

Concepts for a Graphical User Interface for Hypermedia Retrieval

Mounia Lalmas¹, Thomas Rölleke², Frank Turra³, and Norbert Fuhr²

¹ Queen Mary & Westfield College, London, England

² University of Dortmund, Germany

³ PRO-DV, Dortmund, Germany

Abstract. Which concepts are general and specific enough to support hypermedia retrieval? We present in this paper a graphical user interface that makes explicit the following three key concepts for hypermedia retrieval: *content of documents*, *facts about documents* and *structure of documents*. The underlying model is based on a probabilistic object-oriented logic (POOL) that enables content-based querying, fact-based querying, and the exploitation of the structural nature of hypermedia documents. In this paper, we focus on the graphical user interface which reflects the expressiveness of POOL for querying hypermedia documents. We report on the application of the system using a heterogeneous collection of hypermedia documents.

1 Introduction

For nearly a decade, an exponentially growing amount of multimedia electronic information has become widely available, e.g. on CD-ROM, on the Internet and in digital libraries. Information retrieval (IR) aims to provide fast, effective and efficient methods of representing, managing, retrieving and displaying such information [15]. Increasingly typical for electronic documents is their hypermedia character: (1) they are composed of objects of various media (e.g., text, image); (2) objects may be organised into a logical structure (e.g., a chapter and its sections) or a semantic structure (e.g., objects created by the same authors are linked together); and (3) relationships may exist between objects (e.g., spatial and temporal relationships in, respectively, images and videos).

Hypermedia retrieval requires more sophisticated querying facilities and document representations than standard IR. Consider the following example of an information need:

“Find any information about sailing in the Scottish islands. I am looking for nice pictures, for example, containing a sailing boat in front of an impressive old castle. The pictures should have a landscape orientation. Also, check which harbours and companies charter sailing yachts of approximatively 36 feet long. Finally, the hire price per week should not be more than 1000 pounds.”

Standard retrieval systems will be ineffective in providing precise answers to such a query, mainly because search engines are mostly based on a purely

term-based representation of documents and queries; i.e., a query is a bag of terms, sometimes connected by boolean operators. However, hypermedia documents contain more information than is reflected by a bag of non-related terms. Documents contain information about objects: they classify objects (e.g., in a picture we identify an object that is classified as a sailing boat) and they relate objects (e.g., in a picture a sailing boat and a castle are related by the spatial relationship “in front of”).

The need for conceptual models for representing and retrieving hypermedia documents has been pointed by many (e.g., [8,5,7]). [9,10,1] propose new IR models specifically designed for hypermedia retrieval. All aim at obtaining a flexible conceptual data model based on so-called “common concepts” for hypermedia retrieval. The concepts shall be general enough to capture the needs for any digitally available data in hypermedia documents. The concepts must support as well as possible users in their information-seeking activities.

What are reasonable common concepts for hypermedia retrieval? Taking an integrated IR and database view, [13] proposes the following three concepts:

“Content” of documents refers to the knowledge of documents. For example, the knowledge “giulia sister_of francesco” is part of the content of a document.

“Facts” about objects refer to the knowledge about objects. For example, the knowledge “peter is author of document d1” is knowledge about the objects “peter” and “d1”. This knowledge forms the content of a database.

“Structure” of documents refers to the logical and semantic structure of documents. For example, a document consists of sections and a document bears hyper-links to other documents.

In [12], a model that encapsulates the above three concepts is developed. The model is based on a “probabilistic object-oriented logical” (POOL) formalism which allows a uniform view on objects with respect to their hypermedia environment: documents, images, authors, dates, etc. are objects and POOL *programs* model the content of objects (documents), the facts about objects (e.g., authorship), and the structure of objects (documents).

To investigate the applicability and usability of the model, a prototypical system has been developed. The system architecture comprises two major parts: an inference engine (a deductive database) and a graphical user interface (GUI). This paper presents the GUI. Particular attention was paid to the explicit representation of the three concepts “content”, “facts” and “structure” in a novel GUI that formally supports content-based querying and fact-based querying for hypermedia retrieval.

