Distributed Databases and Peer-to-Peer Databases: Past and Present

Angela Bonifati
Icar CNR, Italian National Research Council
Via P. Bucci 41C
I-87036 Rende, Italy
bonifati@icar.cnr.it

Panos K. Chrysanthis
Computer Science Dept.
University of Pittsburgh
Pittsburgh, PA 15260, USA
panos@cs.pitt.edu

Aris M. Ouksel
Dept. of Information and Decision Sciences
University of Illinois at Chicago
Chicago, IL 60607-7124, USA
aris@uic.edu

Kai-Uwe Sattler
Faculty of Computer Science and Automation
TU Ilmenau
D-98684 Ilmenau, Germany
kus@tu-ilmenau.de

ABSTRACT

The need for large-scale data sharing between autonomous and possibly heterogeneous decentralized systems on the Web gave rise to the concept of P2P database systems. Decentralized databases are, however, not new. Whereas a definition for a P2P database system can be readily provided, a comparison with the more established decentralized models, commonly referred to as distributed, federated and multi-databases, is more likely to provide a better insight to this new P2P data management technology. Thus, in the paper, by distinguishing between db-centric and P2P-centric features, we examine features common to these database systems as well as other ad-hoc features that solely characterize P2P databases. We also provide a non-exhaustive taxonomy of the most prominent research efforts toward the realization of full-fledged P2P databases.

1. INTRODUCTION

Content-sharing systems on the Web have renewed interest in the design and deployment of decentralized database management systems. Unlike the early nineties decentralized infrastructures, which were realized as federated or multi-database systems involving a relatively small handful of remote databases, current ones are conceived as large-scale, loosely-coupled peer-to-peer (P2P) systems.

The P2P paradigm offers an interesting alternative to existing information system infrastructures. The most important features are: (1) *scalability* in terms of the number of nodes and distribution, (2) *direct access* to data at the source which guarantees freshness in contrast to centralized repositories, (3) *robustness* and *resilience* against attacks and churn by exploiting self organization principles, and (4) *simplified deployment* because resources (nodes) from the "edge" of the Internet can be used and no special infrastructure is required to join the network (e.g. a new data repository can be added to a P2P network without any particular administrative task or declaration of adherence to a common schema).

A *P2P database system* (PDBS) is conceived as a collection of autonomous local repositories which interact (e.g., establish correspondences or exchange query and update requests) in a peer-to-peer style. That is, local repositories are autonomous peers with equal rights and are linked to only a small number of neighbors. Furthermore, the term 'repository' indicates that a single peer might be a collection of files rather than a full-fledged DBS with established data management functionality. Such repositories may not exhibit a common interface, but they can still provide a DBS-like access functionality, as it happens for Web databases

Clearly, several characteristics of PDBS and past decentralized systems, including autonomy and heterogeneity, are common to the two approaches. This observation underscores the necessity of a detail comparison among the two approaches and the identification of their essential characteristics that would clarify the definition of PDBS and make it distinct. Toward this, this paper compares modern distributed data paradigms, such as P2P database systems [15], with distributed database systems [8, 31] and cooperative multiple database systems, such as federated databases [4] and multi-databases [7]. The goal is to clarify some of the essential differences and similarities from a data management point of view. Hence, we distinguish between *DB-centric* features from *P2P-centric* ones in our comparison. Given that PDBSs are currently on an evolutionary path, such a distinction would help to identify which characteristic, for example, distribution and federation, might be relevant in a P2P environment. Another contribution of this paper is a taxonomy of a non-exhaustive list of existing P2P and dis-

Sigmod Record Vol. Nr. 2008

tributed database prototypes that are compared based on the above features.

This paper is organized as follows. Section 2 reviews established decentralized database systems and compares their data integration architectures to that of PDBS. Section 3 highlights the characteristics of P2P systems that make them similar or different with respect to distributed and federated database systems. Section 4 provides a taxonomy of existing P2P prototypes with respect to the above features, and discusses future research directions. Section 5 concludes the discussion by giving a summary and a few enlightening thoughts.

2. TWO DATA INTEGRATION ARCHITECTURES

A database system (DBS) is a software that manages one or more databases. A distributed database system (DDBS) is a software that manages one or more logically-related databases, spanning a network. Both a federated database system (FDBS) and a multi-database system (MDBS) are collections of pre-existing DBSs in which operations can be applied to multiple component DBSs in a coordinated manner. The key distinction between FDBSs and MDBSs is their methods for integrating the component DBSs and their assumptions about the autonomy of these components. In both FDBSs and MDBSs, component DBSs are typically heterogeneous, for example, they use different data models or formats. To deal with such heterogeneity, FDBSs adopt more traditional DDBS techniques that rely on a single global federated schema. In contrast, multiple federated schemas may coexist in MDBSs between the different cooperating component DBSs, allowing thus partial and controlled data sharing. In addition, any of the component DBS can itself be a DDBS in both FDBSs and MDBSs.

