Database technology and the management of multimedia data in Mirror

A.P. de Vries and H.M. Blanken

CTIT, University of Twente, INF/IS 3055, P.O. Box 217, Enschede, The Netherlands

ABSTRACT

Multimedia digital libraries require an open distributed architecture instead of a monolithic database system. In the Mirror project, we use the Monet extensible database kernel to manage different representations of multimedia objects. To maintain independence between content, meta-data, and the creation of meta-data, we allow distribution of data and operations using CORBA. This open architecture introduces new problems for data access. From an end user's perspective, the problem is how to search the available representations to fulfill an actual information need; the conceptual gap between human perceptual processes and the meta-data is too large. From a system's perspective, several representations of the data may semantically overlap or be irrelevant. We address these problems with an iterative query process and active user participation through relevance feedback. A retrieval model based on inference networks assists the user with query formulation. The integration of this model into the database design has two advantages. First, the user can query both the logical and the content structure of multimedia objects. Second, the use of different data models in the logical and the physical database design provides data independence and allows algebraic query optimization. We illustrate query processing with a music retrieval application.

Keywords: Digital libraries, extensible databases, distributed databases, database architecture, multimedia query processing, multimedia information retrieval, relevance feedback.

1. INTRODUCTION

The progress of multimedia technology results in vast amounts of digitized multimedia data for both personal and professional use. Holiday pictures are displayed on the web instead of in traditional photo albums. Businesses and government agencies create large multimedia digital libraries for applications in medicine, entertainment, education, and remote sensing. In our work, we focus on video archives for entertainment and education.

The management and support of such multimedia collections requires an infrastructure for easily browsing, summarizing and indexing the information. There is a need for general purpose support of the development of applications for content management, similar to the support offered by relational database technology for the development of administrative applications.[1] The digital library environment is however very different from traditional database applications. This implies that the design decisions made for traditional applications may not be valid in the context of multimedia databases. The Mirror project studies the design of database technology in a multimedia digital library setting.

Multimedia objects are usually stored in a database as Binary Large Objects (BLOBs). Reference 2 distinguishes between *active* and *passive* multimedia objects. Users can specify conditions on active objects in their queries, referring to either content or existence. It is not possible to condition on content of passive objects. Most databases view images, audio and video objects as passive objects. Thus, we may look at the tenth video or select employees without a picture, but we cannot retrieve images showing a 'windmill' or music similar to 'The Beatles'.

Handling the digitized data as active objects requires the (transparent) management of *meta-data*. Manually added textual descriptions could be searched using text retrieval. This is however problematic, and not just because of the amount of work involved to acquire these descriptions. Some aspects of multimedia data simply cannot be expressed in textual descriptions.[3]

Content-based retrieval attempts to alleviate this problem. The multimedia object is approximated using automatically derived properties called *features*. The feature vectors usually describe easy-to-calculate *syntactic* properties of the multimedia data, like the color distribution and texture.[4,5] A distance measure in feature space determines

Authors' E-mail (send correspondence to A.P. de Vries): {arjen,blanken}@cs.utwente.nl

which objects are similar. Users do not specify their queries directly using the features, but instead give an example of a good object. This approach is therefore also known as ‘query by example’ (QBE).

In the remainder of this article, we discuss the requirements of multimedia digital libraries and their impact on database design. Section 2 illustrates the need for an open distributed infrastructure. We address this requirement in Sect. 3 with the combination of an extensible database system, an extensible data model and our daemon paradigm, interconnected using CORBA. Section 4 discusses the problem of retrieval in the resulting infrastructure and proposes a new approach to query processing. Its integration into the database management system is the topic of Sect. 5. We then describe some experiments with our multimedia database in the context of music retrieval, and conclude with some remarks on further research.

2. MULTIMEDIA DIGITAL LIBRARIES

A multimedia digital library involves several more or less independent actors. We categorize the actors cooperating in the digital library as follows: end users, content providers, content access providers, and annotators.

The end users are the *consumers* of the digital library. They want to search and browse the digital library to retrieve multimedia objects that satisfy their information need. In Sect. 4 we further discuss this group. The other actors are the *producers* in this environment. They collect and manage the multimedia data; their requirements are the topic of this and the following section.

Content providers are the owners of the multimedia data. They decide in which format the data is delivered to the end users, and they may want to charge money for usage of the footage. Because of copyright issues, they are usually reluctant to have other parties take care of their data.

The content access provider manages the meta-data for searching the collection. A content access provider is not necessarily part of the same business as the content provider. Conversely, the access provider may manage meta-data about multimedia footage of many different owners. For example, a news archive may provide access to video fragments from both public and commercial broadcasters. And, each owner may implement a different policy to serve the data. The end user can decide whether to pay for better quality data based on summary information.

