ABSTRACT

VideoSTAR (Video STorage And Retrieval) is an experimental database framework for video information management. The development of this framework has been motivated by the problems users of traditional video tape archives are facing and by the multimedia technology that makes integrated, digital video archives feasible. Specifically, VideoSTAR addresses issues related to sharing of video information among different users and tools.

Video information is a semantically rich and complex concept. In this thesis we define 9 different classes of meta-data that might be stored in a video database, and we discuss how different types of video database applications rely on the classes of meta data. The VideoSTAR framework includes a data model describing how media and meta-data can be represented. The data model captures generic characteristics of video information such as video document composition, video document structures, and video content indexing. This generic model can be used as a basis for developing application domain specific models.

In this thesis we propose a video query algebra that can be used for querying and browsing video information. This algebra provides operations for utilizing compositional data and temporal relationships in end-user queries. The compositional operations allow end-users to control the degree of meta-data sharing in a shared video information database.

We have implemented prototype tools for integrated video archives on top of the VideoSTAR framework. These tools have been used in three different end-user experiments for evaluating the VideoSTAR framework and for evaluating digital video archive environments. These experiments have shown that video archive end-users appreciate the opportunities for having direct access to large collections of digital video from their own desks.

CONTENTS

PREFACE

This thesis is submitted to the Norwegian Institute of Technology for the doctor degree "doktor ingeniør". The work has been carried out in the Database Systems Group under the supervision of Professor Kjell Bratbergsengen.

Context of Work

The research being carried on in relation to this thesis was started by the author as a one-person project. Throughout the years, several students and researchers have joined this project. All members of the VideoSTAR project have worked closely together contributing to the quality of the resulting prototype. The author of this thesis has been the driving force in the VideoSTAR project but most of the results reported in this thesis have been achieved jointly with other members of the project.

Several users in two of the television companies in Norway, NRK and TV2, and Norwegian Folk Museum have given valuable input to the VideoSTAR project. Since 1994 the VideoSTAR project has been a part of the Norwegian LAVA project which receives funding from Uninett and the Norwegian Research Council in addition to the end-user organizations.

Acknowledgments

I want to thank my advisor, Professor Kjell Bratbergsengen, for giving me the opportunity and freedom to initiate and conduct the VideoSTAR activities. The project would not have been started if I had not received a scholarship from the Norwegian Institute of Technology and an invitation from Kjell to join the database group to study multimedia databases.

The VideoSTAR project has been the environment for my work and has meant a lot to me. Most of all, I wish to thank Assistant Professor Roger Midtstraum for the fruitful teamwork that has lasted for four years. The work would not have been the same without his participation. I also wish to thank Olav Sandstå and Stein Langørgen for their creative participation in the project. A number of diploma students have being part of the project throughout the years. I want to thank them all.

I have met a number of people via the VideoSTAR project who has helped me to understand the challenges that video archive users are facing. Especially, I want to thank Tedd Johansen at NRK, Hilde Brekken at TV2, and Trond Bjorli at Norwegian Folk Museum.

During these years, my wife, Tove, gave me all the support and love I needed, and my two children, Siri and Sverre, helped me understand the real values of life.

1 INTRODUCTION

...we now have routines for audiovisual archiving established – say – around 1980, whereas the sound and visions broadcasts prior to that time are preserved to a much lesser degree and in no systematic fashion. Because of this, society has lost an important part of its memory.

Hans Fredrik Dahl, Keynote Speech, JTS 95:
Technology and our Audio-Visual Heritage, 1995 [Dah95]

Since the last half of the 1890s when Georges Méliès took his film camera out to *capture slices of life on the streets* [Ell90], huge amounts of film and video describing life on earth have been recorded. A survey performed by Library of Congress [Gib94] reports more than 1.6 million hours of film and video stored in the large film and television archives in the world. Based on this survey, it is predicted [Sch94] that there exists more than 6 million hours in archives all over the world, and the growth is estimated to be about 10 percent a year.

Much of this information is unique and important to individuals, to individual organisations, and to our whole society. Thus, it is important that video and film information can be made accessible to end-users when needed. It is also important that users have appropriate tools for taking care of this type of information in consistent and safe ways to avoid that this valuable information is getting destroyed. Due to the lack of such tools, valuable video and film information is already lost.

1.1 CHALLENGES IN FILM AND VIDEO MANAGEMENT

Computer technology plays an important role in making stored information available and searchable for users. Technologies for computerised management of numeric and textual data have been available for several years: *Database systems*, for instance, have been developed to allow end-user data to be integrated and shared in safe ways [Dat86]. *Information retrieval* [Sal88] has in similar ways been developed to allow users search large collections of unformatted texts.

In addition to *media data* – i.e., film and video data – film and video archives also have to manage *meta-data* related to film reels and video cassettes. Up to the last few years, film and video has been recorded, edited and stored in analogue formats. Hence, film and video data have not been available in formats suitable for computerised management.

Meta-data, on the other hand, have been target for computerisation for some years [Tur90]. Traditionally, meta-data have been written on textual catalogue cards stored in manual card archives. These card archives are now mostly being replaced by computerised databases but, still, film and video data themselves are not put under computer control. The problems in film and video information management have arisen from both the fact that film and video data are stored as analogue data, *and* that it is not possible to develop a computerised system having integrated control over both media and meta-data. The following subsections are based on the discussion in [LH91].

Preservation

Analogue data have the bad property that copying adds noise to the original data. One would, thus, want to use the original copy as much as possible. But this will result in more tear, wear, and damages to the original copy. From a preservation point of view, it is very difficult to control the quality of film and video information as long as it is stored as analogue data, and it is desirable to convert analogue video data into digital formats [Kol94].

Ease of Access

Users have to have a physical copy of the video tape for viewing and, thus, they have to be located nearby the archive to have easy access. For nonresident users, the solution is either to spend the time needed for travelling to the site of the archive or to wait a day or so for a copy of the video tape(s) to be delivered by mail or courier. This is especially a problem in distributed organisations such as most larger television companies.

Ease of Retrieval

Searching for video items in the archive may often be a time-consuming process: The process is opened by a search in the meta-data database which results in a list of references to video tapes in the archive. The user has to take these tapes out of the archive and view them sequentially to judge if the content is of any interest. The retrieval process is also a time-consuming one due to the fact that the that the content of video cassettes and film reels is not completely described in the meta-data database. In practice, users have to spend a lot of time browsing the film and video archive to compensate for the lack of meta-data [Whi94b].

Integrity and Consistency

Due to the fact that film and video data as such are not under computerised control, manual routines are needed to ensure data integrity and consistency. Archives experience that manual routines fail time to time: video tapes are not returned after being taken out of the archives and, in some cases, old films have been damaged because users have cut out parts of the original film to be pasted in new productions [Hol94].

Sharing and Reuse

Today, reuse and sharing of pieces of video means that video data has to physically copied onto the new video. Thus, video that is used in several contexts exist in one replica for each context into which it has been used. Today, information about this reuse and replication of video data has to be manually entered. Often, this is not done, and it is then difficult for end-users to retrieve all video items sharing a common portion of video data and to find how these items relate to each other.

Advances in computer technology known as *multimedia* technology have resulted in technology for handling digital video data. Before going into a description of the aim of this thesis, we will give a brief introduction to the area of multimedia technology into which digital video and video database systems belong.

1.2 MULTIMEDIA – WHAT'S BEHIND THE HYPE?

Multimedia is one of the hype-words of the 90's. Going to a computer exhibition is an overwhelming experience of sounds and visual effects. Some authors go behind the hype and investigate what multimedia really is. Buford is one of the researchers who have expressed the needs for *"a clear statements as to the rationale, issues, and directions facing the development of global multimedia information and communication systems, ..."* [Buf94a]. The purpose of this section is to focus on some important properties of multimedia technology. Buford [Buf94a] gives a more detailed discussion of multimedia systems and technology.

The composite term *multi media* indicates that it is the simultaneous use of data in different media forms – e.g., text, audio and video – that is called multimedia. Multimedia technology has several implications such that:

Computer controlled, digital data: Integration of data of different forms in practice assumes that these data are available in digital formats and that these data are put under computer control. Hence, multimedia systems usually means that the user have direct access to the information. Multimedia systems often provides advanced tools for data manipulation because data are available in digital formats.

"New" data types: The term "multimedia" is often used as a synonym for technologies involving continuous media forms such as audio and video. Continuous media are the most demanding of the "new" medias in terms of storage, communications and processing requirements. Continuous media constitute a *stream* of data elements where there are predefined temporal relations and constraints between the elements in the stream. Digital video systems has been available for some time: Macintosh was the first manufacturer to give users inexpensive access to digital video via the QuickTime system [App93]. During the last couple of years, courses and conferences have been broadcast in real time to users all over the world via Internet using video conferencing tools such as IVS (INRIA Videoconferencing System) or NV (Network Video) [Adi93].

Convergence of technologies: Computing, communications, publishing and mass media/consumer entertainment are areas that traditionally have been separated from each other. Multimedia represents a convergence of technologies in these areas. Thus, problems that earlier were specific for one area – e.g., copyright issues in the publication area – now also is a concern in other areas – e.g., what mechanisms to add into communications technology to protect copyrighted information from being misused. A result of this convergence is that new coalitions are established between computer manufacturers, software houses, cable or telephone network operators, and information providers. The project between Time Warner Cable, Silicon Graphics International, AT&T Network Systems, and Scientific-Atlanta to establish video-on-demand services in Orlando [SG94b] may serve as an example of this convergence.

User interfaces: User interfaces to multimedia information systems have a larger repertoire of interface primitives – e.g., speech based interfaces – than traditional information systems. In addition, multimedia information systems have the opportunity to integrate the different interface primitives in powerful ways. In the "multimedia office" the computer is an integrated tool for making multimedia documents, making telephone calls, setting up video conferences and for searching multimedia information stores. The computer will accept input from the keyboard but it may also be able to recognise the user's speech. Other researchers [BD92] have studied multimedia interface design thoroughly.

To refer Grosky, multimedia information system design is *"in its infancy, with no agreement on how to proceed"* [Gro94]. Grosky considers data model design, query processing, browsing support, and multimedia composition and presentation to be the more important issues to study to turn multimedia information design into a mature field.

1.3 WHY VIDEO DATABASES?

Multimedia technology makes digital video storing and processing possible: Video data can be recorded digitally [Son94]; digital video data can be compressed to manageable amounts [Lio91, Gal91]; and digital video editing systems such as AVID Media Composer [Avi93] allow users to combine pieces of digital video into new digital video items. As we will discuss in more details in Chapter 3, digital video can improve the situation for users having collections of video:

1. Information quality can better be preserved when stored as digital data because copies can be made without reducing video quality and because error-checking codes can be used to supervise the quality of the video.
2. The accessibility of video data can be improved because digital video can be transferred and accessed by means of modern data and telecommunications networks.
3. In a digital domain, the need for video data replication can be reduced when video documents are stored as *virtual video documents* [MD89] instead of physically copying video data onto a new video cassette.

However, digital video technology alone does not improve all aspects of the situation today. To address problems related to consistent sharing, reuse and retrieval of video information, one has to develop systems that take integrated control of video and meta-data management. This has been the traditional motivation for choosing database systems; database technology has evolved because users want to share data in consistent and secure ways. Date lists a number of reasons for having database systems for storing information in a multiuser environment [Dat86]:

1. *The data can be shared.*
2. *Standards can be enforced.*
3. *Data independence can be provided.*
4. *Security can be applied.*
5. *Redundancy can be reduced.*
6. *Inconsistency can be avoided.*
7. *Integrity can be maintained.*
8. *Information can be made available on demand.*
9. *Conflicting requirements can be balanced.*

These arguments for using databases are concerned with information in general and are valid for video information as well as traditional record-based data. Take television news archives as an example. Reporters, directors, librarians, and other users will request shared access to video and meta-data stored in the archive in a secure and consistent way where the amount of redundant video data is kept to a minimum and where users need not worry about video data formats – i.e., data independence is provided.

1.4 AIMS AND METHODS

In this chapter we have argued that video information management in a multiuser environment is concerned with many of the problems DBMS'es traditionally have been addressing. However, video information is different than traditional types of information, and there is a need to develop new DBMS functionality for video information management in shared environments. This has been the motivation for this thesis and its objectives which have been:

- To identify generic characteristics of video information and video database applications that should be supported by future DBMS'es for video information management.
- To develop functionality that can be included in future DBMS'es for allowing users to share video information in well-defined ways.

Video *information* is a semantically richer concept than video data. When using the term video information we will include video (media) data *and* necessary meta-data needed to interpret the video documents and their contents – e.g., to describe the composition and contents of video documents. This thesis addresses central issues of video *information* management in shared environments. It identifies semantics that are essential to video information management in general, but more important: The thesis focuses on how a video database can take advantage of video information semantics in providing support for users and applications sharing video information. The research issues dealt with in this thesis are:

1. *Video information semantics.* The question that we address is: What video information semantics should be supported by a future DBMS for video information management – i.e., what are the important and general characteristics of video information?
2. *Video information modelling* is an important issue when multiple users and applications access a shared video database because users need to share a common interpretation of the contents of the video database. The question that we address is: How can these semantics be captured in a generic data model that can be extended and tailored to the needs of specific users and application domains?
3. *Video information management.* The question that we address is: How can video information semantics be utilised by a video database framework for

supporting users in entering, managing, querying, and viewing the various types of media and meta-data in a consistent way?

4. *Querying and browsing in shared environments.* Querying and browsing are complementary methods for providing content-based access. In this thesis we will particularly address the use of compositional data in querying and browsing because compositional data define how video documents share media data. The question we address is: How can compositional data be used to control the degree of meta-data sharing in a shared video database during querying and browsing?

5. *Querying temporal relationships.* Temporal properties distinguish video data from traditional types of data. The question we address is: What query operation can be defined to allow users exploit temporal relationships in video information querying?

The best way to evaluate the proposed functionality is to implement a prototype DBMS for video information management and develop video database tools operating on this DBMS. However, such an implementation would be too ambitious for a dr.ing. work. As an alternative, we have implemented a video database framework called VideoSTAR (Video STorage And Retrieval) and some prototype video archive tools. The VideoSTAR environment allows us to evaluate the proposals and to perform end-user experiments, but it is not a full-fledged DBMS. It is, therefore, necessary to evaluate how relevant the results achieved in developing VideoSTAR are for the development of future DBMS'es.

1.5 OUTLINE OF THE THESIS

This thesis is logically divided into three parts. The first part consists of Chapters 2 through 4. This part is related to our first research question and gives an overall introduction to video information semantics and management:

- *Chapter 2* gives an introduction to video information, its important characteristics and applications.
- Digital video issues such as video formats and compression methods are reviewed in *Chapter 3*.
- In *Chapter 4* we review the state-of-the-art in the area of video databases.

The second part consists of Chapters 5 through 9. This part summarises and discusses the contributions of this thesis:

- *Chapter 5* briefly presents the VideoSTAR video database framework in response to our third research question.
- The VideoSTAR framework provides a generic data model for video information modelling in response to our second research question. This model is discussed in *Chapter 6*.
- In *Chapter 7* we discuss querying and browsing issues in response to our fourth and fifth research questions.
- The VideoSTAR framework has been used a conceptual basis for a video archive prototype. *Chapter 8* discusses implementation and end-user experiences made in relation to this prototype.
- *Chapter 9* concludes the thesis and outlines directions for further work.

The third and last part of this thesis consists of Appendices A through F. This part is a compilation of papers that go into more details than the second part:

- The paper "Video Information Contents and Architecture" [Hje94c], presented at the 4th International Conference on Extending Database Technology, is included as *Appendix A*.
- The paper "Modelling and Querying Video Data" (Hjelsvold et al. [HM94]), presented at the 20th Conference on Very Large Data Bases, is included as *Appendix B*.
- The paper "Databases for Video Information Sharing" (Hjelsvold et al. [HM95]), presented at the Storage and Retrieval for Image and Video Databases III, is included as *Appendix C*.
- The paper "A Temporal Foundation of Video Database" (Hjelsvold et al. [HMS95a]), presented at the International Workshop for Temporal Database Management Systems, is included as *Appendix D*.
- The paper "Searching and Browsing a Shared Video Database" (Hjelsvold et al. [HMS95b]), is included as *Appendix E*. This paper will be included in the Kluwer Academic Press book "Multimedia Database Management Systems I".
- The paper "Integrated Video Archive Tools" (Hjelsvold et al. [HLMS95]), to be presented at ACM Multimedia '95, is included as *Appendix F*.

2 VIDEO INFORMATION AND APPLICATIONS

Because film can give us such a close approximation of reality, it can communicate a precise knowledge that written or spoken language seldom can. Language systems may be much better equipped to deal with the nonconcrete world of ideas and abstractions (...), but they are not nearly so capable of conveying precise information about physical realities.

James Monaco, How to Read a Film, 1981 [Mon81]

Video is a complex type of information which we need to understand to be able to develop database support for this type of information. We also need to know the characteristics of video applications to understand what support such applications should have from a video database. The following discussion on these issues will start by reviewing the contributions that film theory have given to the understanding of film/video as a medium for information exchange.

2.1 FILM THEORY BACKGROUND

As opposite to traditional data types such as numbers, characters and graphics, video data have a temporal dimension. The very first film makers noticed that temporal composition was at least as important as spatial composition. During a series of experiments and studies in the last half of the 1920s, Pudovkin and other Soviet film makers found that the meaning of a piece of film was heavily influenced by the surrounding parts (i.e., its context) [Ell90]. They identified three main steps in film creation – time considerations play the major role in two of these steps:

Pudovkin offers a sort of formula: Film creation equals (1) what is shown in the shots, (2) the order in which they appear, and (3) how long each is held on the screen. [Ell90]

2.1.1 Scene Composition

One may think that the individual frames within a video stream are the "atoms" of video information. As noted by Monaco, however, there are no basic units of meaning in film [Mon81]. Even the individual frame may carry an almost infinite amount of visual information and is, in this sense, multidimensional.

Scene composition (in film theory called *mise en scèene*) is similar to the scene composition that a photographer has to perform. The common saying that "an image tells more than thousand words" is thus equally valid for the contents of the physical scenes shown in a video documents.

2.1.2 Temporal Composition

Much information is also carried in the temporal dimension. If a user, for instance, want to document a certain handcraft, video would be an attractive medium because the documenter is not only interested in each of the actions of this handcraft but he/she also wants to see how they are temporarily connected – e.g., the sequence and duration of the actions.

The Soviet film maker Sergei M. Eisenstein has had a great influence on the understanding of film (and video) by showing *"the fact that two film pieces of any kind, placed together, inevitably combine into a new concept, a new quality, arising out of the juxtaposition"* [Ell90]. Eisenstein illustrates this by a small example: *"For example, take a grave, juxtaposed with a woman in mourning weeping beside it, and scarcely anybody will fail to jump to the conclusion: a widow"* [Ell90].

The main knowledge to learn from this is that the information carried in a video document is not only defined by what is actually seen in the video. The way pieces of video are combined by temporal composition creates a context which influences the way a user interprets what is seen in the video.

In addition to deciding the order of each shot and their lengths, the video maker has a number of means for implementing the transitions between the shots. Some of the means are the *cut* in which the second shot directly follows the first one, the *dissolve* where the second shot is superimposed over, and gradually replaces, the first one, letting the first shot *fade out* to black while the second *fades in* to full exposure [Mil90].

Traditionally, there have been three main artistic modes of film: *narrative fiction* for telling stories, *documentary* for capturing real world events, and *experimental* for creative experiences [Ell90]. In this thesis we will mainly be concerned with the two first modes. The computer have given new and more interactive opportunities. Locatis et al. defines three different ways of presenting video information: *linear video* is the traditional approach where the entire document is the "unit" of information, *interactive video* allows the user to access smaller parts of the document in any order, and *hypervideo* breaks information down into small fragments connected via hyperlinks [LCB90]. Hypermedia is a large research area in its own, so this thesis will mainly be concerned with the two former ways of presenting information.

In film theory, the terms shot, scene, and sequence are used to define video document structure at various levels of abstraction [Fos92, Mon81]. A *shot* represents a piece of video recorded in one contiguous shot; a *scene* is often defined as *"a complete, continuous chain of actions which take place in the same place at the same time"*; a *sequence* represents *"a group of scenes linked together by a definable common thread of action"* [Fos92].

2.2 VIDEO INFORMATION

Traditional film, analogue video, and digital video represent different technologies, but they can all be considered as variants of the same data type – the motion picture data type. In this thesis we will use the term *video* in the broader sense as a collective term for all three technologies. *Video information* is not a very stringently defined term. To make the following discussion on video information sharing clearer, we will define our interpretation more precisely.

At the physical level, video can be viewed as a stream of images. This is, however, not the whole story. A video will often be a complex multimedia document composed of several different shots and audio recordings – e.g., a two hour movie with hundreds of shots and audio recordings. We will use *video document* as the term denoting a video composition. A video document is a composition defined over *media data* such as still images and audio and video data. We will use *video information* as a collective term that includes both media data and *meta-data* to be associated with the media data.

2.2.1 Video Data Characteristics

While traditional data take static values – i.e., values that do not change unless explicitly updated – the contents of a video screen will dynamically change when video data are displayed to the the user. Technically, video data can be considered as an isochronous *stream* of images – i.e., a stream of images at a constant rate, the *frame rate*. The frame rate is measured in frames per second, denoted *fps*[1].

A system managing traditional data, for instance a document previewer, does not need mechanisms for handling strict temporal constraints. If the user give a page-down command, the next page should be displayed to the user "as quick as possible". However, the data quality of the document itself is not reduced if the user has to wait for a while until the page is actually displayed. The situation is different for displaying video. A video viewer system has to adhere strictly to the frame rate, otherwise anomalies such as jitter [Fur94] will occur.

Since the advent of the sound film in the late 1920s, audio has been an integral part of film and video information. Audio data do also represent a stream of data. Audio and video streams have to be *synchronised* so that the user will experience the audio as belonging to the images. This has been an important research issue in the multimedia community for some years, [BCD94, Fur94].

2.2.2 Meta-Data

Meta-data in video databases can be classified in different ways. Grosky [Gro94] classifies meta-data as *content-based* (e.g., colour histograms) or *content-independent* (e.g., audio-video synchronisation data). Content-based data can be further divided into *information-bearing* – i.e., they convey information that is not explicitly encoded into the video data – and *non-information-bearing*. The following example may illustrate the latter distinction: Assume that a shot shows cars driving along a street. The name of this street will be information

[1] Some typical frame rates are 24 fps (film), 25 fps (PAL – the European video standard), and 30 fps (NTSC – the American/Japanese video standard).

bearing meta-data. However, if the shot also shows a sign where the name of the street is written, the street's name would be non-information-bearing.

Jain et al. differentiate between meta-data that relates to individual frames, *image features*, and to temporal sequences of frames, *video features* [JH94]. Jain et al. also makes a distinction between *raw features* (low-level domain-independent image features) and *qualitative features* (features which rely on specific domain models – e.g., the appearance of a goal in a soccer game). Böhm et al. defines six classes of meta-data. The following classification is an extension of this (though *document location metadata* is not included) [BR94]:

- *Media-specific Data.* Some meta-data is required to control playback and rendering of media data – e.g., video format, frame rate, and width and height of the video frames.
- *Compositional Data.* Compositional data define relations between video documents and media data. Traditionally, video documents are stored as one physical video and one (or two for stereo) audio tracks, even when they contain a large number of different shots. Digital video allows for *virtual video* documents [MD89] where the document is represented as compositional data defined over the media segments. The document is not materialised itself but is generated "on the fly" during replay.
- *Version Data.* Video documents – as other documents – may go through a number of revisions. Both complete documents and individual components within a document may exist in various versions.
- *Bibliographic Data.* This category describes the video document as a whole and includes information such as title, date of issue, production team, and actors/reporters contributing in the video document. (Bibliographic data is not a separate class in the classification proposed by Böhm et al. but some of the bibliographic data might go into the class "metadata for content classification".)
- *Structural Data.* Video documents are often well-structured into a structure hierarchy in similar ways as books are organised into chapters, sections, and subsections. (Structural data is not an explicit class in Böhm's classification.)
- *Content Annotations.* Content annotations are textual descriptions of the sensory content in a video. These annotations are manually entered by users and serves as indexes to the content of a piece of video. (Content annotations would be classified as qualitative features by Jain et al. and "metadata for content classification" by Böhm et al.)

