RABBIT: An Interface for Database Access

Michael D. Williams
Frederich N. Tou

Cognitive and Instructional Sciences Group
XEROX Palo Alto Research Center
and
XEROX Office Systems Division
Palo Alto, California

Abstract

A new kind of user interface for information retrieval has been designed and implemented to aid users in formulating a query. The system, called RABBIT, relies upon a new paradigm for *retrieval by reformulation*, based on a psychological theory of human remembering. The paradigm actually evolved from an explicit attempt to design a 'natural' interface which imitated human retrieval processes.

To make a query in RABBIT, the user interactively refines partial descriptions of his target item(s) by criticizing successive example (and counterexample) instances that satisfy the current partial description. Instances from the database are presented to the user from a perspective *inferred* from the user's query description and the structure of the knowledge base. Among other things, this constructed perspective reminds users of likely terms to use in their descriptions, enhances their understanding of the meaning of given terms, and prevents them from creating certain classes of semantically improper query descriptions. RABBIT particularily facilitates users who approach a database with only a vague idea of what it is that they want and who thus, need to be guided in the (re)formulation of their queries. RABBIT is also of substantial value to casual users who have limited knowledge of a given database or who must deal with a multitude of databases.

1. Introduction

Traditionally, database interfaces have been designed from the inside out. That is, designers have implemented formal retrieval schemes with an eye to efficient search and retrieval and then tried to give users more or less direct access to the system implemented. This tactic insures that one has an effective underlying retrieval system, but it often leads to obscure and difficult to use user interfaces. We have begun from the opposite direction. We are trying to build an optimal user interface first and using it to define the internal facilities needed.

We have taken a cognitive theory of human remembering together with some artificial intelligence ideas about knowledge representation and used it to design a new paradigm for database retrieval interfaces for casual users. The paradigm is called *retrieval by reformulation.* A small experimental system based on this new paradigm has been implemented in the Smalltalk programming language [Ingalls, 1978] using KloneTalk [Fikes, 1981] on the Xerox Dolphin and Dorado personal computers.

Part of the motivation for designing a new kind of database interface was the unsuitability of existing database interfaces for casual users. Some database interfaces (e.g., SQUARE [Boyce et al, 1975] and SQL [Chamberlin et al, 1976]) require many hours of instruction to learn; others have a syntax which users find difficult to use and understand (e.g., the boolean expressions of DIALOG [Lockheed, 1979]). Interfaces based on the relational data model [Codd, 1970] usually require the user to know in advance which tables and attributes he will be needing, while users of network databases (such as ZOG [Robertson et al, 1981]) frequently get lost during the course of their search.

To help solve these problems we looked for inspiration to theories of how people retrieve information from their own memory. We believe this approach is promising for two primary reasons: (1) To the extent that the interface between a person and his external memory is like the interface between the person and his internal memory, the external memory may be easier and more natural to use, and (2) to the extent that human memory systems embody a 'solution' to the problems of retrieval from large heterogeneous databases, they may provide useful insights about how to design similar artificial systems.

We began our design process by conjecturing an interface that permitted *descriptive retrieval.* The basic tenet of descriptive retrieval is that people retrieve information from (their own) memory by iteratively constructing partial descriptions of the desired target item [Bobrow and Norman, 1975; Norman and Bobrow, 1979; Williams and Hollan, 1981]. The problem was that our conjectured system appeared to give us little more than the traditional boolean expression schemes such as DIALOG. We simply replaced the technical term 'keyword' with the term 'descriptor'. This led us to a re-examination of the problems inherent in boolean expression interfaces.

Upon consideration we conjectured that there were three major sources of difficulty for casual users of interfaces based upon boolean expressions of keywords: (1) the user has incomplete knowledge about the *descriptive terms* needed to create a query (e.g., what car colors does the database know about? red, crimson, rose, mauve?), (2) the

user doesn't know what kinds of attributes of the item(s) he is seeking are recognized by the database (e.g., does the database even have an attribute for car color?), and (3) many users find the syntax of complex boolean expressions difficult to understand.

