should draw in designing multimedia interfaces. These include: architecture, graphic design, typography, dramatic arts and scene design, industrial design, etc.

Navigation

The next topic was Navigation. The questions here were "Where am I?", "How do I get where I want to go?", "Where have I been?" (backtracking), strategies and exploration. We decided to look at tools for navigation and then strategies. The parameters of Navigation issues include: size of the information space, absolute and relative indications of space and of elements, and subsets of modes, i.e., different views.

We came up with a large set of possible navigation tools including the following:

- Conceptual Map
- Guided Tours
- History lists or maps
- Keyword search (content search, incorporating Booleans)
- Structured search
- Index
- Contents
- Hyperlinks
- Random tool
- (AI) Agents to find something interesting (active or passive)
- Human expert
- Audio/video/animation hyperlinks (annotation)
- Video/animation search tool (index, storyboard)

A question was what new tools does MM require, if any? How do we handle the temporal aspects such as reaction times for various events. Can we track recurrent themes? Can we use the dynamic characteristics of the data within the tool itself or are we forced to map to a spatial representation in order to control it? For example, deaf people run video tapes at extra speed--this works on a VCR so it should in MM too if we are using the VCR metaphor.

Along with the discussion on tools was a discussion of possible strategies which included the following:

- Plan a path to a specific goal or neighborhood
- Get an overview (detailed or superficial, i.e., at or greater than the granularity of the information)
- Exploration
- Ask for help
- Serendipity

The first tool discussed was a Conceptual Map, which gives an overview of the hyperspace as either a textual view or a spatial representation. As a spatial representation this could be a 2 or 3D graph or tree, scrolling walls, rotating 3D trees, or a fish-eye view. The fish-eye view shows the neighborhood of the user in great detail, but areas farther away in less detail. The fish-eye view can be textual or it could be a spatial view of the conceptual map. For a fish eye view the relative "distance" between nodes must be known and so the concept of "distance" in the hyperspace must be defined in some manner.

Maps can be active navigation devices, or they can be passive aids to orientation. An active map would allow the user to immediately jump to any node by pointing to that node, i.e., it would be able to dynamically create a link between the present node and the desired node. This link could be temporary or permanent. A passive map would show the user a path, if one existed, from the present node to the desired node but would be incapable of creating any new links. There could also be a temporal map, for example: audio or music.

Another tool is a Guided Tour where the user follows a predetermined path. There might be several such guided tours, one for each set of users. This is one way to customize the system for different users. The user might be restricted to the Guided Tour or they might be able to leave it at any point to go on a side trip. Note that a restricted Guided Tour reduces hyperspace to a linear space.

Another tool is a History list (list of previously visited nodes) or map (the same information in map form). This allows the user to immediately return to any previously visited node. However, this list might grow very large so an alternative or companion tool would be Bookmarks which the user could use to mark a visited node for returning later. The Bookmark list would be much shorter than the History list.

There are several tools that can be used for queries, i.e., trying to find specific information. An Index or Table of Contents could be used for a topic search and a Keyword search, with Boolean operations, would be a content search. There could also be a structured search, for example "Show me nodes with certain characteristics." Hyperlinks themselves could be considered as tools to navigate Multimedia space. Another tool would be a Random Tool which would be "Take me anywhere but here." There might be a human expert available, via a network, such as in the ShareME system.

We also considered Artificial Intelligence-type agents. These might allow the user to say "Take me somewhere interesting." The user could possibly define "interesting" or the agent could have observed the user and take them to a place similar to previously visited nodes. Another tool would be audio/video/animation hyperlinks and/or annotations such as MOVie. There might also be a video/animation search tool which might be an index or storyboard.

Strategies

We discussed strategies that could be used to plan a path to a specific node or to a neighborhood or else to just browse an area. We divided strategies into two categories: those which aimed at a specific goal and those in which the goal was non-specific or fuzzy. Some tools may be useful for one strategy; some for the other. We created a matrix of tools against strategies, which is shown below.

Tool\Strategy	Specific goal	Fuzzy goal	Exploration
Active Map	X	X	X
Passive Map	X	X	X
Guided Tour	--	X	X
History Lists	X	--	--
History Maps	X	X	?
TOC	--	X	--
Index	X	--	--
Content Search	X	X	--
Structure Search	?	X	--

Tool\Strategy	Specific goal	Fuzzy goal	Exploration
Hyperlinks	X	X	X
Random	--	--	X
Agents	X	X	X

Specific goals = "I want this information" (a specific query)
Fuzzy goals = "I want something about..." (everything about a subject)
- increasing desire for specificity -
Exploration = "What is here?" (general enquiry)

X = The navigation tool is useful in attaining this goal.
-- = The navigation tool is not useful in attaining this goal.
? = The navigation tool may or may not be useful in attaining this goal.

Other strategic possibilities include: ask for help, general overview, serendipity, and exploration. We also noted that there may be system (automatic) strategies as well as user's strategies. For example, the system can guide the user toward a specific goal.

Dimensionality Issues

There are at least four categories of dimensional representation for the user interface: 2D, projected 3D, true, stereoscopic 3D, and Virtual Reality. Each of these may be useful for some applications. It is a research question whether and when 3D interfaces are useful. Also we noted that one can use a 3D metaphor without true 3D display.

Interactive Methods

Here we noted two issues that are especially relevant for multimedia interfaces: temporal issues (dealing with the dynamic character of the information) and 3D representations. Again, 3D input devices (e.g., the dataglove) are not necessarily required for 3D representations.

Personalization

In general, personalization methods may includes adding or deleting links, creating different views, macros, markings, including bookmarks, and annotations. Issues of particular importance to multimedia include speed of interaction and mapping among media. This latter is important, for example, for users with disabilities. Video is of no use to a blind user, so we might try to remap video information to the audio channel. Multimedia applications can also be personalized according to the different skills, interests, or knowledge level of the user.

There are new dimensions of time; the user may want to specify a preferred medium for reasons of personal preference or according to equipment available. This is especially important for visually challenged users. The mapping of one medium to another (or one medium to a different form of itself) is another interesting research area.

What forms of personalization are there?

(1) By content to reflect a person's interests, e.g., sport.
(2) By abilities to present the same information via different media according to personal skills and/or the equipment available.
(3) By knowledge according to whether the user is a beginner or expert.

Information Structures

The integration of different media and/or importing devices defines a central purpose of information structures. Information structures create hierarchies that determine the levels that an author will be able to access and thus, place limits on the author. Such control determines the flexibility the author has of working within the system. Information structures also affect the way the information is presented to the user. Information structures are shaped not only by the media to be integrated but also by the psychosocial patterns that influence the way information is structured. The issues in information structures include the following:

- Granularity
- Restructuring of information
- Metainformation

Logical data models

Logical data models are paradigms or significant structures that determine the capabilities and possibilities that the author has in structuring information. Several categories of logical data models can be identified:

- Nodes and links
- Object Oriented structures
- Relational models
- Ordered sets
- Stacks (as in Hypercard--an example of an ordered set)
- Flow Charts which indicate progression plus synchronization (an authoring metaphor but not really a data model)

The first four categories were felt to be "serious" structure models whereas the last two were "toys." For example, while Hypercard itself can handle huge stacks the question is whether an author can control a large amount of information structured only by stacks.

One thing that all these models have in common is that they are discrete re-presentations based on units of information and that they make use of methods of navigation and selection. There are various ways of defining nodes and links. Logical data models are comprised of a basic structural paradigm and well-defined information units. Information units can either be simple or compound. It is also important to recognize that information units can be realtime or continuous feeds.

Granularity

Granularity appears on different levels. While it may appear in hyperspace (granularity of information space) it is also a factor within one medium (granularity of medium). Granularity of medium has significant ramifications for the integration of different data types. The author potentially has access to any granularity, provided that it is supported by the authoring tools. The author may wish to limit the options available to the user. The granularities offered to the user determine (or, conversely, are limited by) the storage formats and realtime performance.

Restructuring of Information

Restructuring allows the user to modify the information space at hand. The question arises: Is it necessary to have access to the explicit structure in order for it to be modified? Annotation (making comments on a node or link) may, or may not, be an act of restructuring based on the logical model functionality. It may be an overlay or add-on or change the actual node and link structure. Annotations should be "hidden" from other users. Adding and deleting links assumes that the links are embedded in the information. Individual users may have their own webs. Annotations *may* be like a restructuring, but not necessarily.

The following questions arise:

1. What changes can readers make to the original information?

2. What is restructuring? Adding one node? Deleting one node? Adding an annotation? Can a user do these? Permanently? Probably "yes" to all but the last question.

Where is the boundary between adding and modifying the information? Providing that you can get back to the original then it is an add operation; if you lose the original then it is a restructuring. Restructuring might be defined as adding or deleting links. These are very low level operations. High level operations include adding links of a specific type to all nodes (such as adding help). So restructuring is determined by the underlying data types.

Both authors and users may add links, but this depends on the model. You may want to allow user modifications of the form "This link added by xyz." The user may be restricted in restructuring; the author should be able to do the major restructuring but what facilities does this involve? There are implications on the author who restructures since this may mean that user's links become invalid. Restructuring makes the user become a co-author. What constitutes a restructuring is a research issue.

Metainformation

Metainformation is essential for some navigation tools and for complex restructuring operations. For example, the HyperCard model is implicit, so it is difficult to do large-scale restructuring. The storage of metainformation may be internal or external to the information space. We would notice that questions involving the storage of metainformation were not addressed. It is either associated with the information units and link structure or may be completely external to the information space. Metainformation may also be shaped by metaphor and may contain logical implications embedded within the discourse structures suggested by the metaphorical associations.

Authoring Systems

Editing in different media currently may involve several editors, translations, then final integration together. But this is a failing of current MM authoring systems which provide very few integration facilities.

Input and transformation of different media types should be transparent to the user. At the moment, different media types are integrated together via different tools, i.e., editors are medium-specific. We need a more document-oriented environment. It is more convenient for the author to have everything incorporated in the authoring tool. It is of course possible to prepare items outside the authoring system and then import them, but integration is better.

An authoring system is an environment with different tools for different tasks. Sometimes the author has to generate information, import it, modify it, restructure it, provide references to it (especially in a power station or other realtime application where data is constantly being generated), design/decide on the user interface, and then finally integrate it all (by setting up nodes and composing them). This process is subdivided by data types, and further complicated if the application is distributed. SGML can be used to define logical levels of information. Within composition, we also need a metaphor for successful integration.

MM Document Design

What guidelines should the author use? Guidelines are needed that don't overload any one human sense channel. These guidelines are for structuring information together. Should these guidelines be provided by the system? Written guidelines are often ignored. Some systems impose guidelines and accept that the resulting system is not 100% flexible, but this may be a necessary trade-off.

Guidance versus Flexibility

The system should monitor itself and warn the author if the guidelines are being ignored. An example would be grammar checkers and design advisors. The system could ask the author if they really want to do this. The system has *suggested* guidelines but not rigid rules.

A major question is what is good MM document design? Paper documents have standards developed over 400 years whereas MM standards are very young, and are still being developed. This is another good research area.

What language should the guidelines be in? This depends on the metaphor of the application, but how do we describe it? Would "Design Templates" be a better description of these guidelines?

