CONTEXT-AWARE QUERIES USING QUERY BY BROWSING AND CHIROMANCER

Stavros Polyviou[*], Paraskevas Evripidou[†], George Samaras[‡]

Abstract
In this paper we present Chiromancer, a relationally complete visual query language for the Palm handheld platform, based on the Query by Browsing paradigm. We discuss the suitability of the relational model as the description model of all aspects of a pervasive environment, and hence the suitability of Chiromancer as a generic access and application development tool for such an environment[1]*. We briefly introduce the QBB paradigm and an overview of the Chiromancer design. We demonstrate the context-awareness aspects of our proposal by examining the implementation of some examples from the research literature using Chiromancer. Finally, we provide a conclusion as well as some suggestions for future work.*

1. Introduction

Semantic modeling is an important research challenge in pervasive computing [3, 8]; only by capturing all aspects of a pervasive computing environment such as environmental conditions, hardware capabilities, as well as user faculties, goals and mental models [5] in a machine-processable way, will we be able to deliver on the requirements of context-awareness [1, 2, 3, 4, 8, 9, 11, 13, 14, 15], mobility [2, 3, 4, 7, 8, 9, 11, 13], dynamic execution environments [3, 12] and activity-based rather than application-centric interaction paradigms [1, 2, 3, 9, 12, 16]. We believe that a time-proven, adaptable, powerful, distributable, reliable and elegant model such as the relational model [6] is the most suitable candidate for this task. Based on this model we can create schemata in order capture, manage, distribute and analyze data regarding both the virtual and the physical world of a pervasive computing environment . In this paper we discuss Chiromancer, the first relationally complete visual query language (VQL) designed for a handheld device, based on the Query by Browsing paradigm that we developed. We demonstrate Chiromancer's ability to function both as a generic tool to a pervasive computing infrastructure, the need for which is advocated in [12], and as a development tool for pervasive applications [2].

[*]University of Cyprus, Department of Computing, 75 Kallipoleos St, 1075 Nicosia, Cyprus, polyviou@cs.ucy.ac.cy
[†]University of Cyprus, Department of Computing, 75 Kallipoleos St, 1075 Nicosia, Cyprus, skevos@cs.ucy.ac.cy
[‡]University of Cyprus, Department of Computing, 75 Kallipoleos St, 1075 Nicosia, Cyprus, cssamara@cs.ucy.ac.cy

[1]The design of such schemata however is outside the scope of this paper.

[2]This work is partially funded by the Information Society Technologies programme of the European Commission under the IST-2001-32645 DBGlobe and IST-2001-35495 MEMO projects.

2. The Query by Browsing Paradigm

Query by Browsing (QBB) is based on the concept of querying a relational schema using familiar concepts from the desktop user interface paradigm. This is achieved by representing both the schema and the query as a *folder hierarchy*. A folder is a *materialized view* on the data stored in a database table[3] ; the table's columns become the folder's *attributes* and its rows become the folder's *records*. A folder is also a *container of objects*; these objects are used in the definition of the view represented by the folder and the presentation of the data it contains. The first such object is the *subfolder*. The relationship between a parent folder and a subfolder is described using a set of *cross-reference criteria* which include the *correspondence condition*, the *correspondence cardinality* and the *inclusion option*. The former is a first-order three-level predicate logic expression involving the attributes of both folders. The correspondence cardinality defines the number of records in the subfolder for which the correspondence condition must hold true. The inclusion option allows the inclusion of those records from the either the parent folder or the subfolder, or both, that do not satisfy the correspondence condition for the number of subfolder records required by the correspondence cardinality.