The outline of this paper is as follows. Section 2 presents the requirements regarding querying hypermedia documents based on the three concepts “content”, “facts” and “structure”. Section 3 introduces the POOL formalism.

Section 4 describes the GUI which supports the three concepts for querying hypermedia documents. The last section, section 5, depicts the application of the system on a collection of hypermedia documents.

2 Querying hypermedia documents

A conceptual model for hypermedia retrieval has to capture content-based querying, fact-based querying, and the structure of documents.

Content-based querying refers to the *content* of documents. For example, a query about all videos containing a sailing boat is a content-based query. *Fact-based* querying refers to the *facts* about documents. For example, a query on all paintings of a certain artist is a fact-based query. Content can be viewed as *knowledge of objects*, whereas facts can be viewed as *knowledge about objects*.

With hypermedia data, a special form of content-based querying requires attention: queries that refer to *relationships* between objects. For example, a query for all documents with sailing harbours close to Crete refers to a geographical relationship between objects. Also hyper-links from and to documents constitute relationships between objects.

A special type of relationships is the *logical structure* of hypermedia documents. For instance, a hypermedia repository may consist of documents such as a journal, each structured into several articles. Hypermedia retrieval requires not only access to the journal but also to its articles. For example, a content-based query leads to an article and the article is associated with an overview of the journal. The user can browse a retrieval result by reaching from found articles the journal outline. From there, articles of the same issue become accessible.

Relationships described within a document are part of the document content, whereas relationships between documents (or objects) are represented as facts about the documents (or objects). Since structure plays a key role for hypermedia, the structural relationship is modelled explicitly in conjunction to content and fact.

3 POOL: Probabilistic Object-Oriented Logic

POOL is a probabilistic object-oriented logic, based on [3], which enables to integrate content-based and fact-based querying necessary for hypermedia retrieval. POOL programs combine the object-oriented modelling concepts like *aggregation*, *classification*, and *attributes* known from object-oriented databases with the classical IR data model *set (or bag) of terms*, with a *probabilistic* representation of *uncertainty* inherent to IR. A fact in a POOL program is knowledge of an object. A fact can be a classification, an attribute value assignment, or a term.

Consider the example of a POOL program in figure 1 (texts following % are comments). The first clause classifies object d1 to be an instance of the class document. The second clause means that object peter is the value of attribute author of object d1. The third clause opens the context of object d1: the structure of d1 is reflected. Object d1 contains a further object: s1. Object s1 is called a subcontext of d1 and object d1 is called the supercontext of s1. The content of s1 is given by a logical program which contains three facts: the classification “object peter is a sailor” is true, the attribute value assignment “mary is a friend of peter” is true, and the term “sailing” is true. We refer to the objects d1 and s1 as *contexts* since a logical program is defined locally in the *context of these objects*. The POOL program reflects the logical structure *and* the content of the contexts d1 and s1.

```

document(d1)  % CLASSIFICATION
              % object d1 is a member of class document

d1.author(peter)  % ATTRIBUTE VALUE ASSIGNMENT
                 % object peter is an author of object d1

d1[ % AGGREGATION
   % d1 is a subcontext of the global context
s1[ % s1 is a subcontext of supercontext d1
   sailor(peter)
   % peter is a member of class sailor
   peter.friend(mary)
   % mary is a friend of peter
   sailing % TERM
 ] % end of context s1
] % end of context d1

```

Fig. 1. A POOL program

An information need such as “everything about sailing” now can be described as querying for all objects (contexts) where a logical formula like “sailing” is true. For example: The query “?- D[sailing]” searches for all contexts D where the formula “sailing” is true. The framework allows for combining content and factual retrieval, for example, the query “?- D[sailing] & D.author(peter)” combines a typical IR criterion referring to the content (all objects about sailing) with a typical database selection referring to the attribute values (all objects with author peter).