The first thing we did was to define the authoring tasks as follows:

(1) Acquisition and generation of information:
- create information
- import information (file in, static images)
- reference to external information, dynamic
- edit information

(2) Choice of design metaphor

(3) Structuring of information:
- integrate information, e.g., set up nodes
- structurize information
- produce user interface, e.g., layout, colors, etc.

(4) Information and Structure Maintenance and Enhancement:
- update
- correct/repair
- ensure consistency
- restructuring

Several current problems were identified including the following:

(1) Authors use many different tools for completing the task.

(2) Authors tend to do bad document and application design.

(3) The quality or level of editing tools varies for different content classes, e.g., high for text and graphics, lower for images, and even lower for video and audio. Whereas text editors have had several generations for refinement, tools for the other media types are in their first or second generation.

(4) Few tools are available for integrating the different forms of information, e.g., the editing of different classes such as subtitling a film. This is a corollary of point (1). We also need a suitable format for accessing and synchronizing the media--this is a corollary of point (3).

(5) Current tools do not provide sufficient support for spatial or temporal dependencies, e.g., concurrency and synchronization.

(6) Metaphors. The use of multiple metaphors within one system are acceptable in order to achieve different results and granularity of design. The authoring tool should provide metaphors for the user interface, but the author should be able to select and refine these as required. We may require additional metaphors for the author and user, and correlation of metaphors. There needs to be consistent metaphors for authors.

(7) Authors are not usually experts in design or in computers so the authoring tool should be easy to use, e.g., ShareME. There is a trade-off between easy-to-use and low-level flexibility that a programmer should be able to handle. Can we layer the authoring system? We need layered tools with levels of capability and required expertise. Are templates useful for this? The author can use the templates as far as possible but the system should be extensible and allow the author to go beyond these if required.

(8) Ideally, we would like people or authors to be able to do everything, but the creation process will probably require a team of different experts for each of the different tasks. Traditional

methods such as film making involve a vast number of different people for different tasks. Thus, we need tools for team development (both one site & distributed).

Tasks for the Future

(1) Authoring tools should provide a more integrated environment, and provide better integration together of the different media. We need authoring environments and not just authoring tools.

(2) The flexibility of existing tools is not that expressive for temporal and layout tasks, for example, the position of windows is often controlled by the Window Manager. There should be improved placement controls for the author when positioning things in time and space.

(3) We need to find metaphors for authoring systems, and improve existing metaphors such as a script, score, or storyboard. We need to provide consistent metaphors or consistent interoperation between metaphors.

(4) We must improve content editing, both techniques and tools for image and video. An example is MOVie which provides a tool for mapping a screen area for video. We should move towards automatic video and image editing.

(5) We must provide the author with guidelines for good design, i.e., the authoring system should advise the author in the following:

- Selecting the most suitable metaphor
- Selecting the most suitable multimedia layout
- Creating a good structure

(6) What is a good multimedia layout? We need to learn from well established concepts, e.g., typography.

(7) We need to explore the mapping between different media, e.g., text to speech, video to VR, and mapping between different forms of the same media to allow for disabilities, personal preferences, skills, equipment, etc. Which mappings are sensible and which are the most effective? Can they be generated automatically?

(8) We need to provide additional specific components within the authoring systems to enable development of distributed MM applications. e.g., teleconferencing, distributed kiosks. There is a need for:

- Personalization (different user views for distributed information)
- Structuring distributed information
- Specific components within authoring systems to provide for this

(9) We need to provide tools for team development (cooperative authoring systems) or CSCW applied to MM authoring--both one site & distributed.

(10) We need to establish standards for structuring information and cooperative authoring, for example by improving SGML, ODA, HyTime, etc. Make sure these systems are robust enough to be able to cope with emerging solutions and techniques.

Group II Report: Multimedia Systems Architectures

Group II Members: Martin Brenner, Martin Frühauf, Christoph Hornung, Klaus Meyer-Wegener, Max Mühlhäuser, Bernhard Tritsch

Overview

The overall goal of this working group is to define a framework for multimedia architectures, and to state requirements for other groups (such as the work group 1 dealing with multimedia user interfaces). To reach this goal, it was decided to put several position statements together.

First, different views on architectures are presented. They define a multi-dimensional space, each dimension spanning a certain independent aspect. Outgoing from this space, different paths are elaborated in more detail, leading to different position statements. This is clearly not yet a well-defined architecture, but may rather be seen as a first sketch to gather information about what aspects have to be considered. One aspect considered in more detail is formed of the underlying information types, i.e., multimedia and cooperative access. First, a definition of the information type multimedia is given. Herewith, a clearer understanding of the underlying information is achieved, and requirements to other views are formulated. Multimedia is seen here in conjunction with Computer-Supported Cooperative Work (CSCW). This information type is explained next in more detail.

Orthogonal to this, some of the function categories are elaborated in more detail. Data management is concerned with the aspect of long-term archival storage of multimedia information. It thus has to cover device management, data type (schema) management, search facilities, transfer of large volumes of data under time constraints and compression techniques. Communications deal with the transfer of large volumes of multimedia data over various types of networks. Exchange of data has to be executed in realtime and isochronously, i.e., the elapsed time required to transfer a certain amount of data has to be predictable. The data transfer services should be fault-tolerant and reliable. Security aspects of data exchange in common networks have to be considered. At last, the aspect of application development has been considered. Tools are availables and are continuously being improved to handle multimedia data. They must be integrated in a much smoother way than is the case today.

All in all, these are just different paths through the multidimensional architecture space. At some places, they meet--but it is just too early to combine them into a single architecture that covers all relevant aspects of multimedia systems.

1. Introduction

At the beginning of a discussion on multimedia (MM) system architectures, one cannot avoid the question: What is a multimedia system? This alone could be the subject of a fierceful debate. However, there are some proposals, and for the sake of brevity, they will be used without further questioning. In [Stei90a], Steinmetz, Rückert, and Racke suggest that "a multimedia system is characterized by the computer-controlled, integrated processing, storage, presentation, communication, creation, and manipulation of independent information that stems from several time-dependent and time-independent media" (p. 281).

The next question is just as difficult: What is an architecture? An architecture is a certain view of a system (or of large number of similar systems). It gives a comprehensive picture of either the functionality or the implementation of the system, leaving out the details, oversimplifying it to some extent, but making clear the main concepts, building blocks, and/or principles. Hence, an architecture can also be regarded a (complete) set of design principles: It simplifies the task of building a system, since it gives a gross structure and helps to remember all the subtasks that have to be accomplished.

An architecture should also provide a common frame for a variety of systems with different properties concerning performance, cost, etc. Applications will then select and tailor first the functions (services) required and then accordingly the building blocks. Tailoring can be as easy as setting a few parameters, but also as difficult as writing programs. This can be used to support:

- flexibility
- scalability
- interoperability
- quality

- reliability

Consequently, decompositions of existing multimedia systems should be guided by architectures, i.e., they help identify structure in those systems. Various aspects of the systems can be structured, e.g.

a) Structure the functionality w.r.t.

- level of abstraction, i.e., the (virtual) objects manipulated.
- level of integration (of media)
- contents, structure, and access (the CHM cube)
- the media

b) Structure the hardware:

- hardware/software boundary
- bottlenecks
- typical configurations:
  MPC ($5,000), MM workstation ($50,000)
- codecs
- network adapters

c) Structure the software:

- in layers
- in components
- according to functionality
- interfaces and protocols
- codecs

2. Views on Architectures

As we have seen, there is a common understanding that systems usually do not have a 'single' architecture, but that different architectural views can be drawn, describing systems (and problem domains) from different perspectives. First, the following possible basic views on architectures have been identified:

- information types (access, structure, content)
- categories of functions/services

- hierarchy of services (interfaces)
- hardware/software distribution, configuration, service integration
- application integration

Second, it seems worthwhile to summarize possible "elements" which an architecture might be made of. Such enumerations of elements follows below:

- functions (services)
- building blocks (components)
- interfaces and protocols

Third, functions can be grouped with respect to certain application classes or complex interoperations of components, leading to:

a) A CSCW view of multimedia systems:

- concurrent access
- sharing of data and workspaces
- joint editing

b) A data management view of multimedia systems

- archiving functions
- representations = codings = formats
- types instead of formats, i.e., access functions (operators)
- application neutrality

c) A communications view of multimedia systems

- allocations of components to nodes
- client/server systems

d) An application developer's (programmer's, author's) view of multimedia systems

- tools, toolkits
- configurations

All this different aspects and views will be discussed in more detail in the following.

2.1. Basic Views

The basic views are seen to be independent of each other, herewith defining an n-dimensional space.

The information types distinguish between access, structure, and content information. Herewith, basic information entities are identified, out of which complex "documents" can be composed.

The function categories distinguish between processing, transfer/communication, storage/retrieval, presentation, and interaction/capture. This view is based on "abstract devices" providing services.

A complex system consists of several service levels. Hardware, drivers, basic services, toolkits, and frameworks have been identified.

These services then must be mapped onto software and hardware components. This leads then to issues like configuration and service integration (not yet elaborated).

The view application integration deals with the integration of multimedia into applications. Here, the levels 'single media,' 'integration to multimedia,' and 'media-integrated systems' have been identified.

2.1.1. Information Types

A central question when defining the architecture of an information processing system is the definition of the type of the underlying information. To avoid the confusion that often occurs with the misleading usage of the terms multimedia, hypermedia, or cooperative work, a distinction between the following information types has been made:

- access
- structure
- content

These information types are orthogonal to each other and form basic entities, out of which complex documents can be composed. Time-variance is another basic concept which can be applied to all the information types.

2.1.2. Function Categories

This view is based on abstract devices. They can be seen as abstractions of types of real hardware devices providing different kinds of services. The following function categories have been identified:

- processing
- transfer/communication
- storage/retrieval
- presentation
- interaction/capture

The separation of function categories allows the clear definition of basic services, out of which a hierarchy of services with increasing complexity can be build.

2.1.3. Hierarchy of Services

Different levels of abstraction may exist for these function categories; leading to the layering of a software architecture. A common view of such layering is as follows:

Device drivers: This level is not considered visible or important to the application user or programmer.

Basic services: Offered as part of the hardware/software platform (e.g., the operating system or the network services, see below); the interface is usually an API.

Toolkit functionality: These are application-independent functional blocks which often belong to one of the above-mentioned categories; they are often interfaced via APIs.

Platforms: On the same level, one may find what is sometimes called platforms. These are attempts to standardize a set of underlying functions in a domain-independent manner across heterogeneous underlying systems (i.e., media peripherals, computer architectures, or the like). Examples are Apple's QuickTime-based platform or Microsoft's Multimedia Workstation platform.

Frameworks: Combine several underlying functions and tailor them to the needs of an application-domain such as banking, CASE, computer-aided instruction, etc.

Application functionality: off-the-shelf tools like authoring systems, or home-grown, i.e., customized or custom-built dedicated applications.