So, a common characteristic of all past decentralized, multiple database systems, namely DDBS, FDBS and MDBS, consist of component databases which are DBSs with a well-defined database schema. As a result, these distributed systems support data access across component DBSs by means of some form of a common schema that integrates the local component DBS schemas. For DDBSs, the common schema is defined a-priori. On the other hand, for FDBSs and MDBSs, a common federated schema is the result of an agreement between the participants DBSs as shown in the multi-layer data architecture in Figure 1(a).

The federated schema based architecture consists of four layers as shown in Figure 1(a): *(i)* a *local schema*, expressed in the local data model schema; *(ii)* a *local component schema*, which is possibly a translation of the data model of the local DBS into a canonical model; *(iii)* a *local export schema*, which contains those elements of the component schema that the local DBS is willing to share with others, for instance by defining access control policies; and finally, *(iv)* a *federated schema*, which is a *global federated schema* in FDBS and *application-oriented federated schema* in MDBS. A federated schema is the actual global schema that contains information on distribution and allocation of internal export schemas. In MDBS, different federated schemas may coexist to support data sharing for different applications. In both FDBS and MDBS, mappings must be defined between the local schemas and the global federated schemas. Such mappings express the correspondences between elements in the local schemas and elements in the global schemas in Local-As-View(LAV)/Global-As-View(GAV) style [25].

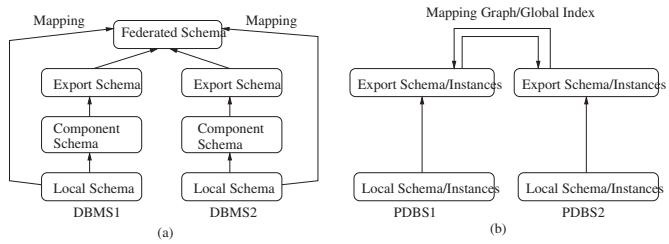

Figure 1: Data Integration Architecture of (a) FDBS/MDBS and (b) PDBS.

The commonly known P2P applications are basically file sharing systems [14, 23], where the notion of a global mediated schema is irrelevant. However, several recent efforts in the database community have been aimed at extending these systems to full-fledged peer-based data management systems [11, 13, 17, 27]. The main data integration and interoperability idea in peer data management is to avoid a global schema by providing mappings between pairs of information sources. Mappings between all pairs are not necessary. It is sufficient that the graph representing the available mappings be connected. Mappings between two sources are then obtained by composing the pairwise mappings such that there is a path connecting the two sources [6, 17, 36], and a much earlier proposal where integration is conceived as binary between two sources, partial, and query-dependent [30]. The query circumscribes the integration context.

Figure 1(b) illustrates a P2P architecture for data integration. Observe that a component schema does not exist at each peer (compared to FDBS/MDBS architecture), since a common mediated schema is less likely in a P2P architecture. Basically, an *export schema* contains only the elements of the local schema that a peer wants to share with the outside world. One can also assume that a local schema does not exist at all, and part of the actual instance is exposed to the outside. Note that *instances* and *schemas* can be used interchangeably in PDBS and the latter are less relevant than in FDBSs and MDBSs, where the availability of schemas is mandatory.

Most importantly, peers autonomously decide the exchanged part with other peers in data integration scenarios [11, 13, 30], by means of mapping rules (*source-to-target dependencies* that connect their local schemas). Note that the mapping rules are not necessarily symmetric. Thus, the top-level layer in Figure 1(b) is what we call the mapping graph or global index. The global index may be either centralized or distributed.

3. P2P AND ESTABLISHED DISTRIBUTED DATABASE SYSTEMS: DIFFERENCES AND COMMONALITIES

3.1 Distribution, Autonomy, Heterogeneity

All decentralized database architectures – P2P or federated, distributed or multi-databases – share a set of common features. The classification given in [31] illustrates the DBS implementation alternatives. As Figure 2 illustrates, the various DBS categories can be characterized along the following

three dimensions:

(i) *Distribution*, ranging from a centralized architecture (no distribution) (D_0) to a client-server distribution (moderate distribution) (D_1) to a peer-to-peer (or to full-scale distribution) (D_2);

(ii) *Autonomy*, ranging from zero autonomy (tight integration)(A_0), semi-autonomy (loose integration)(A_1) to full autonomy or total isolation (A_2);

(iii) *Heterogeneity*, ranging from zero heterogeneity (homogeneous systems)(H_0) to full heterogeneity (H_1).