- *Content Feature Data.* This category includes features that are automatically extracted from video and audio data. Such features can be used instead of – or in addition to – content annotations to provide content-based retrieval. (Content feature data would be classified as raw features by Jain et al. and "content-descriptive metadata" by Böhm et al.)
- *Topic Annotations.* The topics that are presented or elucidated in a video document are determined by the contents of the individual pieces of audio and video used in the document *and* by the way the individual pieces are combined. Content annotations can be used to describe or classify the issues being raised in video documents. (Böhm et al. would put content annotations in the same class of meta-data as content annotations.)
- *Supplementary Information.* Content and topic annotations serve as indexes to the content and topics in video. The user may want to associate other types of information to a piece of audio/video – e.g., for making personal remarks. (Böhm et al. would put supplementary information into the class "metadata for content classification".)

2.3 VIDEO INFORMATION "LIFE-CYCLE"

Video information usually goes through a rather complex process from an idea is conceived until the final version of the document is made available for viewing and stored in the archive [Hol94, Mil90]. A shared video database should be able to manage data generated by tools used in all stages of this process. Figure 2.1 illustrates the "life-cycle" of video information. This figure integrates the video information process as seen by the production team (*planning, recording, editing, and viewing*) [Mil90], the documenter (*planning, recording, analysis, and viewing*) [Smi92], and the librarian (*classification*) [Hol94].

The *planning* process can be more or less exhaustive. In the most exhaustive case, the archive is searched for documents associated with similar topics and for video shots having appropriate image contents. Often, new recordings are sketched, and document composition and structure are outlined.

The *recording* process generates new media data and corresponding media-specific data. Often, recording involves a full production team where a script is responsible for taking notes – e.g., for registering names of people being

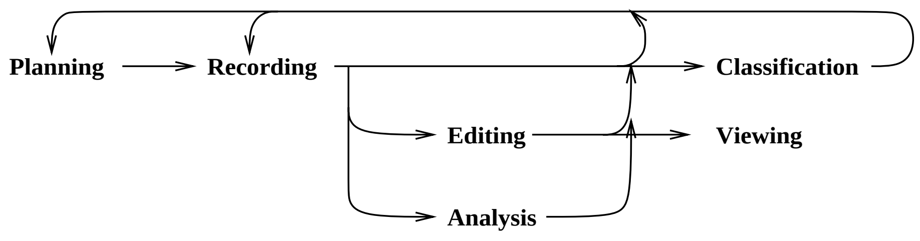

Figure 2.1 Stages in video production and use

interviewed, or for recording data such as time and location for recording, focal length, and camera techniques (zoom, tilt, or pan, for instance).

The *editing* (authoring) process combines recorded media data with media data from the stock to create a video document. Notes and sketches from the planning and recording stages might be accessed and revised. Media data might be combined to create certain visual effects that are stored as new media data. Manuscripts or other descriptions of the topic of the document may also be added.

The *classification* process supplements the work done by the production team in describing the contents and the topics being raised in a video document. This activity is usually being done by librarians who classify and organise media and meta-data to simplify retrieval.

The *viewing* process is the process where the video document is replayed under the control of the user. Video documents may offer various degrees of interactive control (see Section 2.4).

The *analysis* process aims at analysing and organising media data – e.g., for documentation. The output of the process is an analysis that might contain supplementary information related to the video data. The output might also be a "report document" where pieces of video are put together to illustrate the outcome of the analysis.

Today, most of the meta-data gathered during video document production are not stored in an organised manner, and valuable information that is known by the production team during production is lost when the document creation process ends. Users would benefit from having an integrated video database taking care of meta-data gathered throughout the whole production process.

2.4 A VARIETY OF VIDEO APPLICATIONS

A shared video database should support different types of applications in sharing of video information. In this section we will take a closer look at some classes of applications that might use video databases. The purpose of this presentation is to illustrate that a shared video database must give consideration to the needs of more than one specific application. For this purpose, we have chosen five different classes of applications:

- *Video-On-Demand services.* VOD services allow users to search for videos and movies stored on a digital video server [DVV94, HMJ93, L$^+$93, LV94, RBE94]. A typical VOD service is aimed at offering the user flexibility in choosing movies from a large set of available titles. The entire movie is presumed to be the unit of interest and, thus, selection is mainly based on bibliographic data such as title, genre, or director. When a video has been selected, it is assumed that the movie will be viewed from beginning to end with little user interaction: Users may start, stop, pause, or playback a part of the video, but they do not need functionality to skip parts of the video, change the sequence of scenes, or search for parts of the video having a specific content.
- *Interactive video applications.* Most videos and movies on VOD servers can be classified as *linear* [LCB90]. As opposite to linear video, interactive video assumes that the users may access scenes in any order. A *News-on-demand* service [MBG93] is an example of an interactive video application. The users of a news-on-demand service may want to select news items based on topic – and possibly on image contents – and may request the ability to decide in which sequence the news items should be played. Structural data are a prerequisite for offering this kind of functionality.
- *Stock-shot applications* are applications that specifically access information related to audio and video footage. Stockshot applications are especially used by film [Tur90] or television directors and television reporters to retrieve video with a certain image content – e.g., to find video shots showing a specific person or object from a television news archive.
- *Subject research applications* are applications that assist users – e.g., television reporters, mass media researchers, or historians – in finding video documents related to a specific, but not necessarily precisely defined, topic. “People’s Century” is considered to be the biggest and most ambitious

Meta-data	**VOD**	**I-video**	**Stockshot**	**Research**	**Docum.**
Compositional	No	High	No	High	Low
Bibliographic	High	High	Medium	High	Medium
Structural	Low	High	No	High	Med./high
Sensory content	No	Medium	High	High	High
Topic content	Low	High	Medium	High	High
Supplementary	No	Medium	Low	Medium	High

Table 2.1 Importance of meta-data in different applications

historical documentation series that BBC has ever undertaken [Whi94b]. During the work on the series BBC has gained substantial experience with this kind of research: Topic data are needed to retrieve a collection of television programmes that can serve as the starting point for the research. A comprehensive breakdown of the individual programmes, identifying each shot *and* its source, is needed because BBC has to acquire permission from owners and authors in order to reuse archival material. In such a large project, it is also necessary to add personal remarks to interesting shots in the archive for use in later stages of the process [Whi94b].

- *Video documentation applications* support users in using video to document aspects of the real-world. This includes, for instance, anthropology [Smi92], hand craft documentation [Mar94], and user requirements analysis [HS93]. The main difference between this class of application and the others is the strong emphasis on supplementary data; users want to attach their detailed remarks and analysis to the video data.

Table 2.1 summarises the importance of different types of meta-data for these classes of applications. (Version data are equally important to all classes and is not shown in the table.)

2.5 VIDEO DATABASES IN SHARED ENVIRONMENTS

This thesis is devoted to the issue of developing database support for video information sharing. We see two different ways that a video database can support video information sharing: First, the same media data might be used in – and shared among – several documents. Second, tools used in the various stages in the video information "life-cycle" might want to share and exchange meta-data. In the following subsections we discuss what video database requirements these two aspects of sharing put forward.

Modelling Requirements

When several users or tools need to share video information, they also need to share a common view or interpretation of the different types of media and meta-data stored in the video database. A data model for video information will provide such a common view. This model must be rich enough to capture *all* types of meta-data defined in Section 2.2.2 – i.e., bibliographic data, version data, compositional data, media specific data, structural data, sensory and topic content indexes, and supplementary information.

Framework Requirements

In this chapter we have discussed the complexity of video information. Our research goals have been to study what video information semantics should be supported by future DBMS'es. Our point of view is that a video database system should include generic features for video information storage and management that will be useful to most application domains. This means that a video database framework should include a generic data model for video information that can be adapted to specific application domains. A video database framework should also offer an interface for accessing video information in consistent ways and for querying and browsing the contents of the video database.

Querying and Browsing Requirements

Content indexes and annotations may be specific to one user or one video document, or, for instance in the case of sensory content indexes, may be general for all video documents into which a piece of video has been used. Querying and browsing features should allow the user to control whether the scope of the search is document-specific indexes only or whether general indexes or indexes related to other documents using the same pieces of video are searched or browsed as well. This will allow the user to control the degree of meta-data sharing during querying and browsing.

3
DIGITAL VIDEO DATA

A complex weave of communications, electronics, and computer technologies is emerging to create a new multimedia fabric for the next decade. The nature of this cloth is still evolving as an assortment of industries – telecommunications, consumer electronics, computers, cable and broadcast television, and information providers – compete for the emerging market.

J.F.K. Buford, Uses of Multimedia Information
Chapter 1 of Multimedia Systems, 1994 [Buf94b]

Technology has played a significant role in the development from the first short strips of motion picture films in the 1890s until today's digital video technology. In the previous chapter we reviewed video and film from a more or less technology independent point of view. In this chapter we will review digital video technology because this technology has been a prerequisite for developing *integrated* video databases and not meta-data only databases as has been the case up to now [Tur90].

3.1 VIDEO FORMATS

Unfortunately, there are several (incompatible) standards also in the area of video and television. There are, for instance, three dominant analogue television formats: *NTSC* (National Television System Committee) running at 30 frames per second used in USA and Japan, *Secam* (Sequentiel Coleur avec Memoire) running at 25 frames per second used mainly in France and Eastern Europe, and *PAL* (Phase Alternating Line) also running at 25 frames per second which is used in the rest of Europe [Fox91]. In addition, the standards define different number of lines and pixels and are using different – but similar – schemes for handling colour where the television signal consists of three components: the luminance component determines the intensity in each point of the screen (black-and-white television set will use only this component) while two components are used to code chrominance – i.e., colour – information.

The International Radio Consultative Committee (CCIR) has defined a standard (Recommendation 601) for digitising analogue television and video signals. One of the recommended sampling rates are 13.5 MHz for luminance and 6.75 for each of the chrominance components [Fox91]. This will result in a 216 Mb/s[1] (27 MB/s) stream of uncompressed video data.

3.2 WHY GOING DIGITAL?

The ever increasing amount of analogue video being produced indicates that the users consider video as an efficient tool for capturing, storing and exchanging information. In this thesis we propose a database for digital video, but is there really a need for digital video in the first place?

> *The video medium is clearly an extraordinarily powerful force in our society. However, there are still a number of characteristics of video which prevent it from reaching its potential for expressive power, creativity and communications, particularly from the perspective of the individual.* [LH91]

A well-known problem with any analogue data is that they are not robust against noise and tearing. This is also true for analogue video. Each time the video is copied, the noise is increasing and quality is lost. Since aging and regular use of a video tape causes tear, the only way to preserve a video is to take copy of it regularly. But since quality is lost between each copy along the line, the quality inevitably becomes poorer. By going digital, the copying would be an almost error-free process, and each copy would be close to perfect in quality. By using error-detection and/or error-correction codes, the aging and tearing of the digital information can be supervised and new copies can be made in time.

[1]216 Mb/s assumes that 8 bits are used to code each sample. Sony BETACAM [Son94] is using 10 bits per sample giving 270 Mb/s uncompressed.

Direct Access

Among the most important reasons for going digital, are the possibilities for extending the way that users can interact with video information. Digital video, especially when stored on disk-type storage media, allow direct access – i.e., each element within a video stream can be uniquely addressed. Direct access is a required capability for providing interactive video and hypervideo.

Direct access is also a condition for developing advanced searching and browsing tools. To find some video information today, one has to take all relevant video tapes to a video tape recorder. The user must sequentially scan through each tape to find the actual parts he or she was searching for.

Direct access requires that individual frames within a video can be addressed. SMPTE, the Society of Motion Pictures Technical Experts, have defined a way to add time information to each frame within a video tape to be able to address individual frames [Avi93]. The structure of the time code is HH:MM:SS:FF where HH denotes hours, MM denotes minutes, SS denotes seconds, and FF denotes frame number[2]

Most advanced video tape recorders today support the use of time codes to address the individual frames. Time code does not, however, give interactive, direct access to analogue video. A specific frame has to be read to find its actual time code, so lots of time will still be needed because tapes have to been played (in fast forward mode) to advance to the correct position. By going digital, the frame can be accessed directly. An integrated archive tool can also integrate a video player interface in a query tool. This will save the time is needed for finding and retrieving the information searched for.

Video Data Distribution

Distribution of analogue video tapes is cumbersome. When a user not residing close to the video tape archive, needs video information he or she either has to travel to the video tape archive, or a copy of the tape has to be physically sent to the user. The combination of digital video, computers, and high-capacity networks such as ATM [MS95] can give distant users the same service level as resident users.

[2] The 25 frames of the first second in a PAL video would then be coded from 00:00:00:00 through 00:00:00:24, and the next 25 frames would be coded in the same way starting from 00:00:01:00.

Editing

Traditionally, reuse and sharing of film and video information have resulted in new copies of the information or have required that pieces are cut out of older film to be reused. Digital video allows virtual video editing [MD89] such that video information can be shared without making new copies or destroying older films. Digital video also allows nonlinear editing [Whi94a] allowing top-down authoring of video documents instead of the linear, shot-by-shot editing.

Indexing

It is difficult to annotate or index the contents of an analogue video tape and to explicitly store the structure of the video document since it is not possible to interleave such data with video data. Thus, logging and description of the video tapes must be done on other media, and it is not possible to have integrated and consistent control of media *and* meta-data.

Image Quality

Film has one advantage over analogue video – the quality of each frame is very high and one can use individual frames as still images of high quality. The quality of individual frames in an analogue video stream, however, is not generally very good. When going to digital video, each frame may again be of good quality since each frame in the stream is basically a digitised still image.

For those video makers that are experimenting with advanced manipulation of the video image – e.g., by overlaying animations on a video – digital image processing provides a variety of features. If, by an accident, the microphone is visible in a shot it can be removed without any viewer being able to notice.

3.3 DIGITAL VIDEO COMPRESSION

The 27 MB/s data stream resulting from equipment following CCIR Recommendation 601 is hard to handle for a computer system. The data rate is significantly higher than most communications networks and storage devices can manage. Therefore, there is a real need for compressing video data by reducing redundancies that are present in the video data. Redundancies in video data can be divided into four categories (the first three are also utilised in still image compression [GW92]):

Coding redundancies: Coding redundancy is present when the codes assigned to component values do not take full advantage of the probabilities for the values. To avoid such redundancies, one should use less number of bits to encode a frequent value than to encode a value which rarely appears. (The Huffman algorithm [AHU83] is among the best known algorithms for reducing coding redundancies.)

Inter-pixel redundancies: Inter-pixel redundancy is present in most natural images since smaller and large areas have the same or nearly the same colour. One can reduce inter-pixel redundancies by coding the difference between adjacent pixels. An area with the same colour would have zero as difference between adjacent pixels. If using a run-length code where a sequence of equal values is coded as the pair $< value, number\_of\_pixels >$ a large area can be coded by a few bits. (The number of pairs needed per area depends on the shape of the area and the scanning pattern.)

Psycho-visual redundancies: Psycho-visual redundancy occurs because human beings do not respond with equal sensitivity to all visual signals. For instance, intensity is more important than colour information. This is the reason why CCIR 601 recommends using half the sampling rate for chrominance components compared to the sampling rate for luminance.

Temporal redundancies: Temporal redundancy is present because adjacent frames in the video stream often are similar and, in some parts of the video, adjacent frames may even be identical. Compression methods making use of temporal redundancies are using interframe coding – i.e., a frame is coded by coding the difference between the frame and a reference frame (e.g., the previous frame).

Compression methods take advantage of one or more of these redundancies. Two important properties characterise compression methods: The *compression ratio* defining the data reduction, and *lossyness* – i.e., whether or not information is lost during compression [GW92]. *Information preserving* methods compress data without loosing information so the original information can be perfectly reproduced. *Lossy* methods provide higher compression ratios but at the cost of loosing some (possibly not visible) information details.

A number of standards for image and video compression exist:

JPEG: JPEG is an ISO standard for still image compression [Wal91]. It has been – and is – a standard that can be used for compressing video as well, by coding each frame in a video stream as a still image. JPEG does not take advantage of temporal redundancies, and the compression ratio is moderate, typically 15:1 [Fur94]. There is no standard format defining file structures and synchronisation within a video file. It is, for instance, unlikely that the video file format defined and supported by one vendor, e.g., Parallax [Par91], will be recognised by JPEG video players developed by other vendors, e.g., Silicon Graphics [Cha94a].

Direct access can easily be provided without any precautions since each frame is encoded independent from other frames within the stream (see Figure 3.1). When interframe compression is used, such precautions have to be taken. If the actual frame is coded as a difference from a reference frame the reference frame has to be decoded before the frame can be decoded [Lip91]. The reference frame may also be interframe coded so a chain of frames has to be scanned to find the completely encoded frame (called intraframe) at the beginning of the whole chain (see Figure 3.1).

H.261 (px64): CCITT Recommendation H.261, commonly called px64, is a standard specifically devoted to video conferencing over a multiple of 64 KB/s channels (1-30 channels) [Lio91]. The method is intended for video conferencing and other real-time applications, and it is especially optimised for providing fast compression and decompression. The compression ratio covers a wide spectre ranging from 100:1 to 2000:1 [Fur94].

H.261 is not very well suited for interactive applications since direct access will be cumbersome because each frame on the way from the previous intraframe has to be decoded before a given frame can be decoded. The distance between two intraframes may also be quite large.

MPEG: MPEG is an ISO standard family for compressing full-motion video [Gal91]. The first standard in the family, called MPEG-1, is targeted at providing VHS-type quality at 1.5 Mbit/s (the data rate of most CD-ROM players). The compression ratio in MPEG-1 is typically 200:1 [Fur94]. MPEG-2 will give a broadcast-type quality at a data rate in the range 4 to 9 Mb/s. MPEG-4 is targeted at providing video compression at very low bit rates (9-40 Kb/s).

As the case is with H.261, MPEG is making use of interframe coding. The coding scheme, however, is more complicated to facilitate interactive access – e.g., by having direct access in more easy manners and to allow reverse playing. MPEG defines three different types of frames (see Figure 3.1): Intracoded frames (I) are fully and independently encoded. Predictive frames (P) are coded relative to the previous I or P frame by using motion

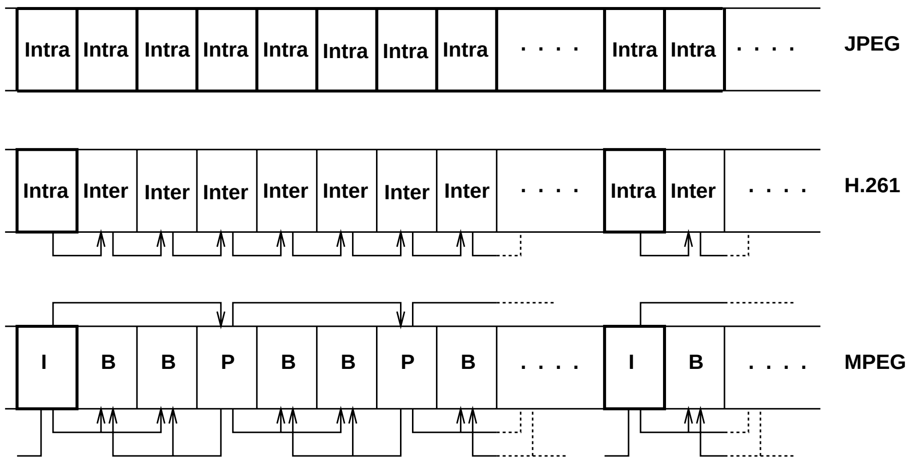

Figure 3.1 Structures of JPEG-, H.261-, and MPEG-coded video streams

vectors. Bidirectional frames (B) uses the previous and the next I or P frame to code current frames. The relative size of the different types of frames is not standardised as such, but according to Lippman [Lip91], the ratios will typically be 5:3:1.

MPEG also defines audio compression algorithms, and it defines a system layer responsible for synchronisation and other supervisory tasks [Fox91].

Defacto Standards: A number of defacto standard formats are available. These formats were developed before the MPEG standard was settled, but their compression methods are similar to MPEG. These systems are system dependent and run on specific computers only: *QuickTime* is an extension to the Apple Macintosh system software [App93]; DVI is an Intel product for video compression/decompression [Don91]; and CD-I (compact disc-interactive) is a piece of consumer electronics manufactured by Philips running interactive, multimedia applications [SvdM91].

Method	Data rate	Per minute	Per hour
H.261 (p=2)	128 Kb/s	960 KB	58 MB
H.261 (p=6)	384 Kb/s	2.9 MB	172 MB
MPEG-1	1.5 Mb/s	11 MB	675 MB
MPEG-2	4 Mb/s	30 MB	1.8 GB
JPEG (half-size)	2.6 Mb/s	19 MB	1.2 GB
JPEG (full-size)	10.4 Mb/s	77 MB	4.7 GB
CCIR 601	216 Mb/s	1.6 GB	97 GB

Table 3.1 Data volume resulting from different compression methods.

Feature	**Magnetic disk**	**Optical disk**	**Low-end tape**	**High-end tape**
Capacity	9 GB	200 GB	500 GB	10 TB
Mount time	None	20 sec.	60 sec.	90 sec.
Transfer rate (per second)	2 MB	300 KB	100 KB	1 MB
Cost	\$ 5,000	\$ 50,000	\$ 50,000	< \$ 1,000,000
Cost per GB	\$ 555	\$ 125	\$ 100	\$ 50 - \$ 100

Table 3.2 Tertiary storage devices in a multimedia system (adopted from Gemmell et al.)

3.4 DATA VOLUMES AND DATA STORAGE

Digital video results in large amounts of data. This is illustrated in table 3.1 which compares the result from using different compression methods and data rates: H.261 with p=2 (used in ISDN [Lio91]) and p=6, MPEG-1 at 1.5 Mb/s, MPEG-2 at 4 Mb/s, JPEG for half-size and full-size PAL format using 1 bit per pixel, and uncompressed video according to CCIR Recommendation 601:

Video databases cannot play a major role in large, shared environments if the storage systems are not able to handle sufficient amounts of data. Fortunately, tertiary storage devices offer cost-effective storage of enormous amounts of data. Such devices are based on robotic arms moving removable tapes or disks from the shelves to a few reading devices [G^+95]. Table 3.2 (adopted from Gemmell et al. [G^+95]) shows some examples.

Take Norwegian Broadcasting Corporation (NRK) as a practical example. In 1993, NRK transmitted 2823 hours television (exclusive programme repeats) [NRK94]. If we assume that all this information is to be stored as MPEG-2 at 4 Mb/s, a fully equiped high-end tape solution (10 TB) can store approximately 2 years full production. Television news counted for 348 hours, and the high-end tape solution can store NRK's television news production through 16 years. We will, therefore, conclude that medium and large scale video archives are technically feasible in a short term perspective and that the research being done in relation to this thesis has practical relevance.

4 DATABASE CAPABILITIES FOR VIDEO DATA MANAGEMENT

The future of the computer and communications industries will be driven by applications, not by scientific breakthroughs like the transistor, the microprocessor, or optical fiber.

Nicolas Negroponte, WIRED, May 1994 [Neg94]

The advent of digital video capturing and compression hardware has given rise to comprehensive R&D activity on digital video software systems and databases. This chapter will review the state-of-the art in video databases relevant to our research questions (see Section 1.4) – i.e., video information semantics, modelling, management, and querying and browsing.

4.1 VIDEO DATA ABSTRACTIONS

Low-level aspects of video data such as decompression algorithms, synchronisation issues, and operating system and file system requirements are often aspects that video applications do not require to control themselves. Thus, a video database framework should offer an abstract video data type that will release applications from solving such issues. However, these aspects determine the *quality* of video playback, and applications may require to be able control them indirectly by requesting a certain quality of service from the underlying video database framework.

