Dynamic Queries for Information Exploration: An Implementation and Evaluation

Christopher Ahlberg, Christopher Williamson, and Ben Shneiderman*

Department of Computer Science
Human-Computer Interaction Laboratory
University of Maryland, College Park, MD 20742
ben@cs.umd.edu

Abstract

We designed, implemented and evaluated a new concept for direct manipulation of databases, called *dynamic queries,* that allows users to formulate queries with graphical widgets, such as sliders. By providing a graphical visualization of the database and search results, users can find trends and exceptions easily. Eighteen undergraduate chemistry students performed statistically significantly faster using a dynamic queries interface compared to two interfaces both providing form fill-in as input method, one with graphical visualization output and one with all-textual output. The interfaces were used to expore the periodic table of elements and search on their properties.

1. INTRODUCTION

Most database systems require the user to create and formulate a complex query, which presumes that the user is familiar with the logical structure of the database [4]. The queries on a database are usually expressed in high level query languages (such as SQL, QUEL). This works well for many applications, but it is not a fully satisfying way of finding data. For naïve users these systems are difficult to use and understand, and they require a long training period [3].

Clearly there is a need for easy to use, quick and powerful query methods for database retrieval. Direct manipulation has proved to be successful for other applications such as display editors, spreadsheets, computer aided design/manufacturing systems, computer games and graphical environments for operating systems such as the Apple Macintosh [8]. Direct manipulation interfaces support:

- Continuous visual representation of objects and actions of interest
- Physical actions or labelled button presses instead of complex syntax
- Rapid, incremental, reversible operations whose impact on the object of interest is immediately visible.
- Layered or spiral approach to learning that permits usage with minimal knowledge.

One of the great advantages of direct manipulation is that it places the task in the center of what users have to do. [7] describes it as "The user is able to apply intellect directly to the task; the tool itself seems to disappear". The success of direct manipulation can be understood in the context of the syntactic/semantic model which describes the different levels of understanding users have [8]. Objects of interest are displayed so that actions are directly in the high level semantic domain. Users do not need to decompose tasks into syntactically complex sequences. Thus each command is a comprehensible action in the problem domain whose effect is immediately visible. The closeness of the command action to the problem domain reduces user problem-solving load and stress.

For databases, there have been few attempts to use direct manipulation. Zloof describes a method of data manipulation based on the direct representations of the relations on the screen, Query-by-Example [10]. Zloof writes "a user dealing with 'simple' queries needs to study the system only to that point of complexity which is compatible with the level of sophistication required within the domain of those queries." Query-by-Example succeeds because novices can begin working with just a little training, yet there is ample power for the expert.

Another attempt to create a more user friendly query language is the PICASSO query language [3]. The authors state that the major contribution of PICASSO and graphical interface ROGUE is that users can pose complex queries using a mouse without knowing the details of the underlying database schema nor the details of first-order predicate calculus or algebra.
The power of direct manipulation can be applied even further. Neither Query-by-Example nor PICASSO provide any visual

---

** Current address: Dept of Comp. Sci., Chalmers Univ. S-412 96 Göteborg, Sweden*

display of actions. Query-by-Example relies on users entering values with a keyboard. Even though PICASSO supports input through mouse and menus, it requires users to perform a number of operations in each step. The combination of graphical input/output is not applied in either system.

A more desirable database interface:

- represents the query graphically,
- provides a visible limits on the query range,
- provides a graphical representation of the database and the query result,
- gives immediate feedback of the result after every query adjustment, and
- allows novice users to begin working with little training but still provides expert users with powerful features.

An interface utilizing dynamic queries possesses the above-mentioned properties [9].

In dynamic queries the query is represented by a number of widgets such as sliders [1] (figure 1). A slider consists of a label, a field indicating its current value, a slider bar with a drag box, and a value at each end of the slider bar indicating minimum and maximum values. Sliding the drag box with the mouse changes the slider value. Clicking on the slider bar increases or decreases the value one step at a time.

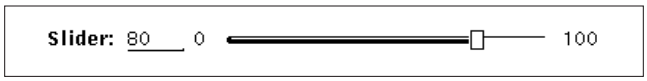

Figure 1. Slider from Open Look.

