Towards the Integration of a Query Mechanism and Navigation for Retrieval of Data on Multimedia Documents

E. Duval*
H. Olivié
Department of Computer Science
Katholieke Universiteit Leuven
Celestijnenlaan 200 A, B-3001 Heverlee, Belgium
E-mail: Erik.Duval@cs.kuleuven.ac.be

July 29, 1992

Abstract

Traditionally, two paradigms are discerned with respect to data retrieval [3, 17]:

1. A *query mechanism* enables the user to express conditions that characterize the data he is looking for. The Data Base Management System (DBMS) is responsible for retrieval of the data that satisfy these conditions.
2. *Navigation* or *browsing* can be used to investigate the content of the database by following links between related data.

We will start with a discussion of these paradigms in the first two sections. We will then explore how both approaches can be combined. Finally, a practical implementation of the advocated approach will be presented. The implemented prototype supports access to a database on multimedia material. A more elaborated discussion of our ideas and results in this area can be found in [16].

1 Query Mechanism

1.1 Introduction

As databases can hold enormous amounts of data, it is important to identify precisely the data one is interested in, so that only these will be included in the result of the query.

*This work has been partially supported by the DELTA project CAPTIVE (D1013)

We will use the generic term *entity* to refer to a 'real world object' (e.g. 'Andy Warhol'). Entities are related to a *topic* (e.g. 'Pop Art'). They are described by tuples in relational databases, records in hierarchical and network databases, objects in object oriented databases. In a database on documents or a document retrieval system, the entities are documents. (The difference between these two kinds of systems is that the former holds *data about* the documents, whereas the latter also holds the documents themselves.) Note that our entity concept is not exactly the same as the one used in entity relationship modelling [7, 17]. (There is some relation between the two concepts but we will not pursue this issue here.)

1.2 Two Components of a Query

A query identifies precisely the data that a user wants to retrieve from the database. It is expressed in a query language that enables the user to:

1. express search conditions that *identify the topic* he is interested in: e.g. 'cars built in Italy before 1950'.

   In SQL [32], the standard query language for relational databases, search conditions for topic identification are expressed in the WHERE-clause. Topic identification is also called *retrieval conditioning* [8].

2. *select the kind of data* he wants to retrieve about the entities in the database that are related to a particular topic: e.g. 'the name and quantity produced of cars ...'.

   This part of the query selects a number of attributes (in relational and object oriented databases) or fields (in hierarchical and network databases) whose values need to be retrieved for the relevant entities. In relational algebra [8, 13, 17], data selection corresponds to the projection of a relation, expressed in the SELECT-clause of SQL queries [32]. Data selection is called *retrieval targeting* in [8].

1.3 Declarativene Nature of the Query Approach

A very important advantage of the query approach is its declarative nature: the user identifies *what* he wants to retrieve from the database. The Data Base Management System (DBMS) will take care of *how* these data can be found within the possibly abundant amount of data stored in the database. Thus the user need not be aware of the difficulties of data management and retrieval. This results in higher level interaction with the database, with more powerfull data retrieval facilities for the user.
The query approach is traditionally mainly used within the context of relational databases, whereas data manipulation languages for network and hierarchical databases mostly rely on the navigational paradigm (see section 2) [17]. However, a query mechanism can also be supported on top of a hierarchical or network database. The approach presented in section 3 on the other hand aims at integrating navigational access into a relational environment.

1.4 On the Quality of Search Conditions

When querying a database on documents, precise topic identification is often problematic, as it is very difficult to model the content of a document very precisely, e.g. by assigning a set of keywords to each document. Obviously, if the search conditions are

1. *too strict*, then some relevant entities will not satisfy the conditions and data on these entities will not be included in the result of the query;

2. *too broad*, then the result will include data on entities that are not relevant to the user.

Topics can be identified by different sets of search conditions. To asses the quality of these sets, the following two criteria are usually considered [4, 9, 29], although other criteria have also been defined [30]:

1. the *response factor* or *recall*: This is the percentage of the relevant entities that is included in the result of the query.

$$response\ factor = \frac{(number\ of\ relevant\ entities\ in\ result)}{(total\ number\ of\ relevant\ entities\ in\ system)}$$

   So, if a database holds 50 relevant entities and the result of a query includes 24 of these, then the response factor of that query is 24/50 = 48%. The response factor can be interpreted as the probability that a relevant entity will be retrieved.

   Of course, regular users are unable to calculate the response factor, as they cannot know how many relevant entities there are in the database. Researchers however can control the number of documents to be retrieved by their (human) laboratory rabbits in experiments. Based on the number of target documents actually retrieved, the response factor can then be calculated.

   An indication of the response factor can also be obtained as follows: a limited number x of relevant entities can be stored in the system before submitting a query. Only a fraction $y \leq x$ of these entities will be included in the result of the query (possibly together with a lot of other entities that were already present in the database). The number y/x can be considered an approximation of the response factor. The difficulty with this approach is that the control set of x relevant entities should be representative for all relevant entities. (See also [4] for a similar approach, based on a control set that is particularly rich in relevant entities. [30] deals with problems that arise when the accuracy factor needs to be determined for a set of queries or when entities in the system are ordered according to their probable relevance.)

