Extending JXTA for P2P File Sharing Systems

Bhanu Krushna Rout(10506012)

Smrutiranjan Sahu(10506034)

Department of Computer Science and Engineering

National Institute of Technology Rourkela

Rourkela-769 008, Orissa, India

Extending JXTA for P2P File Sharing Systems

Thesis submitted in partial fulfillment
of the requirements for the degree of

Bachelor of Technology

in

Computer Science and Engineering

by

Bhanu Krushna Rout(10506012)
Smrutiranjan Sahu(10506034)

under the guidance of

Prof. Sujata Mohanty

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, Orissa, India May 2009

Department of Computer Science and Engineering National Institute of Technology Rourkela Rourkela-769 008, Orissa, India.

Certificate

This is to certify that the work in the thesis entitled E*xtending JXTA for P2P File Sharing Systems,* submitted by Smrutiranjan Sahu and Bhanukrushna Rout is a record of an original research work carried out by them under our supervision and guidance in partial fulfillment of the requirements for the award of the degree of *Bachelor of Technology* in Computer Science and Engineering during the session 2008–2009 in the department of Computer Science and Engineering, National Institute of Technology Rourkela(Deemed University). Neither this thesis nor any part of it has been submitted for any degree or academic award elsewhere.

HOD	Prof Sujata Mohanty
Department of CSE	Department of CSE
NIT Rourkela	NIT Rourkela
Place: NIT Rourkela	Place: NIT Rourkela
Date:	Date:

Signature of External

Acknowledgement

We express our sincere gratitude to Sujata Mohanty Professor Department of Computer science and Engineering, National Institute of Technology, Rourkela, for her valuable guidance and timely suggestions during the entire duration of our project work, without which this work would not have been possible.

We would also like to convey our deep regards to all other faculty members, staff and students of Department of Computer Science and Engineering, NIT Rourkela, who have bestowed their great effort and guidance at appropriate times without which it would have been very difficult on our part to finish this project work.

BhanuKrushna Rout
Roll No. 10506012

Smrutiranjan Sahu
Roll No. 10506034

CONTENTS

List of Acronyms

Acronym	Description
DHT	Distributed Hash Table
SRDI	Shared Resource Distributed Index
RPV	Rendezvous peerview
RDV	Rendezvous Peer
ADV	Advertisement
XML	Extensible Markup Language

List of Figures

ABSTRACT

File sharing is among the most important features of the today's Internet-based applications. Most of such applications are server-based approaches inheriting thus the disadvantages of centralized systems. Advances in P2P systems are allowing to share huge quantities of data and files in a distributed way. In this paper, we present extensions of JXTA protocols to support file sharing in P2P systems with the aim to overcome limitations of server-mediated approaches. Our proposal is validated in practice by deploying a P2P file sharing system in a real P2P network. The empirical study revealed the benefits and drawbacks of using JXTA protocol for P2P file sharing systems.one of the most important concern.

CHAPTER 1

INTRODUCTION

1.1 Introduction

Peer-to-Peer (P2P) systems have become popular due to file sharing among millions of user world wide. Since the appearance of Napster, Freenet and Gnutella, file sharing software has been a hot research topic for industrial and academic purposes. A as a matter of fact, there are still many of issues to understand and better address in P2P file sharing domain. Thus, deciding which protocols to use (Freenet, Gnutella, and Napster use different protocols), how to reduce the traffic generated due to file sharing[9], what schemes to use for efficient indexing/searching files within a P2P file systems are such research questions, to name a few. On the other hand, there are issues related to the design of P2P systems for file sharing that benefit from the decentralized nature of P2P systems yet facilitate efficient file sharing among
peers and P2P file sharing applications. Indeed, there is a whole range from server-mediated architecture P2P systems to fully decentralized systems and appropriate architectures that match different needs of P2P file sharing applications are to be investigated. Actually the most popular alternatives for file sharing are Kazza, Overnet and Bittorrent, each one with its own net and form of searching and indexing files. Kazza uses a big server to provide searching and indexing of the files. Overnet indexes the files in a list of different servers; the user is connected to one of them, through which are done all the searches. In the Bittorrent net, the user has a little file with the specification of each download. There is a central node that organizes all the clients. Unfortunately, all this alternatives are for a general sharing system and a user can’t share a file or a folder with the grant that only a particular group of people can have it.

One could use as an alternative sharing files with people that a user chooses through IM programs (like MSN Messenger), but in this case file sharing is automatic and doesn’t permit that the destination user chooses what files wants from other users (download under demand). Thus, such file sharing systems are not appropriate for academic online campuses, where students working in small groups would like to share notes, do homework in group in real time or help each other. Moreover, these file sharing programs use a centralized architecture, needing either a server or a very special node to control all the net.

Peer-to-Peer (P2P) systems are decentralized, self- organizing distributed systems showing each time more remarkable improvements in scalability, robustness and distributed storage. P2P networks are the alternative to the traditional client-server applications by replacing client-server communication with peer interactions; peers can serve as client, servers, edge peers providing thus much more flexibility; a peer can request files from others and share files with other peers. In the domain of P2P protocols, JXTA technology [4], [6] is an interesting alternative in the file sharing field given that its protocols allow to develop P2P platforms, either pure or mixed. This property is certainly important since pure P2P need not the presence of a server for managing the net. Unfortunately, JXTA protocols do not support file sharing, however it offers a set of basic protocols, such as publication of advertisements, that can be further developed and used for file sharing purposes.

1.2 Motivation

This work is also motivated by the investigation of a new P2P architecture, namely a broker-based P2P system and its benefits for P2P file sharing systems. The idea of P2P architectures using broker peers and client peers has already been exploited for task execution in P2P distributed systems [1], [2]. Our proposal is validated in practice by deploying a P2P file sharing system in a real P2P network. The empirical study revealed the benefits and drawbacks of using JXTA protocol.

1.3 Problem Statement and Objectives

The objective of this work is to extend and improve the JXTA protocols to support the development and deployment of P2P file sharing systems. Moreover, we would like such protocols to be practically useful also for academic context giving thus support to online learning teams. There is a whole range of architectures from server-mediated to pure P2P networks. In the former architecture the central server is responsible for maintaining/indexing shared files and respond to requests for files while peer nodes are responsible to host the files and make available the information on shared files to the server. In the later architecture, there is no central server but peers that can play both client and server roles for sharing and downloading files.

CHAPTER 2

LITERATURE STUDY

P2P Concepts
P2P Routing Algorithms

2.1 What is P2P ?

It is a technology which “enables any communication node (peer) to provide services to another peer”. A peer in P2P network acts as both a client and a server in traditional client/server architecture. Instead of Internet information being held in a few central locations, Peer-to-Peer computing makes it theoretically possible to access the files and data residing on every personal computer connected to the Internet. P2P networks are typically used for connecting nodes via largely adhoc connections. Such networks are useful for many purposes. Sharing content files containing audio, video, data or anything in digital format is very common, and real time data, such as telephony traffic, is also passed using P2P technology.

Classifications of P2P networks

- Centralized P2P network such as Napster.
- Decentralized P2P network such as KaZaA.
- Structured P2P network such as DHT(Chord).
- Unstructured P2P network such as Gnutella.
- Hybrid P2P network (Centralized and Decentralized) such as JXTA (an open source P2P protocol specification).

Advantage of P2P network

- The system is based on the direct communication between peers.
- There is zero reliance on centralized serviced or resources for operations.
- The system can survive extreme changes in network composition.

- They thrive in a network with heterogeneous environment.
- This model is highly scalable.

2.2 P2P Concepts:

Peers

Peer is any entity capable of performing some useful work and communicating the results of that work to another entity over a network, either directly or indirectly.

Useful *work* depends on the type of peer. Three possible types of peers exist in any P2P network:

- Simple peers
- Rendezvous peers
- Router peers

Each peer on the network can act as one or more types of peer, with each type defining a different set of responsibilities for the peer to the P2P network as a whole.

Simple Peers:

- A simple peer is designed to serve a single end user, allowing that user to provide services from his device and consuming services provided by other peers on the network.
- In all likelihood, a simple peer on a network will be located behind a firewall, separated from the network at large; peers outside the firewall will probably not be capable of directly communicating with the simple peer located inside the firewall.

- Because of their limited network accessibility, simple peers have the least amount of responsibility in any P2P network.
- Unlike other peer types, they are not responsible for handling communication on behalf of other peers or serving third-party information for consumption by other peers.

Rendezvous Peers:

- Taken literally, a rendezvous [15] is a gathering or meeting place;
- In P2P, a rendezvous peer provides peers with a network location to use to discover other peers and peer resources.
- Peers issue discovery queries to a rendezvous peer, and the rendezvous provides information on the peers it is aware of on the network.
- A rendezvous peer can augment its capabilities by caching information on peers for future use or by forwarding discovery requests to other rendezvous peers.
- These schemes have the potential to improve responsiveness, reduce network traffic, and provide better service to simple peers.
- A rendezvous peer will usually exist outside a private internal network's firewall. A rendezvous could exist behind the firewall, but it would need to be capable of traversing the firewall using either a protocol authorized by the firewall or a router peer outside the firewall.

Router (Relay) Peers:

- A router peer provides a mechanism for peers to communicate with other peers separated from the network by firewall or Network Address Translation (NAT) equipment.
- A router peer provides a go-between that peers outside the firewall can use to communicate with a peer behind the firewall, and vice versa.
- To send a message to a peer via a router, the peer sending the message must first

determine which router peer to use to communicate with the destination peer.

Services:

- *Services* provide functionality that peers can engage to perform "useful work" on a remote peer. This work might include
  - transferring a file,
  - providing status information,
  - performing a calculation,
  - or basically doing anything that you might want a peer in a P2P network to be capable of doing.
- Services are the motivation for gathering devices into a P2P network; without services, you don't a have a P2P network. Services can be divided into two categories:
  - **Peer services**—Functionality offered by a particular peer on the network to other peers. The capabilities of this service will be unique to the peer and will be available only when the peer is connected to the network. When the peer disconnects from the network, the service is no longer available.
  - **Peer group services—**Functionality offered by a peer group to members of the peer group. This functionality could be provided by several members of the peer group, thereby providing redundant access to the service. As long as one member of the peer group is connected to the network and is providing the service, the service is available to the peer group.

Advertisements:

- Until now, P2P applications have used an informal form of advertisements.