The database is represented on the screen in graphical form. This paper describes a program dealing with the chemical elements and accordingly the periodic table of elements was chosen as the representation. The result of the query can be highlighted by coloring, changing points of light, marking of regions, or blinking.

The combination of a graphical query and graphical output matches well the ideas of direct manipulation. The slider serves as a metaphor for the operation of entering a value for a field in the query - it provides a mental model [5] of the range. Changing the value is done by a physical action - sliding the drag box with a mouse - instead of entering the value by keyboard. By being able to slide the drag box back and forth and getting immediate updates of the query results, it is possible to do tens of queries in just a few seconds, i.e the operation is rapid. The operation is incremental and if the query result is not what users expected the operation is reversible by just sliding the drag box in the opposite direction. Error messages are not needed - there is no such thing as an 'illegal' operation.

The interaction between the database visualization and the query mechanism is important. The sliders have to be placed close to the visualization to reduce eye movement. The highlighting of elements should be in harmony with the coloring scheme of the slider. For example the color of the area to the left of the drag box on the slider bar is the same as the highlighted elements in the visualization, because the values to the left of the drag box are the values that satisfy the query.

The dynamic queries program used for the experiment is an educational program for the periodic table of elements. It allows users to set properties such as atomic number, atomic mass, electronegativity, etc. to highlight elements that satisfy the query displayed on the periodic table. This lets users explore how these properties interact with each other. Other interesting discoveries can be made regarding trends of properties in the periodic table - such as how electronegativity increases from the lower left corner to the upper right corner of the periodic table. Exceptions to trends can also be found easily, such as the two places in the periodic table where the atomic mass does not increase with atomic number.

2. EXPERIMENT

2.1 Introduction

This experiment compared three different interfaces for database query and visualization: a dynamic queries interface, a second interface (FG) providing graphical visualization output but using form fill-in as the input method [6] (Form fill-in - Graphical output) and a third interface (FT) also using a forms fill-in as input but providing output as a list of elements fulfilling the query (Form fill-in - Textual output). The alternative interfaces were chosen to find out which aspect of dynamic queries makes the major difference, the input by sliders allowing users to quickly browse through the database, or the output visualization providing an overview of the database. These were compared using three sets of matched questions.

2.2 Hypotheses

The primary hypothesis was that, because of the visualization of the periodic table in the dynamic queries and the FG interfaces, there would be a major difference compared to the FT interface. Performance results were measured as the time used for each question and the number of correct answers.
For questions asking subjects to find trends in the periodic table, the hypothesis was that the visualization of the periodic table in the dynamic queries and FG interfaces would make the major difference compared to the FT interface. But the ability to perform a large number of queries during a small period of time with the dynamic queries interface would make a difference favoring dynamic queries over FG.

2.3 Interfaces

All interfaces were built using the Developer's Guide user interface development package in the OpenWindows environment on a Sun Microsystems SparcStation 1+ workstation with a 17-inch color monitor and optical three button mouse.

2.3.1 Dynamic Queries interface

The dynamic queries interface (figure 2) provides a visualization of the query result. A periodic table showing the elements is displayed in 40-point Roman font. The elements that fulfill the criteria set by the user's latest query are highlighted by being displayed in red. The rest of the elements are displayed in light grey. Users perform queries by setting the values of six properties using sliders (figure 1). All interfaces included two other buttons, 'Max' and 'Min' that set the values of all input fields to the minimum or maximum value.

The query result is determined by ANDing all six sliders, so all the elements that have an atomic mass less than or equal to X AND an atomic number less than or equal to Y, etc. , fulfill the criteria. The area to the left of the slider drag box is painted in red, corresponding to the red color of the highlighted elements in the visualization and thereby providing feedback about how elements are selected. The sliders are positioned under the periodic table, close to the visualization to minimize the distance users have to move their eyes. One direct manipulation feature in the dynamic queries interface was left out for experimental purposes. It allows users to click on any element and thereby set the sliders to the values of the properties of that element.

2.3.2 FG interface

The FG interface (figure 3) provides users with the same visualization as the dynamic queries interface, but the query is composed by form fill-in. Instead of a slider, a numeric field allowing users to enter a value for that property by keyboard is provided. To the left of the numeric field the range of the criterion is given. If a value bigger than the upper bound is entered, the field is set to the upper bound.

