Peer-to-peer systems for simple and flexible information sharing

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE,
FACULTY OF SCIENCE
AT THE UNIVERSITY OF CAPE TOWN
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF PHILOSOPHY (IN INFORMATION TECHNOLOGY)

BY
SUHENDRAN PATHER
NOVEMBER 2009
SUPERVISED BY
PROF SONIA BERMAN

Acknowledgements

My gratitude goes out to my thesis supervisor, Prof Sonia Berman for allowing me to pursue my research interest, and guiding me through the process. Her willingness to review my work. kindness, support and patience is greatly appreciated.

I would also like to extend my appreciation to all my employers who have given me the opportunity to learn and exposure to technologies that enabled me to formulate my thesis.

To my family who has always encouraged me to pursue my academic goals and trust in my abilities.

Finally I would like to thank my loving wife Loshitha for her unconditional support during my studies and her patience when I had to put in the effort performing my research. You always believed in my abilities and I am thankful that you are part of my life. A lesson to my newly born daughter Thashmiri to always pursue your dreams and strive to do the best in whatever you choose to do in life.

Abstract

Peer to peer computing (P2P) is an architecture that enables applications to access shared resources, with peers having similar capabilities and responsibilities. The ubiquity of P2P computing and its increasing adoption for a decentralized data sharing mechanism have fueled my research interests. P2P networks are useful for sharing content files containing audio, video, and data.

This research aims to address the problem of simple and flexible access to data from a variety of data sources across peers with different operating systems, databases and hardware.

The proposed architecture makes use of SQL queries, web services, heterogeneous database servers and XML data transformation for the peer to peer data sharing prototype. SQL queries and web services provide a data sharing mechanism that allows both simple and flexible data access. While other P2P database sharing systems exist and web services are now widely used, this architecture provides a simple and integrated way of using both the SQL and web services approach.

Table of Contents

List of acronyms

CPU	Central Processing Unit
DBMS	Database Management System
DNS	Domain Name Sever
DHT	Distributed Hash Table
DOM	Document Object Model
ECA	Event Condition Action
ISP	Internet Service Provider
JXTA	Juxtapose
JAX-WS	Java API for XML Web Service
JDBC	Java Database Connectivity
OS	Operating System
PDP	Peer Discovery Protocol
P2P	Peer to Peer
SAX	Simple API for XML
SQL	Structured Query Language
SOA	Service Oriented Architecture
SRDI	Shared Resource Distributed Index
TTL	Time to Live
URL	Uniform Resource Locator
W3 C	World Wide Web Consortium
XOM	XMI. Object Model
XML	Extensible Markup Language

List of Figures

Chapter 1 Introduction

Information is increasingly becoming an important asset for efficient decision making that is utilized in our everyday lives at work and at home. Knowledge workers in corporate environments rely extensively on information to make strategic decisions to assist the organization in growth opportunities, resource planning, and assessment of market trends to name a few. The ubiquity of information emanates from various sources which must be captured and shared to extract its best value. The rationale behind my research realizes the importance of data sharing across a distributed environment to permit collaboration. This will empower users located anywhere on a computer network to share and access data at any time.

A database management system is an ideal means of capturing and sharing information that is easily accessible using a computing device. This project aims to permit data sharing in a peer to peer (P2P) network in such a way as to ensure interoperability across peers with different operating systems, databases and hardware. The goal is also for peers to access data from a variety of data sources such as flat files and relational databases, and to provide both simple and flexible access.

A system architecture based on the JXTA P2P protocol is presented, called dbShare. This allows simple access to metadata and to data sources in their entirety (e.g. to retrieve an entire relational database view) and permits peers to share their data through any relational DBMS or via a web service. For flexible access, peers can also use SQL or XPath to retrieve only data of interest. While other P2P database sharing systems exist and web services are now widely used, the proposed architecture provides a simple and integrated way of using either approach.

SQL queries are very popular for relational data access while XPath is preferred for XML content. SQL is simple and widely used; due to its hierarchical structure XPath offers advantages for automated query generation and semantic interpretation of structured data.

DbShare is a system for information sharing and retrieval that can be used by peers at a number of different levels. At its most basic level peers share a metadata repository providing the

location of a variety of data sources and a description of data to be shared by participating peers. At the next level it provides a means of retrieving data in its entirety from a web service or a public relational database view. At the higher level specific queries may be structured and directed to peers to extract from their resource only data that are of interest.

This thesis describes the design of the dbShare system and a prototype implementation that was created. A complete implementation of a P2P information sharing system will include schema matching, result integration, sophisticated query generators and different user interfaces, which is beyond the scope of this work. These components are thus not included in the prototype, which focuses on the overall architecture and the use of a peer-to-peer system.

1.1 Thesis Structure

Chapter 1: Introduction
The chapter provides an introduction to the thesis.

Chapter 2: Background
The chapter provides a background on several P2P systems in use today.

Chapter 3: .JXTA and SRDI
The chapter provides an introduction to the JXTA protocol and SRDL

Chapter 4: Survey
The chapter outlines a survey conducted to assess the need for a data sharing system.

Chapter 5: System Design
The chapter presents the dbShare system along with reasons for design decisions made.

Chapter 6: Implementation
The chapter describes the implementation of a prototype using JXTA, SRDI and XOM.

Chapter 7: Testing

This chapter outlines testing of the JXTA software and the prototype system.

Chapter 8: Conclusion

This chapter concludes and suggests some avenues for future work.

Chapter 2 Background

A background on peer to peer networks and peer to peer database management systems is provided here along with an introduction to web services.

2.1 Introduction

P2P systems use a decentralized network architecture, whereby individual hosts collaborate to share data (Figure 2.1). It differs from the typical client-server model since there is no communication to and from a central server. The nodes in the peer to peer system could be heterogeneous (different hardware. OS. database and languages) but must communicate using the same networking protocol (Figure 2.1).

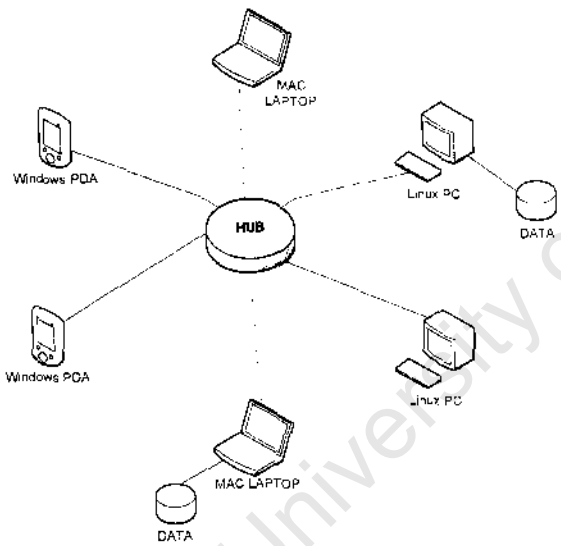


Figure 2.1 Peer to peer network

Decentralized systems rely on distributed index structures, thus avoiding a single point of failure through multiple data stores [41]. P2P systems are able to allow nodes to dynamically join and leave the network at any time with minimal effect to the operation of the system [41]. With the absence of a central server, as nodes are added to the network the P2P system achieves a high level of availability, scalability, redundancy with reduced setup and administration costs [41].

There is no single server that will limit the systems scalability due to server bottlenecks [17].

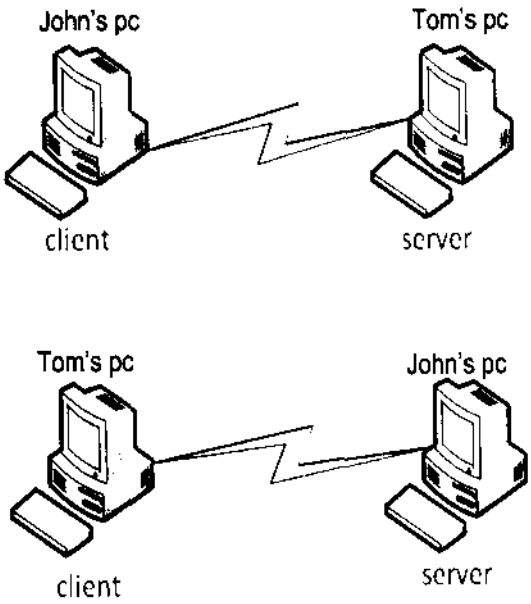


Figure 2.2 Peers can act as clients and servers

All nodes in the peer to peer network can act as both clients (when consuming resources) and servers (when providing resources) (Figure 2.2) [171. Distributed systems that are well architected are proven to ensure almost linear growth as more hosts are added, with each participating node contributing data and computing resources (such as CPU, storage, memory) to the overall network.

Several networks employ a combination of client-server architecture for some tasks and a peer to peer structure for others. Freenet is a decentralized P2P structure for all of its service requirements and referred to as a true peer to peer network [48]. The membership of a P2P system is relatively unpredictable since peers join and leave the network at any time [17].

Decentralization helps eliminate proprietary interests in the system's infrastructure; since trust is diffused over all participants in the system [17]. Since the membership in the system is ad-hoc and dynamic, it is very difficult to predict the location and quality of the system's resources [171. The dynamic nature of the peer to peer system may impose limitations on data consistency and

availability as well as bring about several technical challenges [17]. The impact of maintaining shared indexes across peers may become restrictive with the growing number of peers [17]. The current content sharing systems do not focus extensively on real-time data synchronization, and they typically only support retrieval of objects by name.

The fundamental problem in most P2P systems is the placement and retrieval of data [48]. The P2P world is tacking in the areas of semantics, data transformation, and data relationships, which are some of the core strengths of the data management population [17].

This makes the peer to peer network environment appealing to the database community since it may be well suited for data sharing using heterogeneous data sources [48]. Data management techniques may assist in the development of improved solutions to the data placement problem such that data is placed in strategic locations and then used to improve query performance [17].

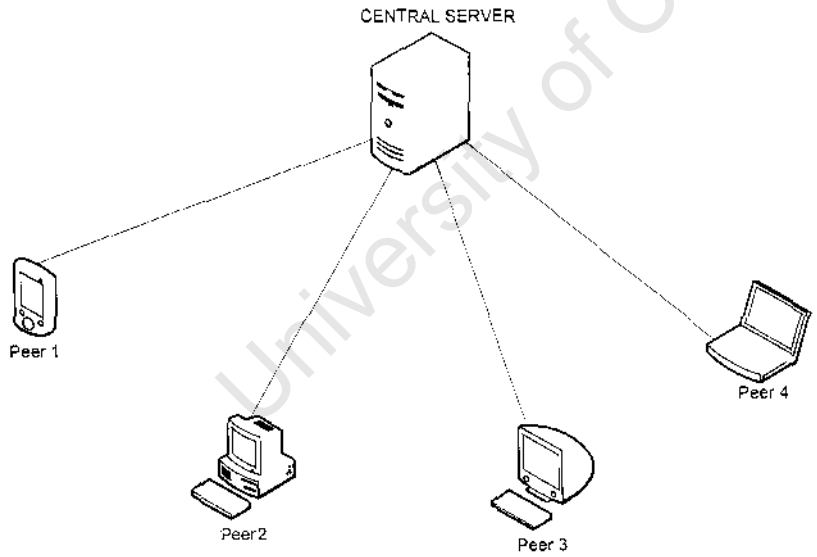


Figure 2.3 Centralized peer to peer network architecture

Clients in a peer to peer network have the same capability, responsibilities and provide resources such as bandwidth, storage space, and computing power [17]. The growing number of nodes in a purely decentralized network enables an increase in the total capacity of the system [17].

The use of a central server in a federated environment (client server) can result in severe bottlenecks when adding more nodes to the network and slower data transfer for all users (figure 2.3). The distributed nature of peer to peer networks increases robustness and avoids single points of failure due to several copies of data replicated across multiple peers [48]. P2P networks are typically used for connecting nodes via largely ad hoc connections.

2.2 Domain Name Server (DNS)

A name server plays an integral part in name resolution and is the backbone of the internet. The primary function of a domain name server is to map a domain name (human readable) to a unique machine address (machine readable). Machines are able to communicate with each other using an IP address however humans tend to remember domain names rather than actual IP addresses (unique number notation).

P2P networks contain or rely on some non-peer elements, such as DNS to ensure name resolution and to fulfill search capabilities.

2.3 Overlay network

An overlay network is a computer network which is built on top of another network. Nodes in the overlay can be thought of as being connected by virtual or logical links, each of which corresponds to a path, perhaps through many physical links, in the underlying network. Overlay networks can be constructed to permit routing messages to destinations while not making use of the IP address.

Overlay networks provide a virtual private network (VPN) capability with the ability to guarantee quality of service on the network. The overlay network operates at the logical layer with limited autonomy over how packets are routed in the underlying physical network.