A major focus in the development POOL is to formally and uniformly capture the logical structure of hypermedia documents. Consider document d that consists of two sections s1 and s2:

```

d[ s1[sailing boats]
   s2[ocean boats] ]

```

The query “?- D[sailing]” retrieves both s1 and d. Context s1 is retrieved, since sailing occurs in s1, whereas context d is retrieved, since s1 is retrieved and it is part of d. Consider the following query “?- D[ocean & sailing]”. The conjunction “ocean & sailing” is true in the context d(s1,s2), i.e., the context that consists of both subcontexts s1 and s2. The term “ocean” is true in d, since it is true in s2, “sailing” is true in d(s1,s2), since it is true in s1. No leaf context on its own satisfies the query; only the context d(s1,s2) satisfies the query. We call d(s1,s2) an *augmented* context since its knowledge is augmented by the knowledge of the subcontexts. An augmented context *accesses* its subcontexts.

To represent the content of documents, POOL programs provide the possibility of specifying four truth values: true, false, inconsistent, and unknown. The truth value unknown allows for an independent assignment of true and false propositions (thus forming facts), i.e., instead of using per default a closed-world assumption (CWA) which would assume a term to be false in a context if it is not assigned, we can use unknown as additional truth value to true and false. This open-world assumption (OWA) is more reasonable in IR, since the assignment of terms is by nature incomplete and it would not be appropriate to assume false for all terms not assigned explicitly. The truth value inconsistent allows for combining the knowledge of subcontexts.

A major topic in IR is the incorporation of the intrinsic uncertainty of knowledge. POOL programs address two dimensions of uncertainty: (1) the uncertainty of the content representation and (2) the uncertainty that a supercontext accesses its subcontexts. For the uncertain content representation, probabilities for the four truth values true, false, inconsistent, and unknown can be defined (see [12] for the formal definition).

The uncertain access reflects the effect of a subcontext on the knowledge of an augmented context. A weight can precede the opening of a subcontext. Consider the following example:

```
d1[ 0.5 s1[ 0.8 sailing ]
    0.5 s2[ 0.6 sailing ] ]
```

In context s1, sailing is true with a probability of 0.8. In s2, sailing is true with 0.6. These probability values reflect the uncertain indexing of leaf contexts. The subcontexts s1 and s2 are accessed with a probability of 0.5. This probability reflects the effect of the knowledge of the subcontexts on the augmented knowledge. The query “?- D[sailing]” retrieves three contexts:

```
0.8 s1
0.58 d1(s1,s2)
0.6 s2
```

The subcontexts are retrieved with the probabilities of sailing. The augmented context d1(s1,s2) is retrieved with a probability of 0.58 which is the summation of three probabilities: the probability that sailing is true if both

subcontexts are accessed plus the probability that sailing is true if only s1 is accessed plus the probability that sailing is true if only s2 is accessed (see [12] for the semantics of the probability computation).

4 A Graphical User Interface for POOL

The query language of the POOL formalism is based on a logical calculus. Terms, classes, and attributes are predicate symbols of the language. A context (object) name is a modal operator (see [12] for the formal definition of the language). In this section, we present a GUI that supports the formulation of queries. The emphasis of the prototypical GUI lies on making the concepts for content- and fact-based querying explicit. We are aware that with respect to human computer issues, this prototypical implementation lacks usability studies [11]. In this first and novel attempt, our aim is to present conceptually a graphical query formulation for POOL, as QBE is one for SQL [16].

4.1 Formulating a Query

For a graphical query formulation for hypermedia retrieval, we identify four aspects: the selection of collections, the description of the content of documents, the description of facts about objects, and the specification of retrieval results.

Selection of collections: Before the actual query processing, the user selects the collections (also referred to as databases) that will be accessed for the query processing.

Description of the content of documents: The content of documents is described by a set of propositions. We distinguish three types of propositions: terms, classifications, and attributes.

Description of facts about objects: The facts about objects is described by classification and attributes.

Specification of the retrieval result: The retrieval result consists of a set of objects. The specification of the result selects which attribute values shall be displayed in the result window. The user can identify the retrieved objects and activate functions attached to the retrieved objects.

Figure 2 shows the entry window of the graphical user interface. We call this window the “query construction window”.

