A Query Language for Data Access in Ubiquitous Environments

Cristiano Silveira[1], Leonardo Eloy[2], José Maria Monteiro[3]

[1] Mestrado Integrado Profissional em Computação - UECE
[2] Centro de Ciências Tecnológicas - UNIFOR
[3] Secretaria da Fazenda do Estado do Ceará - SEFAZ-CE

Abstract. Recent progress in ubiquitous computing technology is allowing development of new and sophisticated applications that are now able to store very large data on the mobile device itself, besides access data stored remotely. Thus emerges the need to query and maintain data from portable equipments. However, traditional query languages such as SQL (Structure Query Language) and QbE (Query by Example) are inadequate for devices with small screens. Hence, this work presents QbZ (Query by Zoom), a query language based on Semantic Zoom for small-screen devices. The proposed mechanism is particularly suitable to locate and access data stored in portable equipments. In order to demonstrate the benefits of using QbZ, usability tests were conducted, as well as a comparative analysis with the existing solutions.

Keywords: Query Language, Ubiquitous Environments, Semantic Zoom

1 Introduction

Recent progress in technology used in portable devices (such as PDAs, cell phones and smartphones) is contributing to diminish the cost of these equipments, making them accessible to a growing group of persons. This progress has contributed to increasing the computational and storage capacities of portable equipment. In the beginning of 2008 HTC, a smatphones manufacturer, introduced the HTC X9501, which has a 40 GB HD [9].

Today, with the advancement of technology in both data communication networks and accessing devices, new and sophisticated ubiquitous applications can be developed, the demand for which should follow the growth trend. Examples of these applications are: teaching and collaborative working, electronic tourist guides, navigation and route planning [1]. Thus, these new applications will handle and store a large volume of information. Consequently, emerges the need to query available data. However, traditional query languages, such as SQL (Structure Query Language) and QbE (Query by Example), are inadequate for devices with small screens [10]. Thereby, it is very important to investigate new paradigms and metaphors to design query languages toward ubiquitous environments. This work introduces QbZ (Query by Zoom), a query language based on Semantic Zoom [11] for small-screen devices. The proposed mechanism makes it easier to locate and access data.

This article is organized as follows: Section 2 discusses related works; Section 3 presents the QbZ language; Section 4 presents the results of tests undertaken; Section 5 concludes this work and points out directions for future works.

2 Basic Concepts

The success of a database system is, to a great extent, connected to the ease of recovering stored information. It is through the database query language that the user is able to express a set of restrictions and select the information desired.

A query language is defined as a high level computing language for recovering and modifying data. A relational query language can be understood as a formally well defined set of operators that can be combined to express queries in databases [4]. The process of formulating the query is carried out in three steps, according to [7]:

1. **Location**: the user selects a set of information of his interest, that is, objects that will make up part of his response set, or that will be used to restrict the set of values of the response;
2. **Definition of requirements**: some restrictions are applied to the objects selected in the previous phase, and these restrictions must be satisfied in order for such objects to become a part of the query result;
3. **Viewing the result**: the result of the query is presented to the user. The same set of information may be presented in different ways, but the choice of the adequate representation allows better interpretation of the results by the user.

Query languages can be classified into two large groups: text languages and visual languages. A text query language (SQL, for example) requires the user to have great knowledge of syntax and of the manner in which the information is stored in the database. Visual query languages, or Visual Query Systems (VQS), present a friendlier environment for recovering information, allowing users unfamiliar with text language syntax to formulate their queries in a simple and intuitive manner [7].

The Visual Query Systems (VQS) use a language to express the queries in visual format, and a variety of functionalities to favor interaction of the user with the system [7]. VQS can be seen as an extension of DBMS (Database Management Systems), allowing data queries to be formulated by less experienced users.