The above layers define a hierarchy, which also indicates a stepwise movement:

- from device-dependent to device-independent
- from generic (application-independent) to specific (application-dependent)

2.1.4. Hardware/Software Distribution, Configuration, Service Integration

This view deals with the integration of multimedia into applications. Here, the following levels have been identified:

- single media
- integration to multimedia
- media-integrated systems

2.1.5. Design Principles and Client/Server Considerations

While today client/server-based architectures are considered the state-of-the-art, they are in fact questionable. One might define three degrees of sophistication of distributed software architectures:

- Monolithic Systems
- Client/Server Systems
- Object Migration

The only distribution aspects of monolithic systems may stem from the capabilities to exchange files or information based on point-to-point user-invoked communication tools.

Client/server systems support distribution and stem from a scenario in which more and less powerful machines exist in a system, the less powerful ones being typically located at the users' sites. Client/server systems draw a lot of attention from the fact that they are mostly programmed based on the RPC (remote procedure call) paradigm, which is claimed to be "almost identical" to the well-known procedure calls in traditional programming.

The client/server concept has a number of drawbacks: First, the natural world is not separated into pure clients and pure servers; rather, most people act as clients and servers interleavingly. Real world modeling in software should allow to adapt to this. Second, since workstation power boosts, the distribution of powerful and less powerful machines may change rapidly in an organization, so the appropriateness of a system as client or server may change faster than the operators can cope with that change. Third, client/server software architectures draw--early in the lifecycle--a solid border line between client and server and define the protocol between them (cf. X Windows and the X-Windows protocol). Over the years, or even from one situation to the other, this borderline may turn out to be inappropriate, and the wish may arise to move this borderline, which is not possible in most systems.

Object-oriented distributed systems lead to a large number of fine-grained objects; their distribution among different workstations can be defined at startup time or even at runtime, if object migration is supported in the system. Moreover, the object-oriented approach supports the real-world modeling as requested above and is not bound to a strict separation of clients and servers.

2.2. Combining Different Views

The different views, when considered isolated, just allow to make rather general statements about the architecture of a multimedia system. However, by combining them, we get a narrower view to certain problems. As an example, the following refinement would be possible:

- multimedia information (definition of data types and methods)
- storage/retrieval of multimedia information (MM data bases)
- service hierarchy in MM data bases
- distributed MM data bases
- distributed MM data bases for banking applications

Please note that important areas such as operating system support and media processing support (cf. "software architecture" components as described in the chapter below) where not addressed due to time constraints and, in particular, due to the fact that no corresponding buzzwords were collected during the workshop.

3. Contents Information: Multiple Media

The first central point discussed in more detail was an attempt to define the characteristics of multiple media been the underlying contents information of a multimedia system. The different media types and their characteristics play the major role in a multimedia system.

3.1. Media Types

The usage of multiple media was seen as being essential to support both an almost natural human-machine interaction as well as an efficient computer support human to human communication. The latter one can be seen as the exchange of messages. Therefore, a first attempt was made to identify natural sending and receiving of information. In a multimedia system, sending is then realized via input devices, while the reception of messages is done via output devices. This leads to a scheme with the following keywords:

- Information Type: the human sense related with the exchanged information
- Devices: computer-supported input/output devices for information exchange
- Message Type: human interpretation of the exchanged informations
- Multimedia Object: computerized representation of the exchanged information

Information Type	Devices	Message Type	Multimedia Object
		Human Output - Computer Input	
Visual	camera	visual gestures	visual gestures
Audio	microphone	sound	sound, speech
Tactile	keyboard, mouse, glove, data suit	tactile gestures	tactile gestures
Tasty	?	-	?
Smelly	?	smell	?
		Computer Output - Human Input	
Visual	monitor, display	images movements	text, graphics, images videos, animations
Audio	speakers	sound	sound, speech
Tactile	glove, data suit, simulator	feel	pressure, temperature, rotation
Tasty	?	taste	tastes
Smelly	?	smell	smells

3.2. Natural Human Media Resolution

In order to define the requirements of multimedia systems, the natural human media resolutions must be pointed out clearly. This includes both temporal and data resolution. The data resolution in bits refers to the amount of data necessary to store one sample of media information.

Information Type	Temporal resolution	"Bit" resolution
Visual (See, Show)	<100 Hz frame rate	24 Bits RGB colors per pixel (true color) >300 dpi pixel resolution
Audio (Hear, Say)	>20 kHz (CD sampling rate = 44.1 kHz)	16 Bits linear (CD resolution) per sample
Tactile (Feel, Touch)	<10 Hz	>24 Bits for gravity, rotation, temperature, pressure and fine tactile
Tasty (Taste, -)	<10 Hz	>24 Bits for sweet, sour, bitter and salty
Smelly (Smell, -)	<10 Hz	>24 Bits (?)

The visual and audio information types ("See" and "Hear") are used for active communication ("Show" and "Say"), sometimes the tactile information type ("Feel") is included ("Touch"). Taste and Smell are almost always passive media.

The amount of data needed for the media is very high for the visual information type (> 1 MByte/sec.), moderate for audio (< 500 KBytes/sec.), and low for the tactile, tasty and smelly information type (< 1 KByte/sec.).

The constraints of a computer system that have to be taken into account when multiple media are to be adapted to computer devices can be derived from the facts mentioned above.

3.3. Computer-Based Media Standards

The first two of the natural human media (= information types) adapted to computers are well understood on the first glance. The handling of image and sound data has a long history in computer time-scales. So, there exist a number of standards.

Image:	Bitmap - TIFF, GIF, etc.
	Vector - HPGL, etc.
	Metafiles - CGM, Windows Metafiles, etc.
	Animated Images - Autodesk Animator, etc.
	Video - QuickTime, AVI, (MPEG), etc.
Sound:	Noise - AIFF, U-LAW, WAVE, etc.
	Speech - ADPCM, etc.
	Music - MIDI, etc.

The tactile medium is of growing interest, especially if it comes to virtual-reality systems. But so far, there is no common file format to store and process tactile information.

For the discussions to follow, smell and taste are not taken into account, because they seem not to be needed for multimedia applications of the near future. Nevertheless, on the long run, they may become more important and basic research work should be done.

The major problem there will be I/O devices, file formats, and processing methods.

3.4. Quality and Reliability of Multimedia Services

As can be seen in chapter 3.2., the quality of graphical media needs to be sophisticated for the natural human capabilities are very high when it comes to "seeing."

Nowadays the different raster formats provide high resolution concerning number of colors and pixels. But when it comes to scalability and cross-platform interoperability they are still poor. On the other hand, losing single units of information (pixels) does not destroy the whole picture. So raster images are easy to use with respect to reliability. This is even more essential for moving raster images.

Vector formats provide a very good scalability. The resulting picture only depends on the capabilities of the output device. But only very few images can be stored vector-oriented besides CAD applications. Moreover, if only single information units (vectors) are lost, the whole image may be ruined.

Metafile formats combine raster and vector images, but cannot solve all problems concerning quality and reliability.

So, what would be necessary is a file format that allows the usage of raster data, is scalable, and is fault-tolerant. A first step may be the fractal analysis of raster images, what allows the reconstruction of the original image to an extend selectable by the user. Scalability is the major issue here. That means size, resolution, colors, etc. must be free parameters, only dependent of the original image, the processing power of the target system, and the user needs. Moreover, the data has to be so fault-tolerant that lost information units will not spoil the whole image. Existing raster image formats might be extended into this direction.

For sound formats the problem seems not to be so difficult. The "spatial" resolution is not as high as with images. Therefore, lost information is not so relevant. Moreover, very sophisticated algorithms concerning error detection in music and speech are in the market.

3.5. Interoperability

The interoperability comes into account when communication is concerned. The same media files are to presented or to be produced on different hardware and software platforms. So, common interchange formats have to be defined, focussing on network communication and storage in databases.

In order to achieve real-time communication, all involved machines need to exchange a set of set-up informations in order to know about each others I/O, storage, processing, and interaction capabilities.

3.6. Intermixing Multimedia Data Types

The intermixing of multimedia data streams is mainly relevant for highly time-dependent media. All the others can be combined by simple relative synchronization methods like "start together," "stop together," "start B after A was running for x time units," "start B when A stops," etc.

Highly time-dependent media like animation sequences, videos, speech etc. need more sophisticated methods, for playback as well as for real-time purposes. So the intermixing of animation, video, and audio files is very important. First steps have been taken with QuickTime and AVI, combining different media into one file format. In the development of adequate tools and applications based on these standardization efforts can be seen as a major issue of the future. This is especially needed for CSCW application, where fast communication and multimedia data exchange is essential.

4. Access Information: Computer Supported Cooperative Work

4.1. General Remarks About the Information Type "Access Information"

While there exist also stand-alone multimedia systems, it is often the case, that multimedia systems are used to support working groups. Therefore, consensus was reached that, when discussing the architecture of multimedia systems, also the problem of cooperative access has to be taken into account. It turned out, that the aspects of cooperative access can be seen orthogonal to the aspects of handling multiple media.

4.1.1. CSCW Based on Access Information

Computer-Supported Cooperative Work (CSCW) is seen to be based on an own information type, called access information. Therefore, the aspects of CSCW can be seen independent of the aspects of contents (media) and organization (hyper structures).

4.1.2. CSCW and Conferencing: Computer Conferencing

One aspect of CSCW is to provide a conferencing environment for the involved working group. Such conferencing environments are based on the following types of communications:

- point-to-point communication
- multipoint communication
- multicasting
- broadcasting

Conferencing may be supported by computers. Typical examples for this support are:

- shared screens
- multipointers
- telepointing

This then leads more and more to computer-supported conferencing, or computer conferencing.

4.1.3. CSCW and Computing: Conference Computing

Besides conferencing, CSCW also provides a computing environment. Computing environments may be distinguished according to the following characteristics:

- degree of freedom
    - shared screen (follow user)
    - individual screens

- social roles
  - reader
  - writer
  - chairman

- interactivity
  - static sessions
  - dynamic sessions

With increasing conferencing capabilities, we approach the conference computing.

4.1.4. The Concept of a Cooperative Document: Private Annotations

One of the characteristics of computing in a conference is the concept of a cooperative document. In traditional systems, a document is represented by a file, having certain access rights. In a cooperative environment, a document (e.g., an electronic book) is shared by multiple users at the same time. Therefore, it is essential that users can make personal annotations to this document, and, herewith, privatize a copy.

4.1.5. Classification of CSCW Environments

CSCW environments may be distinguished between the following categories:

- same time/same place
- same time/different place
- different time/same place
- different time/different place

This has consequences for the implementation of the functions mentioned above as follows:

- communication:
  - same time: conferencing
  - different time: mailing

- computing:
  - same time: interactive conference computing
  - different time: cooperation of working groups

4.2. CSCW and Multimedia

A detailed look at multimedia is given in chapter 3. Here, we only add some comments concerning the usage of multimedia in CSCW environments. Multimedia has two impacts on CSCW: It may be used as a tool for communication as well as a contents of a cooperative document.

4.2.1. Multimedia as a Communication Tool

For CSCW at the same time (interactive CSCW), interactive communication between the users is essential. This means, that at least an efficient audio connection must be provided. Video can be seen as a meaningful add-on.

As mentioned earlier, different kinds of communication have to be implemented to support for example the different cooperation environments:

- point-to-point: for dialog environment.
- hot-keys: to enable dialog communication in cooperative environments (passing the microphone)
- multicast: for building subgroups in large environments
- broadcast: for anonymous teleteaching.