- In Gnutella, the results returned by a search query could be considered An advertisement that specifies the location of a specific song file on the Gnutella network. These primitive advertisements are extremely limited in their purpose and application.
- An *advertisement* [8], [10] is defined as follows:

A structured representation of an entity, service, or resource made available by a peer or peer group as a part of a P2P network. Such as:

- peers,
- peer groups,
- pipes,
- endpoints,
- services,
- content.

Peer Advertisement (XML):

```
JXTA>whoami

XML local peer information

<Peer>MyPeer</Peer>

<Keywords>NetPeerGroup by default</Keywords>

<PeerId>urn:jxta:uuid-
59616261646162614A787461503250333855A703D4E614DB7B54A9BE583FFCD4C03
</PeerId>

<TransportAddress>tcp://asterix:9701/</TransportAddress>
```

<TransportAddress>http://JxtaHttpClientuuid-
59616261646162614A78746150325033855A703D4E614DB7B54A9BE583FFCD4C03
/</TransportAddress>

Entity Naming:

- Most items on a P2P network need some piece of information that uniquely identifies them on the network:
  - **Peers**—A peer needs an identifier that other peers can use to locate or specify it on the network. Identifying a particular peer could be necessary to allow a message to be routed through a third party to the correct peer.
  - **Peer groups**—A peer needs some way to identify which peer group it would like to use to perform some action. Actions could include joining, querying, or leaving a peer group.
  - **Pipes**—To permit communication, a peer needs some way of identifying a pipe that connects endpoints on the network.
  - **Contents**—A piece of content needs to be uniquely identifiable to enable peers to mirror content across the network, thereby providing redundant access. Peers can then use this unique identifier to find the content on any peer.
- In traditional P2P networks, some of these identifiers might have used network transport-specific details; for example, a peer could be identified by its IP address.
- However, using system-dependent representations is inflexible and can't provide a system of identification that is independent of the operating system or network transport. In the ideal P2P network, any device should be capable of participating, regardless of its operating system or network transport.

Finding Advertisements:

- Peers, peer groups, services, pipes, and endpoints are represented as an advertisement.
- That simplifies the problem of finding peers, peer groups, services, pipes, and endpoints. Instead of worrying about the specific case, such as finding a peer, you need to consider only the general problem of finding advertisements on the network.
- A peer can discover an advertisement in three ways:
  - No discovery

involves no network connectivity and can be considered a passive discovery technique

  - Direct discovery
  - Indirect discovery

The last two techniques involve connecting to the network to perform discovery and are considered active discovery techniques.

No Discovery:

The easiest way for a peer to discover advertisements is to eliminate the process of discovery entirely. Instead of actively searching for advertisements on the network, a peer can rely on a cache of previously discovered advertisements to provide information on peer resources.

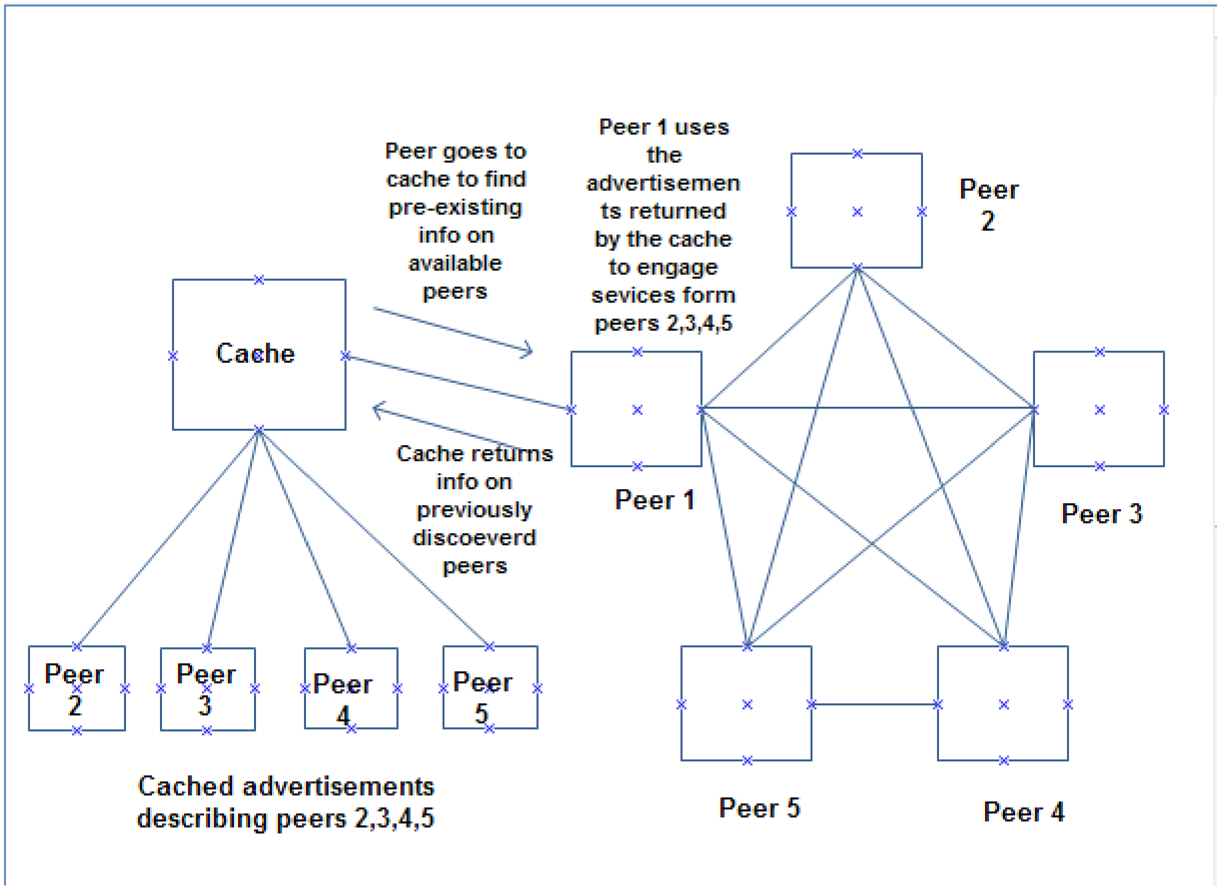

Peer goes to cache to find pre-existing info on available peers
Peer 1 uses the advertisements returned by the cache to engage sevices form peers 2,3,4,5
Peer 2
Cache
Peer 1
Peer 3
Cache returns info on previously discoeverd peers
Peer 2
Peer 3
Peer 4
Peer 5
Cached advertisements describing peers 2,3,4,5
Peer 5
Peer 4

Fig:2.1 No Discovery of Advertisements

- **Advantage**
- reduce the amount of network traffic generated by the peer and allow a peer to obtain nearly instantaneous results, unlike active discovery methods.
- The local cache consist of lists previously discovered rendezvous peers, thereby providing a starting point for active peer discovery.
- At the other extreme, a cache might be as comprehensive as a database of every advertisement discovered by the peer in the past.
- **Disadvantage**
- The potential for advertisements in the cache to grow *stale* and describe resources that are no longer available on the network.

- In this case, stale advertisements in the cache increase network traffic. When a peer attempts to engage a resource over the network and discovers that the resource is no longer available, the peer will probably have to resort to an active discovery method.
- To reduce the possibility that a given advertisement is stale, a cache can expire advertisements

Direct Discovery:

- Peers that exist on the same LAN might be capable of discovering each other directly without relying on an intermediate rendezvous peer to aid the discovery process.
- When other peers have been discovered using this mechanism, the peer can discover other advertisements by communicating directly with the peers
- Unfortunately, this discovery technique is limited to peers located on the same local LAN segment and usually can't be used to discover peers outside the local network.
- Discovering peers and advertisements outside the private network requires indirect discovery conducted via a rendezvous peer.

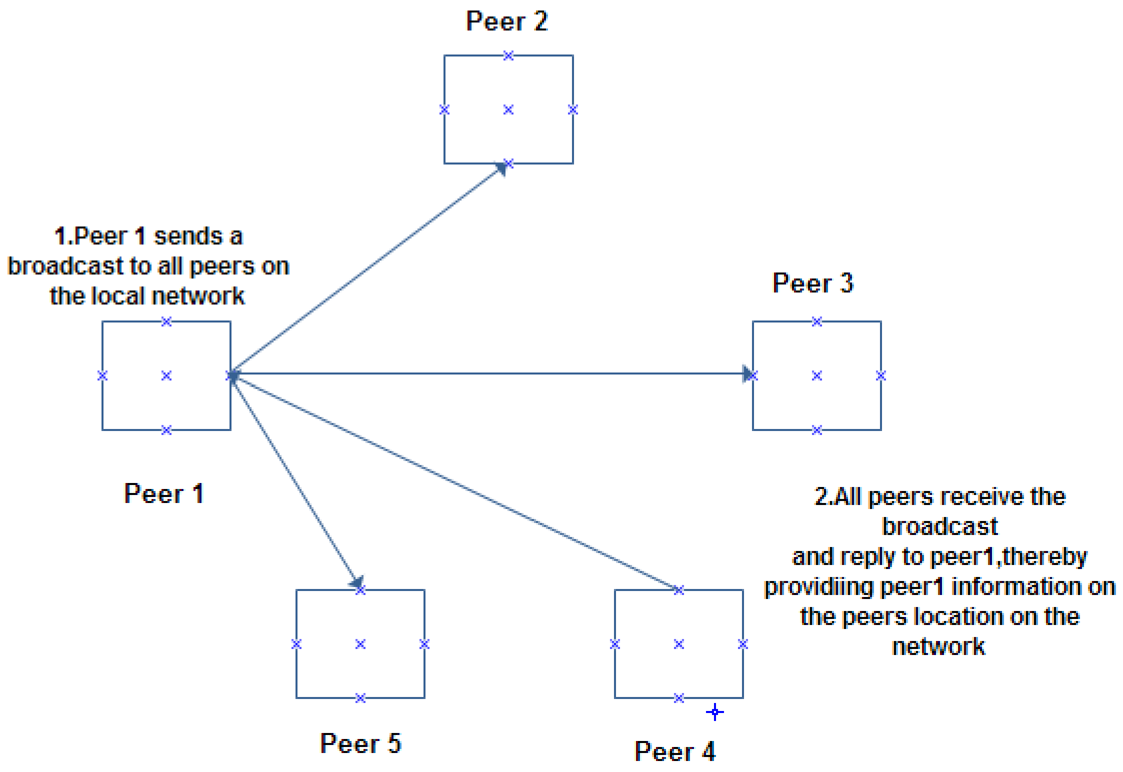

Peer 2
1.Peer 1 sends a broadcast to all peers on the local network
Peer 3
Peer 1
2.All peers receive the broadcast and reply to peer1,thereby providiing peer1 information on the peers location on the network
Peer 5
Peer 4

Fig: 2.2 Direct Discovery of Advertisements

Indirect Discovery:

Indirect discovery requires using a rendezvous peer to act as a source of known peers and advertisements, and to perform discovery on a peer's behalf.