P2P Classification

Peer to peer networks may be categorized into the following categories:

- Centralized P2P network such as Napster
- Decentralized P2P network such as KaZaA
- Structured P2P network such as CAN
- Unstructured P2P network such as Gnutella
- Hybrid P2P network (Centralized and Decentralized) such as JXTA

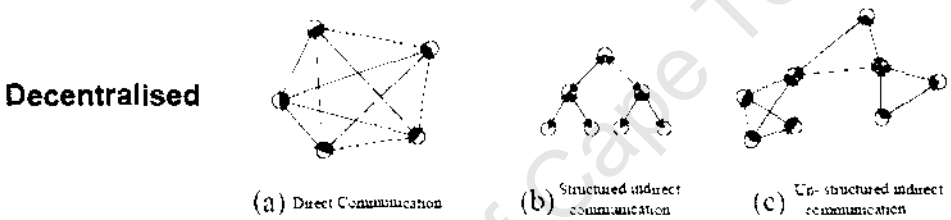


Figure 2.4 Peer to peer classification [421

2.3.1 Centralized and decentral·zed

Peer to peer networks can be classified according to their degree of centralization as shown in Figure 2.4. In a decentralised peer to peer network all the nodes have equal capabilities and responsibilities, combining the roles of clients and server with the absence of a central server [17]. In a completely centralised structure such as Napster the central server infrastructure maintains the metadata information about all peers [41].

2.3.2 Structured and Unstructured

The P2P network forms an overlay network where the participating peers in the network are linked together through a logical connection. Based on how the nodes in the overlay network are linked to each other, the P2P network can be classified as either structured or unstructured (Figure 2.4).

An unstructured P2P network is formed when the overlay links are established randomly [31]. New peers can be added to the overlay network by copying existing links of another node and then form its own links over time. In an unstructured P2P network, when data is requested the query is transmitted across the network flooding the network to find many peers that may possess the requested data however the queries may not always be resolved [31]. Since there is no correlation between a peer and the content managed by it, there is no guarantee that flooding will find a peer that has the desired data.

The request message is sent until the "ITT (Time-to-Live) counter expires [41]. Flooding also causes a high amount of signaling traffic in the network and hence such networks typically have very poor search efficiency [41]. Most of the popular P2P networks such as Gnutella and FastTrack are unstructured [17].

Structured P2P networks have a strict topology with each node storing and maintaining information about other peers in the network [17]. This enables an efficient routing mechanism for requesting data, even if the data is extremely rare. The cost of searching in a structured P2P system is reduced compared to an unstructured P2P network however additional maintenance operations are required when nodes are added or disembark from the network [17].

Routing tables must be updated, new entries added and redundant entries removed [41] when nodes join the network. If the population of the network does not change much over time the structured system can perform better than unstructured systems in terms of resource utilisaton [17].

The most common type of structured P21' network is the distributed hash table (UHT), in which a variant of consistent hashing is used to assign ownership of each file to a particular peer [14].

This is similar to a conventional hash table implementation assigning each key to an array slot [14]. Examples of popular DHT's are Chord [36], Pastry [36], and CAN [31]. HyperCuP[35] and Frecnet[27] are structured peer to peer networks however they are not based on the Distributed Hash Table. They will be discussed in the next chapter.

2.3.3 Hybrid

in a hybrid P2P system some of the operations are cenralised while others are decentralised. Both pure peer to peer and client/ server systems coexist as part of the hybrid model.

2.4 P2P database management systems

As suggested previously one of the key issues in peer to peer systems is data placement and retrieval [17]. Databases have been designed to focus in areas of semantics, handle transformation of data and relationships between data entities. However P2P systems have a different focus to handle semantic free, large-granular requests for objects by an identifier (typically a name). This limits the techniques that might be employed to distribute the data. Database queries ensure the retrieval of data [17]. Views are virtual or logical tables composed of a subset of data which is sorted and displayed in a particular way. Integrity constraints can be used to enforce business rules at the database level.

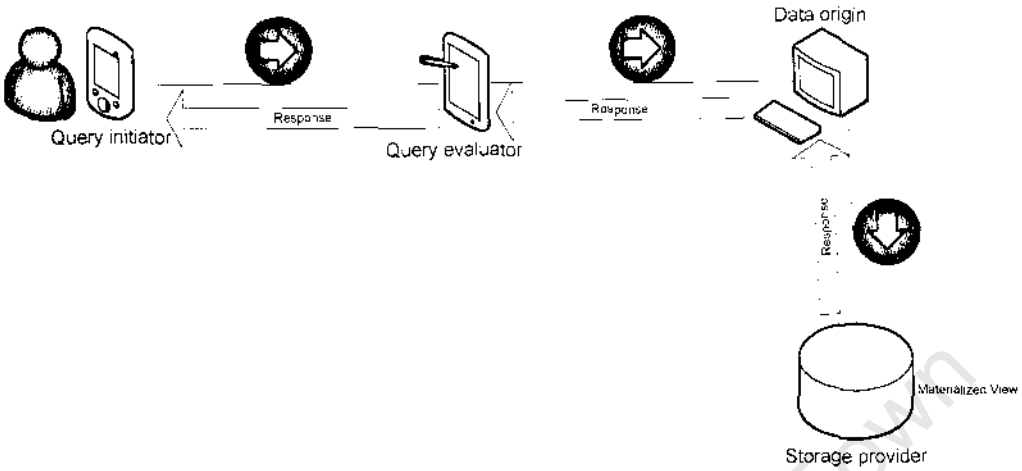


Figure 2.5 Roles in data exchange

When data is exchanged in the network peers fulfill 4 roles:

- *Data origin* is the owner of the data content and the authoritative source of that data [17].
- *Storage provider* is a peer that stores materialized views [17]
- *Query evaluator* uses a portion of its CPU resources to evaluate the set or queries forming its workload [17]
- *Query initiators* are peers acting as clients in the system that pose queries [17].

When queries are initiated from the query initiator resources are consumed in several systems [17]. Network, CPU, memory, and storage resources need to fulfill the client's request are shared and consumed [17]. The overall cost of the request is the cost from interactions between the query initiator and the query evaluator, query evaluator and the storage provider or data origin, and the cost to transfer the results hack to the query initiator [17].

2.4.1 The Piazza peer to peer system

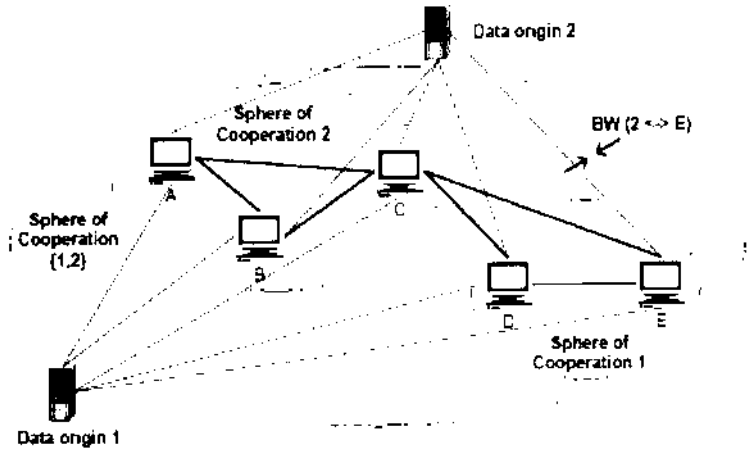


Figure 2.6 Piazza network

The Piazza system was designed to overcome scalability issues and the ability to perform extensive data manipulation in a peer to peer network [19]. The origin of the data in the network is defined as the master data store which is the sole authority for data updates [19]. The peers in the Piazza overlay network may have a mediated schema and may specify schema mappings [19]. A centralised global system catalog has all the mappings of peer relations which are locally cached at peers. The Piazza system uses an XML data model for flexibility [38]. The system offers interoperation of RDF data and XML data using OWL ontologies [38].

Query optimization

Materialized views in a distributed network architecture are used in the Piazza system. When queries are initiated there are several criteria that are evaluated to ensure that efficient data retrieval is performed [19]. Intelligence is built into the system to assess the current query workload, query patterns and rewrite queries to utilize materialized views whenever cost-effective [19]. New materialized views arc created for frequent queries that could return results more efficiently than accessing the base tables [38]. The query initiator must possess the metadata of existing materialized views to ensure efficient query processing and query rewrite

[19]. the Piazza system publishes the metadata about materialized view definitions and location information only to neighbour peers. This limits the network traffic during searches and eliminates flooding and broadcast messages ensuring scalability [19].

Consolidating query evaluation

When data cannot be retrieved from known peers the original data source must be accessed to fulfill query requirements [5]. When there are queries which cannot be evaluated within the same sphere of cooperation, an efficient evaluation strategy should be defined [5]. Broadcast query messages will be distributed across the cluster and commonalities identified. The cost of the query will determine an efficient execution plan to be chosen [5].

Guaranteeing data freshness

The use of extensive materialized views in the Piazza system requires a mechanism to update the views regularly to ensure data freshness [191. In order for the system to scale, data items use a time based mechanism to age out old data. Network traffic will be reduced as a result. However it will not achieve the strong semantics of traditional database systems [19].

2.4.2 Hyperion

Hyperion is a peer database management system (pdbms) extended with a peer to peer interoperability layer [5]. The layer facilitates the sharing and coordination of data by peers without relinquishing their autonomy. and establishes or abolishes an acquaintance semi-automatically at runtime [5].

Acquaintances are formed between databases to enable the seamless sharing of data [2]. The metadata between the acquainted databases are shared by the peers once the acquaintances are established. Metadata at the data level are expressed as mapping tables which are used as correspondence between data values of acquainted databases [5].

Hyperion supports pairwise acquainted peers for data sharing purposes. Event Condition Action (ECA) rules are used to facilitate the sharing or data between databases [5]. Hyperion is unlike the Piazza system since it performs data sharing at the data level rather than the schema level [5].

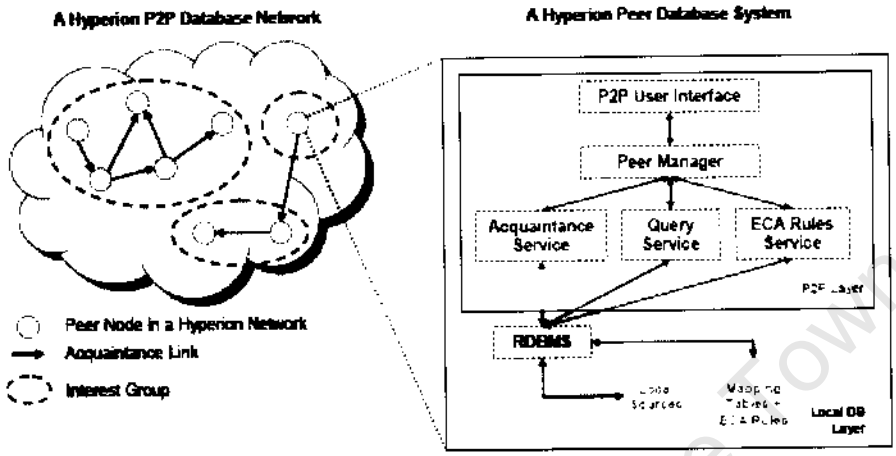


Figure 2.7 Hyperion database [5]

Mapping tables and ECA rules are applied at the P2P layer for sharing data between acquainted peers [5]. The peer to peer layer includes the P2P user interface which is the interface that users would use to submit queries to be executed either locally or globally (on other peers) [5].

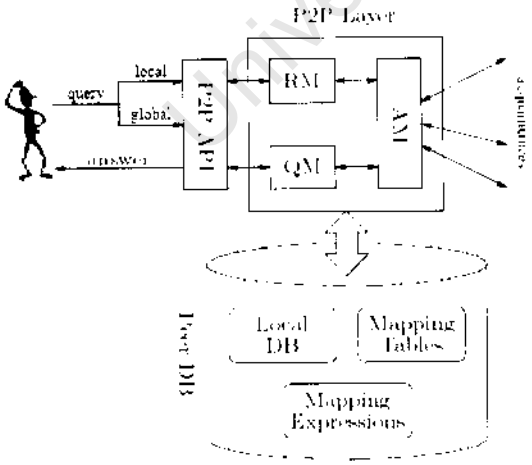


Figure 2.8 Hyperion database components [5]

Mapping tables and expressions are created and managed by the acquaintance manager that enables heterogeneous data sources to communicate and share information [5].
The query manager ensures that queries are executed against the relevant data source. Local queries are executed by using data only in the local database while global queries will use data on acquainted peers as well as the local peer to fulfill the request [5]. The query manager will rewrite the query to be syntactically correct to be executed against the acquainted data source and will use the acquaintance manager to facilitate the process [5]. The mapping tables and the expressions provided by the acquaintance manager are used to transform the query and data [5]. The rules manager manages and executes distributed ECA rules in order to enforce consistency policies and coordinate updates between peers [5].

Event Condition Action (ECA) is employed in Hyperion which uses database triggers to enforce data consistency and data exchanges across peers. Acquaintances are established rules arc created to support the data exchange process [5]. The rules make use of mapping tables and expressions to coordinate metadata and data between peers.

2.4.3 Edutella

Edutella is a peer to peer database system which builds on an RDF based metadata infrastructure using the JXTA framework [29]. Peers join the Edutella network and exchange RDF metadata information to be used to locate data and resources managed by the peers [29].

The initial Edutella services had a *Query Service* (standardized query and retrieval of RDF metadata), *Replication Service* (providing data persistence, availability and workload balancing while maintaining data integrity and consistency), *Mapping Service* (translating between different metadata vocabularies to enable interoperability between different peers), *Mediation Service* (define views that join data from different meta-data sources and reconcile conflicting and overlapping information) and *Annotation Service* (annotate materials stored anywhere within the Edutella Network) [29].

It was designed to exchange education media at universities across an overlay P2P network [29]. The university acts as a content provider and a content consumer. The metadata model is used to integrate heterogeneous peers (using different repositories, query languages, and functionalities) [29].