2. the ***accuracy factor*** or ***precision***: This criterium refers to the percentage of the entities in the result that are relevant.

   $$accuracy\ factor = \frac{(number\ of\ relevant\ entities\ in\ result)}{(total\ number\ of\ entities\ in\ result)}$$

   If, in the previous example, the total result of the query would consist of 40 entities then the accuracy factor would be 24/40 = 60%. The accuracy factor can be interpreted as the probability that an entity in the result is relevant.

   This factor can very well be calculated by ordinary users themselves and gives an indication of the usefulness of the data that belong to the result.

1.5 Topic Identification: a Compromise

Although the goal is to obtain a response factor and an accuracy factor that are both as close to 100% as possible, it has been empirically observed [4, 9] that, in the context of document retrieval, the sum of both factors is typically less than 100%:

$$response\ factor + accuracy\ factor \leq 100\%$$

The reason behind this observation is that when trying to augment one factor, one automatically reduces the other. In order to obtain a high response factor e.g., topic identification will be based on very weak search conditions, so that no relevant entities will inadvertently be excluded from the result. Consequently, the result will also contain a lot of irrelevant entities. In other words: the accuracy factor will be low.
To take this to an extreme: in order to obtain a response factor of 100%, one ought to retrieve all entities from the database, just to make sure that not one relevant entity is not included in the result. The corresponding accuracy factor will be

$$\frac{(number\ of\ relevant\ entities\ in\ the\ database)}{(total\ number\ of\ entities\ in\ the\ database)}$$

and this number can be very small.
On the other hand, the more accurate a query is, the less response it will deliver: if one wants to be absolutely sure that the result does not include a single irrelevant entity, then the search conditions will be so stringent that the result will include almost no entities whatsoever.
Moreover, as databases grow bigger and bigger, users will have to sort out the relevant material from a a huge amount of irrelevant material if they want to find most of the relevant material. Suppose e.g. that a database holds 100 relevant entities. If a user wants 60% of these to be included in the answer to his query, then:

$$\begin{array}{l} response\ factor = 60\% \\ \underline{accuracy\ factor \leq 100\% - response\ factor} \\ => accuracy\ factor \leq 40\% \end{array}$$

This implies that the result will consist of $\pm 60/0.4 = 150$ entities, including 90 irrelevant ones. Also, $100 - 60 = 40$ relevant documents will not be included in the answer. If the user wants to be sure he retrieves 90% of the relevant documents, then the accuracy factor will drop to around 10%, and the answer will consist of $\pm 90/0.1 = 900$ entities. The user will have to sort out this material himself, in order to identify that part of it that is really relevant. If a high response factor is required, then the so called futility point may be reached [29]: this happens when the system returns so many documents that the user is drowned in irrelevant or partly relevant documents and decides to quit browsing the results before he finds the document he was searching.

2 Navigation

2.1 Introduction

Navigation is another paradigm for information retrieval. For many years, this used to be the most widespread access method for ***hierarchical*** and ***network*** databases [3, 17]. The more recent commercial success of and research interest in ***hypertext*** [11, 12, 28, 31] and *hypermedia* [2, 23, 28] systems has lead to a renewed interest in this paradigm.

2.2 Navigation as Another Paradigm for Document Retrieval

Hypertext [11, 12, 28, 31] enables readers to navigate through atomic pieces of information that are linked to one another. This approach can also be used for access to databases: navigational tools (commonly called 'browsers') can be constructed that enable the user to wander around in and inspect the content of a database, pursuing links between the different data items. One can start e.g. with a certain person, follow a family link to identify his father who may have worked for a particular company. The link to this company can be used in order to obtain more information about the company, its products, etc. Data manipulation in hierarchical and network databases has traditionally been based on this paradigm [17].
Hypermedia systems [2, 23, 28] apply the approach sketched above (hence 'hyper-') to multimedia systems, resulting in systems that offer navigational access to text, images, sound, etc. Integration into these systems of techniques that originated in Artificial Intelligence research is attempted in so called 'intelligent hypermedia systems', hypermedia systems that focus heavily on knowledge representation [2, 23].
The main ***advantage*** of the navigational paradigm is that it makes users aware of the context and structure of the information. If the structure is well designed, then users can find their way in a large amount of information. Moreover, navigational support can be provided to the user, as the retrieval system can trace back how the user arrived where he is. Expert system techniques e.g. can be used to infer the ultimate goals that underly the actions taken by the user. Thus the system may suggest interesting alternative exploring routes when the user is unable to locate the information he is looking for.

The main *problem* however is to structure the data in a way that suits every user. That is why sometimes different structures are supported, corresponding to different 'viewpoints' [24] (called 'views' in [5]) on the subject domain. The user can then be provided the structure that corresponds most closely to his own view.