4.2.2. Multimedia as Document Contents

When dealing with cooperative documents, we have to consider the facts that now different users may have access to different contents information and that the contents of such a document have to be presented to different screens at the same time.

4.2.3. Multimedia User Interfaces

In a cooperative environment, one has to deal with at least the following different information types:

- access information
- contents information
- structural information

The user interface has to reflect this fact. Effective presentation and interaction tools to deal with these information types have to be provided. It could be useful to associate different media with different information types. For example, requests from other users could be presented as sound even if they are not caused by spoken input.

4.3. CSCW and the View of Functional Categories

The concept of CSCW has an impact on all of the functional categories mentioned above.

4.3.1. CSCW and Processing

Concerning processing, algorithms for the following problems have to be developed:

- access management
- user identification
- user authorization
- ...

4.3.2. CSCW and Storage/Retrieval

Compared to the storage in a single user environment, the following additional issues arise:

- fine-grained transactions (very small access units)
- integration of (dynamic) access rights into the data base, reflecting the current status of the actual session

4.3.3. CSCW and Presentation/Interaction

Concerning the user interface, new metaphors have to be developed and implemented. The access status of a cooperative document has to be presented, allowing the user to see what parts of a document are free and which ones are used be others. When accessing an entity, conflicts may occur. Especially in large and widely distributed platforms, the update time and, herewith, the recognition of such problems may take a while. The user has to be informed about such situations (e.g., by a message "negotiating...").

An other important issue is to require document entities currently used by others. This can be done by conferencing, but also by sending a message and, for example, highlighting the requested zone.

Especially interesting is CSCW for hyper-documents. Tools have to be developed to get access to hierarchically structured information.

4.3.4. CSCW and Transfer/Communication

The concept of CSCW puts special requirements onto the transfer/communication functionality. As in other distributed multi user applications, CSCW is based on the notion of a transaction. Requests have to be checked, a time stamp must be attached and access conflicts have to be detected and solved.

4.4. CSCW and Application Integration

CSCW, as mentioned earlier, is based on the information type "access information." Therefore, there is an intrinsic difference between cooperative and non-cooperative applications.

4.4.1. Generic Cooperation Support: Shared Applications

The idea of Generic Cooperation Support is to embed traditional applications in a cooperative environment. Here, the whole document underlying the application is seen as a single access entity. All users have the same rights, conceptually, we see a single user, but multiplicated application.

4.4.2. Joint Applications

In joint applications, still the document is a single access entity. However, the communication features are more elaborated. The write access right to the document may change during the session, and there may be a hierarchy of users. An example may be the blackboard in a classical teaching environment, where the teacher and the students may have write access, but the teacher has the privilege to give the access right. This type of application still can be implemented as a "conference computing shell" over a traditional application.

4.4.3. Cooperative Applications

In true cooperative applications, the access information is an inherent part of the underlying data structures. The underlying document now consists of several access entities (fine-grained access). Several users may have write access to different access entities in the same(!) document and at the same time.

5. Data Management

Multimedia systems have different requirements with respect to the management of their data. First, short-term storage must be distinguished from long-term storage. While the former deals with manipulations, direct access, specific formats, realtime, and synchronization, the latter has to cope with large data volumes, storage device management, heterogeneous environments (in space and time), search requests, and compression methods. Short-term storage is determined by the application and the computer system used, long-term storage must be much more general. Short-term storage can be further subdivided into main memory (not yet applicable to sound and video of reasonable size) and local secondary storage.

Second, the different media can be discussed. They have already been characterized in chapter 3, and these characteristics lead to many different requirements on data management. Some of the media are time-dependent, others are not. They all consist of a large number of small elements (e.g., characters or pixels), but these elements have very little in common and are arranged in different patterns, e.g., as set, sequence, or matrix.

Third, data management can be used to represent the organization of large numbers of media data objects. Hyperstructure is one example of such an organization, but not the only one. Media objects can also be organized in sets, either disjunct or overlapping. This can be used to classify them (e.g., X-rays, landsat photos, aerial photos).

Finally, the third dimension of the CHM cube is also relevant for data management: Access to the data can be supported in many different ways. They have to be located, either navigational or descriptive, and then interpreted. Concurrent access from several users should be allowed, but should be coordinated to preserve consistency. In particular, atomic sequences of updates on various data (transactions) should be supported.

5.1. What is a Multimedia Database?

A multimedia database is a large collection of media data objects on secondary (non-volatile) storage. Any such collection can be regarded as a data base. However, it makes a big difference how such a collection is managed. The operating system (OS) is in control of the external storage devices and offers low-level functionality to use them. It is suggested that there should be an additional data management system in between the application and the OS. However, long-term storage will need a different data management system than short-term storage.

A multimedia database management system (MMDBMS) enhances the functionality of a multimedia system with respect to the long-term storage and retrieval of multimedia data objects. The user may archive multimedia data objects and later search the archive. The objects are accessed not only by name (identifier), but also in a navigational and descriptive way.

Such an archive is also used to share data among a group of users. So data moved into the archive can be made available to others. Certain modes of information exchange and cooperation are supported by this.

Finally, the archive serves as a kind of public or corporate library. It offers access to standards, guidelines, templates, background info, legal info, etc.

It is important for the user not to be bothered with the specifics of matching his/her local facilities (workstation, peripherals) with those of the database server. Instead, the server should adapt to the environment of each particular user. So if the user wants to see an image, listen to a sound recording, or watch a video, he/she should be able to do so without selecting formats, converters, viewers, etc. In a first step, the system should offer a default viewing/listening mode (making the best of available resources). At wish, the user could then customize the presentation.

5.2. DB versus File Handling

Deciding about using either a DBMS or a file system is not so much an issue of end-user functionality but of system implementation. The same functionality (see 4.1) can be achieved with either a DBMS or a file system. There will be a difference in speed, in cost, and in maintenance effort. However, the implementation of some functionality (e.g., sharing and multi-user operation) requires a significant effort if done with a file system. Hence, in discussing the functionality of either the file system or the MMDBMS, we are talking about an API, not a UI (see 4.5).

File system and DBMS are two different "instances" of the service named "data/information structures." They offer different (internal) service to the application programmer.

The advantages of using a file system are:

- immediately available
- low cost
- highly optimized solutions possible (but not guaranteed)
- efficiency
- performance, speed (saving milliseconds)

The advantages of a (to be developed) MMDBMS are:

- independence of application programs (and toolkits) from storage and compression technology
- data abstraction (application neutrality, openness)
- explicit type info (metadata) for multimedia data objects
- "integration" with formatted data
- represent action of (typed) relationships
- powerful and efficient search facilities
- consistency (centralized control)
- multi-user operation with fine-grained concurrency control
- fault-tolerance (transactions, logging and recovery)
- potential for incremental and global optimization (single point of control)
- maintenance support after changes in environment (new clients, new storage technology; saving man-months)

It should also be noted that there are intermediate solutions. A system that offers data independence but not multi-user operation and transactions is much smaller than a full-fledged DBMS and thus also cheaper. Unfortunately, todays commercially available DBMS cannot be tailored to that extent.

The choice should be made be the application programmer depending on the needs (and the budget) of the application.

In the following, we shall use the generic term "data management system" to refer to both a file system (plus application-specific functionality) and a DBMS.

5.3. Dynamic Data/Time Dependencies

DBMS until now had to cope with static data only. However, there has been some work on real-time databases, where queries had to be executed under tough timing constraints [SIGMOD88a]. The technology is there, but due to the limited market, only a few systems are commercially available on the specialized platforms used in real-time applications.

Here, it is very important that the data management system can distinguish types of data (and does not just offer containers for bytes). Storage management for time-based data should be guided by the real-time requirements. For instance, to achieve the required bandwidth in I/O, disk-striping can be used. Also, an appropriate storage device must be chosen.

So, only if the information about timing and time constraints is available to the data management system, it can react and provide specific means to obey them.

5.4. IIF versus Image Data Types

A format is a set of rules about how to code the media object. It defines a (physical) layout for the media object. Representation and coding are synonyms of format. A media object must have a format to exist, i.e., to be stored anywhere.

A type identifies a set of operations that allow to work with a (media) data object without knowing the format. The implementation of the operations can use different formats in different environments, still belonging to the same type. To design a proper set of operations is a difficult task.

The system should maintain and use metadata that provide information about the type (and format) of the data objects.

In short-term storage, usually very specific formats are used that match the local tools. There is no need to encapsulate them as types. The long-term archive however must cope with many different formats. To avoid redundancy, it will map them to the most general and/or most efficient format and encapsulate this. The create operator of the media data type thus accepts any format as input and converts it into the internal storage format used. The retrieve operator on the other hand can provide the client application with the format it needs. So, in the "create" and "retrieve" operations one has to identify the format used locally, and the data management system decides about the internal format to use.

While there are many formats, there is still a need for good types!

5.5. End-User Interface to Data Management

As said before, the data management system is considered to be of main use for the programmer of the multimedia application, not the end-user. Nevertheless, once the functionality of a DMS is fixed, it could also be made available to the end-user to let him/her look into the system and learn about the implementation (meaning?) of the higher-level functionality.

However, this should be compared to reading the output of a compiler, or even worse: to modifying it. The interface is designed to rather low-level. As a consequence, you can do more, but you have to do it in smaller steps.

5.6. Interoperability

Interoperability of DMS means the enhancement of a single DMS instance so that it can be accessed as one of several DMS instances (maybe at different sites) that cooperate to provide an answer to a query or to allow for consistent updates. The problems of schema integration, transaction management, and query optimization have not been solved for standard formatted data [Brei90a], so it seems to be too early to attack them with the additional burden of multimedia data objects. Some of the proposals for object-oriented database systems might extend to the management of multimedia data as well.

6. Communications

6.1. Requirements on Communication Services for Multimedia Applications

Due to the mass of data to be transferred in distributed multimedia systems, the overall requirements on communication services are very high speed and almost unlimited bandwidth. Exchange of data has to be executed in realtime and isochronously, i.e., the elapsed time required to transfer a certain amount of data must be predictable. The data transfer services should be fault-tolerant and reliable. Security aspects of data exchange in common networks have to be considered.

From the users or application programmers point of view all those aspects are summarized using the term "Quality of Services."

6.2. Standards

Obviously, standards for communication protocols are very important if data exchange in wide area or metropolitan area networks is considered. Unfortunately, the complexity of higher levels of protocol standards limits the peak performance provided by high-speed networks. On the other hand these protocols are more convenient for application programmers.

Most of todays networks and standards for communication protocols do not fulfill the requirement of isochronity.

6.3. Trade-Offs in Communication Protocols

Mechanisms to provide fault-tolerance and security require additional data exchange and database queries. These are considered to be an overhead by programmers and users of applications which transfer huge volumes of data in short time.

6.4. Quality and Reliability of Services/Servers

Redundancies of servers, workers and services in a distributed system increase the fault-tolerance if any component in the entire system should be out of order.

6.5. Client/Server Architectures for Multimedia

In a client/server architecture the server and the communication channels of the server might be a bottleneck as far as high volume data transfer is concerned. Is is more evident the more clients rely on a single server. On the other hand the distribution of common data over many nodes in the network requires additional services to ensure the consistency of the distributed databases at any time.