Edutella replication ensures data persistence and availability by replicating data and metadata across multiple peers. The query service queries a distributed RDF repository for the metadata information. Each peer has different service capabilities. However they will possess at the least RDF storage capabilities [29].

2.5 Web services

A service is a unit of work done by a service provider to achieve desired end results for a service consumer [4]. Web Service technologies provide standard, simple and lightweight mechanisms for exchanging structured and typed information between services in a decentralized and distributed environment [4]. "By focusing solely on messages, the Web Service model is completely language, platform, and object model-agnostic" [4]. Web services aim to simplify the process of interacting with applications by defining a standardized mechanism to describe, locate, and communicate with online applications using XML messages and open standards [12]. The messages may be based on SOAP (Simple Object Access Protocol) which is an XML protocol for messaging and remote procedure calls (RPC's). RESTFul web services are becoming popular which is based on collection of resources. Rather than define a new transport protocol, SOAP works on existing transports such as HTTP [12]. The web service description language (WSDL) is used to describe the web service's interface.

Chapter 3 JXTA and SRDI

The JXTA framework and the Shared Resource Distributed Index (SRDI) presented here are used later in the dbShare prototype. The SRDI provides a means to create an index of peers when they join the P2P network and is used to quickly locate peers on the network.

3.1 The JXTA framework

JXTA is an open source project introduced by Sun Microsystems. It is a set of XML, protocols to cover P2P functionality [31]. It is a layered approach to creating P2P applications with a framework that consists of the JXTA core layer, services layer, and application layer [31]. The peerlD uniquely identifies peers on the JXTA network [2]. The ID will be retained by the peer even though they may logoff or acquire a new IP Address from an ISP. The discovery of network resources makes use of advertisements which are published on the network [39].

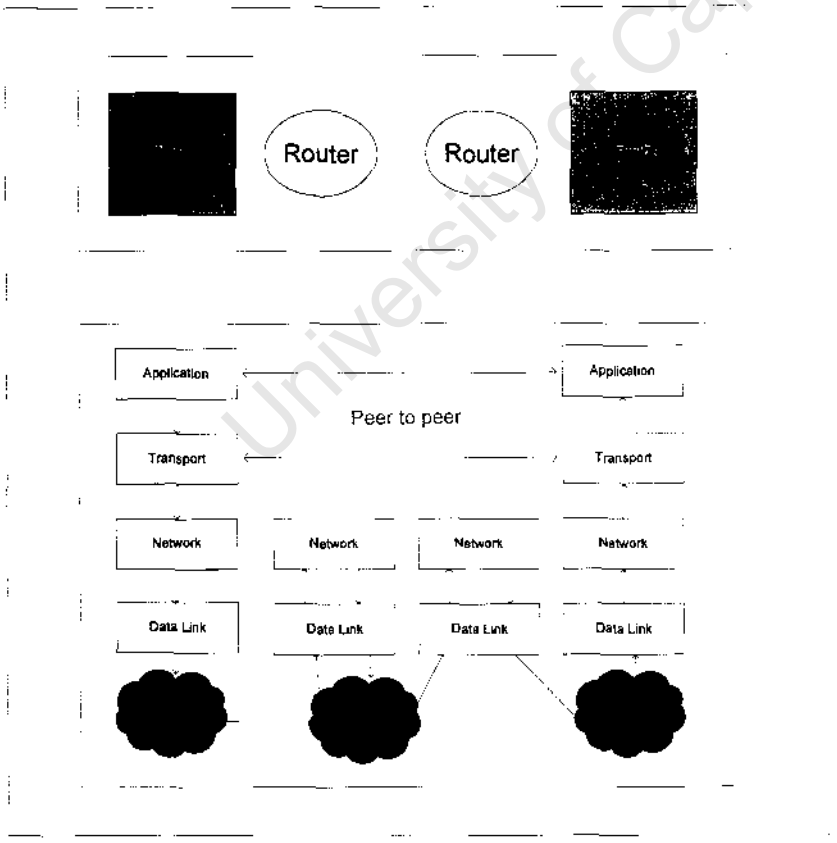


Figure 3.1 high level peer to peer network architecture

The Java JXTA platform allows peers to be deployed on different operating platforms and hardware devices giving the users flexibility for sharing information [15]. The application based on the JXTA platform does not necessarily ensure interoperability between peers. It provides a base layer that could ease interoperability problems [15]. Figure 3.1 shows a high level overview of peer to peer networks operating at the application and transport layers of the OSI stack.

JXTA protocols employ XML (eXtensible Markup Language) as a basis for data representation and data exchange. XML is chosen due to its strength of supporting rich structured content over the web that can be stored, queried, and manipulated easily to suit its requirement [15].

The JXTA framework in figure 3.2 accommodates several value added services such as database and file sharing, seamless peer discovery and database caching.

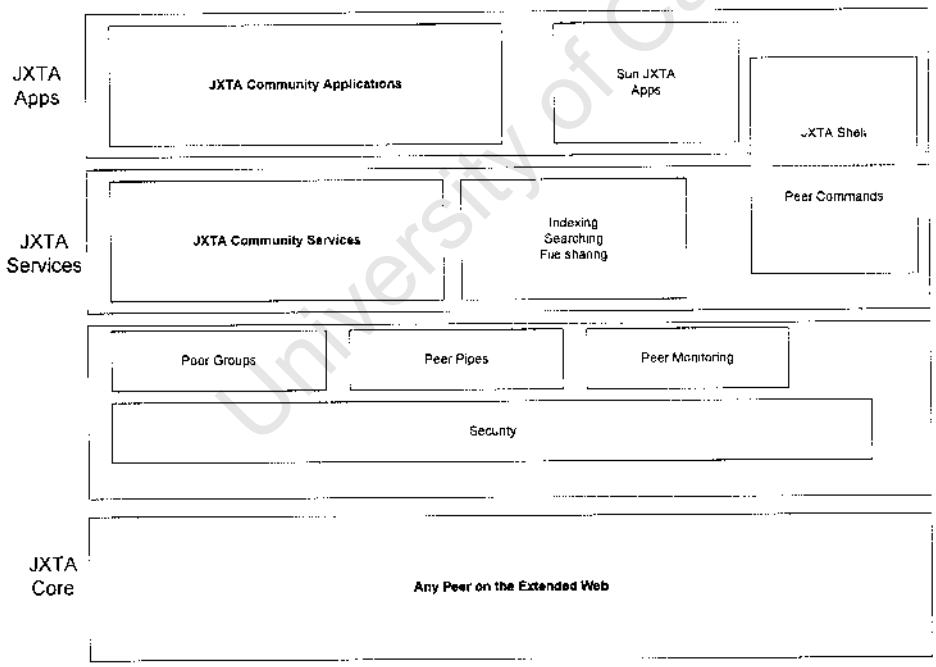


Figure 3.2 JXTA platform [15]

Metadata information on JXTA clients (such as presence, location, uptime, peer id, peer label, URI) are distributed to enable efficient search capability due to non reliance on a centralised server infrastructure. The distributed nature of the database sharing application allows for new nodes to join the network with the system scaling almost linearly [48].

The JXTA protocols allow peers to [16]

- Advertise content they would like to share
- Discover content they would be interested in
- Form and join peer groups
- Assist in routing and forwarding of peer messages transparently

Advertisements are in the form of XML messages when peers are joining, discovering, and sharing content. The SRDI which is part of the JXTA 2.0 framework indexes the advertisements and distributes the index for efficient retrieval of requested data.

3.2 Peer Management

JXTA 2 introduces the concept of a rendezvous super-peer network which greatly improves scalability by reducing propagation of traffic and an implementation of the shared resource distributed index (SRDI) [41]. In JXTA 2, the rendezvous super-peer network forms a loosely consistent DHT to resolve distributed queries [41]. The rendezvous peer view (RPV) implementation utilizes a super-peer architecture and a shared resource distributed index (SRDI) to index advertisement in aid of efficient query lookups [15]. The SRDI creates and maintains an index of resources in the network [41]. The rendezvous peers are searched to locate the relevant nodes to service a request [39].

Peers are able to join and leave the network at any time which should not affect the performance or availability of the system [48]. When peers attempt to join the distributed network a process is followed as shown in figure 3.3 to enable the peer to become part of the network [48]. Peer groups in the JXTA platform allow peers to connect and execute services. It is possible for peer groups to enable peers with similar query patterns to cluster together. A special peer group

called the world peer group which all peers arc part of is instantiated at bootstrap [15]. The dynamic discovery process using the Peer Discovery Protocol (PDP) is employed to discover peers in the network either through multicast messages or rendezvous peers [15]. The rendezvous peers maintain a list of all peers in the network through the discovery process. Once the peer that possesses the required data has been identified using the DOT it will share its data (figure 3.4) directly with the querying peer without following the route taken during the identification/ search process [15]. 'Ibis helps with efficient and rapid data retrieval.

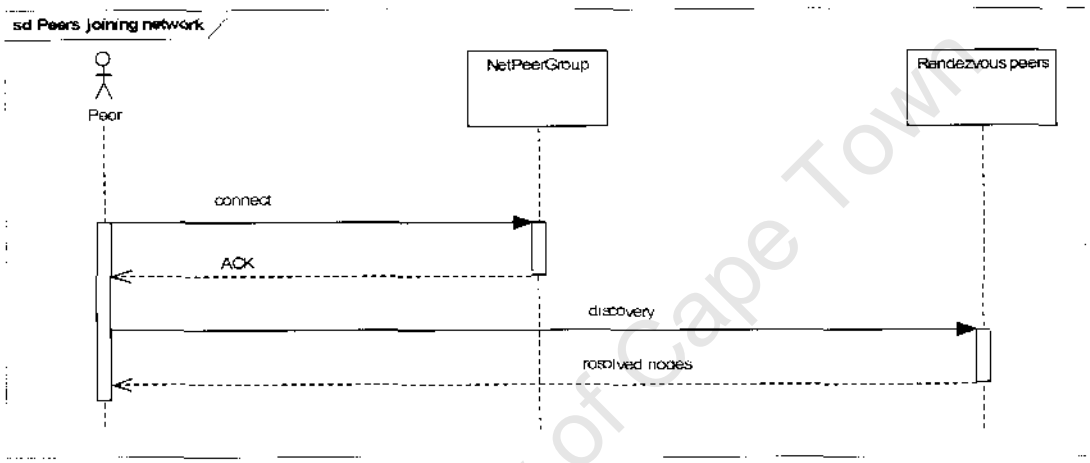


Figure 3.3 Sequence diagram of Peers joining the network

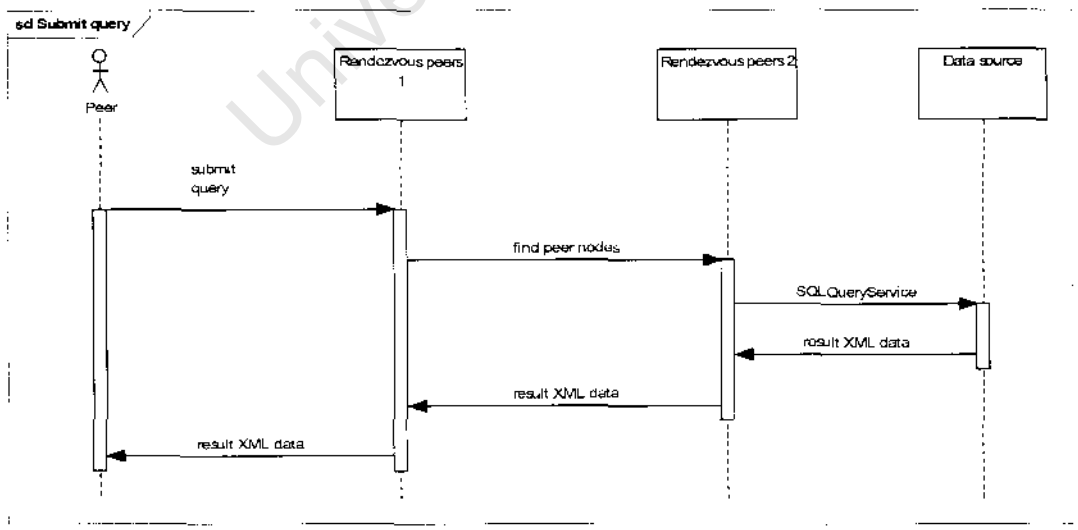


Figure 3.4 Sequence diagram of data retrieval

3.3 Shared Resource Distributed Index

The Shared Resource Distributed Index (SRDI) improves scalability by creating a loosely consistent, fault resilient, distributed hash table. The Distributed Hash Table (DHT) implementation helps to resolve distributed queries. All peers in the network will be identified by an index of the peer advertisement created using the SRDI service. This will assist peers in efficient query lookups. SRDI is part of JXTA. When a peer is brought online or has some content to share it will create an advertisement to publish the existence of a peer resource. This advertisement is indexed using the SRDI of that peer and pushed to its rendezvous peer. The DHT function is applied and a mapping is created of the index to a rendezvous peer. The index is then stored in that rendezvous peer and two other peers next to that rendezvous peer. Edge peers reside on the border of the JXTA network. When a search or discovery is initiated the edge peer sends an advertisement to its rendezvous peer and other peers in its subnet. The query will look for a match based on search criteria by sending the message to all peers in its peer group.

The SRDI is a service provided in JXTA 2.0 for edge peers to index their advertisements on appropriate rendezvous peers. An edge peer maintains a list of rendezvous peers in its local cache which it will try to establish a connection with.