Thus the set of possible database systems is characterized by the Cartesian product $\{D_0,D_1,D_2\} \times \{A_0,A_1,A_2\} \times \{H_0,H_1\}$. For instance, element (A_0, D_1, H_0) identifies properties of *distributed database systems*, i.e., no heterogeneity and no autonomy, as discussed in the introduction. Elements (A_1, D_0, H_1) and (A_1, D_1, H_1) capture properties of *heterogeneous federated database systems* and *distributed heterogeneous federated database systems*, respectively. These latter systems are instances of the class of FDBSs. They are semi-autonomous in the sense that they may act independently but may still cooperate to selectively share data.

Multi-databases and *distributed multi-databases* are captured by (A_2, D_1, H_1) and (A_2, D_2, H_1), respectively. These systems belong to the class of MDBSs: they are highly decentralized, heterogeneous and totally independent of one another, in the sense that each DBS component is not aware of the existence of all other DBSs and their databases.

Below we will further discuss the concept of heterogeneity adopted for such database systems. However, as opposed to heterogeneity, the dimensions of autonomy and distribution need to be refined in order to better classify the modern PDBSs.

A cursory observation might classify PDBSs as another instance of MDBSs. However, a closer look at their data integration architecture reveals that these two systems support completely different data access methods. Specifically, MDBSs support a query interface on top of a multi-database layer. A query, referred to as *global request*, is issued through this interface. The query is then shipped to the component databases as Figure 3 shows. As the query reaches the component databases, it is translated into a local request. Although a user is seldom aware of the presence of the underlying component databases it always receives back a complete answer.

On the other hand, in a PDBS, a query is submitted to a local peer and it may or may not be forwarded to the subsequent peers in its original form or in a form modified by the visited peers. The forwarding depends on the mapping graph. Thus, no global request is submitted to all peers with the requirement that a response is expected. However, a complete response is not guaranteed. Further, unlike DBS components of MDBSs, a peer in a PDBS is free to join or leave the network at will and has no obligation to perform administration tasks. Thus, a peer in PDBS exhibits a much higher degree of autonomy than DBS components of MDBSs. To capture this distinction, we introduce a new point A_{P2P} and redefine the meaning of A_2 along the autonomy dimension in Figure 2. This new point A_{P2P} now reflects *full autonomy* or *total isolation* as opposed to A_2, which now reflects *quasi-full autonomy*.

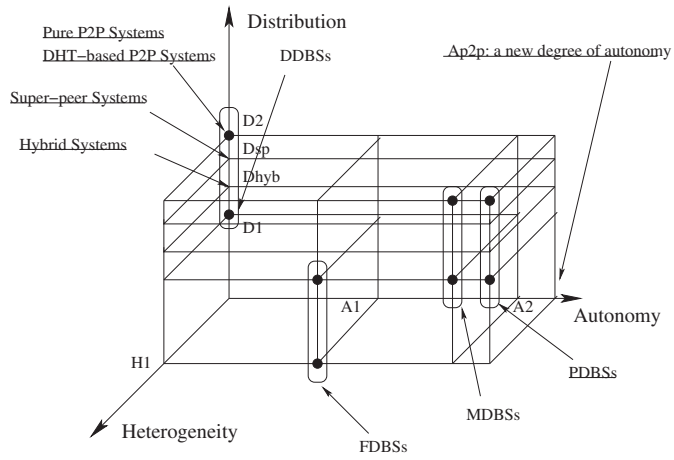

Figure 2: DBS Implementation Alternatives with Modern P2P Architectures.

The distribution dimension needs to be refined as well. Distribution in PDBSs is strongly dependent on the underlying P2P network. Existing P2P networks can be classified into three broad categories: *pure P2P*, *super-peer systems* and *hybrid systems*. Pure P2P are systems positioned at point (A_{p2p}, D_2, H_0) [1] i.e., systems in which all participants have the same functionality and do not store global indexes. Super-peer networks are networks in which a number of peers (super-peers) may have internal indexes that describe the data of other peers and other super-peers. In such super-peer systems, the communication and level of distribution is done in two phases: at the super-peers level and underneath at the peers level. Finally, hybrid systems are systems in which servers or clusters may play a role in storing global indexes. Both hybrid systems and super-peer systems may be classified somewhere between client-server (D_1) and pure P2P systems (D_2) and are denoted D_{Hyb} and D_{SP} in the figure, respectively.

The above pure P2P networks are referred to as *unstructured* networks in that restrictions are not imposed on data placement. These networks are among the early P2P networks basically used for file sharing. Recently, the so-called *structured* P2P networks have been gaining momentum. Such networks are based on DHTs (Distributed Hash Tables) in which uniform hash keys are used to enable efficient lookups. These networks use a protocol to maintain locally information about a subset of their neighbors and enable efficient routing. From a distribution perspective, structured networks can be classified at same level D_2 (in Figure 2) as the pure P2P unstructured networks. However, we shall see in the remainder that, based on other features, there are differences between PDBSs over these different P2P networks.

