

# Querying Distributed Data in a Super-peer based Architecture<sup>\*</sup>

Zohra Bellahsène<sup>1</sup> and Mark Roantree<sup>2</sup>

<sup>1</sup> LIRMM, UMR 5506 CNRS/Université Montpellier II, France - bella@lirmm.fr

<sup>2</sup> Interoperable Systems Group, Dublin City University, Ireland - mark@computing.dcu.ie

**Abstract.** Data integration is a significant challenge: relevant data objects are split across multiple information sources, and often owned by different organizations. The sources represent, maintain, and export the information using a variety of formats, interfaces and semantics. This paper addresses the issue of querying distributed data in a large scale context. We present a p2p information mediation framework based on the notion of super-peers, providing a *super-peer* network. This makes it possible for a super-peer to reach every other peer (data source) in the system, thus realizing the concept of a integrated schema formed from all possible information sources. This is achieved by classifying data sources into domains and creating user profiles for query optimization purposes.

## 1 Introduction

The area of information integration has been moved towards flexible architectures [18], and more recently the integration of purely semi-structured sources [7], and both structured and semi-structured data in the same system. Some of this research has focused on large scale integration where the number of information sources numbers thousands [2]. Here, we are interested in providing a logical architecture where the number of information sources has no set limit. The assumption is that data is spread over a wide area and includes database systems, legacy systems and unstructured and semi-structured sources such as web servers. In order to provide such an architecture, it is necessary to use research and technology more associated with wide area networking than information management. This is due to the fact that conventional integration architectures cannot scale to the numbers envisaged in these systems.

**Motivation** Before the emergence of internet-based data, information was generally stored in database systems or proprietary file systems. There have been many solutions offered to the integration of these systems, and although this is still an on-going problem, the issues involved are well-understood. In fact, the problems have (more or less) moved from structural integration issues to semantic integration issues [8], and sometimes focuses on the integration of behavior [24], metadata modeling [20], or integrity constraints [5]. What is generally accepted is that no solution is complete without addressing the more recent topic of semantic integration. As is shown in [8], the

---

<sup>\*</sup> Supported by Ulysses Research Grant FR/2004/032.

integration of data sources from homogeneous domains generally assists the process of semantic integration.

However, the focus point in data integration has shifted for many researchers with the proliferation of web-based data sources. In many applications, it is no longer sufficient to integrate traditional database systems, as often there is equally important information located in newer unstructured or semi-structured sources. Furthermore, web servers are many and thus, the additional problem of large-scale integration must be addressed. The motivation for this work is two-fold: to provide an architecture which can address the large-scale issue; and also to provide a smarter integration strategy assuming that the concept of a single global schema is infeasible due to the large scale and heterogeneity of data sources. In this respect, the support services (query and metadata) are vital in the provision of scale, quality (in terms of data accessed), and dependability.

**Contribution of this Paper** Our contribution to this problem is to employ networking technology in the form of peer to peer networks, while modifying the traditional approach to suit the integration of information systems. This usage of P2P is not new as it has been used in the Piazza project [9] but this work is extended by using the *super-peer* concept [17] to assist in the management of global servers, where a super-peer acts as a mediator to a cluster of information sources. Additionally, we adopt a flexible approach to integration in mediated schemas, as this is best suited to a super-peer network approach. This paper primarily focuses on architectural issues, but also includes an overview of Query Service pragmatics.

**Outline** This paper is organized as follows. The large scale context of data integration is described in §2; the logical architecture of our system is presented in §3; We then move to describe our Query Service in §4 while §5 reviews different solutions currently available both in the industry and in the literature; concluding remarks and current research are described in §6.

## 2 The Peer to Peer Context

The limitation of a traditional client-server architecture is clear in a large scale distributed environment as resources are concentrated into a small number of nodes and features such as replication and caching must be introduced in advance. Peer to Peer systems appear to offer an alternative to the client/server architecture in a large scale network because they distribute the main costs of sharing data (i.e. disk space and bandwidth for transferring them). Each node participating in a peer to peer system acts as both client and server, and as a result each peer allows other peers access to some of its resources. There are several types of file exchange in peer to peer with different degrees of centralization. In pure peer to peer systems like Gnutella, peers all play the same roles (client and server) and have the same responsibilities, as there is no centralization policy. A query is addressed to a peer which then forwards this to its neighbors and so on, until an answer is found or the lifetime of the message has exceeded. The advantages of the pure peer to peer approach are a better load balance of communication and

processing, and a natural robustness. The main drawback of this approach is the large number of messages that are generated in the network. In peer to peer systems like Napster, there is a centralized directory of all sites. A query is sent to the site housing the directory which finds the appropriate sources. As a result, the hybrid approach provides an improved query performance. However, the storage of the directory is costly: the single node may entail a performance and scalability bottleneck. Furthermore, this approach is oriented towards a client-server rather than peer to peer architecture.