The creation of meta-data about the content is the task of the annotators. We can distinguish between human annotators and meta-data extraction software. Again, the annotation process does not have to take place in the same business as the content provider or content access provider. We expect the rise of small innovative enterprises that develop very specialized extraction software, like for instance ‘face recognition of European politicians’. Such parties will sell their extraction service rather than their software. Even in the case when all annotation is performed by the content provider itself, the people involved in (human) annotation typically work in different departments to the ones responsible for content delivery.

The independency of each of these cooperating parties conflicts with the traditional approach of using a centralized database system. Apart from the organizational matters mentioned above, technical arguments also favour a separation between content and meta-data. Searching and browsing with many simultaneous users has different system requirements to streaming multiple video streams.[6] Multimedia communication technology develops fast and is best handled by the operating system.[7] Similarly, meta-data extraction should be handled separately from its usage in retrieval. The extraction process may involve much computational processing, need access to special knowledge bases, or use specific hardware. A good example of the latter is the extraction of closed caption from the analog video using special decoder hardware.

3. AN OPEN DISTRIBUTED ARCHITECTURE

The cooperation between independent parties and the technical considerations of the previous section make it impossible to use the traditional database approach, in which a centralized monolithic database controls the whole environment (see also Ref. 8). We need an open distributed architecture that is in many ways similar to the web. We should however maintain the core ideas of database technology. In particular, we need the notions of *schema* and *data model*, describing the specification and structure of the data independent from the data itself. Concepts from relational database technology such as algebraic query optimization and concurrency control are crucial elements to enable scaling up to large collections of multimedia data and many simultaneous users. We further need access management, to control who can add data, who can remove data, and who can make changes to the schema. These can only be achieved through some central unit of control.

A very important issue in the context of digital libraries is *extensibility.* Without taking the system down, we must be able to add new data types and operations. New data types can be new kinds of annotations or new formats of digitized data. New operations are needed to do conversions between data formats. Also, new algorithms to extract meta-data from the multimedia objects should be added to the system as soon as they are available.

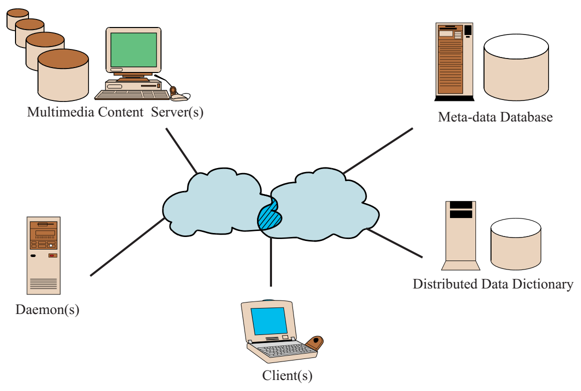

Figure 1. The open distributed architecture used in Mirror.

Figure 1 shows an open distributed architecture, developed especially for multimedia digital libraries. The architecture consists of clients, daemons, content servers, a meta-data database, and the distributed data dictionary*. A ‘software bus’ connects the component systems and provides a single name-space for all components. The meta-data database is also called *database.* We refer to the complete architecture as the *multimedia database.*

The distributed data dictionary manages the specifications of data types and operations on these types. This is also the only interface to access the data, hiding the details of the physical implementation of the data model from the other components. Extensions to and usage of data types or operations must be registered with the distributed data dictionary. It thus has a complete overview of data and operations in the multimedia database, providing an opportunity for query optimization, concurrency control and transaction management. This is also the place to implement caching policies and handle security issues.

Operations include complex meta-data extraction techniques, but also relatively simple operations like moving an object from one location to another or data format conversions. Because such operations may be implemented by annotators, content providers, as well as the access provider, we introduce the *daemon* paradigm. Daemons perform the operations specified in the schema and are only loosely connected to the other components. The daemon paradigm is further explained in Sect. 3.2.

Multimedia content servers are managed by the content providers. A content server handles the delivery of multimedia data to the clients. It is responsible for playing synchronized audio and video streams. The content servers also communicate with daemons that need the multimedia data for their operations. Because the content servers are separate components, they may be implemented on special hardware with an operating system that is especially suited to handle multiple streams of varying quality of service.

The meta-data database stores the locations of multimedia objects in the content servers, the relationships between objects, as well as the meta-data associated to these objects. This database is tightly connected to the distributed data dictionary. When a new data type is registered in the architecture, the distributed data dictionary creates the tables in the database to store instances of the new type and manages the access to these tables.

3.1. The data model

The data model supported in Mirror is based on a semantic network. A semantic network is a graph where nodes represent concepts and arcs represent relationships between concepts.[9] The three modeling entities in Mirror are:[10]

- *Semantic object*: the basic concept in the network. It models media objects, as well as things, persons, places, and actions that occur in the real world.