An often-cited reason to favor P2P solutions are their scalability and stability and self-repairing characteristics. In the case of PDBSs, this is reflected in the behavior of the mapping graph. However, these properties will only hold for some PDBSs, and most are at risk of a melt-down if the system experiences frequent membership changes, a problem known as churn. In effect, an existing path in the mapping graph may quickly disappear due to a membership change. This is

[1]Note that pure P2P distributed systems are positioned at point (A_0, D_2, H_0) according to the classification given in [31] which does not have the A_{p2p} point.

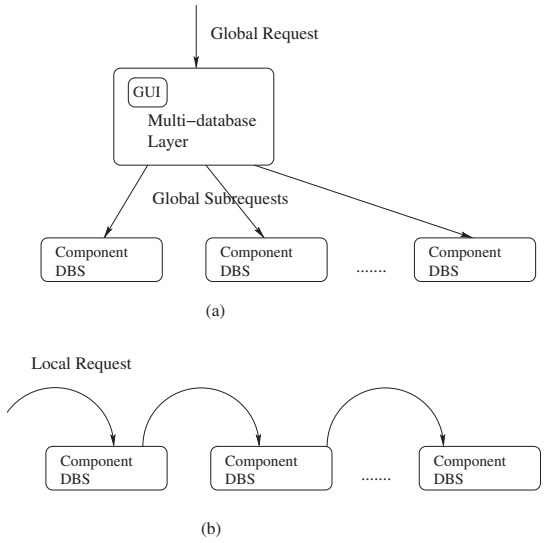

Figure 3: Simplified System Architecture of (a) a MDBS and (b) a PDBS.

a common pattern in distributed systems, where the participating nodes are at the edge of the network infrastructure, that is, machines of simple users. Query processing is based on forwarding from peer-to-peer. Its performance will depend on the length of mapping path, which can be excessive in unstructured P2P networks.

The three dimensions discussed above are considered fundamental, but they are not sufficient for a comprehensive comparison between old and new distribution paradigms, especially with respect to heterogeneity. In the following we introduce two additional dimensions, i.e., *database-centric* dimensions, which have been extensively studied in the context of classical distributed data management and *P2P-centric* dimensions originating for the P2P paradigm.

3.2 Database-centric Dimensions

One of the main goals in distributed database systems is to provide transparency while exploiting the features of a distributed environment. This comprises transparency at the level of fragmentation, replication, and transaction support. The dimensions related to these issues are the following.

Fragmentation and allocation design. DDBS allow a flexible design of fragmented relations and their placement at different sites in a top-down fashion. In contrast, data integration systems like FDBS and MDBS follow a bottom-up approach by integrating data kept at the original sites. In structured DHT-based systems, the placement of data is determined by the system hash function, i.e., the node responsible for a given data item is 'computed' by the system. Though, a DHT maintains the allocation dynamically in order to deal with leaving or joining nodes, this allocation is still system-defined. In contrast, in DDBS the allocation is defined by the user as part of the database design step. Unstructured PDBSs in their pure form (i.e., without exploiting structured indexes) act like data integration systems where each node keeps authority on its own data and thus decides on allocation autonomously.

Data independence. DDBS implement the notion of data independence, i.e., the fact that the logical model and the physical implementation are kept separate, thus allowing a set of abstractions that realize the data management tasks in a suitable way. In P2P systems, a data independence notion is also desirable [18] as data need to be independent of their physical location in the network.

Transactional support. The ability to support ACID-style transactions is a fundamental requirement in many database applications that require strong data consistency. However, for the distributed case this is only addressed in DDBS by special protocols. There are also some proposals of federated and peer-to-peer databases that support relaxed consistency criteria [32], but basically it is still an open issue if and how ACID properties can be achieved in loosely-coupled systems of autonomous nodes. Nevertheless, transaction support is required to some extent if replicas of data are maintained by a PDBS and updates are supported in a PDBS.

View on the world. The typical assumption in a classical database system is the closed world assumption meaning that all relevant facts are stored in the database and returned if requested by a query. However, this is difficult or even impossible to achieve in a PDBS where nodes are allowed to join and leave the network at any time. Thus, these systems are usually based on the open world assumption (i.e., the assumption that the data and results are incomplete) and return only certain query answers [17].

Recall and Query Services. The query capabilities are diverse in traditional DDBS, where location and fragmentation transparency are embedded in the query languages. In PDBS, instead, the query services are still limited and highly depend on the kind of network underneath. In particular, unstructured networks support keyword-based queries, whereas structured network handle both lookups [34] and range queries [16]. The query language expressiveness may still be extended in both kinds of networks. Another difference between structured and unstructured networks relies in the perfect/non-perfect recall. Under this respect, structured networks are similar to traditional distributed architectures as they achieve perfect recall (i.e., equal to 1), whereas unstructured networks do not (less than 1).