The search is performed when users press the return key. The cursor indicating which numeric field is active stays in the same numeric field. Entering new values is done by either modifying the old one or deleting it and entering a new one. This is to provide an easy way to do the fine-tuning often needed when completing tasks. Users change the active field by using the up/down arrow keys. The left and right keys move the cursor inside the numeric field. The graphical output is exactly the same as in the dynamic queries interface.

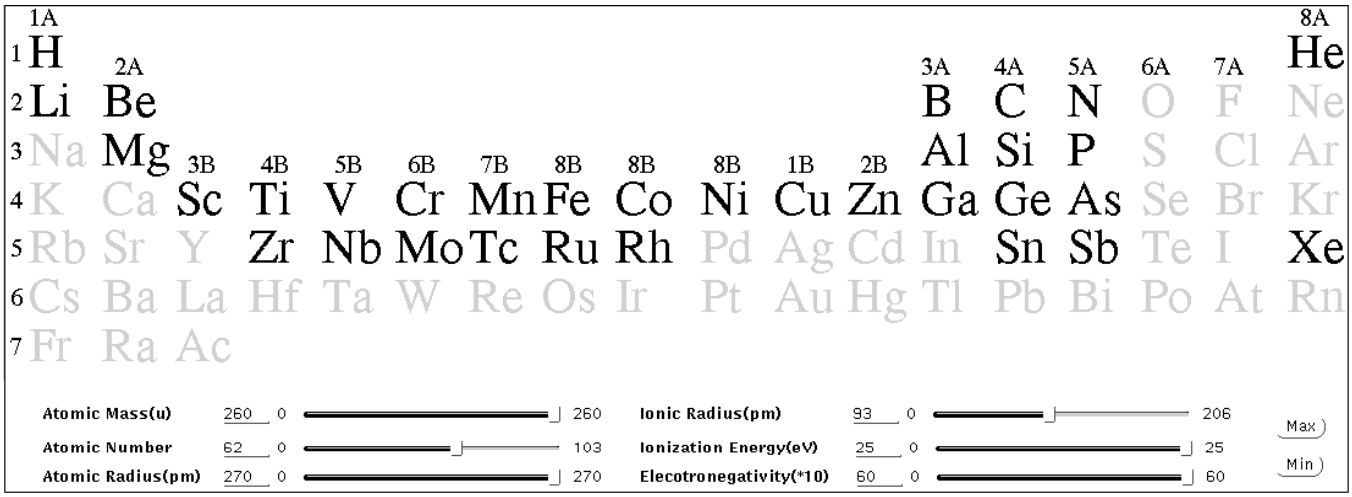

Figure 2. Dynamic Queries interface for the periodic table of elements

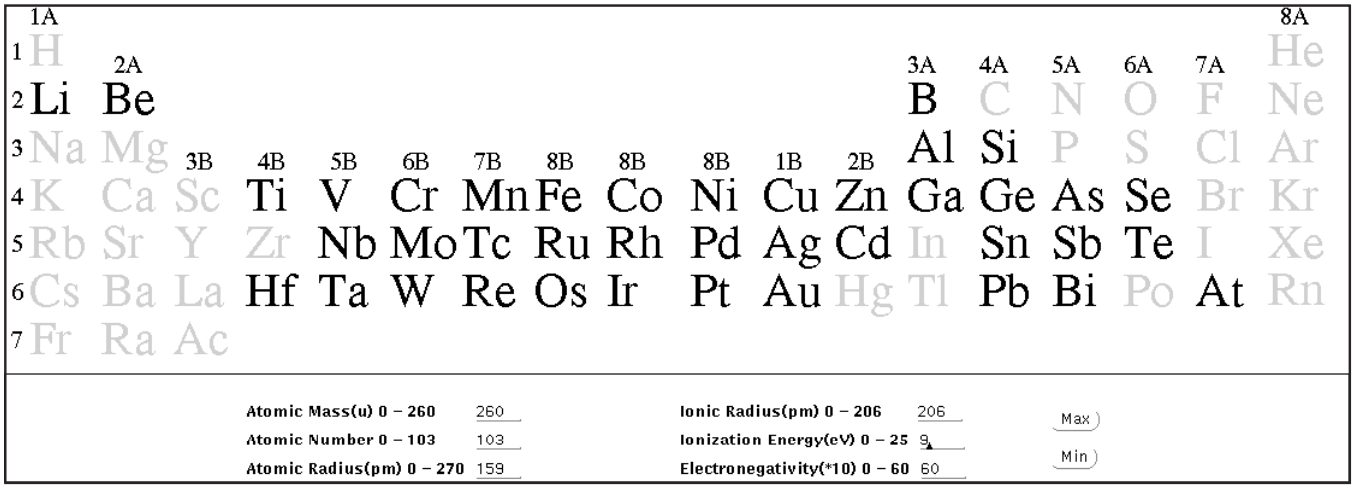

Figure 3. FG interface for periodic table

2.3.3 FT interface

The all textual interface (figure 4) provides exactly the same style of input as the FG interface but the output is given in an all textual manner. The elements that fulfill the criteria are listed in order of atomic number in a text window above the input fields. To be able to answer the questions, subjects were provided with a printed periodic table when using this interface.

2.4 Experimental variables

The independent variable in the experiment was the type of interface, with three treatments:

i. Dynamic Queries
ii. FG
iii FT

The dependent variables were:

i. Time to find answers
ii. Number of correct answers
iii. Subjective satisfaction

2.5 Tasks

Subjects were presented with a set of five matched questions for each interface. The questions, chosen in cooperation with a chemistry professor at University of Maryland, were divided into five categories:

1. Out of a certain set in the database, find a certain element fulfilling a simple criteria. This task required subjects to concentrate on a part of the database such as a group or period and find the element that, for example, had the highest ionization energy.

2. This more complex task required subjects to make at least two queries to complete the task; comparing the characteristics of one element to that of another.

3. Combine sliders/fields to get a subset of elements and find the element fulfilling a certain criteria in this set. This task required the set to examine to be formed by combining several criteria.

4. Find a trend for a property. The task requires subjects to create a mental picture of how a property changes through the database. This might be how atomic mass increases with atomic number.

5. Find an exception to a trend. This task asked subjects to find, from a given number of elements, the element that didn't follow 'normal behavior' .

2.6 Pilot Study Results

A pilot study of four subjects was conducted. It led to several changes in the experiment design. The initial manual timing procedure was changed to a computerized procedure. The instrument used for measuring subjective satisfaction was the Questionaire for User Interface Satisfaction (QUIS) [2], but shortened to 30 of the 72 original questions.

2.7 Participants

Eighteen undergraduate students, 9 females and 9 males, from summer session chemistry classes at University of Maryland participated voluntarily in the experiment. Only two participants had used the Sun SparcStation 1+ used as the platform for the experiment. All but three subjects had used a mouse before, generally Macintosh or some IBM PC mouse, but not the optical mouse that the Sun SparcStation 1+ uses. The subjects' chemistry education ranged from one to four undergraduate courses.