2.3 Lost in Hyperspace and How to Avoid it

Users sometimes loose their way as they wander around in the data that are presented to them, following links that lead them in a direction they can often hardly foresee. This problem has become known in the hypertext community as the 'lost in hyperspace' syndrome [28].

A navigation structure respresents a viewpoint on the data that should correspond as closely as possible to that of the user. The better the semantic representation of the navigation structure and the more appropriate the structure, the less severe the problem will be [14]. Both local and global maps, with an indication of the user's current position, can be very helpful in this respect.

Moreover, facilities ought to be provided that enable a user to get back on the right track once he does get lost. A backtrack mechanism takes the user back to the previous node in the navigation structure. Users should also be able to mark items while navigating: at any moment they should be able to get back to one of the marked items. Both these mechanisms enable users to get back to a node in the information structure that they are familiar with.

It can also be very helpfull if the user is able to suppress temporarily a part of the structure, so that only those links that are important to him appear on the screen [6]. This reduces the number of links that lead him to a part of the information structure where he might get lost. Of course, it should also be possible to display the original structure again.

Finally, users ought to be able to add their own structures, thus creating access paths that correspond to their own view on the subject domain.

As explained in [5], the author of a navigational structure is confronted with a similar problem: he may become disoriented while gathering the information on which the structure will be based. View administration tools may help in overcoming this problem, but we will not pursue this issue here.

3 Integration of a Query Mechanism and Navigation

In this section, we present a paradigm for retrieval of data on multimedia documents that combines a query mechanism (section 1) and navigation (see section 2). First, in section 3.1, some problems will be discussed that occur when a user, constructing a query, needs to choose appropriate attribute values for topic identification conditions. In section 3.2, navigation in the domain of values for an attribute will be presented as an interesting alternative for the determination of comparison values. The benefits of this approach in the context of content modelling for a database on multimedia documents will be covered in section 3.3. Finally, section 3.4 will show that navigation in the domain of attribute valuc requires more database management support for attribute domains.

3.1 Choice of Comparison Values: Problems

The topic identification part of a query (see section 1.4) is based on comparison ($=$, $<$, $\leq$, $>$, $\geq$, $\neq$) of attribute values for entities in the database with comparison values from the same domain: e.g. 'employees that work for a department with name = "research"' or 'people whose age $\geq$ 100'.
Choosing appropriate comparison values can sometimes be difficult:

1. *The comparison value must belong to the attribute domain* [8, 13]. That is why e.g. 'people whose age $\geq$ 100' is a valid search condition and 'people whose age = "research"' is not.

   However, the domain of a specific attribute is not always clear, particularly for casual users. A database on hotel reservations e.g. might have an attribute 'hotel category'. The domain of this attribute can be
   {0, 1, 2, 3, 4, 5}, or
   {'', '*', '**', '***', '****', '*****'}, or
   {'rudimentary', 'basic', 'comfortable', 'luxurious'}, etc.

   If the user is a computer litterate, then he may be able to find out that the type of the attribute 'hotel category' is the set of strings, but even then he still doesn't know wether it is
   {'', '*', '**', '***', '****', '*****'} or
   {'rudimentary', 'basic', 'comfortable', 'luxurious'}.

   Note that the two domains have a different number of elements. We are currently further investigating the relationship between data types and domains.

   The consequence is that when a query for all hotels in the category '*' yields an empty result, this may imply that there are no hotels in the relevant category. However, the empty result can also be caused by the fact that '*' does not belong to the domain of hotel categories, in which case it may very well be that there are plenty of hotels that do meet the user's demands.

2. Even if the domain is known, then *the meaning of the values may be unclear*, if their interpretation isn't obvious [26].

   If e.g. the domain of hotel categories is
   {0, 1, 2, 3, 4, 5}
   then, without additional information, it is unclear wether 0 is the most or the least luxurious category. An extreme consequence could be that someone looking for very basic lodgings spends a lot of money as he inadvertently books a luxurious suite.

3. Even if the domain and meaning of values for a particular attribute are clear, the selection of appropriate comparison values for topic identification can be difficult when the user has no idea of ***the values actually present in the database.*** Although aggregate functions can be used in order to obtain the minimum, maximum, average, total number of different values, etc., this kind of information will not be sufficient in many cases.

   Suppose e.g. that one wants to retrieve information about the ten highest mountains on which information is stored in a database with geographical information. As most query languages support sorting based on some value of the result, one could retrieve the desired information for all mountains, and sort the result according to mountain height. Following this approach, the query topic will be 'mountains', although the user is only interested in 'the ten highest mountains'. Consequently, a lot of unnecessary information will be retrieved from the database. This can be a severe problem in the context of e.g. multimedia databases where the data related to a particular entity can be quite voluminous [18, 19, 22, 27].