A VQS can be divided into two parts: external query model, and internal query model [4]. The external model defines the manner in which the user will visualize and manipulate the information. The internal model, in turn, is used to represent the queries elaborated by the user through the external model, in the query language defined by the implementation environment. For example, the external query model may use QbE, while the internal model uses SQL. In this case the user elaborates his query using QbE, and this query is mapped to SQL, so it can be executed on the DBMS. In order to allow translation between

the external and internal models, a mapping module must exist that promotes the relationship between the operations and representations of each model [4].

The external model must present great power of expression and a very friendly query language. The internal model on the other hand, depends on the DBMS used and the query language it supports. The possibility of using different models allows the environment for interaction with the user to be defined without considering the implementation aspects.

VQS can be classified into four paradigms: form-based language, diagram-based language, icon-based language, and hybrid language [4]. Although VQSs have been used with considerable success in applications that execute in personal desktop computers, their use in small-screen devices, like cell phones and smartphones, for example, has not proved adequate [10]. Hence, the investigation and definition of new paradigms for the construction of VQSs that are more appropriate for portable devices have become an important research area.

3 Related Work

Massari [10] presents a prototype of a query processing facility that supports the exploration and query databases from mobile computer based on the manipulation of icons. This prototype is called Query By Icons (QBI). In this approach the user primarily interact with the system with a pointing device, such as a pen or a mouse, and compose query by arranging icons. However, this prototype was not implemented for mobile phones.

In [6] the authors propose a graphical query language called MoSQL to be the basis of general mobile information systems. This language has an icon-based interface. That is, the same class of data represented in the database system corresponds to the same icon. This windows-based interface is suitable for mobile devices that can be operated by clicking or dragging the mouse, like notebooks and PDAs. So, it is not suitable for mobile phones.

The paper [2] presents the design of a high-level web-based user interface using IRE (Information Requirement Elicitation), in the context of "Hospital database queries" by a mobile web user. The prototype is based on the notion of Query-By-Object (QBO) approach of building a query using multiple user-level steps. The Query-by-object (QBO) interface has been studied for Geographic Data Systems.

In [1] the authors introduce a free-form database query language which was implemented in a database query system prototype for WAP-enabled mobile phones. Free-form is a concept which is based on the universal relation [1]. It can be used to reduce the number of terms needed in a query. Hence, using the language, any relational query can be formulated without the need to provide precise inputs. Though, due to imprecise input may lead to many interpretations in terms of the attributes to be displayed as outputs. Furthermore, the prototype makes intensive use of menus and the database schema is showed in a very limited manner (using in a list component).

Polyviou, et al., [12] discussed expressive queries in their work which implement directory-like interface for query formulation. However, their method is suitable only to be used with devices that have pen input mechanism. As the majority of mobile phones lack this type of input mechanism, we believe that a conventional way of input, i.e., input using normal phones keypad and function keys, should be considered. Another interesting option to be considered for inputs would be voice input. However, as being highlighted in [5], this method has one major problem of voice recognition which needs further research.

4 Query by Zoom Language

The objective of this section is to present and describe the QbZ (Query by Zoom) language. QbZ consists of a visual query language based on Semantic Zoom [11], directed to small-screen devices. The QbZ language follows the standard organization of visual languages, containing an external model and an internal model. The external model uses Semantic Zoom as main metaphor, while the internal model uses the SQL text language.

The basic idea of Semantic Zoom is to allow the user to have control over a large volume of visual objects [3]. As the user closes in on the information, more details of said information are exhibited [11]. Control is carried out by using: Zoom In, whereby the user closes in on the desired information and, depending on the approximation, there is a level crossing; Zoom Out, whereby the user backs off from the information; and Pan, where there is a lateral movement at the same level. Although it is not a recent idea, interfaces based on Zoom have become more prominent in the construction of visual interfaces for portable devices [3].