Yet, if people actually recall information by descriptive retrieval then they must face the same problems; they must have some trick to get by those problems. We found such a trick in *retrieval by instantiation.* Retrieval by instantiation postulates that the information retrieved at each iteration of the retrieval process is often in the form of an *instantiation*, i.e., an example item suggested (e.g., analogically or metaphorically) by the partial description [Williams, 1981]. The common consequence of such an instantiation is that one is 'reminded' of something similar to the original item [Schank, 1980; Kolodner, 1980; Bower, Turner, and Black, 1979]. We conjecture that this reminding serves to counter all three of the problems noted for boolean expression schemes. The instantiations provide a *template* for describing the target item, access to the descriptive terms, and can provide the basis for an incremental reconstruction of the target item that avoids much of the complexity inherent in highly structured descriptions.

2. Retrieval by Reformulation

The basic principle underlying RABBIT is a new paradigm, retrieval by reformulation, for information retrieval elaborated from the notion of retrieval by instantiation. The user makes a query by first constructing a *partial description* of the item(s) in the database for which he is searching. RABBIT then provides a description of an *example instance* from the database which matches the user's partial description. Since it is unlikely that the first instance will be exactly what the user is looking for the user can then select any of the attributes shown in the example and incorporate those descriptors, or variations of them, into his partial description, thus, *reformulating* his initial query. At any time the user can request a new example instance, one which matches the latest version of his (partial) description, and then use the descriptors of that new instance to refine his query description still further.

Figure 1 shows RABBIT in the midst of a retrieval interaction. The interface consists of four primary window panes. The 'Description' pane specifies an implicitly defined boolean expression that appears to the user as a partial description of the item(s) he is seeking. The 'Example' pane contains an example item that matches the partial description as of the last user initiated retrieval cycle. More precisely, it contains a description, called the *image,* of an instance presented from some well-defined *perspective* (e.g., "STAR 8011 computer" can be viewed from the perspectives of "a manufactured product," "a computer," "an electronic device," "a piece of office equipment," and "a piece of stock in a store."). The 'Matching Examples' pane lists instances which satisfy the partial description as of the last retrieval cycle. The 'Previous Description' pane contains the description used on the last retrieval cycle. That description determines the perspective for presentation of the example and the list of matching examples. Figure 1 also shows the 'Example' pane command pop-up menu.

The example instance mentioned above is a central element of the interface. It serves several purposes: it functions as a *template*, it permits *access* to additional descriptors, it provides *semantic resolution* of potentially ambiguous terms, and it frequently serves as a *counterexample.*

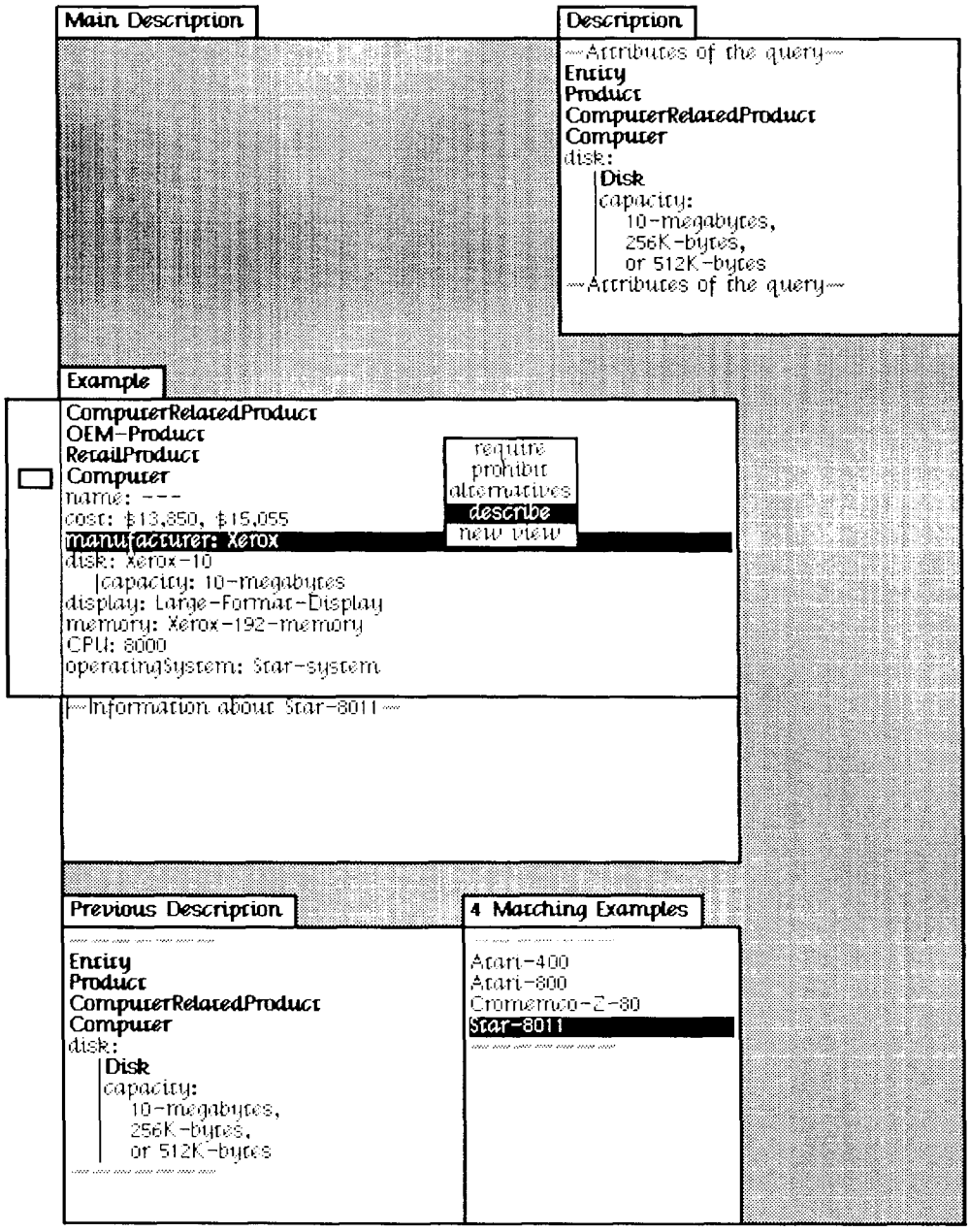
The example instance is a template in the sense that its presentation provides a pattern for making a query using the descriptors in the instance's image. It permits access to new descriptive terms through the alternatives and describe commands elaborated below.