This technique can be used by peers on a local LAN to find other peers without using broadcast or multicast capabilities, or by peers in a private internal network to find peers outside the internal network.

Rendezvous peers provide peers with two possible ways of locating peers and other advertisements:

- **Propagation—**A rendezvous peer passes the discovery request to other peers on the network that it knows about, including other rendezvous peers that also propagate the request to other peers.**Cached advertisements**—In the same manner that simple peers can use cached advertisements to reduce network traffic, a rendezvous can use cached advertisements to respond to a peer's discovery queries.

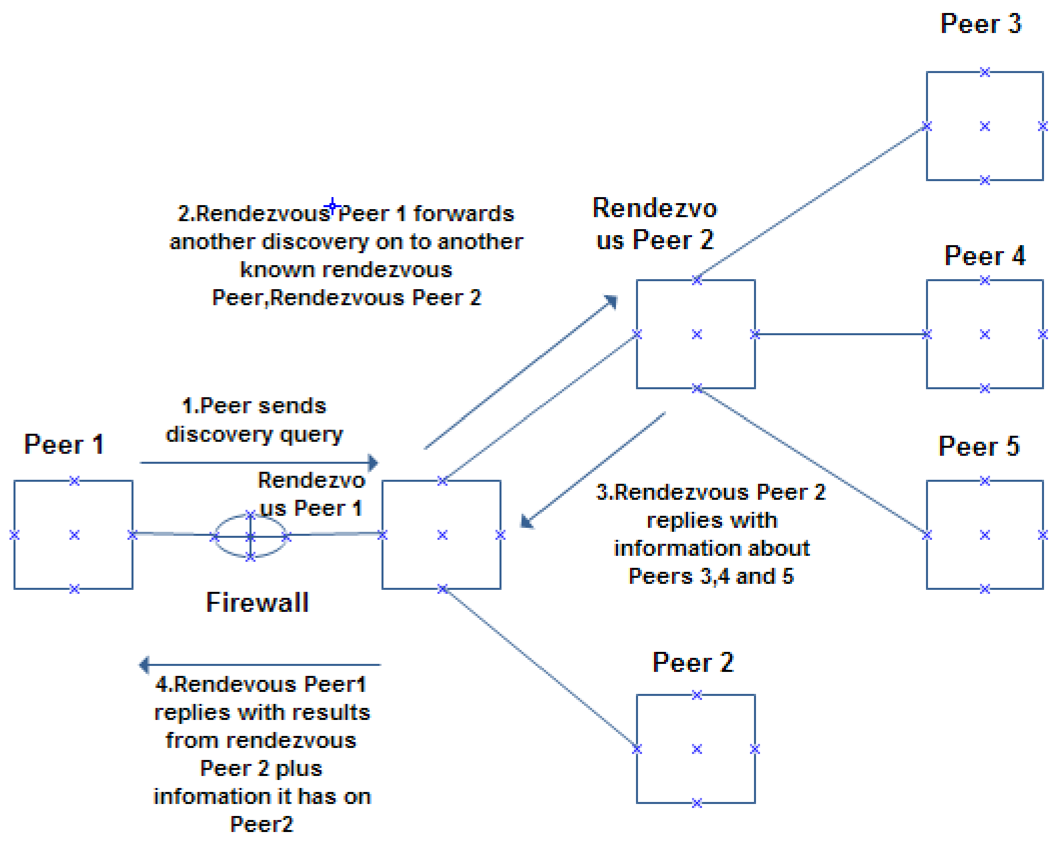

Peer 3
2.Rendezvous Peer 1 forwards another discovery on to another known rendezvous Peer,Rendezvous Peer 2
Rendezvous Peer 2
Peer 4
1.Peer sends discovery query
Peer 1
Rendezvous Peer 1
Peer 5
3.Rendezvous Peer 2 replies with information about Peers 3,4 and 5
Firewall
Peer 2
4.Rendevous Peer1 replies with results from rendezvous Peer 2 plus infomation it has on Peer2

Fig: 2.3 Indirect Discovery of Advertisements

Indirect Discovery propagation chaos:

Propagation of discovery queries without restriction can cause network congestion because of one in/many out problems.

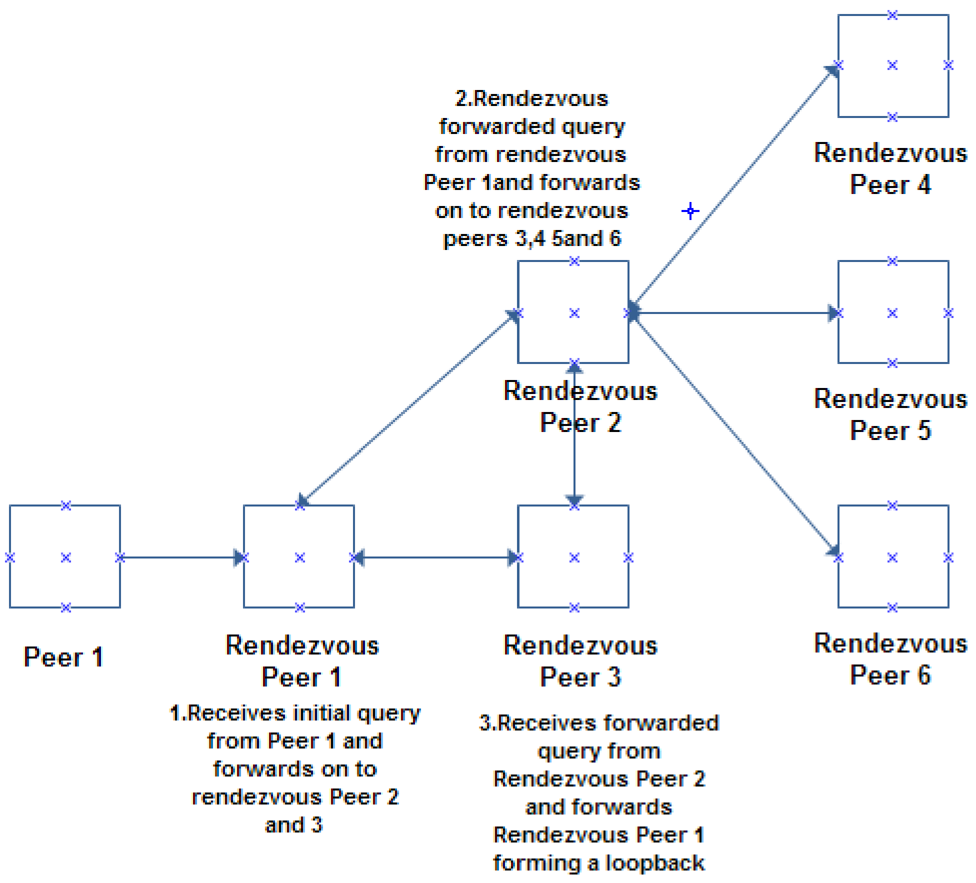

2.Rendezvous forwarded query from rendezvous Peer 1and forwards on to rendezvous peers 3,4 5and 6
Rendezvous Peer 4
Rendezvous Peer 2
Rendezvous Peer 5
Peer 1
Rendezvous Peer 1
Rendezvous Peer 3
Rendezvous Peer 6
1.Receives initial query from Peer 1 and forwards on to rendezvous Peer 2 and 3
3.Receives forwarded query from Rendezvous Peer 2 and forwards Rendezvous Peer 1 forming a loopback

Fig: 2.4 Indirect Discovery propagation chaos

Indirect Discovery time to live (TTL):

TTL is expressed as the maximum number of times a query should be propagated between peers in a network uses a decrement system on each message. Consequently, each message has a maximum radius on a network (assuming each peer does the decrement).