4.1 The External Model

A query in QbZ is represented by a sequence of actions. These actions can be classified in three categories:

- Visual actions of semantic zoom;
- Query formulation actions;
- Data recovery actions;

In all, QbZ offers 15 operations (or actions), as showed in Figure 1:

- **AZ**: Activates ZOOM mode. From this point onward the keys of the mobile device are ready to activate the ZOOM IN and ZOOM OUT modes.
- ZI^N: Indicates ZOOM IN. Enlarges the selected screen by "n" times.
- ZO^N: Indicates ZOOM OUT. Reduces the selected screen by "n" times.
- **DZ**: Disables ZOOM mode.
- **SHIFT UP, SHIFT DOWN, SHIFT LEFT, SHIFT RIGHT**: Activates displacement, displacing the visualization content, changing the part that will become visible on the portable device screen.

– **TABLE SELECTION**: Selects a table.
– PROJECTION: Selects columns in the indicated table.
– **SELECTION**: Selects the lines of the indicated table that satisfy a certain condition (predicate).
– **JOIN MODE**: Indicates the start of a joining operation.
– **JOIN FIELD ON LEFT** and **JOIN FIELD ON RIGHT**: Indicate the columns that will make up the joining condition.
– **HOLD JOIN**: Concludes a joining action.

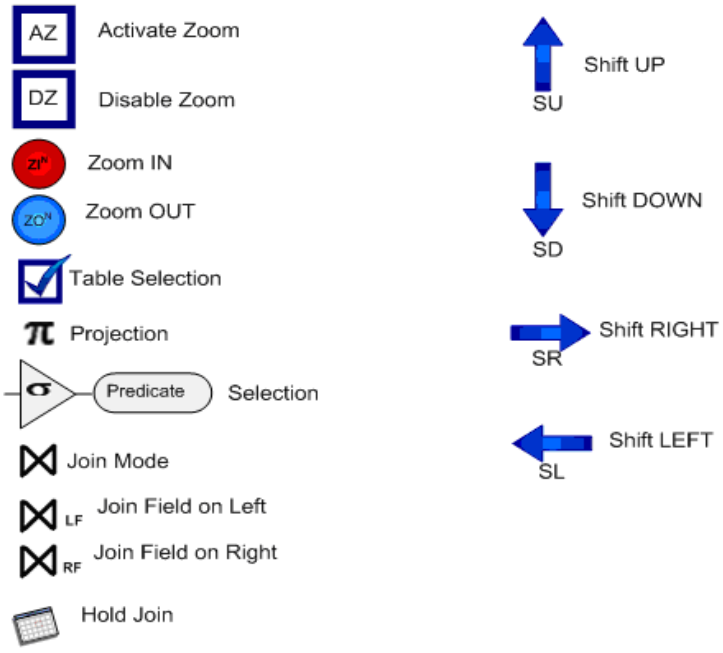


Fig. 1. Operators of the QbZ External Model.

4.2 The Internal Model

The internal model of the QbZ language uses the SQL text language. Thus, the queries elaborated using the external model of QbZ must be translated to SQL.

4.3 Translation from the External Model to the Internal Model

A query elaborated from the external model of QbZ corresponds to a sequence of actions. This sequence of actions can be represented by a graph. In order to make it easier to understand the process of translating from the external model to the internal model, we will represent each action that composes the external model of the QbZ with a symbol. Figure 1 illustrates the actions that compose the external model of the QbZ language.

The translation process starts by capturing the actions executed by the user by means of the external model [4]. Next, a graph of QbZ operators is generated to represent the query elaborated by the user. The next step consists of mapping the QbZ operator graph to a graph of relational algebra operators. Note that, for this, the visual actions (operations) of Semantic Zoom are discarded, as are the data recovery actions that may exist on the QbZ operator graph. To conclude the translation process the relational algebra operator graph is translated to a text query in SQL.

4.4 Semantic Zoom Levels

The designed strategy uses three levels of Semantic Zoom: the level 1, which initially shows information about the database schema; the level 2, appears after selecting a table and shows information about the selected table, such as: name, size and type of attributes, etc; and the level 3, displayed after the selection of a particular attribute, which displays the details related to the selected attribute and the actions that can be executed from this attribute, as the addition of a projection or a selection condition. These three levels are exemplified in Figure 2.