*We call this the *distributed* data dictionary to distinguish it from the data dictionary of the meta-data database system.

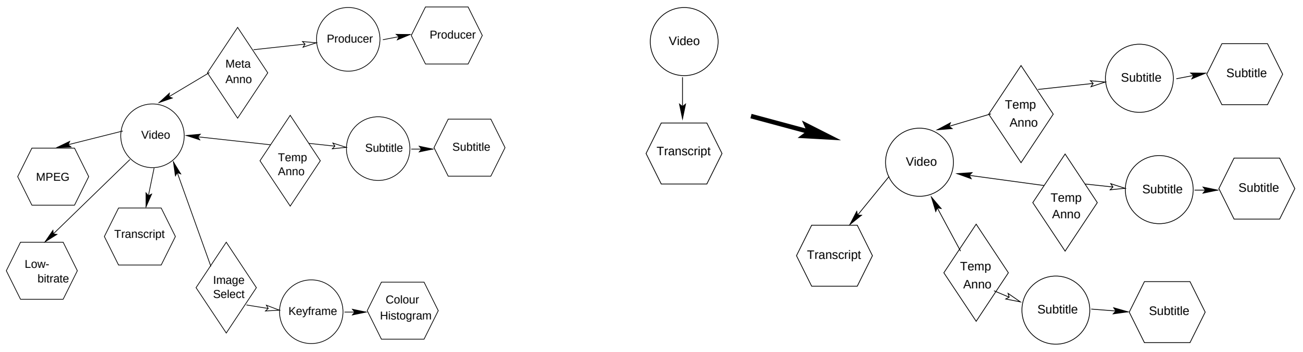

Figure 2. Representation of a video (a) and an operation that transcribes a video (b).

- *Representation object*: the terminal node in the network, that stores a *value* for the semantic object it is connected to. A representation object can only be attached to one semantic object.
- *Semantic relationship*: a semantic relationship links two semantic objects. It is used to model the annotation process.

Figure 2^a demonstrates an instance of a video in this data model. Semantic objects are drawn as circles, representation objects as hexagons and semantic relationships as diamonds. Each node has a data type, shown in the figure as the labels in the nodes. The video has the following representations: a digitized MPEG file, the text transcript from the closed caption decoder, and a low-bitrate version for viewing over a low bandwidth connection. We see annotations for its producer and a subtitle. Finally, we see how a keyframe and its color histogram are attached to the video through an image select relationship.

Different types in Mirror's data model do not necessarily correspond to different base types in the schema of the meta-data database. For example, both subtitle and producer are stored as string values in the database. Also note that the semantic object for subtitle and its text representation are different entities in the data model. In the physical design, we could of course combine the semantic object subtitle and its text representation. A representation of a digitized media objects stores a reference to its location in the content server instead of the real value.

3.2. The daemon paradigm

Daemons are components of the architecture that perform operations. A daemon requires an input object, and generates an output object. Figure 2^b illustrates a daemon transcribing a video object. Conceptually, a daemon is simply a user-defined function in the multimedia database system. Unlike user-defined functions in extensible databases, daemon operations are only loosely connected to the meta-data database. These operations may be performed on a different machine than the database server if we want to separate the creation of meta-data from its management.

A daemon can operate on its own initiative or on request. In the first case, it requests input objects with its `get_work` query. This query consists of three parts: the parent condition p and two child conditions c_1 and c_2. The query returns the collection of object identifiers for objects that satisfy p, are connected (in the data model) to an object fulfilling c_1, but are not connected to an object fulfilling c_2. The daemon in the figure requests video objects (p) that have a transcript (c_1) but do not have subtitle annotations (c_2). The distributed data dictionary registers the work that has been assigned to each daemon. The daemon commits to performing its operation within a certain time. After that time, it may be assigned to a different daemon in subsequent `get_work` calls.

3.3. Prototype implementation

Monet is an extensible parallel database kernel.[11,12] It is intended to serve as a backend in various application domains; e.g. the Acoi algebra is an extension module that supports image retrieval.[13] Monet has also been used succesfully for geographic information systems as well as commercial data mining applications.