Topologies of networks and workers/servers

Multicasting vs. client/server solutions

6.6. Interoperability

Interoperability is defined as the capability of autonomous systems to cooperate on a common task and solve a common problem.

7. Development Support

7.1. Implementers and Users

When talking about tools for multimedia we have to distinguish the implementer of multimedia functionality and the user of multimedia functionality, e.g., the author of a system with a multimedia user interface.

At the moment due to the variety of technical problems in the area of multimedia it is impossible to give an overview of the desired toolkits for multimedia Implementers. So let us focus on tools and toolkits for authors of systems using multimedia.

Multimedia scene in nowadays reminds me of the early times of personal computing when you could make good money by building and selling your own word processor. Like word processors multimedia applications will become widely used and so, for being able to build many multimedia applications well fitted to the enduser's needs, we need standards, platforms and powerful toolkits.

7.2. Services

What we need for multimedia is something like X is for computer graphics. We need a (software) platform that reduces users deal with hardware dependencies to a minimum. Of course, the user still has to take into account what functionality he can get on a specific hardware platform. But it should not be his problem how to get it.

- These services should be hardware independent.
- The services should be offered to the user both on local and remote host in the same way.
- These services should include capture, presentation, transport across the network, storage and processing of multimedia objects.

7.3. Tools and Toolkits

We need a toolkit that allows the author to deal with multimedia objects--not with a lot of bits and bytes. This toolkit should provide a set of basic multimedia objects and should map the functionality of these basic objects onto the platform mentioned above.

Properties of this toolkit:

- Openness towards new interaction media
- Object-Orientation

Each object offers the author/user all properties changeable and all operations executable in the specific situation.

- Direct Composition

No programming needed for specifying layout of a user interface.

Less programming needed for specifying the interactive behavior by use of spatialization of structure and time.

- Dynamic behavior can be defined interactively.
- Integration with graphical user interfaces.
- Use of MM in the Design-Process.

7.4. Application Framework

The construction of the application framework should be done by means of the toolkit, not by programming. To achieve this goal some requirements have to be made for the toolkit:

Aggregation

The toolkit allows the author to combine objects to a single more complex object. The author defines interactively the dynamic behavior of the whole object, the dynamic behavior of the parts of the object and the relations between them. In this way the author defines application specific object (or classes).

Example: Medical Doctors often want to mark an "area of interest" in X-ray images. The author combines a container element, a pixmap and a polyline to a new object of the kind of "X-ray image." The author defines: The polyline is invisible. When the user (the medical doctor) clicks on the image, a visible copy of the polyline is created and the polyline changes according to users mouse movements. In this way no programming was necessary to create a application specific object.

Each object contains its design functionality.

8. Conclusions

The goal of this working group was to identify views on multimedia architectures. To reach this goal, first different aspects of multimedia systems have been identified and isolated. This has led to a more detailed investigation of media as the basic contents information and computer-supported cooperative work based on access information. Requirements of multimedia user interfaces in cooperative environments have been stated. Data management and communication as central components have also be investigated. At the end, the working group has discussed aspects of development support for multimedia applications.

Altogether, this report reflects the different options stated in the two days working group discussions and may serve as a first initial step towards the development of a reference model for multimedia system architectures.

9. References

Brei90a Breitbart, Y., "Multidatabase Interoperability," ACM SIGMOD Record, vol. 19, no. 3, Sept. 1990, pp. 53-60.

SIGMOD88a ACM SIGMOD Record, Special Issue on Real-Time Database Systems, vol. 17, no. 1, March 1988.

Stei90a Steinmetz, R., Rückert, J., und Racke, W., "Multimedia-Systeme," Das aktuelle Schlagwort, Informatik-Spektrum, Bd. 13, H. 5, S. 280-282 (in German).

Group III Report: Digital Video in Multimedia Systems

Group III Members: Bernd Girod, Ralf Guido Herrtwich, Georg Rainer Hofmann, Takahiko Kamae, Meinrad Niemöller, Stefan Noll, Ralf Steinmetz, Ana Sofia Vieira

1. Introduction

The integration of motion video into computer systems offers many new opportunities for multimedia applications. Unfortunately, it also implies a variety of technical problems still to be overcome. The data rate of a raw video signal in PCM format can be more than 100 Mbit/s. On the other hand, the bandwidth available on computer busses, mass storage devices, and local area networks is typically a lot smaller. For example, Ethernet has a peak rate of 10 Mbit/s, typical hard disks will support a sustained rate up to 10 Mbit/s, a CD-ROM runs at a few Mbit/s, ISDN has a rate of 64 kbit/s. Thus, data compression is a key technology to integrate motion video into computers.

Video compression has made great progress in the last ten years, however, it has been developed and optimized for broadcasting and videotelephony applications. With a few exceptions, these developments have assumed a fixed rate digital channel with negligible error rate and well-defined parameters of the video source and the display. Only recently, there has been an increasing awareness of the special requirements that have to be met for the integration of digital video into computer systems.

This report summarizes the results of discussions of a working group (Working Group III) that was formed during the 1992 Dagstuhl workshop on "Multimedia System Architectures and Applications," being held from November 2nd to 4th in 1992. We have focussed on fully digital integration of motion video into the computer. Currently, there are many systems that combine motion video and computers by controlling analog video devices. We are not discussing these systems here, since they will probably disappear as soon as there are fully integrated digital solutions. In our work, we concentrated on open system architectures that are typical for a workstation or personal computer environment. We are aware, however, that there are many multimedia applications for the consumer market, where closed systems (such as CD-I) are likely to prevail. Many of our conclusions hold for both types of systems.

This report is structured as follows. In Section 2, we outline a few typical application scenarios and attempt a classification of applications. In Section 3, we summarize the requirements that result for video compression schemes for the different classes of applications. Section 4 discusses the performance and features of some of the proposed compression methods, both standardized and proprietary. Scalable video is such an important topic that we devoted the entire Section 5 to it, while Section 6 discusses issues arising from the relationship between video and other media modalities, such as the representation of still images or synchronization with audio.

2. Video Applications

In the area of multimedia systems, there is a considerable number of applications which make intensively use of information types which are referred to by terms as

(1) analogous and digital video,
(2) motion pictures,
(3) image sequences,
(4) time-variant images,
(5) and the like.

The variety of the above listed terms shows that it is worth to consider

(6) the underlying structures,
(7) the kind of user interaction, and
(8) the corresponding data types

which are employed by the various applications. The following paragraphs illuminate these applications.

2.1. Application Scenarios

The following paragraphs describe--exemplary versus exhaustive--some applications in more detail.

Aspen Video Map

The Aspen Video Map has been one of the early developments in the area of interactive video systems (at MIT, in the late 1970s). It is a collection of realworld video sequences which have been taken from the city of Aspen, Colorado. The video sequences show in essence drive-through pictures from the various streets and places of the city. These video sequences are arranged as "clips" in a storage-and-retrieval system. On a map, the user may interactively select the street or place he would like to see. The system retrieves and displays the according video sequence.

Karaoke

In the area of consumer-electronics-oriented multimedia systems which are serving for entertainment, for leisure, or as electronic games, Karaoke is of increasing importance for the Japanese market. Karaoke systems are presentation-only systems for the multimedia presentation of "songs" for singing-along. A video/audio sequence is displayed simultaneously with the text of the song--whereas the audio is containing only the background music. The actual parts--which correspond to the background music--of the text are highlighted.

Video-On-demand (on-line/off-line)

Systems for video-on-demand are oriented towards the consumer and home video market. The basic idea is that the user--in contrast to the traditional television system--shall not be restricted to the watching of centrally broadcasted TV programs. Instead, the user may interactively select from a list--comparable to videotext tables--of films, shows, etc., what he would like to see. The selected TV program is then sent to the user's home TV set either in the "on-line" or the "off-line" modus. In case on-line modus is used, the selected TV program is distributed from a central TV program supplier via broadband networks to the user. The user's system is a presentation-only system, with no need to store the distributed TV program. For the off-line modus, the selected TV program is send to the user not necessarily in realtime. Therefore, for distribution, narrowband networks may be employed, as well as network-independent data storage devices (like laser discs, etc.). In case a network is employed, the user's TV-like device must intermediately store the distributed TV program, until the user would like to watch it.

Videophony

Videophony--sometimes referred to as "picture telephone" as well--is a point-to-point (with 2 participants), or even multipoint (more than 2 participants, see also CSCW below), telecommunication service. Videophony allows for the signal-oriented transmission of both, video and audio. According to the bandwidth of the network, however, the image quality of the transmitted video may be reasonable low, both in terms of spatial and temporal resolution of the video. To meet this problem, the videophony standards (such as CCITT Rec H.261) make intensive use of sophisticated compression algorithms, which include transform coding, motion prediction and compensation, and the like.

TV/Video on LAN

Some videophony-like applications use local area networks (LANs) for the inhouse (local) communication of TV/Video signals and/or the according data streams. Those applications may include the "netcasting" of audio and/or video messages, e.g., inhouse announcements and/or bulletins. Other applications may be surveillance-oriented, where video data is transmitted--either occasionally or permanently--over LANs for the centralized monitoring of facilities. This may include also user interaction to allow, e.g., for the remote steering of camera positions, and the like.

CSCW/Multipoint Conferencing

There is a number of applications for collaborative work. These are affecting technical, e.g., CAD-oriented, work as well as meeting and workshop scenarios. For workshop and meeting scenarios, the following points have to be taken into consideration:

- Multipoint interconnection needs to be established. Typical numbers of conference participants range from at least 2 up to 4 to 5. Groups with more than 6 conferencing participants are becoming increasingly unsuitable for teleconferencing.

- A "forum" situation must be achieved, where passing over and taking the conference chair, the floor (for speaking up), and stepping into a discussion needs to be provided by the system

- The privacy of the user must be shielded against unwelcome visual insights. That is, the user must not be watched by cameras in his office/private space unless he explicitly wants it.

- The spontaneous setting-up of conferencing sessions (especially for point-to-point connections) must be achievable. The corresponding conferencing hardware systems and tele-communication terminals--under ideal conditions--shall be implemented as desktop systems.

- For conferencing not only video and audio is needed, but communication needs also to be assisted by telematic services, such as:

    - transmission of high-resolution still images, to allow for the readability of text with even smaller letters,

    - for point-to-point connections: telewriting, which provides vector-graphics shared sketchpad functionality incl. hardcopy,

    - for multipoint connections: shared whiteboard, whose functionality maybe quite comparable to the telewriting devices.

- The conferencing equipment must be affordable in terms of financial investments per workstation and/or per user.

Multimedia Mail

Multimedia Mail forms extensions to the well-known email systems, as provided by, e.g., the message handling systems (MHS) specified by CCITT Rec X.400 series. This is achieved by extending the functionality of the "messages" which can be handled and passed forward by the MHS. Therefore, the message structure's capability is increased by adding the "multimedia" data types (still image, image sequence, audio, etc.) as additional message body part types. The traditional modus of store-and-forwarding mail messages, as used by the MHS, is applied to the Multimedia Mail systems and applications.

2.2. Classification of Scenarios

Taking into account the above mentioned scenarios, a classification of these, with respect of the functionality of user interaction, can be given and characterized as follows:

(A) Broadcasting

The user may select among the broadcasted channels and programs the one that he would like to watch on his receiver set. The user may step in and out the broadcasted program by switching ON/OFF the receiver device.