A useful abstraction for many video applications is to consider video as a window which can replay stored video data. The applications should have the ability to control spatial properties – e.g., size and the position of the video window – and controlling the video playback itself. Several existing application frameworks offers predefined visual classes for text and graphics display and integrated control of such classes. Schnorf [Sch93] has proposed extensions to existing application frameworks to be suitable for video display as well.

Traditional video tape recorders and video discs are the most used metaphors for digital video playback control [App93, KE93, HS93, SG94a]. According to such metaphors, the application should be able to request a video database framework to load (eject) a video document into (from) a video window, to force the video player to jump to a specific interval within the document, to initiate playback of the active interval, and to control the speed and direction (forward/backward) for the the playback. A video tape recorder will often also maintain status information – e.g., identity of the current video document and current speed and position – which might be interrogated by the user.

4.2 VIDEO COMPOSITION

In this section we will discuss various paradigms and models that have been proposed for representing compositional data in video/multimedia databases. Hardman [HvRB93] discusses different authoring paradigms; comparing traditional bottom-up approaches such as timelines and scripts with more structured, top-down approaches. We would like to stress that the database model for compositional data may support different authoring paradigms even if it is best suited for only a subset of authoring approaches.

As discussed in Section 2.2.2, compositional data describe how video data are used in video documents' compositions. Thus, compositional data give an explicit representation of video data sharing. When compositional data are explicitly stored, video documents can be generated "on the fly" and need not be stored in a materialised form in the database. This will reduce the amount of video data replication when video data are shared among different documents.

Compositional data will be created during video document editing and will be used to generate the video stream constituting the document. In the discussion of each individual model we will evaluate how well the model supports editing and playback. We will also discuss the formalism or foundation on which the model is based.

4.2.1 Timeline Models

The timeline has been the most used abstraction for video composition used in authoring tools such as Avid Media Composer [Avi93]. A timeline gives an explicit representation of the time axis and describes how events such as shot transitions relates to the time axis of the video composition. The video document is often composed of several *tracks* – e.g., one video track and two audio tracks in the case of stereo television. Figure 4.1 (adopted from Little [Lit94]) shows a timeline example. Both inter- and intra-track synchronisation are defined by the explicit binding to the timeline.

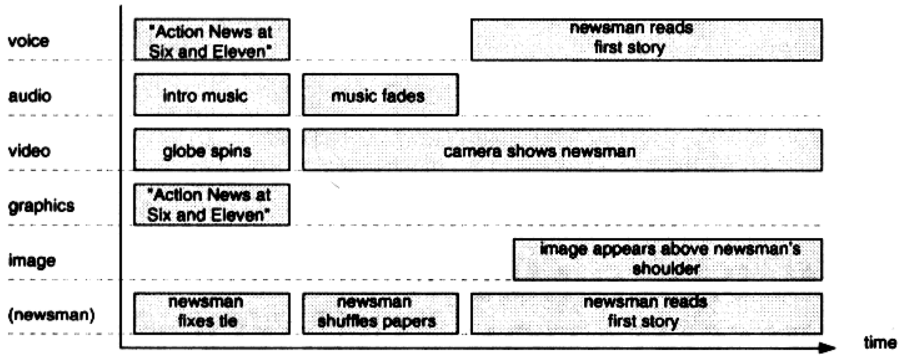
voice
"Action News at Six and Eleven"
newsman reads first story
audio
intro music
music fades
video
globe spins
camera shows newsman
graphics
"Action News at Six and Eleven"
image
image appears above newsman's shoulder
(newsman)
newsman fixes tie
newsman shuffles papers
newsman reads first story
time

Figure 4.1 Timeline example (adopted from Little)

A timeline based composition may be represented as an *edit decision list* (also called edit-list) [Whi94a] where each entry defines track identity and start and end time for the shot within the time axis of the video document, the identity of the media segment to be used, the start and end time within the time axis of the media segment, and type of transition between the shot and the previous one.

Breiteneder et al. [BGT92] proposes a timeline model based on four entities (see Figure 4.2): The *movie* entity represents the video document's composition which is temporally composed of *track*s. Tracks are derived from *media* data in such a way that each track is derived from one type of media. In addition to these basic entities, the model allows tracks to be grouped into an intermediate *layer* entity. Layers support reuse of combinations of tracks, and they allow alternate groups – i.e., one, and only one, track is selected from the layer when the movie is played back.

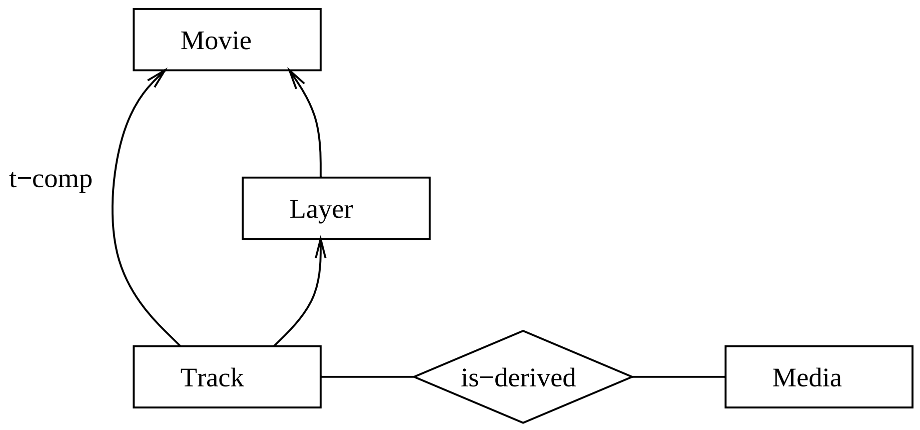

Figure 4.2 An ER diagram for timeline models (adopted from Breiteneder et al.)

Two different types of relationships are used to express the relations between the entities in the model: Temporal composition, `t-comp`, defines how tracks are grouped into layers and how layers and tracks are to be synchronised to each other in a movie. Derivation relationships, `is-derived`, define how a media entity is physically stored as media samples. They also define the mapping between track time and media time.

One of the advantages of the timeline model is that it represents a rather direct representation of a video document which is familiar to many video users – e.g., video directors. Individual shots are explicitly bound to playout-time and only small amounts of computation are required to map the video document onto the corresponding media data.

The main criticism against the timeline approach [HR94, HvRB93, Lit94] is that the absolute timing specification complicates editing and copying. If, for instance, the author wants to insert a video shot into a video document, all subsequent shots in the document must be given new start and end times. Even worse, the timeline does not specify relative relationships between different tracks, so all audio tracks, for instance, must also be updated when a new video shot is inserted into the video track to maintain inter-track synchronisation.

The timeline itself does not provide higher level abstractions that allow the user to group objects from different tracks into semantically meaningful components – e.g., scenes. Thus, the timeline model may result in cumbersome operations if an author wants to restructure a document by modifying the sequence of scenes or if he/she wants to reuse scenes from other documents.

4.2.2 Timed Petri Net Models

Timeline models are *point-based* models in that each event in the model is associated with a point in time space – e.g., the start of a video shot or the end of a music clip. Little et al. [LG90, Lit94] have proposed a timed petri net (also called object composition petri net – OCPN) model that is *interval-based*: *"Each place in the OCPN represents the playout of a multimedia object while each transition represent synchronisation points."* [Lit94]. Figure 4.3 (adopted from Little [Lit94]) shows the OCPN representation for "Action News" with the timeline representation shown in Figure 4.1.

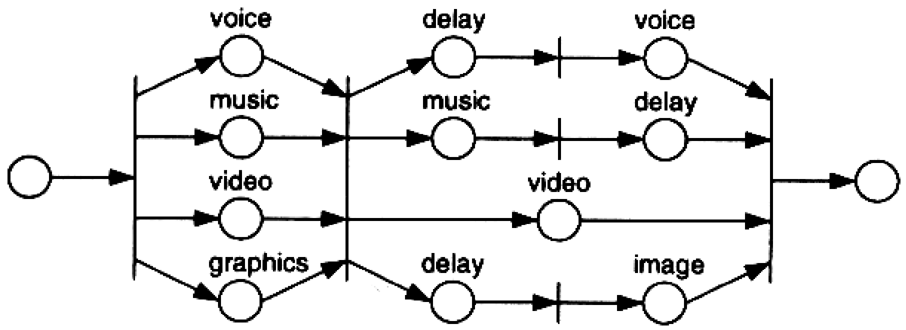

Figure 4.3 "Action News" – its OCPN representations (adopted from Little)

Timed petri net models have been proposed to overcome the absolute timing specification of the timeline models. Thus, no additional computation is needed to update timing information when the petri net is modified. The penalty, however, is that a playback schedule has to be computed when a user selects the video document before the document can be replayed. A complex document composition may contain many places and transitions, and much computational effort is needed to determine the playback schedule.

Timed petri net models assume that each place represents an atomic value. Individual frames represent the "real" atomic values in a video stream. It would, however, not make sense to define one node in the petri net for each frame in a video stream. Instead, shots are often considered to be the atomic values, and, thus, some inherent synchronisation mechanisms are needed to synchronise video frames to each other and to accompanying audio samples [Lit94].

As noted by Prabhakaran et al., the OCPN model does not have the expressive power to deal with user interactions. Assume that the user interrupts video replay – e.g., to skip the rest of the first news item to proceed to the second one [PR94]. The petri net model shown on Figure 4.3 does not indicate how the playback synchronisation should be performed in this case. An augmented petri net model, AOCPN, has been proposed to handle user interactions [PR94]. The disadvantage with this solution is that AOCPN model results in even more complex petri nets than the "pure" OCPN, and this complexity must be handled by the video database tools.

4.2.3 Hierarchical Models

Different research groups have proposed data models for compositional data that have hierarchical characteristics. These define complex video objects by recursively combining simpler objects. Another common characteristic is that hierarchical models define relative temporal relations between elements and not absolute relation to document playout time axis as timeline models do. We will review the *Time-Interval-Based* (TIB) model [Lit94], the *Layered Multimedia Data Model* (LMDM) [SW94], and the models used in *Xavier* [HR94], *Algebraic Video System* [WDG95], and *OVID* [OT93]. Some of these models also give support to spatial composition, but these features will not be discussed here.

Time-Interval-Based Model

The timeline and the timed petri net models do not support structured, top-down authoring directly. Little et al. have developed a *temporal-interval-based* (TIB) modelling approach [Lit94, LG93]. The model is based on Allen's work on temporal intervals [All83] which showed that two temporal intervals can be related to each other in 13 distinct ways. In TIB, n-ary extensions of the temporal relations are used to define composite video objects. Figure 4.4 (adopted from Little [Lit94]) illustrates the temporal hierarchy corresponding to the "Action News" from Figure 4.3.

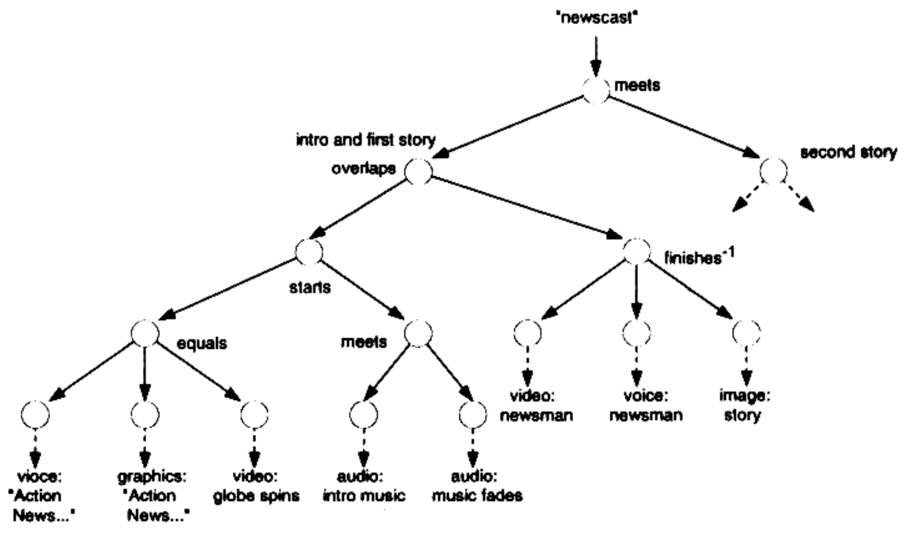

Figure 4.4 Temporal hierarchy for the "Action News" from Figure 4.3

Not all 13 temporal relationships are suited for video composition. Duda discusses some of the problems and concludes by defining six operations that captures most of the expressive power that is needed to define video compositions [Dud95].

Layered Multimedia Data Model

The Layered Multimedia Data Model (LMDM) [SW94] provides a multimedia event calculus for specifying temporal structures. The term 'event' is used to denote a groups of objects – e.g., a shot is an event consisting of the related video frames. The operations in the calculus are: *Subevent* defines a cut-out section of an event in ways similar to the `is-derived` relationship proposed by Breiteneder. *Concatenate* defines an event to be the concatenation of two other events, while *overlay* defines an event where the two events occur at the same time. *Extend* defines an event which contains i concatenations of an input event. *Insert* defines a new event where one existing event is inserted into a specific point of another. *Wait-until-signal* defines an event which consists of an input event followed by a pause of arbitrary duration which ends when a specific signal is received. *Repeat-until-signal* defines an event where the input event is repeated until the specific signal is received. *Pad* defines an event where silence events of given durations precede and follow the input event. LMDM also provides a *synch()* operator which can be used to define synchronisation points.

Xavier

The Xavier library offers a number of operations for combining multimedia objects into composite objects [HR94]. The *Section* operation is similar to the `is-derived` operation proposed by Breiteneder et al. and can be used to identify one (sub)interval from a temporal multimedia object. The *SEBox* operation arranges the element along the temporal dimension such that the start of one element directly follows the end of the previous one. The *Overlay* operation specifies that the elements are to be replayed in parallel. The *Loop* operation repeats the element for a designated time period. The *Position* operation can be used to bind an object to an absolute time-scale as in the timeline model. The *Constraint* operation is used to attach synchronisation constraints to the elements – e.g., by specifying that two elements shall have the same start time during replay. Figure 4.5 (adopted from Hamakawa et al. [HR94]) illustrates the use of Xavier composition operations.

The Structural and Temporal Object-oRiented Multimedia (STORM) DBMS [Adi95] is a multimedia extension to the O_2 object server. STORM provides operations for sequential and parallel composition. Two sequential and five parallel operations are defined to allow for different temporal relationships between the starting and ending points of the input objects – e.g., *par-equal* is used when the two input objects shall have the same start and end times while *par-start* denotes that the two input objects shall have the same start time but might have different ending times.

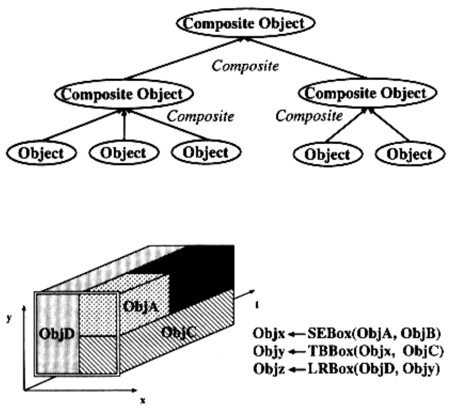

Figure 4.5 Composition in Xavier adopted from Hamakawa et al.

OVID and Algebraic Video System

Algebraic Video System [WDG95] and OVID [OT93] add more semantics to the definition of video objects than the two previously discussed models. In the latter systems, a general video object contain descriptive data that are related to an interval of video. One piece of video may, thus, be part of several different video objects. When combining such video objects, one has to be cautious to avoid that intersecting intervals are not duplicated. Composition operations in Algebraic Video System and OVID are defined to handle this.

Table 4.1 adopted from Weiss et al. [WDG95] summarises the composition operations offered by Algebraic Video System, while Figure 4.6 adopted from Weiss et al. [WDG95] shows how intersecting video objects can be combined into a non-redundant video stream.

The video objects in OVID are even more complex in that a video object may be related to a set of non-intersecting intervals of video instead of just one, contiguous interval. OVID defines three operations for video composition. *Interval projection* is used to create new video objects that represent temporal parts of other objects. *Interval set merge* is similar to the union operation in Algebraic Video System while *interval set overlap* is similar to the intersection operation.

Operation	**Usage**	**Function**
Concatenation	$E_1 \circ E_2$	E_1 follows E_2
Union	$E_1 \cup E_2$	E_1 follows E_2; common footage is not repeated
Intersection	$E_1 \cap E_2$	Only common footage of E_1 and E_2 is included
Difference	$E_1 - E_2$	Only footage of E_1 that is not in E_2 is included
Parallel	$E_1 \| \| E_2$	E_1 and E_2 are played concurrently and start simultaneously
Parallel-end	$E_1 \wr\wr E_2$	E_1 and E_2 are played concurrently and stop simultaneously
Conditional	`(test)`?:E_1 : E_2 : ... : E_k	E_i is included if `test` evaluates to i
Loop	`loop` E_1 *time*	E_1 is repeated for a duration of *time*
Stretch	`stretch` E_1 *factor*	E_1 is included but with playback speed scaled by *factor*
Limit	`limit` E_1 *time*	Includes the part of E_1 not exceeding *time*
Transition	`transition` $E_1 E_2$ *type time*	Defines *type* transition effect of duration *time* between E_1 and E_2
Contains	`contains` E_1 *query*	Includes expressions from E_1 that match the query

Table 4.1 Algebraic Video System composition operations

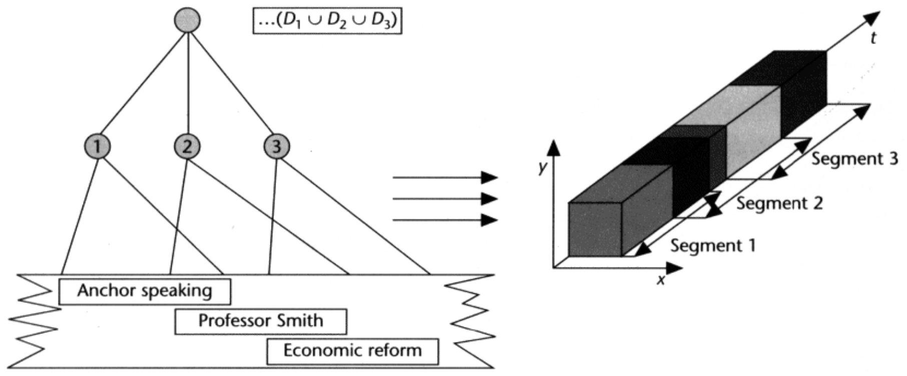

Figure 4.6 Algebraic Video System's union operation (adopted from Weiss et al.)

Summary

The main strengths of the hierarchical models are explicit support of structured, top-down authoring and relative timing of elements which eases editing and reuse of composite objects. Contrary to timeline models that only models static, temporal relationships, hierarchical models also defines operations that can be used for creating compositions. Such operations have to be implemented by toolkits or by authoring tools on top of the timeline model.

In general, hierarchical models are semantically more expressive than timeline models. A video object in a hierarchical model is not just a piece of video, but it can be a piece of video with a coherent semantic contents or meaning. This can be exploited in contents indexing as we will discuss further in Section 4.4.

The penalty of relative timing is that when the user selects a composition for playback, the system has to recursively parse the nodes and compile a schedule file for each node. The composition hierarchy my become very deep for a complex document, especially if the video objects are fine-grained so that each shot must be defined as the union of several video objects.

4.2.4 Document Encoding Schemes

As discussed in the first part of this section, different systems represent compositional data in different ways. To encourage exchange of compositional data and to encourage development of generic presentation and scheduling engines, standardisation efforts have been undertaken to define standard formats for representing multimedia documents.

In this section we will briefly discuss the ISO initiatives HyTime [NKN91] and MHEG [MBE95, Pri93]. The two standards are based on different concepts – e.g., HyTime is defined in Standard Generalized Markup Language (SGML) while MHEG is defined in Abstract Syntax Notation One (ASN.1). Though the processes differ in details, presentation of MHEG and HyTime documents follow the same type of schema as illustrated in Figure 4.7: First, the definition of the document and its component objects are retrieved from a document store and parsed/decoded. The decoded document description is then transferred to the engine which performs the document layout and scheduling. The engine may also accept interaction from the user application - e.g., to interrupt playback or to allow for user input.

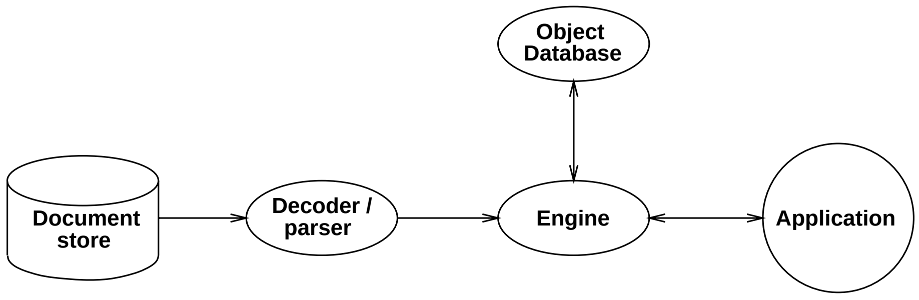

Figure 4.7 Document processing steps

HyTime and MHEG are mainly intended and suited for exchanging compositional data. If they become widely accepted, the will encourage exchange of multimedia documents between different systems. HyTime and MHEG will probably not play a significant role in the exchange of video only documents because some defacto standard edit-list formats already make exchange video documents between different editing systems possible. From a database point of view, MHEG and HyTime are document scripting languages that are external to the database and they do not define "real" data models which can be exploited by the database systems.

4.2.5 Summary

What can be learned from the current state-of-the-art of video document composition is:

- Timeline models represent video composition in a direct way that is efficient for playback but not for video editing.
- Timed petri net models stores relative timing information which is efficient for video editing but which require some computational effort for computing the actual playout schedule.
- Neither timeline nor timed petri net models support structured, top-down authoring directly.
- Hierarchical models offer a number of operations for creating composite video objects that can reflect the author's grouping of objects into semantically meaningful groups.
- Hierarchical models require some computational effort for computing the actual playout schedule.
- Multimedia document encoding standards offer little value-add for video databases because edit-lists can be used to exchange traditional video compositions between different systems. MHEG and HyTime are more appropriate when the video document includes hyperlinks or is tightly related to a hypermedia document.

4.3 DOCUMENT STRUCTURES

The contents of books are often organised into chapters, sections, and subsections. The contents of video documents may also be organised into hierarchical structures in similar ways. As discussed in Section 2.1.2, traditional film theory defines a three level hierarchy consisting of shots, scenes and sequences.

In the multimedia community, hierarchical structures are also widely used. The Virtual Video Browser [L$^+$93] implements a one-level structure where each movie – i.e., video document – is divided in a sequence of scenes. Kameyama et al. [KHT91] and Simonnot et al. [SS95] propose a two level structure where each video document is divided into sequences, and where each sequence is a group of shots. Rowe et al. have implemented the full, three-level structure of shots, scenes, and sequences in the Video Database Browser [RBE94].

Each node in a hierarchical composition model may also represent a structural component of a video document. The second highest level nodes in Figure 4.4, for instance, correspond to individual stories in the "Action News". These would have been classified as "sequences" according to traditional film theory. A composition hierarchy may have more than three levels and, thus, the corresponding structure may have a broader variety in semantic meaning than the traditional three level structure. The disadvantage, however, is that it may be difficult to identity nodes in the hierarchy at the same abstraction level – e.g., to find all scenes.

4.4 VIDEO CONTENT INDEXING

Hampapur et al. defines: *"Content based access is the ability of a system to retrieve information based on its meanings or semantics"* [HJW95]. For video databases, content based access usually means retrieving video with specific visual – e.g., video showing a moving car or a walking man – or topic contents – e.g., video covering president elections. At the physical level, video is a temporal sequence of two-dimensional pixel arrays with no direct relation to its semantics. Video contents indexing is the process of structuring video and meta-data – i.e., defining an video information model – and the process of inserting data into this model.