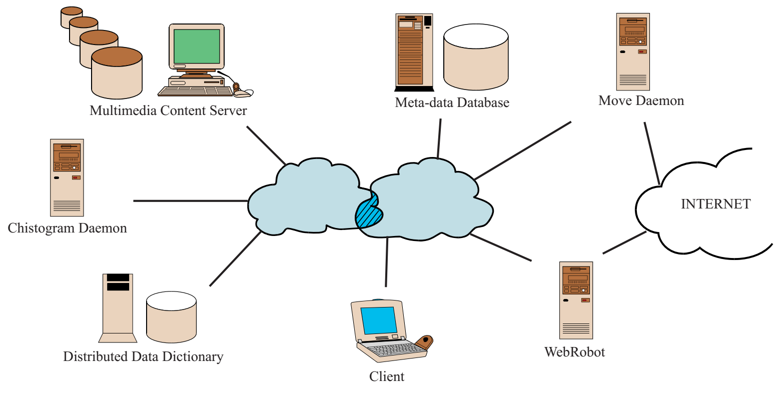

Figure 3. The demonstration system.

The architecture's prototype implementation is based on a combination of Monet and CORBA middleware.[14] The public domain 'ORBacus' object request broker (ORB) connects the component systems. These components implement the interface definitions provided by the Mirror architecture. These interfaces are specified in the CORBA Interface Definition Language (IDL). The distributed data dictionary is implemented in Java. It uses Monet for its persistence, and publishes the schema to the other components through CORBA naming service. The data model is mapped to tables in Monet.

We implemented a simple web-based image retrieval system to demonstrate the use of this architecture (Fig. 3). The `WebRobot` searches the internet for image URLs, and inserts them in the meta-data database. At regular time intervals, the `Move daemon` requests image URLs whose images have not yet been stored on the media server. It then moves a copy of the image to the media server. The `Chistogram daemon` extracts color histograms of these images. Our content server is a simple Java server using a file system to store the data. The client application can use the color histograms for color-based image retrieval. Each of the components of the demonstration system runs on different hosts. The client application may also run in a web browser; the ORB in the browser will communicate transparently with the ORB of the multimedia database.

4. SUPPORTING RETRIEVAL BY CONTENT

The design of the architecture so far supports mainly the producers in the digital library, addressing the requirements discussed in Sect. 2. The end users want to use the multimedia database to satisfy their information needs. In the following section, we describe their problems with retrieval by content in the multimedia database developed in the previous section. We conclude with a potential solution for these problems, based on a query process in which the database system supports the users with query formulation.

4.1. The query formulation problem

Meta-data describing multimedia objects can be related to three different views: the appearence, the structure, and the content. These views are also known as the *layout*, the *logical*, and the *content* structure of multimedia data.[15] For an image, the height of the image is in the layout structure, the name of the photographer is in the logical structure, and the color histogram is in the content structure.

Database query languages are designed to query multimedia objects by their logical and layout structure: e.g. 'find videos directed by Spielberg', or 'select employee homepages with an image in the top-left corner'. Unfortunately, databases do not easily support retrieval of multimedia objects by content.[3] The conceptual gap between the user's perception of multimedia objects and the low-level features modeling the content is too large. Each feature representation is only partly related to the semantic properties in which the user is interested. For instance, the texture feature 'circularity' models an image in a rather syntactic manner which is of little help to an end user. Especially non-verbal aspects of multimedia, such as emotional and aesthetic values, are hard to capture in a query. The multimedia objects are however easily recognized and compared by the end user. Multimedia querying seems to require an *iterative* search process.

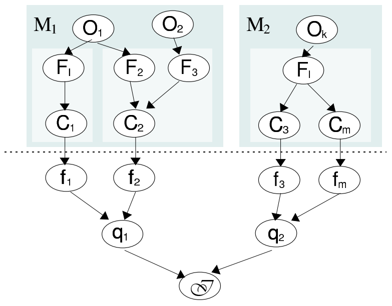

Figure 4. Our version of the inference network retrieval model.

A consequence of the extensibility in the Mirror architecture is the dynamic evolution of the meta-data available for querying the multimedia data. Rather than a 'society of models',[16] our situation resembles anarchy. Using a single feature space is already problematic for query formulation; now the user has to deal with a wide variety of representations. A combination of feature spaces may however help to improve the retrieval process, as demonstrated in e.g. Ref. 16 and 17. No single feature space appears to be 'best' for all queries. But, given the right combination of feature spaces, good retrieval results have been achieved. The Refs mentioned estimate weights for the contribution of different feature spaces with an iterative learning process.

An extra problem to consider is the overlap between feature spaces with respect to the perceptual aspects that are modeled. Evaluation of several color based video segmentation algorithms (a task considerably similar to image retrieval using color histograms) has shown that the effect of the particular color model on the quality of the result is insignificant.[18] Although such dependencies between feature space must be taken into account during the retrieval process, this should not be the responsibility of the end user.

