Working Group Report on Managing and Integrating Data in P2P Databases

Peter A. Boncz
CWI, The Netherlands

Angela Bonifati *
National Research Council, Italy

Peter Janacik
University of Padeborn, Germany

Birgitta König-Ries
Jena University, Germany

Arantza Illarramendi
Basque Country University, Spain

Wolfgang Lehner
TU Dresden, Germany

Pedro J. Marròn
University of Stuttgart, Germany

Wolfgang May
Göttingen University, Germany

Aris Ouksel
University of Illinois at Chicago, USA

Kay Römer
ETH Zurich, Switzerland

Brahmananda Sapkota
DERI Research Center, Ireland

Kai-Uwe Sattler
TU Ilmenau, Germany

Heinz Schweppe
Freie Universitaet Berlin, Germany

Rita Steinmetz
University of Padeborn, Germany

Can Türker
ETH Zurich, Switzerland

ABSTRACT

In this report, to our best recollection, we provide a summary of the working group results at the Dagstuhl Seminar nr.6431 on 'Scalable Data Management in Evolving Networks', held on October 23 − 27 in Dagstuhl (Germany).

1. PEER-TO-PEER DATABASES: LET US FIRST PROVIDE A DEFINITION

We started the discussion by trying to identify a correct definition for a peer-to-peer (P2P) database. Such a database has a peer-to-peer network as back-end and, independently of the nature and topology of the network (e.g. structured, unstructured, hierarchical etc.), assumes that there is a collection of peers, each one of which is willing to share data and resources. Such peers are also autonomous and their knowledge boundaries are limited to their own local repository. Therefore, the peers never have knowledge of the global status of the network. As an example, they cannot assume to know their neighbors' data at a given time. Coming back to our definition, we can formulate it as follows: *A P2P database is a collection of peers, that offer transparent access to one or more data management services.* As this definition will become much clearer in the remainder, we recall that, for instance, an external service using a P2P database can access the latter and use its data management functionality while actually ignoring that the underlying infrastructure is a P2P network.

An important aspect that should be taken into account when talking about a P2P database is how this database can be distinguished from a federated or distributed database. We have discussed some differences and similarities. For instance, in a distributed database there is one logical schema, whereas in a P2P database no mediated schema exists. A distributed database has a well-defined fragmentation and data allocation model, whereas a P2P database cannot assume such a model a priori. Moreover, consistency guarantees are more relaxed in a P2P database. Finally, a P2P database can be thought as a collection of nodes with different functionalities, such as storage nodes, routing nodes etc. whereas such a distinction does not exist in a distributed database. In a federated database, there is always a database administrator, whereas in a P2P such administration is not required. Moreover, a federated database [9] as a general term can be considered as a collection of databases, each of which can be a distributed database or, recursively, another federated database. In a P2P database, the smallest granularity is a peer, although we can assume there exist spheres of cooperation among peers [3].

2. DESIGN DIMENSIONS OF PEER-TO-PEER DATABASES

When trying to come up with a definition, we did realize that the first issue to solve is that of defining suitable design dimensions for a P2P database. We could yield a listing of them as follows:

*Working Group Chair.

Dagstuhl Seminar Proceedings 06431
Scalable Data Management in Evolving Networks
http://drops.dagstuhl.de/opus/volltexte/2007/950

- **Topology**: this dimension has to do with the nature of the network; when designing a P2P database, one has to decide which underlying network topology to adopt, among structured DHT-based networks, e.g. CAN and Chord, or super-peer networks, e.g. Kazaa, or unstructured flooding-based P2P networks, e.g. Gnutella.

- **Autonomy**: this dimension again depends on the network; thus, for instance, peers in DHT-based networks are autonomous up to the sharing of a distributed hash function, whereas peers in super-peer networks are autonomous up to the existence of a super-peer to coordinate them.

- **Data Availability**: a peer may decide to share all the data in its repository or part of it. Thus, we can presume there is a degree of data ownership, going from 'share all' to 'share nothing'. In the latter case, one can think that the peer does not have to participate to the network at all, but this is not always true: the peer can still contribute services or other resources rather than data. Of course, the data availability of the entire network is affected by the single decision of a peer. Thus, the maximum (minimum, resp.) data availability is reached when the peers share all (none of) their data.

