UNIVERSITY
OF TRENTO

DEPARTMENT OF INFORMATION AND COMMUNICATION TECHNOLOGY

38050 Povo – Trento (Italy), Via Sommarive 14
http://www.dit.unitn.it

COORDINATING MOBILE DATABASES:
A SYSTEM DEMONSTRATION

Ilya Zaihrayeu and Fausto Giunchiglia

May 2004

Technical Report # DIT-04-030

Also: demonstration session of the First Annual International Conference on
Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous
2004), 2004

Coordinating Mobile Databases:
A System Demonstration

Ilya Zaihrayeu and Fausto Giunchiglia

Dept. of Information and Communication Technology

University of Trento

38050, Povo, Trento, Italy

{ilya, fausto}@dit.unitn.it

I. ABSTRACT

In this paper we present the *Peer Database Management System* (PDBMS). This system runs on top of the standard database management system, and it allows it to connect its database with other (peer) databases on the network. A particularity of our solution is that PDBMS allows for conventional database technology to be effectively operational in mobile settings. We think of database mobility as a database network, where databases appear and disappear spontaneously and their network access points may change, and are not known a priori. There is a further request (and proposed PDBMS satisfies it) that databases must know, independently of their network access points, how to locate other databases, and how to interoperate with them on servicing user requests (i.e., queries and updates).

PDBMS is implemented on top of the Peer-to-Peer platform JXTA [1]. *Peer-to-Peer* (P2P) is a decentralized networking model where each party (called a *node* or a *peer*) has equivalent abilities in providing other parties with data and/or services. Peers are largely autonomous from other peers, and they interoperate in a local, point-to-point manner. All these notions are crucial from the point of view of mobility – databases may come and go, interact with different databases at different times or for answering different queries, the size of the network can dynamically shrink and expand depending on how many nodes are online, and databases can benefit from collaboration with one other by coordinating their data at runtime.

JXTA helps us to implement mobility by providing an IP-independent naming space to address nodes; it is system, and networking platform independent. This allows PDBMS to be completely portable and, therefore, "pluggable" on top of multiple host platforms. Moreover, the proposed software solution is a self-contained application that can be fit on a small capacity storage device as a flash drive, which can be easily handled around.

Each peer on the network provides a *source database* described by a (*source*) *schema*, or supplies only the schema. In this latter case a node acts as a kind of *mediator* in transitive propagation of data. Peers define *semantic data dependency links* between their schemas and use these links to coordinate data, i.e., answer input queries, propagate query results and updates. Input queries in the system are formulated w.r.t. the source schemas of single nodes. Peers are largely autonomous, in particular in what data they store, in which nodes they establish semantic data dependency links with and coordinate their data, etc.

PDBMS implements a fully decentralized data coordination model [2]. The four notions at the core of our model are Interest Groups, Acquaintances, Correspondence Rules, and Coordination Rules. The first notion allows for a global aggregation of nodes carrying similar information, while the second allows for a local logical point-to-point data exchange between databases. The acquaintance is not a symmetric notion, i.e. the fact that a node is acquainted with another node does not necessarily mean that the vice versa also holds. A node is an *acquainted node* for some other node if the latter is an acquaintance of the former. Acquaintances are associated with a set of acquaintance queries, which are used to import data from acquaintances' databases. An acquaintance query is the minimal block for building semantic data dependency links between peer databases. An acquaintance query is a conjunctive query [3], which head refers to some relation at a node, and its body is a query over the relations of a node's acquaintance. Correspondence Rules solve the heterogeneity problem at the instance level, namely they specify mappings between objects of the domains of the two nodes' databases. Finally, Coordination Rules are responsible for *data coordination* with acquaintances and acquainted nodes.

The data coordination model is implemented inside a concrete logical architecture, see Figure 1 (first level) and Figure 2 (second level). A node consists of *PDBMS*, a *Source Database* (SDB) and a *Source Schema* (SS). SS describes a shared part of SDB. PDBMS consists of *User Interface* (UI), *Database Manager* (DBM), *JXTA Layer* and *Wrapper*. DBM implements the four basic notions described above. JXTA Layer is responsible for all node's activities on the network, such as discovering of new nodes and interest groups, joining and leaving groups, sending and receiving queries and query results, and so on. Wrapper manages connections to SDB, it is responsible for extraction and maintenance of the source schema. Since different databases may require different database drivers, this module is adjustable depending on the underlying database.

