P2P XQuery and the StreetTiVo application

Ying Zhang
Centrum voor Wiskunde en Informatica
P.O.Box 94079, 1090 GB
Amsterdam, the Netherlands
Y.Zhang@cwi.nl

Peter Boncz
Centrum voor Wiskunde en Informatica
P.O.Box 94079, 1090 GB
Amsterdam, the Netherlands
P.Boncz@cwi.nl

1. INTRODUCTION

In the AmbientDB [2] project, we are building MonetDB/XQuery*, an open-source XML DBMS (XDBMS) with support for distributed querying and P2P services. Our work is motivated by the hypothesis that P2P is a disruptive paradigm that should change the nature of database technology. Most of the existing distributed DBMS technologies were developed to be used in (small-scale) local-area networks (LAN). Those technologies usually assume that *(i)* there is a central controller and/or peers have complete knowledge of the whole system, *(ii)* peers are uniform and highly available, *(iii)* placement of data happens in a controlled way and is rarely changed and *(iv)* a global database schema is used. Peer-to-Peer (P2P) networks have led the distributed DBMS research to reconsider existing technologies in such a new environment, where *(i)* systems have decentralized architectures, *(ii)* peers join or leave the network at any time, *(iii)* placement of data is out of the system's control and it changes frequently, *(iv)* each peer can have its local database schema (or no schema at all), and *(v)* data owned by the peers are often incomplete, overlapping and even conflicting.

Challenges. While the P2P database concept has generated a research niche, the concept has not yet been widely recognized as relevant. A first problem is that P2P database technology is understood by different researchers to mean different things, and there is no "role model" system (like what System-R was for the RDBMS) as an orientation point for the community. Secondly, most proposed techniques (e.g. P2P query processing algorithms) are evaluated in simulations whose results are hard to extrapolate to behavior in real-world circumstances. A third and related problem is that so far no "killer applications" for P2P database technology have been recognized (in contrast to P2P systems – of which various mostly file-downloading systems have found a large user audience).

Strategy. Our strategy for advancing the state-of-the-art is to incrementally develop a working P2P database prototype as a test-bed for our research and to work on applications that benefit from P2P database technologies. This strategy requires – besides research effort – a large investment in prototype engineering. We are glad to be able to build on MonetDB/XQuery [1], an open source XDBMS based on purely relational query processing that supports XQuery and the XQuery Update Facility (XUF). The choice for XML as a data model – and web standards in general – eases many aspects of distributed data management (i.e. the XML data format is platform independent, and there is ubiquitous support for URIs and specifically HTTP networking, that we use for data and query transport). We obtain P2P XDBMS functionality by *orthogonally* extending XQuery with support for *(i)* distributed querying (XRPC) and *(ii)* P2P services.

StreetTiVo is a showcase application being developed by the Dutch national research project MultimediaN, that unites multimedia and database researchers in various academic and industrial research institutes. The StreetTiVo application is a plug-in for so-called Home Theater PCs (MythTV and Windows Media Center Edition), which one can consider programmable digital video recorders. The StreetTiVo plug-in enables real-time content-based video retrieval and meta data generation, by distributing compute-intensive video analysis over multiple peers that recorded the same TV program. This application involves distributed collaborator discovery, work coordination, and result exchange in a volatile WAN environment (but not video file exchange – it is strictly legal). We think that deploying ready-to-run P2P data management technology enables quick development of this application.

2. XRPC AND P2P

We proposed XRPC [5], a small XQuery language extension that adds the concept of RPC, by allowing remote invocation of XQuery user-defined functions. The XRPC syntax is similar to XQueryD [3]:

$$\text{``execute at''} \ \text{``\{''} \ Expr \ \text{``\}''} \ \text{``\{''} \ FunApp \ (\ ParamList \ ) \ \text{``\}''}$$

where *Expr* is an XQuery `xs:string` expression that specifies the URI of the peer on which *FunApp* is to be executed. This URI should start with the new `xrpc://` naming scheme. The network protocol of XRPC is open and based on SOAP (i.e. XML messages) over HTTP. XRPC parameters are passed *by-value*, that is, node identity is not preserved across calls; actual XML node parameters refer to the SOAP request message (viz. return values refer to the response message).

In MonetDB/XQuery*, each peer runs both a Distributed Hash Table daemon [4], as well as a database process. An additional naming scheme `dht://` is used to to identify search keys as URIs. Thus, one can *logically* address nodes with a key in XRPC
execute at calls rather than physically with a node name or IP address. The DHT provides salient features such as robustness against

peer failure, automatic data replication, and allows key search queries in log(N) network hops.

Example. One of the features provided by StreetTiVo is Automatic Speech Recognition (ASR) on TV programs. The collaborative indexing phase and result collection is done using MonetDB/XQuery*as well, but this example focuses on querying the results. The list of peers that have indexed a particular TV program can be obtained through the DHT, by retrieving an XML list of node names that is stored under the TV program name (as key). Querying the DHT is done below using a `tivo` module that provides a `getPeers()` function. Each peer can then be queried using the `searchKeywords` function that gets two parameters. The first parameter `pid` identifies the TV program in which text fragment should be searched. The second parameter `keywords` is a list of search keywords. Note that both `getPeers()` and `searchKeywords()` are standard XQuery functioned defined by a user in a XQuery module.

Suppose a StreetTiVo user wants to search in the today's newscast for text fragments that were about the earthquake in Hawaii, but he/she did not record the newscast. Then the search request needs to be send to other StreetTiVo peers that have recorded the newscast. Assume the newscast's program ID is "newscast123", this scenario can be implemented by the following pseudo-code:

```
let $peers := execute at { "dht://dht_id/newscast123" }
    execute at { "dht://dht_id/newscast123" } { tivo:getPeers() }
for $p in $peers
  execute at { $p }
          { tivo:searchKeywords("newscast123",
                                ("earthquake", "hawaii")) }
```

3. CONCLUSION

MonetDB/XQuery*is a fully functional publicly available XML DBMS that has been extended with distributed and P2P data management functionality. [1] Our (minimal) XQuery language extension XRPC adds the concept of RPC to XQuery, and we outlined our approach to include the services offered by diverse P2P network structures (such as DHTs), in a way that avoids any further intrusion in the XQuery language and semantics. We also discussed the StreetTiVo application were MonetDB/XQuery*is being used for data management in a large P2P environment. new construct called XRPC.

4. REFERENCES

[1] P. Boncz, T. Grust, M. vKeulen, S. Manegold, J. Rittinger, and J. Teubner. MonetDB/XQuery: A Fast XQuery Processor Powered by a RDBMS. In *SIGMOD*, 2006.

[2] P. A. Boncz. AmbientDB: P2P Database Technology for Ambient Intelligent Multimedia Applications. *ERCIM News*, (55), 2003.

[3] C. Re, J. Brinkley, K. Hinshaw, and D. Suciu. Distributed XQuery. In *IIWeb*, 2004.

[4] S. C. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz. Handling Churn in a DHT. In *USENIX Annual Technical Conference, General Track*, 2004.

[5] Y. Zhang and P. A. Boncz. Loop-Lifted XQuery RPC with Deterministic Updates. Technical Report INS-E0607, CWI, Amsterdam, The Netherlands, 2006.

---

[1] MonetDB/XQuery can be obtained from http://monetdb.cwi.nl/XQuery.