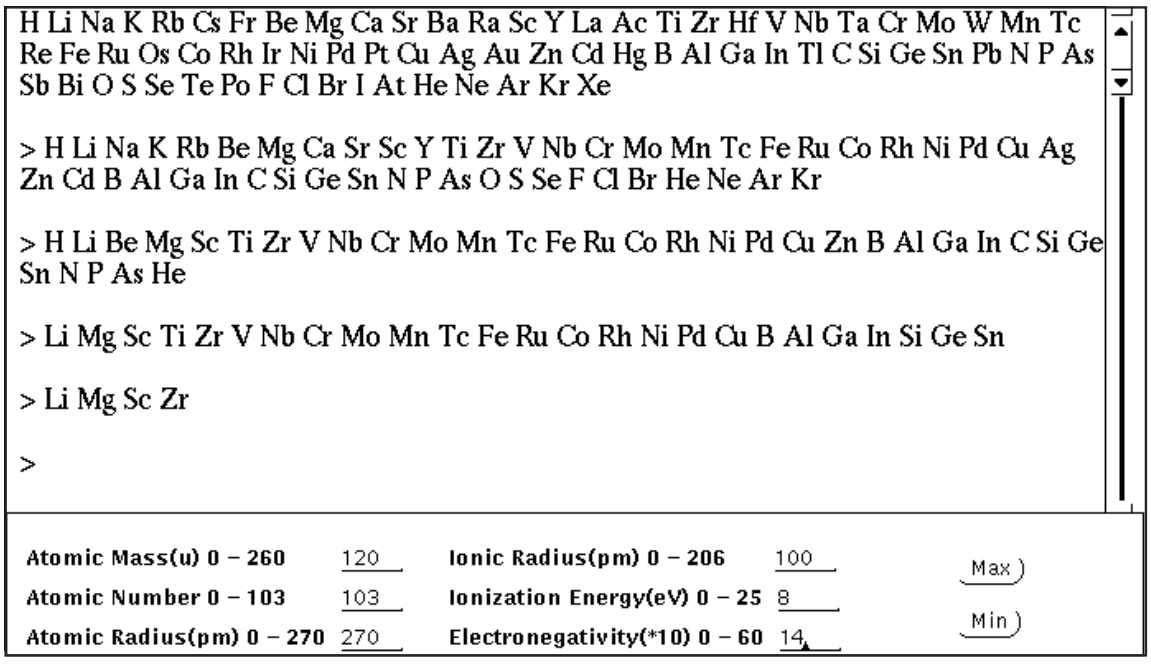

Figure 4. FT interface for periodic table

2.8 Procedures

A counterbalanced within-subjects design was used. The question sets were always given in the same order. Each session lasted an hour and consisted of four phases:

1. Introduction and training: Subjects were given a description of the purpose and procedures of the experiment and were also given training with the mouse and controls of the interfaces.

2. Practice tasks: Two practice tasks were given for each interface. During these tasks subjects were free to ask questions about both tasks and interface.

3. Timed tasks: For each interface five questions were given. Before answering each question the interface was set to the initial state. Subjects read the question, and were asked if they fully understood it. If so they pushed the Start button and started the query. This was to eliminate variations in subjects comprehension speed. When subjects found the answer they wrote it down and pushed the Done button.

4. Subjective evaluation: Subjects were asked to fill out a shortened QUIS-form after having completed each interface and to provide open commentary while answering questions.

Phases 2, 3 and 4 were repeated for each interface.

Administration

The experiment was run over a period of 12 days. Subjects were asked to work as quickly and accurately as possible. The experimenter sat next to the subject, presented questions and ensured that the subject initialized the query and followed the proper timing procedures.

3. RESULTS

Analysis of the timed tasks was done using an ANOVA with repeated measures for interface type. Observing the mean times to complete all tasks 1-5, shows a significant main effect, $F(2,34)=36.1$ ($p<0.001$). Similarly a significant main effect was found for individual tasks 1,2,4 and 5, $F(2,34)=19.0$, 16.4, 21.4, 20.2 respectively ($p<0.001$) and for task 3 $F(2,34)=7.1$ ($p<0.005$).

Tukey's post-hoc HSD analysis was used to determine which interface(s) was significantly faster. The dynamic queries interface had a significantly faster mean time for completing all tasks than both FG and FT interfaces, ($p<0.005$) and ($p<0.001$) respectively.

The time to complete each task is shown in Table 1. Figure 5 gives a bar chart of the same data.

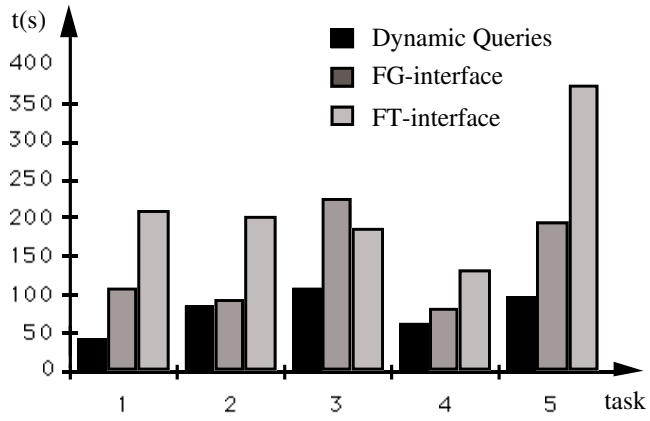

Figure 5. Mean time to complete each task

Timing Data For Each Task

	Dynamic Queries		FG		FT
1	40.6 (21.5)	← .05	108.8 (62.3)	← .001	210.2 (129.3)
2	87.3 (92.3)		91.5 (44.8)	← .001	200.8 (79.1)
3	111.0 (55.8)	← .005	225.2 (105.1)		187.8 (114.5)
4	60.4 (41.4)		81.4 (30.9)	← . 001	126.8 (32.0)
5	95.9 (51.4)	← .05	202.5 (101.6)	← .001	367.9 (180.1)
∑	412.0 (216.1)	← .005	709.5 (182.9)	← .001	1093.6 (336.3)