- **Data Ownership**: this dimension is strictly related with the previous one; we have to think about the degree of ownership that the peer wants to have when sharing its data. For instance, a peer may decide that the data it is sharing with other peers can be 'property' of the network, i.e. the network can decide on the data, e.g. on the location (which peer is responsible for storing this data), apply changes on it, remove it etc. By opposite, the peer still maintains the 'access rights' on its data. This dimension thus varies in between 'peer being the owner' to 'network being the owner'.

- **Data Replication**: this third dimension concerning the peer data, has to do with the decision taken at the level of a peer, to replicate or not its data beyond its local repository. Thus, a maximum level of replication holds whenever the peer allows the other peers to copy its data in their replicas, whereas a minimum level of replication would only allow to replicate if a given failure happens (e.g. network disruption).

- **Query capabilities**: this dimension describes the query functionalities the network is enabled to. These can vary in between key-lookup-based queries, boolean queries or more sophisticated queries, such as complete SQL or XQuery statements.

- **Update Policy**: this dimension describes the level of updatability which is reserved to peers data and metadata; in particular, the updates are always possible on local data and on all data for which the network is the owner. Also, local repository changes need to be propagated to global indexes, wherever these are present.

- **Homogeneous vs. Heterogeneous Network Content**: content-centric networks are characterized by varied data schemes. In such networks, the peers can own data that has the same schema as other peers or data that has a rather different schema. If one schema is employed, the network is considered homogeneous, otherwise it is heterogeneous.

- **Resource, Data and Service Discovery**: this dimension describes the discovery mechanisms for resource, data and services. This may depend on the network topology. For instance, all data is indexed through a DHT in structured networks, thus new data entering the network just need a key in the DHT. By opposite, in unstructured networks, a simple message propagation mechanism guarantees that peers get to know newly entered data. Of course, one can decide which kind of discovery is needed in a P2P database and choose among the alternatives above.

- **Possible Guarantees**: the last dimension is about the consistency and atomicity of the network operations. It seems that the concept of transactions as we know it in centralized databases does not directly apply to P2P databases. The policy 'all or nothing' for transactions would not be acceptable in P2P networks where partial results are also sound. Thus, new transaction protocols need to be defined for such kinds of databases and this opens up new research challenges.

The previous list, while not pretending to be exhaustive, offers a clear understanding of what features have to be considered while designing a P2P database. Of course, other dimensions can be added and the existing ones can be further refined, but this is beyond the scope of this report.

3. REQUIRED FUNCTIONALITIES

Among the database functionalities, we could list the following ones:

- **Data Model**: the data model chosen within a given P2P database is obviously relevant; this data model can be XML, relational, O-O, RDF or a combination of those etc.

- **Query language**: depending on the data model, one can choose the corresponding query language, being SQL, XQuery, SPARQL, IR-based etc.

- **Database Interface**: for the user and administrator, this database interface has to specify what to do and how to do it; an interesting issue is to build this interface in such a way that the underlying network is somehow transparent.

- **Resource and Service Discovery**: provides a discovery functionality that defines the usage of the network.

- **Distributed Transactions**: provides a functionality that offers guarantees on both data and operations on data.

4. CLASSES OF APPLICATIONS

There are several examples of applications for these new kinds of databases. Here we first give a classification of application types, and then list some related projects we have been working on.

Cross-organizational Workflows: each node of such workflows is a peer in the network; each node only knows about the next task that needs to be performed and does not know about the entire workflow model; tasks can be replicated among peers and distributed based on some properties, such as semantics, observed performance and quality of service etc.