3.2 Navigation in Attribute Domain to Determine Values

The problems mentioned in the previous section can be solved when the comparison values are determined by navigation in their domain. For this purpose, the domain can e.g. be presented to the user as a sorted *list of values* if the domain can be ordered in a meaningful way and contains only a limited number of values. A more general representation of the domain is a *network* of nodes that represent a value and that are connected to other nodes representing related values.
Navigation starts e.g. at the first or last node according to some ordering. Users follow links to nodes representing related values. At any node, users can include the value represented by that node in the set of comparison values for topic identification.
The three problems presented in section 3.1 can now be solved easily:

1. When the comparison values are determined by navigation, *they will necessarily belong to the attribute domain*, as the user cannot navigate outside the domain.

   When selecting comparison values for topic identification in e.g. a database on hotel reservations, the list of all possible values for the attribute 'hotel category' can be presented to the user. Navigation in this simple case is limited to selection of a set of values in the list.

2. Contextual information can be made available in a navigational system in order to help casual users to *understand the meaning* of the domain values.

   In the case of a database on hotel reservations, contextual information can include the ordering of the different values, as this indicates how the different values relate to each other. It can also include e.g. explanations about the meaning of the different categories, made available to the user upon request.

3. When information about the ***values actually stored*** in the database is made available to the user, the third problem mentioned in section 3.1 disappears.

   When the user is e.g. querying a database with geographic information, a sorted list of heigths of mountains actually described in the database could be presented upon request. The user can e.g. select the ten highest values and use this set in the condition for topic identification.

More elaborated examples present the benefits of this approach in sections 3.3.2 and 4.

3.3 A More Complex Example: Content Modelling

The advantages of the advocated approach (navigation in the domain of an attribute for selection of appropriate values) become more apparent when the domain is more complex. Consider e.g. a text retrieval system (a database of alphanumerical documents).

3.3.1 Keywords

Keywords are often used in text retrieval systems to model the content of documents, as:

1. this approach is very flexible and easy to understand;
2. the keywords can be generated automatically, because the document itself is a collection of words [9]

Reconsidering the problems mentioned in section 3.1, we see that:

1. The *definition of the domain* is relatively simple: the set of strings. The maximum length of a string can be limited. Sometimes the domain contains only a limited number of keywords, e.g. in order to avoid complications with synonyms. In the latter case, the keywords selected by the user for topic identification should belong to this domain.
2. It may be quite problematic to characterize the content of a document accurately with only a limited number of keywords, as their *meaning is often unclear* without contextual information [6]. Documents described by the keyword 'food' e.g. may relate to the effect of nutritional habits on life expectancy, or the food production in different countries, or the chemical substances contained in different kind of foods, or appropriate dietetic schemes for specific diseases, or famines in the third world, or national agricultural policies, etc., etc.
3. It may be important for the user to know which *keywords* are *actually stored* in the database, e.g. because of the presence of synonyms. If he is interested in documents described by the keyword 'manufacturing', then he will probably also want to know wether there are documents in the database described by the keyword 'production'. This leads to problems if there is no adequate support, as the user can never be sure he has included all possible keywords that may describe the documents he is looking for.

3.3.2 Classification

As an alternative for the description with keywords, a document can also be *classified* in a structure describing the content domain. Standardized such structures exist for many specific scientific domains. In the Captive project (see section 4) e.g., we used the Medical Subject Headings (MESH) to classify audio-visual material related to medicine [15, 24]. Similar structures for e.g. computer science have also been defined [1]. More general classifications for non-fiction documents are used by libraries; these include amongst others the Dewey Decimal Classification (DDC), the Universal Decimal Classification (UDC) and the 'Schema voor de Indeling van de Systematische catalogus in Openbare bibliotheken (SISO)', used in the Netherlands and Belgium [20].
The classification structure is commonly a hierarchy and consequently a tree, but the approach we advocate can be used for network structures in general. Each node in the structure corresponds to a particular topic, which can be identified by a set of keywords, as in the previous section. The classification of a document in the structure can then be represented by the set of keywords identifying the nodes that correspond to the topics treated in the document. This set of keywords can be stored in the database to describe the content of the document.
When the user wants to query the database, he navigates in the domain structure, selecting nodes that correspond to the topics he is interested in. The selected nodes can be represented by the set of keywords identifying the nodes. It is then possible to automatically construct a query that retrieves from the database the documents connected to the selected topics. The topic identification part of this query can be based on a comparison of the keywords stored in the database (which describe documents), with the keywords that identify the selected nodes.
As already mentioned in section 3.2, the problems that arise when the keyword approach of section 3.3.1 is used, can thus be avoided:

1. Users do not construct keywords themselves, but select nodes in a classification. Therefore the domain of keywords is irrelevant for end users. The domain of topics is presented to them in a structured way, as a classification.

2. The navigation structure includes contextual information as nodes are connected to other nodes representing related topics. If these links are well designed (i.e. if the structure corresponds to a well elaborated viewpoint on the domain), then the meaning of the topics will be clear.