Table 1. Table showing mean time to complete each task. Variance is shown in parantheses. An arrow from one cell to another indicates significantly smaller time for the cell being pointed at. Significance level is given above arrow.

For task 1 the dynamic queries interface was significantly faster than the FG interface, ($p<0.05$) and the FG interface was significantly faster than the FT interface, ($p<0.001$). For task 2, no difference between dynamic queries and the FG interface was found, but both were significantly faster than the FT interface, ($p<0.001$).

For task 3 the dynamic queries interface was significantly faster than both FG and FT interfaces, ($p<0.005$) and ($p<0.05$) respectively, no significant difference between the FG and FT interfaces was found. Actually, the mean time for the FT interface was 37.4 seconds faster than the FG interface.

For task 4 both the dynamic queries and FG interfaces were faster than the FT interface, ($p<0.001$). Task 5 showed significantly faster mean time for the dynamic queries interface compared to the FG interface, ($p<0.05$) and the FG interface was significantly faster than the FT interface, ($p<0.001$).

Figure 6 shows the number of errors subjects made for each task and interface, out of a total of 18 questions.

For the QUIS, there was a statistically significant difference between the dynamic queries and FT interfaces for **all** questions. There was also a statistically significant difference between the FG and FT interfaces for all questions; but no significant differences between the dynamic queries and FG interfaces.

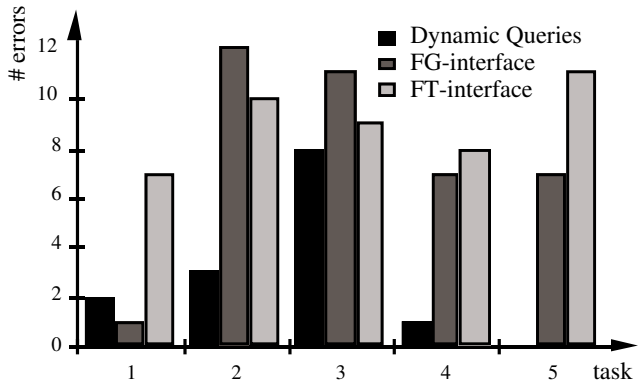

Figure 6. Table showing number of errors for each interface and task**.**

4. DISCUSSION

The hypothesis that the dynamic queries interface would perform better than both the FG interface and the FT interface was confirmed. Similarly the FG interface produced faster performance times than the FT interface. The major difference in mean performance times was between the dynamic queries and FG interfaces compared to the FT interface. This was also confirmed in participants' comments which indicated that the visualization is the most important part.

The lack of difference in performance between the dynamic queries and FG interfaces in task 2 and four was surprising. The results for task 2 can possibly be explained by the fact that it was similar to task 1, and therefore subjects learned how to apply a good strategy. For task 4 subjects already had an idea of what the answer should be from their coursework. The range of the properties was limited and not too many values had to be checked to get a picture of the trend, therefore the slider did not make a big difference.

4.1 Timed tasks

Task 1: The dynamic queries interface performed significantly better than both the FG and FT interfaces. The correct answer could be found by adjusting the correct slider until either the first or the last element in the subset changed color. Using the FG interface or the FT interface required subjects to use some kind of binary search method to find the correct element since each query had to be typed-in, which accounts for the slower performance. Using the FT interface required users to locate the subset of the periodic table in question in the larger set retrieved from the database with the query, which accounts for the longer performance time using that interface.

Task 2: Surprisingly no difference in performance time were found between the dynamic queries interface and the FG interface. This can probably be explained by the fact that the task was similar to task 1, and subjects figured out a good strategy while solving task 1. Similarly to task 1, the FT interface performed poorly as participants had to locate the relevant subset of elements to be analyzed in the larger set.