3.3 P2P-centric Dimensions

The following dimensions address special characteristics of P2P-based approaches, that are also desirable for data management in PDBS.

Degree of coupling. The degree of coupling is intended as the "awareness" of the existence of other peers. In DDBS, all nodes are known by other sites (or at least by the coordinator site) at any time, thus realizing a tightly coupled scenario. In PDBSs, peers can join or leave the network dynamically. In such case, the degree of coupling among peers is less tight, as a peer can be aware of the existence of a few neighbors, and this awareness changes over time. The degree of coupling also determines the level of self-organization. In structured P2P systems where the system controls the data placement, the ability of peers to self-organizing in a PDBS is limited. In contrast, peers in an unstructured P2P network can continuously self-organize to a cluster or a hierarchy.

Overlay Network topology. The different classes of P2P overlay networks differ mainly in their topology. Unstructured PDBS are similar to DDBS: there is no fixed topology – the overlay network is a result of the connections established between the nodes.

The next level is formed by super-peer networks where dedicated peers maintain more information about their as-

sociated peers and are interconnected with other super-peers in a predefined way (e.g., a ring or hypercube). In contrast, structured PDBS are based on a fixed topology like a hypercube [33], a ring [34], a tree [19], a binary tree [21], or a B-tree [26].

Routing strategies. This dimension is tightly connected with the topology dimension. In systems without a fixed topology where information is stored at the neighbor peers, the only choice to answer requests is flooding. However, several solutions have been proposed which are based on maintaining routing information in order to allow directed semantic routing. In contrast, structured PDBS rely on information about neighbors and usually implement some kind of greedy routing. For instance, in [34], finger tables are maintained on each peer and used to route the search toward the neighbors having an identifier closer to the search key.

Scalability. Unstructured networks differ from structured ones for what concerns scalability. Unstructured networks based on flooding are poorly scalable as the messages may flood the network quickly. Super-peer networks partially solve the problem whenever super-peers are used as proxies and flooding is only performed between those. Random walks may also be beneficial since the query is forwarded to only one peer at a time, thus significantly reducing the the network traffic. Structured networks are more scalable than unstructured ones since the queries are only routed to selected peers and can guarantee perfect recall. The goal in such networks is to achieve a less than linear increase in complexity, as the capabilities of the distributed system grow and more hosts join it.

Anonymity and Security. A feature that characterizes P2P systems is anonymity, i.e., in some applications, the origin of both requests and information should remain unknown. By routing requests through many peers and also replicating content, the identity of participants should be kept hidden. Another level of abstraction is the security measures of a PDBS, i.e., only authorized users must be granted access to privileged data. This entails the ability to authenticate users.

4. TAXONOMY OF EXISTING P2P DATABASE SYSTEMS

In this section, we review some of the proposals for PDBS and DDBS, and classify them on the basis of the features identified above. As a disclaimer, the reader must notice that this list is meant to be illustrative rather than exhaustive, thus further systems may be added to our taxonomy. In particular, we do not survey XML P2P systems, which can be found in [24], where XML data management techniques for P2P are discussed and compared. Research challenges on search and security issues and the view materialization problem for P2P databases are discussed in [10] and in [15], respectively. Finally, in this paper we do not survey P2P content distribution models, for which a comprehensive survey can be found in [3]. Our aim is instead that of putting *together* and giving a *unified view* of representative sets of present PDBS and past DDBS.

We first realized that the systems we have taken into consideration fall into three categories: *super-peer PDBS*, which embody the D_{SP} degree of distribution in Figure 2, *Structured (DHT-based) PDBS*, that correspond to D_2, and *Hybrid PDBS*, that are at D_{Hyb}. Moreover, we consider DDBS as representatives of the old architectures[2]. The compared systems are reported in Table 1.

4.1 Unstructured super-peer PDBS

Edutella [28] is a super-peer PDBS, in which super-peers are responsible for query routing in first place, and requests are only later forwarded to simple peers. The kind of queries it can handle are RDF-based top-K queries. Moreover, scalability is highly affected by the presence of super-peers and clusters of nodes. Fragmentation and transaction support are to be added, along with access control and security issues that are still unsolved.

4.2 Structured PDBS

Pier [20] is an Internet-scale query processor that can be applied to P2P file-sharing (see next subsection). It implements the *logical data independence* principle of relational databases. It does not have a persistent storage, as each item is kept alive for a 'soft-state' lifetime, after which it is discarded. This way, the ACID storage semantics of distributed databases is sacrificed. It implements the forward-progress multi-hop routing strategy, in which query processor upcalls can be used to drop redundant messages in the network. Metadata is not stored in a catalog as in distributed DBS, but computed on the fly when needed.