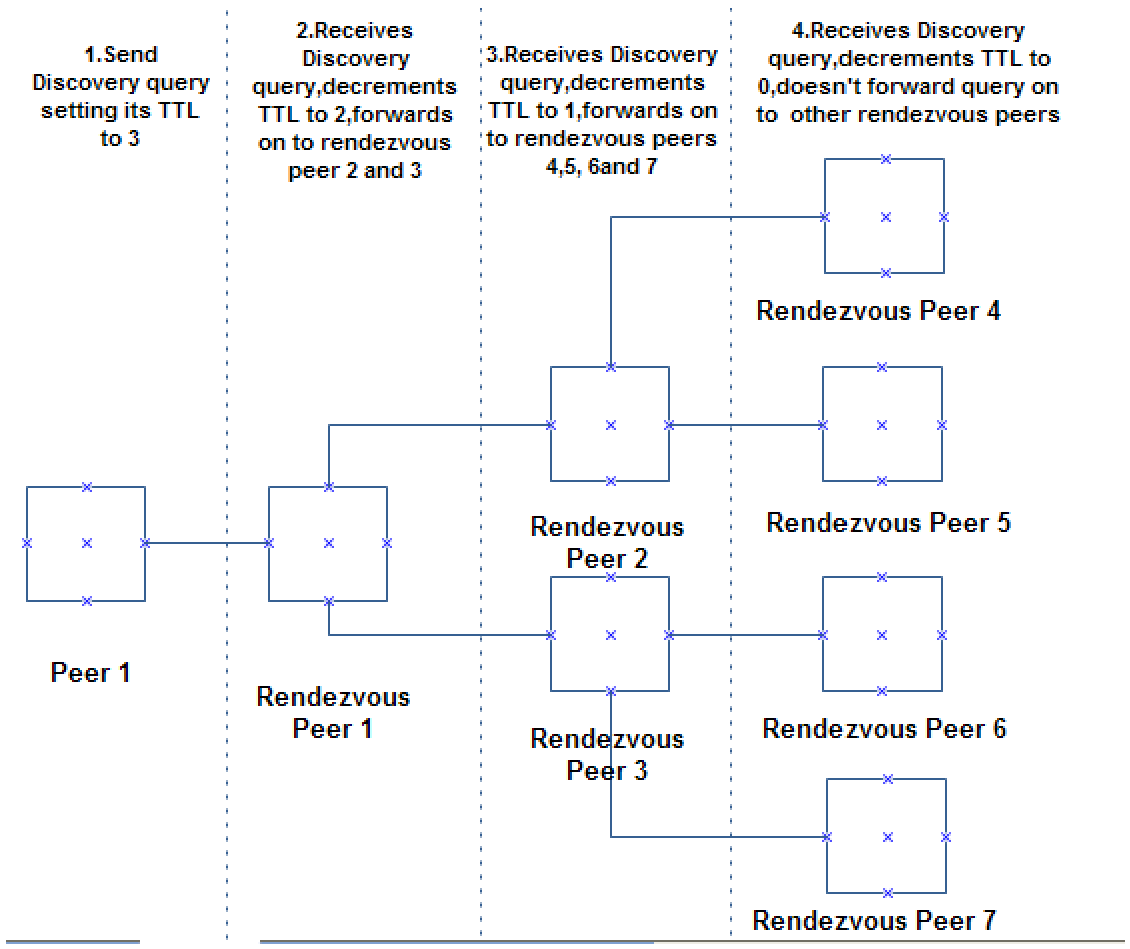

1.Send Discovery query setting its TTL to 3
2.Receives Discovery query,decrements TTL to 2,forwards on to rendezvous peer 2 and 3
3.Receives Discovery query,decrements TTL to 1,forwards on to rendezvous peers 4,5, 6and 7
4.Receives Discovery query,decrements TTL to 0,doesn't forward query on to other rendezvous peers
Rendezvous Peer 4
Rendezvous Peer 2
Rendezvous Peer 5
Peer 1
Rendezvous Peer 1
Rendezvous Peer 3
Rendezvous Peer 6
Rendezvous Peer 7

Fig: 2.5 Indirect Discovery time to live

2.3 Peer to Peer Routing:

- Routing [12], [14] on these networks is either centralised or statically configured and is therefore unproblematic.
- Another class of P2P networks is the overlay network. Overlay networks build a virtual topology on top of the physical links of the network. Nodes leave and join this network dynamically and the average uptime of individual nodes is relatively low. The topology of an overlay network may change all the time. Once a route is established, there is no guarantee of the length of time that it will

be valid.

Algorithm Designing Issues:

- Scalability : a measure of how a system performs when the number of nodes and/or number of messages on the network grows.
- Complexity : the order of steps required for a packet to travel from one host to another in a worst case scenario.
- Anonymity : entails hiding the source of a File/Data.

4 types of algorithm:

- **Gnutella Routing/Flooding**
- **Distributed Hash Tables** (DHT)
- **Semantic Routing**
- **Freenet**

Here we have explained DHT only because JXTA is based on DHT (a Loosely Consistent DHT).

Distributed Hash Tables:

A hash function takes a variable length string of bytes and returns a number that it generates from this. DHT algorithms [13],[14] work by hashing all file/data identifiers and storing their locations in a giant hash table, which is distributed across the participating nodes. All n nodes also use this function to hash their IP address, and conceptually, the nodes will form a ring in ascending order of their hashed IP. successor(x) is the next actual node in the logical ring after x.

Share File: When a node wants to share a file or some data, it hashes the identifier to generate a key, and sends its IP and the file identifier to successor(key). These are then stored at this node. In this way, all resources are indexed in a large distributed hash table across all participating nodes. If there are two or more nodes that hold a given file or resource, the keys will be stored at the same node in the DHT, giving the requesting node a choice.

Request File: So when a node wants something, it will hash its identifier and send a request to successor(key), which will reply with the IP of the node that holds the actual data. When it doesn't know its IP, but only the key ,it maintains a finger table. This contains a list of keys and their successor IP's, and is organized such that each node holds the IP of a exponential sequence of nodes that follow it. Entry i of node k's finger table holds the IP of node :   k + 2^i.

Search File: Searching for a node is a log(n) procedure. Each node knows the IP of the next real node in the ring. If the key lies between k and the next real node, then successor(key) is the next node. Otherwise the finger table is searched for the closest predecessor of the key. The request is forwarded to this node and it repeats the same procedure until the node is found. The request contains the IP of the requesting node so when the search terminates at the key node, it can reply instantly, with no back propagation required.

Join and Leave: On joining, a node hashes its IP to form a number and sends a request to any node on the network to find successor(number). It then takes a certain amount of its successors hast table and a message is sent to the predecessor to inform it of its new successor. When a node leaves the network, it sends its table to its successor and also tells its predecessor of its departure. On both these events, finger tables will end up with wrong values, and so each node runs a periodic process, which sends messages around the ring to find new nodes.

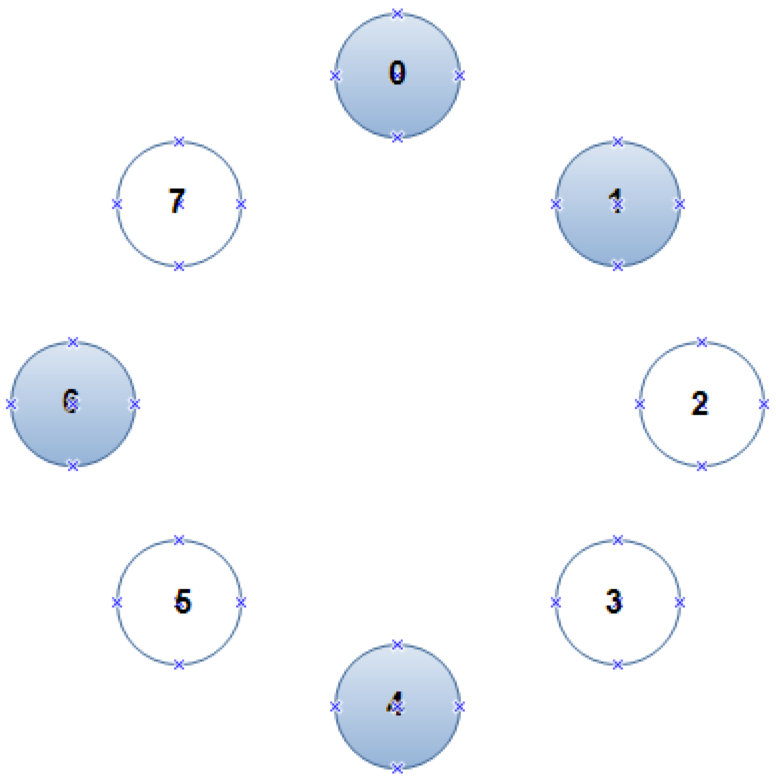


Fig: 2.6 Working of DHT

What is JXTA™ Technology?

JXTA is an open network computing platform designed for peer-to-peer (P2P) computing by way of providing the basic building blocks and services required to enable anything anywhere application connectivity. The name "JXTA" is not an acronym [8]. It is short hand for juxtapose, as in side by side. It is a recognition that P2P is juxtaposed to client-server or Web-based computing, which is today's traditional distributed computing model. JXTA provides a common set of open protocols backed with open source reference implementations for developing peer-to-peer applications.

JXTA 2 Architecture:

Shared Resource Distributed Index(SRDI):

In JXTA, resources are described by metadata in the form of advertisements (essentially XML documents). SRDI [8], [16] is used to index these advertisements network wide, through a set of specified attributes. The distributed index maintained resembles a hash table, with the indexed attributes as hash keys and the hash values mapping back to the source peer containing the actual advertisement. Queries can then be made anywhere in the rendezvous network based on these attributes. In this way, SRDI can answer advertisement queries in the network by locating the peer that has the required advertisement.

Under SRDI, it is no longer necessary to remotely publish any advertisement. Only the index of the advertisements stored on a peer is published. This index information is "pushed out" to the DHT (rendezvous super-peer network) through a single connected rendezvous.

Loosely Consistent DHT:

JXTA2 network act as distributed data structure: a virtual hash table containing the index of all the published advertisements in the entire JXTA group. An edge peer can query the hash table at any time by supplying a set of attributes -- the hash keys in the table. The query is resolved by the network (actually the rendezvous super-peer network) by hashing the key to the required value .

To create this DHT, each peer caches advertisements locally, and all locally stored advertisements are indexed. The index is pushed to the rendezvous node (JXTA 2 edge peer connects to only one rendezvous node at any time). The rendezvous super-peer network maintains the DHT containing the amalgamated indices. Queries are always sent to a rendezvous peer [16].

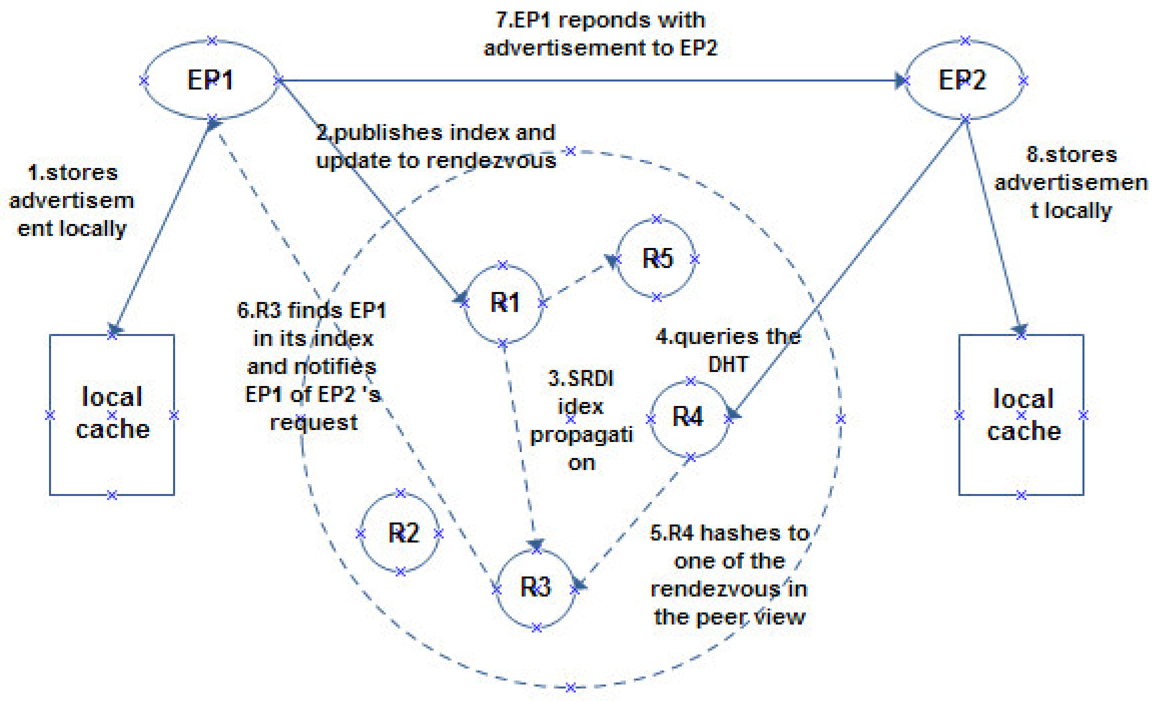

7.EP1 reponds with advertisement to EP2
EP1
EP2
1.stores advertisement locally
2.publishes index and update to rendezvous
8.stores advertisement locally
R5
R1
6.R3 finds EP1 in its index and notifies EP1 of EP2 's request
4.queries the DHT
local cache
3.SRDI idex propagation
R4
local cache
R2
R3
5.R4 hashes to one of the rendezvous in the peer view