More recently, the *super-peer* approach has been designed for the Morpheus system [17]. This approach presents a cross between pure peer to peer and hybrid systems. A super-peer is a node in the peer to peer network that acts as a centralized server to a subset of nodes representing clients. Super-peers are connected to other super-peers in the network. Queries are submitted by clients to the super-peer which then finds the appropriate sources and returns the results to the clients. Super-peers combine the pure and the centralized distributed approaches and therefore represent a hybrid approach. More precisely, it combines the efficiency of a centralized client-server architecture with the autonomy, load balancing and robustness of distributed search. The behavior of the super-peer system has been studied in [26], where a set of basic rules that capture the main tradeoffs in super-peer networks have been proposed.

However, queries are executed “blindly” with no requirement for specifying data sources or prior knowledge of database schemas. This is acceptable where the architecture is aimed at a general purpose solution. However in the context of this research, we are dealing with the integration of information systems where it is assumed that some form of traditional query processing is supported. Thus, solutions such as the Time to Live (TTL) protocol use by Gnutella are not sufficient here. With this method, a query is broadcast in all directions, and continues to move past each super-peer until the required information is found or its TTL expires (typically by decrementing some counter after each super-peer is visited). This solution is insufficient as no exploitation of schema knowledge takes place.

### 3 The XPeer Logical Architecture

In this section, we present the logical architecture of our current project named *Xpeer*. The goal of Xpeer is defining and implementing a mediation system that allows querying a large scale P2P Database Management System by hiding the distribution, localization and heterogeneity of data sources and providing acceptable query response times. The architecture for our system is based on the *super-peer* concept. We recall that the only assumption needed regarding the topology of a peer to peer network is that it contains a strongly connected graph to provide all required functionality. Thus, the network topology shown in this section is provided for reasons of consistency only. Furthermore, we assume that both data and metadata flows are XML-based.

In order to exploit the *data web* one of the main issues is to describe data in semantic terms in order to facilitate its correct usage by software systems and improve understanding for the human reader. There are currently two main approaches to this problem. The first approach is adopted by the A.I. community: it uses knowledge representation languages to specify concept senses or reasoning techniques on the data

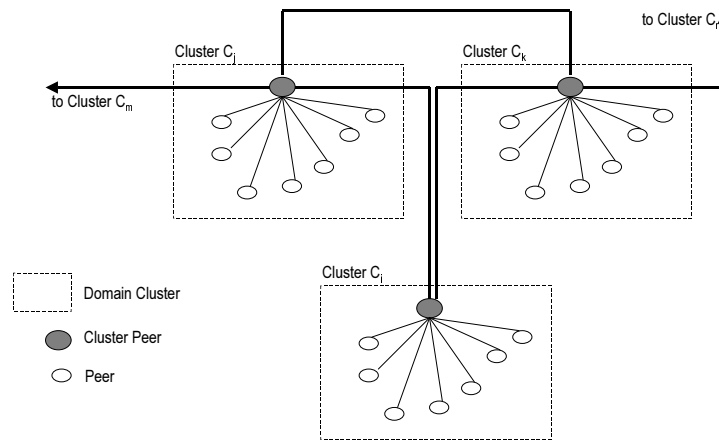
which follows the RDF format. However, most of these approaches exploit neither the structures of data items nor the fact that there is an increasing volume of XML data on the web.

The strong interest of the database community for the semantic web can be seen as a continuation of data integration research. The second approach is adopted by projects such as Piazza [9] where mediation is based on semantic descriptions not only from data sources but also from the links between those sources (or peers). Scalability is performed in a simple incremental manner: adding a new peer (or source) requires that this peer describes its data, and adding new links to its closest neighbors (on a semantic level). However computing all result sets for conjunctive queries in the case where the network is non-acyclic is too expensive. To reduce this complexity, the authors of [11] propose to use an ontology based on an atomic class and to describe the mapping as query containment. Queries are then a combination of atomic classes. However query expressiveness is unsatisfactory and query performance is not considered.