On the second level architecture we "open" the DBM and JXTA Layer. Rectangles with rounded corners stand for data repositories which store various information. Normal rect-
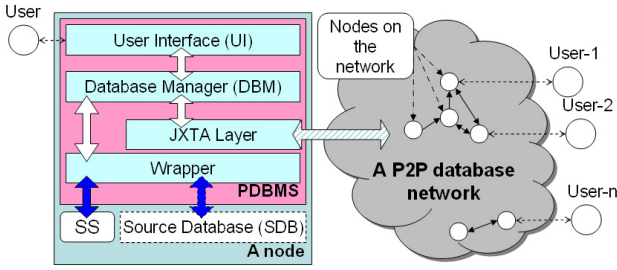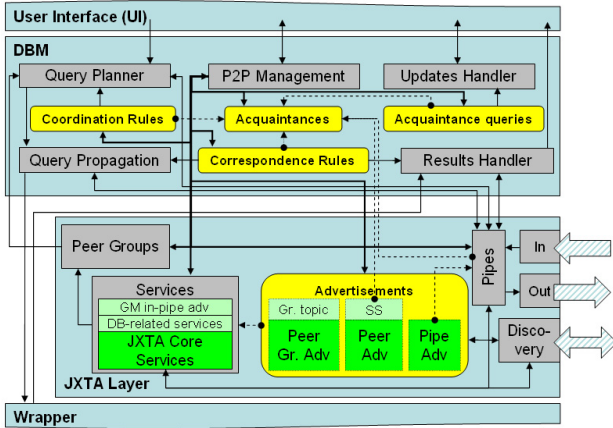
Fig. 1.   First level architecture: a node

Fig. 2.   Second level architecture: DBM and JXTA Layer

angles represent executive modules. The meaning of arrows between UI, DBM, JXTA Layer and Wrapper is the same as in Figure 1, namely, they represent procedure calls. Consider the JXTA Layer. The advertisements repository stores all discovered and locally created JXTA advertisements (see [2] for details on JXTA advertisements). Inside the rectangle, three advertisement types are represented, although in practice there are also others. The peer advertisement includes the source schema information. The Services module implements the core JXTA services (see [2] for details on JXTA core services) and DB-related services (i.e., the services required to run peers without databases).

Consider now DBM. The P2P Management module allows users to control other modules and repositories from both the DBM and JXTA Layer. For instance, it makes it possible to create a new communication link (called pipe), to make a new acquaintance or to modify a coordination rule. The control lines are shown as thick arrows from P2P Management to other components. Query Planner processes all input queries. It uses acquaintance queries, acquaintances and interest groups information in order to detect groups and nodes for propagation. The Query Propagation (QP) module takes this information as input and uses correspondence rules for query rewriting. Finally, it uses pipes to send translated queries to acquaintances. When necessary, QP submits queries to the source database. Results Handler receives results coming from acquaintances and translates them using Correspondence Rules. If these results are for a user query, then it reports them to UI. Otherwise, it sends them backward to the node which sent respective network query. Apart from this, Results

Handler gets results coming from Wrapper, and sends them either to UI or to the network. Finally, Update Handler provides all functionality necessary for updates processing.

In order to facilitate performance study experiments, we provide some peer (called *super-peer*) with some additional functionalities. In particular, that peer can read acquaintance queries for *all* peers from a file and broadcast this file to all peers on the network. Once received this file, each peer looks for relevant for that peer acquaintance queries, reads them, and creates necessary pipe connections. If an acquaintance queries file is received when a peer has already set up acquaintance queries and pipes, then it drops "old" queries and pipes, and creates new ones, where necessary. Thus, a super-peer can change the network topology at *runtime*. This is extremely convenient for running multiple experiments on different topologies.

For the purposes of collecting experimental data, each node has an additional statistical module (not shown on Figure 1). This module accumulates various information about queries (and updates) such as: total execution time of a query, number of query result messages received per acquaintance query and the volume of the data in each message, and so on. During the lifetime of a network, each node accumulates this information. A super-peer has the possibility to collect, at any given time, statistical information from *all* nodes on the network. Then, the super-peer processes all incoming statistical messages, aggregates them, and creates a final statistical report.