Galanis et al. [12] uses a DHT-based infrastructure to realize XPath searches. Structural summaries and value summaries are used to bias the search toward the correct peers. Scalability and routing protocols are the same of a DHT.

GridVine [2] is a DHT-based semantic overlay network, based on P-Grid [1]. Contrary to other DHTs, it uses an order-preserving DHT function, that allows compute prefix and range queries, while not affecting the scalability. The queries supported are RDF-based. The routing strategy is based on Semantic Gossiping, i.e. mappings on RDF schemas. UniStore [22], which is also based on P-Grid, supports similarity-based selections and joins as well as top-K and skyline queries.

4.3 Hybrid PDBS

Piazza [17] is a peer-to-peer data integration system that enables sharing heterogeneous data in a distributed and scalable fashion. Peers are related to each other by means of semantic mappings, i.e., equalities or subsumptions between query results on different peers, as well as by means of storage expressions. The topology of the network is a freely interconnected mesh of peers with semantic relationships between them. Piazza has a centralized index rather than a distributed one. This makes it more similar to a search engine than to a DHT. Nevertheless, this index is scalable with the number of attributes of the individual peers. HePToX [6] is a P2P data integration system that supports data/metadata heterogeneity and guarantees query answering by means of the mapping graph along and against the direction of mappings.

PIERSearch [27] is an hybrid solution that fuses a DHT-based search for rare items and a flooding search strategy for

[2]We limit ourselves to consider a sample of distributed databases, i.e., the ones that closely resemble modern P2P data management infrastructures. Many of the multidatabases and federated databases proposals followed the architecture given in Figure 1, where queries are posed against a global schema. We do not report them here for space reasons.

popular items, being the latter based on Gnutella [14] query processor. Being an hybrid solution, it benefits from both technologies by taking the best of them. In particular, scalability is improved by the logarithmic search in DHTs, and the routing strategy is mainly semantic, based on inverted lists.

PeerDB [29] is a full-fledged P2P data management system, that employs agents to enable an effective query processing strategy. The network consists of simple nodes (peers) and LIGLO (location independent global names lookup) servers. These servers assign unique IDs to the peers and keep trace of their current status (online/offline). The queries are formulated in SQL and the query processing is agent-aided. In particular, to ensure a secure connection among peers, a 128-bit encryption scheme is employed.

4.4 Distributed DBS

Mariposa [35] can be considered a pioneering PDBS, since it is a database distributed over a WAN network, as opposed to its predecessors that were distributed in LAN networks. Mariposa adheres to a total decentralization model, in which there is no central authoritarian administration, not even for data and query allocation. Moreover, there is no upper bound to the number of machines that can be connected and no global synchronization is demanded. These characteristics make it an early precursor of PDBS. All the distributed DBS features, and in particular, the routing strategy are reformulated in microeconomic terms.

R* [38] is a distributed database that realized a distributed query evaluation strategy, according to which a global query plan is yielded at the master site and local query plans are executed by the apprentices, i.e., local sites that decide on the local part of the computation. R* did not support replication or fragmentation, but had implemented the location transparency, as well as a resilient support for transaction management. The metadata catalog can be stored within both local and remote sites, thus guaranteeing that routing of a request is done by using the catalog entries.

SDD-1 [5] was the first distributed database, federating data module (DM) sites and transaction module (TM) sites, having separate data and transaction management functionalities. Fragments can be stored redundantly and the user is not aware of their allocation. The data necessary for a computation is saved into local workspaces, and can be retrieved by reducer programs, that assemble it on a processing site. Due to its modular design, this system was tremendously anticipating the modern distributed architectures.

5. CONCLUSIONS

The emergence of PDBS, a new type of decentralized data management system over P2P networks has raised a number of interesting questions: What DDBS or MDBS features would be adopted in PDBSs? What operations would require execution on multiple peers, and if so, how would they be handled? Which distribution and federation characteristics might be relevant in a P2P environment?

In this paper, we addressed these and similar questions by providing a comparison among past decentralized database systems and PDBSs in which DB-centric and P2P-centric features were distinguished. Using these features, we analyze a number of existing systems summarized in Table 1.

Whereas DB-centric features characterize the distributed architectures of the past, very few P2P systems realize the data independence principle, while none of them have strategies for replication and fragmentation, and, most importantly, none of them has support for transactions. We believe that these features are of utmost importance to realize full-fledged P2P database systems.

Concerning P2P-centric features, a surprising result of our analysis has been the tremendous modernity of distributed paradigms, and their anticipation of the times. Most of the lessons learned from these systems are about scalability and query routing strategies. It would be interesting to see how to apply the latter strategies to P2P databases.