(B) Interactive Retrieval

The user may select among the provided pieces of multimedia information, which is, e.g., select by title or by index. The system provides typically capabilities for the playing/displaying the information, as well as searching, fastforwarding, and rewinding through the multimedia documents. In contrast to broadcasting, the user may additionally halt/pause the information retrieval, and resume later on. Of course, It is also possible, in contrast to broadcasting, to retrieve the document's information repeatedly.

(C) Editing

The editing of document necessarily encompasses the functionality of the interactive retrieval of documents (see paragraphs above). Referring to the retrieval as a "read" functionality, editing must encompass also a "write" functionality as well, in order to allow for the storage of multimedia information.

For editing, two areas need to be distinguished carefully:

- The editing of data structures (as described by syntactical rules).
- The editing of contents (as described by semantical contexts).

(D) Conferencing

The functionality which needs to be provided to the user by conferencing systems may be grouped as follows:

- Broadcasting functionality; see chapter (A) above. There may be participants allowed to attend at--even one or more--conferences who have an "observer" status. For these, it is essential to step in/out the ongoing conferences.

- Information retrieval functionality; see chapter (B) above.

- Editing functionality; see chapter (C) above.

- Floor control functionality: The floor, which includes to handle and pass over the right to speak up, as well to step into the ongoing debate, must be controlled by the chair. By separate mechanisms, the role of the chair may be passed over to another participant.

- Workspace and device management: The use and the # of the available telematic services (telewriting, shared whiteboard, etc.) must be controlled by the conference.

3. Criteria for Video Compression Schemes

The sheer amount of digital video (which is 216 Mbit/s for CCIR 601 encoded television images) leads to the need for compressing it when the data is stored or exchanged with today's computing equipment. To arrive at a data volume that this equipment is able to handle, compression by a factor of at least 100 is required. Using today's technology, this is only achievable with lossy compression.

A general problem for every lossy compression algorithm is to reduce the amount of data to the maximum extent possible while still providing acceptable (or appropriate) video quality, low algorithmic complexity, and short compression delay. Whereas rate, complexity, and delay can be measured easily, video quality can hardly be determined in an objective manner. To determine whether a compression algorithm yields the desired level of quality, one usually chooses a set of video sequences typical for a particular application domain and empirically lets a larger number of users evaluate the compressed samples. In general, the application domains in Section 2.2 require the following levels of quality:

Applic. Scenario:	**Level of Quality:**	**Example:**
Broadcasting:	medium high	(television "broadcast quality")
Interactive Retrieval:	medium low	(VHS quality)
Editing:	high	(television studio quality)
Conferencing:	low	(below VHS quality)

How much data a compression algorithm produces or generates within a certain amount of time determines the bit rate of the data stream. Yet, the mere information of bit/s provides no information about the sophistication of a compression method as it is not related to the underlying raw data, e.g., the image size. How good a compression algorithm is can better be characterized by the compression ratio achieved; it can be measured in bit/pixel. Good compression algorithms for motion video today achieve a compression ratio below 1 bit/pixel. With this compression ratio, the following bit rates have turned out to provide the appropriate quality for the above mentioned applications (thereby also providing a more precise quality measurement):

Application Scenario:	**Bit Rate:**
Broadcasting:	5-10 Mbit/s
Interactive Retrieval:	1-5 Mbit/s
Editing:	5-20 Mbit/s
Conferencing:	0.05-2 Mbit/s

How complex a compression algorithm is depends on its processing and memory requirements. Algorithmic complexity determines above all whether the compression can take place in real time, i.e., whether before the arrival of the next video frame all compression steps for the previous frame have been computationally completed. Arguably, one can also measure complexity in terms of hardware required to perform the compression task--both in terms of speed and in terms of cost. For our applications, complexity requirements are as follows:

(1) Broadcasting: Can be high for compression, should be low or medium for decompression.

(2) Interactive Retrieval: Can be high for compression, should be low or medium for decompression.

(3) Editing: Should be low.

(4) Conferencing: Should be low.

The delay of a compression algorithm, i.e., the time it takes for the algorithm to process a frame, obviously also determines its interactive qualities. Application requirements on end-to-end or roundtrip delay impose restrictions on the time the compression may take. For the applications we consider, the requirements on delay reflect the tolerance on complexity:

Application Scenario:	**Bit Rate:**
Broadcasting:	below 1 s
Interactive Retrieval:	well below 1 s
Editing:	below 150 ms
Conferencing:	below 150 ms

Apart from these major criteria for evaluating the appropriateness of a certain compression technique, several additional features of a video compression scheme should be considered. These include:

(1) Random access: Certain applications do not access a video stream sequentially from the start, but start playing it from an arbitrary position; they may browse through the stream in fast forward or rewind mode or even play the stream backwards. For these applications, random access to all positions within the stream is required. For interactive retrieval and editing applications, random access is a must. It is usually not required for broadcasting and conferencing.

(2) Editability: Not all compression formats lend themselves easily to editing the video data. In general, the extent to which inter-frame encoding is used determines the ease of editing: The more individual images stand for themselves, the easier the editing process is. Obviously, editability and random access are tightly coupled.

(3) Symmetry: A compression algorithm is called symmetric if compression and decompression take about the same number of operations. Today, symmetric algorithms are used for real-time interaction, whereas asymmetric algorithms are applied to

one-way communication. Symmetric is, hence, desired for editing and conferencing, whereas asymmetric algorithms can be found in the broadcasting and information retrieval domains.

(4) Robustness: How robust a decompression algorithm is depends on its ability to cope with bit or message errors in the video stream to be processed. In distributed environments the exchange of video streams across a network may induce such errors. Neither should these errors cause the decompression algorithm to stop, nor should the errors become notable to the users. While robustness is a general requirement for all compression algorithms used in a distributed environment, it is a must in conferencing applications as these are always run across networks.

An additional requirement for video compression algorithm that results from the dynamic environment in which they are used is scalability. We delay the discussion of scalability to Section 5 and first examine how today's compression schemes meet the requirements outlined above.

4. Scalable Video

4.1. Definition of Scalable Video

In the discussion on the requirements of the different classes of video application scenarios we found that the scalability of video is highly desirable for all classes.

We define loosely scalable video as the ability of a system to adapt to a certain bandwidth "on the fly." Bandwidth of a system may be reduced by the allocation of computing or network resources to other tasks. In the dynamic environment of a multitasking system, it is therefore necessary to adaptively scale down or scale up the bandwidth of the video task.

To understand the usefulness of scalable video, it is helpful to consider human perception of visual information. It is a well-known fact that the image perceived by a human is progressively build up from a coarse resolution to finer detail. Moreover, the human perception process dynamically scales up the resolution around the focus of interest (point of view). We believe that computer scientists can learn a lot from the experiences made by television engineers when defining a coding scheme for TV broadcasting. This applies to the coding of still images as well as to the coding of motion video.

Experiments have shown that the relation between the bandwidth in terms of bits per second to the image quality perceived by humans is not linear. With increasing data rate the image quality quickly rises up, but then further increase of the data rate leads only to a slow, asymptotical increase of image quality towards a threshold.

4.2. Domains of Scaling

We do not understand scaling video as the mere change of the video image size. The scaling of the bandwidth of a video can be achieved in different domains:

Time:
This is the most common approach; here, the rate of frames presented per second is changed dynamically. As we can see in TV technology a rate of 20 to 30 frames per second gives humans the impression of smooth movement. In case of, e.g., congestions on a network even a lower frame rate can be accepted.

Space:
The resolution of a (video) image is defined by the number of pixels per viewing angle. An image can be progressively build up by increasing the number of pixels in a hierarchical fashion.

Frequency:
If a image is transformed into the frequency domain, the bandwidth can be reduced by cutting the higher DCT-coefficients. The back-transformation then leads to unsharp images (loss of detail).

Amplitude:
An image quantized with a resolution of 8 bits per pixel, e.g., can be reduced in bandwidth by cutting the least significant bits.

Color space:
A resolution of 8 bits per color (red, green, blue) and pixel is called "true color" (24 bit). This brute force approach does not take the human perception into account. The TV coding with the its separation and subsampling of the chrominance and luminance signals is one approach to reduce overall bandwidth.

Best results can be achieved by combinations of these approaches.

To review the different standards and proprietary schemes for compression of video in the context of scalability it is useful to identify some criteria:

Graceful degradation:
In case of the drop of one or more frames the quality of the video image should not go down to zero but rather decrease in a graceful way. In this context it is also an important question how long a certain compression scheme needs to recover (e.g., from drops of frames) to full image quality.

Partial decoding:
For the transmission of video over networks it is useful to progressively transmit the video images. For that, a compression scheme should allow for the partial decoding of the incoming chunks of video information.

4.3. Review of Standards

Based on these criteria and the characteristics outlined in chapter 4, we can review the standards and proprietary schemes:

MPEG-1:
The MPEG-1 compression scheme is not scalable. We believe that the relevance of MPEG-1 could only arise from the mass production of dedicated chips, which will be not scalable. Because of the complex structure of I-, P-, and B-frames in MPEG-1 it is especially difficult to recover from frame drops (caused by network congestions).

MPEG-2:
The follow-up of MPEG-1 will be scalable in the extent that the bandwidth will be allowed to drop from the MPEG-2 rates (5-20 Mbits/s) to MPEG-1 rate (1.5 Mbit/s).

JPEG and Motion-JPEG:
Because there is no interframe compression in JPEG, is very easy to drop frames and also to recover from these drops.

px64 and H.26x:
The px64- and H.26x-compression schemes are not scalable. Only future versions of the H.26x standards will address to the scalability problem.

Roadpizza:
Quicktime's compression scheme roadpizza makes use of scaling of bandwidth in the time domain (at least). The bandwidth can be dynamically adapted to the workload of the processor. Details are not known by the authors because of the proprietary character of the compression scheme.

Fluentlinks:
Fluentlinks allows video communication across networks and uses a Motion-JPEG scheme with some proprietary add-ons.

4.4. Software-Codecs and Hardware-Bottlenecks

The scalability feature of a compression/decompression scheme is especially interesting when it comes to software-only realizations on general-purpose processors. Beside demanding workload-dependent scaling for decompression, software-codecs call also for simple and fast algorithms. The MPEG-1 standard is therefore not a good candidate for software-only realizations.

We expect that customers will more likely accept poor image quality with a cheap software-only solution today when they can expect better image quality with the next generation of general-purpose CPUs. In contrast to that, a customer subscribing to a hardware solution of a codec, e.g., a dedicated board, can only grade up to the next generation (of image quality) by throwing away the boards or plugging in new processors.

So, an interesting question is, when the different compression schemes can realized software-only on general-purpose CPUs. Projecting the last years progress of increase of CPU power into the future, we can expect to see software-only realizations of the MPEG-1 standard in about 4 to 5 years.

It is an open question how the development of other components of a general-purpose computer like RAM, bus and network adapter can keep pace with the development of CPUs. For the storing of video images not only a huge memory size is necessary, but also the access-time needs to be low. For the cooperation of (multiple) CPUs with the RAM the system bus needs to provide a high bandwidth.