4.2. Retrieval as inference

Summarizing, searching multimedia by its content structure requires a dialogue between user and system. The system should infer a query against the meta-data describing content that (hopefully) reflects the user's information need. The inferred query should use a combination of content representations, also taking into account their interdependencies. This leads to the following query process. The system infers queries for those parts of the user's information need that refer to the content structure of multimedia objects. Content-based querying follows the QBE paradigm: the conditions on content are specified through example objects. The first query simply uses the given examples to infer the queries in the database. In subsequent iterations, the user's relevance judgments are used to adapt the corresponding similarity query to better reflect the user's information need.

Modeling retrieval as an inference process has been studied extensively by the information retrieval (IR) community.[19] In an IR system, the *retrieval model* specifies the (text) document representation, the query representation, and the inference procedure. The succesful information retrieval system InQuery has been designed to use multiple representations of queries and documents.[20,21] Its retrieval model is based on a probabilistic theory of evidential reasoning using Bayesian networks, called the inference network retrieval model. In Ref. 22, we proposed a generalization of this retrieval model for multimedia retrieval with feature based retrieval techniques. The network computes the probability $\mathcal{P}(\mathcal{I} \mid \mathrm{O}_i)$ whether multimedia object O_i satisfies the user's information need $\mathcal{I}$. Probabilistic inference provides the necessary abstraction of specific feature spaces and their distance measures, allowing the combination of information from different content representations into one final judgment. The modularity of the network matches the extensibility of the whole architecture.

In IR, the words in a text document are the basic units of content representation. A word naturally refers to a *concept* in the real world. A feature representation F_k is just a point in multidimensional space and usually unique. It only identifies some particular instance, and not a concept. Therefore, we group several instances together into clusters C_l using unsupervised clustering. We then use these clusters similar to words in InQuery.

The inference network, shown in Fig. 4, consists of two component networks: the object network and the query network. Nodes are binary variables, and the arcs between nodes represent dependencies. The object network is static for a given collection. At the top, we have a node O_i for each multimedia object. The gray boxes M_j correspond to

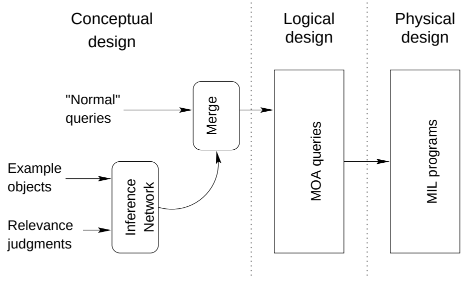

Figure 5. The three levels of query processing.

semantic object types of the data model. We restrict ourselves to atomic objects without further internal structure. Feature spaces are shown as light gray boxes; the feature representations of the objects have nodes F_k. Although the feature spaces in Fig. 4 are assumed independent, dependencies between them can be modeled analogous to thesaurus terms in the original model. The query network may change dynamically in interaction with the user. Its nodes q_n represent example objects and relevance judgments, and the f_m represent the related clusters.

Dependencies between nodes (the arcs) require the specification of conditional probability distributions. The conditional probability $\mathcal{P}(\mathrm{C}_j \mid \mathrm{F}_i)$ is the probability that cluster C_j is represented by feature representation F_i of the multimedia object. For text retrieval, this probability is approximated with a measure comparing the term distribution in this document with the term distribution in the complete collection. In our model, this probability would be estimated using the relative position of that point in the cluster and the distribution of the other feature points in that cluster.

5. THE MIRROR DATABASE ARCHITECTURE AND ITS IMPLEMENTATION

The content-based retrieval process as outlined in 4.2 may of course be implemented as a special application on top of an extensible database system. One drawback of a design in which the inference about content is not integrated in the database architecture is that it is not trivial to combine conditions on the content structure with other conditions, for example on the logical structure of the objects. Ad-hoc querying becomes almost impossible, and application developers must develop their own solutions for the query formulation problems. Also, the inference process generates a series of queries in the meta-data database. Its query optimizer must know the relationship between these requests to determine efficient query execution plans. Performing such processing outside the scope of the database system will have a dramatic negative impact on performance.[10]

The goal of the Mirror project is therefore the design of a database system in which retrieval by content is an integrated part of the database architecture. Our design consists of three levels, which we refer to as the conceptual, the logical and the physical level of the design, see Fig. 5.

5.1. Design

At the conceptual level, the data model described in Sect. 3.1 is adapted to make explicit which representation objects relate to the content structure of multimedia objects. We therefore introduce *content objects*, a specialization of representation objects. On insertion, instances of these types are not just stored in the meta-data database, but also added to the object network. The conceptual level also defines commands to begin and end *dialogues.* A dialogue is defined as all interaction between the user and the system concerning a single information need. At the begin of a dialogue, we create an *interaction object.* This interaction object stores the relevance feedback provided by the user during the dialogue, and is used by the inference component to construct and adapt the query network. After ending the dialogue, the interaction object is not discarded but stored in a special section of the distributed data dictionary. These stored dialogues provide the necessary data to learn dependencies between feature spaces.