4.5 Strategies for User Interface Design

According to [4], the goal of the people working with a VQS is to retrieve the desired data. This is usually accomplished through the following two main activities:

1. **Understanding the reality of interest**. The goal of this activity is the precise definition of the fragment of schema involved in the query.
2. **Formulating the query**. The goal of query formulation is to formally express the operands involved in the query, with their related operators.

Understanding the reality of interest by the user is a task that previously was not considered by the query systems. The complexity of this task is directly related to the size of the database schema and workload. This process of understanding the model can be very slow, depending on the knowledge that the user already has from the model. The developed strategies try to ensure a gradual construction of the mental model, with the information already caught about the analyzed data model. All new information captured generate a change in the mental model trying to make it close to reality [4]. There are three strategies that can be adopted in order to facilitate this task: Top-down, Browsing and Schema Simplification [4]. The Browsing technique consists in the study of element to element in a diagram of entities. Each selected element is checked, the useful information is acquired by the user and the process continues by neighboring elements.

Query formulation is the fundamental activity in the process of data retrieval. For this task, the main techniques that can be used are: Schema Navigation, Subqueries and Matching [4]. The query formulation strategy by schema navigation

has the characteristic of concentrating on a concept (or a group of concepts) and moving from it in order to reach other concepts of interest, on which further conditions may be specified. Such a strategy differs according to the type of path followed during the navigation [4].

In the mechanism proposed in this paper we used the technique "Browsing" to understanding the reality of interest. For the query formulation task was used "schema navigation". However, to implement browsing and schema navigation we propose to use Semantic Zoom.

In the case of Semantic Zoom, the schema is unique, and the concepts are layered in terms of levels of importance; the schema can be examined at several levels, so that only objects above a specified importance level are visible. The user can also graphically edit the schema, so that irrelevant objects can be removed from the screen. The schema is also unique, but the objects may be examined at different hierarchical detail levels.

5 Running Example

In order to demonstrate the efficiency of the proposed approach a QbZ prototype was implemented, called NanoZoom, using NanoBase as DBMS. NanoBase provides a relational view for access to data on the JME CLCD/MIDP platform, allowing the use of DDL/DML clauses, integrity restrictions, as well as several index structures [8]. NanoZoom was implemented using the JME CLDC/MIDP platform.

5.1 Example 1

Consider the following task: for each dependent retrieve your ID, your name and the name of the employee involved. Figure 3 shows the QbZ Expression, generated by the QbZ external model, to retrieve the requested data. Figure 4 shows the QbZ operator graph, generated by the QbZ internal model, for the Example 1.

This QbZ graph is translated to a relational algebra expression:

$$\Pi_{D.Id,D.Name,E.Name}(Dependent \bowtie_{D.Employee=E.Id} Employee)$$

Next, the relational algebra expression is translated to a SQL clause:

```
SELECT D.ID, D.NAME, E.NAME
FROM DEPENDENT D, EMPLOYEE E
WHRE D.EMPLOYEE = E.ID
```

Finally, the SQL expression is send to the DBMS and the result is displayed.

6 Conclusions and Future Works

In this article we present QbZ (Query by Zoom), a query language based on Semantic Zoom, for portable devices. This language makes it easier to locate and access data stored in the mobile devices. The information recovery in QbZ system has yet some additional features, such as: visual representation, definition of detail levels and navigation on the graph scheme, storage of queries for later use, conversion of a query form visual to textual representation or vice versa, among others.

As future works we intend to conduct more strict usability tests, use the TinySVG graphical library in the implementation of the visual interface and implement other relational operators like sort and aggregation operations.