To extend these approaches, we propose a “semantic” representation of both sources and data in order to improve query performance. Data sources are modeled through a hierarchy (ontology) of concepts associated with a super-peer architecture.

### 3.1 Cluster Level

In the XPeer architecture, an information system plays the role of a *peer*. We group together a set of peers sharing the same schema into a cluster where the mediated schema for the set is managed by a *cluster-peer* node. This is illustrated in Fig.1 where the *cluster-peer* is connected to each of its *peers*. One can see that a *cluster-peer* plays the same role as a *super-peer* in the present literature [26]. We also retain the term *cluster* to group together a *cluster-peer* with its *peers*.



**Fig. 1.** Clusters in the XPeer Architecture

### 3.2 Domain Level

While a cluster-peer provides an integration of a set of peers within its cluster, this alone is insufficient in providing a large scale information system. For example, if we want to construct an information system for healthcare we need to group together information from physicians, clinics, medicine industries as well as government resources. These different resources do not share the same schema and even the same resource may have different schemas and data access. To facilitate these heterogeneities, we introduced two further layers in our peer hierarchy: domain-peers which a single domain-peer manages a set of cluster-peers, and a global-peer which manages the entire set of domain-peers. We define a *domain* as a set of clusters sharing the same category of information (physicians for instance). We also define a special node called a *domain peer* for each domain, and this acts as the entry point for this domain. We next describe the different functionalities for each of the system components.

## 4 Query Services

### 4.1 Schema Mediation

In data integration systems [16], queries are performed through a mediated schema. The system translates the user query into appropriate queries over related data sources according to semantic mappings. There are two main ways to specify mappings between the mediated schema and the data sources, and both are based on view mechanisms. The first approach relies on defining the mediated schema as a set of views over the schema of the data sources. This is the so-called Global as View approach or GAV [18]. The main advantage of this approach lies in the fact that the process of rewriting queries on the mediated schema is more straightforward. However the integration of new data sources is hard since the mediated schema must be updated accordingly.

The second approach for specifying mappings, called Local as View or LAV requires describing the data source schemas as views over the mediated schema. The main drawback of this approach is rewriting a query in terms of views: this problem is known to be difficult and costly [12] as one must use the views in the reverse direction. However, integrating new sources in the system can be managed locally by updating the view schema of the added sources. The Local as View formalism is also more appropriate in a heterogeneous integration system that includes many sources, since a newly added source is described in its own data model and need not be translated (to a common model format). Therefore, this approach supports scalability.

In our architecture a peer may join the system in one of two roles: posing queries and providing a schema to be shared with other peers (client and server); and posing queries only (client). As each peer describes its own data in the form of a schema, it is necessary to express the description of the mediated schema in terms of the peers which comprise the unified view. Given the mediated architecture of XPeer (i.e. a mediated schema per cluster) and the dynamism of the peer to peer context, the Local as View is the more appropriate approach. However, since each peer may also act as a client, XPeer differs from the original research as it employs a mixed approach for describing

the connection between the schemas of peers and the mediated schema of the cluster-peer schema within each cluster.

More precisely, within a cluster peers must conform to a unique schema (i.e. XML Schema). This means that each peer should export its data according to this DTD, and is used in services as WAP. Concerning the view on which a peer acts as a client, a simple process filters the cluster schema according to the needs of each peer, thus a peer is provided with a view over the cluster DTD. To summarize, schema mappings between peers and their cluster peer are defined following the Local as View approach where the cluster schema is a fixed DTD. The main advantage of our approach is that a peer may be added or dropped without affecting other peers and more importantly, without affecting the mediated schema. The sole requirement is to update the list of available peers. Furthermore, when a client wishes to query the entire information system, the peer is provided with a mediated schema over pertinent cluster schemas. This process is done with respect to the Global as View approach.