Fig: 2.7 Loosely consistent DHT

When a rendezvous goes away, the chunk of index that it is maintaining becomes unavailable for a period of time (until the responsible peers publish it again). Based on an adaptive approach, JXTA 2 can maintain a loosely consistent DHT, one that can deal with the transient nature of peers in a P2P network.

Rendezvous peerview and RPV walker:

The RPV is the list of known rendezvous to the peer, ordered by each rendezvous' unique peer ID. The hash function used in the DHT algorithm is identical in every peer and is used to determine the rendezvous that a (locally irresolvable) query request should be forwarded to. Any rendezvous that becomes unreachable is removed from a peer's RPV. Each rendezvous in the super-peer network regularly sends a random subset of its known rendezvous to a random selection of rendezvous in its RPV. This is done to ensure eventual convergence of RPV network-wide and to adapt to any partitioning or merging occurring in the underlying physical network.

Maintaining the DHT, SRDI will store incoming index information to a selected rendezvous peer in the super-peer network based on a fixed hashing function. To cope with the loosely consistent nature, the index information is redundantly replicated to additional rendezvous peers adjacent on the RPV (recall that the RPV is an ordered list of known rendezvous by their universally unique peer ID). This will ensure that if the targeted rendezvous crashed, the probability of successfully hashing to that index information during a query is still quite high.

When resolving queries, the hashing function is performed against a rendezvous' own RPV. Because it is foreseeable that multiple existing rendezvous may have disconnected or multiple new rendezvous may have joined the super-peer network, if hashing does not immediately resolve the query, an RPV walker is introduced and forwards the query to a limited number of additional rendezvous. The algorithm used by this limited range walker is designed to be "pluggable," or customizable for more specific network scenarios.

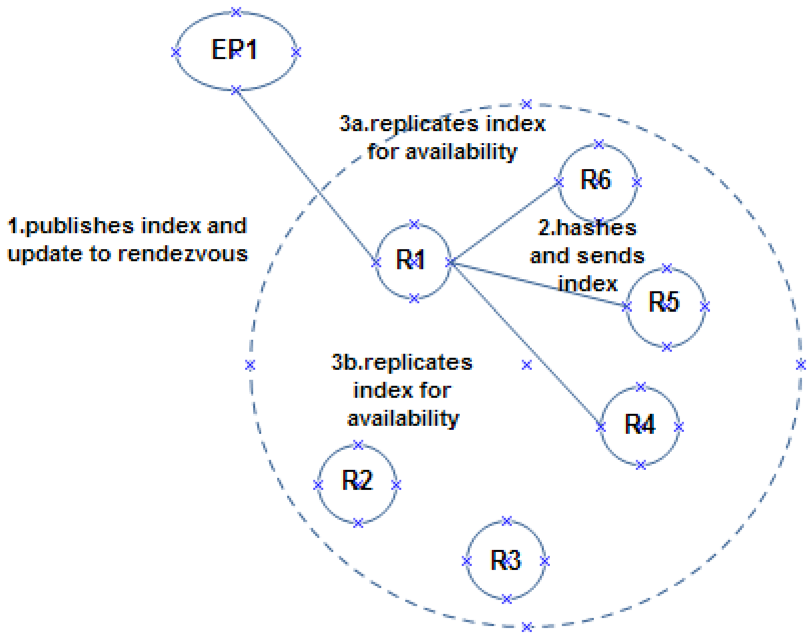

EP1
3a.replicates index for availability
R6
1.publishes index and update to rendezvous
R1
2.hashes and sends index
R5
3b.replicates index for availability
R4
R2
R3

Fig: 2.8 RPV Walker

JXTA Protocols:

For the communication between peers is used a group of asynchronous protocols based in the model request/reply. The different protocols [8] standardizes the way to find other peers, the communication between them, the formation of groups, the publication and the discovery of different advertisements. As part of these protocols, there are functionalities called services

(a) Rendezvous Service: used for publishing messages outside the peerGroup;

(b) Discovery Service:usedto discover Advertisements (peers, peerGroups, pipes and other services) of a peerGroup;

(c) Pipe Service: offers operations to send and receive data over the pipes;

(d) Endpoint Service: to transmit simple messages (used in the old version 1.0);

(e) Resolver Service: offers a generic mechanism to send requests and receive replies;

(f) Peer Info Service: allows to obtain information about nodes of the group;

(g) Membership Service: allows a peer to establish a unique identity in order
to guarantee the presence in a group.

The above-mentioned services are the basis for the development of P2P file sharing systems. Moreover, JXTA offers other protocols for peer discovery, peer communication, publishing of advertisements and more generally discovering the information of other peers.

The JXTA-Overlay

JXTA-Overlay project is an effort to use JXTA technology for building an overlay on top of JXTA offering a set of basic primitives (functionalities) that are most commonly needed in JXTA-based applications. The proposed overlay comprises primitives for:

• Peer discovery

• Peer’s resources discovery

• Resource allocation

• Task submission and execution

• File/data searching, discovery and transmission

• Monitoring of peers, groups, tasks etc.

This set of basic functionalities is intended to be as complete as possible to satisfy the needs of JXTA-based applications. The overlay is built on top of JXTA layer and provides

a set of primitives that can be used later by other applications, which on their hand, will be built on top of the overlay, with complete independence. The JXTA-Overlay project has been developed using the latest updated JXTA libraries. In fact, the project offers also several improvements of the original
JXTA protocols/services to increase the reliability of JXTA- based distributed applications.

CHAPTER 3

EXTENSION OF JXTA PROTOCOLS

In The JXTA-Overlay already offers several improvements of JXTA protocols, including:

Presence management: Periodically the PeerAdvertisement is published and the local cache is synchronized.

• Connection group: a peer is connected to the general groups through a broker. The different brokers available are known by a previously published list. Furthermore, the user is connected automatically to the different groups he belongs to.

• Communication mechanism: JXTA-Overlay provides an error control in message sending of and organizes the net in such a way that every peer can send a message in

a straightforward to any peer by only knowing the peer name.

• Incoming messages mechanism: the messages are queued in peer's incoming queue in order to ensure messages are received and correctly treated at the peer's side.

In spite of the improvements provided by JXTA-Overlay, there are still two limitations for using it as a basis for P2P file sharing systems. The first limitation has to do with the treatment of the advertisements.

3.1 The treatment of the advertisements: The original JXTA-Overlay processes all the advertisements received by a peer just at the time they are received. If we would use the advertisements for publishing information on files shared by a peer, this could be very problematic. Indeed, the number of files shared by a peer could be large and therefore the computational cost would be very high (prohibitive for practical purposes). Moreover, as the file sharing advertisement would be processed at the at the time they are received by a peer, this could cause peer collapse. The solution to this problem is to treat the advertisements only whenever their information is needed. When an advertisement is received, its information is saved in the original XML format if it is really needed, and then it is treated. This treatment is done only for the first time, because when it is treated, the information is saved as it is done in the original JXTA-Overlay. This idea is also applicable to the opposite direction: when an advertisement has to be published, if the XML is already built it is published; if not, the XML is built only the first time (see Fig. 2).

3.2 Publishing of the advertisements: In the JXTA- Overlay, the file advertisements are sent very frequently (within very short time intervals) and have a short lifetime. This is done so in order to ensure that only the online peers will have advertisements operative. However, if a peer shares a large quantity of files, the file advertisement will be large and computationally costly to publish and to treat; again, publish ing the file advertisement very frequently could certainly be problematic. Our solution is that the file advertisements should be periodically sent in rather long time intervals solving thus the large quantity of files to be processed. Yet, with this specification, advertisements must have a longer lifetime implying that if a peer disconnects, its advertisements continue in the net. In order to deal with this problem, we use broker peers: whenever a broker works with the file advertisements, it always checks that the user is connected, checking thus the existence of the information advertisement (and its main attributes) that every peer sends to the net to communicate that it is online.

3.3 Managing the connection to groups: Any peerGroup needs a rendezvous peer. The principal peerGroup is that of NetPeerGroup, the connection to which is done according to the peer's configuration: (a) connection to the default rendezvous of JXTA. This is certainly inefficient since we cannot control the rendezvous; (b) connection to rendezvous pre-specified by the concrete JXTA application; (c) connection to rendezvous specified in a rendezvous list. In this later case, the JXTA application just needs to know the address where to find the rendezvous list. Note that the rendezvous list can be changed independently of the application, which is desirable in P2P applications. Moreover, once a peer is connected to the P2P network through the principal peerGroup,it can join any peerGroup by knowing the GroupAdvertisement of such groups. However, for this to be done, the peerGroups must exist, there must be at least one rendezvous and the GroupAdvertisements must be propagated. In our P2P network of broker peers and client peers, the brokers are the governors of the network and the organization of the groups. Thus broker peers are in charge of creating the groups and propagate the GroupAdvertisement accordingly.