A subfolder's records are always restricted by the cross-reference criteria that relate it to its parent folder. In order for the parent folder's records to be restricted by these criteria however, the subfolder needs to be *activated* [4] . Multiple parent folders can be defined for a subfolder through the use of *folder shortcuts*, which are another type of QBB object. A different set of cross-reference criteria is required between a subfolder and each of its parent folders. Folder shortcuts may be activated independently of the subfolder they refer to. Folders are created based on a *folder template*. *Templates* are QBB objects that contain *intentional queries*[5] . These are queries on the structure of the database schema and that of QBB objects which are described using a set of relational tables, collectively referred to as the QBB catalog [6].

Templates[7] can be used to create other QBB objects such as *documents* and *applications*, which are materialized views on the data contained in their parent folder and its subfolders [8]. The distinction between the two is that documents are used for data display, whereas applications can be used to insert, update and delete data in the database, provided the view defined by the application is updatable. If a document or application references the attributes of a subfolder, then the subfolder is *implicitly activated* [9]. A special kind of application, called a *filter*, is used for *restricting* the records of its parent folder. Based on user action, a filter produces a first-order three-level predicate logic expression, referred to as the *filtering criteria*, which is used to restrict the records of its parent folder whenever the filter is *activated*. Filters are based on *filter templates*.

The collection of criteria generated by all activated objects (subfolders, filters and folder shortcuts) contained in a folder is referred to as the *folder criteria*. Each folder is associated with a *criteria combination option* which defines the way these various object criteria are combined into the folder criteria. The criteria generated by each object are treated as a unit. The combination option defines

[3] We use the term 'table' to mean 'relation'; this includes base tables and views for example.

[4] Activating a subfolder can be thought of as assigning the result of the relational operation between a parent folder A and its subfolder B to the parent folder, i.e. A = A θ B, where θ is a relational operator such as JOIN or DIVIDE.

[5] As opposed to extensional queries which query the actual data stored in the database.

[6] This can be thought of as an extended version of a relational database's system catalog which includes information regarding tables, columns, domains, functions, stored procedures and constraints (triggers).

[7] The importance of templates in activity-based pervasive computing scenarios is pointed out in [12].

[8] Documents and applications usually contain a projection of the data contained in the parent folder and its subfolders.

[9] Folder shortcuts can also be implicitly activated in this way.

a range in the number of criteria that needs to be satisfied or failed. *Grouping folders* can be used to group activated objects, thus fulfilling a role similar to that of parentheses in Boolean algebra.

A QBB object such as a folder, document, application or filter can be independently static or dynamic at the *intentional level*, if it is based on a static or dynamic template respectively. A static template is impervious to changes in the QBB schema such as the creation of new folders, attributes or database tables. A dynamic template on the other hand reacts to such events and may cause an update in the contents of any objects that are based on it. A QBB object can also be static or dynamic at the *extensional level*, regardless of whether it is based on a static or dynamic template. An extensionally static object represents a snapshot of the database state, at the time the object was created. A dynamic template can be thought of as placing *triggers* [10] on the QBB catalog tables. Similarly a dynamic object places triggers on the table or tables it draws its data from.