It also provides semantic resolution in that the context of a term such as the role name 'manufacturer' establishes and refines the term's meaning. The role name 'manufacturer' could refer to a person or a nation or a corporation. The statement 'manufacturer: Xerox' in the context of a description of a computer product helps resolve a host of potential meanings.

The example instance is also a counterexample to the user's intentions whenever it is not exactly what the user is looking for. Rather than simply permitting the user to express his displeasure with the counterexample and having RABBIT try to guess what is wrong with it, the system tries to encourage the user to articulate what is wrong with the instance presented. The counterexample's simple presence reminds the user that his query description is incomplete or wrong and, in addition, points out the particular parts of his description that need correction or modification.

The current implementation of RABBIT supports a set of 5 basic operations for creating a query description given the descriptors provided in the image of the example instance. These operations are require and prohibit (which specify that the given descriptor is or is not to be a descriptor of the retrieved instance, respectively), alternatives (which presents the user with a pop-up menu of alternative descriptors to the given one), specialize (which shows the specializations of the given descriptor), and describe (which allows the user to examine a description of a given descriptor or to describe recursively what that descriptor should be). The describe command provides the user with the capability to build *embedded descriptions*, an example of which appears in figure 1 where the value of the attribute 'disk' is itself a description. [Tou, 1982] and [Tou, Williams, Fikes, Henderson, and Malone 1982a] contain a more complete discussion of the paradigm of retrieval by reformulation and the user interface to RABBIT. Each critiquing action the user takes immediately modifys the description being constructed. Thus, if the user were to require 'manufacturer: Xerox' that role value pair would be added to the description in the description window. If the user were to ask for alternatives and select several from the pop up menu (say Xerox, Apple, and Atari) then a descriptor 'manufacturer: Xerox, or Apple, or Atari' would be added to the growing description. If the user were to ask to describe 'manufacturer: Xerox' RABBIT would open up a new description window with the appropriate description, example, etc. panes except that the description would begin with the concept **'manufacturer'** and have as its initial example "Xerox" described as a manufacturer. The user would build his embedded description in the same manner the basic description was constructed and return to the original description having this new description embedded in the appropriate way.

This paradigm of retrieval by reformulation, in effect, defines a form of interaction by which RABBIT can assist casual users in formulating queries. Much of the intelligence of RABBIT comes from control of this interaction by appealing to the conceptual structure of the database.