CHAPTER 4

IMPLEMENTATION DETAILS

4.1 Required Components:

www.jxta.org has Java and C implementations of the core protocols.

- The C version is based on the APR (apache portable runtime) and trails behind the JAVA version in terms of functionality and ease of use.
- For Java – requires the J2SE JDK and NetBeans IDE 6.5
  Download Latest JXTA library, documentation, source code and tutorials from http://download.java.net/jxta/

JXTA's core uses 13 other JAR files like Jetty portable Web/Servlet Server, Log4J apache's generic logging API.

Directory Structure[8] so far

- /InstantP2P -> A "full-fledged" instant P2P application
- /lib -> The JAR Files
  - /lib/jxta.jar -> Contains the JXTA Programming API
- /Shell -> Command-line Interface to JXTA
- /tutorials -> Tutorials that we can download individually

Running JXTA Applications

You need to include the classpath option specifying the location of the required jar files.

Eg C:> java -classpath .\lib\jxta.jar;.\lib\bcprov-jdk14.jar;. JXTAp2p

4.2 Implementation Steps:

Configure Peer:

JXSE includes a simple UI configurator. This default may be overridden by using a programmatic platform configurator such as the NetworkConfigurator class, or by using the NetworkManager, which abstracts configuration to one of the preset configurations: Ad-hoc,Edge, Rendezvous, Relay, Proxy and Super.

Here we have used the default JXTA configurator.

Creating various ID types:

Every ID indicates its format immediately after the urn:jxta: prefix.ID Formats may choose to ignore the constructor seed information entirely, use it as random number generator seed values.

Advertisements:

Advertisements are typically represented as a text document (XML). Advertisement instances are created via AdvertisementFactory rather constructors on the Advertisement classes. These private implementations are registered with the AdvertisementFactory. New advertisement types must be registered with the AdverisementFactory by augmenting META-INF/services/net.jxta.document.Advertisement.

Searching for Group:

We created a new net peer group and got instance of discovery service. Then we searched for a custom group locally then remotely for 5 times with delay of 5 seconds. If the peer finds the custom group then it joins otherwise it creates a new group and publish its advertisement in the network.

Creating a Group:

Group authentication credentials are defined then we published its advertisement locally and remotely. The new group owner acts as broker peer.

Joining a Group:

We have taken the instance of Membership Service. User authenticates using JXTA configurator putting its credentials. Peer group validates new member by checking its group credentials. If the user meets membership validation then it joins the group and publishes its advertisement.

Listing Peers:

We retrieved all peer advertisements by checking local copies and accessing remote peers and got peer name from each advertisement. Then we added the peer names to the peer list.

Send Message:

We took instances of rendezvous Service, discovery Service. and pipe Service. Peer creates output pipe and create pipe advertisement. Peer finds remote pipe advertisements and gets its input pipe. Bind the input and output pipes. Close input pipe, add data to message and send.

Receive Message:

Peer creates input pipe. Then it creates input pipe advertisement and publishes. It gets message from pipe message event and display it.

Sharing Files:

We initialized Content Manager and got CMS object. We stored all the files in the selected shared path in an array. We retrieved all their descriptive attributes like name, size, content type, checksum etc and displayed in a table of shared contents. A user can share different files with its peerGroup. When the user shares a file, the overlay computes its entire attribute and the configuration is saved in the file ./userData/username/clientConf/groupname. The next time that the peer joins the overlay, this configuration will be loaded and the files will be checked for possible changes. Every pre-fixed interval of time (15 minutes in our program) the information on shared files are collected advertisements to be published in the net. We used CMS to store metadata in broker peers. We have used clock before and after this module to check the time taken to share each file.

Searching Files:

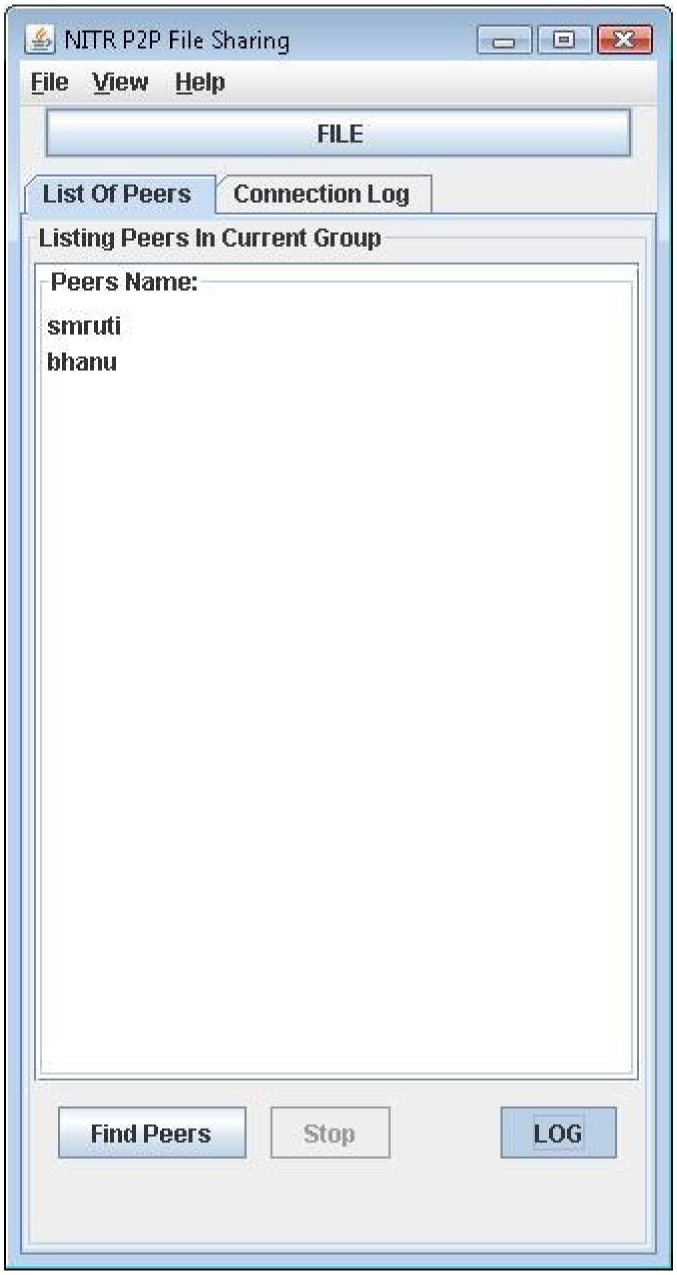
A user can search a file with a few specified parameters (file name) with the objective to find certain files that are shared in the net. When the user introduces the parameters, the peer client sends this request to the broker of the peergroup. Upon receiving the request, the broker will find in the correct advertisements all the files that the users share, according to the search parameters. Every advertisement has the files that a user shares in a peergroup. The broker needs to assure that the peer is connected. So the broker checks if there is a “Client Peer Information Advertisement” of this peer and in case the broker finds it, it means the peer is connected because peer info advertisements have a very short lifetime. Once the broker checked the advertisement of a peer, it will start checking one by one all the files that this peer shared in this peergroup and will save in a list all the files that have all the attributes that the user have requested. Then it sends the complete list of the found files to the requesting peer, which shows this information to the user in a table, from which the user can request to download different files.

Download Files:

User selects a searched file from the table and then selects the path to save. From the content advertisement we can find the file source peer. We can use simple pipe, JxtaSocket or JxtaBiDiPipe to transfer data between peers. We have used Secure Unicast Pipes to transfer data. After the file is downloaded completely we calculate its checksum and compare it with the file advertisement checksum. If checksum matches then download is validated. We have used clock before and after this module to check the time taken to download that file.

4.3 Results:

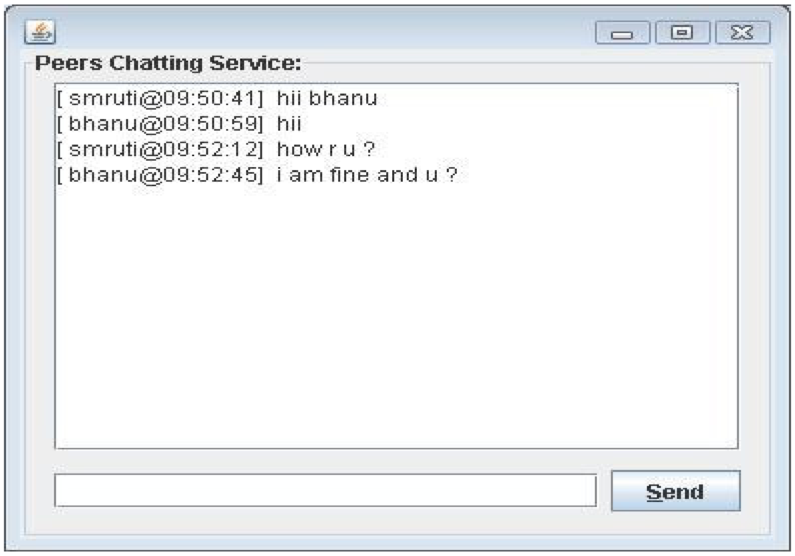
Main Window showing PeerList snapshot:



smruti
bhanu

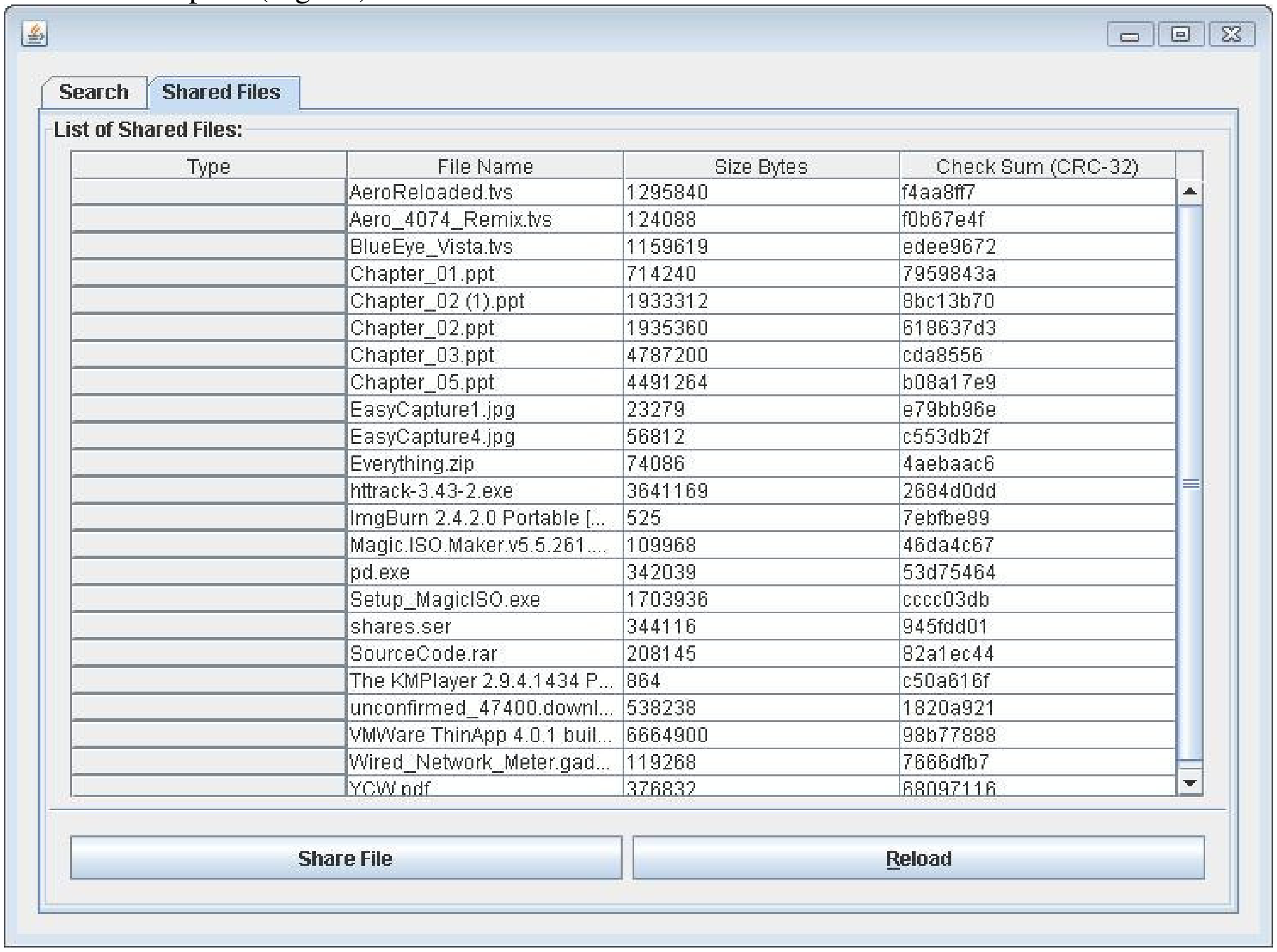
Fig: 4.1

Chat Window Snapshot(Fig: 4.2):



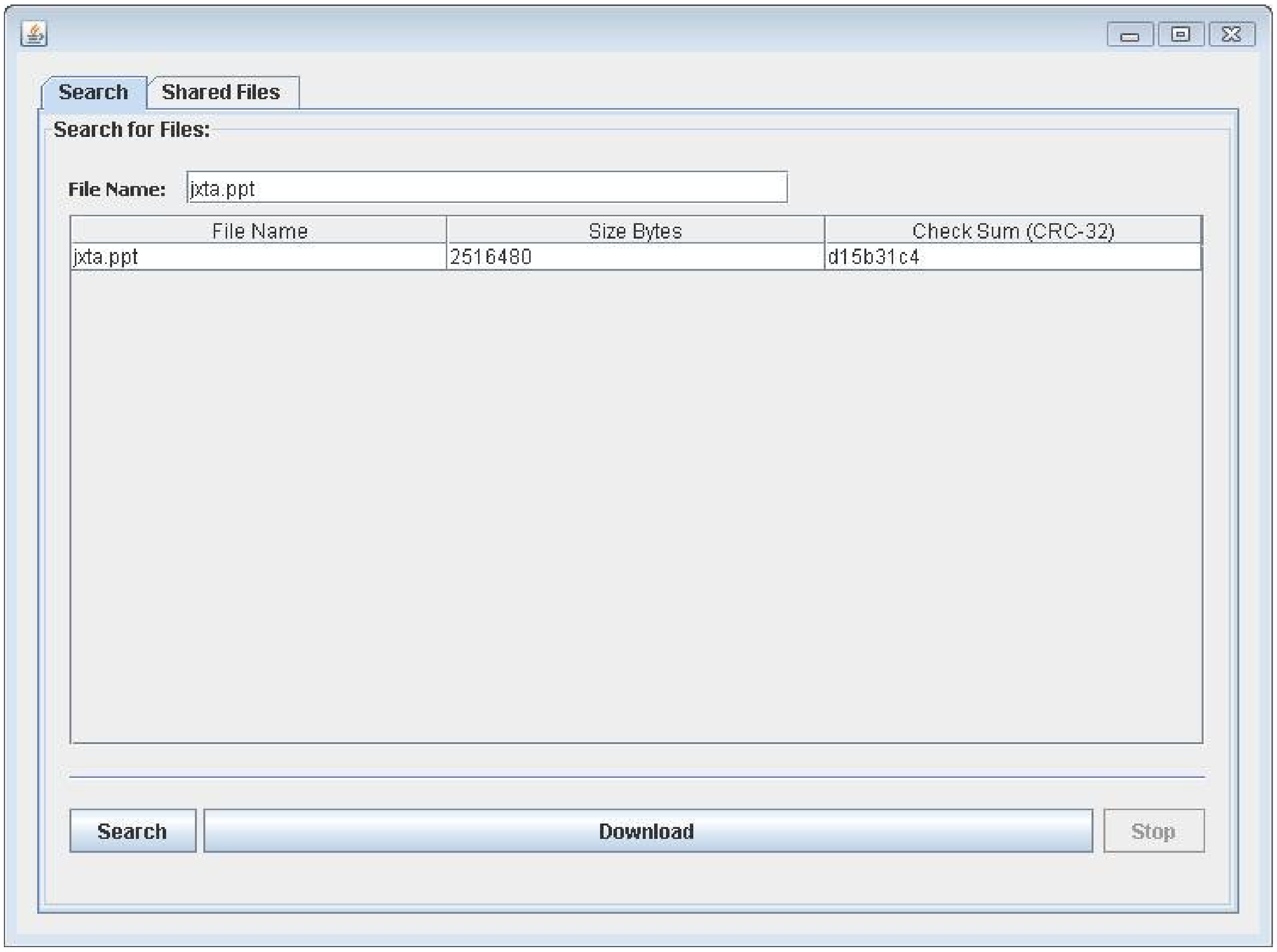
```
[ smruti@09:50:41]  hii bhanu
[ bhanu@09:50:59]  hii
[ smruti@09:52:12]  how r u ?
[ bhanu@09:52:45]  i am fine and u ?
```

Share File Snapshot(Fig 4.3):



Type	File Name	Size Bytes	Check Sum (CRC-32)
	AeroReloaded.tvs	1295840	f4aa8ff7
	Aero_4074_Remix.tvs	124088	f0b67e4f
	BlueEye_Vista.tvs	1159619	edee9672
	Chapter_01.ppt	714240	7959843a
	Chapter_02 (1).ppt	1933312	8bc13b70
	Chapter_02.ppt	1935360	618637d3
	Chapter_03.ppt	4787200	cda8556
	Chapter_05.ppt	4491264	b08a17e9
	EasyCapture1.jpg	23279	e79bb96e
	EasyCapture4.jpg	56812	c553db2f
	Everything.zip	74086	4aebaac6
	httrack-3.43-2.exe	3641169	2684d0dd
	ImgBurn 2.4.2.0 Portable [...	525	7ebfbe89
	Magic.ISO.Maker.v5.5.261....	109968	46da4c67
	pd.exe	342039	53d75464
	Setup_MagicISO.exe	1703936	cccc03db
	shares.ser	344116	945fdd01
	SourceCode.rar	208145	82a1ec44
	The KMPlayer 2.9.4.1434 P...	864	c50a616f
	unconfirmed_47400.downl...	538238	1820a921
	VMWare ThinApp 4.0.1 buil...	6664900	98b77888
	Wired_Network_Meter.gad...	119268	7666dfb7
	YCW.pdf	376832	68097116

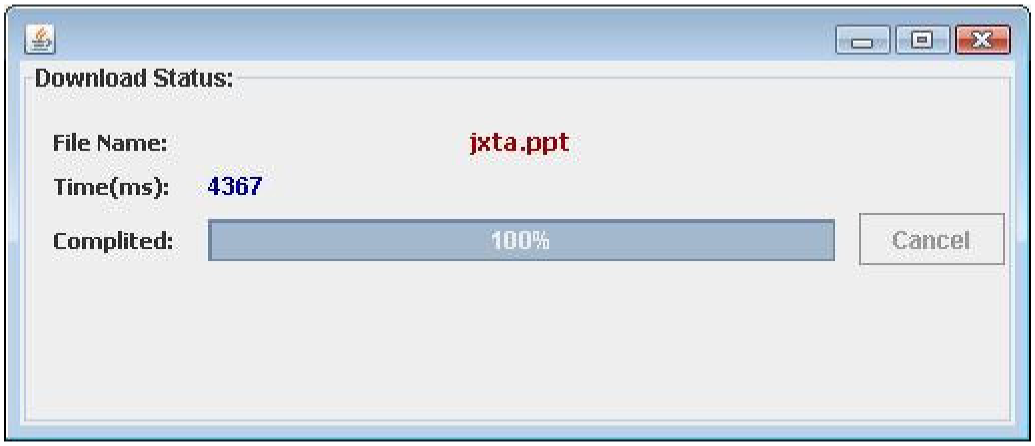
Search File Snapshot(Fig 4.4):



File Name: jxta.ppt

File Name	Size Bytes	Check Sum (CRC-32)
jxta.ppt	2516480	d15b31c4

Download File Snapshot(Fig 4.5):