3. Chiromancer in Context

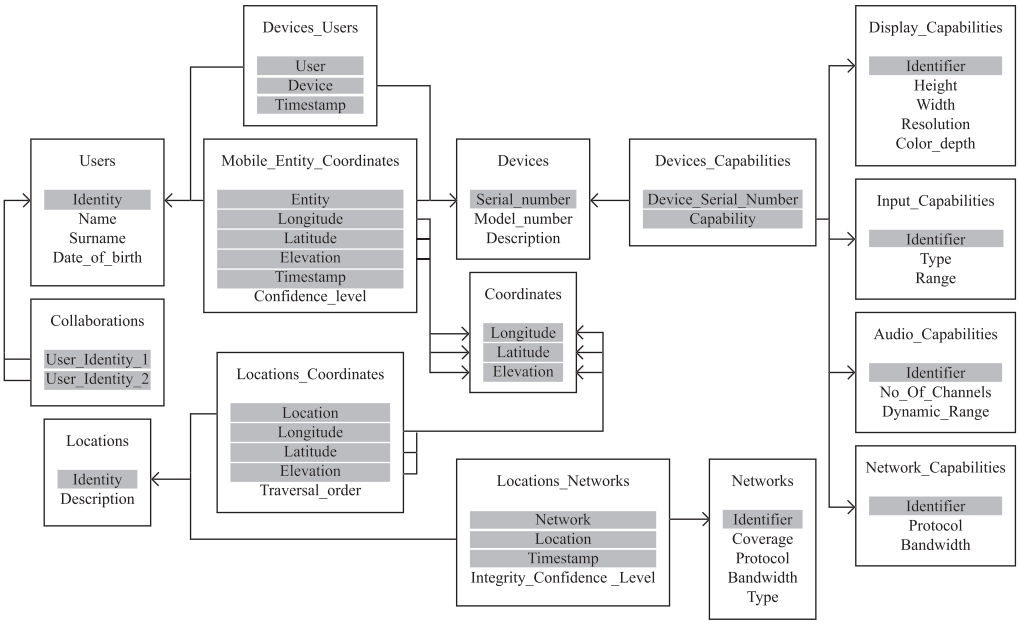

Figure 1. A rudimentary context description schema

We have organized the elements of context in nine categories. The 'user' category contains information such as the user's identity, emotional state, locus of attention, preferences and so forth. The 'network' category contains information about the available network infrastructure such as bandwidth, reliability, and range. The 'time' category captures the current (local) time and date. The 'services' category includes any software service that provides a capability such as transcoding [16]. 'Devices' refers to all available devices in the pervasive environment and their capabilities. The 'collaborators' category encompasses all people who are relevant to the pervasive task at hand. 'Location' is an important aspect that can be described using a Cartesian (geometric) or topological [11] (symbolic) terms [8, 11]. Exploitation of the information in this category helps us create so-called *attentive environments* [16] or opportunistic communication [10, 11]. 'Social context' involves issues of etiquette. Finally, the 'environment' category encompasses all aspects of the physical environment that affect the execution of the task at hand, such as temperature, visibility, noise and light levels etc.

[10] For a discussion of the use of triggers in pervasive computing scenarios refer to [10].

[11] A topological model can in fact be derived based on a Cartesian model involving GPS coordinates [11].

Context information is collected through a network of sensors and services. Each piece of information collected is associated with a *confidence level*, which expresses the probability of its correctness. We describe this information using a set of relational tables, thereon referred to as *context description tables*. By using a flexible querying interface such as Chiromancer, we are able to accurately define the who, where, when and why [1] of a pervasive interaction. Such context description tables maintain a historical record of the user's context [1] collected from the lower levels of the pervasive infrastructure; by creating dynamic QBB objects based on these tables, we are able to create context-aware pervasive queries. Figure 1 shows a subset of a rudimentary schema that could be used for describing context in a pervasive computing environment.

4. The Chiromancer Interface

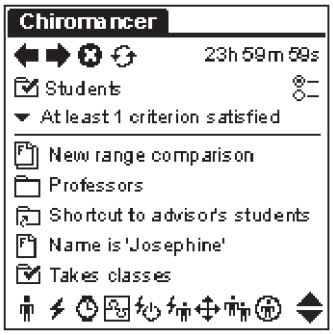

Figure 2. The Chiromancer interface

As can be seen in figure 2, the Chiromancer interface resembles in part a familiar browser interface using icons such as 'back', 'forward', 'stop' and 'refresh'. Chiromancer's folder template displays the folder hierarchy in columns, arranged from left to right, starting from the higher levels of the hierarchy and moving towards the lower levels, in a manner similar to the Apple iPod portable music player. The 'back' and 'forward' buttons allow the user to move through the columns. The 'stop' button cancels the current transaction over the network, while the 'refresh' button updates the contents of the currently displayed object based on the latest data in the database. The elapsed time since the last update is displayed next to this button. Such information is important in pervasive applications, as pointed out in [8]. The name of the currently displayed object and an icon indicating the type of the object are also displayed. The 'options' icon next to it invokes a modal form that allows the user to sort or group the object's contents. In the case of dynamic objects, the user can specify the time period (when, for how long) and frequency (how often) with which the contents of the object should be refreshed. Once the object has been updated, the user can be notified of this fact.