Task 3: The dynamic queries interface performed significantly better than both the FG interface and the FT interface. No significant difference between the FG interface and the FT interface was found but the mean time for the FT interface was actually shorter than the mean time for the FG interface. The task required subjects to set two input fields to find a subset of elements and in this subset find one element that fulfilled a criteria. As the subsets were rather big the visualization of the dynamic queries and FG interfaces caused some problems. To see one element shifting color when moving the slider or entering values was found to be hard. The dynamic queries interface compensated for this by making it possible to quickly change the value. The FT interface performed better than the FG interface as it was possible to see the result of the latest queries on the screen. By comparing the line length of the current and the previous result subjects could easily find the correct element. The FG interface posed an interesting problem for subjects that were novice computer users. Trying to find which element was the first to change from red to gray, required them to enter values repeatedly. In doing this, novices had to look down at the keyboard, press <return> and before they had moved their eyes to the screen, the change had already taken place.

Trying to see which element changed color, subjects were found leaning backwards to get an overview. This problem is probably a result of two factors, the colors used and the width of the window. The colors were found to be good in the QUIS results, ~8 on the 1-9 scale, but maybe some better combination can be found.

Task 4: This task required subjects to find an overall trend in the database. The hypothesis that the visualization would make the major difference was confirmed. Finding a trend is

greatly simplified by getting an overview of the database, which was reflected in the experiment results. But comparing the dynamic queries interface with the FG interface showed no difference which was not in line with our hypothesis. The reason for this is twofold, a lot of the students already had a general idea of the answer and only had to confirm it, and even if they did not know the answer they only had to type in a few values to find the solution using the FG interface.

Task 5: The dynamic queries interface performed significantly better than both the FG interface and the FT interface, this stemming from the two advantages of the dynamic queries interface, the visualization and the sliders. The visualization allowed subjects to see exceptions easily when they showed up on the screen and the sliders allowed subjects to quickly change the values to find the correct answer. This task was very hard to solve with the FT interface, as subjects didn't have any visualization and had to use the keyboard to enter the values.

4.2 Interface Characteristics

4.2.1 Dynamic Queries interface

Studying slider use revealed several interesting possibilities for improvements. Most subjects had never used the optical mouse before and had problems pointing accurately enough with it. This caused problems with the slider as the drag box was small. Similarly several subjects found it hard to click on the slider bar to "fine tune" the setting. Also the fine tuning feature caused problems as the mouse arrow moved to the end of the slider bar when users clicked on it. For experimental purposes, subjects were unable to type in a value for the slider setting, which several subjects did request. Moving the slider can cause confusion if you move it too fast, and several subjects were found clicking at the sides of the slider bar, to adjust the slider up/down one step at a time, when making big changes.

The interface was wide, ~14 inches, which many subjects found to be a problem. They were observed leaning backwards to get an overview of what was changing on the screen. This was in sharp contrast to the FT interface where subjects were observed to lean forward, put fingers on both screen and the provided periodic table to create some sort of mental model of what they saw. Although the colors used were found to be good by participants, question 3 asking for the largest element in a fairly large set of elements caused problems because it was difficult to see when one single change occurs in the graphical query result. This problem can be overcome by either high-lighting elements that changed last or introducing a short "click" sound every time the graphical output changes.

4.2.2 FG interface

Using the keyboard proved to give participants several problems. Subjects invariably failed to remember that they had to delete the last number and forgot to move the cursor to reach another field. It should be noted that three subjects, having somewhat extreme problems with the mouse, stated their definite preference of the FG interface and felt they had more control using it.

Participants found it hard to know the range of the property they were manipulating, even though the range was given to the left of the field. Analogously participants found it hard to know when they reached the upper bound. With the slider it was easy to grasp both the range and the current value. The slider provides an intuition about which set is selected by painting the area to the left of the drag box red and vice-versa for the area to the right. This can not be done metaphorically with textual input, and accordingly subjects were found having trouble grasping which elements were selected.

4.2.3 FT interface

The FT interface performed very poorly compared to both the dynamic queries interface and the FG interface. This was also reflected in the user subjective evaluation (see section 4.3). This was to be expected but it was interesting to see how subjects reacted when the model of the periodic table was taken away, and they had to create one of their own. Using the FT interface, participants were found holding one hand pointing at the screen and the other on the provided printed periodic table, trying to interpret the query result.