3. Exploring the domain structure, users can find out which topics are covered by the documents stored in the database.

3.4 More Support for Attribute Domains

As pointed out in the previous sections, a user may need information about the attributes stored in a database, such as their value domain, information about the meaning of the values or about the distribution of values actually stored in the database. This kind of information is commonly called ***meta-data*** [17, 26], although traditionally these do not include information about the meaning or distribution of values. However, we will use the term in a broader sense, including all additional information about attributes that is needed to support navigation for topic identification.
Clearly, the meta-data need to be stored somewhere:

1. Preferably, they should be stored *in the database itself.* This way, the database management system (DBMS) can make the data available to all application programs accessing the database. The DBMS should be responsible for the maintenance of the meta-data:

   (a) When a new attribute is defined, its domain and eventually some information about the meaning of the values ought to be included in the meta-data.

   (b) When an attribute value for a particular entity is added, modified or deleted, then the information about the values actually stored in the database ought to be modified accordingly.

   In relational databases, the meta-data can be stored in the catalog, which is itself a relation that can be accessed by application programs, in exactly the same way as data can be retrieved from user defined relations [8, 13, 17]

2. However, most currently available commercial database management systems do not provide adequate support for attribute domains (Codd considers this deficiency one of their main shortcomings in [8]). We feel that such support could be based on the concept of abstract data types, and are currently investigating this issue.

   Because of this lack of DBMS support, the maintenance of the meta-data about attributes needs to be taken care of by the application programs *outside of the database itself.* An obvious drawback to this approach is that every application program that accesses the database must implement itself the required functionality. More dangerous is the risk that application programs may change the data, without recording the corresponding changes in the meta-data. This will result in loss of integrity and incorrect system behaviour.

4 A Database on Audio-Visual Material

4.1 CAPTIVE: a short overview

In the *CAPTIVE* project [15, 24, 25], funded by the European Commision under the framework of the DELTA program, we have set up a prototype database on educational audio-visual material in the field of medicine. The audio-visual material itself is *not* stored in the database, but representative parts of it (stills, short fragments) are stored on a separate image store, based on an analogue WORM disk and developed at University College London (G.B.). Information about the parts stored on the image store is included in the database. The separation of descriptive data and the actual content leads to a design similar to that of SPRITE [33], a technical document management system.
Both the database and the image store are integrated in a European communications network, based on public packet switched data networks (PSDN, e.g. EARN, internet, etc.) [10] and direct broadcast satellite (Olympus). This network is represented in figure 1.