In the two following subsections we will discuss content index characteristics, and we will have a short look on the indexing process. The rest of this section will review different ways to organise indexes for describing the contents of intervals of video.

4.4.1 Types of Indexes

Video content indexes can be classified in several dimensions. Rowe et al. differentiates between *sensory* and *topic* contents [RBE94], where sensory content relate to visual or audible objects, while topic contents relate to the more abstract "message" carried by the video.

A distinction can also be made [Gro94, HJW95, JH94] between alphanumeric (keyword-based) indexes (in [JH94] called *qualitative features*) and image/audio feature indexes (in [JH94] called *raw features*). The qualitative features are domain dependent and must be assigned from a domain model. A rich model for semantic domain knowledge will be useful and necessary in the management of qualitative features [Cha94b]. Raw features – e.g., colour histograms and image edges [GW92], on the other hand, are domain-independent but provide indexes at a low level of abstraction. Raw features will only be applicable to sensory content, while qualitative features might relate to either sensory or topic contents.

A last consideration is the spatio-temporal extent for indexes [JH94]. Some indexes may be valid for single frames only (the colour histogram, for instance, will normally differ from one frame to another). Other indexes, for instance indexes related to persons or objects shown in the video, will describe contents of a contiguous interval of frames. A similar distinction can be made in the spatial domain between indexes that describe the frame as a whole and indexes that describe a spatial region [BKW94] within a frame or an interval of frames.

4.4.2 The Indexing Process

Manual, keyword-based indexing assumes that the user manually enters the keywords and the descriptions, either as part of the production process or as a post-production activity. The criticism against manual indexing is that 1) it is very time-consuming, 2) the user will index only a small fraction of the actual contents, and 3) the indexing may have a subjective bias imposed by the user. It is, however, reasonable to say that when the indexing is done by experienced librarians, the resulting indexes will describe the contents in a fairly objective and compact way with most of the relevant parts of the contents indexed.

The alternative to manual indexing is computer-controlled video/image analysis. The automatic indexing can be divided into three levels [GW92]: The *low-level processing* performs image enhancements – e.g., noise reduction – to have as good basis for the analysis as possible. The *intermediate-level processing* segments the image and extracts the main components – e.g., regions. The *high-level processing* is aimed at recognising objects in the image and interpreting image contents. Scene analysis in video is similar to image analysis. Some processing (segmentation, for instance) may be easier to perform in a video because motion information can be utilised [CVB95].

The advantage of the automatic process is that it is objective and that it might be used also when the user wants to retrieve objects that where not originally indexed. The problems associated with automatic indexing are especially in the high-level processing. Gonzales puts it like this: *Computerised image interpretation is an exceedingly complex process. Difficulties arise from both the large amount of data that must be processed and the lack of fundamental processing tools to get from what is given (an array of pixels) to the desired result (a detailing of image content)* [GW92]. Hampapur et al. proposes semi-automatic indexing procedures where automatic techniques assist users during manual indexing [HJW95]. Image/video recognition is a large and active research area which we do not intend to cover in much detail in this thesis.

4.4.3 Segmented Indexing

The traditional way to do video indexing is to *segment* the video into chunks (often, each chunk correspond to a shot) [L^+93, SS95, Tur90]. A set of keywords is associated with each chunk of video and indexes the content of the chunk.

The advantage with the segmentation method is that it is straightforward to find all indexes relevant for a given chunk. The fact that all keywords are associated with the complete chunk of video gives rise to some serious problems [Smi92, Dav94]: Assume that n keywords are associated with a chunk of video, and that a user wants to add a new keyword that is only valid for a smaller chunk in the middle of the larger chunk. The original chunk must now be divided into three parts: The part of the original one appearing before the smaller one, which is indexed by the n keywords, the smaller chunk which is indexed by $n+1$ keywords, and the last part of the original chunk which is indexed by the original n keywords.

4.4.4 Stratification

To overcome the problems resulting from segmenting, Smith introduced the concept *stratification* [DSP91, SP91, Smi92]. The key idea is to segment descriptions instead of segmenting the video: each keyword would be associated with distinct pieces of video. One can then follow one keyword or descriptive attribute as a layer (stratum) through the video. To obtain a full description of a specific frame or an interval of frames, one will have to merge the different strata. Other researchers have taken approaches similar to stratification [RBE94, Dav94].

The main disadvantage with the stratification approach is that some computational effort is needed to find the various indexes that are related to a given interval of video frames. The consequence is also that combined video queries might be more complex. For instance, assume that a user query requests pieces of video that is indexed by two different indexes. In the segmented approach, this can be determined by investigating the individual segments to see if the two indexes are part of the description set. In the stratification approach, each of the two indexes will identify a set of video chunks. One has to compare the individual members of these two sets to find the pieces of video where both indexes apply.

4.4.5 Hierarchical Indexing

Hierarchical indexing provides means for combining single indexes into more complex concepts. For instance, the Algebraic Video System [WDG95] defines algebraic operations used for video composition (see Section 4.2). The algebraic operations can also be used in the system to define nested strata. By nesting strata, the system provides a more direct way to define the relationship between different strata related to the same piece of video than the original stratification approach. Figure 4.8 (adopted from Weiss et al. [WDG95]) shows the nesting of strata using the algebra.

The video objects in OVID [OT93] identify an interval of video and a set of attribute/value pairs which can be used to index the contents of the video object. The database is schema-less and it is all up to the user to define valid attribute names – i.e., attribute types. As discussed in Section 4.2, new video objects can be created by using some of the composition operations defined. Through so called *inclusion inheritance*, composed objects will inherit attribute/value pairs from the object(s) from which it was created. To control the degree of inheritance, an object may define its attribute/value pairs as public or private. Only pairs defined as public will be inherited through inclusion inheritance.

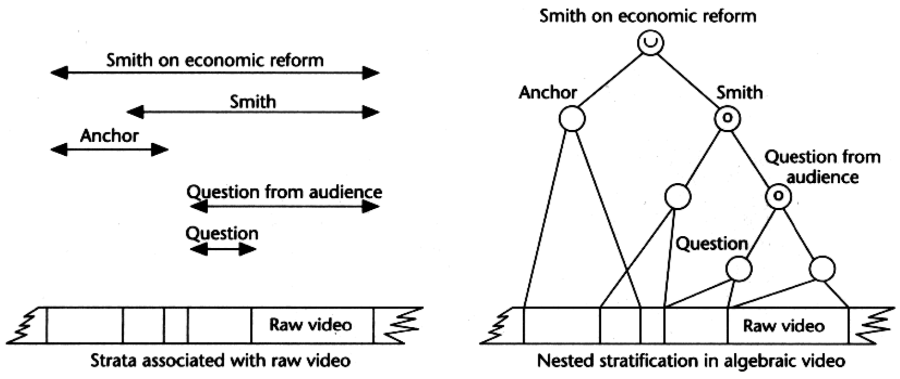

Figure 4.8 Nested stratification (adopted from Weiss et al.)

4.4.6 Summary

Segmentation is a method that allows direct access to all descriptions for a segment of video. For a complex combination of indexes, it will, however, generate many segments, and several segments may have many common indexes. Stratification allows each index to be defined independent of any other index. The drawback is that it is computational harder to retrieve all indexes related to a given piece of video. The hierarchical methods uses a stratification type of approach but try to define a hierarchy of indexes to better represent the relation between different indexes.

4.4.7 Meta-Data Acquisition

One important question for the users of a video database is how to produce meta-data. There are several approaches that can be taken. The most ambitious one, *feature extraction*, assumes that characteristic features can be automatically extracted from media data. A less ambitious approach, *in-production capturing*, assumes that the tools used for recording and/or video editing (semi) automatically collects meta-data – e.g., when recording tools automatically register date and time for recording, or when compositional data are generated by authoring tools. The approach that is chosen by most users today (though this is a time-consuming task), *post-production capturing*, assumes that users

Meta-data	**Feature extraction**	**In-production**	**Post-production**
Compositional	Cut-detection	Yes	Yes
Bibliographic	No	Some	Yes
Structural	Simple structures	Yes	Yes
Sensory content	Limited domains	Little	Yes
Topic content	Hardly	Some	Yes
Supplementary	No	Yes	Yes

Table 4.2 Usability of different meta-data capturing methods

– e.g., librarians – manually enters meta-data when the production process is completed. Table 4.2 (adopted from Hjelsvold et al. [HMS95b]) indicates how the different methods apply to different types of meta-data.

4.5 QUERYING AND BROWSING

Video database querying is the process of retrieving information from the database satisfying a user specification. Browsing, on the other hand, is a more explorative way of retrieving information from the database [GMP91]. Browsing complements querying in that it gives users means to retrieve information relevant in a given context without having to specify the contents of this information.

4.5.1 Querying Temporal Properties

Video data are inherently temporal – although not in the traditional temporal database sense [T^+93a]; temporal databases are addressing the needs for maintaining past, present, and future data. A video stream defines a temporal sequence of frames with a change in frame contents during replay. A sequence of video frames then constitutes a time interval within the time system of the corresponding video document.

Research on temporal data and databases has resulted in knowledge that can be useful in the video database domain. The work by Allen [All83] on temporal intervals, for instance, showed that there are 13 distinct temporal relationships that can exist between two arbitrary time intervals. This fact is also valid in video databases given the additional condition that the two video intervals are parts of the same video. Similarly, temporal variants of set-operations [CC93] are equivalent to the ones provided by the Algebraic Video System [WDG95].

Temporal visual query language TVQL [HR95] is a query language that can be used to support video analysis and to identify temporal trends in video data. Temporal trends are identified via querying the database for temporal relationships between video annotations. The TVQL user interface offers four sliders that can be combined to specify any temporal relationship between pieces of video.

4.5.2 Querying Approaches

Querying in traditional record-oriented databases is a mature research field. Relational databases, for instance, are based on a formal foundation defining basic algebra operations [AA93] and have motivated for the widespread use of the query language standard SQL [MS93]. This research is also moving into the field of more complex data types [AHV95]. Keyword-based indexes (see Section 4.4) can often be structured as records and, thus, traditional querying mechanisms can be used in video databases. In fact, several projects have been taken this approach:

- The *Virtual Video Browser* [L$^+$93] uses the POSTGRES DBMS as query engine where queries are written in PQUEL.
- The *Video Database Browser* [RBE94] is also based on POSTGRES. Application specific functions have been included in the database – e.g., *overlaps* which computes whether two pieces have common frames.
- The *Algebraic Video System* [WDG95] uses a simple predicate query language that searches the hierarchy of every algebraic video node in a given collection and returns the result set of nodes that satisfy the query. The query mechanism ensures that the same nodes is not searched more than once and a specific *Hide-content* can be used during composition to indicate that a sub hierarchy should not be searched.
- *OVID* [OT93] offers a SQL-type of query language, VideoSQL, for retrieving OVID video objects having a specific attribute/value pair, having a set-type attribute that contains given values, or defined over a given interval of video.

- The *Video Query System* [HS95] defines a SQL-type of query language that includes operations for determining whether two intervals of video intersects, operations for computing intersections, unions, and complements of pieces of video, and operations for set iterations (FOREACH and FORALL).

Media Streams [Dav93, Dav94] offers an iconic language for video annotation and querying where icons are organised into semantic hierarchies allowing generalisation and specialisation of icons. Media Streams supports search for pieces of video associated with a given set of icons. In addition, generalisation or specialisation transformations of icons allow more semantically based retrieval. The most advanced technique is temporal analogically retrieval that retrieves pieces of video that matches a certain semantic contents and temporal structure. Media Streams also adds the ability to represent and match on temporal relations.

Information retrieval is, in the same way as database querying, a mature field [vR79, Sal88]. The contents of video can be described as a textual abstract/synopsis that can be searched as other texts. This approach has been studied by Simonnot [Sim95, SS95] who has proposed a retrieval process using term weighting, relevance feedback and result ranking.

Image recognition techniques might be used to develop alternative ways of querying a video database. Image processing and image recognition constitute a large, demanding area which we will not try to cover in much detail. Feature-based retrieval that has been described includes querying by specifying content features such as in QBIC [A$^+$95], querying by sketching [ZSW95], and similarity retrieval [J$^+$95].

4.5.3 Browsing Approaches

Two different browsing operations that one will seek in video databases are:

1. The ability to retrieve video or meta-data that is relevant to a given piece of video. The stratagraph [SP91] is such a browsing tool that displays indexing data describing a piece of video.
2. The ability to browse through the contents of a video document in an efficient way. This is a useful operation because it is difficult and time-

consuming to browse a video document by fast forward/backward replaying only.

A number of proposals have been made to represent the contents of an interval of video frames. A *salient still* [Teo93] is a single images that represent a whole range of frames. The salient still reflects the aggregate of the temporal changes of frames within a sequence of frames. The *Video Streamer* [Ell92] and *STREAMS* (see Figure 4.9 adopted from Cruz et al. [CH94]) display a sequence of frames as a three-dimensional cube (with dimensions x, y, and t) generated by stacking individual frames on top of each other with a small displacement in the x- and y-direction. The *Video X-ray* [T^{+}94b] performs a similar operation by slicing out segments in the x-t plane (or, alternatively, the y-t plane).

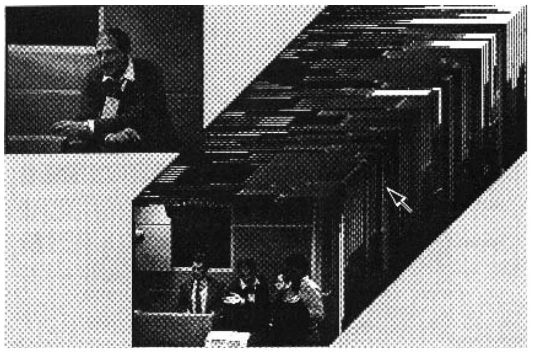

Figure 4.9 Three-dimensional browsing in STREAMS (adopted from Cruz et al.)

Shots usually consist of frames with a coherent semantic. One could organise shots into hierarchies where each level represents clusters at a given abstraction level. A prerequisite for such organisation is that one knows the shot transitions. If these are not inserted in the database as part of the authoring process, one needs tools to identify shot transitions from a stream of video. Several researchers have proposed algorithms for automatically detecting shot transitions, among these [SZ94, A^{+}94, HJW94].

One straightforward way of organising shots in hierarchies is to group a fixed number of shots into a first level abstraction node. Then a fixed number of first level nodes could be grouped into second level nodes etc. Zhang et al. proposes such a browser in [ZSW95]. The disadvantage with this "brut-force" organisation is that it clusters shots and intermediate nodes without paying attention to semantic coherence between the components. An alternative approach could be to organise shots in such ways that each node at a given level has roughly the same (time) duration. The hierarchy would in that case at least partition the video document into temporally homogeneous clusters.

Yeung et al. [Y^+95] have proposed an alternative procedure that uses low-level features for clustering shots with common properties. They propose to cluster shots based on colour, shape, and image correlation similarity. That means that shots that are similar in contents – e.g., a sequence of shots from a crowd of people – will be grouped together.

Existing hierarchical browsing methods do not require a priori domain models. If such models exist, however, it might be possible to automatically organise shots into semantically meaningful entities. One typical domain having a sufficiently limited syntax is television news where most of the news item often are introduced by an anchor person. Research has shown [ZSW95, SSJ93] that it is feasible to have automatic recognition of anchor person scenes.

Users will seek methods that browse a video document's logical structure (see Section 4.3) whether this structure was generated manually, semi-automatically or automatically.

4.6 CONCURRENCY, INTEGRITY AND VIDEO DATA DELIVERY

Concurrency and integrity are important issues in database systems. Today, most video databases are read-only databases in the sense that video and metadata are inserted once and rarely modified or deleted later. Therefore, concurrency and integrity in video databases have not been specifically focused by researchers. It can, however, be expected that video databases (at least metadata) in the near future will be more actively updated. Then there is a need to study whether specific integrity requirements and transaction models are needed to maintain consistency and integrity of video databases. A brief discussion of concurrency and integrity issues are given in [Hje94b].

Video delivery is one of the most active fields in multimedia database research and is one of the central issue for the NOSSDAV conferences [Ran92, S^+93, LG95] and for other video/multimedia conferences. Problems that are being studied include admission control (e.g., [HBP94]), data placement/data distribution (e.g., [T+93b, T+94a, FD95]), disk scheduling (e.g., [GH94, RW94]), buffer management (e.g., [FD95]), operating system support (e.g., [Tok94, Ste95]), and video server/storage system architectures (e.g., [T+94a, RC95]). A tutorial on these issues can be found in [G+95].

4.7 COMMERCIAL DATABASE SYSTEMS

Commercial database manufacturers have recognised the market for video databases, and commercial video database products are already available. In this section we will briefly present two such products, the *Oracle Media Server* and *Gain Momentum* (a product from Sybase, Inc.).

Oracle Media Server [LOP94] is an extension to the Oracle 7 relational DBMS. Oracle Media Server supports data types such as BLOBs, texts and isochronous data. Isochronous data are stored in a realtime, striped file system that allows concurrent, random access to video and audio while meta-data are stored as structured data in the Oracle 7 database. A real-time stream server controls CPU, memory and disk resources to provide isochronous delivery of concurrent streams. The server also provides full VCR-like controls to the user.

Gain Momentum [BR93] is a more ad-hoc extension to the Sybase relational DBMS. Gain Momentum supports data types such as text, vector graphics, bitmap images, audio, video, animation, and timelines. Video can be incorporated into Gain Momentum applications in two ways. The first is to use the specific video capture utilities provided by Gain Momentum to generate a low-quality video that can be played back with no special hardware. Such video objects will be stored internally in the application object store. The second approach typically offers higher quality playback, but it requires using third party hardware and/or software at playback time. Video data have to be stored outside the Gain Momentum but video display can be integrated into a Gain Momentum application's display.

4.8 STATE-OF-THE-ART SUMMARY

In this chapter, we have reviewed state-of-the-art in video databases with an emphasis on what video information semantics to be included in video databases. The implications for our research are summarised in this section.

Video Information Modelling

Several researchers argue that video databases should support a structured authoring paradigm where a hierarchical structure of sequences, scenes, and shots is adopted from film theory. On the other hand, no agreement has been made on what model to use for representing compositional data. Some researchers propose timeline based models, while others adhere to time petri nets or hierarchical models. These observations indicate that video databases should include hierarchical video document structures and that video databases should support structured authoring.

Current state-of-the-art indicates that the stratification approach should be chosen in preference to the segmentation approach because the stratification approach allows addition, deletion, and modification of indexes without having to resegment the video.

Indexes may be based on keywords or on automatically extracted media data features. In the current state-of-the-art, keyword based indexing is needed because some of the indexes cannot easily be extracted from the media data – e.g., topic content indexes. Also, in the current state-of-the-art, feature based indexing is limited to well-defined application domains because object recognition is model-driven and, thus, models of the objects to recognise are required. Such models can only be defined in limited, well-defined application domains.

Querying and rowsing

Different techniques have been proposed for keyword-based querying, including text and icon based database query languages and information retrieval approaches. There is no clear indication that one of these methods are significantly better than the other. As already mentioned, feature-based retrieval requires application domain models that are hard to develop. For the time being, it seems that video database systems should include querying facilities for keyword based retrieval. The temporal aspects of video information are important, and current research indicates that video query facilities should allow users to utilise temporal relationships in video query formulation.

Research has shown that contents browsing tools such as the stratagraph [SP91] are useful for presenting index data to the user that might describe the contents of the video. Research has also shown that temporal browsing is useful as a way to quickly browse to large collections of video, and video databases should support these kinds of browsing.

Video Information Access and Management

Little research has been done in this respect. Current research, however, indicates that the video tape recorder abstraction is useful for many applications and should be implemented as a set of video player control functions in video databases.

5 THE VIDEOSTAR FRAMEWORK

In summary, multimedia data storage management is complex but essential for the development of multimedia information systems. Effective and easy-to-use design methods and tools for multimedia data storage management are an absolute necessity for further developments of such systems.

P. Bruce Berra et al., Guest Editor's Introduction
IEEE Trans. on Knowledge and Data Eng., Aug. 1993, [Ber93]

In the previous chapters, we have shown that video is a complex type of information. This complexity is a major concern for system developers when designing and implementing digital video applications even in the case where different tools and users are not expected to share video information. The situation becomes even more complicated in a shared environment where different users and tools would like to share video information in consistent ways.

In the last part of this thesis, we will discuss video information management in shared environments and what support a video database can give application providers and users in this respect. We will also present the VideoSTAR video database framework that has been implemented as an answer to our third research question from Section 1.4: *How can video information semantics be utilised by a video database framework for supporting users in entering, managing, querying, and viewing the various types of media and meta-data in a consistent way?*

This chapter will briefly present the major parts of the VideoSTAR framework developed by Hjelsvold et al. [HM95]. We refer to the original paper, included as Appendix C, for a more complete presentation of the framework.

5.1 DESIGN GOALS

In Section 1.3 we listed a number of reasons for storing video information in databases and thereby having more (centralised) control of how video information is accessed and manipulated. To facilitate sharing and reuse of video information, a video database must be able to provide a "standard" way for managing common characteristics of video information and – at the same time – allow specific applications to make their own additions or to put specific interpretations on the common part.

The three main design goals for the VideoSTAR framework have been:

1. To design a framework which provides a common view and understanding of video and meta-data and relations between different types of data, and which offers a set of common operations on these data;
2. to design a framework that explicitly supports sharing of video *and* meta-data in non-redundant and consistent ways; and
3. to develop a framework that can be implemented on top of existing database management systems.

The motivation for the third design goal lies in the main objective of this thesis: to propose functionality to be included in future DBMS'es for video information management. By building VideoSTAR on top of existing DBMS'es, we will have a better understanding of what extensions are needed to these DBMS'es.

5.2 THE VIDEOSTAR ARCHITECTURE

Figure 5.1 shows the architecture of the VideoSTAR framework. Applications access video and domain-independent meta-data via the application interface. Domain-specific meta-data – e.g., data associated with real-world objects – are stored outside the framework but are linked to the domain-independent VideoSTAR data. Applications have to access domain-specific data directly.

Media data and generic, domain-independent meta-data are stored in one of the four data repositories included in the framework. The VideoSTAR data model describes the elements stored in these repositories and the relations between data elements. The data model is generic and can be specialised and linked to a domain-specific model to fulfil the needs from specific application areas.

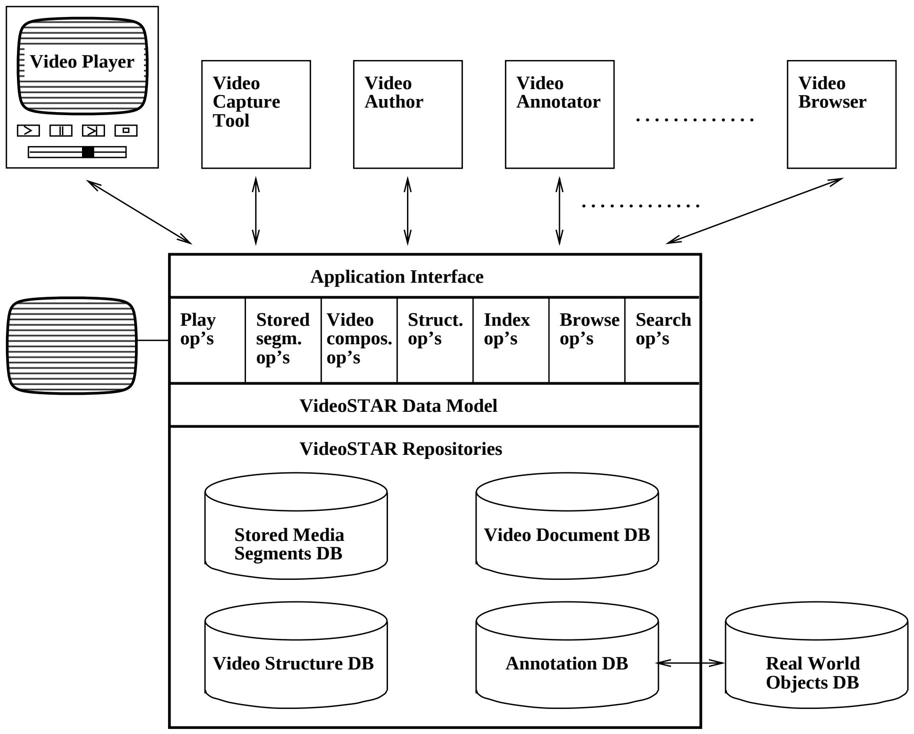

Figure 5.1 The VideoSTAR architecture

5.2.1 VideoSTAR Repositories

As shown in the figure, VideoSTAR defines four different repositories:

1. The *Stored Media Segment* repository stores uninterpreted multimedia objects – i.e., media data – together with media specific data and bibliographic data related to the segments.
2. The *Video Document* repository stores compositional data and bibliographic data related to the video documents.
3. The *Video Structure* repository contains a structural description of video documents.