3.4 Publishing an advertisement.

Each edge peer is associated with a rendezvous peer that helps to discover resources on the P2P network. When a peer (P1) is brought online it will publish advertisements to its rendezvous peer (R2) as shown below in figure 3.5. The advertisements created are the ModuleClassAdvertisement and ModuleSpecAdvertisement together with a pipe advertisement to connect to the service. "through the pipe messages can he sent between peers without having to know anything about the underlying infrastructure" [I 6]. The input pipe advertisement will allow messages to be received from clients, which are other peers in the network requesting the service.

The SRDI service creates an index of the advertisements using identifiers defined in the advertisements. Each rendezvous peer has a rendezvous peer view (RPV) which is an ordered list of other known rendezvous peers in its peer group. The rendezvous peer will use a Dar function (H (adv1)) to map the index derived from the SRDI service to a rendezvous peer in its local RPV.

As shown in figure 3.5 if the function returns R5 for example it will push the index to that rendezvous peer. The index can he replicated to other rendezvous peers to increase the probability of the peer being located. Using a replication distance of e- L -1) the index can be replicated to rendezvous peer R4 and R6.

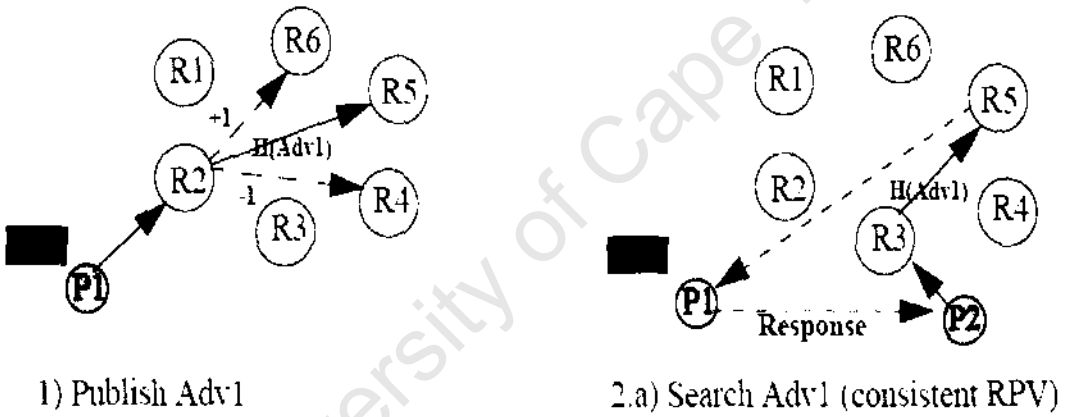


Figure 3.5 Rendezvous super peers [41]

3.5 Querying advertisements

The client will try to locate the service in its local cache before executing a remote call to discover the service.

The getRemoteAdvertisements() method takes 5 arguments shown below:

- java.lang.string peerid — ID of a peer to send query to; if null, propagate query request
- int type — DiscoveryService.PEER, DiscoveryService.GROUP , DiscoveryService.ADV
- java.lang.string attribute — attribute name to narrow discovery to
- java.lang.string value — value of attribute to narrow discovery to
- int threshold — the upper limit of responses from one peer

As shown in figure 3.5 if the edge peer P2 is looking for advl it will issue a resolver query to its rendezvous peer R3. The DHT function (H (advl)) is computed to locate the rendezvous peer that contains the index for advertisement advl.

Since it has responded with R5 its rendezvous peer R3 will forward the request to R5 which will send the request to P1 to respond to P2.

The distributed index is maintained by rendezvous peers. For each advertisement (advl) publication a peer sends an SRDI message of its advertisement to its rendezvous peer. The DHT function (H (advl)) is computed to store the index value on the relevant rendezvous peer. The input to creating an index is the peer id.

- rendezvous peer computes hash function of the index
- determines which rendezvous peers it must send the index to

The rendezvous peer stores the index for the peer in its SRDI and sends the index to its neighbours.

XML documents are used to represent the peer advertisements on the network. The distributed algorithm of the SRDI creates an index of resources in the peer to peer network. 'Hlls helps the SRDI to rapidly locate a peer containing a required advertisement. Adverts are used to locate a peer on the network since they can contain the source peer's identification information. A

requesting peer publishes an advertisement with its search criteria. This advertisement is matched to published advertisements to retrieve the data.

Chapter 4 Survey

In order to evaluate whether there is a need for the peer to peer information sharing system a questionnaire was formulated to gather user responses. The questionnaire was inexpensive, easy to setup and speedily provided us with information.

4.1 Description of Survey

The survey was conducted using a questionnaire to establish whether there is a need for peer to peer information sharing applications. The initial draft questionnaire was tested by obtaining feedback from friends and family which helped to revise the experiment before being taken to a larger audience. The questions were clearly defined in the survey and explained to participants if required when the author was present (e.g. when less literate subjects were involved). The experiment makes use of structured questions asked of participants using a custom built online web based application. This helped with reaching a larger target audience and easily collecting information from respondents. The respondents to the survey were only able to login once onto the online application using a password protected userid which eliminated the manipulation of results. Once the questionnaire has been completed the participants could access the results on the Internet which improved the response rate to the survey. The survey results are stored in a database and presented using graphs.

The online survey was inexpensive and covered a wide range of users. Due to the large number of participants in the survey percentage of responses is provided rather than actual numbers of respondents.

6. How often in the day would you want to access this information?

A Once

B $^{<5}$ times

C> 5 times

7. Do you have information on your computer that you would like to share with others?

A Yes

B No

8. What device would be convenient for you to access the information?

A laptop B desktop C cellphone

9. If there was an Internet based application that could provide most of your information requirements such as weather reports and news would you consider using it?

A Yes

B No

C Maybe

4.2 Participants

The sample used for the online survey was a random sample. collecting information from seventy respondents. People who would he unable to answer the online survey because of the technical difficulty were assisted by the author to ensure that the survey covered skilled and unskilled individuals.

The participants approached directly included:

- Wits University students from different faculties.
- IT and non IT company employees across different functional areas such as admin, HR, finance, marketing, facilities, procurement and IT.
- Shopping centre customers in a rural township as well as in an affluent suburb.
- Family and friends.

Some of the participants of the survey were making use of computers on a daily basis whilst others had very little or no computer experience. However the need to access information is common throughout the sample studied.

4.3 Survey results anti discussion

1. How often do you make use of the internet?

The question identifies the pervasiveness of the internet. 43.5% of the respondents make use of the Internet on a daily basis to acquire information, while 30.6% of them utilise the internet on a weekly basis. 16.1% of the respondents have never used the internet. This was due to some of the sample participants being from an unskilled labour force with minimal education and no access to the web.

2. What is the main source you use to locate information (such as weather reports and news)?

A reasonable number of the sample (33.8%) makes use of the internet as their main source of information. Since there were more internet users identified in question 1 (over 74%) it may be worthwhile noting that some of those internet users do not use the internet as a primary source for access to weather reports and news. This could either be due to limited access to the internet when required or other sources being more favourable. 21% of the participants make use of newspapers as a primary source of information. The newspaper is probably easily accessible for some participants due to companies purchasing daily copies of leading newspapers that could be shared between several individuals. 19.4% of the respondents use the television for locating current information. Televisions are widely available in most homes across poorer and affluent communities. All the free to air channels broadcast morning news shows between 6am and 8am daily cover weather reports, business news as well as current affairs. However due to show times it may not attract a sufficient audience as compared to the radio. The radio attracts 25.8% of the respondents as a primary source of information. Most radio channels have very well defined news reports providing important information to its audience. Due to time spent in transit to work the radio is an efficient medium to communicate the required information. The

sources of information above such as radio, television and newspaper may not provide the flexibility of the type of information that users want. The Internet however can provide personalization of content and selective retrieval of information when required. The ubiquity of the internet will provide up to date information to users when requested and to a variety of endpoint devices (cellphone, laptop, desktop).

The following questions are based on the response to question 2 identifying the primary source of information.

3. Is the data easy to locate?

Just over half the respondents (51.6%) have suggested that the information using their main source of information is easily accessible. The participants are probably very familiar with using their source of information and possess the ability to promptly access relevant content. However 37.1% of the participants in the survey found difficulty in accessing relevant content. Based on the respondents that use the internet as a primary source of information retrieval, 85.7% of them found it easy to locate their required content. Among the respondents using other mediums (non Internet) such as newspapers, televisions, and radio only 34.2% found it easy to locate the required content. This could be due to the inflexibility of using mediums such as radio, television and newspapers to acquire information. The information sharing application must address the need of easily accessing relevant content to be successful.

4. How long does it take on a daily basis to acquire this information?

This question aimed at identifying the amount of time spent using the respondents primary source of information to obtain the information. 41.9% of the participants take less than ten minutes to obtain the relevant content. Once the source of information is available the retrieval of information is fairly quick due to frequency of use and familiarity with the source. The remaining 58.1% of respondents take between 10 minutes and 24 hours to acquire the content. 95.2% of the respondents that make use of the interne[as their primary source of information retrieval take less than ten minutes to obtain their data. In contrast only 14.6% of the respondents using media (non internet) such as newspapers, televisions, and radio take less than ten minutes

to obtain their data. The results suggest that the use of non internet medium to obtain content is time consuming. Participants either do not have the urgency to obtain the data or have difficulty in getting to their source.

5. Is the information presented to you in a convenient format from your data source?

Most of the participants in the survey (53.2%) did not find the information required presented in a convenient format. 66.7% of the respondents found the internet to provide the information in a convenient format. However only 36.6% of the respondents using media (non internet) such as newspaper, radio, and television find the information in a convenient format. Since the sources of information were targeting a broad audience it is difficult to create an easy to use format of the data mainly due to the inflexibility in the data source. This is a key driver to enhance the user experience through an information sharing application.

6. How often in the day would you want to access this information?

On a daily basis most of the participants (45.2%) want to access the data frequently. A similar number of participants only require the information once a day. 19.3% of respondents rarely access data (< 5 times) such as news and weather reports. 84% of the respondents that make use of the internet as a primary source of information retrieval want to access to the information frequently (> 5 times). This may be attributed to their need to be informed and their experience of easily accessing up to date information. On the other hand only 26.8% of the respondents using non internet medium such as newspaper. television and radio require frequent access to information. This is probably due to the difficulties expressed in question 5 as well as reaching their source. The frequent requirement for data access can make a web-based information sharing application useful to participants.

7. Do you have information on your computer that you would like to share with others?

In the survey 43.2% of respondents do not possess any data that they would like to share. The lack of sharing is not suited to our goal of creating a peer to peer information sharing application. This may be due to the sample in the survey controlling sensitive information, not trusting the security mechanism of internet data sharing, and fears of intrusive behaviour of the application. These considerations should be considered when providing the peer to peer data sharing system to users. Overall, a majority do have data they would consider sharing, making the project worth pursuing.

8. What device would be convenient for you to access the information?

The data show that 45.2% of the respondents prefer to make use of cellphones to receive information. The cellular phone penetration in South Africa is high and accessible to even poor communities. The extensive mobile networks and availability of data services such as the intemet on most cellphones provides an excellent platform to receive information. 32.3% of participants would prefer using a laptop to access the information. Desktops are becoming less popular with 22.6% of the respondents preferring to access the information using this medium.

9. If there was an internet based application that could provide most of your information requirements such as weather reports and news would you consider using it?

66.1% of the participants in the survey would consider using an internet based information sharing application. This is a considerable amount of respondents that are eager to consider a service fulfilling their data requirements. However there may be concerns relating to security, intrusive behaviour of the application, personalisation of service, and relevance of content delivered by the application to be addressed. A small number of participants (21%) did not see the need for such an application. The remaining 12% of the respondents was not sure whether they would be interested in such as system.

4.4 Summary of findings

The survey provided a good insight into the need for a peer to peer data sharing application. The questions answered by a random sample of respondents have identified the need for such a system and some basic user requirements. Most of the current sources of information studied such as radio, newspaper and television may not provide the flexibility, frequent access of up to date content and format of information required by the respondents. The ubiquitous interne[and cellphone penetration in South Africa covering rural and urban areas provides an ideal platform to push customized web content to users. Based on the findings or the survey there is a need for peer to peer information sharing applications. This leads to my implementation of dbShare a simple data sharing peer to peer system.

Chapter 5 System Design

The main goal during the design phase was to create a system to permit data sharing in a peer to peer network in such a way as to ensure interoperahility across peers, for peers to access data from a variety of data sources, and to provide for simple and flexible data retrieval. The architecture is based on both SQL query access as well as web services access. This chapter gives an overview of dbShare and its user interface.