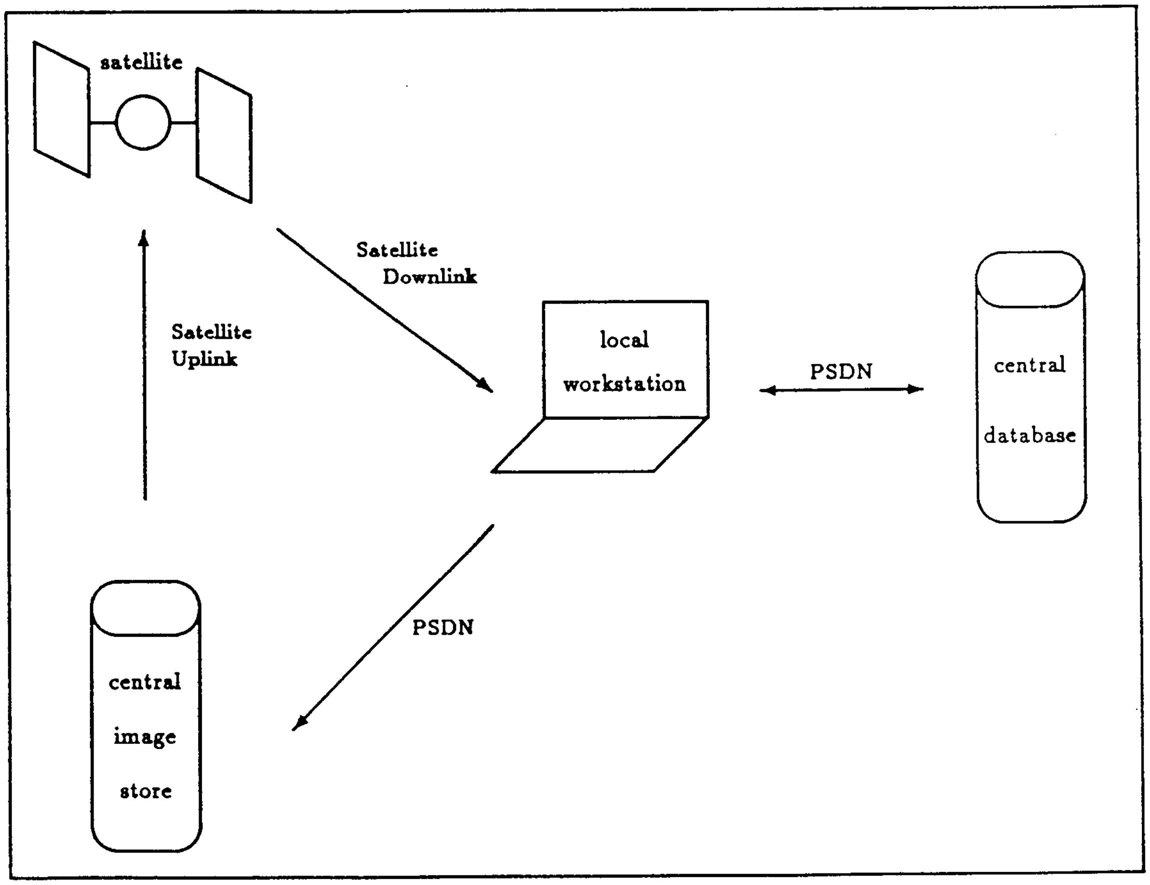
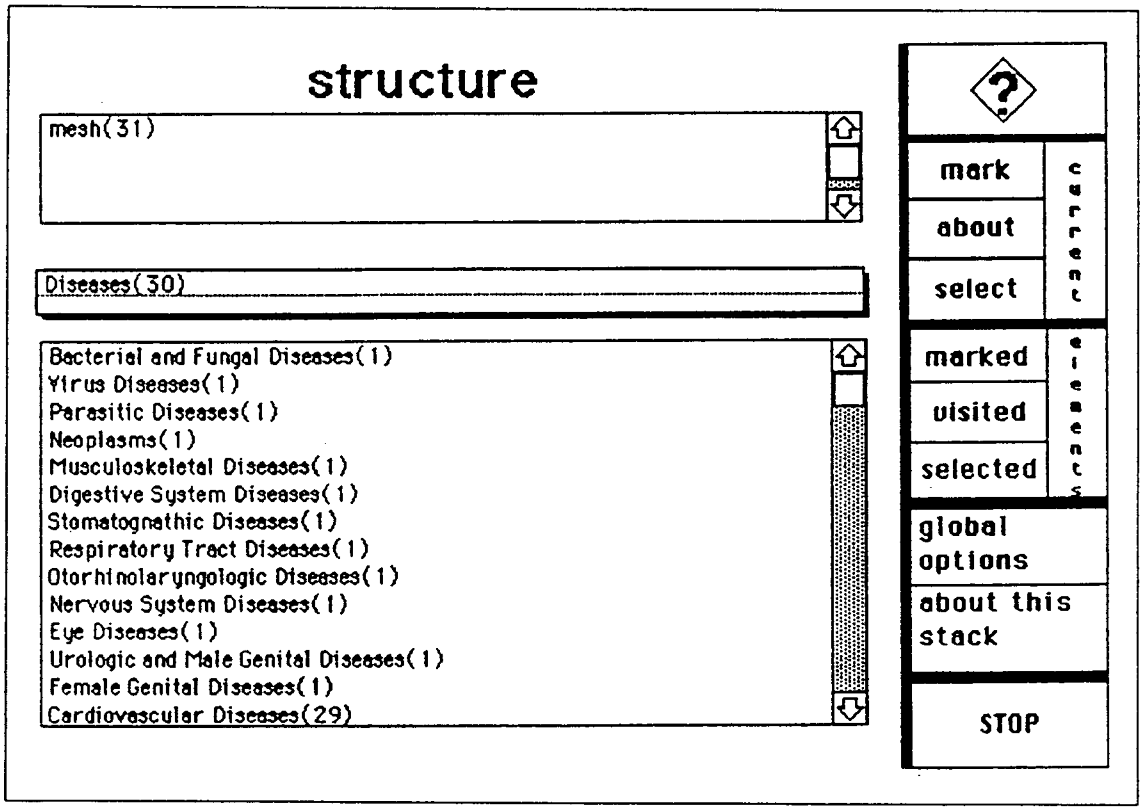


Figure 1: Captive network

In order to limit the communication traffic, users do not interact with the central computer directly, but with a local retrieval component that builds a query for the central database, transmits the query to the central computer, and initiates processing on that computer. The answer is retrieved by the local component and presented to the user upon request.

4.2 Document Retrieval: a Practical Implementation

The interaction between the local retrieval component and the user is based on a combination of the navigational and query approach along the lines of section 3. The local component supports topic identification by navigation in the Medical Subject Headings (MESH), a well elaborated classification for medical topics (the application domain for the project). Hypercard [21] was used to implement a prototype of this component. A typical screen during navigation is presented in figure 2.



mesh(31)

Diseases(30)

Bacterial and Fungal Diseases(1)
Virus Diseases(1)
Parasitic Diseases(1)
Neoplasms(1)
Musculoskeletal Diseases(1)
Digestive System Diseases(1)
Stomatognathic Diseases(1)
Respiratory Tract Diseases(1)
Otorhinolaryngologic Diseases(1)
Nervous System Diseases(1)
Eye Diseases(1)
Urologic and Male Genital Diseases(1)
Female Genital Diseases(1)
Cardiovascular Diseases(29)

Figure 2: A typical CAPTIVE navigation screen

The small middle box presents the topic corresponding to the current node during navigation in the MESH structure. The upper box holds a list of more general topics that include the current topic as a subtopic. In a hierarchical structure, the upper box contains only one topic (corresponding to the 'father' node), but in a more general network structure, the box can contain several topics. The big lower box contains a list of subtopics included by the current topic.