The current version of the PDBMS implements Acquaintances and Coordination Rules, and partially implements Interest Groups (only one base interest group is supported) and Correspondence Rules. Amongst other things, the prototype is capable in: discovering nodes and publishing node's resources on the network; remotely monitor other nodes (e.g. check whether their pipe connections are ready); send queries to acquaintances, receive and reconcile incoming query results; discover network topology defined by paths of interdependent acquaintance queries; execute global update procedure on the network [4].

The prototype is implemented in Java and is about 6 Mbytes in size including the JXTA libraries and excluding all metadata files (e.g. source schemas, JXTA advertisements, etc). The Java Virtual Machine environment (about 40 Mbytes) is required to run the application. Thus a self-contained application package can fit in space of about 46 Mbytes, which can be placed on a flash drive. The results of the first experiments show reasonable query answering and update propagation times in small size networks (up to 20 nodes). For the experiments we created various source databases with several thousand of tuples at each node, with different degrees of the overlapping of data at different nodes.

The combination of database and P2P technologies has already received a lot of attention, see for instance [5], [6], [7], [8]. Among many other things (see [9], [2] for a detailed discussion of the related work) our solution considers a new dimension for P2P databases – mobility, where PDBMS, database, or both, can be mobile ([9], in particular, provides the vision of our approach).

II. Preliminary requirements

For the demo we will require:

- 3 computers (2 of them can be ours). One (or two) of them should have Linux as the operating system, and remaining two (or one) should have the Windows operating system;
- Wireless should be available in order to demonstrate networking platform independence property;
- On the machines provided by the organizers, a DBMS (preferably MySQL) should be installed.

III. Script

Our proposal for the demonstration of PDBMS is the following:

- Run a set of DB peers (from 6 to 10) distributed over 2-3 machines (see Figure 3). The machines may have different system and networking platforms. Some peers (2-3) will be running from flash drives.
- Demonstrate how different peers discover other peers on the network (Figure 4);
- Nodes retrieves acquaintance queries from a file, and set up necessary pipe connections (Figure 5);
- Various queries are submitted (one example is on Figure 6) from different locations, query results are reported (Figure 7);
- Show how intermediate nodes (including those running without database) process queries and query results (Figure 9);
- Demonstrate how different queries at the same node, or the same query at different nodes impose different "views" on the network, i.e. different nodes are involved for answering and different acquaintance queries are used for propagation (Figure 8);
- Demonstrate the location, system and networking platform independence by plugging in a flash drive, containing a PDBMS, to different machines. We will also demonstrate that a change of the machine does not effect query answering;
- Demonstrate how the network topology can be re-configured at *runtime* and what effect it makes on query answering;
- A short demonstration of the topology discovery (Figure 11) and global update (Figure 10) algorithms;
- Demonstrate how the query propagation algorithm actually works and show how it guarantees termination in the presence of loops in the topology;
- Demonstrate how nodes collect statistical information, how this information is collected and aggregated to a network statistical report (Figure 12).