5.1 Overview of dbshare system.

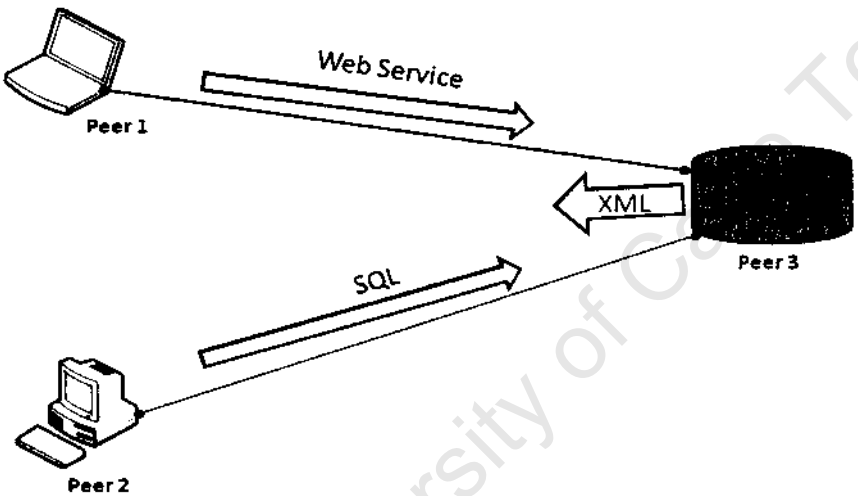


Figure 5.1 data sharing among peers

The dbShare peer-to-peer information sharing system runs on top of standard database management systems and/or web services, allowing it to connect peers and data sources on the network, as shown in figure 5.1. The system provides peers with an interface to enable easy and flexible retrieval of shared content and metadata. the JXTA client located at each peer makes use of either SQL or Web Services to retrieve data. Peers can use different kinds of request, from XPath expressions to SQL syntax or a URL (Uniform Resource Locator) in order to query data on the network.

Data that is returned from the peer servicing the request is returned in XML format. The system is based on the JXTA framework which provides peers on the network with the ability to easily locate other peers satisfying query needs. The system enables distributed devices to share data through the use of JXTA protocols, advertisements and services.

The following types of information can be shared using dbShare:

- Database metadata
- Database data (such as relational database views)
- Web service metadata
- Web service data

For simple access to data the entire source is retrieved; alternatively SQL or XPath can be used to extract only part of the data meeting specific search criteria.

Some of the issues identified with existing JXTA P2P Database implementations are:

- Peers can misinterpret the data that is retrieved from remote peers
- Only peer data in a relational database may be shared
- Web services are excluded

The data accessed from remote peers may be misinterpreted. DbShare metadata adverts provide semantics such as each attribute's name, data type, integrity constraints, textual descriptions, synonyms and sample values that a data administrator can supply. The dbShare implementation through the use of a java program and web services can support sharing data from non relational data sources such as spreadsheets. The data from the non relational source can be converted to XML format and exposed by the peer through web services. These web services may be advertised and invoked by other dbShare peers seamlessly sharing information.

Relational data provides a flat structure while XML data can he structured and hierarchical in nature. DbShare thus allows different ways to query data using either SQL or XPath, to cater for more types of data and for user interfaces that are either flat or hierarchically organised. If SQL is used, search criteria are incorporated into the SELECT statement before it is sent out on the

network. An XPath expression is applied to the returned XML at the client to allow arbitrary querying of web services and database information.

In a production system semi/automated SQL- and XPath-generation components should be provided, but this is out of scope for the project. Some simple queries are constructed from user input, but complex queries need to be supplied in SQL.

The logical architecture for dbShare follows the approach of [50] as shown in figure 5.2. It contains multiple layers allowing the P2P application to share data.

The wrapper manages connectivity to and from the source database allowing interoperability with different databases such as Oracle, MySQL ., and SQL Server. This layer abstracts over the DBMS access and the creation of XML results, allowing the Data Manager to simply supply a SELECT statement and receive the response correctly formatted.

The JXTA layer is used for all the node's activities on the network such as peer discovery, creating pipes with other nodes, sending queries and obtaining the query results. This layer provides an abstraction which hides the JXTA API, as well as the XML creation or parsing that is required before or after certain JXTA calls.

The data manager is the middle layer that receives calls from the User Interface, JXTA and Wrapper Layers and invokes the corresponding operation in another layer. The data layer is an abstraction over the data source and translates and formats requests as required. It performs the following functions when queries are initiated from the peer:

- receives queries from the user interface,
- transforms the query into an appropriate format,
- sends queries to the JXTA input pipes of other peers

The data manager layer performs the following functions when responses are received:

- receives response from the output pipe,
- translates the data received,
- applies any XPath expression if required.
- sends the data received to the user interface

The user interface is used to obtain administrative and query requests and! or display the result set from peers on the network.

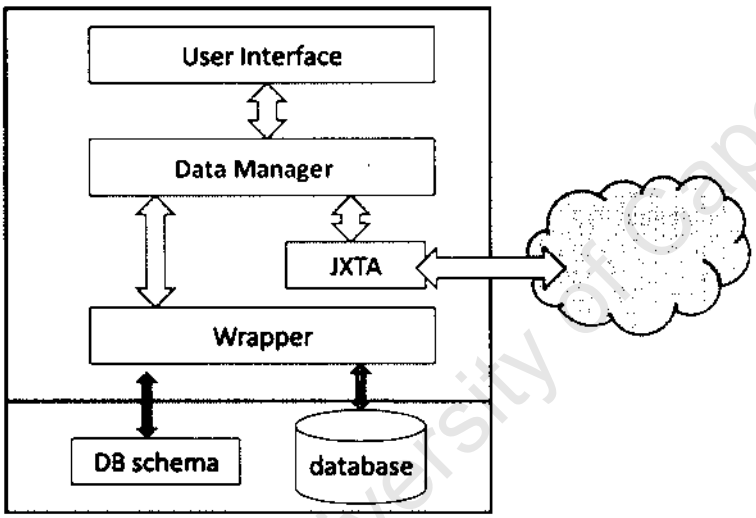


Figure 5.2 dbShare architecture [50]

5.2 Using dbShare

The system possesses 4 interfaces for peers to access data.
This is through

L Metadata access
2. SQL queries
3. Web service invocation
4.)(Path access

Metadata access allows peers to discover what information is available. This can be done by simply retrieving names and descriptions of data sources, or by extracting full details of the attributes as well. Database relations or views are exposed to enable access to shared data through the use of advertisements. They can be retrieved in their entirety or specific data meeting criteria of interest can be extracted using SQL. For arbitrary queries, SQL can be complex since the schema and SQL syntax must be known prior to the peer initiating the request. However a peer with this capability is able to execute a variety of queries to access shared database content offering flexibility in the system.

A web service implementation eases the data retrieval process through its simple interface. However it is restricted in terms of what data can be requested, so XPath can be used to filter results.

These four interfaces together provide ease of use and convenient peer data retrieval. In dbShare peers with data to share also have the flexibility to expose content through either mechanism, namely an SQL interface or web services, or both.

The software installed on each peer is the JXTA software and the Java Runtime Environment (JRE) software.

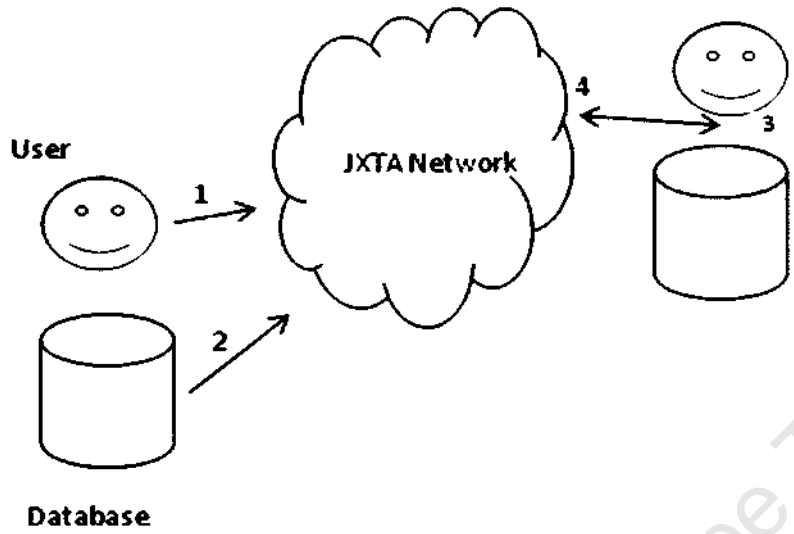


Figure 5.3 JXTA database sharing outline

As shown in figure 5.3 a user connects to the network W. The peer advertises its database (2) in the JXTA network and retrieves other database adverts. Once the appropriate advert has been received (3), other peers can make use of that advert to request data from the database (4). This request uses information from the advert to send an instruction to the relevant peer.

The application creates and publishes input pipes using pipe advertisements when the database peer is launched into the JXTA network. The peers joining the network will discover these database peers and create output pipes to them.

The use of the JXTA protocol allows interoperability across peers with different operating systems such as Linux and Windows and different hardware devices. This is due to the Java programming language that is platform independent and executed inside a Java Virtual Machine (JVM). DbShare uses JDBC for interoperability across differed relational databases such as Oracle and MySQL.

5.4 Database metadata

When a peer wants to share database content separate advertisements are created with the metadata for each schema and web service to be exposed. The information in the advertisement for the SQL query interface is designed to capture all necessary details:

- *Creator Name* contains the name of the schema creator.
- *PeerlD* contains the peerlD of the creator peer.
- *Table* contains the metadata of a relation/view in the database schema.
- *Table Name* contains the name of the relation/view to be shared.
- *Column Name* and *Column lime* define attributes of the relation/view.
- *Integrity constraints* (optional) contain rules defining valid data.
- *Textual descriptions* (optional).
- *Sample values* (optional) contain common values of an attribute.
- *Synonyms* (optional) contain alternative names for an attribute.

5.5 Web service metadata:

The information in the web services advertisement contains the index and the peer id. It is a pointer to the peer that has the advert. Uniqueness is maintained since peer id will always be unique as it is maintained by JXTA. This advertisement was designed to capture all necessary details, viz.:

- Web service name
- Web service description
- Creator
- Peer ID
- Version
- Web service URI
- Textual descriptions (optional)
- Element names
- Synonyms for element names (optional)
- Sample values (optional)

The index hash value for the peer advertisement is located at rendezvous peers for both the SQL and web service implementations. The indexed advertisement contains the Peer ID as shown above to locate the source of the requested data.

5.6 High level user view

DbShare allows the sharing of the following types of data:

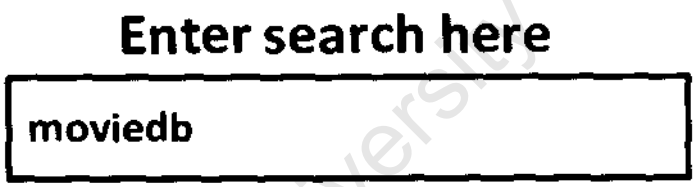
- Database metadata
- Web service metadata
- Data records

The user interface expects certain input data in order to be able to return the required information to the dbShare client. DbShare peers contain their own local cache which persist schema advertisements received from remote peers. When a query is performed the peer will evaluate the metadata in its local cache prior to querying for the data on the P2P network.

The kinds of input that can be entered from the user interface are shown below along with example output.

Input and Output scenarios

L Search for moviedb as shown below in the dbShare GUI (database access).



There is no select statement entered therefore dbShare needs to compile the query. It is not known whether the query will he serviced from a database using SQL access or web services access. JXTA will create an advertisement to locate the peer that contains the required data on the P2P network. The metadata that is published by the peer containing the required data will indicate if the source is a database table or a web service. Once the metadata is found dbShare will execute the query.

For the database query using SQL and retrieving all data in its entirety a query as shown below is created by dbShare.

```
select * from moviedb
```

2. Search for moviedb as shown below in the dbShare Gill (web service access).
DbShare will follow a similar process as shown above to identify the peer containing the requested data and if the data is exposed through a web service interface.

For the query using the web service implementation also retrieving all data in its entirety the web service URI is invoked as shown below.

```
http://41.11.9.202/news/resources/headlines
```

The response is returned as an XML document for both the SQL or web services access. The data manager transforms the)(NIL data into a format suitable for the user interface as shown in the output below.

Result Set

Title	Actor	Genre	Year
Transformers	Keanu Reaves	Action	2008
The Boss	Bruce Lee	Action	1996
Men in Black	Will Smith	Comedy	2002
Las Vegas	Robert Redford	Thriller	2001

Figure 5.4 dbShare results (all data)

3. User specific search for movies as shown below (database access) retrieve only data meeting given criteria.

```
select * from moviedb where actor ="Bruce Lee"
```

The query follows a similar process as shown in scenario 1 above to locate the peer with the data. The query specified is executed on the remote peer and the result is returned as an XML document.

Result Set

Title	Actor	Genre	Year
The Boss	Bruce Lee	Action	1996

4. Performing a join

Data required may be located in multiple data sources, therefore joins can be required. A peer may request data such as the example below using ajoin.

```
select title, actor, genre, year, director,
producer, revenue_earned
from moviedb, moviedb_survey
where moviedb.title=moviedb_survey
```

The query is decomposed, shown in the examples below.

```
select title, actor, genre, year from moviedb
```

```
select title, director, producer, revenue_earned from moviedb_survey
```

Messages are created and queries executed on the peers containing the data and then response messages are returned. The results from multiple sources are joined and then returned to the dbShare user interface as shown below.