At the logical level, we use the *MOA object algebra.*[23] MOA provides an extensible nested object data model with bags, objects, and tuples. We extended MOA with structures for content representation and probabilistic inference using the Bayesian network.[24] The resulting language allows us to express in a single MOA expression both the 'normal' query conditions, referring to logical and layout structure, and the QBE query components, referring to content structure. When one condition specifies the photographer name, an additional algebraic optimization phase could rewrite the merged expression such that the cheap select on photographer name is performed before we compute probabilities in the inference network handling the content.

MOA expressions are automatically translated to query plans expressed as *MIL programs.* MIL is Monet's language to access primitive algebraic database operations (e.g. join, select). The use of a different physical data model for the execution of the MOA expressions provides *data independence.* We do not believe that the implementation of objects at the database level should closely follow the view on the objects at the logical and conceptual levels. The MOA expressions describe queries tuple-at-a-time. The translation produces set-at-a-time operations at the physical level. Such operations are usually performed more efficiently by database systems. For example, they can use Monet's access structures to speed up execution. The translation also provides an abstraction that makes it simpler to move toward a parallel implementation, in which large data sets are fragmented over different databases. The two-step approach with MOA as an object-oriented front-end to Monet has been shown to be succesful for an OO version of the one Gigabyte TPC-D decision support benchmark.[23]

5.2. Implementation

The implementation of the complete architecture is work in progress. We are still working with a prototype that is not yet suited for end users. We manually construct the MOA queries to express an information need. The MOA tool translates these MOA expressions to MIL programs, and presents the results to the user. The queries may contain conditions on both the logical structure and the content structure of the multimedia objects. We have yet to implement the mapping from the conceptual level to the logical level.

The retrieval model of Sect. 4.2 has been implemented as follows. We extended MOA with new structures for probabilistic inference that can be combined with the standard structures tuple and bag. The following code example models a document collection as a collection of tuples with author name and document content. The MOA expression on the right uses probabilistic inference to compute a score of usefulness for documents written by 'Adams', given a query network `query` and global collection statistics `stats`. The `getBL` function in the inner `map` computes concept probabilities $\mathcal{P}(\mathrm{C}_j \mid \mathrm{O}_i)$, which are combined to a document probability $\mathcal{P}(\mathcal{I} \mid \mathrm{O}_i)$ in the outer `map`.

```
BAG<
  TUPLE<                            map[sum(THIS)](
    Author : str,                     map[getBL( THIS.Content, query, stats ) ](
    Content: DOCREP                     select[ =( THIS.Author, "Adams" ) ]( docs )
  >                                 ));
>;
```

Clustering feature spaces is not yet part of the database implementation. Rather, we use the Autoclass implementation of Bayesian unsupervised clustering.[25] We use Perl scripts[26] to post-process Autoclass's output to produce data files that are then bulk-loaded into the database.

5.3. Case study: music retrieval

We are mainly interested in the subjective aspects of multimedia querying by content. Because subjective user judgments are especially important with music, we investigate the field of music retrieval. Searching music collections by content would be useful for the small Dutch start-up 'Symbol Automatisering', that develops tools supporting the creation of radio programs and selecting background music in bars or restaurants. For their users, the similarity between pieces of music is not defined by the melody, but rather by a similar overall 'sound' of the music.

We implemented a feature extractor for content-based retrieval of audio following the Musclefish approach.[27] We also implemented a very simple rhythm detector based on peaks in the autocorrelation function of the lowest parts of the frequency domain. The data set *Symbol-1*, created for this case study, consists of 287 songs. From each song, we sampled between one and two minutes of digitized data which we segmented into fragments of 5 seconds each. This results in a collection of 3363 fragments. For these fragments, we calculated the feature vectors, classified the feature space using Autoclass (which resulted in 53 clusters), and loaded the database with the output of the clustering process. For each fragment, we assigned the concept node according to the cluster with the highest probability. We then modeled a song as a collection of these concepts. For the computation of probabilities in the network, we treated the songs represented in clusters as if they were text documents.

To get an impression of the resulting system, we did the following experiment. The songs in the collection have been manually classified into six main categories by domain experts at Symbol Automatisering: rock, house, alternative, easy listening, dance, and classical. We simulated relevance feedback in an iterative retrieval process by taking the most frequently occurring concepts in half the songs belonging to a category. We then used this query to see if it retrieves the other songs of the same category. Of the top 20 songs for the query based on 'rock', 15 had also manually been classified as rock. Of the other 5 songs, only 2 do clearly not classify as rock songs. With the 'classical' and 'house' songs, we found hardly any misses. Results for the category 'alternative' were however hardly better than chance, but maybe this is partly because the category is not well defined. The current system can not be used for automatic song classification. However, it seems possible to interactively retrieve groups of similar songs, particularly for well defined categories.