The query construction window is the front-end of a client that is connected to a server. The client manages the formulation of the query, the server receives the query, executes the query, and delivers the retrieval result to the client side. On the left hand side of the query construction window we find the four aspects mentioned above. Selecting the first aspect (selection of the databases) brings up on the right hand side a frame displaying all the available document collections (databases). In Figure 2, a number of

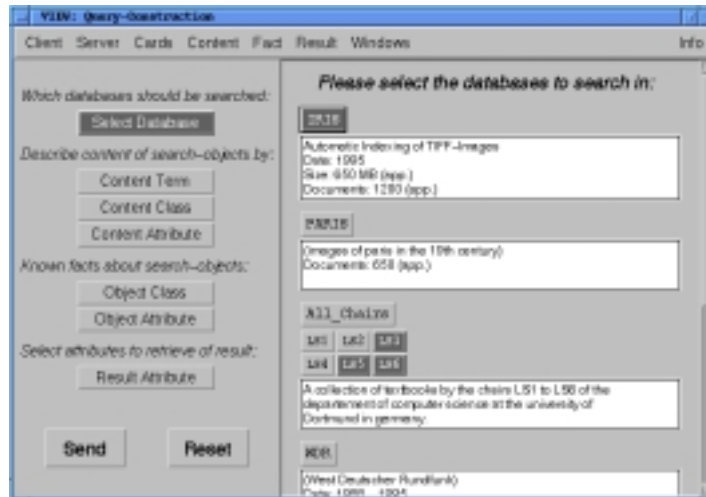


Fig. 2. The Query Construction Window

databases appear (their description is given in Section 5), among which the databases IRIS and three databases (LS3, LS5, LS6) of the university chairs at Dortmund have been selected.

The main menu row is composed of control functions for both the client (e.g., display a survey of the query) and the server (e.g., set query parameter). The next pull down menu “Cards” enables a selection of an input card (window to input data), in which support for editing the content, facts, and result cards is provided (respectively, “Content”, “Fact” and “Result” pull down menus). Available predicates (terms, classes, and attributes) are offered and can be selected. Figure 3 shows an example of the query formulation for the description of the content of a document by “terms”. Relevant documents shall be those described by the terms “forestscene” or “clouds and forest”. Result cards can be used for specifying the attributes of interest. For example, if present/valid for a retrieved object, then the video URL and the small image URL shall be displayed; thus the video and icon (small image) of the retrieved object can be accessed. The last menu entry “windows” is used for working with different queries and retrieval results. The system also offers a history and reformulation of queries¹.

The result of a query is a list of objects (contexts). An example is given in Figure 4. For each object, the attribute values that are specified in the result attribute card are displayed. Each retrieved context reflects an entry point into a structured document (the URLs shown in Figure 4).

¹ The system allows relevance feedback. For instance, in Figure 4, a user has specified the first image as relevant (POS), and the third one as not relevant (NEG). This information is used by the system to reformulate the query [15].