4. The *Annotation* repository contains generic annotation objects which can be used to link user indexes, feature data, and supplementary information to pieces of video. The annotation repository is the primary link between real-world objects and the generic video information.

The VideoSTAR API implements operations needed to manipulate the contents of the repositories. Thus, the framework can then be implemented on top of different data storage systems such as file system files, relational DBMS'es, and object-oriented DBMS'es without affecting the applications.

5.2.2 A Generic Video Data Model

The generic data model provided by VideoSTAR captures common characteristics of video information and defines entities and relations corresponding to the types of meta-data discussed in Section 2.2.2, though version data is not explicitly modelled. The data model, which is discussed in more details in Chapter 6, will define a common format and understanding of the various types of meta-data. If used in a shared environment, it will define a "standard" way of representing video documents, structures, indexes, and the other types of meta-data.

5.2.3 Video Player API

As discussed in the first part of this thesis, digital video playback is a complex task involving video data decompression, synchronisation issues, and video data transfer. VideoSTAR offers a video player abstraction that hides this complexity from the applications. Via the video player API, a VideoSTAR application can instantiate a video player window and gain control over it. The video player implements all functionality needed for playing back video data. The video player API offers a set of commands that can be used by the application to control the replay and to request status information from the player (see Appendix C).

5.2.4 Video Information Management API

The video information management API offers operations for accessing – i.e., inserting, reading, updating, and deleting – contents of the VideoSTAR repositories. The video information management API provides a high-level interface hiding some of the complexity of the data model and ensuring integrity and consistency.

High-Level Interface

By offering a high-level interface, VideoSTAR can better separate the operational interface from the actual data model and data structures. In Section 4.2, we reviewed a number of composition data model that has been proposed because the timeline model is not very well suited as a *authoring* paradigm. The VideoSTAR architecture allows us to use the timeline model for storing compositional data while still being able to offer a structured authoring paradigm to the applications and the users.

Integrity and Consistency

A major concern for video databases is the size of the video objects. Video objects may become very large (even a low quality format such as MPEG-1 generates 11 MB video data per minute). It is, therefore, important that sharing of video information does not imply replication of video objects. If video data are not replicated, the same media segments may be shared among different documents. It is then important that the database controls individual users' access to access and modify shared media data, for instance by refusing users to modify shared media data to avoid that they modify other users' documents without their knowledge.

We have discussed integrity and consistency issues in more details in [Hje94b]. The only issue we will discuss in relation to this thesis is that of allowing various users to share index data in consistent ways. This will be discussed in much more details in Chapter 6.

Data Independence

From many applications' point of view, the actual format of the media data is not a major concern. A video annotator [HS93], for instance, requires the possibility to assigning pieces of text to specific intervals of video – no matter whether the video is compressed according to H.261, MPEG-2 or DVI. VideoSTAR defines abstract data types such as *media streams* and *stream intervals* (see Chapter 6) which are media data independent.

5.2.5 Querying and Browsing Interfaces

The querying and browsing interfaces offer operations for browsing the contents of VideoSTAR (`Browse op's`) and for creating and executing queries (`Search op's`). The main concerns for these operations in VideoSTAR are how to exploit temporal relationships in query formulation and how to control the degree of meta-data sharing. These issues will be discussed in much more details in Chapter 7.

5.3 CLAIMED CONTRIBUTIONS

As discussed in Section 2.2.2, a number of researchers have proposed various classes of meta-data. We claim that the VideoSTAR framework supports a wider range of such meta-data types than known video application frameworks. For instance, the framework proposed by Schnorf [Sch93] is mostly concerned with media and media-specific data, while the framework proposed by Grosky [Gro94] only makes the crude distinction between feature data and alphanumeric data.

We also claim that separating operational aspects from representational aspects – e.g., by providing a structured authoring paradigm on top of a timeline data representation – allows us to solve different problems independent of each other. This makes it possible to optimise solutions for sub-problems – e.g., by having a timeline data representation and still being able to provide a structured authoring paradigm.

5.4 FRAMEWORK EVALUATION

The reason for developing the VideoSTAR framework was to study what support a video database should offer to applications for managing, querying, and accessing various types of media and meta-data. Our goal has been to develop a framework that can simplify development of video database applications, a framework that specifically supports video information sharing, and a framework that can be built on top of existing DBMS'es.

Currently, the VideoSTAR framework stores meta-data in a “database” of ordinary text files, but we are working on two new versions of the framework:

One version running on top of the Oracle 7 relational DBMS [KL95] and one version running on top of the SHORE object-oriented DBMS [Gro95].

The following evaluation is based on the experiences made from implementing digital video archive tools (see Appendix F). To validate whether or not the goals have been met, we have to answer the following questions:

1. Does the high-level interface *really* support and simplify video application development?
2. Does the framework support the types of operations that users and applications would like to perform?
3. Does the framework support applications in sharing video information consistently?
4. Is it possible to implement the framework efficiently on top of existing database platforms?

5.4.1 Application Development

The question put forward is: *Does the high-level interface* really *support and simplify video application development?*

Our experiences regarding the different parts of the interface are:

- *Video player API.* It is a demanding task to develop a digital video player – especially when media data are stored remotely. The video player API, however, allows database applications to control video playback in an easy manner. Thus, video playback complexities can be hided for applications not requiring to access or manipulate media data directly. We experienced this when modifying the video player from reading local video files to receiving video data from the network; the extensive modification of the video player software needed to handle networking did not influence the video player API and the video tools.
- *Video information management API.* The video information management API supports applications in providing high-level interfaces (e.g., for video composition), in implementing integrity rules, and in providing video data independence. The current implementation is not very well fitted for evaluating these aspects because no editing tool has yet been developed to test

a high-level video composition API, only a few integrity rules are defined, and video data exist as JPEG video data only.

- *Querying and browsing API.* The VideoSTAR algebra implements rather complex functionality (3-4000 lines of C++ code) that can be directly accessed by applications through an algebra operation interface for creating and computing video information queries.

5.4.2 API Appropriateness

The question put forward is: *Does the framework support the types of operations that users and applications would like to do?*

Our experiences regarding the different parts of the interface are:

- *Video player API.* The tools that we have implemented do not require – nor want to – access media data directly, but they require functionality for controlling video playback. Our experience is that VideoSTAR offers the set of control functions that are needed to control video playback.
- *Video information management API.* Due to the implementation status, we have not been able to evaluate this part of the interface.
- *Querying and browsing API.* VideoSTAR offers a pure algebra interface. Most users would not want to formulate queries on the algebra level directly. The current version of VideoSTAR does not implement query language interface nor a query optimiser. Thus, usable query tools would have to implement a query language themselves, if requested by users.

5.4.3 Video Information Sharing

The question put forward was: *Does the framework support applications in sharing video information consistently?*

Due to the status of the video information management API, we have not been able to test consistency and integrity issues. However, we have experienced that the generic data model provides a common view of video information that allows different tools to share and exchange video and meta-data.

5.4.4 Portability

The question put forward is: *Is it possible to implement the framework efficiently on various database platforms?*

The current version is implemented on top of a dedicated, file system based database. The next version that is currently under development, will also be available on top of a relational DBMS and on top of an OODBMS. The DBMS'es are treated as *persistent stores* and only a small fraction of the DBMS' functionality is used to implement the VideoSTAR data model and to access specified entities in the database. The experiences achieved in the design of the various versions indicates that the interface between applications and VideoSTAR can be the same independent of the actual DBMS and that only a few methods have implementations that are DBMS specific.

Due to the implementation status, we have not gained enough experiences on different database platforms to evaluate how efficiently VideoSTAR can be implemented.

6 A GENERIC VIDEO DATA MODEL

But where is the meaning? The meaning is not entirely in the data model and it is not entirely in the situation being modelled – it lies somewhere between the two and cannot be located precisely. You might argue that once a situation has been described in a data model then others could use this "objective" description as the basis for computer system development. Unfortunately the act of reading a data model is itself interpretive and subject to the same difficulties experienced when the modeller interpreted the situation to produce a data model.

C. de Carteret and R. Vidgen, Data Modelling for Information Systems, 1995 [dCV95]

Any information system, including video information, needs a video data model as a basis for interpreting the various types of media and meta-data (see Section 2.2.2) that this system is intended to manage. The VideoSTAR framework is based on a data model that captures the generic characteristics of video information and which can be included as part of domain-specific models. This data model has been developed as an answer to our second research question from Section 1.4: *How can video information semantics be captured in a generic data model that can be extended and tailored to the needs of specific users and application domains?*

Predecessors of the VideoSTAR model have been presented by Merok [Mer93] and Hjelsvold [Hje94c] (see Appendix A). Variants of the models have been presented by Hjelsvold et al. [HM94, HM95] (see Appendix B and C). In this chapter we will review and discuss the proposed model. We refer to the given appendices for more information.

6.1 MODELLING GOALS

The data model is aimed at modelling the generic characteristics of video information. Specifically, this means that the model will provide means for representing the types of meta-data presented in Section 2.2.2 (though version data is not included, yet). The data model is especially concerned with issues related to shared environments (see Section 2.5):

1. It should allow application domains to extend the model with domain specific concepts, and it should allow application domains to implement only parts of the model. For instance, users should be able to use the indexing mechanisms even if all video documents are preauthored and stored as one physical media segment – i.e., the exact composition is not known. This is not easily achieved in OVID [OT93], for instance, where indexing is an integrated part of composition.
2. It should allow users to share meta-data, such as indexing information, in well-structured way – e.g., it should be possible to make a distinction between content indexes that is document-independent from indexes that are relevant for specific video documents only and, thus, allowing documents to share document-independent indexes. This issue will be further discussed in Section 6.2.3.

6.2 CHARACTERISTICS TO BE MODELLED

Figure 6.1 shows the various types of data that are modelled and illustrates how these types relate. The model includes mechanisms for representing *documents and structures*, *media data and compositions*, and *index anchors*. Enhanced Entity-Relationship diagrams [EN94] will be used in the rest of this chapter to illustrate how these mechanisms are modelled. The complete VideoSTAR data model proposed by Hjelsvold et al. [HM95] is presented in Appendix C.

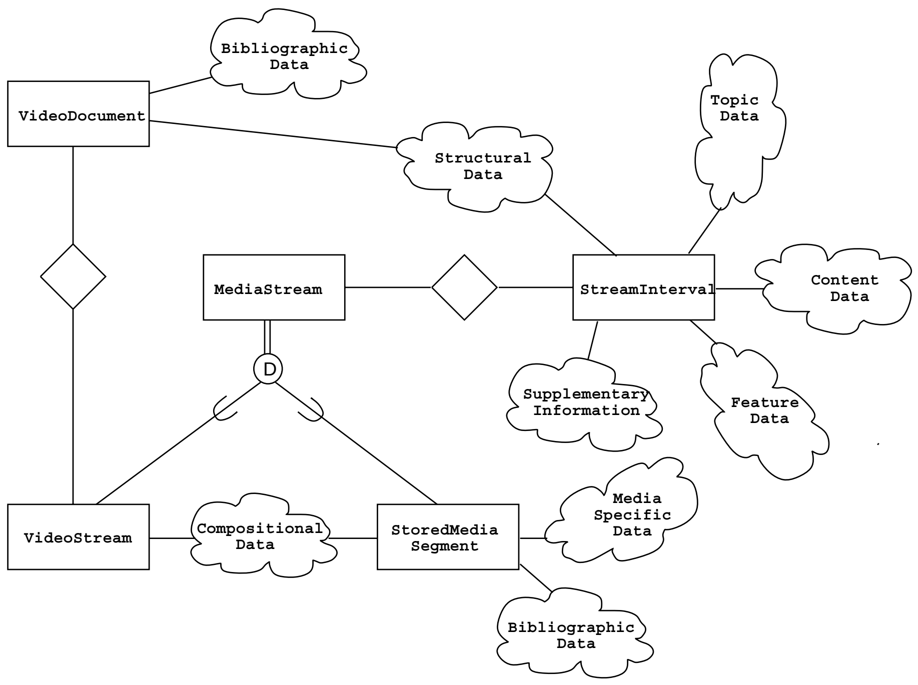

Figure 6.1 Overview of the generic VideoSTAR data model

6.2.1 Document Composition and Media Data

The main difference between the model presented in [HM94] (Appendix B) and [HM95] (Appendix C) is the way compositional data are defined. The following discussion is based on the latter version. In the core of the model, we find a specialisation hierarchy of continuous media objects (see Figure 6.2). The continuous media object entity is the top-element of this hierarchy and defines the common characteristics of continuous media data, such as play operations (see Section 5.2.3). All `CMObjects` have attributes defining the starting and ending times for their defined time intervals.

The model makes a distinction between virtual video streams (the `VideoStream` entities) and video/audio recordings (specialisations of `StoredMediaSegments`).

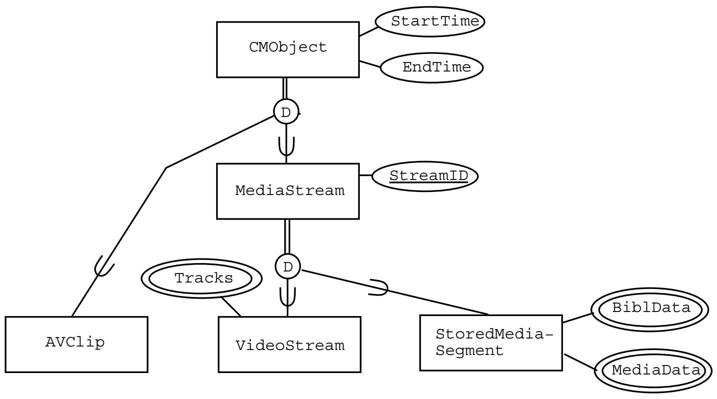

Figure 6.2 EER diagram for the continuous media object hierarchy

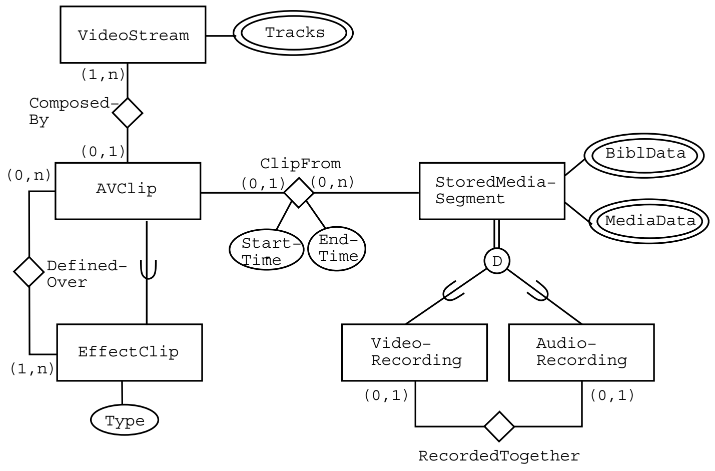

Figure 6.3 EER diagram for document composition

All these entities are, however, specialisations of the `MediaStream` entity. The `MediaStream`s are important components because each of them defines a time coordinate system identified by the `StreamId` attribute. (In the complete model shown in Appendix C, this attribute is incorrectly shown as an attribute of `CMObjects`.)

Referring to Section 4.2, VideoSTAR implements a timeline approach: The contents of a video document is represented by a logical `VideoStream` which may be composed of pieces from several video and audio recordings. Each recording is stored as contiguous sequence of audio samples stored in `AudioRecordings` and video frames stored in `VideoRecordings`. The mapping between the logical `VideoStream` and `StoredMediaSegment`s is represented by audio-/video-clips (`AVClip`). `AVClip`s are grouped into tracks within the `VideoStream` in a way similar to the approach used in QuickTime [App93].

During production the author may want to add sound to the original sound – e.g., by adding music or commentaries. When other authors want to reuse parts of a video document, they may want to add other sound tracks to the original sound recording. Therefore, it is useful to store relations between audio and video recordings that were (originally) recorded together. The `RecordedTogether` relation can be used to represent this information.

Images, text and other non-continuous data may have a temporal dimension within a `VideoStream` – i.e., they may be displayed to the user in a time interval during the playback. This is not yet covered in the model, but the model can be extended by having `AVClip`s relate to non-continuous data instead of audio or video recordings.

Film and video makers are using several effects in the transition between shots. To preserve meta-information related to the original recordings used to create an effect transition, the VideoSTAR model defines a special type of `AVClip`s – the `EffectClip`. Each effect clip keeps a relation to the `AVClip` from which it is derived. An `EffectClip` may, or may not, be stored explicitly as a `StoredMediaSegment`.

It is not necessary to store effects as separate segments if special play operations are implemented that are capable of generating video effects during playback. By supporting separate storing of effects, however, VideoSTAR can store all effects that can be generated by any editing tool without requiring the video player to be able to generate them during playback.

6.2.2 Documents and Document Structure

The VideoSTAR data model provides the "traditional" way to structure a video document into a hierarchy of *shots*, *scenes*, and *sequences*. Furthermore, the data model allows the user to define a hierarchy of an arbitrary number of *super-sequences* (in earlier papers called *compound units*) on top of this. Figure 6.4 shows the structure part of the data model. (The model is slightly modified from the complete model shown in Appendix C in that more attributes have been attached. (To make the figure more readable, the media stream generalisation hierarchy has been short-cut, and the `StreamInterval` is linked directly to a `VideoStream`).

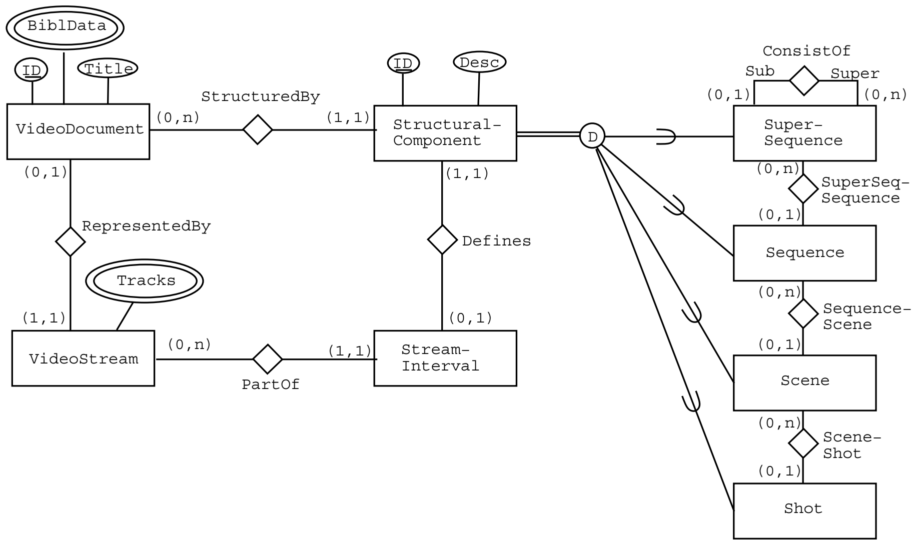

Figure 6.4 EER diagram for the document structure

In the model, video documents are represented by a video stream and are structured by structural components. The model defines four specialised types of structural components, as already described. Each structural component is related to one contiguous stream (time) interval from this video stream.

As seen from the diagram, the generic model does not put strict requirements on the presence of the a structural hierarchy. Components may be created at all levels without having to create components at the next higher or next lower level. This allows for great flexibility in defining structures that are useful for the applications. In addition to the data model, integrity rules are defined so that the stream intervals of adjacent components in the hierarchy are consistently defined – e.g., that all shots within a scene have stream intervals that are within the temporal range of the scene's stream interval.

6.2.3 Anchors for Video Indexing

In Section 4.4 we discussed needs for annotating and indexing the contents of a piece of video. Inspired by this, we have concluded that it should be possible to index/annotate arbitrary pieces of video independent from document structures. Indexing information should also be orthogonal to compositional structures.

In the VideoSTAR model, one can annotate an arbitrary stream interval from a `MediaStream` – i.e., a piece of video from an edited video document or a piece of video/audio recording – by defining the `StreamInterval` of interest and establishing a relationship between it and the relevant `Annotation`. Figure 6.5 shows the indexing part of the data model.

This diagram is taken from the complete model in Appendix C except that the linking to domain-specific information is made more explicit by introducing the `DomainAnnotation` entity and the `LinkedTo` relationship that explicitly links real-world objects to generic `Annotation` objects.

6.2.4 Model Support for Sharing and Reuse

Sharing and reuse is supported in that a composition can be stored as compositional data defined over a set of stored media data that can be used as part of multiple video documents. This is favourable both because it will reduce the storage demands (which are huge anyway) and because it makes it easier to preserve the consistency of the database.

An even more important advantage of the model is that it facilitates sharing of indexing data because indexes can be related to *either* video document streams *or* media data. If all indexes where related to the specific documents only, it would be difficult to decide whether a specific index would also be applicable to other documents using the same video data. (OVID solves this problem by

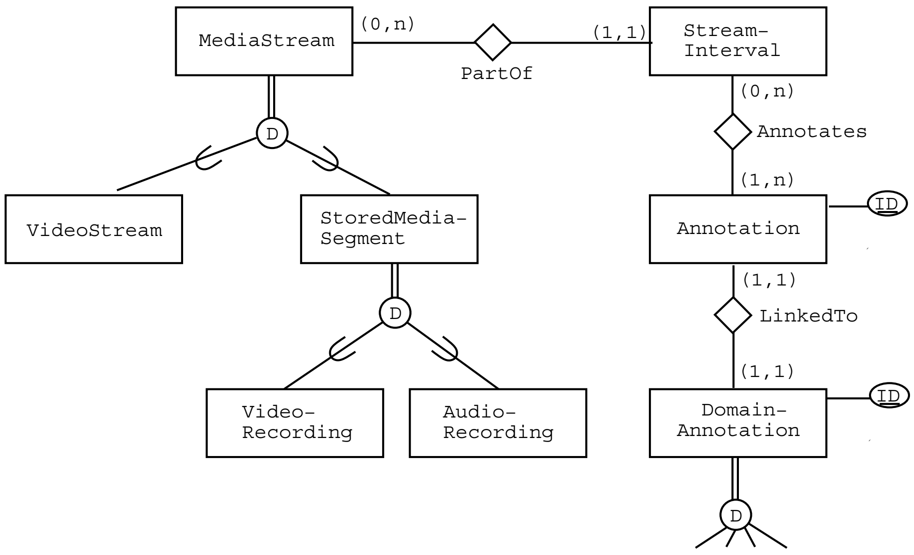

Figure 6.5 EER Diagram for index anchors

typing a video object's attribute as public – i.e., inherited by other documents – or private – i.e., only applicable for the specific object). In the VideoSTAR model, indexes that are document-independent can be related to stored media segments while indexes that are only valid for a specific document can be related to this document's video stream. This is further exploited in the definition of *contexts* and querying/browsing operations (see Chapter 7).

6.3 DOMAIN-SPECIFIC EXTENSIONS

The proposed model is a kernel model that must be tailored to the needs of specific applications domains – e.g., a television news archive. First, the components defined in the model must be given an interpretation useful to the application domain – e.g., by interpreting `Sequence`s as news items in the television news example. Second, specific application domains must define a domain-specific and a real-world model, and link the real-world model to the VideoSTAR model.