A final observation is devoted to anonymity, security, and access control problems, which are still open and challenging issues in P2P data management. While these issues have been discussed for file-sharing systems [37, 10], their impact on P2P database security is yet to be investigated.

6. ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their valuable comments. Some of the ideas behind this paper have been conceived while the authors were at a panel in Dagstuhl, whose report can be found online at [9]. The authors would like to acknowledge all the participating people. Panos K. Chrysanthis was partially supported by NSF awards ITR-ANI-0325353 and IIS-0534531 and Aris N. Ouksel by NSF awards ITR-0326284, IIS-SGER-0713336 and IGERT-DGE-05449489.

7. REFERENCES

[1] K. Aberer. P-Grid: A Self-Organizing Access Structure for P2P Information Systems. In *Proc. of CoopIS*, 2001.

[2] K. Aberer, P. Cudr-Mauroux, M. Hauswirth, and T. V. Pelt. GridVine: Building Internet-Scale Semantic Overlay Networks. In *Proc. of SWC*, 2004.

[3] S. Androutsellis-Theotokis and D. Spinellis. A survey of peer-to-peer content distribution technologies. *ACM Computing Surveys*, 36(4):335–371, 2004.

[4] A.P.Sheth and J.A.Larson. Federated database systems for managing distributed, heterogeneous and autonomous databases. *ACM Computing Surveys*, 22(3):183–236, 1990.

[5] P. Bernstein, N. Goodman, E. Wong, C. Reeve, and J. Rothnie. Query Processing in a System for Distributed Databases (SDD-1). *ACM TODS*, 6(4):602–625, 1981.

[6] A. Bonifati, E. 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.

[7] A. Bouguettaya, B. Benatallah, and A. Elmagarmid. An overview of Multidatabase Systems: Past and Present. In M. R. A. Elmagarmid and A. Sheth, editors, *Management of Heterogeneous and Autonomous Database Systems*, pages 1–32, 1999.

[8] S. Ceri and G. Pelagatti. *Distributed Databases: Principles and Systems*. Mc-Graw Hill Book Company, 1984.

[9] Dagstuhl Working Group Report on Managing and Integrating Data in P2P Databases. `http://drops.dagstuhl.de/portals/index.php?semnr=06431`.

[10] N. Daswani, H. Garcia-Molina, and B. Yang. Open Problems in Data-Sharing Peer-to-Peer Systems. In *Proc. of ICDT*, 2003.

[11] A. Fuxman, P. G. Kolaitis, R. J. Miller, and W. C. Tan. Peer Data Exchange. *ACM TODS*, 31(4):1454–1498, 2006.

[12] L. Galanis, Y. Wang, S. Jeffery, and D. DeWitt. Locating Data Sources in Large Distributed Systems. In *Proc. of VLDB*, 2003.

[13] G. D. Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. On Reconciling Data Exchange, Data Integration and Peer

DB-centric features						
System	Frag. & Alloc. Design	Data Indep.	Transact. Support	View on World	Query Services	Recall
Edutella	NA	NA	NA	OWA	RDF-based, top-K	≤ 1
Galanis et al.	NA	NA	NA	OWA	XPath	1
Pier	NA	Yes	NA	OWA	Keyword	1
GridVine	NA	Yes	NA	OWA	RDF-based	1
PeerDB	NA	NA	NA	OWA	SQL	1
Piazza	NA	NA	NA	OWA	XQuery	≤ 1
PIERSearch	NA	Yes	NA	OWA	Keyword	1
Mariposa	Cost-based	Yes	Yes	CWA	SQL	1
R*	Only Alloc.	Yes	Yes	CWA	SQL	1
SDD-1	Yes	Yes	Yes	CWA	SQL	1

P2P-centric features					
System	Deg. of Coup.	Net. Topology	Routing Strat.	Scalability	An. & Sec.
Edutella	Loos. with SP	Cluster of Peers	Semantic	NA	NA
Galanis et al.	Loos. coupled	DHT	Semantic	Logarithmic	NA
Pier	Loos. coupled	DHT	Forward-progress	Logarithmic	NA
GridVine	Loos. coupled	DHT	Semantic Gossiping	Logarithmic	NA
PeerDB	Loos. with LIGLO	Cluster of Peers	Agent-assisted	NA	128-bit enc.
Piazza	Loos. with Virtual Peers	Mapping-based	Semantic	Nr. of peer attributes	NA
PIERSearch	Loos. with DHT	Gnutella-graph/DHT	Semantic	DHT for rare items	NA
Mariposa	Tightly coupled	NA	Economic-based	WAN-based	NA
R*	Tightly coupled	Master/Appr.	Catalog-based	NA	NA
SDD-1	Tightly coupled	DM/TM	Reducer-based	NA	NA

Table 1: DB-centric and P2P-centric Features of Some Peer-to-Peer and Distributed Databases.

