# A Database Interface for Mobile Computers

Rafael Alonso
Eben M. Haber
Henry F. Korth
Matsushita Information Technology Laboratory
182 Nassau Street
Princeton, NJ 08542-7072
{alonso,hfk}@mitl.com haber@cs.wisc.edu

**Abstract**

Computer-based personal information service is evolving beyond simple applications such as retrieval of phone numbers to include interaction with large, geographically distributed information bases. Concurrently, small, pen-based, mobile computers are becoming the machine of choice for personal computing. These two trends place conflicting demands on the design of database interfaces. The latter trend suggests simple interfaces that are easy-to-use, avoid keyboard use, and are suited for the small screens and small (relatively speaking) memory sizes of mobile machines. The former trend, however, suggests an increased sophistication in database interfaces, so as to provide access to the larger databases that are now part of a personal information service.

We describe a pen-based graphical database language that begins to combine these conflicting demands for simplicity and sophistication. We compare this language with previous work on graphical user interfaces designed for workstations. A prototyping effort has recently begun in our lab, and we provide a summary of its status.

## 1 Introduction

As computer users begin to shift more of their work to pen-based mobile computers (mobilestations), the demands placed on such machines grow. While the typical user's initial expectations are for inter-personal communication (e.g., electronic mail), simple computations (e.g., spreadsheets), and basic record-keeping (e.g., calendars), very soon they try to carry out on their mobilestations the same tasks which they are accustomed to perform on their workstations.

An important application for many computer users is accessing information stored in a database management system (DBMS). There are two important ways to access such information. These are querying the DBMS directly via a data manipulation language such as SQL, and accessing the data indirectly by using an application program with embedded database commands. Both of these interfaces are supported by most of the commercial DBMS's of which we are aware.

As applications are ported to mobilestations, users can continue to access DBMS information in the usual manner. However, the obvious approach for direct DBMS access which is using the handwriting recognition software of the mobilestations to translate pen strokes into, say, SQL commands has several problems. First, the state of the art in handwriting recognition is such that most users will quickly decide that the effort required to query a DBMS in this manner is excessive. Secondly, the size of the typical mobilestation is sufficiently smaller than that of a normal workstation (48 square inches vs. 165 square inches for some of our local computers), that one must devise new strategies for displaying information. Lastly, many users are interested in accessing data having a complex schema, and thus require a more powerful interface than is possible by supporting solely SQL commands.

While better algorithms and more CPU power will improve current character recognition capabilities, even if perfect recognition were available one could argue that most users would rather not compose SQL queries but would instead prefer a more graphically-oriented interface. (A preference for these types of interfaces also holds for normal workstations, as argued by much of the work referenced in the next section.) Even if one claims that mobile CPU's will get faster, their memory and disks larger, and communication more reliable, it is difficult to argue that portable displays will change radically in size. Thus, we feel that a useful DBMS interface for mobile environments must (1) be graphically-oriented (2) minimize pen-strokes, (3) be more powerful than a simple SQL front-end, and (4) make judicious use of the limited screen area.

The main objective of this paper is to present our research on new interfaces for mobile computers that meet the conditions listed above. Clearly, much more than a well-designed interface is needed to support mobile database access. For example, the increased likelihood of disconnection for mobile computers forces us to renew research on database partitions, and the lesser computational capabilities of such machines leads to a reconsideration of query optimization techniques. However, we will not discuss these or other issues in this paper, and instead concentrate on interface issues.

In the remainder of this paper we first present a short guide to the relevant database literature. We also provide a brief description of our interface design, and present some of the details of our current prototype implementation. We conclude with a few observations on our current work, and present our plans for future research in this area.

# 2 Related Work

Our approach is based on previous work in data modeling and in graphical user interfaces for databases. In this section, we provide a brief introduction to both research areas.