4.3 Subjective Evaluation

The superior performance using dynamic queries compared to the FG interface was not reflected in the QUIS results. This is surprising as several QUIS questions addressed commands and ways of solving tasks.

Although it was not reflected in the QUIS results, subjects' delight was most obvious using the dynamic queries interface. They offered comments such as "The sliders are more fun than the key punch", "With the sliders you can watch the periodic table and see what changes color right before your eyes", "dynamic queries presented a more direct method of entering data for trial and error attempt", "You can play around more without worrying about messing it up".

Subjects having problems with the mouse stated for the FG interface: "You have more control over the numbers and you can read better what changes you have made." Some subjects using the dynamic queries interface asked: "Can I set the value directly instead of this guessing?" Participants were very critical of the FT interface, which also was reflected in the users subjective evaluation, the QUIS. But some positive responses were found, one subject stated "You can see what you have done before".

5. FUTURE RESEARCH

Further research about dynamic queries is needed. The sliders must be examined further:

- construct sliders giving ranges not bound to the minimum or maximum values by providing two drag boxes, and the issues of displaying such a range.
- select a set of sliders from a large set of properties, and
- select boolean combinations of sliders.

The visualization is equally important to examine. For example how to:

- find good visualizations for databases that do not have natural representations as a map.

- solve the problem of visualizations too large to fit into one screen, and
- find the best highlighting methods, such as colors, points of light, blinking, etc.

The last and maybe most important issue to examine is other applications of dynamic queries. How can direct manipulation of databases not consisting of well-formed ordinal data be implemented?

6. CONCLUSIONS

Results of this experiment suggest that direct manipulation can be applied to database queries with success. Results showed that visualization of the database and query result is the most important part of the dynamic queries, but that sliders direct manipulation of the query are also important.

For dynamic queries to be successfully implemented, several issues must be addressed. A good visualization must be found, such as a map, an organization chart, or a table, with good color combinations for highlighting. The control panel manipulating the query must be placed in a logical way to reduce eye and mouse movement. Sliders must be implemented so they are easy for novice users to use, i.e the drag box must be big enough and the slider must provide enough information without being cluttered. The search time must be immediate so that users feel in control and have a sense of causation.

7. ACKNOWLEDGEMENTS

We appreciate the support of Sun Microsystems, Inc. and NCR Corporation; as well as the comments from Rick Chimera, Holmes Liao and other members of the HCIL. Thanks also to Staffan Truvé and Dr Samuel O. Grim for help in constructing the tasks.

8. REFERENCES

1. *OPEN LOOK - GUI Functional Specification.* Sun Microsystems, Inc. Reading, MA, 1989.

2. Chin, J., Diehl, V., Norman, K. Development of an instrument measuring user satisfaction of the human-computer interface. *Proc. CHI'88 Human Factors in Comp. Systems Conf.,* ACM Press, pp. 213-218.

3. Kim H., Korth H, Silberschatz A. PICASSO: A Graphical Query Language, *Software - Practice and Experience 18*, 3 (March 1988), pp. 169-203.

4. Larsson, James A. A Visual Approach to Browsing in a Database Environment. *IEEE Computer 19*, 6 (June 1986), pp. 62-71.

5. Norman, Donald A. *The Psychology of Everyday Things.* Basic Books, Inc., New York, 1988.

6. Rowe, L.A. Fill-in-the-Form Programming. *Proc. 11th International on Very Large Databases*. ACM Press, 1985, pp. 394-403.

7. Rutkowski, Chris. An Introduction to the Human Applications Standard Computer Interface. *Byte 7* (Oct. 1982), pp. 291-310.

8. Shneiderman, Ben, Direct Manipulation: A step beyond programming languages, *IEEE Computer 16*, 8 (August 1983), pp. 57-69.

9. Williamson, C., Shneiderman, B. *The Dynamic HomeFinder: Evaluating Dyunamic Queries in a Real-Estate Information Exploration System*, University of Maryland CS-TR-2819, College Park, MD, 1992.

10. Zloof M. Query-by-Example. *Proc. National Computer Conference*, AFIPS Press, 1975, 431-437.