5. Video Standards

5.1. Outline of Video Standards

Video standards consist mainly of a video format, a compression algorithm, and a data frame structure. There are two major kinds of analog TV system today:

- the 525/60 system (NTSC), and
- the 625/50 system (PAL, SECAM).

These two systems also form the basis for digital systems. To fill the gap of this two different system, a common video format is important, the so-called Common Intermediate Format (CIF). The Common Intermediate Format or CIF is the only existing video format that is globally recognized. The quarter CIF (QCIF) is the corresponding standardized low resolution format.

CIF is suitable for the usage at a bitrate 384 Kb/sec, or higher; while QCIF is suitable at a bitrate 64 Kb/sec or 128Kb/sec.

Compression algorithms that currently applied in video standards are

- discrete cosine transform (DCT),
- motion compensation (MC),
- Interframe Interpolation, and

- variable length coding (VLC).

More efficient compression algorithms generally cause more coding delay. Reasonably low delay is required in the realtime video applications such video telephone and video conferencing. On the other hand, the requirement for coding delay is much less severe in the video storage applications such as video in compact disk or laser disk. Bi-directional interframe interpolation contributes greatly to high compression rate, although it increases coding by an order of several 100 msec. Data frame structure is necessary to find the start point and the break point of video data. In video telephone/conferencing, the frame structure also specifies a way to multiplex video, audio and control data.

5.2. International Video Standards

(1) MPEG-1 and H.261 JTC1 of ISO/IEC and CCITT SG15 cooperated to set the motion video standards called MPEG-1 and H.261. These video standards can be applied to a digital video at a bitrate of 2 Mb/sec, or lower. The main purpose of MPEG-1 is the video storage in the compact disk. The bi-directional interframe interpolation makes MPEG-1 more efficient than H.261. H.261, in turn, is suitable for video telephone and video conferencing. Both, MPEG-1 and H.261 are based on CIF.

(2) JPEG is the standard for full color still image. JPEG has been developed by JTC 1 of ISO/IEC. JPEG can be applied very widely from very fine color photographs to freeze frame video. Lossless coding of full color images can be done using JPEG. In the computer applications of motion video, JPEG is attractive because every frame of motion video can be independently encoded, and also decoding can start at any frame. In this kind of application, JPEG is frequently called Motion-JPEG or MJPEG.

(3) For MPEG-2 and H.26X, JTC 1 of ISO/IEC and CCITT SG15 are continuing their cooperation to set video standard for high quality motion video. The targeted quality is equivalent to, or even higher, than broadcast video quality. The necessary bitrate is generally believed to be somewhere between 5 and 10Mb/sec. Scalable motion video is one of the most important features The standard is tentatively called MPEG-2 at the JTC-1 side and H.26X at the CCITT side; however they are believed to be defined identically.

5.3. Industry Video Standards

Several industry motion video standards are in use in multimedia workstations and packaged motion video.

(1) DVI (Digital Video Interactive) is INTEL's proprietary scheme. Encoding needs relatively high processing power, because DVI is mainly used for packed video. Thus, the encoding is done off-line using super computers. While DVI took a pioneering role, it will not be supported in the future.

(2) Apple Computers have introduced the proprietary Roadpizza compression scheme as part of their Quicktime software. Roadpizza is a fully scalable, highly asymmetric software codec, that supports the continuous image resizing and partial decoding of the bitstream. Image quality is still very low on the existing platforms, but it is getting better as new, more powerful computer are introduced.

(3) Fluent Inc. has developed a scalable compression scheme based on MJPEG that is intended for network applications. In the early product the receiving side monitors whether it receives frames too late and provides a massage back to the sender to reduce the frame rate to be sent.

5.4. Future Role of Video Standards

Video standards enable various binds of interworking. In realtime video applications, such as video telephone and video conferencing, terminals based on the international video standards can be interconnected at an international scale. The package media such as compact disc (CD) can be restored by any CD player in such a way that audio CD can be played in any part of the world. In the computer applications, standard video data can be interchanged through LAN and/or wide area networks. However, in the computer applications, software-only coding and decoding are frequently discussed. As computers become more powerful, software-only coding and/or decoding attracts more attention.

The role of video standards might change. Suppose that coding and decoding process can be written using a standard software language, video decoding software may be able to be transmitted prior to transmitting video data. Then any video data can be decoded. In such a situation, standardized video compression algorithms may not be necessary; scalable video formats standard may become more important.

6. Video and Its Relationship to Other Media

6.1. Synchronization with Audio

The main interdependence between video and audio results from the fact that typical videos have a soundtrack accompanying them. The synchronization of video with audio can be achieved in different ways. Depending on the granularity of synchronization there are two possibilities:

(1) start/stop synchronization
(2) frame synchronization

The start/stop synchronization defines a maximum tolerance between the start (and stop) of two media S1 and S2. The starts of the two media are synchronized by a trigger T1, which will be generated after the start of S1 and triggers the start of S2. The frame synchronization generates triggers for each frame. The triggers Ti can refer to

(a) relative next frame
(b) absolute frame number
(c) global time (GMT, CET, etc)

In cases (a) and (b) the application uses an internal unique time. In case (c) the application refers to a global time system and an adjusting of the internal times (clocks) of each workstation/partner side is necessary. This could be done by handshaking to figure out the delay between source and sink, and sending of time stamps. Independent of the way of synchronization the interleaving of the audio and video data streams is useful to realize the synchronization.

For some multimedia applications in the domain of presenting documents the start/stop synchronization is sufficient, because of lower requirements. For example, in most MM applications video and audio come from the same source and the delay between the starts of audio and video will be small ($d < 150$ msec.). In conference applications there should be a global time synchronization and the requirement is to try to synchronize as closely as possible. For lip synchronization a frame synchronization is necessary.

6.2. Still Images / Documents

One video frame could be understood as one still image, and a still image could be understood as an image sequence of length 1 (one frame). The mathematical model for video images and still images is the same, except for the time parameter. This implies that image editing is the same as video content editing. But there are big differences in usage, applications, quality and compression. One example are the different viewing distances, when perceiving a video or a still image. Usually is a distance of 1 x diagonal length for still images and 4-6 times the diagonal length for video (i.e., contours are sharper viewable on motion pictures).

This will have an impact on user interfaces for workstation integrated video. In this context the distinction of still images into two classes is important:

(1) scanned images/photographic material
(2) binary documents/object-oriented graphics

Documents typically have a higher resolution than photographic material and video. For photographic material and video there can be used lossy compression algorithms. For video MPEG or M-JPEG was recommended, for photographics JPEG is useful (see above). For documents a lossless compression is needed. Possible standards are JBIG, G3/G4 FAX and TIFF. It was mentioned, that for some applications lossy compression is reasonable, but for legal reasons lossless compression is required (i.e., X-ray pictures). Scalability is important also for archiving still images (photographics).

7. Conclusions

In this report, we have summarized the results of discussions of a working group on "Video Integration" at the 1992 Dagstuhl Workshop on Multimedia System Architectures and Applications. Starting from a few prototypical applications, we have grouped applications into four classes. These application scenarios are

(1) broadcast-type,
(2) interactive retrieval,
(3) editing, and
(4) conferencing.

We have identified typical requirements with respect to image quality, delay, and information access. It turns out that scalability of the video representation is highly desirable for all application classes. This requirement arises from both the wish to resize video windows on the computer screen and the varying bandwidth limitations in computer systems. Software-only codecs will become feasible in the future, providing another strong motivation for scalability. We concluded that scalability needs to be supported from the maximum bandwidth down to zero.

Unfortunately, existing motion video compression standards such as CCITT H.261 or ISO MPEG-1 do not support scalability. Therefore, manufacturers are introducing their own proprietary schemes. M-JPEG, the extension of the intraframe coding scheme JPEG to video, has several desirable features, its compression ratio, however, is clearly inferior to interframe schemes. Future programmable platforms will probably support multiple compression algorithms. While CPUs powerful enough for realtime compression/decompression are expected within 5 years, it is not clear whether other bottlenecks in the computer architecture might possibly delay high-quality software-only codecs further.

For the integration of motion video into computers, we have to keep in mind its relationship to other media. Video is usually accompanied by audio, and provisions are required to synchronize these data streams. Still images and binary documents are media forms closely related to motion video. While there is a continuum between a fast slide show of still images and a low frame rate movie, still and moving images have typically very different requirements such that a separate treatment is the likely solution for the near future. Spatial scalability is important both for moving images and for still images.

In summary, there are several open research questions. The current compression standards for motion video are not satisfactory for computer applications. Most of the research questions are centered around the issue of scalable video. Scalable video is the key to fully integrate motion video into computers.

Participants

Jerker Andersson
Infologics AB
Box 91
19122 Sollentuna
Sweden
E-mail: jerker.andersson@infolog.se
Tel: +46-8-922000, Fax: +46-8-960846

Friedrich Augenstein
Institut für Informatik
Universität Freiburg
Rheinstraße 10-12
7800 Freiburg
Germany
E-mail: augenste@fidji.informatik.uni-freiburg.de
Tel: +49-761-203-4860, Fax: +49-761-203-3889

Jay Bolter
School of Literature, Communication, and Culture
Georgia Institute of Technology
Atlanta, GA 30332-0165
USA
E-mail: jay.bolter@lcc.gatech.edu
Tel: +1-404-894-2735, Fax: +1-404-853-0373

Martin Brenner
Abt. ZFE ST SN 71
Siemens AG
Otto-Hahn-Ring 6
8000 München 83
Germany
E-mail: brenner@zfe.siemens.de
Tel: +49-89-636-2825, Fax: +49-89-636-48000

Victoria Burrill
c/o Zentrum für Graphische Datenverarbeitung
Wilhelminenstraße 7
6100 Darmstadt
Germany
Tel: +49-6151-155-229

José Encarnação
Fachbereich Informatik
TH Darmstadt
Wilhelminenstraße 7
6100 Darmstadt
Germany
E-mail: jle@igd.fhg.de
Tel: +49-6151-155-130, Fax: +49-6151-155-199

Participants

James Foley
GVU Center
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332
USA
E-mail: jim.foley@gvu.gatech.edu
Tel: +1-404-853-0672, Fax: +1-404-853-0673

Martin Frühauf
Fraunhofer-Institut für Graphische Datenverarbeitung
Wilhelminenstraße 7
6100 Darmstadt
Germany
E-mail: fruehauf@igd.fhg.de
Tel: +49-6151-155-231, Fax: +49-6151-155-299

Bernd Girod
Kunsthochschule für Medien Köln
Peter-Welter-Platz 2
5000 Köln
Germany
E-mail: girod@khm.uni-koeln.de
Tel: +49-221-201-8931, Fax: +49-221-201-8917

Ralf Guido Herrtwich
IBM ENC
Tiergartenstraße 8
6900 Heidelberg
Germany
E-mail: rgh@dhdibm1.bitnet
Tel: +49-6221-404-218, Fax: +49-6221-404-450

Wolfgang Herzner
Hauptabt. Informationstechnik
Forschungszentrum Seibersdorf
2444 Seibersdorf
Austria
E-mail: herzner@zdvaxa.una.ac.at
Tel: +43-2254-80-3121, Fax: +43-2254-80-2118