Facilities have been provided to help the user in overcoming the 'Lost in Hyperspace' problem (section 2.3). While navigating, users can mark nodes with the 'mark current' button. If they become disoriented later on, they can retrieve a list of the marked nodes, using the 'marked elements' button. The 'visited elements' button delivers a similar list, containing all nodes visited during a session. Both lists enable the user to go back to one of the nodes contained in the list. Information about the current topic can be obtained with the 'about current' button.
A query to the database can be constructed by selecting topics during navigation. The current topic can be included in a list of selected topics with the 'select current' button. Upon activation, the 'selected elements' button displays all selected topics, enabling the user to eventually delete elements from the list. When navigation is finished, a query is build by the local retrieval component, with its topic identification part based on the list of selected elements, as described in section 3.3.
The numbers that appear in round brackets after the topic name indicate how many different audio-visual entities (still image, sequence, video tape, etc.) about the topic are described in the database. This information can be very important as it enables a user to either refine or broaden his topic identification criteria, before even submitting the query to the database, if the number of audio-visual entities is too high or too low respectively.
Of course, there is also a help facility (upper right button), and there are some 'global options' concerning the languages used etc.

4.3 Meta-data Needed by the Local Retrieval Component

As the content of the database is rather static, the meta-data about the attributes that describe the content of the audio-visual material are stored locally. This approach limits the communication between the local user's site and the central database, as interaction with the meta-data can thus be carried out locally.
The meta-data include the classification of the domain (the MESH structure) and, for each topic included in the structure, the number of audio-visual entities described in the database that cover that topic.

4.4 The Search Continues

When alphanumerical information about the audio-visual material is retrieved from the database, this information can then be used to select the entities that look most promising and order images from the image store that are representative for the selected entities. The images can be transmitted over satellite from the central image base to the local user (see figure 4.1). Based on a review of these images, the user can finally select that multimedia material that seems best suited for his purposes and order it from the copyright holder.
We believe that this approach in stages is very important: users should be able to broaden or refine their criteria during the search process. Otherwise, they might be flooded with irrelevant information or valuable information may not be found.

4.5 The Future: Multimedia Teleschool

We have just started working on a new European DELTA project, called 'Multimedia Teleschool', again in close collaboration with University College London (G.B.). In the new project, the results and achievements of CAPTIVE will be further developed. Some of the issues involved include:

- support for multilinguality;
- development of a new digital multimedia object store;
- integration into an ISDN network;
- support for data-input;
- automatic update of locally stored meta-data;
- development of a local data input tool.

We will also aim at a tighter integration of the different components of the system, according to a client-server model. The digital multimedia object store, the analogue image server and the database will act as servers that respond to queries issued by the different clients.

5 Conclusion

We have shown that a query mechanism and navigational data access can be combined when navigation in the domain of an attribute is used to determine comparison values for the topic identification part of a query. This approach integrates to some extent the two most important data access paradigms and solves some of the problems that occur when either one of the approaches is used separately.
As explained in section 3.4, a more general implementation of the advocated approach requires more elaborate database management system support for attribute domains. We are currently investigating how such support can be organised and what its implications would be for the relational and object oriented data model.

References

[1] The full computing reviews classification system [1991 version]. *Computing Reviews*, 32(1):12–22, January 1991.

[2] Robert M. Akscyn, Donald L. McCracken, and Elise A Yoder. KMS: A distributed hypermedia system for managing knowledge in organisations. *Communications of the ACM*, 31(7):820–835, July 1988.

[3] Charles W. Bachman. The programmer as navigator. *Communications of the ACM*, 16(11):653–658, November 1973.

[4] David C. Blair and M. E. Maron. An evaluation of retrieval effectiveness for a full-text document-retrieval system. *Communications of the ACM*, 28(3):289–299, March 1985.

[5] P. D. Bruza and Th. P. van der Weide. Assessing the quality of hypertext views. *ACM SIGIR FORUM*, 24(3):6–25, Fall 1990.

[6] Robert Carr. Nieuwe gebruikers-interfaces voor cd-rom. In *CD-ROM, een nieuw opslagmedium*, pages 189–199. Kluwer technische boeken B.V., Deventer-Antwerpen, 1988. Translated from *CD-ROM. The new papyrus.*

[7] Peter Pin-Shan Chen. The entity-relationship model – toward a unified view of data. *ACM Transactions on Database Systems*, 1(1):9–36, March 1976.

[8] E. F. Codd. *The Relational Model for Database Management – Version 2.* Addison-Wesley, 1990.