## 2.1 Graphical User Interfaces

There has been a substantial amount of previous research on graphical database interfaces for workstations. Recent work includes [AGS90, BH86, CERE90, KKS88, Kun92, SBMW91], and older work is surveyed in [Kim86]. Although space does not permit a survey of these interfaces, some generalizations will explain why an alternative approach is needed for mobile, pen-based computers. First, while workstations interfaces attempt to exploit the mouse to simplify user interaction with the database, they do not eliminate the need for a keyboard. Second, previous interfaces offer at most a very limited capability for users to navigate through a large database schema. The need for such a capability will become increasingly important as "world-wide" information bases come into common use. Third, many of the previous interfaces are aimed at the relational model, and thus lack the flexibility needed to deal with heterogeneous distributed databases that include a rich variety of data.

Despite these limitations, there are numerous concepts and features in existing workstation-based interfaces that are suitable for pen-based machines. These include graphical representation of schema, graphical representation of queries, and the ability to use the results of previous queries in the formulation of subsequent queries via "cut-and-paste."

## 2.2 Data Modeling

Also serving as a basis for our work is previous research in data models and data semantics. Both the small screen space of the typical mobile computer and the need for semantic simplicity indicate the use of a relatively high-level semantic data model. However, the need to access existing data, much of which is in relational or tabular form, requires simple incorporation of standard relational database concepts within the model.

The universal-relation model [KKF+84, Ull89] simplifies the relational model by providing a "relationless" view of a relational database schema. Some concepts from the universal-relation approach and related research on acyclic schemas (see, e.g., [Fag83]) can be used to simplify pen-based interfaces.

However, use of the universal-relation model alone does not meet our requirements. By presenting the database schema as a uni-level collection of relational attributes, the universal-relation model makes all structure within the database schema implicit. It is our view that structure within the database schema often helps rather than hinders the user in understanding the semantics of the database and in formulating queries. This is particularly true for hierarchically structured data as arises in document data, multimedia data, and even some table- or relation- based applications such as those using multiple spreadsheets. Furthermore, incorporation of structuring in our model permits certain parts of the database schema to be "abstracted away" or aggregated into a single entity type. This not only provides semantic simplification, but also allows for more compact representation of a database schema on the limited screen real estate of a mobile computer.

For this reason, work in nested relational databases (see, e.g., [Mak77, KR89]) as well as in object-oriented databases (surveyed in [Kim90]), is relevant to our research. Our goal is to combine concepts of structuring from these two domains, with the concepts of structure-elimination from the universal relation approach to design a multi-level semantic data model that includes all structure relevant to the user while eliminating most, if not all, low-level structuring that is not relevant to the user. Our ideas in this area are still preliminary and will be discussed in further detail in the full presentation.

# 3 Prototype

As we explained at the end of the previous section, we are using standard database concepts such as universal relations and nested relations in a new way. For example, we are making use of the universal relation idea to minimize the amount of actual "writing" that a user must undertake, and the nested relation concept to coalesce related information and display less material on the screen.

There are also some novel ideas in the current work to make the interface more powerful. We have designed our system so that it may connect to non-standard information sources such as the file system (this requires exploring seemingly strange ideas such as joining a relation with a file). We are also exploring new caching ideas [BA92] so that we may improve the performance of the interface.

There is still much research to be done on the above areas before we can create a truly useful interface. In the meantime, we have built a preliminary prototype to test some of our ideas and get a better understanding of the relevant problems.

Our system currently resides in a non-mobile machine (a DECstation 5000/200). Coding it has been relatively simple, both in terms of time (two and a half person-months) and effort (about 8000 lines of C++ code). Our work has been substantially speeded up by the use of the InterViews toolkit [LVC89] to generate X Window screens.

So far, the prototype has been connected only to a Sybase relational DBMS. This DBMS contains only a few relatively small synthetic databases, each with a dozen or fewer relations. The relations themselves contain a small number of tuples (less than a hundred in all cases). While such a database is too small to point out any major problems, so far we have not noted any obvious drawbacks to our approach. Not surprisingly, performance of the interface is acceptable with this amount of sample data. We expect soon to populate Sybase with a more "industrial-strength" sample database, and we will report on this experiment in the final version of this paper.