## 4.2 The Peer View

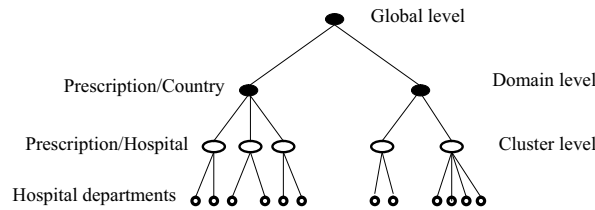
When a peer (as client) joins the system, it specifies its requirements by accessing the complete list of domains ( $D_1$  to  $D_n$ ). Those are specified as label descriptions (visible to clients), and managed by the Metadata Service in order to filter clusters of interest, and to optimize queries by removing lookup steps. In this way, each client will have their own customized Schema. This interface is actually a set of domain schemas, called the Peer View. Each domain schema is an integrated view of different exported clusters schemas on a specific domain. The main issue is to define the mappings between the view schema and the exported cluster schema. Indeed, for building such a view it is necessary to perform a semantic integration of the exported cluster schemas. Schema matching is not the main focus of this paper as it is covered in [23] where the semantic integration and the mapping generation are addressed.

Consider a healthcare application in the European Union. Suppose one is interested in studying some diseases and their treatment used in different countries across the E.U. This is illustrated in Fig. 2 where a *cluster* represents the prescriptions for one hospital, a *peer* a hospital department and a *domain* the prescriptions for a country.

*Example 1.* Assume that a doctor working in a French hospital has a requirement to perform statistics on prescriptions concerning asthma disease in Ireland. For this purpose, he defines the following view using XQuery:

```
For $d in domains
  Where $d/name = "Ireland" AND d/clusters/keyword = "asthma"
  Return
  <cluster_desc>
  {$o/name/clusters/desc}
</cluster_desc>
```

The returned clusters are integrated to form a single Domain Schema on which the peer will formulate its queries.



**Fig. 2.** Hospital Sample Schema

### 4.3 Query Processing

In the XPeer system, a peer (i.e. the user) formulates its query  $Q$  using the mediated schema corresponding to its (peer) view. Next, this query is sent to the cluster peer, which will process the query as follows:

The query  $Q$  formulated by the peer on its Peer View schema is rewritten into a set of queries  $\{Q_1, Q_2, \dots, Q_k\}$ , where  $k$  is the number of cluster schemas that have been integrated. Each query  $Q_i$  is related to one cluster schema. Each cluster may now process the related query as follows:

Inside each cluster, the sub query  $Q_i$  ( $i=1, k$ ) is translated to a set of local peer queries  $\{Q_{i1}, Q_{i2}, \dots, Q_{im}\}$ , on the real peer schemas according to mapping rules, where  $m$  is the number of peer schemas that have been referenced in the related cluster. Finally, the results sent by the local peers are integrated by the domain peer, via the cluster peers, into one result that will be returned to the peer that initiated the query.

*Example 2.* Consider a query  $Q$  that retrieves the drugs prescribed for the asthma disease.

```

For $h in hospitals
Return
<name>
{$o/hospitals/prescriptions/drugs/name}
</name>

```

Let us briefly explain how this query will be processed. Suppose a doctor formulates this query via the peer  $P$ . At first,  $Q$  is rewritten into a query  $Q'$  with respect to the mappings in the cluster where  $P$  is located. Next, the system uses the mappings between the  $P$  View Schema and the related cluster schema to expand  $Q$ . Then, inside each related cluster peer  $C_k$ ,  $Q$  will be rewritten into query  $Q_k$  ( $k=1, M$ ) according the mappings between  $C_k$  schema and its peers schemas, where  $M$  is the number of cluster schemas that have been returned in the view schema previously computed from peer  $P$ . Finally, each cluster will return one result to the cluster peer which has initiated the query  $Q$ . The union of all the results provided by the cluster peers and those of  $Q'$  is computed. Then, this cluster peer returns the final result after eliminating redundancies to  $P$ .

## 5 Related Work

The first attempt to design a database based on a peer to peer network was done in the Piazza project [9]. Their architecture focused on dynamic data placement and was organized on spheres of cooperation. A data origin is modeled as a distinct entity from the other peers. Larger spheres of cooperation may be built by nesting existing ones.