6.4 CLAIMED CONTRIBUTIONS

There exists a number of data models for video information. Still, we claim that the VideoSTAR data model contributes to the understanding of how video databases should be organised. We claim that the way the model is organised into independent parts, gives the users greater flexibility in adapting the model to their domain. For instance, assume that a user wants to make personal annotations to full-length movies that have been digitised from analogue tapes. This video document may consist of hundreds of shots and a number of scenes and sequences and it would, originally, be a composition of a large number of recordings. The solutions chosen in OVID [OT93] and Algebraic Video Systems [WDG95], for instance, combines composition and annotation in ways that are unsuited for this user. In VideoSTAR, composition and indexing/annotation are modelled as orthogonal features, and any part of the video stream can be indexed independently of the document composition.

We also claim that the timeline approach is a reasonable choice. The critics being raised against the time-line approach – e.g., by Little [Lit94] – are all concerned with problems related to multimedia (and video) editing. The absolute binding to playout time is the main cause for these problems. We have chosen the time-line approach in spite of these objections because (see also Section 4.2):

1. Video editing is not the only application of a video database.
2. The time-line approach introduces less complexity for playout and querying than the other approaches.
3. Editing is less time-critical than playout and querying and, thus, emphasis should be given to playout and querying at the expense of editing, if necessary.

The disadvantages are compensated by the high-level VideoSTAR API for video information management, as discussed in Section 5.2.

The two-level composition approach have been chosen deliberately to avoid unnesting an arbitrary number of levels during replay and querying. As described in Section 4.2, the opposite choice have been taken in OVID [OT93] and Algebraic Video System [WDG95] which will give the authors of video or multimedia documents greater flexibility in reusing composite components and which will handle document dependencies more directly. VideoSTAR compensates this by offering a set of high-level operations to handle reuse of composite components.

We claim that the model is a sound basis for separating topic annotations and other document-specific indexes from sensory, domain-independent indexes. This is further discussed in Chapter 7.

6.5 END-USER EVALUATION

If applications are to share information, they must also share a common interpretation of the shared data. The motivation behind the VideoSTAR model is to have a generic model that can capture common characteristics of video data and that can serve as a basis for applications to share media and meta-data. Our goals have been that it should be easy to adapt the generic model to specific application domains and that it should provide means for meta-data sharing. To evaluate whether or not these goals have been met, we have to test the data model in end-user application domains. The following questions are discussed in Chapter 8:

1. Does the data model really capture the important characteristics of video information?
2. How easy is it to tailor the model to specific application domains: How easy is it to implement only parts of the model and how easy is it to attach domain-specific models to the generic model?
3. How useful are the mechanisms for meta-data sharing?
4. Is the two-level composition powerful enough for representing video document sharing?
5. What are the consequences regarding efficiency on user and database operations?

7 QUERYING AND BROWSING VIDEO INFORMATION

Internet is just like sitting in an Italian restaurant having only an Italian menu. You know that there are several dishes you would love to order; you just don't know which they are because their names don't reveal their contents.

Peter Deutsch, at the Network Services Conference, Pisa, Italy, October 1993.

Content-based video querying and browsing is a highly desirable feature in video databases – as in any other database system. The complexity of video information makes video querying and browsing demanding: Scene (i.e., image) complexity makes it difficult to describe the contents; the temporal aspect of video information is a dominant property that needs specific consideration; and composition establishes relations between video documents and video data that might be shared by other documents as well.

In this chapter we will describe VideoSTAR functionality for querying and browsing. The querying and browsing capabilities have been developed as answers to our fourth and fifth research questions from Section 1.4 – i.e., how compositional data can be used in querying and browsing, and how temporal relationships can be exploited in content-based querying.

Several VideoSTAR papers discuss various aspects of this issue. We refer to these papers included as appendices to the thesis for more details: Hjelsvold et al. [HM94] (see Appendix B) discuss use of temporal and compositional relations in video queries; Hjelsvold et al. [HM95] (see Appendix C) also discuss operations for browsing; and Hjelsvold et al. [HMS95a] (see Appendix D) we present the temporal foundation of the VideoSTAR query mechanisms. A more complete discussion of the issues is presented in Hjelsvold et al. [HMS95b] (see Appendix E).

7.1 DESIGN GOALS

VideoSTAR addresses the temporal and compositional complexities in video information querying/browsing. It is outside the scope of this thesis to invent new querying/browsing mechanisms for handling image complexity. Based on the conclusions in Section 4.8, querying and browsing mechanisms in VideoSTAR have been designed to fulfil the following goals:

1. Query mechanisms should allow users to exploit temporal relationships when formulating queries.
2. Query mechanisms should allow users to define the search scope when media and meta-data are shared.
3. Browsing mechanisms should allow users to browse a document's structure.
4. Browsing mechanisms should allow users to browse media or meta-data that are relevant to specific parts of a video document.
5. Browsing mechanisms should allow users to define the browsing scope when media and meta-data are shared.

7.2 SHARING META-DATA THROUGH CONTEXTS

In Chapter 2 we referred experiments performed by Soviet film makers that proved that the meaning and contents of a piece of video is not only depending on what is shown in the piece itself. Information is also communicated in the way that individual pieces of video are combined within a video document. Thus, some index data may only be valid within one document's context while other index data may be related to what is actually shown in the frames and will be valid for all use of this piece of video.

In a shared environment, it is necessary to know what index data are generally valid and what index data are specific for given documents. The VideoSTAR data model allows indexes to be related to intervals from video document streams *or* to intervals from stored media segments. This feature can be used for differentiating between general and document-specific index data by organising index data this way:

- Indexes that are related to stored media segments are only those indexes that are valid independent of specific document contexts; while
- indexes that are devoted to one specific video document are related to this specific document's video stream.

This allows us to define three different classes of meta-data with varying relevance and uniqueness for a given document (see also Figure 7.1):

- The *primary context* contains meta-data that are specifically relevant and unique for the given document. Primary context meta-data are related to the document's video stream. The topic of a television news item will, for instance, be part of news document's primary context.
- The *basic context* contains meta-data that are relevant – but not unique – to the given document. Basic context meta-data are equally relevant for all other documents into which the media data have been used, and are therefore related to the stored media segments. Names of persons shown in the video are examples of basic context meta-data.
- The *secondary context* is defined when the same media data is shared among several documents. One document's primary context is part of the secondary context of the other documents sharing the media data. Figure 7.1 shows an example of two documents sharing media data. In this example, *primary context II* constitutes the secondary context of *document I* and vice versa.

When a stored media segment is shared among several video documents, the basic context meta-data related to this media segment is also shared. Thus, basic contexts motivate for a consistent and information preserving sharing of meta-data: consistent because document-independent indexes are stored only once; information preserving because whenever new data are added to the basic context, these data will automatically be visible to all documents using the media segment. On the other hand, primary contexts are assumed to be document-dependent and, thus, meta-data in one document's primary context

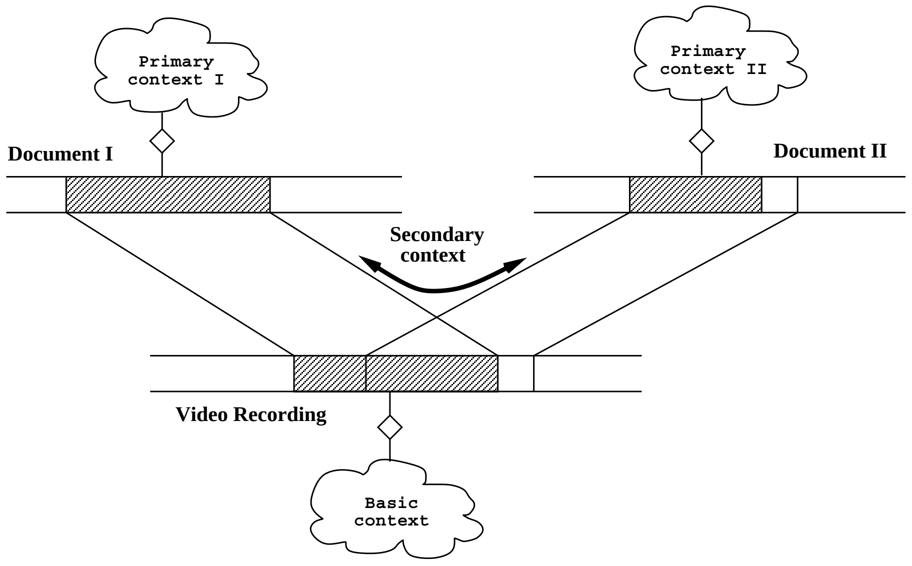

Figure 7.1 Relation between primary, basic, and secondary context

will not be intermixed with meta-data from other document's contexts even if the documents share media data. Secondary context, however, allows users to mix primary context meta-data from documents sharing media data on demand.

7.3 QUERYING AND BROWSING BASICS

In Hjelsvold et al. [HMS95a] (see Appendix D) we presented the VideoSTAR algebra for video querying and browsing. Table 7.1 gives an overview of the operations defined in the algebra. In the following subsections we will briefly review this algebra as a basis for querying and browsing.

Set Operations	
A `AND` B	Returns elements present in both A and B
A `OR` B	Returns elements present in A or B
A `MINUS` B	Returns elements present in A minus those also present in B
Temporal Set Operations	
A `tAND` B	Returns elements representing intervals from elements in A which intersect elements in B
A `tOR` B	Returns elements representing the merge of intervals from elements in A and B
A `tMINUS` B	Returns elements representing intervals from elements in A which do not intersect elements in B
Filter Operations	
A `tREDUCE(temp_rel)` B	Return those elements in A having the given temporal relation to at least one element in B
Compositional Operations	
`Decompose` A	Maps elements in A onto basic context
`MapToComposition` A	Maps elements in A onto primary context
`MapToStream(stream)` A	Maps elements in A onto the given stream's primary context
Browsing Operations	
`ANNOT(type)` A	Retrieves all annotations of the given type having stream intervals intersecting elements in A
`STRUCT(type)` A	Retrieves all structural components of the given type having stream intervals intersecting elements in A

Table 7.1 Querying and browsing algebra operations

7.3.1 Mapped Video Object Sets

The algebra is defined over a special type of sets that are called *mapped video object sets*. Elements in a mapped video object set have two parts where the first part, *ObjRef*, is a reference to an object in the video database while the last part, *StreamInt*, contains the stream interval (identified by the identity of the stream, and start and end times for the interval) that this object is mapped onto.

7.3.2 Select Operations

It is assumed that an application domain specific video database provides select operations that will retrieve objects from the database having given (real-world) properties, create mapped objects corresponding to the retrieved database objects, and insert them into mapped video object sets that can be used as input to VideoSTAR algebra operations.

7.3.3 Set Operations

Intersection, union, and difference of mapped video object sets are defined in the normal way where two elements are defined to be equal if they refer to the same object over the same interval. As noted by Clifford and Crocker [CC93], normal set-theoretic operations will not combine elements having intersecting stream intervals. The VideoSTAR algebra defines temporal variants of these operations that return the intersection, union, and difference of the *stream intervals* of the input sets. For instance, given two sets A and B, the normal set-theoretic intersection `AND`, and the temporal variant `tAND`, A `AND` B will return the elements in A that are also elements in B. On the other hand, A `tAND` B will return a set of elements identifying the stream intervals where elements from A and B intersect.

7.3.4 Filter Operations

The algebra defines a filter operation, `tREDUCE`, that compare elements of one set with elements of a second set to return those elements in the first set that have a given temporal relationship to at least one element of the second set. Any of Allen's 13 temporal relationships [All83], for instance intersects, can be used in the filter operation: Given two sets A and B, the operation A `tREDUCE(intersects)` B will return the elements in A whose stream intervals *intersect* with elements in B.

7.3.5 Compositional Mapping Operations

Composition establishes relations between documents' video streams and stored media segments. A stream interval from one document's video stream will correspond to a set of stream intervals from stored media segments and vice versa. Compositional mapping operations use compositional relations to map objects from one type of context to another. The *Decompose* operation maps objects in the input set from the primary context onto the basic context. The *MapToComposition* operation maps objects from the basic context onto the corresponding primary contexts. The *MapToStream* operation maps objects onto one specific stream. Figure 7.2 (adopted from Hjelsvold et al. [HMS95b]) shows how one primary context index is decomposed into two basic context indexes by *Decompose* and how one basic context index is mapped to two video streams by *MapToComposition*.

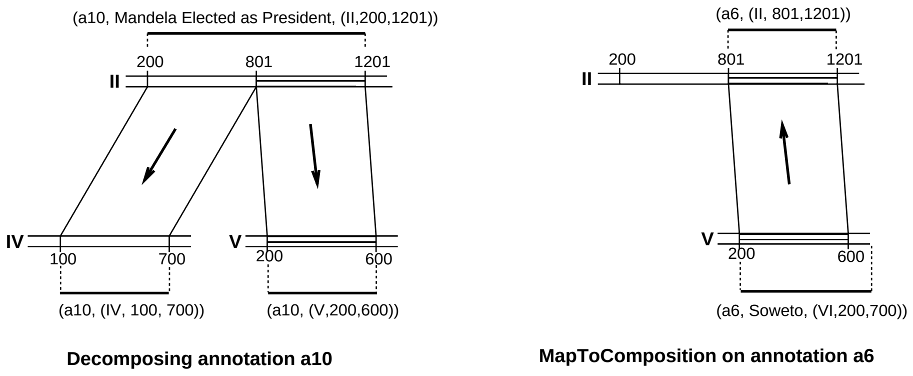
(a10, Mandela Elected as President, (II,200,1201))
200
801
1201
II
IV
100
700
V
200
600
(a10, (IV, 100, 700))
(a10, (V,200,600))
Decomposing annotation a10
(a6, (II, 801,1201))
200
801
1201
II
V
200
600
(a6, Soweto, (VI,200,700))
MapToComposition on annotation a6

Figure 7.2 Compositional mapping operations (adopted from Hjelsvold et al.)

7.3.6 Browsing Operations

Two operations are defined specifically to support meta-data browsing. The *ANNOT* operation takes one input set and one annotation type as arguments and returns all annotations of the given type that intersects any of the elements in the input set. In a similar way, the *STRUCT* operation returns structural components of a given type.

7.4 VIDEO QUERYING

In the following subsections we will shortly discuss how the VideoSTAR algebra can be used in query processing. This issue is further discussed in [HMS95a] (see Appendix D) and [HMS95b] (see Appendix E).

7.4.1 Query Processing

This subsection is taken from Hjelsvold et al. [HMS95b] (see Appendix E).

Query processing usually involves parsing a query, breaking it into basic (often algebra-based) operations, determining – and possibly optimising – a query plan defining the sequence of basic operations, and performing this plan. The current version of VideoSTAR offers a pure video query algebra interface that allows us to test the usefulness of the algebra discussed in Section 7.3 without having to implement a complete query processor. Figure 7.3 gives an overview of the four steps that an application has to go through when requesting VideoSTAR to process video queries.

Step 1: The application retrieves objects from the VideoSTAR repositories and inserts them into appropriate input sets for the query processing.

Step 2: The application instructs VideoSTAR to create the query graph with the corresponding operations. VideoSTAR will return a reference to each node that can be used by the application to access the node – e.g., for modifying the operation or for retrieving its result set.

Step 3: The application instructs VideoSTAR to perform the computation. The VideoSTAR *Algebra Operation* module computes the operations one-by-

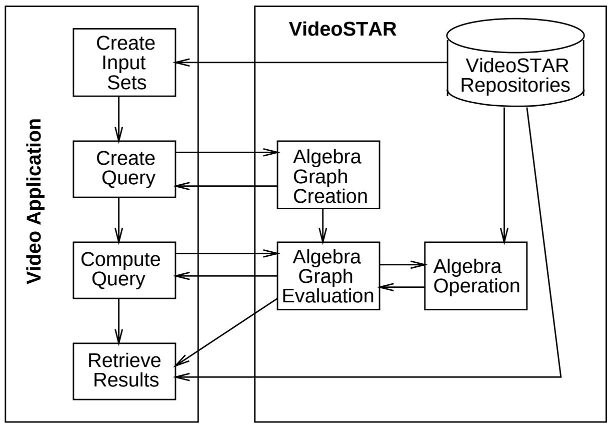

Figure 7.3 Query Processing Steps

one in the sequence defined by the user application. The intermediate results are explicitly stored in each node and are used as input sets to other operations.

Step 4: The application accesses the appropriate node in the query graph and retrieves the corresponding result set.

The VideoSTAR repositories are used in the first step for selecting input sets, they are accessed by the *Algebra Operation* module to perform annotation, structure, and mapping operations, and they are accessed in the last step when the application retrieves the resulting objects themselves.

7.4.2 Combined Predicates

In the rest this section we will use examples from a television news archive domain. Assume that the domain specific part of the video database implements a *MapSelect* operation that creates a mapped video object set based on a specification of an application domain specific annotation database, a query predicate, and an indication of which type of context – i.e., basic or primary – to search. The following piece of pseudo code illustrates how this operation is intended to be used to create a set of mapped video objects which show president Bill Clinton – i.e., the objects will be parts of the basic context – assuming that person annotations are stored in PersonDB:

```
S1 = MapSelect(PersonDB, Name="Bill Clinton", Context="Basic")
```

The *MapSelect* operation will identify all basic context annotations on Bill Clinton from PersonDB. The operation will create one mapped video object for each of these annotations by putting the annotation's key/identity into the *ObjRef* part and by copying the related stream interval into the *StreamInt* part. These mapped video objects will be inserted into the mapped video object set that is returned by the operation.

Assume that the user is searching for pieces of video showing president Bill Clinton together with president Boris Jeltsin:

```
S2 = MapSelect(PersonDB, Name="Boris Jeltsin", Context="Basic")
S3 = S1 tAND S2
```

In VideoSTAR, each annotation is related to one specific stream interval, and thus, there is no direct relation between the elements of *S1* and *S2*. The only way to find the pieces of video that are common to the two sets, is to compute the stream intervals where elements from *S1* and *S2* intersect. The temporal intersection operation (`tAND`) performs this computation.

7.4.3 Querying Shared Data

Assume further that the user is searching for complete news items which contain pieces of video showing the two presidents together, and that news items are represented as structural *sequence* components:

```
S4 = MapToComposition(S3)
S5 = STRUCT(Sequence) S4
```

S3 contains elements that are part of basic context which is common to all documents using a piece of video. To identify the specific news items, one first has to map the elements in *S3* to the time systems of the documents into which they appear (see Figure 7.4). Then, one has to retrieve the *sequence* components from the structure database which intersect these elements.

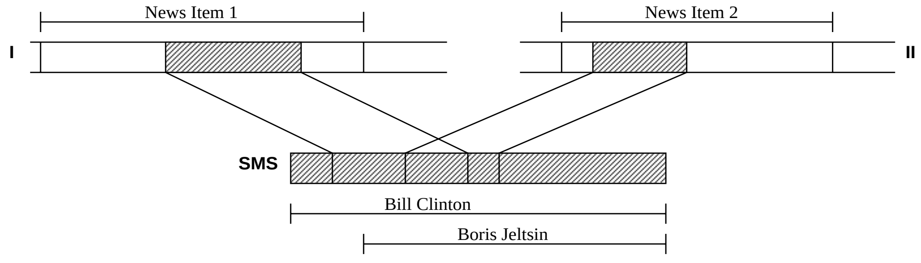

Figure 7.4 Two news items sharing basic context

7.4.4 Temporal Relationships

Assume that the user is specifically interested in those news items showing the two presidents together discussing the topic nuclear weapons:

```
S6 = MapSelect(KeywordDB, Keyword="nuclear weapons",
               Context="Primary")
S7 = S5 tREDUCE(intersects) S6
```

S7 is created by using the filter operation *tREDUCE(intersects)* to identify the news items in *S5* which intersect annotations from a keyword database related to the topic nuclear weapons.

Assume that a mass media researcher is searching for news items showing the two presidents together which end with video showing Bill Clinton only:

```
S8  = MapToComposition(S1)
S9  = MapToComposition(S2)
S10 = S8 tMINUS S9
S11 = S5 tREDUCE(finished-by) S10
```

S10 contains pieces of video document streams showing Bill Clinton but not showing Boris Jeltsin. *S11* is created by using the filter operation with the *finished-by* temporal relationship as filter function to identify the news items in *S5* which end with video showing Bill Clinton but not Boris Jeltsin.

7.5 VIDEO BROWSING

VideoSTAR offers two video browsing approaches: document structure browsing which allows the application to browse the hierarchical document structure; and content browsing which allows the application to retrieve meta-data related to specific pieces of a video document. VideoSTAR browsing mechanisms have been discussed in [HM95] (see Appendix C), in [HMS95a] (see Appendix D), and in [HMS95b]) (see Appendix E).

7.5.1 Document Structure Browsing

Structural data can be used to generate a *table of contents* for the video document. The table of contents gives the user an overview of the contents of a video document and can be used to navigate within the document. Assume that the user of a television news archive wants to browse the structure of the evening news broadcast September 30, 1995. The application can retrieve the structural description of this document by:

```
V1 = MapSelect(NewsDocDB, Title="Evening news" AND
               Date="30/09/95")
S1 = STRUCT(ALL) V1
```

The *MapSelect* operation retrieves the evening news from the news document database and its corresponding video composition stream interval and returns the mapped video object set *V1* that contains one element. The element's *ObjRef* contains the reference to the news document while the *StreamInt* contains the stream interval defined by the news document's video stream. The operation *STRUCT(ALL)* retrieves all structural components defined over this stream interval. This set can be used by the application to generate the table of contents. The VideoSTAR API also offers operations for navigating the hierarchical structure [HM95] (see Appendix C).

7.5.2 Browsing Contents Data

In VideoSTAR, contents browsing allows the user application to identify annotations from the database intersecting a given interval of a video stream. Assume for instance, that a user of the television news archive wants a list of all topics being discussed in the evening news identified by *V1*:

```
  A1 = ANNOT(Keyword) V1
```

A1 identifies all keyword annotations defined over the document's video stream and can be used to generate a list of topics.

The previous example shows how the application can browse the *primary* context of a video document. The VideoSTAR algebra operations can also be used to browse *basic* and/or *secondary* contexts as well. For instance, assume that a user wants to generate a list of names of persons being shown in the evening news – i.e., browse basic context – and that *SId* is the identity of the news document's video stream:

```
  V2 = Decompose V1
  A2 = ANNOT(Person) V2
  A3 = MapToStream(SId) A2
```

V2 identifies all stored media segment intervals being used in the video document. *A2* identifies all person annotations defined over these intervals while *A3* contains these annotations mapped onto the primary context of the document.

Assume that a media researcher is watching a video document and finds a piece of video that seems interesting and that this piece of video is identified by a one-element mapped video object set *Vi*. He/she might want to know what other documents this piece of video has been used in and what the topics of these documents where – i.e., he/she wants to browse the secondary context of the video document.

```
  V3 = Decompose Vi
  V4 = MapToComposition V3
  A4 = ANNOT(Keyword) V4
  A5 = Decompose A4
  A6 = MapToStream(SId) A5
```

After decomposing *Vi* and mapping it onto video document streams again, *V4* identifies the intervals constituting the secondary context for *Vi*. *A4* identifies all keyword annotations in the secondary context which are then mapped onto the primary context again (*A6*).

7.6 CLAIMED CONTRIBUTIONS

In a shared database where the same pieces of stored media segments can be used in several video documents, it is important to have mechanisms that allow sharing of index data as well. Oomoto et al. [OT93] and Weiss et al. [WDG95] have proposed schemes where index data are inherited along a composition hierarchy. These schemes assume that it is up to the author to decide what index data one specific video object should share/inherit. We claim that the definition of *contexts* and *mapping operations* is a better approach because this gives the querying user the freedom to decide to what extent index data should be shared. In this chapter we have shown how contexts and mapping operations can be utilised in querying and browsing.

From a user's point of view, structural and indexing data serve different purposes. VideoSTAR handles these concepts independently and specific operations are provided to better support the purposes these data serve: Distinct operations are provided to retrieve the indexes related to a piece of video, to retrieve the structural description, and to browse the structure hierarchy. Current systems – i.e., those systems reviewed in Chapter 4 – provide only a subset of this functionality.