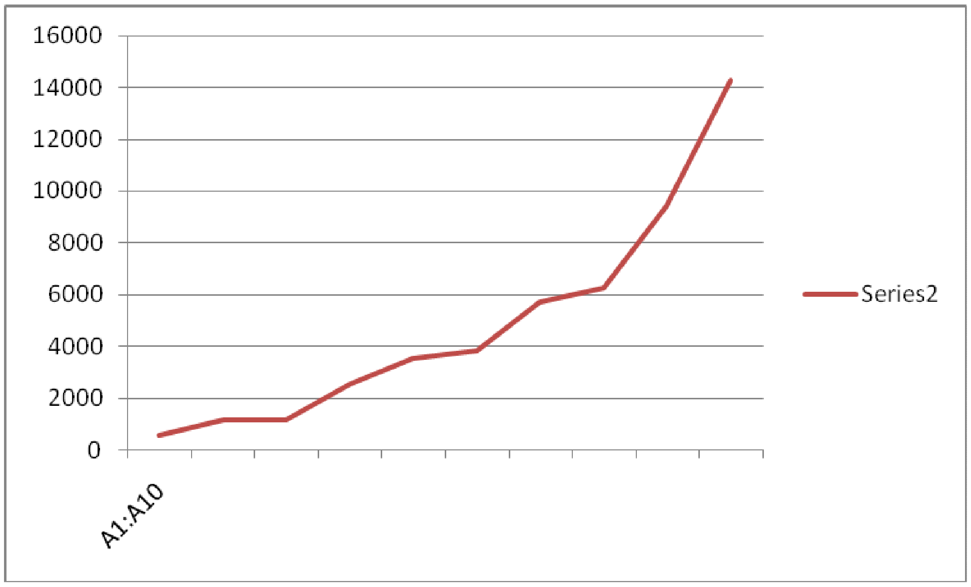


File Name: jxta.ppt

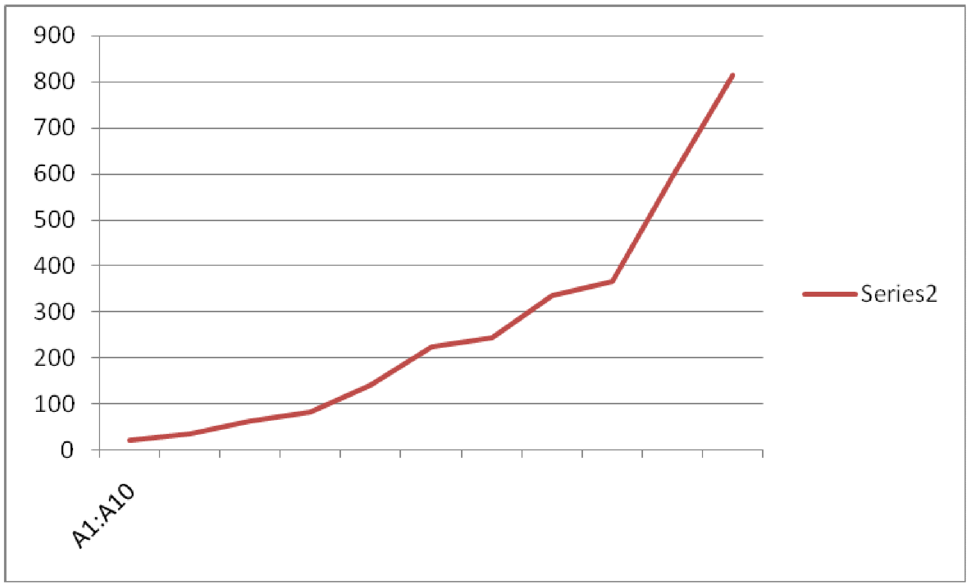
Time(ms): 4367

Complited: 100%

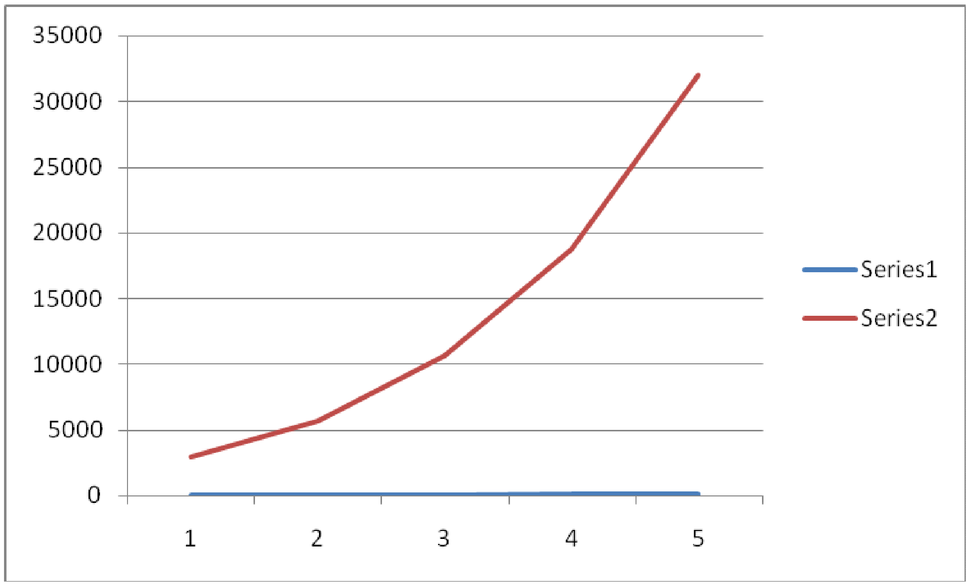
Download Time vs File Size(Fig 4.6):



Upload Time vs File Size(Fig 4.7):



Adv. Publication Time vs No of shared Files(Fig: 4.8):



CHAPTER 5

Conclusion and Future Works

In this thesis we have presented extensions of JXTA protocols to support development of file sharing systems in JXTA-based P2P applications. The extension of the basic protocols of JXTA required the definition and management of file advertisements to enable file sharing as well as efficient searching of files. Several issues were encountered in extending the JXTA protocols for file advertisement. Among them, the efficient management of advertisement was identified and studied. Indeed, file advertisement could signify an important computational burden to broker peers due to the large size of file advertisements and the large quantity of file advertisements published by different peers that must be processed by the broker.

Another important issue is that of the lifetime of file advertisements in order to increase the reliability of the file systems in the P2P net. Certainly, short values of life- time would imply very frequent publishing of advertisements, which on turn, could provoke broker peers collapse; on the other hand, large values of the lifetime would imply the "existence" of files in the P2P net while peers sharing the files are already disconnected from the net. Our approach proposes a separation of types of advertisement in peer proper information advertisement, file advertisement and group files advertisement. This separation allows for an adequate fine tuning of the lifetime parameter according to the type of advertisement. By this approach we are able to alleviate the computational load of broker as well as increase the reliability of the file systems in the P2P net.

The approach has been experimentally studied by deploying the JXTA-based P2P network in a small LAN network of our institute.

In our future work we plan to complete the experimental study by considering a larger P2P network (PLANET LAB). Also, we would like to study the differences between P2P file sharing and Web traffic of server-mediated file sharing approaches, which could reveal important differences in both approaches. In the same context, it would be interesting to study possible bandwidth savings in JXTA-based P2P file-sharing architectures. Finally, we are interested to apply our JXTA-based P2P file sharing system to support cooperative work of online teams at our virtual campus.

REFERENCES

References :

1. F. Xhafa, R. Fernandez, Th. Daradoumis, L. Barolli and Santi Caball´ e. Improvement of JXTA Protocols for Supporting Reliable Distributed Applications in P2P Systems. In Proceedings of Network-Based Information Systems, First International Conference, NBiS 2007, Regensburg, Germany, September 3-7, 2007, Lecture Notes in Computer Science, Vol.
4658, 345-354, Springer, 2007

2. J.E. Riasol and F. Xhafa. Juxta-cat: a jxta-based platform for distributed computing. Proceedings of the 4th Int. Symposium on Principles and Practice of Programming in Java, 72–81, 2006. ACM Press

3. Fatos Xhafa, Leonard Barolli, Raul Fern´ andez, Thanasis Daradoumis, Santi Caball´ and Vladi Kolici. Extending JXTA Protocols for P2P File Sharing Systems. International Conference on Complex, Intelligent and Software Intensive Systems Issue , 4-7 March 2008 Page(s):35 - 40 © 2008 IEEE.

4. S. Oaks, B. Traversat, and L. Gong. JXTA in a Nutshell. O'Reilly, 2003

5. S. Man Lui and S. Ho Kwok. Interoperability of peer-to-peer file sharing
protocols. SIGecom Exch.Journal, 3(3), 25–33, ACM Press, New York,
NY, USA, 2002.

6. D. Brookshier, D. Govoni, N. Krishnan, and J.C Soto. JXTA: Java P2P Programming. Sams Publishing, 2002

7. Joseph D. Gradecki. Mastering JXTA Building Java Peer-to-Peer Applications.

Wiley Publishing, Inc, 2002

8. JXTA Java™ Standard Edition v2.5: Programmers Guide. Sun Microsystems, Inc, 2007

9. Th. Karagiannis, A. Broido, M. Faloutsos and Kc Claffy. Transport layer identification of P2P traffic. Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, 121–134, Taormina, Sicily, Italy, ACM Press, New York, NY, USA, 2004.

10. Daishi Kato. GISP: Global Information Sharing Protocol, a distributed index for peer-to-peer systems. Proceedings of the Second International Conference on Peer-to-Peer Computing (P2P'02)

11. Xin Xiang, Yuanchun Shi, Ling Guo. Rich Metadata Searches Using the JXTA Content Manager Service. Proceedings of the 18th International Conference on Advanced Information Networking and Application (AINA'04)

12. Stephanos Androutsellis-Theotokis and Diomidis Spinellis, "A survey of peer-to-peer
content distribution technologies," ACM Computing Surveys, Volume 36 Issue 4, Pages 335 – 371, December 2004

13. Ion Stoicay , Robert Morrisz, David Liben-Nowellz, David R. Kargerz, M. Frans Kaashoekz, Frank Dabekz,Hari Balakrishnanz. Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications. 149—160,ACM SIGCOMM. 2001

14. Emmanuel Stone, Tim Czerniak, Colm Ryan, Rob McAdoo. Peer to Peer Routing. http://ntrg.cs.tcd.ie/undergrad/4ba2.05/group6/index.html.

15. Mathieu Jan. JXTA Overview. IRISA Rennes – Projet Paris

16. Sing Li. JXTA 2: A high-performance, massively scalable P2P network. http://www.ibm.com/developerworks/java/library/j-jxta2/#author1. 2003