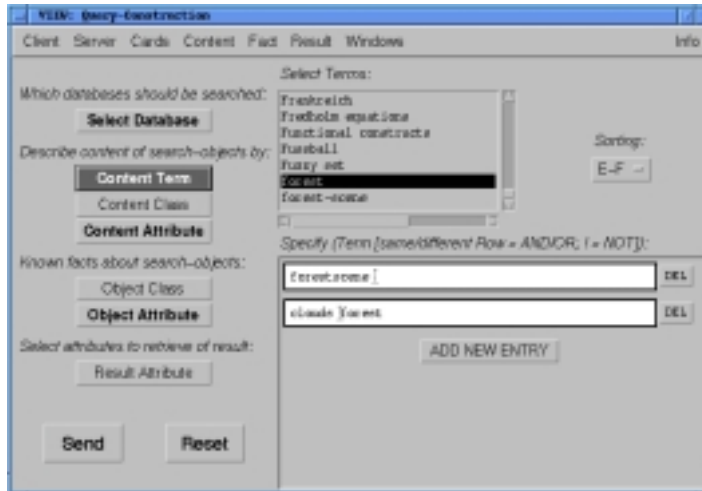


Fig. 3. Content-based query with terms

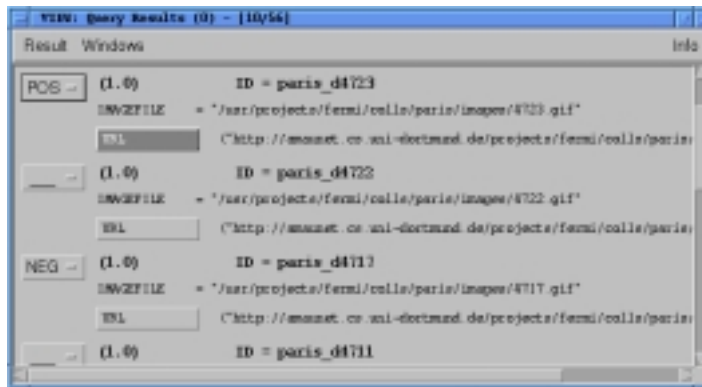


Fig. 4. Query result

4.2 Querying with Relationships

In image retrieval, particular attention must be paid to spatial relationships. To formulate a query on spatial relationships, predicate logic is used to represent spatial relationships. Consider the following formulation of a query:

$$?- D[X.isa(femme) \& Y.isa(homme) \& X.right_of(Y)]$$

The formal notation expresses that the user is seeking all documents D in which an object X exists that is a “femme” (French for woman), and in which an object Y exists that is an “homme” (French for man), and in which X is right of Y . The input card *content attribute* in Figure 5 corresponds to

the graphical formulation of the above query. Formulating the above query expression directly is a task for expert users. The input card of the GUI supports “semi-expert” users in formulating the query by displaying the possible attributes (all possible attributes are listed in the window above the logical formulation) and by automatically adding a corresponding entry into the specification of the query. Sorting functions support the browsing of the attribute list. It is also possible to have disjunction and conjunction of “object - attribute - value”.

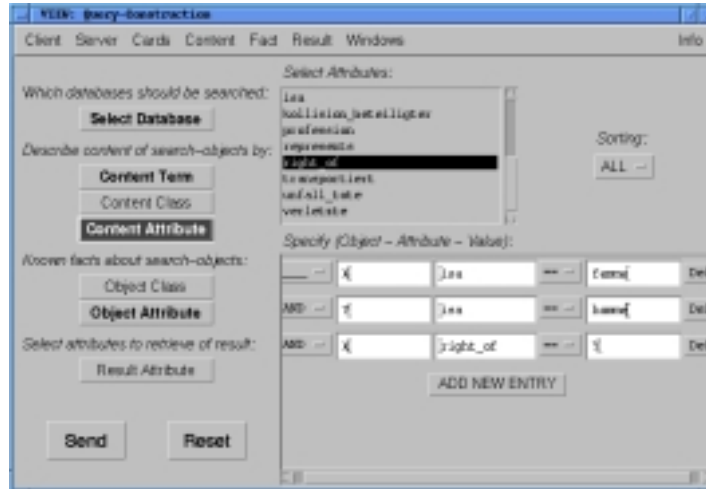


Fig. 5. Content-based query with relationships

4.3 Factual Queries and Vague Predicates

A factual query searches for all documents that belong to a specific class or that have specific attribute values. Consider the following example:

?- document(D) & D.year(X) & X.lt(1994)

The query searches for all documents published from year 1994 onwards.

It is important to allow for vague attribute values for fact-based query. A typical example is the specification of author names. Often, it is necessary to consider different spellings of a name, since users may not be familiar with the spelling of some names (e.g., names for other countries than that of the user). For example, “maier”, “mayer”, “meier” are three possible spelling of the same German name. Assume a predicate “soundex_code” that associates a code with a name (m600 is the soundex_code for the above three names). We can define a predicate soundex as follows:

```
X.soundex(Y) :- X.soundex_code(Code) & Y.soundex_code(Code)
```

The pairs (X,Y) are objects that are associated with the same soundex code. We call “soundex_code” a vague predicate since it selects a group of objects for one attribute value. We use the predicate soundex for formulating vague queries:

```
?- D.author(X) & X.soundex(maier)
```

The query retrieves all documents of authors that have a name sounding like maier.