The temporal properties are important aspects of video-related data. VideoSTAR provides a rich algebra based on temporal databases that can be used to exploit temporal relationships when formulating queries. Rowe et al. [RBE94] have shown how temporal relationship functions can be included in POSTGRES, and Hibino et al. [HR95] have shown how temporal relationships can be defined in a graphical interface, but no known system has defined a generic query algebra incorporating temporal relationship operations.

7.7 QUERY ALGEBRA EVALUATION

A major concern for a video database based on the VideoSTAR principles is whether the query algebra can be efficiently implemented. In the current version, the mapped video object sets are implemented as lists sorted on the stream interval part of the elements. As noted by Leung et al. [LM93], sorted lists allow stream oriented processing of temporal algebra operations. There is, however, still much research to do to investigate how the algebra can be efficiently implemented – especially in large video databases.

A second concern is whether automatic, feature-based querying and browsing can be added to VideoSTAR. We see two ways in which VideoSTAR can be combined with feature-based tools:

1. Cut-detection tools can be used on the media-data to automatically detect shot transitions in a media segment that is a composition of shots. This information can be entered into VideoSTAR as structural data to make it available for other video archive tools.
2. A specific application domain may define a set of domain-specific annotations for storing feature data. These annotations can be related to individual frames or sequences of frames as any other type of annotation. If feature matching tools are implemented by the application developer to retrieve pieces of video having specific features, VideoSTAR algebra operations can be used to exploit temporal relationships and context operations on top of feature extraction.

7.8 END-USER EVALUATION CRITERIA

Temporal relationships can be defined among meta-data, such as index and structure data, that are related to specific intervals of media streams. The VideoSTAR querying and browsing algebra provides means for exploiting temporal relationships in query formulation. In addition, querying and browsing in VideoSTAR is based on the context concepts and compositional mapping operations are provided for controlling the context(s) to be searched and browsed. To evaluate the querying and browsing mechanisms in VideoSTAR, we would like to have end-user response to the following questions:

1. How useful is context handling in practice?
2. Are the query facilities rich enough to express the queries that a user wants formulate regarding temporal properties?
3. Is the complexity of the algebra really necessary?

These questions will be discussed in relation to the end-user experiments in Chapter 8.

8 RESULTS

Multimedia.
I didn't know what it was 10 years ago. I don't really know what it is today, and no one else does either. Show me some value*! Solve a problem for me.*

George Colony, WIRED, March 1995 [Bue95]

The aim of this thesis has been to study video as a new data type for databases and to develop database functionality that supports users and applications in managing video data. The success of this thesis and the VideoSTAR prototype lies in how well VideoSTAR takes care of the needs of video database users. This chapter summarises the experiences we have made from discussing VideoSTAR issues with end-users, from demonstrating prototype VideoSTAR applications to end-users, and from using prototype tools on sample user data.

8.1 EXPERIMENTAL ENVIRONMENT

We have implemented an integrated video archive environment to be able to perform VideoSTAR user experiments. The environment, presented by Hjelsvold et al. [HLMS95, HMS95b] (see Appendices F and E), consists of:

- A prototype implementation of the *VideoSTAR database framework* where media and meta-data have been stored in a dedicated, file-based repositories. The prototype implements the video player operations and the

VideoSTAR querying and browsing algebra. Only parts of the video management API is currently implemented. The functionality for creating and managing video compositions is not implemented – a one-to-one relationship between video documents and stored media segments represents video composition.

- A *digital video player* running on SUN Sparc workstations with a Parallax Xvideo [Par91] card for JPEG video compression and decompression. The video player implements the video player API.
- A *registration tool* for creating structure and content annotations and for relating these to the video data. The current version of the registration tool supports contexts – i.e., content annotations can be defined as *basic* or *primary* context annotations.
- A *document browsing tool* for document structure and contents browsing: The document's structure is displayed to the user as a table of contents with the active item highlighted. The user may select one of the items in the table of contents and thereby instructing the video player to jump to this item. As the video is playing, the annotations valid for the current part of the video is displayed along with a context type tag.
- A *video querying tool* for keyword based querying of basic and primary context contents. Three types of indexes are supported: Persons, keywords, and locations. The querying tool offers lists of persons, keywords, and locations that are registered in the database. Users create a query by selecting items from this list and by using the query algebra to combine different items. The current version of the querying tool provides direct access to the VideoSTAR query algebra – i.e., the user selects individual algebra operations and identifies their input set(s).
- A *tool manager* that keeps track of which tools are active and that broadcasts messages between the individual tools so that all active tools are synchronously updated.

Video applications can be divided into several classes (see Section 2.4). The results presented in this chapter is based on response from end-users having a special interest in *shot-stock* and *subject research* applications (NRK and TV2) and *documentation* applications (Norwegian Folk Museum):

- We have discussed video database issues with librarians at the Norwegian Broadcasting Corporation (NRK) in several occasions [DHS94, Hje94a, Hol94, Mer93]; requirements have been discussed, sample media and meta-data have been studied, and several versions of the prototype have been demonstrated.

- We have used the VideoSTAR prototype tools on television news from six days from the Norwegian TV2. The contents and structure of these news documents have been annotated according to print-outs from TV2's current database.
- Conservators at the Norwegian Folk Museum consider digital video tools and video databases as means for cultural and historical documentation. A work station running the VideoSTAR prototype has been installed at the museum and is currently being used in different projects – i.e., one project for indexing old film and video recordings, one project for documenting traditional handcrafts, and one project for documenting children at play [Aam95, GP95].

The experiences that we have made will be influenced by these users' preferences and cannot unconditionally be transferred to other types of applications – e.g., *Video-On-Demand* or *interactive video* applications.

8.2 END-USER FEEDBACK

The following subsections summarise the feedback given by the users during the experiments. Based on this feedback, we will also discuss the questions raised in Chapters 6 and 7.

8.2.1 Tools and Environment

The primary goal of the experiments has been to gain end-user feedback to the VideoSTAR framework and its way to organise and manage video information in shared environments. Users cannot play with a framework such as VideoSTAR directly; user experiments have to be connected to specific systems implemented on top of the framework. Therefore, one has to be careful to distinguish feedback related to the specific system from feedback valid for the framework itself. In this subsection, we will focus on feedback specifically related to the integrated video archive prototype.

Video Player

Not surprisingly, end-users appreciate direct and remote access to video data. Directors at NRK, for instance, estimate that half of the time used in video editing is spent on winding and rewinding video tapes to find the pieces of interest [DHS94]. Even more time is needed to fetch video tapes from the video tape archive and bring them to the editing station. Direct and remote access to the video archive will make the information stored in the video archive more available, and the directors foresee a significant growth in the use of archival material.

The video player interface and control functions are considered as appropriate in most cases *except* in professional video editing environments where efficient video player operation is requested. Mouses and keyboards are not satisfactory input devices in these cases; video directors in NRK, for instance, demand a specific handwheel input device for *shuttling* – i.e., variable speed forwards backwards playing – and *jogging* – i.e., frame-by-frame stepping.

Registration Tool

The VideoSTAR registration tool encourages users to make more accurate annotations. Today, smaller or larger segments of a video document are annotated as one, complete unit. More accurate annotations can reduce the time needed to find the relevant pieces of video retrieved in a query. Accurate annotations are also a prerequisite for index data sharing. Basic context annotations, for instance, cannot safely be shared among different documents using parts of a video recording, unless the exact temporal extent of these annotations are known.

The price one has to pay for this improvement in index accuracy, is increased indexing time. The amount of extra time needed for indexing depends on the registration practice and it is difficult to produce general figures. Much of the extra time is spent on detecting shot boundaries. Thus, the extra time needed can be significantly reduced if shot boundaries are known in advance.

The registration tool assumes that users follow one specific registration procedure. Users stress that the tool should be adapted to the actual registration practice being used by specific organisations. Users also point out that the registration tool should be capable of importing meta-data being generated by other tools – e.g., production planning or support tools.

The registration tool assumes that complex concepts are broken down in independent annotations. This results in cumbersome registration procedures because users have to create several annotations to index one single concept. A more suitable real-world model should allow one single annotation to represent a complex concept – e.g., by allowing several keywords to be related to the same annotation.

Querying Tool

Users appreciate that the querying tool – combined with the video player – provides direct access to the retrieved objects and, thus, reduces the time needed to search and retrieve pieces of video from the archive. They also appreciate the opportunity to differentiate between sensory contents and topic contents by selecting the context to be queried.

Users are, however, not satisfied with the querying tool's user interface. First, a raw query algebra interface is too complex for ordinary users. Second, the way search items are selected from sequential lists is not appropriate when the database becomes larger and the lists get longer.

Browsing Tool

The structural part of the browsing tool is considered as a useful addition to the querying tool when users are not able to formulate a specific queries. The document structure summarises the contents of the document and the user can directly access parts of the document that seem to be of interest. This way, the user can evaluate the contents of the video document quicker than playing sequentially through it.

Annotations serve two purposes: First, they serve as indexes that makes content based queries feasible. Second, they serve as means for relating supplementary information to pieces of video – information that cannot easily be extracted from the video data themselves. The contents browser displays this information to the user and, thus, provides a description of the video information context. Users at Norwegian Folk Museum consider this to be an essential feature when using video as a tool for documentation.

Users have requested one feature that has not been implemented in the browsing tool: When the number of annotations become large, it becomes difficult to comprehend the meaning of the annotations unless they are organised in one way or another. The user may, for instance, be interested in only one type of annotations – e.g., person annotations. The possibility to attach priorities to – or put restrictions on – the annotations to be displayed will allow the user to control the amount of information and the order in which the annotations are presented.

Integration of Tools

The prototype tools are of great value to the users individually. In addition, the integration of tools has positive effects. Most important is the integration of the individual tools and the video player. Neither of the video archive tools would have been very useful if the tools did not have complete control of the video player.

The combination of the querying tool and the browsing tool has proven to be very useful. The reason for this is that while the querying tool retrieves small pieces of video and presents them to the user out of their context, the browsing tool presents the structural context into which the retrieved items belong.

Video Data Sharing

The experimental environment did not include a video editing tool. The main reason for this has been lack of interest from end-users to use experimental video editing tools in realistic experiments. The consequence of this is that the experimental database does not contain shared video or meta-data; there is a one-to-one relationship between video documents and shared media segments.

8.2.2 Data Model

Together with the users, we have developed two domain-specific models (television news model and cultural/historical video documentation model). The following discussion is based on the experiences gained in these experiments.

Generality

The question put forward is: *Does the data model really capture the important characteristics of video information?*

The data model has been designed to capture all types of meta-data discussed in Section 2.2.2 except version data. During the experiments we have found that the domain-specific part of the model which are being used needs to be better tailored to the application domain, but no lack of functionality has been found in the generic part of the model.

An annotation in the VideoSTAR generic model is related to a temporal interval from a video stream. The spatial dimension of annotations, however, is not addressed. Thus, if a user requires to be able to annotate spatial regions within a stream interval, the application has to provide necessary functionality to define such regions and to define the regions' motion throughout the stream interval.

Extensibility

The question put forward is: *How easy is it to tailor the model to specific application domains: How easy is it to implement only parts of the model and how easy is it to attach domain-specific models to the generic model?*

VideoSTAR handles compositional data, indexing data, and structural data as independent and orthogonal concepts. An application domain, thus, has the freedom to choose which of these concepts to include in the application. (Though, some compositional data will always be present – at least as a one-to-one mapping between video documents and the corresponding stored media segment representing its video stream.)

Our experience so far is that it is rather straightforward to create domain-specific types of annotations. What the applications need to take care of, however, is the fact that VideoSTAR will not return domain-specific annotations themselves. VideoSTAR will only return the identities of these annotations and the application must implement methods to be able to retrieve the annotations of correct type based on these identities.

Meta-Data Sharing

The question put forward is: *How useful are the mechanisms for meta-data sharing?*

Due to the lack of editing tools, we do not have enough experiences to answer the question completely. Our experiments show, however, that the distinction between primary and basic context annotations is a good basis for sharing index data in a consistent way because indexes that are valid independent of specific document contexts are stored only once as part of the basic context. Primary context indexes, on the other hand, are stored in relation to the specific document and are not messed up with other documents' primary context indexes.

Two-level Composition

The question put forward is: *Is the two-level composition powerful enough for representing video document sharing?*

The two-level approach does not take care of the document composition history. For instance, if a television director finds a scene from a television news document which he/she wants to paste in a new document, the two-level approach does not store information telling that the second document reused parts of the first document. If this information is essential in a given application domain, this must be covered by application specific extensions to the generic model.

Efficiency

The question put forward is: *What are the consequences regarding efficiency on user and database operations?*

The disadvantage with the stratification approach is that it breaks complex notions down to individual strata. For instance, one of the evening news from TV2 shows *two doctors carrying a dead body in front of an ambulance.* If *rescuers*, *wounded people*, and *ambulances* where strata of interest, three annotations have to be created to describe the contents of this small piece of video.

The nested stratification approach used in Algebraic Video System [WDG95] and the interval inclusion approach used in OVID [OT93] define a hierarchy of video nodes and define how indexing data related to one node in this hierarchy also index nodes at higher/lower levels in the hierarchy. This means that the complete description of one node can be retrieved by traversing the successor and ancestor nodes in the hierarchy. In VideoSTAR, the only way to retrieve the full description of a piece of video is to search the database for all annotations that have a temporal intersection with the given stream interval.

8.2.3 Querying and Browsing

The querying and browsing functionality presented in Chapter 7 is aimed at offering means for the user to control the degree of meta-data sharing during querying and browsing, and to utilise temporal relationships in query formulation. The key question is how well these goals are achieved.

Context Handling

The question put forward is: *How useful is context handling in practice?*

The experimental environment allows us to evaluate the separation of basic context data from primary context data. This has been proven especially valuable in television news archive where reporters in programme research search for pieces of video discussing a certain topic – i.e., primary context queries – while directors are searching for video with a specific image contents – i.e., basic context queries.

Due to the lack of an editing tool, we have not been able to evaluate how useful the *secondary context* concept is.

Querying Temporal Relationships

The question put forward is: *Are the query facilities rich enough to express the queries that a user wants formulate regarding temporal properties?*

The experimental basis for answering this question is too narrow to formulate very reliable answers. So far, we have not experienced user queries that we have not been able to express in the VideoSTAR algebra. The algebra should be tested on a variety of end-user queries towards a large collection of videos being indexed according to the stratification approach. Such an environment does still not exist.

Algebra Complexity

The question put forward is: *Is the complexity of the algebra really necessary?*

This question can only be answered after a more extensive user-experiment has been accomplished.

	Type of meta-data	**No. per document**	**Duration**
Structure Components	Super Sequences	1	12 min.
	Sequences	14	54 sec.
	Scenes	27	27 sec.
	Shots	112	6.5 sec.
Annotations	Basic context	80	14.5 sec.
	Primary context	45	68 sec.

Table 8.1 Amounts of meta-data in TV2 evening news

8.3 EXPERIMENTAL DATA

The experiment allows us to investigate the amount of meta-data that one can expect to see in a video database. Of course, the amount of meta-data will depend on the given application domain. We have experienced this in our research. For instance, a social science researcher working on a project documenting children at play brought a 22 minute long video which corresponded to one physical shot. This video was be divided into 6 logical parts (scenes). The researcher did not create any basic context annotations, but she created 22 basic annotations.

The evening news videos from TV2 generated most meta-data. Table 8.1 gives the mean values for the number and duration of structure components and annotations. As seen from the table, the shots are very short (about six seconds) and the content is extensively indexed. Thus, it is reasonable to use these data as a typical example of a "meta-data intensive" archive.

We will assume that these figures represent a typical upper bound of expected meta-data in a video database, and we can compute how much meta-data this will correspond to per hour video. We find that one might expect up to 770 structure components of various types and 625 different content annotations per hour video. These figures can be used as a basis for estimating the size needed to store the meta-data in a real video database.

In the experiments, we had no real sharing of video data. The amount of basic context annotations may be reduced when video data are shared because basic context meta-data will not be replicated. New experiments are needed to evaluate the effect of meta-data sharing among different documents.

9 CONCLUSION AND FURTHER WORK

Things I think we are doing wrong

- *Too much continued focus on the same old problems (there continue to be papers on new concurrency control algorithms, hash join algorithms, yet more exotic RAID devices, ...)*
- *Not enough emphasis on new application domains*
- *Too many systems that never become anything more than statistical models or simulators*
- *Too many studies with no connection to how real systems work*
- *Not enough emphasis on building working prototypes*

David J. DeWitt, invited talk, VLDB '95 [DeW95]

The objective of this thesis has been to achieve an in-depth understanding of video information and applications, and to use this knowledge to propose new functionality to be included in future DBMS'es for video information management. A prototype video database system has been implemented to study how well the proposed features suites real end-users. In this very last chapter we will conclude the work by reviewing our research questions presented in Section 1.4, by evaluating our research method, and by outlining directions for further research.

9.1 MAJOR ACHIEVEMENTS

We believe that this thesis has contributed to the state of the art in multimedia database research. In the following subsections we will summarise these contributions in relation to our research questions.

9.1.1 Video in Shared Environments

Our first research question was to study what video information semantics should be supported by future DBMS'es for video information management. We claim that our study has contributed to the understanding of video information management in shared environments:

- We have studied different classes of video applications and have described what kind of meta-data the various classes needs to be able to manage.
- We have proposed a classification of meta-data in multimedia databases that is more specific than state of the art classifications. This classification is useful when studying what support a given video application needs from a video database.
- We have obtained end-user feedback to an integrated, digital video archive environment that should be useful to system developers working on video archive systems.

9.1.2 Data Models for Video Information

Our second research question was to propose a data model that could capture the generic, domain-independent, characteristics of video information which can be extended and tailored to the needs of specific users and application domains. We claim that the proposed data model contributes to the field in that:

- We have shown that the model can be used as the kernel of domain specific models, a kernel which takes care of the domain-independent complexities, allowing the domain modeller to focus on domain-specific complexities.
- The model separates concepts such as composition, document structure, and indexing data and, thus, allows a specific applications to implement or use only parts of the model.

- The data model allows indexing data to be defined in the context of specific documents or to be defined as document-independent.

9.1.3 Video Information Management

Our third research question was to study how video information semantics could be utilised for supporting users in accessing the contents of video information. We claim that the VideoSTAR framework prototype has contributed to the understanding of video information management in that:

- The VideoSTAR API simplifies the development of large groups of video database applications because it provides a high-level interface hiding some of the complexity of the underlying database.
- End-user experiments have given us experimental data that can be useful as a starting point for estimating the size of real video database systems.

9.1.4 Querying and Browsing

Our two last research questions were how to use compositional data and temporal relationships in query formulation and processing. We claim that the querying and browsing mechanisms proposed contribute to the understanding of video information retrieval in that:

- The definition of contexts explicitly distinguishes between meta-data that are document-specific and meta-data that are document-independent.
- The query algebra includes compositional operations that can be used to explicitly control the degree of meta-data sharing in querying and browsing.
- The query algebra includes temporal set operations and temporal filter operations allowing temporal relationships to be exploited in queries.
- The query algebra includes browsing operations which can be used by browsers and/or querying tools for using browsing as an addition to querying.

9.1.5 Future DBMS'es

At the end of this discussion of major achievements and contribution, one might ask: What impact should this research has on future DBMS'es. Our experiments indicate that the VideoSTAR framework has proven useful for video information management, and we will like to claim that future DBMS'es should:

- Include modelling primitives and/or macros for video information modelling which implicitly covers domain-independent semantics.
- Offer a high-level interface of semantically rich operations that can hide some of the underlying complexity of the video information database.
- Extend existing query mechanisms with temporal relationship operations, compositional operations, and browsing operations.

9.2 RESEARCH METHOD EVALUATION

The research presented in this thesis is a result of the activity in the whole VideoSTAR research group. The members of the project group have been working closely together and contributed to the quality of the research. The strengths of our research method can be summarised as:

- We have proposed database functionality *and* implemented prototypes having this kind of functionality.
- End-users have participated in formulation of video database challenges and requirements and in evaluation of the proposals.
- We have addressed the specific issues put forward by the users, but we have developed generic solutions to these issues.

The major objections that can be raised are:

- The individual steps in our research – i.e., requirements analysis, design, implementation, and end-user evaluation – have not been completed as comprehensive as they could have been due to the limited time and resources that have been available.

- Too few experiments on the sharing aspects have been performed, due to the lack of resources to implement a video editor and due to the lack of interest from professional end-users to test experimental video editors.

9.3 DIRECTIONS FOR FURTHER WORK

Digital video and multimedia databases are just about to enter the arena for video information management, and lots of challenging issues remain to be solved. We would like to extend the work started by this thesis in the following directions:

- We would like to have the tools and framework tested more by end-users. Especially, we see a need for testing the query algebra and the context handling more extensively.
- The query algebra interface provided by the querying tool is not satisfactory as an end-user interface. We would like to have a video query language and/or an alternative query interface for the end-users.
- The query algebra raises new issues for efficient query processing and for query optimisation. We would like to investigate what data structures and algorithms are needed for efficient implementation of the video query algebra operations.
- We have not included version data in the work being done. Version data may play an important role in VideoSTAR because the data model is based on the two-level approach and, thus, compositional data do not represent how video documents have been composed of parts from other video documents.
- The end-user experiments have been performed in a single-user mode. We see a need for studying multi-user access to video databases – e.g., to find suitable transaction models.
- In this thesis we have argued that meta-data such as compositional data and structural data should be captured during production. However, large collections of video documents will exist where these data are not known to the database. We would like to study new techniques for temporal browsing in such collections.

REFERENCES

[A$^+$94] Arman et al. Content-Based Browsing of Video Sequences. In *Proceedings of ACM Multimedia '94*, pages 97–103, San Francisco, USA, October 1994.

[A$^+$95] J. Ashley et al. Automatic and Semi-Automatic Image Retrieval Methods in QBIC. In *Proceedings of Storage and Retrieval for Image and Video Databases III - part of IS&T/SPIE's Symposium on Electronic Imaging: Science and Technology*, San Jose, CA, February 1995.

[AA93] P. Atzeni and V. De Antonellis. *Relational Database Theory.* The Benjamin/Cummings Publishing Company, Inc., 1993.

[Aam95] P.A. Aamot. Database for Cultural/Historical Video - Draft Requirement Specification. Technical report, Norwegian Institute of Technology, July 1995. In Norwegian.

[Adi93] C. Adie. A survey of distributed multimedia. Technical Report 5, RARE, January 1993.

[Adi95] M. Adiba. STORM - Structural and Temporal Object-oRiented Multimedia database system. In *Proceedings of the 1995 International Workshop on Multimdia Database Management Systems*, pages 12–19, Blue Mountain Lake, NY, August 1995.

[AHU83] A.V. Aho, J.E. Hopcroft, and J.D. Ullman. *Data Structures and Algorithms.* Addison-Wesley Publishing Company, 1983.

[AHV95] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases.* Addison-Wesley Publishing Company, Inc., 1995.

[All83] J.F. Allen. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, November 1983.

[App93] Apple Computer, Inc. *QuickTime.* Inside Macintosh. Addison-Wesley Publishing Company, 1993.

[Avi93] Avid Technology Inc. *Avid Media Composer User's Guide*, 1993.

[BCD94] G.S. Blair, G. Coulson, and N. Davis. System Support for Multimedia Applications: an Assessment of the State of the Art. *Information and Software Technology*, 36(4), 1994.

[BD92] M.M. Blattner and R.B. Dannenberg. *Multimedia Interface Design.* ACM Press, 1992.

[Ber93] P.B. Berra. Guest Editor's Introduction: Multimedia Information Systems. *IEEE Transaction on Knowledge and Data Engineering*, 5(4):545–550, August 1993.

[BGT92] C. Breiteneder, S. Gibbs, and D. Tsichritzis. Modelling of Audio/Video Data. In *Proceedings of the 11th International Conference on the Entity-Relationship Approach*, pages 322–339, Karlsruhe, Germany, October 7-9 1992.