References

[1] JXTA project, see http://www.jxta.org.
[2] F. Giunchiglia and I. Zaihrayeu, "Implementing database coordination in p2p networks," *DIT technical report # DIT-03-035, the University of Trento, Italy*, November 2003.
[3] A. Halevy, "Answering queries using views: a survey," *VLDB Journal*, 2001.
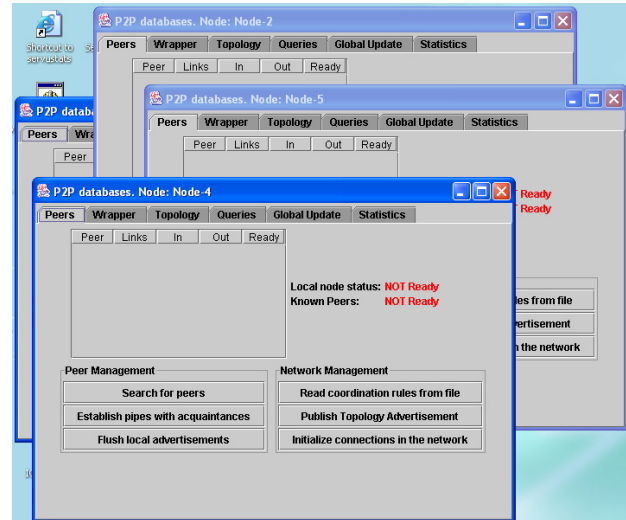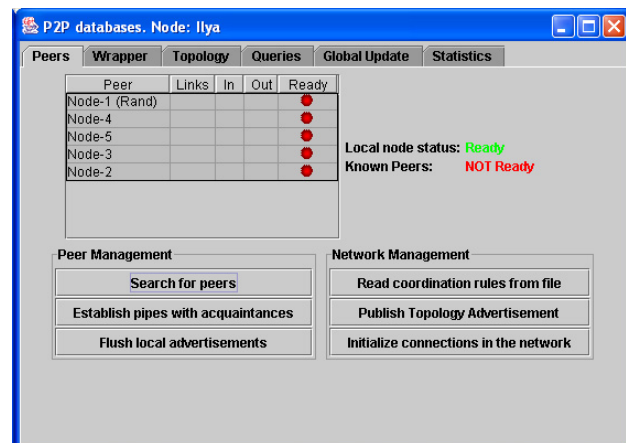
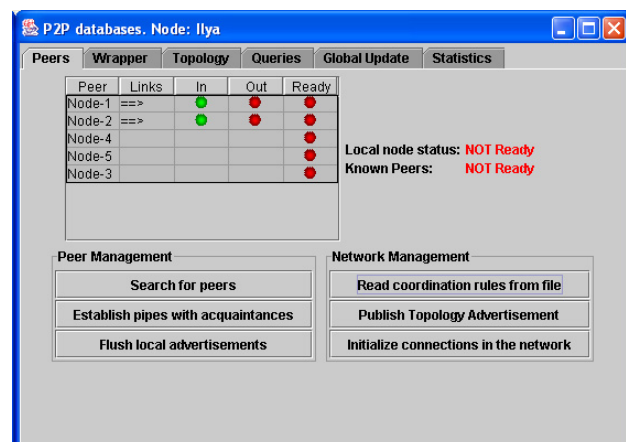Fig. 3. Running nodes

Fig. 4. Discovery of peers on the network

Fig. 5. Setting up acquaintance queries and pipes

Fig. 6.   User query

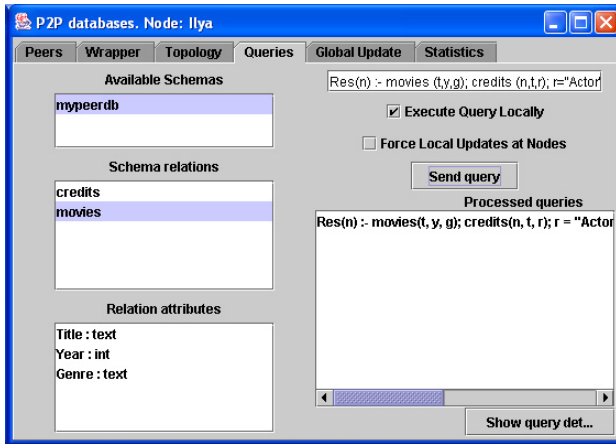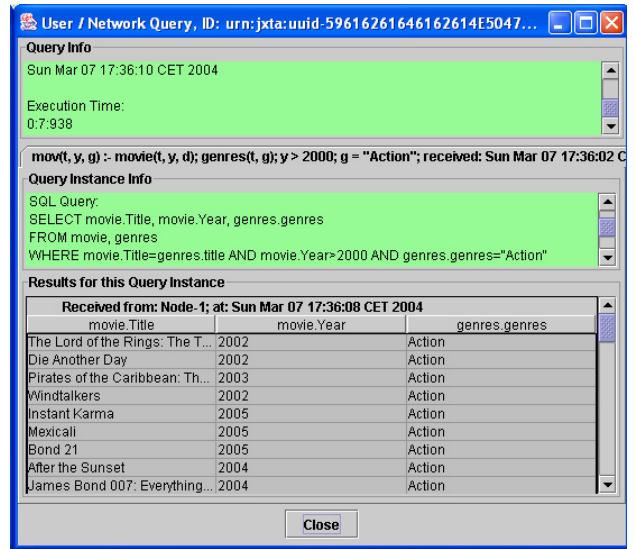
Fig. 7.   Query results

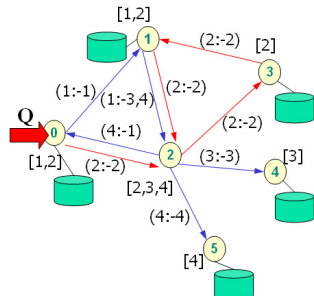Fig. 8.   Network view for a query at a node