In Figure 1 we show a sample screen of the prototype. By selecting the *database* menu the user can connect to any DBMS which the system recognizes. As soon as a connection is made, the relations in the DBMS are displayed. Actually, only "top-level" relations are shown initially; as we already mentioned, we imposed a nested relation structure on our non-nested DBMS for the sake of conserving screen space. With further selections on the database menu users can select any number of relations, and then display all the possible join paths between them.

The auxiliary screen on the right in Figure 1 shows that there are many possible join paths between the two selected relations in this example (163). The user can now select the desired join. In practice one would limit the possible paths in a reasonable way; we currently plan to limit the user to the five or so shortest paths.

The resulting join is displayed in the auxiliary screen at the bottom of Figure 1. It shows the actual SQL query being sent to Sybase, and also the actual answer set for that query. The query can be manually modified before being executed, and the answer set can be scrolled in the obvious way.

# 4 Conclusions

We have argued that support for database access will be of increased importance for mobile computers, and have presented some of the problems that must be tackled in this area. Our interface design addresses these problems in a novel way, by using a mix of new approaches and

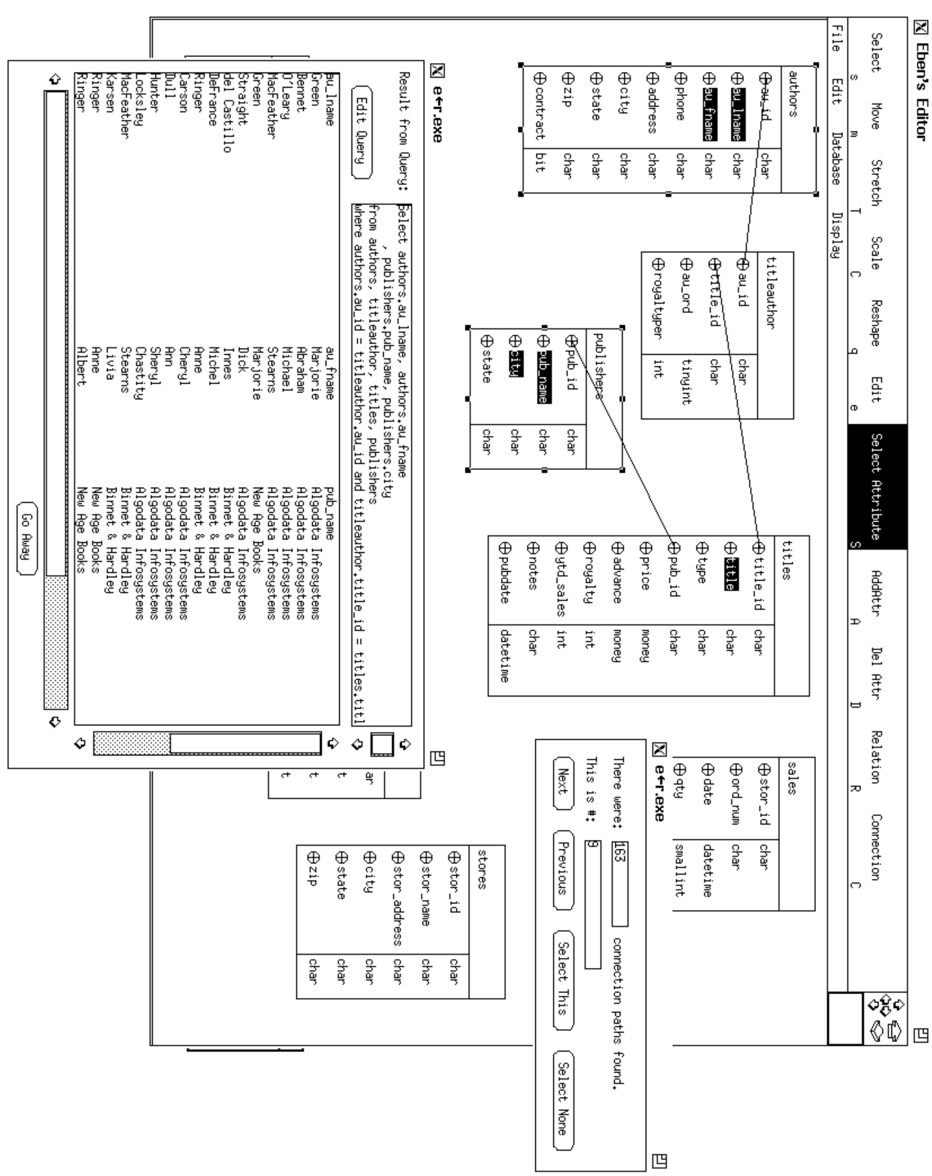


Figure 1: Sample Screen

old research recast in a different light. We have already developed a fairly powerful prototype interface to test our proposed ideas.