The authors would like to stress that due to the small scale of these experiments, they do not provide much evidence that our retrieval model is effective for music retrieval, let alone multimedia retrieval in general. The experiments have only been reported as proof of concept to get an idea whether a retrieval model originally developed for text retrieval can indeed be generalized to multimedia retrieval. Much more experimentation is necessary before we can draw more definite conclusions about our approach to retrieval.

6. CONCLUSIONS AND FURTHER WORK

Multimedia digital libraries are not supported well by traditional database technology. We categorized the participants in the digital library into two groups: the producers and the consumers. Both groups of users impose new requirements on the design of multimedia databases. The producers are more or less independent parties that co-operate in the digital library. They need an open distributed environment rather than a single monolithic system. Also, the producers require extensibility of data types and operations. The consumers are interested in access to the data in the multimedia library. The content of multimedia data is modeled with features describing syntactic aspects like color distribution or texture. To bridge the conceptual gap between these feature representations and the user's information need, the database should support an iterative query process.

The Mirror architecture presented in this paper addresses the requirements of both user groups. The combination of an open distributed environment with the notion of a central unit of control makes our multimedia database more open, without sacrificing opportunities for query optimization and concurrency control. The data model supports extensibility of data types and operations. Distribution of operations is possible using the deamon paradigm. The query processor based on InQuery's retrieval model supports the end users with query formulation. The design of the conceptual level allows the user to query both the logical and the content structure of multimedia objects. The use of different data models in the logical and the physical database design provides data independence and allows the application of algebraic query optimization.

The main contribution of our work is that the Mirror architecture is designed for scalability. We develop a multimedia database system in such a way that it may benefit from the techniques that made relational databases so popular for business applications. We are still a long way from implementing query optimization, concurrency control and caching policies. However, although the current prototype implementation of the proposed architecture is very immature, its design provides the opportunity to build a complete system that provides performance and scalability.

Improving the basic functionality of the prototype is a topic high on our research agenda. From a technical viewpoint, we should implement clustering in our architecture. We need a tool to translate query specification at the conceptual level to MOA expressions at the logical level. Also, the MOA tools need further development for data definition and manipulation. From a theoretical viewpoint, we wish to develop better probability models based on the clustering. Also, we want to experiment with multiple representations in the database. The foundation of the model in the theory of probabilistic networks provides a strong theoretical framework.[28–30] Within this framework, there is a lot of scope for experiments and we would like to investigate its use to model and learn dependencies between representations.

An important but open research issue is the development of an evaluation methodology for multimedia retrieval. The inherent subjectivity in multimedia searching makes it impossible to develop a test suite that is not related to a real user task. We believe the music domain provides a context well suited to evaluate how the query process adapts to subjectivity of the users. However, content modeling of music is not easy and the success criteria are vaguely defined. To evaluate the effect of multiple representations and their interdependencies in retrieval, retrieval from publishers' photo and video archives may provide a better context. However, the challenge in this domain is to construct a test suite with realistic user tasks and clearly defined success factors, without making the evaluation process too expensive (amount of data) and elaborate (user studies).

ACKNOWLEDGMENTS

We would like to thank especially Dick Theissens of Symbol Automatisering for providing data for the music retrieval experiments. Further thanks go to the Monet team of the CWI in Amsterdam, and to Peter Boncz in particular. We would also like to acknowledge the help of Jan Flokstra and Annita Wilschut with MOA. Finally, Henk Ernst Blok and our students Erik van het Hof and Harold Oortwijn contributed significantly to the prototype implementation.