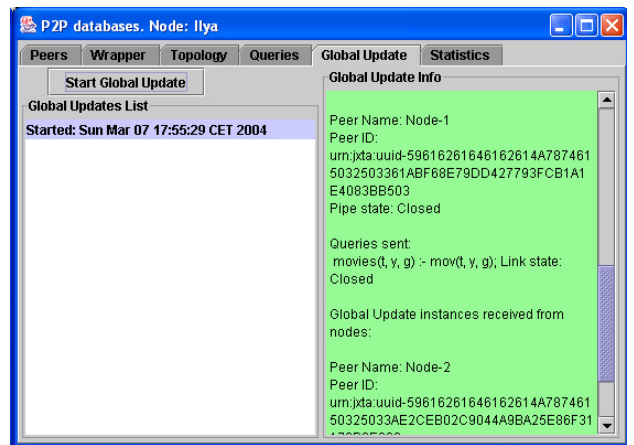Fig. 9.   Processing of a query at an intermediate node

Fig. 10.   Global Update processing

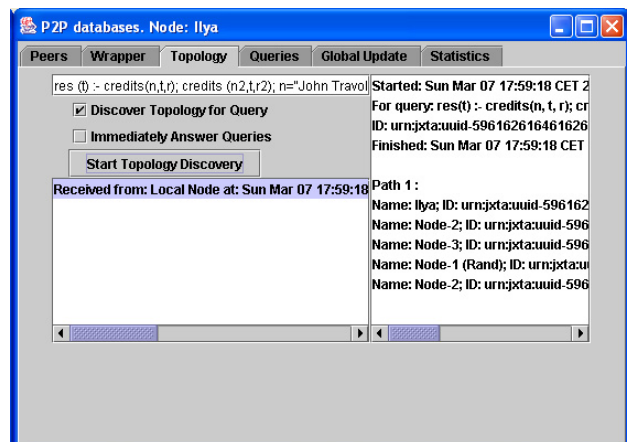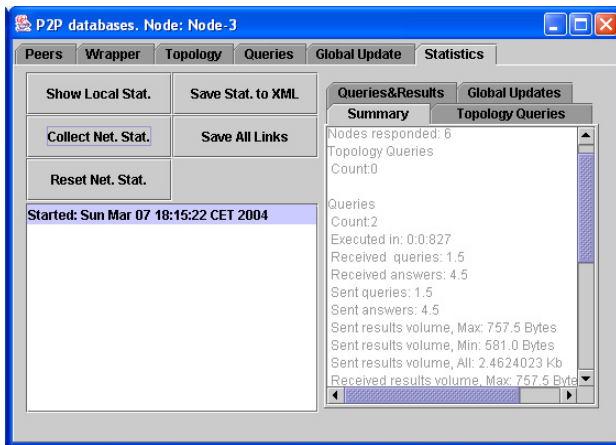
Fig. 11.   Topology discovery for a query

Fig. 12.   Network statistics

[4] E. Franconi, G. Kuper, A. Lopatenko, and I. Zaihrayeu, "A distributed algorithm for robust data sharing and updates in p2p database networks," *Proceedings of the P2P&DB international workshop, Heraklion - Crete, Greece*, March 2004.

[5] A. Kementsietsidis, M. Arenas, and R. Miller, "Data mapping in peer-to-peer systems," *ICDE*, 2003.

[6] S. Gribble, A. Halevy, Z. Ives, M. Rodrig, and D. Suciu, "What can databases do for peer-to-peer?" *WebDB, Workshop on Databases and the Web*, June 2001.

[7] A. Halevy, Z. Ives, P. Mork, and I. Tatarinov, "Piazza: Data management infrastructure for semantic web applications," 2003. [Online]. Available: citeseer.nj.nec.com/halevy03piazza.html

[8] W. Ng, B. Ooi, K. Tan, and A. Zhou, "Peerdb: A p2p-based system for distributed data sharing," *ICDE*, 2003.

[9] F. Giunchiglia and I. Zaihrayeu, "Making peer databases interact - a vision for an architecture supporting data coordination," *6th International Workshop on Cooperative Information Agents (CIA-2002), Madrid, Spain, September 18 -20*, 2002.