[9] Gregory Colvin. De stand van zaken op het gebied van retrieval software. In *CD-ROM, een nieuw opslagmedium*, pages 163–169. Kluwer technische boeken B.V., Deventer-Antwerpen, 1988. Translated from *CD-ROM. The new papyrus.*

[10] D. Comer. *Internetworking with TCP/IP.* Prentice Hall, 1988.

[11] Jeff Conklin. Hypertext: An introduction and survey. *IEEE Computer*, 2(9):17–41, September 1987.

[12] Andries Van Dam. Hypertext '87 keynote address. *Communications of the ACM*, 31(7):887–895, July 1988.

[13] C. J. Date. *An Introduction to Database Systems.* Addison-Wesley Systems Programming Series. Addison-Wesley, fourth edition, 1986.

[14] Glorianna Davenport, Thomas Aguierre Smith, and Natalio Pincever. Cinematic primitives for multimedia. *IEEE Computer Graphics & Applications*, 11(4):67–74, July 1991.

[15] E. Duval. An 'intelligent' database system of audio-visual material. In S.A. Cerri and J. Whiting, editors, *Learning Technology in the European Communities, Proceedings of the DELTA Conference on Research and Development - The Hague, 18-19 October 1990*, pages 231–241. Kluwer Academic Publishers, 1992.

[16] E. Duval and H. Olivié. Integrating a query mechanism and navigation for data retrieval. Technical Report CW 147, K.U.Leuven, May 1992.

[17] Ramez Elmasri and Shamkant B. Navathe. *Fundamentals of Database Systems.* Benjamin/Cummings, 1989.

[18] Edward A. Fox. The coming revolution in interactive digital video. *Communications of the ACM*, 32(7):794–801, July 1989.

[19] Edward A. Fox. Standards and the emergence of digital multimedia systems. *Communications of the ACM*, 34(4):26–29, April 1991.

[20] G. Genie, E. Heidbuchel, R. Van den Bremt, W. Vanderpijpen, and L. Vervenne, editors. *Inleiding tot het openbaar bibliotheekwerk. Syllabus van de leergangen tot het behalen van de akte van bekwaamheid voor het ambt van bibliothecaris in een openbare bibliotheek*, chapter E. Indeling van de wetenschap, pages E1–E6. Dienst voor Openbaar Bibliotheekwerk, Brussel, sixth, augmented, revised and corrected edition, 1988.

[21] Danny Goodman. *The complete Hypercard 2.0 handbook*. Bantam books Toronto, third edition, 1990.

[22] Jack Grimes and Mike Potel. Guest editors' introduction: Multimedia - it's actually useful! *IEEE Computer Graphics & Applications*, 11(4):24–25, July 1991.

[23] Frank G. Halasz. Reflections on notecards: Seven issues for the next generation of hypermedia systems. *Communications of the ACM*, 31(7):836–852, July 1988.

[24] J. Van Heddegem, H. Olivié, and E. Duval. Final report on an intelligent knowledge base system used within the framework of the captive project. Technical Report D1013/WP2.1, K.U.Leuven, March 1991.

[25] D.G. Jameson. Captive final project report. Technical Report D1013/P7097, DELTA Exploratory Action, April 1991.

[26] Roger G. Johnson. Integrating data and metadata to enhance the user interface. In J. Longstaff, editor, *Proceedings of the Third British National Conference on Databases (BNCOD3)*, British Computer Society Workshop Series, pages 29–39. Cambridge University Press, 11-13 July 1984.

[27] A. Desai Narasimhalu and Stavros Christodoulakis. Guest editors' introduction: Multimedia information systems - the unfolding of a reality. *IEEE Computer*, 24(10):6–8, October 1991.

[28] Jakob Nielsen. *Hypertext & Hypermedia*. Academic Press, 1990.

[29] Hans Paijmans. An inventory of models in information retrieval. In Hans Weigand and Hans Paaijmans, editors, *Workshop Artificial Intelligence en Information Retrieval*, pages 6–18, June 1992.

[30] Vijay V. Raghavan, Peter Bollmann, and Gwang S. Jung. Retrieval system evaluation using recall and precision: Problems and answers. In N. J. Belkin and C. J. van

Rijsbergen, editors, *Proceedings of the Twelfth Annual International ACMSIGIR Conference on Research and Development in Information Retrieval*, pages 59–68. ACM Press, 1989. Special Issue of the SIGIR FORUM.

[31] John B. Smith and Stephen F. Weiss. Hypertext. *Communications of the ACM*, 31(7):816–819, July 1988.

[32] Rick F. van der Lans. *The SQL standard. A complete reference.* Prentice Hall Academic Service, 1988.

[33] Hans Weigand, Maria Briales, Joost Dijkstra, Berrie Kremers, Frans Laurijssen, Liu Ling, Mark Martens, Robert Meersman, and Olga de Troyer. Implementation of a document management system. In Hans Weigand and Hans Paaijmans, editors, *Workshop Artificial Intelligence en Information Retrieval*, pages 62–73, June 1992.