[BKW94] V. Burrill, Thomas Kirste, and Jochen Weiss. Time-varying Sensitive Regions in Dynamic Multimedia Objects: a Pragmatic Approach to Content-based Retrieval from Video. *Information and Software Technology*, 36(4):213–223, 1994.

[BR93] H. Beech and S. Renner. Gain Momentum - An Analysis of Software Architecture and Concepts. Technical report, Sybase, 1993.

[BR94] K. Böhm and T.C. Rakow. Metadata for Multimedia Documents. *SIGMOD RECORD*, 23(4):21–26, 1994.

[Bue95] D.J. Buerger. Power Pundit. *Wired*, 3(3):128, March 1995.

[Buf94a] J.F.K. Buford. Architectures and Issues for Distributed Multimedia Systems. In J.F.K. Buford, editor, *Multimedia Systems*, chapter 3, pages 45–64. Addison-Wesley Publishing Company, Inc., 1994.

[Buf94b] J.F.K. Buford. Uses of Multimedia Information. In J.F.K. Buford, editor, *Multimedia Systems*, chapter 1, pages 1–25. ACM Press and Addison-Wesley Publishing Company, 1994.

[CC93] J. Clifford and A. Crocker. The Historical Relational Data Model (HRDM) Revisited. In A.U. Tansel et al., editors, *Temporal Databases: Theory, Design, and Implementation*, chapter 1. The Benjamin/Cummings Publishing Company, Inc., 1993.

[CH94] G. Cruz and R. Hill. Capturing and Playing Multimedia Events with STREAMS. In *Proceedings of ACM Multimedia 94*, pages 193–200, San Francisco, USA, October 1994.

[Cha94a] K. Chaix. *Cosmo Compress for IRIS Indigo*, 1994.

[Cha94b] A.S. Chakravarthy. Toward Semantic Retrieval of Pictures and Video. In *RIAO 94 Conference Proceedings*, pages 676–686, New York, NY, October 1994.

[CVB95] E. Chalom and Jr. V.M. Bove. Segmentation of Frames in a Video Sequence using Motion and other Attributes. In *Proceedings of Digital Video Compression: Algorithms and Technologies 1995*, volume 2419, pages 230–237, San Jose, CA, February 1995. SPIE.

[Dah95] H.F. Dahl. The Meaning of Memory. Keynote speech JTS95: TEchnology and our Audio-Visual Heritage, January 1995.

[Dat86] C.J. Date. *Database Systems - Volume 1.* Addison-Wesley Publishing Company, 4th edition, 1986.

[Dav93] M. Davis. Media Streams: An Iconic Language for Video Annotation. In *Proceedings of 1993 IEEE Symposium on Visual Languages*, Bergen, Norway, 1993.

[Dav94] M. Davis. Knowledge Representation for Video. In *Proceedings of the Twelftht National Congress on Artifical Intelligence (AAAI '94)*, pages 120–127, Seattle, WA, 1994.

[dCV95] C. de Carteret and R. Vidgen. *Data Modelling for Information Systems*. Pitman Publishing, 1995.

[DeW95] D.J. DeWitt. Database systems: Road kill on the information superhighway? Invited talk, *21st International Conference on Very Large Data Bases*, Zürich, Switzerland, September 1995. URL: `http://www-db.stanford.edu/dbqual/95.html`.

[DHS94] T. Dybå, T. Holte, and A. Sæter. LAVA Report: Analysis of Television Production. Technical report, SINTEF DELAB, December 1994. In Norwegian.

[Don91] J.W. Donovan. Intel/IBM's Audio-Video Kernel. *BYTE*, December 1991.

[DSP91] G. Davenport, T.A. Smith, and N. Pincever. Cinematic Primitives for Multimedia. *IEEE Computer Graphics & Applications*, July 1991.

[Dud95] A. Duda. Structured Temporal Composition of Multimedia Data. In *Proceedings of the 1995 International Workshop on Multimdia Database Management Systems*, pages 136–142, Blue Mountain Lake, NY, August 1995.

[DVV94] D. Deloddere, W. Verbiest, and H. Verhille. Interactive Video On Demand. *IEEE Communications Magazine*, 32(5):82–88, May 1994.

[Ell90] J.C. Ellis. *A History of Film.* Prentice Hall, 3rd edition, 1990.

[Ell92] E. Elliott. Multiple Views of Digital Video. MIT Media Laboratory Interactive Cinema Group, March 1992.

[EN94] R. Elmasri and S.B. Navathe. *Fundamentals of Database Systems.* The Benjamin/Cummings Publishing Company, 2nd edition, 1994.

[FD95] C.S. Freedman and D.J. DeWitt. The SPIFFI Scalable Video-on-Demand System. In *Proceedings of the 1995 ACM SIGMOD*, pages 352–363, San Jose, CA, May 1995.

[Fos92] B. Foss. *Narrative Technique and Dramaturgy in Film and Television.* SVT Training, 1992.

[Fox91] E.A. Fox. Advances in Digital Multimedia Systems. *IEEE Computer*, October 1991.

[Fur94] B. Furht. Multimedia Systems: An Overview. *IEEE Multimedia*, Spring 1994.

[G$^+$95] D.J. Gemmell et al. Multimedia Storage Servers: A Tutorial. *IEEE Computer*, pages 40–49, May 1995.

[Gal91] D. Le Gall. MPEG: A Video Compression Standard for Multimedia Applications. *Communications of the ACM*, 32(4), 1991.

[GH94] D.J. Gemmell and J. Han. Delay-Sensitive Multimedia on Disks. *IEEE Multimedia*, pages 56–67, Fall 1994.

[Gib94] G.D. Gibson. Report on an International Survey of 500 Audio, Motion Picture Film and Video Archives. Talk given in the annual FIAT/IASA Conference, Bogensee, Germany, September 1994.

[GMP91] F. Garzotto, L. Mainetti, and P. Paolini. Modelling Multimedia Data Bases: The Importance of Exploration v.s. Queries. Technical Report TECH.REP. 03-91, Politecnico di Milano, Italy, 1991.

[GP95] S. Gjertsen and H. Pedersen. Video in Cultural/Historical Documentation. To be submitted as a master thesis (in Norwegian) at the Norwegian Institute of Technology, December 1995.

[Gro94] W.I. Grosky. Multimedia Information Systems. *IEEE Multimedia*, Spring 1994.

[Gro95] Shore Project Group. *An Overview of Shore.* University of Wisconsin - Madison, May 1995.

[GW92] R.C. Gonzales and R.C. Woods. *Digital Image Processing.* Addison-Wesley Publishing Company, 1992.

[HBP94] B.K. Hillyer, A. Biliris, and E. Panagos. The Calico Project for Continuous Media Services. In *Proceedings of the ACM Multimedia '94 Conference Workshop on Multimedia Database Management Systems*, pages 63–70, San Francisco, CA, October 1994.

[Hje94a] R. Hjelsvold. Digital Television Archives - Combining Computer Technology and Video. In *FIAT/IASA Internationaler Kongress 1994*, Bogensee, Germany, September 1994. Battert Verlag, Baden.

[Hje94b] R. Hjelsvold. Sharing and Reuse of Video Information. In *Proceedings of the ACM Multimedia'94 Conference Workshop on Multimedia Database Management Systems*, San Francisco, California, October 1994.

[Hje94c] R. Hjelsvold. Video Information Contents and Architecture. In *Proceedings of the 4th International Conference on Extending Database Technology*, pages 259–272, Cambridge, UK, March 1994.

[HJW94] A. Hampapur, R. Jain, and T. Weymouth. Digital Video Segmentation. In *Proceedings of ACM Multimedia '94*, pages 357–364, San Francisco, USA, October 1994.

[HJW95] A. Hampapur, R. Jain, and T. Weymouth. Indexing in Video Databases. In *Proceedings of the IS&T/SPIE Symposium on Electronic Imaging Science and Technology, Conference on Storage and Retrieval for Image and Video Databases III*, pages 292–306, San Jose, CA, February 1995.

[HLMS95] R. Hjelsvold, S. Langørgen, R. Midtstraum, and O. Sandstå. Integrated Video Archive Tools. In *Proceedings of the ACM Multimedia'95*, San Francisco, California, November 1995.

[HM94] R. Hjelsvold and R. Midtstraum. Modelling and Querying Video Data. In *Proceedings of the 20th VLDB Conference*, pages 686–694, Santiago, Chile, September 1994.

[HM95] R. Hjelsvold and R. Midtstraum. Databases for Video Information Sharing. In *Proceedings of the IS&T/SPIE Symposium on Electronic Imaging Science and Technology, Conference on Storage and Retrieval* *for Image and Video Databases III*, pages 268–279, San Jose, CA, February 1995.

[HMJ93] W. Hodge, S. Mabon, and J.T. Powers Jr. Video On Demand: Architecture, Systems, and Applications. *SMPTE Journal*, September 1993.

[HMS95a] R. Hjelsvold, R. Midtstraum, and O. Sandstå. A Temporal Foundation of Video Databases. In J. Clifford and A. Tuzhilin, editors, *Recent Advances in Temporal Databases. Proceedings of the International Workshop on Temporal Databases*, Zürich, Switzerland, September 1995. Springer Verlag.

[HMS95b] R. Hjelsvold, R. Midtstraum, and O. Sandstå. Searching and Browsing a Shared Video Database. To be included in: K. Nwosu (ed.), *Design and Implementation of Multimedia Database Management Systems, I.*, Kluwer Academic Publishers, 1995.

[Hol94] T. Holte. Digital Television Archives. Master's thesis, Norwegian Institute of Technology, 1994. In Norwegian.

[HR94] R. Hamakawa and J. Rekimoto. Object Composition and Playback Models for Handling Multimedia Data. *Multimedia Systems*, 2(1):26–35, 1994.

[HR95] S. Hibino and E.A. Rundensteiner. A Visual Query Language for Identifying Temporal Trends in Video Data. In *Proceedings of 1995 International Workshop on Multimedia Database Management Systems*, pages 74–81, Blue Mountain Lake, NY, August 1995.

[HS93] M.E. Hodges and R.M. Sasnett. *Multimedia Computing. Case Studies from MIT Project Athena.* Addison-Wesley Publishing Company, Inc., 1993.

[HS95] E. Hwang and V.S. Subrahmanian. Querying Video Libraries. To be published in *Journal of Visual Communication and Image Representation*, 1995.

[HvRB93] L. Hardman, G. van Rossum, and D.C.A. Bulterman. Structured Multimedia Authoring. In *Proceedings of ACM Multimedia 93*, pages 283–290, Anaheim, CA, August 1993.

[J^+95] R. Jain et al. Similarity Measures for Image Databases. In *Proceedings of Storage and Retrieval for Image and Video Databases III*, volume 2420, pages 58–65, San Jose, CA, February 1995. SPIE.

[JH94] R. Jain and A. Hampapur. Metadata in Video Databases. *SIGMOD RECORD*, 23(4):27–33, 1994.

[KE93] R. Keller and W. Effelsberg. MCAM: An Application Protocol for Movie Control, Access, and Management. In *Proceedings of ACM Multimedia 93*, pages 21–29, Anaheim, CA, August 1993.

[KHT91] W. Kameyama, T. Hanamura, and H. Tominaga. A Proposal of Multimedia Document Architecture and Video Document Architecture. In *Proceedings of ICC '91 - The International Conference on Communication Conference Record*, Denver, USA, 1991.

[KL95] G. Koch and K. Loney. *ORACLE: The Complete Reference*. McGraw-Hill, Inc., 3rd edition, 1995.

[Kol94] K. Kolstad. Protecting and Preserving Archival Film in a Broadcasters Storage Vault. In *FIAT/IASA Internationaler Kongress 1994*, pages 171–172, Bogensee, Germany, September 1994. Battert Verlag, Baden.

[L$^+$93] T.D.C. Little et al. A Digital On-Demand Video Service Supporting Content-Based Queries. In *Proceedings of ACM Multimedia 93*, pages 427–436, Anaheim, USA, August 1993.

[LCB90] C. Locatis, J. Charuhas, and R. Banvard. Hypervideo. *Educational Technology, Research and Development*, 38(2), 1990.

[LG90] T.D.C. Little and A. Ghafoor. Synchronization and Storage Model for Multimedia Objects. *IEEE Journal on Selected Areas in Communications*, pages 413–427, April 1990.

[LG93] T.D.C. Little and A. Ghafoor. Interval-Based Conceptual Models for Time-Dependent Multimedia Data. *IEEE Transactions on Knowledge and Data Engineering*, 5(4), 1993.

[LG95] T. Little and R. Gusella, editors. *Proceedings of the 5th International Workshop on Network and Operating System Support for Digital Audio and Video*, Durhan, NH, April 1995.

[LH91] M. Liebhold and E.M. Hoffert. Toward an open Environment for Digital Video. *Communications of the ACM*, 34(4), 1991.

[Lio91] M. Liou. Overview of the px64 kbit/s Video Coding Standard. *Communications of the ACM*, 32(4), 1991.

[Lip91] A. Lippman. Feature Sets for Interactive Images. *Communications of the ACM*, 34(4), April 1991.

[Lit94] T.D.C. Little. Time-Based Media Representation and Delivery. In J.F.K. Buford, editor, *Multimedia Systems*, chapter 7, pages 175–200. ACM Press and Addison-Wesley Publishing Company, 1994.

[LM93] T.Y.C. Leung and R.R. Muntz. Stream Processing: Temporal Query Processing and Optimization. In A.U. Tansel et al., editors, *Temporal Databases: Theory, Design, and Implementation*, chapter 14. The Benjamin/Cummings Publishing Company, Inc., 1993.

[LOP94] A. Laursen, J. Olkin, and M. Porter. Oracle Media Server: Providing Consumer Based Interactive Access to Multimedia Data. In *Proceedings of the 1994 ACM SIGMOD*, pages 470–477, Minneapolis, MN, May 1994.

[LV94] T.D.C. Little and D. Venkatesh. Prospects for Interactive Video-on-Demand. *IEEE Multimedia*, 1(3):14–24, Fall 1994.

[Mar94] A.O. Martinussen. Documenting Handcrafts on S-VHS. In *From Imagination to Reality.* Norske Kunst- og Kulturhistoriske Museer, 1994. In Norwegian.

[MBE95] T. Meyer-Boudnik and W. Effelsberg. MHEG Explained. *IEEE Multimedia*, pages 26–38, Spring 1995.

[MBG93] G. Miller, G. Baber, and M. Gilliland. News On-Demand for Multimedia Networks. In *Proceedings of ACM Multimedia 93*, pages 383–392, Anaheim, CA, August 1993.

[MD89] W.E. Mackay and G. Davenport. Virtual Video Editing In Interactive Multimedia Applications. *Communications of the ACM*, 32(7):802–810, 1989.

[Mer93] P. Merok. Data Models for Digital Film and Video Archives. Master's thesis, Norwegian Institute of Technology, 1993. In Norwegian.

[Mil90] G. Millerson. *Effective TV Production.* Focal Press, 1990.

[Mon81] J. Monaco. *How to Read a Film. The Art, Technology, Language, History and Theory of Film and Media.* Oxford University Press, 1981.

[MS93] J. Melton and R. Simon. *Understanding the New SQL: A Complete Guide.* Morgan Kaufmann Publishers, 1993.

[MS95] D.E. McDysan and D.L. Spohn. *ATM - Theory and Application.* McGraw-Hill, 1995.

[Neg94] N. Negroponte. Bit by Bit on Wall Street: Lucky Strikes Again. *Wired*, 2(5):144, May 1994.

[NKN91] S.R. Newcomb, N.A. Kipp, and V.T. Newcomb. The HyTime Hypermedia/Time-based Document Structuring Language. *Communications of the ACM*, 34(11):67–83, November 1991.

[NRK94] NRK 1993-1995. Report made by Department for Public Relations, June 1994. In Norwegian.

[OT93] E. Oomoto and K. Tanaka. OVID: Design and Implementation of a Video-Object Database System. *IEEE Transactions on Knowledge and Data Engineering*, 5(4):629–643, 1993.

[Par91] Parallax Graphics, Inc. *XVideo User's Guide*, 1991.

[PR94] B. Prabhakaran and S.V. Raghavan. Synchronization Models for Multimedia Presentation with User participation. *Multimedia Systems*, 2(2):53–62, 1994.

[Pri93] R. Price. MHEG: An Introduction to the future International Standard for Hypermedia Object Interchange. In *Proceedings of ACM Multimedia 93*, pages 121–128, Anaheim, CA, August 1993.

[Ran92] P. Venkat Rangan, editor. *Network and Operating System Support for Digital Audio and Video. Third International Workshop*, La Jolla, California, USA, November 1992.

[RBE94] L.A. Rowe, J.S. Boreczky, and C.A. Eads. Indexes for User Access to Large Video Databases. In *Proceedings of the IS&T/SPIE Symposium on Electronic Imaging Science and Technology, Conference on Storage and Retrieval for Image and Video Databases II*, San Jose, CA, February 1994.

[RC95] R. Rooholamini and V. Cherkassky. ATM-Based Multimedia Servers. *IEEE Multimedia*, pages 39–52, Spring 1995.

[RW94] A.L.N. Reddy and J.C. Wyllie. I/O Issues in a Multimedia System. *IEEE Computer*, 27(3):69–74, 1994.

[S$^+$93] D. Shepherd et al., editors. *Proceedings of the 4th International Workshop on Network and Operating System Support for Digital Audio and Video - 4th International Workshop*, Lancaster, U.K., November 1993.

[Sal88] G. Salton. *Automatic Text Processing - The Transformation Analysis, and Retrieval of Information by Computer.* Addison-Wesley Publishing Company, 1988.

[Sch93] P. Schnorf. Integrating Video into an Application Framework. In *Proceedings of ACM Multimedia 93*, pages 411–417, Anaheim, USA, August 1993.

[Sch94] D. Schüller. Beyond Petabytes: Strategies to Safeguarding the Audio and Video Heritage in the Digital Domain Mass Storage Systems. Talk given in the annual FIAT/IASA Conference, Bogensee, Germany, September 1994.

[SG94a] Inc. Silicon Graphics. *CoBRa - A Constant Bit-Rate Server. Overview, Architecture, and Interface*, April 1994.

[SG94b] Inc. Silicon Graphics. Silicon Graphics' Interactive Technologies Unveiled with Time Warner Cable's full Service Network. Press release, December 1994.

[Sim95] B. Simonnot. A Cooperation Model for Video Document Retrieval. In *Proceedings of Storage and Retrieval for Image and Video Databases III*, volume 2420, pages 307–317, San Jose, CA, February 1995. SPIE.

[Smi92] T.G.A. Smith. If You Could See What I Mean... Descriptions of Video in an Anthropologist's Notebook. Master's thesis, MIT, 1992.

[Son94] Sony Corporation. Digital BETACAM DVW-510P Component Digital Videocassette Player, 1994. A ten pages' product information sheet.

[SP91] T.G.A. Smith and N.C. Pincever. Parsing Movies in Context. In *Proceedings of the 1991 Summer USENIX Conference*, pages 157–167, Nashville, USA, 1991.

[SS95] B. Simonnot and M. Smaïl. Model for Interactive Retrieval of Videos and Still Images. In *Proceedings of 1995 International Workshop on Multimedia Database Management Systems*, pages 128–135, Blue Mountain Lake, NY, August 1995.

[SSJ93] D. Swanberg, C.-F. Shue, and R. Jain. Knowledge Guided Parsing in Video Databases. In *Proceedings of Conference on Storage and Retrieval for Image and Video Databases II*, San Jose, CA, 1993. SPIE.

[Ste95] R. Steinmetz. Analyzing the Multimedia Operating System. *IEEE Multimedia*, pages 68–84, Spring 1995.

[SvdM91] F. Sijstermans and J. van der Meer. CD-I Full-Motion Video Encoding on a Parallel Computer. *Communications of the ACM*, 34(4):81–91, April 1991.

[SW94] G.A. Schloss and M.J. Wynblatt. Building Temporal Structures in a Layered Multimedia Data Model. In *Proceedings of ACM Multimedia 94*, pages 271–278, San Francisco, CA, October 1994.

[SZ94] S.W. Smoliar and H. Zhang. Content-Based Video Indexing and Retrieval. *IEEE Multimedia*, 1(2):62–72, Summer 1994.

[T$^+$93a] A.U. Tansel et al. *Temporal Databases - Theory, Design and Implementation.* The Benjamin/Cummings Publishing Company, Inc., 1993.

[T$^+$93b] F.A. Tobagi et al. Streaming RAID – A Disk Array Management System for Video Files. In *Proceedings of ACM Multimedia 93*, pages 393–400, Anaheim, CA, August 1993.

[T$^+$94a] B. Tierney et al. Distributed Parallel Data Storage Systems: A Scalable Approach to High Speed Image Servers. In *Proceedings of ACM Multimedia 94*, pages 399–405, San Francisco, CA, October 1994.

[T$^+$94b] Y. Tonomura et al. Structured Video Computing. *IEEE Multimedia*, pages 34–43, Fall 1994.

[Teo93] L. Teodosio. Salient Video Stills: Content and Context Preserved. In *Proceedings of ACM Multimedia 93*, pages 39–46, Anaheim, CA, August 1993.

[Tok94] H. Tokuda. Operating System Support for Continuous Media Applications. In J.F.K. Buford, editor, *Multimedia Systems*, chapter 8, pages 201–220. ACM Press, 1994.

[Tur90] J. Turner. Representing and Accessing Information in the Shotstock Database at the National Film Board of Canada. *The Canadian Journal of Information Science*, 15(4), December 1990.

[vR79] C.J. van Rijsbergen. *Information Retrieval.* Butterworths, 2 edition, 1979.

[Wal91] G.K. Wallace. The JPEG Still Picture Compression Standard. *Communications of the ACM*, 34(4):30–44, 1991.

[WDG95] R. Weiss, A. Duda, and D.K. Gifford. Composition and Search with a Video Algebra. *IEEE Multimedia*, 2(1):12–25, Spring 1995.

[Whi94a] C. White. Heavy Metal Video: EDLs, Edit Suites and Nonlinear Editors. *Digital Video*, pages 76–79, December 1994.

[Whi94b] C. Whittaker. People's Century - Through the Archives. In *FIAT/IASA Internationaler Kongress 1994*, pages 123–129, Bogensee, Germany, September 1994. Battert Verlag, Baden.

[Y+95] M.M. Yeung et al. Video Browsing using Clustering and Scene Transitions on Compressed Sequences. In *Proceedings of Multimedia Computing and Networking 1995*, volume 2417, pages 399–413, San Jose, CA, February 1995. SPIE.

[ZSW95] H. Zhang, S.W. Smoliar, and J.H. Wu. Content-Based Video Browsing Tools. In *Proceedings of Multimedia Computing and Networking 1995*, volume 2417, pages 389–398, San Jose, CA, February 1995. SPIE.

A

VIDEO INFORMATION CONTENTS AND ARCHITECTURE

This appendix contains the paper presented at the 4th International Conference on Extending Database Technology (EDBT '94) held in Cambridge, UK, March 1994.

Appendix A

B

MODELLING AND QUERYING VIDEO DATA

This appendix contains the paper presented at the 20th Conference on Very Large Databases (VLDB '94) held in Santiago, Chile, September 1994.

Appendix B

C

DATABASES FOR VIDEO INFORMATION SHARING

This appendix contains the paper presented at the IS&T/SPIE Symposium on Electronic Imaging Science and Technology (EI 95), Conference on Storage and Retrieval for Image and Video Databases III held in San Jose, California, February 1994.

Appendix C

D
A TEMPORAL FOUNDATION OF VIDEO DATABASES

This appendix contains the paper presented at the International Workshop on Temporal Databases held in Zürich, Switzerland, September 1995.

Appendix D

E

SEARCHING AND BROWSING A SHARED VIDEO DATABASE

This appendix contains a book chapter to be included in: K. Nwosu (ed.), Design and Implementation of Multimedia Database Management Systems, I, Kluwer Academic Publishers, 1995.

Appendix E

F
INTEGRATED VIDEO ARCHIVE TOOLS

This appendix contains the paper presented at the ACM Multimedia '95 held in San Francisco, California, November 1995.