REFERENCES

1. T. Mills, K. Moody, and K. Rodden., "Cobra: a new approach to IR system design," in *Proceedings of RIAO'97*, pp. 425–449, July 1997.
2. E. Bertino, B. Catania, and E. Ferrari, *Multimedia Databases in Perspective*, ch. Query Processing, pp. 181–217. Springer Verlag, 1997.
3. A. de Vries, G. van der Veer, and H. Blanken, "Let's talk about it: Dialogues with multimedia databases. Database support for human activity," *Displays* **18**(4), pp. 215–220, 1998.
4. M. Flyckner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker, "Query by image content: The QBIC system," *IEEE Computer* **28**(9), pp. 23–31, 1995.
5. J. Smith and S.-F. Chang, *Intelligent multimedia information retrieval*, ch. Querying by color regions using the VisualSEEK content-based visual query system, pp. 23–41. AAAI Press/MIT Press, 1997.
6. D. Gemmell, "Multimedia storage servers: A tutorial," *IEEE Computer* **28**, pp. 40–49, May 1995.
7. I. Leslie, D. McAuley, R. Black, T. Roscoe, P. Barham, D. Evers, R. Fairbairns, and E. Hyden, "The Design and Implementation of an Operating System to Support Distributed Multimedia Applications," *IEEE Journal on Selected Areas in Communication* **14**, pp. 1280–1297, September 1996.
8. A. Silberschatz and S. Zdonik, "Strategic directions in database systems - breaking out of the box," *ACM Computing Surveys* **28**, pp. 764–778, December 1996.
9. G. Sagerer and H. Niemann, *Semantic networks for understanding scenes*, ch. Knowledge representation. Plenum press, 1997.
10. A. de Vries, B. Eberman, and D. Kovalcin, "The design and implementation of an infrastructure for multimedia digital libraries," in *Proceedings of the 1998 International Database Engineering & Applications Symposium*, pp. 103–110, (Cardiff, UK), July 1998.
11. P. Boncz and M. Kersten, "Monet: An impressionist sketch of an advanced database system," in *BIWIT'95: Basque international workshop on information technology*, July 1995.
12. P. Boncz, C. Quak, and M. Kersten, "Monet and its geographic extensions," in *Proceedings of the 1996 EDBT conference*, 1996.
13. N. Nes and M. Kersten, "The Acoi algebra: A query algebra for image retrieval systems," in *Advances in Databases. 16th British National Conference on Databases, BNCOD 16*, pp. 77–88, (Cardiff, Wales, UK), July 1998.
14. R. Orfali and D. Harkey, *Client/Server programming with JAVA and CORBA*, John Wiley & Sons, Inc., 1997.
15. C. Meghini, F. Rabitti, and C. Thanos, "Conceptual modeling of multimedia documents," *IEEE Computer* **24**, pp. 23–30, October 1991.
16. T. Minka and R. Picard, "Interactive learning using a "society of models"," *Pattern Recognition* **30**, pp. 565–581, April 1997.
17. G. Sheikholeslami, S. Chatterjee, and A. Zhang, "Neuromerge: An approach for merging heterogeneous features in content-based image retrieval systems," in *4th International Workshop on Multi-Media Database Management Systems (IW-MMDBMS'98)*, (Dayton, Ohio), August 5–7 1998.
18. U. Gargi and R. Kasturi, "An evaluation of color histogram based methods in video indexing," in *Proceedings of the first international workshop on Image databases and multi-media search (IDB-MMS '96)*, pp. 75–82, (Amsterdam, The Netherlands), August 1996.

19. C. van Rijsbergen, *Information retrieval*, Butterworths, London, 2nd ed., 1979. Out of print, available online from http://www.dcs.glasgow.ac.uk/Keith/Preface.html.
20. H. Turtle, *Inference networks for document retrieval.* PhD thesis, Univeristy of Massachusetts, 1991.
21. J. Callan, W. Croft, and S. Harding, "The INQUERY retrieval system," in *Proceedings of the 3rd international conference on database and expert systems applications*, pp. 78–83, 1992.
22. A. de Vries and H. Blanken, "The relationship between IR and multimedia databases," in *The 20th IRSG colloquium: discovering new worlds of IR*, (Grenoble, France), March 1998.
23. P. Boncz, A. Wilschut, and M. Kersten, "Flattening an object algebra to provide performance," in *Fourteenth International Conference on Data Engineering*, pp. 568–577, (Orlando, Florida), February 1998.
24. A. de Vries and A. Wilschut, "On the integration of IR and databases," in *Database issues in multimedia; short paper proceedings, international conference on database semantics (DS-8)*, pp. 16–31, (Rotorua, New Zealand), January 1999.
25. P. Cheeseman and J. Stutz, "Bayesian classification (AutoClass): Theory and results," in *Advances in Knowledge Discovery and Data Mining*, AAAI Press, 1995.
26. L. Wall, T. Christiansen, and R. Schwartz, *Programming PERL*, O'Reilly & Associates, Inc., 2nd edition ed., 1996.
27. E. Wold, T. Blum, D. Keisler, and J. Wheaton, "Content-based classification, search, and retrieval of audio," *IEEE Multimedia* **3**(3), pp. 27–36, 1996.
28. J. Pearl, *Probabilistic reasoning in intelligent systems: Networks of Plausible Inference*, Morgan Kaufmann, California, 1988.
29. R. Fung and B. D. Favero, "Applying Bayesian networks to information retrieval," *Communications of the ACM* **38**, pp. 43–48, March 1995.
30. D. Heckerman, "A tutorial on learning with Bayesian networks," Tech. Rep. MSR–TR–95–06, Microsoft Research, Advanced technology division, March 1995. Revised edition November 1996.