Scientific Repositories: each peer holds scientific data, which is of interest to the others; some of these data need to be reconciliated or integrated with other external data (e.g. protein sequences from different labs); modifications are not relevant, whereas additions are important; transactions are less important than for frequently updated networks;

Collaborative multimedia indexes: such networks host image data which needs to be annotated in order to ensure that searches take place; thus indexing here is crucial to guarantee the efficiency of the discovery process; additions/deletions are relevant and need to be reflected by the index; such operations can be executed in parallel, without being in conflict. Thus strict transaction protocols are not needed;

Web archives: the idea of Web Archives is borrowed from A. Tanenbaum; a centralized version of a Web archive already exists under the name of `Archive.org`; a P2P version of a Web archive would have to handle large volumes of historical web data, whereas the importance of local updates would turn to be negligible; in particular, interesting queries restrict to those across snapshots (e.g. aggregation and datawarehouse-like queries), which are transparent to local updates.

4.1 Experience from the participants

Finally, we have discussed our experience with P2P databases within some projects (in alphabetical order) we are currently running in our own institutions:

AmbientDB [2]: AmbientBD is DHT-based and XML-based, uses XQuery as query language, data placement is left to the individual peers.

Diane [5]: Diane is a service-oriented architecture that uses a declarative request language, and implements a semantic overlay network.

HePToX [1]: HePToX is a P2P matching tool for data integration, uses a logic-based language for mappings among peers, and a subset of XQuery as query language.

Osiris [8]: Osiris is a service-oriented P2P architecture, uses a declarative request language, and enables a P2P execution of workflow processes.

Scopes [6]: Scopes formalizes the semantic reconciliation approach of a typical human integrator, by building contexts of information.

TripCom [7]: TripCom is a semantic communication infrastructure that achieves machine-to-machine interaction at Web scale, by relying on the definition of triple spaces of information.

UniStore [4]: UniStore is DHT-based, uses a Universal Relational Model, and a subset of SQL as query language; it realizes the concept of 'public data management'.

5. FINAL REMARKS AND CONCLUSIONS

The P2P paradigm is a promising approach for distributed data management, particularly in scenarios where scalability is a major issue or where a central authority/coordinator is not a viable solution. P2P data management has several dimensions affecting the design, the capabilities as well as the limitations of the system. In this report, we have sketched a set of important dimensions. Furthermore, based on own experiences we discussed representative application examples which show the potential of P2P databases. It turned out that there are a lot of different interpretations of the term "P2P Databases", depending on the research context. Also, the distinguishing characteristics against distributed and federated databases are not always strict. In the discussion, we strived to clarify these notions.

6. REFERENCES

[1] A. Bonifati, E.Q. Chang, T. Ho, L.V.S.Lakshmanan, and R. Pottinger. HEPTOX: Marrying XML and Heterogeneity in Your P2P Databases (demo). In *Proc. of VLDB*, 2005.

[2] W. Fontijn and P. A. Boncz. AmbientDB: P2P Data Management Middleware for Ambient Intelligence. In *Workshop on Middleware Support for Pervasive Computing (PerWare), In conjunction with the IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2004.

[3] Steven D. Gribble, Alon Y. Halevy, Zachary G. Ives, Maya Rodrig, and Dan Suciu. What Can Database Do for Peer-to-Peer? In *Proc. of WebDB*, 2001.

[4] Marcel Karnstedt, Kai-Uwe Sattler, Martin Richtarsky, Jessica Müller, Manfred Hauswirth, and Roman Schmidt. UniStore: Querying a DHT-based Universal Storage. In *ICDE*, 2007.

[5] Ulrich Küster, Birgitta König-Ries, and Michael Klein. Discovery and Mediation using DIANE Service Descriptions. In *First Workshop of the Semantic Web Service Challenge 2006 - Challenge on Automating Web Services Mediation, Choreography and Discovery*, 2006.

[6] Aris M. Ouksel. In-context peer-to-peer information filtering on the web. *SIGMOD Rec.*, 32(3), 2003.

[7] Brahmananda Sapkota, Edward Kilgarriff, and Chris Bussler. Role of Triple Space Computing in Semantic Web Services. In *APWEB*, 2006.

[8] C. Schuler, C. Türker, H.-J. Schek, R. Weber, and H. Schuldt. Scalable peer-to-peer process management. *International Journal on Business Process Integration and Management*, 1(2), 2006.

[9] Amit P. Sheth and James A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surv.*, 22(3), 1990.