3. Perspectives

The KL-ONE epistemology for representing knowledge [Brachman et al, 1979] has had a major influence on the development of RABBIT. One of the main uses of KL-ONE is the implementation of *perspectives.* A perspective is simply a way of describing an event or item from a particular viewpoint [Bobrow and Norman, 1975; Bobrow and Winograd, 1977: Goldstein and Bobrow, 1980; Goldstein, 1980]. In RABBIT, a perspective specifies which descriptors are included in the image of any instance presented to the user. RABBIT perspectives are dynamic in that the perspective from which the user views the instances in the database changes depending on the current partial description and on where he is within the database.

The principal function of perspectives is to limit the amount of information presented about any given instance. Consider the example of the description of a Star-8011 computer. The database could 'know' hundreds, even thousands, of attributes of this particular object. In addition to the information presented, the database could know about the production schedule of the machines, maintenance records, power consumption, physical size, salesmen selling these devices, locations of the devices,

serial numbers, etc. RABBIT deals with this problem by limiting the information presented to that which can be inferred to be relevant based on the query description that the user has given so far.

There are two distinct mechanisms RABBIT uses to construct a perspective. First it filters the attributes to be presented to a user by including only attributes implicitly acknowledged by the user. Since the partial description is a representation of the user's intent to the computer, that description is a legitimate basis for determining what information to include in the image of the example instance. In RABBIT the attributes included in the image are exactly those that belong to the instance classes occurring in the partial description. For example, if one were to see the computer descibed in figure 1 retrieved under the partial description **'Product'** (i.e. without the descriptor **'Computer'**) then only the attributes 'name', 'manufacturer', and 'cost' would be presented. Once the user refines the partial description to specify that he is seeking a computer, additional attributes (e.g., 'disk,' 'CPU,' ...) would appear.

A second mechanism for creating perspectives actually extends the perspective of any given instance beyond attributes directly held by the object. Note in figure 1 that because the user has created an embedded description about the disk of the computer sought, aspects of the disk that the user considers important (e.g., capacity) have been compressed into the image of the computer.

Perspectives serve three main functions in the RABBIT interface in addition to limiting the type and amount of information presented:

--facilitating the user's understanding of instances (i.e., supporting semantic resolution).
--enforcing certain kinds of semantic consistency.
--organizing and managing heterogeneous data.

4. A Practical Vision for RABBIT

RABBIT is not intended to provide a mechanism for rapid and efficient retrieval in the same sense that a computer scientist might see the problem. Rather, RABBIT is a specification for an interface that is easy to use and highly expressive from a human point of view. Presently, we have a vision of RABBIT with its associated KL-ONE conceptual structure residing in an interface machine (e.g., a Dolphin, Dandelion, or some other symbolic processor) connected to a database machine over a network. What RABBIT is calling instances should reside in the database machine. A search planner [such as that suggested by Kellogg, 1982] that takes a RABBIT query and converts it into something digestible (and possibly efficient) to the database machine might be a part of the interface machine. Because some KL-ONE descriptors may not have convenient representation in the query language of the remote machine, some local processing might be necessary to filter the instances returned.

5. Summary

This paper has briefly described the process of designing a novel type of database interface named RABBIT. RABBIT relies on a new paradigm for information retrieval, *retrieval by reformulation*, derived from a cognitive science theory of human remembering together with some artificial intelligence ideas about knowledge representation. The four main ideas underlying this paradigm are:

1) retrieval by constructed descriptions
2) interactive construction of queries
3) critique of example instances
4) dynamic perspectives.

The first three of these ideas had their origins in human psychology. The development of the fourth idea—dynamic perspectives—was motivated and influenced strongly by the KL-ONE knowledge representation language. A detailed characterization of RABBIT and the design tradeoffs we have struggled through can be found in [Tou, Williams, Fikes, Henderson, and Malone, 1982a].

Our experimental implementation of RABBIT looks very promising, but only usage by real users can determine the effectiveness and usefulness of the paradigm of retrieval by reformulation.

Acknowledgements

This paper is derived in large part from two related papers (Tou, Williams, Fikes, Henderson, and Malone, 1982b, and Williams, Tou, Fikes, Henderson, and Malone, 1982).

References

Bobrow, D.G., and Norman, D.A. "Some Principles of Memory Schemata," in D.G. Bobrow and A.M. Collins (Eds.), *Representation and Understanding: Studies in Cognitive Science.* New York: Academic Press, 1975.

Bobrow, D.G., and Winograd, T. "An Overview of KRL: A Knowledge Representation Language," *Cognitive Science,* **1**, pp. 3-46, 1977.

Bower, G.H., Black, J.B., and Turner, T.J. Scripts in Text Comprehension and Memory, *Cognitive Psychology,* Vol 1, 177-220. 1979.