Keeping the user up-to-date with regards to the state of the pervasive computing environment is important, as noted in [8]. The context-awareness status area fulfills this role by displaying nine status icons: user, network, (local) time, infrastructure, related devices, related people, location, social context and environment. The user can set minimum confidence levels for each category for each object; if the current confidence level falls below this minimum requirement, then the icon is shown in inverse video. The user can tap on such an icon to manually enter that portion of the context. Chiromancer also provides a set of filter templates that correspond to the basic SQL predicate types. These are 'New comparison', 'New range comparison', 'New missing test', 'New text match' and 'New list comparison'. In addition, three document and application templates allow users to view and edit

the displayed data as a table ('View as table'), as a list ('View as list') and as columns ('View as columns').

5. Some Illustrative Examples

We will now discuss some examples based on Chiromancer's querying and context-awareness capabilities. Each one of the examples represents one of the five application types listed in [14] which represent typical scenarios that have been explored in the research. These are routing, addressing, messaging, caller awareness and screening.

Routing involves the delivery of messages to a device that is within the recipient's physical context. This can be achieved in Chiromancer by creating a subfolder based on the 'Devices' table called 'Co-located devices' inside a folder based on the 'Users' table. The two folders could be joined on location via the 'Mobile_Entity_Coordinates\Coordinates\Locations_Coordinates\Locations' folder. This subfolder therefore would now contain all devices in the user's vicinity. A third party can access the 'Co-located devices' folder and restrict the 'Users' folder using a filter to include only the user he is interested in contacting. This way, the 'Co-located devices' folder would contain only those devices that are co-located with the intended recipient of the communication.

Addressing involves selecting the recipients of some communication based on context. Sending for example a message to one's co-workers involves creating a folder based on the 'Users' table and restricting that table to the user's identity, using a filter. The folder could be named after the user, say 'John Smith'. Creating a subfolder called 'Collaborators' based on the 'Users' folder and joining to the 'John Smith' folder via the 'Collaborations' table, will populate the subfolder with a list of all the user's collaborators. Creating a folder shortcut to the 'Locations' [12] subfolder of the 'Collaborators' folder and joining it to 'John Smith' based on location, would further restrict the list of recipients to those collaborators who are co-located with the user.

Messaging involves associating a message with a specific context, such as a location, and delivering that message once that context is realized. Using the procedure outlined in the previous example, the user could create a dynamic folder of co-located collaborators, and instruct Chiromancer to notify him every time the folder's contents are updated. i.e. every time a collaborator is in the user's vicinity.

Caller awareness involves 'sharing an awareness of one's environment and the context of friends, family and colleagues'. A folder could collect environmental information such as temperature, humidity, rainfall etc, and notify a user whenever the weather situation has changed, i.e. the rain is over. Such information could be routed to a co-located device such as an ambient display, using a folder based on the 'Devices' table.

Screening 'concerns establishing communication under appropriate conditions'. Using the same 'Collaborators' subfolder, one could access information such as a colleague's mood or social context, not shown in our rudimentary schema, to decide whether it is the right time to communicate with him. The user would of course have to be authorized to access such information; collecting such sensitive information in a folder and granting access to specific users for that folder maintains a user's privacy and control.

[12] 'Mobile_Entity_Coordinates\Coordinates\Locations_Coordinates\Locations' to be precise.