Data Management. In *Proc. of PODS*, 2007.

[14] Gnutella homepage. `http://www.gnutella.com/`.

[15] S. D. Gribble, A. Y. Halevy, Z. G. Ives, M. Rodrig, and D. Suciu. What Can Database Do for Peer-to-Peer? In *Proc. of WebDB*, 2001.

[16] A. Gupta, D. Agrawal, and A. E. Abbadi. Approximate Range Selection Queries in Peer-to-Peer Systems. In *Proc. of CIDR*, 2003.

[17] A. Y. Halevy, Z. G. Ives, D. Suciu, and I. Tatarinov. Schema Mediation in Peer Data Management Systems. In *Proc. of ICDE*, 2003.

[18] J. M. Hellerstein. Toward Network Data Independence. *SIGMOD Record*, 32(3):34–40, 2003.

[19] K. Hildrum, J. D. Kubiatowicz, S. Rao, and B. Y. Zhao. Distributed Object Location in a Dynamic Network. In *Proc. of SPAA*, 2002.

[20] R. Huebsch, B. N. Chun, J. M. Hellerstein, B. T. Loo, P. Maniatis, T. Roscoe, S. Shenker, I. Stoica, and A. R. Yumerefendi. The Architecture of PIER: an Internet-Scale Query Processor. . In *Proc. of CIDR*, 2005.

[21] H. V. Jagadish, B. C. Ooi, and Q. H. Vu. BATON: A Balanced Tree Structure for Peer-to-Peer Networks. In *Proc. of VLDB*, 2005.

[22] M. Karnstedt, K. Sattler, M. Richtarsky, J. Müller, M. Hauswirth, R. Schmidt, and R. John. UniStore: Querying a DHT-based Universal Storage. In *Proc. ICDE 2007*, pages 1503–1504, 2007.

[23] The Kazaa Homepage. `http://www.kazaa.com`.

[24] G. Koloniari and E. Pitoura. Peer-to-peer Management of XML Data: Issues and Research Challenges. *SIGMOD Record*, 34(2):6–17, 2005.

[25] A. Y. Levy, A. Mendelzon, Y. Sagiv, and D. Srivastava. Answering Queries Using Views. In *Proc. of PODS*, 1995.

[26] P. Linga, A. Crainiceanu, J. Gehrke, and J. Shanmugasundaram. Guaranteeing Correctness and Availability in P2P Range Indices. In *Proc. of SIGMOD*, 2005.

[27] B. T. Loo, J. M. Hellerstein, R. Huebsch, S. Shenker, and I. Stoica. Enhancing P2P File-Sharing with an Internet-Scale Query Processor. In *Proc. of VLDB*, 2004.

[28] W. Nejdl, B. Wolf, C. Qu, S. Decker, M. Sintek, A. Naeve, M. Nilsson, M. Palmér, and T. Risch. EDUTELLA: a P2P Networking Infrastructure Based on RDF. In *Proc. of WWW*, 2002.

[29] W. S. Ng, B. Ooi, K. Tan, and A. Zhou. PeerDB: A P2P-based System for Distributed Data Sharing. In *Proc. of ICDE*, 2003.

[30] A. M. Ouksel and C. F. Naiman. Coordinating Context Building in Heterogeneous Information Systems. *J. Intell. Inf. Syst.*, 3(2):151–183, 1994.

[31] M. T. Ozsu and P. Valduriez. *Principles of Distributed Database Systems, Second Edition*. Prentice-Hall, 1999.

[32] K. Ramamritham and P. Chrysanthis. *Advances in Concurrency Control and Transaction Processing*. IEEE Computer Society Press, 1996.

[33] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A Scalable Content-addressable Network. In *Proc. of SIGCOMM*, 2001.

[34] I. Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications. In *In Proc. of SIGCOMM*, 2001.

[35] M. Stonebraker, P. M. Aoki, W. Litwin, A. Pfeffer, A. Sah, J. Sidell, C. Staelin, and A. Yu. Mariposa: A Wide-Area Distributed Database System. *VLDB J.*, 5(1):48–63, 1996.

[36] I. Tatarinov and A. Halevy. Efficient Query Reformulation in Peer-Data Management Systems. In *Proc. of SIGMOD*, 2004.

[37] D. Wallach. A Survey of Peer-to-Peer Security Issues. In *Proc. of ISSS*, 2002.

[38] R. Williams, D. Daniels, L. Haas, G. Lapis, L. P. Ng, R. Obermarck, P. Selinger, A. Walker, P. Wilms, and R. Yost. R*: An Overview of the Architecture. *Readings in Database Systems*, 1988.