Boyce, R.F., Chamberlin, D.D., King, W.F., and Hammer, M.M. "Specifying Queries as Relational Expressions: The SQUARE Data Sublanguage," *Communications of the ACM* **18**, 11 (Nov. 1975), pp. 621-628.

Brachman, R.J., Bobrow, R.J., Cohen, P.R., Klovstad, J.W., Webber, B.L., Woods, W.A. "Research in Natural Language Understanding: Annual Report, 1 September 1978 to 31 August 1979," *BBN Report No. 4274.* Cambridge, MA: Bolt Beranek and Newman Inc., August, 1979.

Chamberlin, D.D., Astrahan, M.M., Eswaran, K.P., Griffiths, P.P., Lorie, R.A., Mehl, J.W., Reisner, P., and Wade, B.W. "SEQUEL 2: A Unified Approach to Data Definition, Manipulation, and Control," *IBM Journal of Research and Development* **20** (Nov. 1976), pp. 560-575.

Codd, E.F. "A Relational Model of Data for Large Shared Data Bases," *Communications of the ACM* **13**, 6 (June 1970), pp. 377-397.

Fikes, R. "Highlights from KloneTalk: Display-Based Editing and Browsing, Decompositions, Qua Concepts, and Active Role-Value Maps," Proceedings of the 1981 KL-ONE Workshop, Jackson, New Hampshire, October, 1981.

Goldstein, I.P. "PIE: A network-based personal information environment." *Proceedings of the Office Semantics Workshop,* Chatham, Mass., June, 1980.

Goldstein, I.P., & Bobrow, D. Descriptions for a programming environment, *Proceedings of the First Annual National Conference on Artificial Intelligence,* Stanford, CA, August, 1980.

Ingalls, D.H. "The Smalltalk-76 Programming System: Design and Implementation," *Conference Record of the Fifth Annual ACM Symposium on Principles of Programming Languages,* Tucson, AZ: January 1978, pp. 9-16.

Kellogg, C. "Knowledge Management: A Practical Amalgam of Knowledge and Data Base Technology." *Proceedings of the National Conference on Artificial Intelligence.* Pittsburgh, PA: August 1982, pp 306-309.

Kolodner, J.L. Retrieval and Organization Strategies in Conceptual Memory: A Computer Model. Research Report #187, Department of Computer Science, Yale University, New Haven, CT. 1980.

Lockheed Information Systems. *Guide to DIALOG Searching,* Palo Alto, CA, 1979.

Norman, D.A., and Bobrow, D.G. "Descriptions: An Intermediate Stage in Memory Retrieval," *Cognitive Psychology* **11** (1979), pp. 107-123.

Robertson, G., McCracken, D., and Newell, A. "The ZOG Approach to Man-Machine Communication," *International Journal of Man-Machine Studies* (1981) **14**, pp. 461-488.

Schank, R.C. Failure-driven memory. *Cognition and Brain Theory,* Vol. 1, *4,* 41-60, 1980.

Tou, F.N. *RABBIT: An Interface for Information Retrieval by Reformulation,* unpublished S.M. thesis, Massachusetts Institute of Technology, Cambridge, Mass., May, 1982.

Tou, F.N., Williams, M.D., Fikes, R.E., Henderson, A., and Malone, T.W. RABBIT: an Intelligent Interface. Xerox Technical Report, forthcoming, 1982a.

Tou, F.N., Williams, M.D., Fikes, R.E., Henderson, A., and Malone, T.W. "RABBIT: an Intelligent Database Assistant." *Proceedings of the National Conference on Artificial Intelligence.* Pittsburgh, PA: August 1982b, pp. 314-318.

Williams, M.D., Tou, F.N., Fikes, R.E., Henderson, A., and Malone, T.W. "RABBIT: Cognitive Science in Interface Design" *Proceedings of the Cognitive Science Society.* Ann Arbor, MI: August 1982, pp. 82-85.

Williams, M.D. "Instantiation: A Data Base Interface for the Novice User," Xerox Palo Alto Research Center Working Paper, 1981.

Williams, M.D., and Hollan, J.D. "The Process of Retrieval from Very Long Term Memory," *Cognitive Science* **5** (1981), pp. 87-119.

Zloof, M.M. "Query by example," in *Proceedings of the National Computer Conference,* AFIPS Press, Arlington, Va., May 1975, pp. 431-437.