Result Set

Title	Actor	Genre	Year	Director	Producer	Revenue Earned
Transformers	Keanu Reaves	Action	2008	Pat Smith	ABC	$250m
The Boss	Bruce Lee	Action	1996	Joe Knight	Hollywood productions	$135m
Men in Black	Will Smith	Comedy	2002	Ace Santana	Earth Inc	$200m
Las Vegas	Robert Redford	Thriller	2001	David Turner	Warner Bros	$66m

5. Peers publishing metadata

Once a peer joins the dbShare network it needs to publish its database schema or web service metadata for other peers to access its shared data. This is achieved by clicking a button as shown below in the dbShare GUL

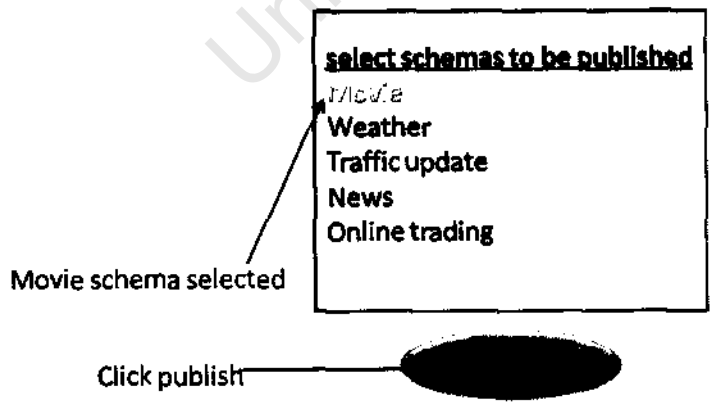


Figure 5.5 publish schema example

A separate interface is used to provide metadata of the content peers would like to share. The metadata is then published so that it can he queried by remote peers. It contains information as shown in section 5.4 (for database metadata) and section 5.5 (for web service metadata). Metadata such as attribute names and types can be automatically generated from database schemas, while other information such as synonyms and textual descriptions need to be entered by the user.

Chapter 6: Implementation

A prototype implementation of dbShare based on the JXTA P2P protocol was built using relational database access through SQL queries and web services access. Data sources arc accessed either in their entirety or by searching for specific data of interest.

6.1 JXTA Implementation

The peer to peer system dbShare enables the sharing of data while maintaining interoperability of the system. The peers in the network typically possess their own local data, with the information in these data sources shared amongst peers. The solution encompasses a distributed architecture such that peers can be either clients or servers depending on whether they are consuming or providing services.

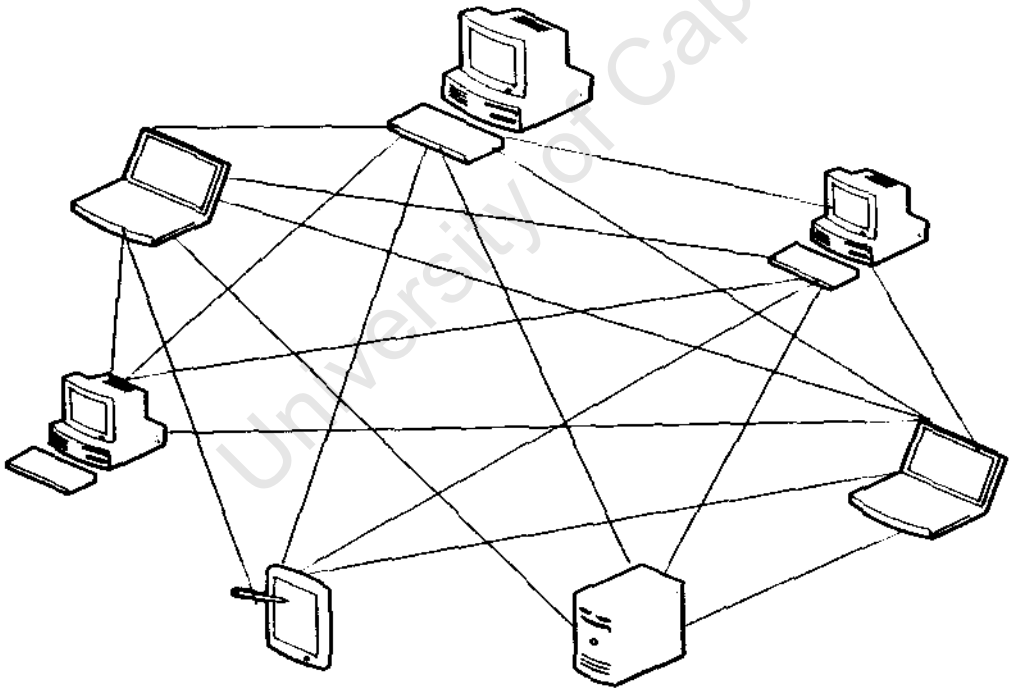


Figure 6.1 dbShare peer to peer network

The database peer advertises a pipe using an advertisement like the following. Each of the pipe advertisements has a unique ID which is maintained by JXTA.

```
<?xml version="1.0" encoding="UTF-8"?>
<jxta:PipeAdvertisement>
        <Name>DSSDatabaseDataSaveInputPipe</Name>
         <Id>urn:jxta:uuid-CE19C9353ED641B6A1
          879527667BBC32699352C32E0D40579C17A7CCB7CE781C04</Id>
         <Type>JxtaUnicastSecure</Type>
</jxta:PipeAdvertisement>
```

When an SQL query is initiated JXTA will locate the pipe advertisement of the peer containing the requested data. The output pipe that is created will be used to acquire the data from the database.

The example below shows the ModuleClassAdvertisement and ModuleSpecAdvertisement that is created for a headline web service.

```
// creates the new advertisement
ModuleClassAdvertisement mcadv = (ModuleClassAdvertisement)
AdvertisementFactory.newAdvertisement(
ModuleClassAdvertisement.getAdvertisementType());

//creates a jxta module class advertisement headlines
mcadv.setName("JXTAMOD:headlines");
mcadv.setDescription("News headlines web service in Gauteng, SA");

//initialises the advertisement
ModuleClassID mcID = IDFactory.newModuleClassID();
mcadv.setModuleClassID(mcID);
```

The advertisement is created and initialized, published in the local cache and propagated to rendezvous peers. The ModuleSpecAdvertisement is created that contains all the information required for a client to connect to the service.

```
// creates the new advertisement
ModuleSpecAdvertisement mdadv = (ModuleSpecAdvertisement)
AdvertisementFactory.newAdvertisement(
ModuleSpecAdvertisement.getAdvertisementType());
```

```
// initialises the name, version, etc. of the service
mdadv.setName(headlines);
mdadv.setVersion("Version 1.0");
mdadv.setCreator("Suhen Pather");
mdadv.setModuleSpecID(IDFactory.newModuleSpecID(mcID));
mdadv.setSpecURI("http://localhost:8080/news/resources/headlines/");
```

A peer will query data from associated peers for database metadata, web service metadata and data records.

6.2 Schema Advertisement

The SQL query interface utilizes schema advertisement messages to provide database metadata to associated peers on the network. Once the content is ready to be shared by a peer XML messages are sent to all peers in its peer group using the peer resolver protocol (PRP). These advertisements are exploited by the SQL query interface and indexed using the SRDI for rapid peer discovery.

Web services metadata will be represented by an advertisement as described in section 5.5. The medical_records table e.g. may be represented by an XML document listed below giving its metadata (optional elements omitted).

```
<SchemaAdvertisement>
<CreatorName> SuhenPather</CreatorName>
<PeerID> Peer101</PeerID>
<Table>
<TableName> medical_records</TableName>
<Column>
<ColumnName>name</ColumnName>
<ColumnType>char(20)</ColumnType>
</Column>
<Column>
<ColumnName>surname</ColumnName>
<ColumnType>char(20)</ColumnType>
</Column>
<Column>
<ColumnName>id</ColumnName>
<ColumnType>number</ColumnType>
</Column>
<Column>
<ColumnName>allergy</ColumnName>
<ColumnType>char(20)</ColumnType>
</Column>
</Table>
</SchemaAdvertisement>
```

Listing 1

DbShare interfaces

The dbShare information can be accessed from the frontend in two ways:

- SQL queries
- Web services

6.3 Client application

Queries are initiated from a client application which sends either a SQL or web service request to the peer where the data source is located. The destination peer accepts the request from the client. It processes the query and returns the formatted result to the query initiator. SQL enables the data to be queried and manipulated at the logical level. The dbShare client has the ability for peers to perform data queries. This can be a command line tool or a web based user interface. The web interface allows queries to he sent from a web browser and directed to the distributed data sources to fulfill the query requests.

The metadata shown in listing I is used to formulate the SQL query. There is no need for a peer to understand the SQL syntax when simple access is carried out. DbShare will ensure that for simple access all data is retrieved from a peer, similar to performing the following SQL query:

```
select from moviedb;
```

When a peer requires flexible query access knowledge of the SQL syntax is required however it empowers the peer to query data of interest. The SQL query interface offers flexibility in the network since a peer may be able to access specific content such as name and surname of a patient extracted from the medical records table, as shown below.

```
select name, surname from medicalrecords where Id=100;
```

The web service implementation will enable access to underlying shared database content from peers using a simple URL.
(E.g. http://localhost:8080/news/resources/headlines /).

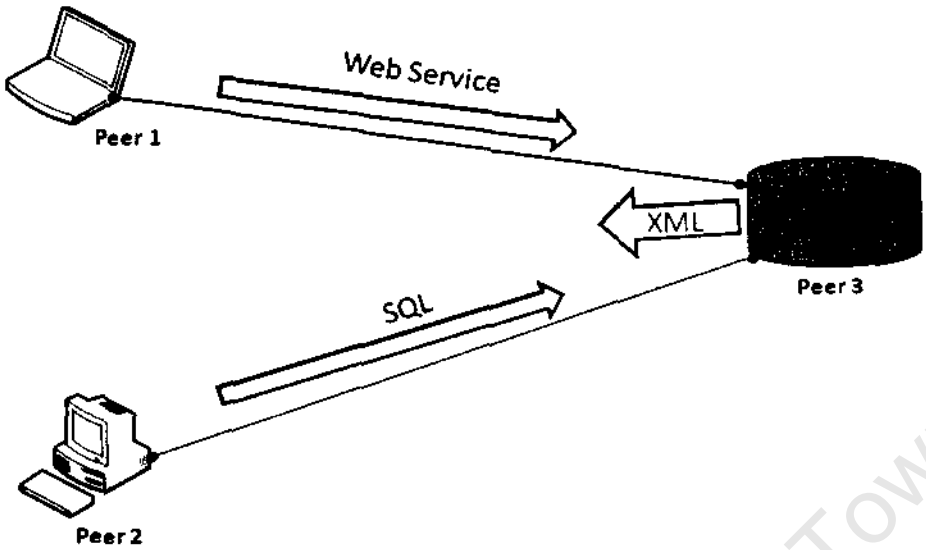


Figure 6.2 data sharing among peers

The metadata information about peers on the P2P network is used to determine if the data that is shared require SQL access or web service access. If the data retrieval requires SQL access the select statement is used otherwise the URI is initiated for web service access.

6.4 SQL Query Implementation

The dbShare system enables access to heterogeneous data sources. Each of these databases such as Oracle, SQL Server. MySQL. and DB2 has different datatypes, interfaces, and capabilities which bring about several challenges. Through the use of JDBC however, interoperability has been maintained.

Display 10 Save Run

```
select * from emp
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7839	KING	PRESIDENT	-	17-NOV-81	5000	-	10
7698	BLAKE	MANAGER	7839	01-MAY-81	2850	-	30
7782	CLARK	MANAGER	7839	09-JUN-81	2450	-	10
7566	JONES	MANAGER	7839	02-APR-81	2975	-	20
7788	SCOTT	ANALYST	7566	09-DEC-82	3000	-	20
7902	FORD	ANALYST	7566	03-DEC-81	3000	-	20
7369	SMITH	CLERK	7902	17-DEC-80	800	-	20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30

Figure 6.3 SQL query interface and results

Figure 6.3 shows an example of a command line SQL query interface that was built as part of the prototype for both simple and complex (flexible) access. The simple access mechanism using the SQL syntax retrieves all the data from a database table. SQL queries are sent to the database peer and XML data are returned. DC A is used for peer discovery and to send SQL queries to peers satisfying query criteria and return XML data.

Once the XML data is received from the source nodes the data is stored in memory at the requesting peer node to be sorted and filtered. The data may be paged to disk if the result set is too large.

The JXTA peer group is a virtual grouping of peers that enables SQL and web services frontends to share data through the use of the peer resolver protocol (PRP). The peer group contains peers which have common goals and interests. The protocol allows messages to be sent to peers in the peer group as well as receive the corresponding message response. SQL queries are wrapped in a generic format by the 1X FA peer resolver service [49]. By means of the resolver protocol it sends and receives message to and from peers in the peer group [49].

XML messages are exchanged between peers during the request and response process. Different options to construct query and response messages were evaluated and the solution proposed by N.Zhang [45] was adopted. Two types of messages namely the ResolverQueryMessage and ResolverResponseMessage are exchanged between peers. The format for the ResolverQueryMessage is listed in figure 6.4 below and the ResolverResponseMessage is listed in figure 6.5.

The Credential is for verification of the sender's details
The HandlerName is a string specifying how the query should be handled
The Query LD is the ID of the query
The SrcPeerlD is the ID of the peer initiating the query
The Query is the search that is initiated

The ResolverQueryMessage contains the query to be sent to peers in the peer group.

```
<jxta:ResolverQuery xmlns:jxta="http://jxta.org">
    <HandlerName>...</HandlerName>
    <Credential>...</Credential>
    <QueryID>...</QueryID>
    <SrcPeerID>...</SrcPeerID>
    <Query>...</Query>
</jxta:ResolverQuery>
```

Figure 6.4 ResolverQueryMessage format [45]

The ResolverResponseMessage sends the response back to the querying peer.

```
<jxta:ResolverResponse xmlns:jxta="http://jxta.org">
        <HandlerName>...</HandlerName>
        <Credential>...</Credential>
        <QueryID>...</QueryID>
        <Response>...</Response>
</jxta:ResolverResponse>
```

Figure 6.5 ResolverResponseMessage format [45]

The contents in the DBQueryMessage shown below are extracted from the ResolverQueryMessage XML document.