6. Conclusion and Future Work

By combining the power and elegance of the relational model, the usability and familiarity of the desktop paradigm and the convenience of context-awareness, Chiromancer creates new avenues for the implementation of pervasive scenarios. We believe that our QBB paradigm combined with a suitable context-aware schema describing all aspects of a pervasive environment can go a long way towards realizing the vision of ubiquitous computing. In the future we plan to devise such a comprehensive schema and to evaluate Chiromancer and QBB using real users in real pervasive scenarios. We would also like to explore the concept of templates as the means for creating pervasive applications. If successful, pervasive computing will make interaction with computers as 'refreshing as taking a walk in the woods' [17]. We believe that the road to the realization of that goal will be as exciting as a roller coaster ride.

References

[1] ABOWD, G. D., MYNATT, E. D., RODDEN, T. The Human Experience In *P*ervasive Computing, Jan-Mar 2002.

[2] ANHALT, J., SMAILAGIC, A., SIEWIOREK, D. P., GEMPERLE, F., SALBER, D., WEBER, S., BECK, J., JENNINGS, J. Toward Context-Aware Computing: Experiences and Lessons In Intelligent Systems, May-Jun 2001.

[3] BANAVAR, G., BERNSTEIN, A. Software Infrastructure and Design Challenges for Ubiquitous Computing Applications In *C*ommunications of the ACM, Vol. 45, No. 12, Dec 2002.

[4] BURRELL, J., TREADWELL, P., GAY, G. K. Designing for Context: Usability in a Ubiquitous Environment In *P*roceedings of the 2000 Conference on Universal Usability, 2000.

[5] CIARLETTA, L., DIMA, A. A Conceptual Model for Pervasive Computing In *P*roceeding of the 2000 International Workshop on Parallel Processing, 2000.

[6] CODD, E. F. *T*he Relational Model for Database Management: Version 2, Addison-Wesley, 1990.

[7] DAVIS, G. B. Anytime/Anyplace Computing and the Future of Knowledge Work In *C*ommunications of the ACM, Vol. 45, No. 12, Dec 2002.

[8] DIX, A., RODDEN, T., DAVIES, N., TREVOR, J., FRIDAY, A., PALFREYMAN, K., Exploiting Space and Location as a Design Framework for Interactive Mobile Systems, in: ACM Transaction on Computer-Human Interaction, Vol. 7, No. 3, Sep 2000.

[9] HESS, C. K., CAMPBELL, R. H., An application of a context-aware file system, in: Personal and Ubiquitous Computing, Volume 7, Issue 6, Dec 2003.

[10] HUANG, A. C., LING, B. C., PONNEKANTI, S., Pervasive Computing: What Is It Good For?, in: Proceedings of the 1st ACM international workshop on Data engineering for wireless and mobile access, 1999.

[11] LEE, D. L., XU, J., ZHENG, B., Data Management in Location-Dependent Information Services, in: Pervasive Computing, Jul-Sep 2002.

[12] NEWMAN, M. W., SEDIVY, J. Z., NEUWIRTH, C. M., EDWARDS, W. K., HONG, J. I., IZADI, S., MARCELO, K., SMITH, T. F., Designing for Serendipity: Supporting End-User Configuration of Ubiquitous Computing Environments, in: Proceedings of the conference on Designing interactive systems, 2002.

[13] SAHA, D., MUKHERJEE, A., Pervasive Computing: A Paradigm for the 21st Century, in: Computer, Mar 2003.

[14] SHILIT, B. N., HILBERT, D. M., TREVOR, J., Context-Aware Communication, in: Wireless Comm., Oct 2002.

[15] SMAILAGIC, A., SIEWIOREK, D., Application Design for Wearable and Context-Aware Computers, in: Pervasive Computing, Oct-Dec 2002.

[16] SPOHRER J., STEIN, M., User Experience in the Pervasive Computing Age, in: Multimedia, Jan-Mar 2000.

[17] WEISER, M., The Computer for the 21st Century, in: Scientific American, Sep 1991.