Vague predicates are predicates that select a set of objects. This selection is an uncertain process (see [12] for how this uncertainty is represented). Figure 6 shows an example of a fact-based query and a vague predicate. Retrieved documents should be written from 1994 onwards by authors whose name sounds like “perlis”.

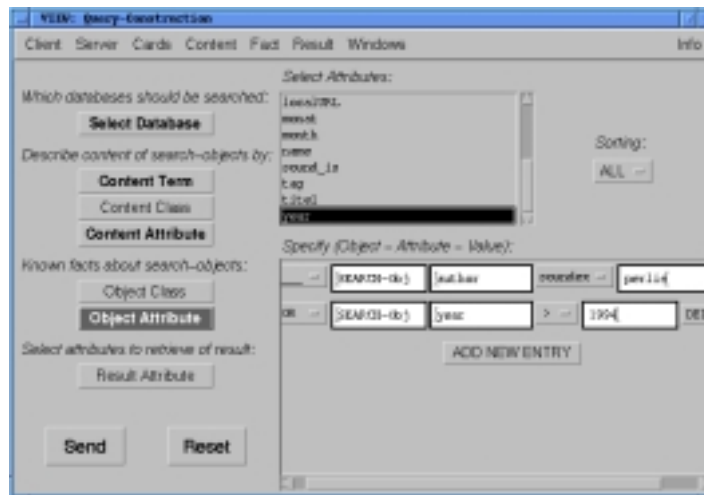


Fig. 6. Fact query and vague predicate

Factual queries and vague predicates play a crucial role in hypermedia retrieval. Non-text objects such as images and video possess attribute values (also called features) such as colour, contour, size, and length, upon which most “content-based” image retrieval approaches are based (e.g., the QBIC system [4]). We can deduce or extend the content description of images from their attribute values. With vague predicates, we can represent similar objects (e.g., $p1.colour(p2)$ means that two pictures are similar in colour distribution).

5 The application of the System on a Heterogeneous Collection

The implementation of the system is based on Java™ applets interacting over the web with the retrieval engine (see [14] for the complete description of the implementation). The architecture is designed to allow for the distributed processing of the query formulation and its execution. When the user submits the query, the query client sends the query to the query server. The query server connects a retrieval engine that is based on POOL. The POOL layer is translated into probabilistic Datalog (see [6]). Probabilistic Datalog programs are executed by HySpirit, an uncertain inference engine². The engine is connected to several databases in which document representations are maintained. The design supports the management of large data sets. The data is maintained by database management systems. Access paths provided by the databases assure fast and flexible access to the indexing data. The current data set comprises about 300 MB of indexing data which leads to 1.5 GB data maintained in the databases.

To demonstrate our system, we built a multi-collection consisting of text, images, and videos. This demonstration shows the uniform manipulation of the representations of documents that vary in their media type and available indexing information. The prototype demonstration system works with the following collections:

IRIS: A collection of 1200 colour pictures in TIFF format. This collection has been built for developing automatic semantical indexing methods (see [2]). The indexing algorithm is trained for landscape pictures and the collection contains typical landscape as well as other pictures.

PARIS: A collection of 650 black and white pictures in GIF format. The purpose of this collection is the evaluation of retrieval models that enable a richer query formulation than just keywords. The images of the collection show motives of Paris in the early years of the 20th century. In particular, the logical structure of the images and relationships between objects are represented in the document index, which has been created manually.

WDR: A collection of 200 Video news clips of the German TV broadcast station WDR. This collection has been set up for working with a rich semantical index where the events and persons occurring in the news are identified. Events and persons are classified and attribute values are specified.