```
<xsd:element name="DBQueryMessage">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="QueryID" type="xsd:string"/>
      <xsd:element name="TableName" type="xsd:string"/>
      <xsd:element name="QueryMessage" type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

Figure 6.6 DBQueryMessage schema [45]

As shown in figure 6.6 above the DBQueryMessage contains the QuerylD, TableName. and Query Message [45]. These are part of J XTA with java packages that perform these tasks.

The QuerylD is a unique identifier for the query.
The TableName is the relation/view that is to be queried by the peer.
The Query Message contains the SQL query statement.

The following algorithm shows the query propagation process for a peer group [45]:

- Receive single table query from user. (Multi-table is also supported which uses a join)
- Assign a query ID to the query
- Construct DBQueryMessage
- Generate ResolverQueryMessage from DBQueryMessage

An example DBQueryMessage is shown below:

```
<DBQueryMessage>
        <QueryID>q001</QueryID>
        <TableName>medical_records</TableName>
        <QueryMessage>
        Select name, surname, id, allergy
        From medical_records
        </QueryMessage>
</DBQueryMessage>
```

The following algorithm shows the process to return the results [45j:

- Receive ResolverQueryMessage
- Extract DBQueryMessage
- Parse DBQueryMessage
- Extract QueryMessage (SQL syntax)
- Receive query result from local database based on QueryMessage
- Structure query results into DBResponseMessage
- Generate ResolverResponseMessage to the querying peer

An example DBResponseMessage is shown below:

```
<DBResponseMessage>
        <PeerID>PDB001</PeerID>
        <QueryID>q001</QueryID>
        <TableName>medical_records</TableName>
        <QueryResult>
                <Row>
                        <name>Suhen</name>
                        <surname>Pather<surname>
                        <id>11223344</id>
                        <allergy>flu</allergy>
                </Row>
                <Row>
                        <name>Joe</name>
                        <surname>Bloggs</surname>
                        <id>11223345</id>
                        <allergy>eczema</allergy>
                </Row>
        </QueryResult>
    </DBResponseMessage>
```

The DBResponseMessage contains the XML result which is constructed into a ResolverResponseMessage XML document and returned to the requesting peer. Peers are found through advertisements published in the PSP network and distributed metadata located in the peer's local cache or at rendezvous peers.

6.5 Web Services

The application may also make use of web service technologies to exchange information between peers on the network that is built on technical standards defined by the World Wide Web Consortium (W3C). Our implementation makes use of the Java API for XML Web Services (JAX-WS) version 2.0. The standard ensures a simplified yet powerful approach for web service deployment.

Prototype web services were built using the RESTful approach. However the SOAP-based approach may also be used. Entity classes are created from the data source and contain the attributes of the relational data tables. Object-relational mapping is implemented using the Java Persistence API, specifically the open source Apache OpenJPA. In NetBeans when the entity classes are generated from the data source a persistence unit is created. The configuration of the persistence unit is stored in a file called persistence.xml containing login details, and data source. Once the entity class is created, the RESTful web service may be created which presents the database content in XML format. The web service's WSDL information describing the web service is published in the JXTA advertisement.

6.6 XML data transformation

The XML Object Model (XOM) is implemented in the dBShare prototype to provide flexibility for web services allowing queries to be applied to retrieved data. XML Object Model (XOM) is an open source tree based XML processing API which is similar to DOM. Lt was found to be more efficient and simpler to use with a more intuitive interface compared to other parsers such as SAX and DOM.

The XML Object Model in the dbShare prototype is applied at the client requesting the data (client side processing) in order to transform the XML content retrieved into an appropriate subset and format. Data is retrieved in its entirety from the web service. The required subset and format of data at the requesting peer is known therefore XOM may be applied at the client to transform the XML document appropriately. The result set will therefore match what is expected.

The use of)(OM at the peer servicing the request (server) can also he supported. The benefits are that reduced network traffic is returned to the requesting peer since transformation occurs at the server. This may speed up the data retrieval. However this is not implemented in order to decouple data access from data transformation, which is applied after a response is received by the data manager on the client side.

The open source XOM implementation improves interoperability, performance and simplicity when used to manipulate XML content. Some of the nodes in the tree can be processed while the document is being built which enables dual streaming and programs to operate almost as fast as the underlying (SAX) parser can supply data. The application does not need to wait for the document to be completely parsed before it can start working with it.

XOM allows for filters to be provided, so that only relevant parts of the document are parsed. Lt also effectively allows for documents to be processed piece-by-piece, thus providing some of the benefits of the much lower-level SAX parser, enabling the processing of massive documents that would not otherwise fit into memory.

However XOM is not a standard part of Sava SE and therefore to use XOM its libraries must be bundled with the application. Ln addition, this may pose a problem to applications deployed on non-standard devices, such as cellular phones.

Chapter 7 Testing

This chapter outlines the testing of the JXTA implementation for node discovery, peers joining and leaving groups, peers sending and receiving queries and obtaining query results. It also includes the testing of the dbShare prototype. The database access has been tested as well as the web services. Queries were sent across the network to access databases and web services.

JXTA

A simple command line interface was first used for testing JXTA. Lt was convenient to demonstrate and examine the JXIA peer to peer network. A small local area network (LAN) was set up with each machine having a unique LP address. The lab contained 3 machines that were used for testing.

Once JXTA is installed, peers may be created to enable communication. .JXTA may be invoked from a Java program since its reference implementation is in the Java programming language. A JXTA shell is an application built on top of the JXTA Java binding to demonstrate the implementation of JXTA applications.

The JXTA shell is used to enable peer discovery, group creation and related peer management tasks.

The following creates a pipe advertisement for a peer, which can he inspected via the variable *myadv.*

```
JxtA>myadv    mkadv -p
```

Once the advertisement is created it must be published to be made available to other peers in the P2P network using the share command.

```
JXTA>share myadv
```

Once peers are created on the .LX TA P2P network they possess the ability to create, join or leave groups. The example below illustrates the creation of a peer group named myGroupL

```
JXTA>mygroupadv = mkadv -g myGroup1
JXTA>mkpgrp -d mygroupadv
JXTA> groups -r
group discovery message sent
JXTA>groups
group0: name = NetPeerGroup
group1: name = myGroup1
JXTA>
```

'the peer can join the newly created group my Group] as shown below:

```
JXTA>join -d group1
Stopping rdv
Enter the identity you want to use when joining this peergroup (nobody)
1Identity: jra
JXTA>
JXTA>groups
JXTA>join
 Joined Group    : myGroup1        (current)
 Joined Group    : worldgroup
 Joined Group    : netgroup
```

Peers on the P2P network can be listed by executing the "peers' command. The name of the peer in this example is "1ª2P Thesis' as shown below.

```
JXTA>peers
peer0: name = P2P Thesis
```

An XML document can be obtained as shown below representing information on peer nodes. Information such as peergroup (to which peers belong), unique peer LD. and IP address of peers may be presented.

```
JXTA>whoami
<Peer>P2P Thesis</Peer>
<PeerId>urn:jxta:uuid-59616261646162614A78746150325033D8673F9DF0F24CAFBB62A9BBC1B8EB3903</PeerId>
<TransportAddress>jxtatls://uuid-
59616261646162614A78746150325033D8673F9DF0F24CAFBB62A9BBC1B8EB3903</TransportAddress>
<TransportAddress>tcp://10.231.121.28:9701</TransportAddress>
<TransportAddress>relay://uuid-
59616261646162614A78746150325033D8673F9DF0F24CAFBB62A9BBC1B8EB3903</TransportAddress>
```

A user peergroup may be created by peers to satisfy a particular data sharing need such as for peers to share movies. Once the peergroup is created a peer must discover and join the group to communicate with the peers in the group.

The following command sends a peer discovery message across the network.

```
JXTA>peers -r
peer discovery message sent
```

There are two peers on the P2P network now, namely "P2PThesis 1" and "P2P Thesis 2" that are discovered.

```
JXTA>peers
peer0: name = P2P Thesis 1
peer1: name = P2P Thesis 2
```

Database SQL access

SQL queries like the following were used to test the dbShare system:

```
select * from medical_records where id=100;

NAME       SURNAME             ID ALLERGY
---------- ---------- ---------- ----------
Suhen      Pather             100 eczema
```

The example above shows the command line tool that was built in the dbShare prototype for SQL data access. The query illustrates the flexible access providing the requesting peer with the medical record for patient Suhen Pather (1D=100). In this example the peer providing the service will expose the medical_records table in the distributed catalog as metadata. Data of interest is retrieved and returned to the requesting peer. A simple GUL interface was also tested. Access to either a MySQL or a SQLServer database was tested to check interoperability.

Web services

Use of web services was tested with the Glassfish lava application server rendering dynamic content from the data source.

The "…yNews " web service accessed data in a relation with attributes category, popularity, reporter, and title.

The Firefox browser ᵛa s used to deliver the XML content when the myNews web service is requested. The XML output from the database is shown below retrieving data in its entirety.

Figure 7.1 myNews Web Service XML

Filtering was tested by reading data from the My SQL database and building an XML Object Model (XOM) from the data. Using the XOM APT, the retrieved document was parsed, and using the simple XPath API, the 'titles" were retrieved and printed to standard output.

There are 2 java classes shown, namely Headlines (listing 3) and NewsFinder (listing 4).

The Headlines Java class is the main java class which is the launcher application (listing 3).

The NewsFinder Java class is called from the Headlines Java class and takes the input stream which is the URL of the my News web service as shown below.

httpidocalhost:8080/news/resources/ MN News/

The data retrieved from the myNews web service is used to build an XOM tree structure. XPath is then used to query the data and retrieve the title attribute. An array list is created of the results and then returned to the Headlines launcher application. XOM is applied for data transformation to the XML document and returns the data. This can he as strings or XML.

The program is invoked by executing the news.Headlines Java code as shown below.

- Java news.Headlines

```
/*
* The news package
*/

package news;
import java.util.*;
/**
* A simple application to print the latest news headlines from a MySQL * database.
*/
public class Headlines
{
public static void main(String[] args) throws Exception
{
if (args.length < 1)
{
System.err.println("Expected: <source>");
System.err.println(" source: <url>");
System.err.println(" or 'slashdot', 'cnn', 'makezine'");
System.exit(1);
}
// if arg is not recognised source, treat as explicit URL
String url = newsSources.get(args[0]);
if (url == null)
{
url = args[0];
}
// Get the news headlines
NewsFinder finder = new NewsFinder();
List<String> headlines = finder.getHeadlines(url);
// Display
System.out.println("Headlines from News" + args[0]);
System.out.println("as of: " + new Date() + "\n");
for (String headline : headlines)
{
System.out.println("* " + headline);
}
}
// news source, accessible by keyword
private static final Map<String, String> newsSources =
```

```
new HashMap<String, String>();
static
{
newsSources.put("db", "http://localhost:8080/news/resources/myNews/");
}
}
```

Listing 3

```
/*
*News
*/

package news;
import java.io.*;
import java.net.*;
import java.util.*;
import nu.xom.*;
/**
* The component that looks up news headlines from a backend database, using the XOM API and XPath
*/
public class NewsFinder
{
/**
* Return the latest headlines as a List of strings
*/
public List<String> getHeadlines(String uri) throws IOException,
URISyntaxException, ParsingException
{
// Get an inputstream from the URI (could be file or network)
InputStream in = new URI(uri).toURL().openStream();

try
{
// Build a XOM tree
Builder builder = new Builder();
Document news = builder.build(in);

Nodes results = news.query("//*[local-name()='title']");

// Build the list of headlines
List<String> headlines = new ArrayList<String>();
for(int x = 0; x < results.size(); x++)
{
// Add the text contents of each title to the list
headlines.add(results.get(x).getValue());
}
return headlines;
} finally
{
// Try to close the stream
try
```

```
{
in.close();
} catch (Exception ignored)
{}
}
}
}
```

Listing 4

Data transformation results

```
$ java news.Headlines
```

produces the following output:

```
Headlines from Newsdb
as of: Wed Dec 10 17:06:17 CAT 2008

* Smart CIOs Innovate in tough times
* No funding for tech start-ups
* Black Hole At Center of Milky Way Confirmed
* When Teachers Are Obstacles To Linux In Education
* Paul McCartney Releases Album As DRM-Free Download
```


Testing Summary

JXTA, database SQL access, web service access and XPath filtering have been tested successfully in our lab environment, consisting of three peers in the dbShare network and a mix of web services, MySQL and SQLServer data sources.

Chapter 8 Conclusion

8.1 Summary

The thesis presents an architecture and implements a prototype of a peer to peer system (dbShare) using the JXTA protocol. DbShare is a distributed information sharing system where each peer can act as a client or a server depending on whether it is consuming or providing resources. 'Me application empowers users who are part of the dbShare network to share and query database and file content distributed across several peer nodes. The system architecture of dbShare ensures interoperability between different data sources accessible using SQL, XPath and web services.

A survey was performed through the use of a questionnaire that explored the need for a peer to peer data sharing application. The results indicated that more convenient formats for Internet data access was needed and people were interested in an information sharing application.

The thesis gives impetus to the creation of distributed database applications supporting user collaboration for knowledge and content sharing that is both location and technology independent.