Our current plans include porting our code to our mobile environment to test fully the capabilities of our system. Our current mobile environment consist of PC-compatible 386 and 486 computers, running Mach 3.0; we also have a variety of pen-based machines. Many of these machines communicate via WaveLAN cards (NCR's wireless ethernet product).

We also expect soon to connect the interface to a variety of non-database storage systems, including a powerful mail management tool being developed at MITL, the Unix file system, and public information services such as Compuserve.

# References


[AGS90] R. Agrawal, N. H. Gehani, and J. Srinivasan. Odeview: The graphical interface to Ode. In *Proceedings of ACM-SIGMOD 1990 International Conference on Management of Data, Atlantic City, New Jersey*, pages 34–43, May 1990.

[BA92] Matt Blaze and Rafael Alonso. Dynamic hierarchical caching for large-scale distributed file systems. In *Proceedings of the 12th International Conference on Distributed Computing Systems*, 1992.

[BH86] D. Bryce and R. Hull. SNAP: a graphics-based schema manager. In *Proceedings of the Second International Conference on Data Engineering, Los Angeles*, page 151, February 1986.

[CERE90] B. D. Czejdo, R. A. Elmasri, M. Rusinkiewicz, and D. W. Embley. A graphical data manipulation language for an extended entity-relationship model. *IEEE Computer*, 23(3):26–36, March 1990.

[Fag83] R. Fagin. Acyclic database schemes (of various degrees): A painless introduction. Research report rj3800, IBM Almaden Research Center, San Jose, CA, February 1983.

[Kim86] H.-J. Kim. Graphical interfaces for database systems: A survey. In *Proceedings of the 1986 Mountain Regional ACM Conference, Santa Fe, NM*, 1986.

[Kim90] W. Kim. *Introduction to Object-Oriented Databases.* MIT Press, Cambridge, MA, 1990.

[KKF+84] H. F. Korth, G. M. Kuper, J. Feigenbaum, A. Van Gelder, and J. D. Ullman. System/U: A database system based on the universal relation assumption. *ACM Transactions on Database Systems*, 9(3):331–347, September 1984.

[KKS88] H.-J. Kim, H. F. Korth, and A. Silberschatz. PICASSO: a graphical query language. *Software Practice and Experience*, 18(3):169–203, March 1988.

[KR89] H. F. Korth and M. A. Roth. *Query Languages for Nested Relational Databases*, volume 361 of *Lecture Notes in Computer Science*, pages 190–206. Springer-Verlag, 1989.

[Kun92] M. Kuntz. The gist of GIUKU – graphical interactive intelligent utilities for knowledgeable users of data base systems. *ACM SIGMOD Record*, March 1992.

[LVC89] Mark A. Linton, John M. Vlissides, and Paul R. Calder. Composing user interfaces with interviews. *IEEE Computer*, pages 8–22, February 1989.

[Mak77] A. Makinouchi. A consideration on normal form of not-necessarily-normalized relation in the relational data model. In *Proceedings of the Third International Conference on Very Large Databases, Tokyo*, pages 447–453, 1977.

[SBMW91] G. H. Sockut, L. M. Burns, A. Malhotra, and K.-Y. Whang. GRAQULA: A graphical query language for entity-relationship or relational databases. Research report RC 16877, IBM T. J. Watson Research Center, Yorktown Heights, NY, March 1991.

[Ull89] J. D. Ullman. *Principles of Database and Knowledge-Base Systems*, volume 2. Computer Science Press, Rockville, MD, 1989.