Georg Rainer Hofmann
Fraunhofer-Institut für Graphische Datenverarbeitung
Wilhelminenstraße 7
6100 Darmstadt
Germany
Tel: +49-6151-155-140

Participants

Christoph Hornung
Fraunhofer-Institut für Graphische Datenverarbeitung
Wilhelminenstraße 7
6100 Darmstadt
Germany
E-mail: hornung@igd.fhg.de
Tel: +49-6151-155-234, Fax: +49-6151-155-299

Wolfgang Hübner
Zentrum für Graphische Datenverarbeitung
Wilhelminenstraße 7
6100 Darmstadt
Germany
Tel: +49-6151-155-229

Takahiko Kamae
NTT Human Interface Laboratories
Nippon Telegraph and Telephone Corporation
1-2356 Take Yokosuka-Shi
Kanagawa 238-03
Japan
Tel: +81-468-59-2000, Fax: +81-468-55-1160

Kenneth Knoespel
School of Literature, Communication, and Culture
Georgia Institute of Technology
Atlanta, GA 30332-0165
USA
E-mail: kk9@prism.gatech.edu
Tel: +1-404-894-2730, Fax: +1-404-853-0373

Klaus Meyer-Wegener
Institut für Mathematische Maschinen und Datenverarbeitung
Friedrich-Alexander-Universität Erlangen Nürnberg
Martensstraße 3
8520 Erlangen
Germany
E-mail: meyer-wegener@informatik.uni-erlangen.de
Tel: +49-9131-85-7885, Fax: +49-9131-32090

Max Mühlhäuser
Institut für Telematik
Universität Karlsruhe
Kaiserstraße 12
7500 Karlsruhe
Germany
E-mail: max@telematik.informatik.uni-karlsruhe.de
Tel: +49-721-608-4790, Fax: +49-721-606-097

Participants

Meinrad Niemöller
Abt. ZFE ST SN 72
Siemens AG
Otto-Hahn-Ring 6
8000 München 83
Germany
E-mail: meinrad@zfe.siemens.de
Tel: +49-89-636-45504, Fax: +49-89-636-48000

Stefan Noll
Fraunhofer-Institut für Graphische Datenverarbeitung
Wilhelminenstraße 7
6100 Darmstadt
Germany
E-mail: noll@igd.fhg.de
Tel: +49-6151-155-213, Fax: +49-6151-155-299

G. Scott Owen
Hypermedia and Visualization Laboratory
Department of Mathematics and Computer Science
Georgia State University
Atlanta, GA 30303
USA
E-mail: owen@siggraph.org
Tel: +1-404-651-2245, Fax: +1-404-651-2246

Ralf Steinmetz
IBM European Networking Center
Postfach 10 30 68
6900 Heidelberg 1
Germany
E-mail: steinmet@dhdibm1.bitnet
Tel: +49-6221-59-3300, Fax: +49-6221-59-4280

Bernhard Tritsch
Fraunhofer-Institut für Graphische Datenverarbeitung
Wilhelminenstraße 7
6100 Darmstadt
Germany
E-mail: tritsch@igd.fhg.de
Tel: +49-6151-155-235, Fax: +49-6151-155-299

Kaisa Väänänen
Zentrum für Graphische Datenverarbeitung
Wilhelminenstraße 7
6100 Darmstadt
Germany
E-mail: kaisa@igd.fhg.de
Tel: +49-6151-155-244, Fax: +49-6151-155-299

Participants

Ana Sofia Vieira
Departamento de Matemática
Universidade de Coimbra
Apartado 3008
3000 Coimbra
Portugal
E-mail: teixeira@matuc2.mat.uc.pt
Tel: +351-39-28097, Fax: +351-39-32568

Frank Marcus Weller
FB Informatik
AG Hagen
Universität Kaiserslautern
Postfach 30 49
6750 Kaiserslautern
E-mail: fweller@informatik.uni-kl.de

Zuletzt erschienene und geplante Titel:

K. Compton, J.E. Pin , W. Thomas (editors):
Automata Theory: Infinite Computations, Dagstuhl-Seminar-Report; 28, 6.-10.1.92 (9202)

H. Langmaack, E. Neuhold, M. Paul (editors):
Software Construction - Foundation and Application, Dagstuhl-Seminar-Report: 29, 13..-17.1.92 (9203)

K. Ambos-Spies, S. Homer, U. Schöning (editors):
Structure and Complexity Theory, Dagstuhl-Seminar-Report; 30, 3.-7.02.92 (9206)

B. Booß, W. Coy, J.-M. Pflüger (editors):
Limits of Modelling with Programmed Machines, Dagstuhl-Seminar-Report; 31, 10.-14.2.92 (9207)

K. Compton, J.E. Pin , W. Thomas (editors):
Automata Theory: Infinite Computations, Dagstuhl-Seminar-Report; 28, 6.-10.1.92 (9202)

H. Langmaack, E. Neuhold, M. Paul (editors):
Software Construction - Foundation and Application, Dagstuhl-Seminar-Report; 29, 13.-17.1.92 (9203)

K. Ambos-Spies, S. Homer, U. Schöning (editors):
Structure and Complexity Theory, Dagstuhl-Seminar-Report; 30, 3.-7.2.92 (9206)

B. Booß, W. Coy, J.-M. Pflüger (editors):
Limits of Information-technological Models, Dagstuhl-Seminar-Report; 31, 10.-14.2.92 (9207)

N. Habermann, W.F. Tichy (editors):
Future Directions in Software Engineering, Dagstuhl-Seminar-Report; 32; 17.2.-21.2.92 (9208)

R. Cole, E.W. Mayr, F. Meyer auf der Heide (editors):
Parallel and Distributed Algorithms; Dagstuhl-Seminar-Report; 33; 2.3.-6.3.92 (9210)

P. Klint, T. Reps, G. Snelting (editors):
Programming Environments; Dagstuhl-Seminar-Report; 34; 9.3.-13.3.92 (9211)

H.-D. Ehrich, J.A. Goguen, A. Sernadas (editors):
Foundations of Information Systems Specification and Design; Dagstuhl-Seminar-Report; 35; 16.3.-19.3.9 (9212)

W. Damm, Ch. Hankin, J. Hughes (editors):
Functional Languages:
Compiler Technology and Parallelism; Dagstuhl-Seminar-Report; 36; 23.3.-27.3.92 (9213)

Th. Beth, W. Diffie, G.J. Simmons (editors):
System Security; Dagstuhl-Seminar-Report; 37; 30.3.-3.4.92 (9214)

C.A. Ellis, M. Jarke (editors):
Distributed Cooperation in Integrated Information Systems; Dagstuhl-Seminar-Report; 38; 5.4.-9.4.92 (9215)

J. Buchmann, H. Niederreiter, A.M. Odlyzko, H.G. Zimmer (editors):
Algorithms and Number Theory, Dagstuhl-Seminar-Report; 39; 22.06.-26.06.92 (9226)

E. Börger, Y. Gurevich, H. Kleine-Büning, M.M. Richter (editors):
Computer Science Logic, Dagstuhl-Seminar-Report; 40; 13.07.-17.07.92 (9229)

J. von zur Gathen, M. Karpinski, D. Kozen (editors):
Algebraic Complexity and Parallelism, Dagstuhl-Seminar-Report; 41; 20.07.-24.07.92 (9230)

F. Baader, J. Siekmann, W. Snyder (editors):
6th International Workshop on Unification, Dagstuhl-Seminar-Report; 42; 29.07.-31.07.92 (9231)

J.W. Davenport, F. Krückeberg, R.E. Moore, S. Rump (editors):
Symbolic, algebraic and validated numerical Computation, Dagstuhl-Seminar-Report; 43; 03.08.-07.08.92 (9232)

R. Cohen, R. Kass, C. Paris, W. Wahlster (editors):
Third International Workshop on User Modeling (UM'92), Dagstuhl-Seminar-Report; 44; 10.-13.8.92 (9233)

R. Reischuk, D. Uhlig (editors):
Complexity and Realization of Boolean Functions, Dagstuhl-Seminar-Report; 45; 24.08.-28.08.92 (9235)

Th. Lengauer, D. Schomburg, M.S. Waterman (editors):
Molecular Bioinformatics, Dagstuhl-Seminar-Report; 46; 07.09.-11.09.92 (9237)

V.R. Basili, H.D. Rombach, R.W. Selby (editors):
Experimental Software Engineering Issues, Dagstuhl-Seminar-Report; 47; 14.-18.09.92 (9238)

Y. Dittrich, H. Hastedt, P. Schefe (editors):
Computer Science and Philosophy, Dagstuhl-Seminar-Report; 48; 21.09.-25.09.92 (9239)

R.P. Daley, U. Furbach, K.P. Jantke (editors):
Analogical and Inductive Inference 1992 , Dagstuhl-Seminar-Report; 49; 05.10.-09.10.92 (9241)

E. Novak, St. Smale, J.F. Traub (editors):
Algorithms and Complexity for Continuous Problems, Dagstuhl-Seminar-Report; 50; 12.10.-16.10.92 (9242)

J. Encarnação, J. Foley (editors):
Multimedia - System Architectures and Applications, Dagstuhl-Seminar-Report; 51; 02.11.-06.11.92 (9245)

F.J. Rammig, J. Staunstrup, G. Zimmermann (editors):
Self-Timed Design, Dagstuhl-Seminar-Report; 52; 30.11.-04.12.92 (9249)

B. Courcelle, H. Ehrig, G. Rozenberg, H.J. Schneider (editors):
Graph-Transformations in Computer Science, Dagstuhl-Seminar-Report; 53; 04.01.-08.01.93 (9301)

A. Arnold, L. Priese, R. Vollmar (editors):
Automata Theory: Distributed Models, Dagstuhl-Seminar-Report; 54; 11.01.-15.01.93 (9302)

W. Cellary, K. Vidyasankar , G. Vossen (editors):
Versioning in Database Management Systems, Dagstuhl-Seminar-Report; 55; 01.02.-05.02.93 (9305)

B. Becker, R. Bryant, Ch. Meinel (editors):
Computer Aided Design and Test , Dagstuhl-Seminar-Report; 56; 15.02.-19.02.93 (9307)

M. Pinkal, R. Scha, L. Schubert (editors):
Semantic Formalisms in Natural Language Processing, Dagstuhl-Seminar-Report; 57; 23.02.-26.02.93 (9308)

H. Bibel, K. Furukawa, M. Stickel (editors):
Deduction , Dagstuhl-Seminar-Report; 58; 08.03.-12.03.93 (9310)

H. Alt, B. Chazelle, E. Welzl (editors):
Computational Geometry, Dagstuhl-Seminar-Report; 59; 22.03.-26.03.93 (9312)

J. Pustejovsky, H. Kamp (editors):
Universals in the Lexicon: At the Intersection of Lexical Semantic Theories, Dagstuhl-Seminar-Report; 60; 29.03.-02.04.93 (9313)

W. Straßer, F. Wahl (editors):
Graphics & Robotics, Dagstuhl-Seminar-Report; 61; 19.04.-22.04.93 (9316)

C. Beeri, A. Heuer, G. Saake, S.D. Urban (editors):
Formal Aspects of Object Base Dynamics , Dagstuhl-Seminar-Report; 62; 26.04.-30.04.93 (9317)