Interoperability in dbShare is achieved through the use of XML. Java, JDBC and JXTA to access heterogeneous data sources which is desirable for data sharing applications. The experiment found JXTA to be a good basis for building peer to peer sharing applications. The software has several large implementations for sharing content and is highly customizable to suit different application requirements. However the JXTA platform is constantly being changed with the addition of new functionality and may be an immature product for production use.

Initially the thesis explored the use of SQL to query remote peers. The use of the web service approach proved to be also useful, offering peers easy access to shared data. There was no need for peers to understand the SQL syntax when using the web service interface and through the use of the XML Object Model (XOM) XPath is also supported, offering the flexibility of what data

may be retrieved, and how. A peer can thus access and deliver shared content in different ways. offering flexibility in the system. While other P2P database sharing systems exist and web services are now widely used, the dbShare architecture provides a simple and integrated way of using either approach.

The dbShare implementation based on the JXTA platform realizes the goals of providing interoperability for a simple and flexible information sharing application in a distributed peer to peer environment.

8.2 Future work

The use of schema matching to identify common elements in XM L and relational schemes across peers is beyond the scope of this project, and incorporating existing schema matching software is reserved for future work.

Similarly, an intelligent GUT component that generates SQL and XPath expressions dynamically is useful. However its implementation in dbShare is also out of scope for this project and reserved for future work. Lt will simplify the query process to accommodate users that are not well versed in the SQL or XPath syntax.

Limited security is provided by the dbShare application for ensuring trust between peers, namely that of the underlying DBMS and the use of a secure JDBC-ODBC bridge. Future research can investigate encryption of data between peers at lower levels of the OSI stack such as at the transport layer (TLS) as well as the support for public key cryptography which makes use of public and private (secret) keys for secure data transport.

The use of mobile devices in the dbShare peer to peer network may provide a valuable way to share data. The Java programming language employed in the prototype simplifies the process to use mobile devices. However the version of the peer to peer software to run on mobile devices is reserved for future work.

Bibliography

1. K. Aberer, P. Cudrè-Mauroux, A. Dana, Z. Despotovic, M. Hauswirth, M. Punceva, and R. Schmidt. "P-Grid: A Self-organizing Structured P2P System", voL 32, no. 3, ACM SIGMOD Record (Association For Computing Machinery Special Interest Group On Management Of Data), Sep 2003.
2. K. Aberer, P. Cudré-Mauroux, A. Datta, *Z.* Despotovic, M. Hauswirth, M. Punceva, R. Schmidt, and J. Wu, "Advanced Peer to peer Networking: The P-Grid System and its Applications", PIK Journal (Praxis der lnformationsverarbeitung and Kommunikation), vol. 26, no. 3, June 2003.
3. K. Aberer, A. Dana, and M. Hauswirth, 'Peer to peer Systems and Applications, Lecture Notes in Computer Science", Springer Verlag, 2005
4. M. Amoretti, F. Zanichelli, G. Conte, Whitepaper "Enabling peer to peer web service architectures with JXTA-SOAP Information Technology Department. University of Parma, 2007.
5. M. Arenas, V. Kantere, A. Kementsietsidis, L Kiringa, R. J. Miller, and J. Mylopoulos. The Hyperion Project: From Data Lntegration to Data Coordination", voL 32, no. 3, SIGMOD Record, March 2003.
6. R. Bernstein, "A survey of approaches to automatic schema matching", VLDB (Very Large Database) Journal. vol. 10, no. 4, pp 334-350, 2001.
7. Y. Beverly: G. M. Hector, "Designing a Super-peer Network", *IEEE International Conference on Data Engineering,* Proceedings. 19th Lnternational Conference, pp 49 -60, March 2003.
8. J. Broekstra, M. Ehrig, P. Haase, F. van Harmelen, A. Kampman, M. Sabou, R. Siebes, S. Staab, H. Stuckenschmidt and C. Tempich. "A Metadata Model for Semantics-Based Peer to peer Systems". Ln Proceedings of the WWW 2003 Workshop on Semantics in Peer to peer and Grid Computing, pp 22-25, 2003.
9. T. Burkard. Herodotus: "A peer to peer web archival system", Master's thesis, MLT. ²00².

10. M. Castro. P. Druschel, A. Ganesh, A. Rowstron and D.S. Wallach. "Secure routing for structured peer to peer overlay networks", ACM Special Lnterest Group on Operating Systems (SIGOPS), VoL 36, No. SI. (2002), pp. 299-314.

LL X. Chen, S. Ren, H. Wang, and X. Zhang, "SCOPE: scalable consistency maintenance in structured P2P systems", IEEE INFOCOM 2005, Miami, 24th Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 3, pp 1502-1513, March 2005.

12. F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi, and S. Weerawarana, "Unraveling the Web Services Web: An Introduction to SOAP, WSDL, and UDDI ", *Internet Computing, IEEE* In Lnternet Computing, LEEE, VoL 6, No. 2.. pp. 86-93. March-April 2002.

13. A. Doan, P. Domingos, and A. Halevy, "Reconciling Schemas of Disparate Data Sources: A Machine Learning Approach", Ln Proceedings of the ACM SIGMOD Conference, pp 509-520, 200L

14. B. Godfrey, K. Lakshminarayanan, S. Surana, R. Karp and I. Stoica, - Load Balancing in Dynamic Structured P2P Systems", LNFOCOM 2004. Twenty-third Annual/ Joint Conference of the IEEE Computer and Communications Societies, VoL 4 (2004), pp. 2253-2262.

15. L. Gong, "Project JXTA: A Technology Overview, Sun Microsystems, Inc.",www.jxta.org

16. J. D. Gradecki, *"Mastering JXIA, Building JXTA Peer to Peer Applications* - , John Wiley & Sons, 2002 (TSBN: 0471250848).

17. S. Gribble, A. Halevy, Z. Ives. M. Rodrig, and D. Suciu, 'What can databases do for peer to peer?" Proceedings of the Fourth Lnternational Workshop on the Web and Databases, WebDB 2001, Santa Barbara. California, USA. May 24-25, 2001, in conjunction with ACM SIGMOD 200L

18. S. Groppe, and S. Böttcher, "XPath query transformation based on XSLT stylesheets", Fifth ACM CIKM (Conference on Lnformation and Knowledge Management) International Workshop on Web Information and Data Management (WLDM 2003), New Orleans, Louisiana, USA, November 7-8, 2003.

19. A. Y. Halevy, Z. G. Ives, P. Mork, and L Tatarinov. 'Piazza: Data management infrastructure for semantic web applications". Proceedings of the 12th international conference on World Wide Web (WWW), pp. 556-567, 2003.

20. D. Hughes, 0. Coulson, and I. Warren, "A Framework for Developing Reflective and Dynamic P2P Networks (RaDP2P)", 4th International Conference on Peer-to-Peer Computing (P2P 2004), Zurich, Switzerland, 15-17 August 2004.

21. C. M. Jones, M.B. Kermarrec, A.M Rowstron, A. Theimer, M. Wang, and H. VVolman, "An evaluation of scalable application-level multicast built using peer to peer overlays", *INFOCOM 2003. twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE,* Vol. 2, pp. 1510-1520, 2003.

22. G. Koloniari, Y. Petrakis, and E. Pitoura, Content-based overlay networks for xml peers based on multi-level bloom filters," in Proceedings of the Tnternational VLDB Workshop on Databases, Information Systems and Peer to Peer Computing (DBISP2P), 2003.

23. S.S. Lam and H. Liu, Failure Recovery for Structured P2P Networks: Protocol Design and Performance Evaluation. Proceedings of the International Conference on Measurements and Modeling of Computer Systems. SIGMETRICS 2004. June 10-14, New York, NY, USA 2004.

24. J. Madhavan and P. A. Bernstein and E. Rahn), Generic Schema Matching with Cupid", In the VLDB Journal, pp. 49-58, 200L

25. J. Madhavan, P. Bernstein, K. Chen, A. Halevy, and P. Shenoy. "Corpus-based Schema Matching". Proceedings of the 21st International Conference on Data Engineering, ICDE 2005. 5-8 April 2005, Tokyo, Japan 2005.

26. J. Madhavan, P. Bernstein, K. Chen, A. Halevy, and P. Shenoy. Matching schemas by learning from others". In working notes of the IJC Al-03 workshop on Data Integration on the Web. 2003.

27. C. S. G. Miller, T. W. I long, 0. Sandberg, and B. Wiley. "Protecting free expression online with Freenet". *IEEE Internet Computing,* VoL 6, No. L (2002), pp. 40-49.

28. W. Nejdl, M. Wolpers, W. Siberski, C. Schmitz, M. Schlosser, L Brunkhorst, and A. Loser, "Super-Peer-Based Routing Strategies for RDF-Based Peer to peer Networks", *Web Semantics: Science. Services and Agents on the World Wide Web,* In 2003 World Wide Web Conference, Vol. I, No. *2.* (February 2004), pp. 177-186.

29. W. Nejdl, B. Wolf, C. Qu S. Decker, M. Sintek, A. Naeve, M. Nilsson, M. Palmer, and T. Risch, "EDUTELLA: a P2P networking infrastructure based on RDF", Proceedings of the 11th international conference on World Wide Web, 2002.

30. R.A. Pottinger and P.A. Bernstein, "Creating a Mediated Schema Based on Initial Correspondences," *IEEE Data Engineering Bulletin,* Volume 25, No. I (March. 2002).

31. S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, A Scalable Content-Addressable Network," In Proceedings of ACM SIGCOMM, pp. 161-172, 200L

32. E. Rescorla, "Introduction to Distributed I lash Tables", Internet Engineering Task Force (IETF) Journal, Vol. 2, No. I, September 2006.

33. P. Rodriguez-Gianolli, M. Garzetti, L. Jiang, A. Kementsietsidis, L Kiringa, M. Masud, R. Miller, and J. Mylopoulos. In Proceedings of the International Conference on Very Large Databases (VLDB) 2005, p. 1291-1294.

34. C Rouse and S Berman (2007) Mapster A Peer-to-Peer Data Sharing Environment" South African Computer Journal, vol. 39, pp. 35— 46, December 2007.

35. M. Schlosser, M. Sintek, S. Decker, and W. Nejdl, "HyperCuP Hypercubes, Ontologies and Efficient Search on P2P Networks," Agents and Peer-to-Peer Computing, First International Workshop, AP2PC 2002. Bologna, Italy, July 2002.

36. L Stoica and R. Morris and D. Karger and F. Kaashoek and H. Balakrishnan, Chord: A scalable peer-to-peer lookup service for internet applications". In *Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications* (San Diego. California. United States). SIGCOMM '0L ACM, New York, 200L

37. B. Tang Z. Z. Kashyap, A. T. Chiuch, An integrated approach for P2P file sharing on multi-hop wireless networks", *IEEE International Conference,* Wireless and Mobile Computing. Networking and Communications, 2005. (WiMob 2005).

38. L Tatarinov, Z. Ives, J. Madhavan and A. Halevy, D. Suciu, N. Dalvi, X. Dong, Y. Kadiyaska, G. Miklau, and P. Mork. The Piazza Peer Data Management Project". ACM SIGMOD Journal, Vol. 32, No. 3, 2003.

39. B. Traversat, M..Abdelaziz, M. Duigou J. C. Hugly, E. Pouyoul, B. Yeager, A. Arora, C. Haywood, Project JXTA 2.0 Super-Peer Virtual Network. Sun Microsystems. Inc., www.jxta .org

40. B. Traversal, The white paper "Project JXTA 2.0 Super-Peer Virtual Network," (Project JXTA, May 2003), describes the inner workings of the rendezvous super-peer network in intricate details, accessed June 2008.

41. D. D. Vecchio and S. H. Son, Flexible Update Management in Peer to peer Database Systems", Ninth International Database Engineering and Applications Symposium (IDEAS 2005), 25-27 July 2005, Montreal, Canada 2005.

42. J. Walkerdine, L Melville and I. Sommerville. "Designing for Presence within P2P Systems", Technical Report COMP-003-2004, Computing Department, Lancaster University, 2004.

43. L. Xu, D. Embley, ' Combining the Best of Global-as-View and Local-as-View for Data Tntegration , Information Systems Technology and its Applications, 3rd International Conference ISTA2004, June 15-17. 2004, Salt Lake City, Utah, USA, Proceedings 2004.

44. R. Yerneni and C. Li and H. Garcia-Molina and J. D. Ullman, Computing Capabilities of Mediators", SIGMOD 1999, Proceedings ACM SIGMOD International Conference on Management of Data. June 1-3, 1999.

45. N. Zhang, Peer to peer Distributed Database System, Master of Engineering", Taiyuan University of Technology, 1995.

46. Y. Zhu and IL Wang and Y. Hu, "A Super-Peer Based Lookup in Structured Peer to peer Systems", Proceedings of the ISCA 16th International Conference on Parallel and Distributed Computing Systems. August 13-15. 2003, Atlantis Hotel. Reno, Nevada, USA 2003.

47. Project JXTA: Getting started, www.jxta.org, accessed June 2008.

48. Project JXTA: An Open, Peer-to-Peer Collaboration Platform using Java and XML, http://www.webreference.com/xml/column32 , accessed June 2008.

49. Project JXTA v 2.0: JavaTM Programmer's Guide, Sun Microsystems, May 2003, accessed June 2008.

50. L Zaihrayeu and F. Giunchiglia, :•Coordinating mobile databases", Proceedings of the 1st International Workshop on Peer- to-Peer Knowledge Management (P2PKM'04), 2004