We have translated the indexing information into our probabilistic object-oriented framework for representing documents. Figure 7 shows some documents of the retrieval result for the following query:

² <http://www.hyspirit.de>

```

retrieve(D) :- D[hiver]
retrieve(D) :- D[prinz]
retrieve(D) :- D[snow]
retrieve(D) :- D[X.right_of(Y) & femme(X) & homme(Y)]

```

We search for all documents that are about “hiver” (French for winter), or are about a prince (in German!), or about snow (in English), or show a woman right of a man (partly French!).



Fig. 7. The Multimedia Result

The window in the upper left corner summarises the query formulation of the selected databases, the content description, the factual description, and global query parameters such as open-world or closed-world assumption. In the query formulation, the context name “Search-Obj” reflects the objects searched for. Within the context of the object, each line contains a conjunction that has to be true in the retrieved objects. The survey is updated with any input added to the query formulation. Thus, the user is provided with an actual overview of the whole formulation at any time. In the lower left corner, a video with Prince Charles is played. We see winter pictures, pictures of snow, and pictures with women right of men. The centre of the window shows the query construction window.

6 Conclusion

We have presented a GUI for POOL, a probabilistic object-oriented logical model for hypermedia retrieval. POOL is designed to support the representation and retrieval of hypermedia objects. Content, facts, and structure, key concepts for hypermedia objects, can be formally modelled by POOL programs. The three concepts are explicit in the presented GUI, thus allowing integrated content-based querying and fact-based querying with graphical input support.

We demonstrated the GUI with a multi-collection composed of texts, images and video clips. The demonstration with the prototypical implementation has been the first step towards a GUI with general as well as specific concepts for hypermedia retrieval.

Future work will address human computer issues regarding the usability of the GUI.

References

1. Y. Chiamarella. Browsing and querying: Two complementary approaches for multimedia information retrieval. In *Hypertext — Information Retrieval — Multimedia (HIM)*, pages 9–26. 1997.
2. Y. Chiamarella and M. Mechkour. Indexing an image test collection. Technical report, FERMI Deliverable 10, ESPRIT BRA 8134, 1997.
3. R Fagin, J.Y. Halpern, Y. Moses, and M.Y. Vardi. *Reasoning about Knowledge*. MIT Press, 1995.
4. M. Flickner, 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 and Video Content: The QBIC System. In *Intelligent Multimedia Information Retrieval*, pages 23–31. 1997.
5. P. Fraternali and P. Paolini. A conceptual model and a tool environment for developing more scalable, dynamic, and customizable web applications. In *Proceedings of EDBT*, pages 421–435, 1998.
6. N. Fuhr. Probabilistic datalog - a logic for powerful retrieval methods. In *Proceedings of SIGIR*, pages 282–290, 1995.
7. D.M. Levy and C.C. Marshall. Going digital: A look at assumptions underlying digital libraries. *Communications of the ACM*, 38(4):77–84, 1995.
8. E. Megalou, T. Hadzilacos, and N. Mamoulis. Conceptual title abstractions: Modeling and querying very large interactive multimedia repositories. In *Multimedia Modeling — Towards the Information Superhighway*, pages 323–338, 1996.
9. C. Meghini, F. Sebastiani, U. Straccia, and C. Thanos. A model of information retrieval based on a terminological logic. In *Proceedings of SIGIR*, pages 298–308, 1993.
10. C. Meghini and U. Straccia. A relevance terminological logic for information retrieval. In *Proceedings of SIGIR*, pages 197–205, 1996.
11. W. Newman and M. Lamming. *Interactive System Design*. Addison-Wesley, 1995.

12. T. Rölleke. *POOL: Probabilistic Object-Oriented Logical Representation and Retrieval of Complex Objects - A Model for Hypermedia Retrieval*. Shaker Verlag, 1999. Phd Thesis.
13. T Rölleke and N. Fuhr. Querying for facts and content in hypermedia documents. In *Proceedings of FQAS*, pages 294–302, 1998.
14. F. Turra. Interaktive logische anfrageformulierung für hypermedia-retrieval. Diploma thesis, Universität Dortmund, 1998.
15. C. J. van Rijsbergen. *Information Retrieval*. Butterworths, London, 2. edition, 1979.
16. M. Zloof. Query by Example. *AFIPS*, 44, 1975.