References

1. R. Ahmad and S. Abdul-Kareem. A free-form database query language for mobile phones. *Communications and Mobile Computing, International Conference on*, 3:279–284, 2009.
2. S. Bhalla and M. Hasegawa. A query interface for ubiqitous access to database resources. In *COMAD '06: Proceedings of Conference on Management of Data*, New Delhi, India, 2006.
3. T. Buering, J. Gerken, and H. Reiterer. User interaction with scatterplots on small screens - a comparative evaluation of geometric-semantic zoom and fisheye distortion. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):829–836, 2006.
4. T. Catarci, M. F. Costabile, S. Levialdi, and C. Batini. Visual query system for databases: A survey. *Journal of Visual Languages and Computing*, 1997.
5. E. Chang, F. Seide, H. M. Meng, Z. Chen, Y. Shi, and Y.-C. Li. A system for spoken query information retrieval on mobile devices. *Speech and Audio Processing, IEEE Transactions on Publication*, 10:531– 541, 2002.
6. Y.-H. Chang. A graphical query language for mobile information systems. *SIGMOD Rec.*, 32(1):20–25, 2003.
7. B. Czejdo, R. Elmasri, M. Rusinkiewicz, and D. W. Embley. A graphical data manipulation language for an extended entity-relationship model. *Computer*, 23(3):26–36, 1990.
8. L. Eloy, V. Arajo, J. M. Monteiro, and A. Brayner. A tiny relation database manager for the jme cldc/midp platform. *Revista Tecnologia*, 29(1):7–15, 2008.
9. HTC. Htc x9501. 2008. Available at http://www.htc.com/www/product.aspx.
10. A. Massari and S. W. P. K. Chrysanthis. Supporting mobile database access through query by icons. In *Distributed and Parallel Databases Jour*, pages 4–249, 1996.
11. K. Perlin and D. Fox. Pad: An alternative approach to the computer interface. In *Proceedings of SIGGRAPH 93*, 1993.
12. S. Polyviou, G. Samaras, and P. Evripidou. A relationally complete visual query language for heterogeneous data sources and pervasive querying. In *ICDE '05: Proceedings of the 21st International Conference on Data Engineering*, pages 471–482, Washington, DC, USA, 2005. IEEE Computer Society.

The application starts at the first level showing the database schema.

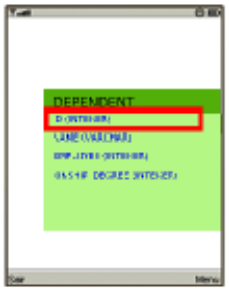
Level 1 – View 1

The user turns the Zoom and runs *Zoom In*, closing to an area of interest.

Level 1 – View 2

The user runs *Zoom In*, closing to the tables Dependent and Employee.

Level 1 – View 3



The selection area of the table Dependent is turn on. In this moment, the user can select the Dependent table and goes to the second semantic level.

Level 2 – View 4

The user selects the Dependent table and it is included in the query. The level of showed details is increased.

Level 2 – View 5

The column selection area is turn on. In this moment, the user can select a column and goes to the third semantic level.

Level 3 - View 6

Fig. 2. Semantic Zoom Levels.

Fig. 3. QbZ Expression for the Example 1.

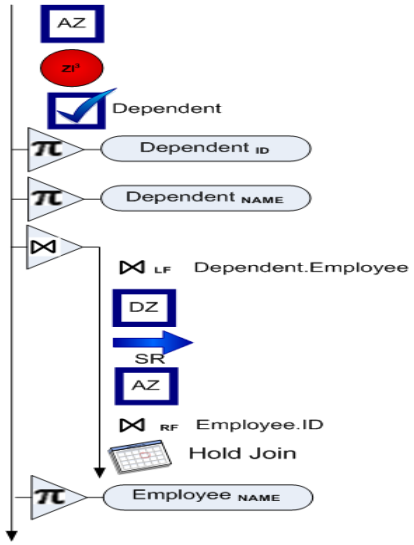


Fig. 4. QbZ Operator Graph for the Example 1.