A solution to data mediation in peer to peer database management systems has been presented in [13]. That paper focused on a peer programming language for mediating peer schemas. More precisely, this language allows users to specify mappings between a pair or small set of peers using peer descriptions instead of a mediated schema. This language also enables the storage description of each peer. Given these descriptions, a query formulation algorithm has been proposed. There are many differences between our approach and those of Piazza:

1. Our architecture is based on the super-peer network whereas Piazza is based on a pure peer to peer network. Therefore, our approach combines the efficiency of centralized information retrieval by providing a mediated schema per cluster with those of pure peer to peer (reliability, distribution of the load, etc.)
2. Piazza does not provide a mediated schema but mappings between a small set of peers. In our approach, a single mediated schema per cluster is provided.
3. A user is provided by a set of mono-domain mediated schemas. In Piazza, a peer is connected into a small set of peers, and by iteration, it can reach many peers. However, this solution is prone to degradation of query performance.

A distributed caching system for OLAP queries based on a peer to peer network has been proposed in [15]. This architecture is called PeerOLAP and was described as a set of peers that access data warehouses and pose OLAP queries. Each peer has a local cache and implements a mechanism for sharing its cache contents and computational capabilities with other peers. Queries are addressed to a peer that may either process them locally if it has the required data or propagate the queries to its neighbors. The network architecture is based on classical peer to peer. Therefore the efficiency of query processing is compromised. Moreover, data at sources may change at different speeds and intervals, making it difficult to maintain both the fragments stored at the peers, and at the different data warehouses.

The main difference with our approach relies in the fact that our proposal is aimed at providing an infrastructure to data mediation where data resides at the original sources. However, for performance reasons, we store materialized views at some peers.

A peer to peer system called P-Grids [1] focuses on the construction and maintenance of the Grid or P2P system using randomized algorithm, as opposed to managing Information Systems which are connected using a P2P approach. At some starting point, the system assumes that no information regarding access paths between nodes is known. Thus, the entire search space is split into two sections where two peers, know the location of each other and then take responsibility for each half of the search space. Eventually, a Grid is formed as access paths between each set of pairs becomes known. This results in a graceful scaling in terms of both storage and communication costs and enable the access to both reliable and unreliable peers.

With respect to our research, the P-Grid approach concentrates on P2P issues while we are more concerned with data management issues such as querying and metadata. Furthermore, our approach does not take a binary approach to peer management as this causes an initial problem if the search space (or global information system) is very large. As It was shown in section 3, we divide the search space into a far greater number of sections, depending on the number of information system domains available.

## 6 Conclusions

In this paper we presented an architecture which is suited to large scale data integration. The architecture was designed using super-peer network features to cope with the issues caused by very large numbers of data sources and an environment in which nodes were either fixed or mobile and permanent or temporary. In our approach we view the network as a large searchable database, with special roles for super-peer elements. Specifically, a peer can be promoted to one of these three roles: (1) **cluster peer**: a mediator for its cluster; (2) **domain peer**: management of all cluster peers within a given domain; and (3) **global peer**: management of the domain set. By specifying client profiles (§4) and through the operation of a metadata service which models all system entities [3], it is possible to query all roles in an infinitely large system.

Our current research is focused on three primary areas. The first of these is the full deployment of our XML metamodel to represent all aspects of the system including mediated schema interfaces, metadata service repositories, domain management and unstructured data sources. The second task is to complete the current implementation of our system in order to test and optimize the performance of external queries. Finally, we are investigating the issue of selecting views for materialization and placement as part of the query optimization process.

**Acknowledgements.** The authors would like to acknowledge the contribution of Laurent Mignet in the section covering cluster management.

## References

1. Aberer K. P-Grid: A Self-Organizing Access Structure for P2P Information Systems. *Proceedings of 9th International Conference on Cooperative Information Systems (CoopIS)*, pp. 179-194, LNCS 2172, Springer, 2001.
2. Abiteboul S., Cluet S., Ferran G. and Rousset M. The Xyleme Project. *Computer Networks* Vol. 39, pp 225-238, 2002.
3. Bellahsène Z. and Roantree M. Large Scale Integration Using the XPeer Architecture. *Technical Report no. ISG-03-06*, Dublin City University, [www.computing.dcu.ie/~isg](http://www.computing.dcu.ie/~isg), October 2003.
4. Bernstein P. A., Giunchiglia F., Kementsietsidis A., Mylopoulos J., Serafini L., Zaihrayeu I.: Data Management for Peer-to-Peer Computing : A Vision. *WebDB 2002*: 89-94
5. Cali A., Calvanese D., DeGiacomo D., and Lenzerini M. Data Integration under Integrity Constraints. *Proceedings of 14th International Conference on Advanced Information Systems Engineering (CAiSE 02)*, LNCS 2348, pp. 262-279, Springer, 2002.
6. Conrad S., Hding M., Saake G., Schmitt I., and Trker C. Schema Integration with Integrity Constraints. *Proceedings of 15th BNCOD*, pp. 200-214, LNCS 1271, Springer, 1997.

7. Draper D., Halevy A., and Weld D. The Nimble XML Data Integration System. *Proceedings of the 17th International Conference on Data Engineering*, (ICDE 2001), IEEE Computer Society Press, 2001.
8. Delobel C., Reynaud C., Rousset M., Sirot J., and Vodislav D. Semantic Integration in Xyleme: a Uniform Tree-based Approach. To appear in *Journal on Data & Knowledge Engineering*, 2003.
9. Gribble S., Halevy A., Ives Z., Rodrig M., Suci D. What Can Database Do for Peer-to-Peer? *Proceedings of the Fourth International Workshop on the Web and Databases*, WebDB 2001, in conjunction with ACM PODS/SIGMOD 2001.
10. Gnutella Website. [www.gnutella.com](http://www.gnutella.com), 2002.
11. Goasdoue F. and Rousset M. Querying Distributed Data through Distributed Ontologies: A Simple but Scalable Approach. In *Proceedings of IJCAI03 Workshop on Integration on the Web (IIWeb 03)*, 2003.
12. Halevy A. Answering queries using views: a survey. *VLDB Journal* 10:4, pp 270-294, 2001.
13. Halevy A., Ives Z., Suci D., and Tatarinov I. Schema Mediation in Peer Data Management Systems. To appear in *19th International Conference on Data Engineering (ICDE)*, 2003.
14. Hull R. and Zhou G. A Framework for Supporting Data Integration Using the Materialized and Virtual Approaches. *Proceedings of ACM SIGMOD Conference*, pp. 481-492, ACM Press, 1996.
15. Kalnis P., Ng W., Ooi B., Papadias D., and Tan K. An Adaptive Peer-to-Peer Network for Distributed Caching of OLAP Results. *Proceedings of 2002 ACM Sigmod Conference*, 2002.
16. Lenzerini M. Data Integration: A Theoretical Perspective. PODS, *Proceedings of the 21st ACM Symposium on Principles of Database Systems (PODS)*, pp. 233-246, 2002.
17. Morpheus Website. [www.musiccity.com](http://www.musiccity.com), 2002.
18. Manolescu I., Florescu D., and Kossmann D. Answering XML Queries on Heterogeneous Data Sources. *Proceedings of 27th International Conference on Very Large Data Bases*, Morgan Kaufmann 2001.
19. McBrien P. and Poulouvassilis A. A Semantic Approach to Integrating XML and Structured Data Sources. *Proceedings of 13th International Conference on Advanced Information Systems Engineering (CAiSE 01)*, LNCS 2068, pp. 330-345, Springer, 2001.
20. Roantree M., Kennedy J., and Barclay P. Using a Metadata Software Layer in Information Systems Integration. *Proceedings of 13th International Conference on Advanced Information Systems Engineering (CAiSE 01)*, LNCS 2068, pp. 299-314, Springer, 2001.
21. Roantree M., Kennedy J., and Barclay P. Integrating View Schemata Using an Extended Object Definition Language. *Proceedings of 9th International Conference on Cooperative Information Systems (CoopIS)*, pp. 150-162, LNCS 2172, Springer, 2001.
22. Saltor F. Castellanos M., and Garcia-Solaco M. Suitability of Data Models as Canonical Models for Federated Databases. *ACM SIGMOD Record* 20:4, 1991.
23. Tranier J., Baraer R., Bellahsène Z., and Teisseire M. Where's Charlie: Family based heuristics for Peer-to-Peer Schema Integration. To appear in *IDEAS 2004*.
24. Vermeer M. and Apers P. Behaviour Specification in Database Interoperation. *Proceedings of 9th International Conference on Advanced Information Systems Engineering (CAiSE)*, pp. 61-74, LNCS 1250, Springer, 1997.
25. Wiederhold G. and Genesereth M. The Basis for Mediation. *Proceedings of the 3rd International Conference on Cooperative Information Systems (CoopIS-95)*, pp. 140-157, 1995
26. Yang B., and Garcia-Molina H. Designing a Super-Peer Network. To appear in *19th International Conference on Data Engineering (ICDE)*, 2003.