CEN 502: COMPUTER SYSTEMS II  
  
A SURVEY REPORT ON  
  
P2P FILE SHARING SYSTEM

Group Number: 32

Sandesh Shetty (1209395990)   
Mohit Nangare (1209409068)

**SURVEY**

Peer-To-Peer Network Architecture:

Peer-To-Peer (P2P) Network Architecture is used primarily for the distribution of digital media files. In a P2P network each computer acts as both a server and a client—supplying and receiving files. No real hierarchy among peers and so it is also referred as distributive architecture and is widely used for file sharing. Such a decentralized network uses resources more efficiently than a traditional Client-Server Network and is less vulnerable to systemic failure and its robust which means if one peer stops working for some reason, the network does not crash and it continues working. P2P networks are largely driven by online file sharing. [4]

The different exiting P2P architectures for File Sharing based on the availability of one or more servers, and to what extent the peers depend on the services provided by those servers are:

1. Centralized P2P Systems.
2. Decentralized P2P Systems.
3. Hybrid P2P Systems.

* **Centralized P2P Systems**

This type of P2P Architecture combines both the features of Client-Server Network and Decentralized Network. Like a client-server system, there are one or more central servers, which help peers to locate their desired resources or act as task scheduler to coordinate actions among them. To locate resources, a peer sends messages to the central server to determine the addresses of peers that contain the desired resources, or to fetch work units from the central server directly. However, like a decentralized system, once a peer has its information/data, it can communicate directly with other peers without going through the server anymore.

In general, P2P computing by definition emphasizes the equality of functions and responsibilities of all peers, which play the roles of both resource providers and resource requestors. A centralized P2P system enjoys two merits:

* It speeds up the process of resource location, and guarantees finding all possible peers that maintain the desired files.
* It is easy to maintain, organize, and administer the whole system through the central server.

As in all centralized systems, this category of P2P systems is susceptible to malicious attacks and single point of failure. Moreover, the centralized server will become a bottleneck for a large number of peers, potentially degrading performance dramatically. Finally, this type of system lacks scalability and robustness.

Some examples of this architecture include Napster, SETI@home and BOINC. [5]

NAPSTER:

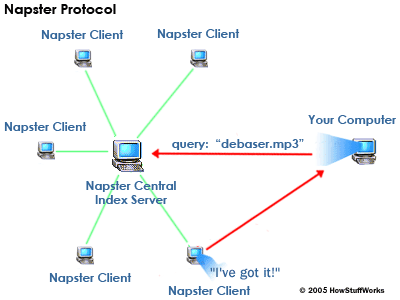


Figure 1: P2P ARCHITECTURE OF NAPSTER

Napster provides us three functionalities: joining Napster, resource discovery and downloading files. Firstly, any client will send a request for joining Napster to the central server and then complete the registration process. Now, that client is a Napster Client. The data of this Napster Client is now stored in the centralized server. Secondly, a peer sends a resource discovery query to the central server. The centralized server looks up in its repository and returns the resource and the index of the peer/s that has that resource which was asked in the query. Finally, the query client establishes the connection with the client which contains the desired resources.

* **Decentralized P2P Systems**

In a P2P system of fully decentralized architecture, each peer is of equal responsibilities and rights, so that none is superior to the other. The system can run smoothly while peers join or leave the network at any time. Several existing P2P systems belong to this category, such as the original Gnutella, FreeNet, FreeHaven, Chord, PAST, OceanStore, etc. Based on the criterion of “the structure of overlay network”, Decentralized P2P systems can be further classified into two subcategories:

* Unstructured P2P Systems.
* Structured P2P Systems.

Unstructured P2P Systems:

In unstructured P2P systems, the content resided on each peer has no relationship to the “structure” of the underlying overlay network. Each peer chooses the content to store at will. So given a query, it is hard to know precisely the location where the results are. The solution is to search all or a subset of the peers in the network. Most of unstructured P2P systems adopt broadcast-based query routing strategies to discover expected resources. The advantage is that it can easily accommodate a highly transient node population, since the query can be spread to a large number of peers within a short time. The disadvantage is that it widely floods the query to many peers in the network no matter whether they can answer the query or not, which causes heavy network traffic of exponential query messages. Examples of the unstructured P2P systems are the Gnutella, FreeNet, and FreeHaven. [6]

GNUTELLA:

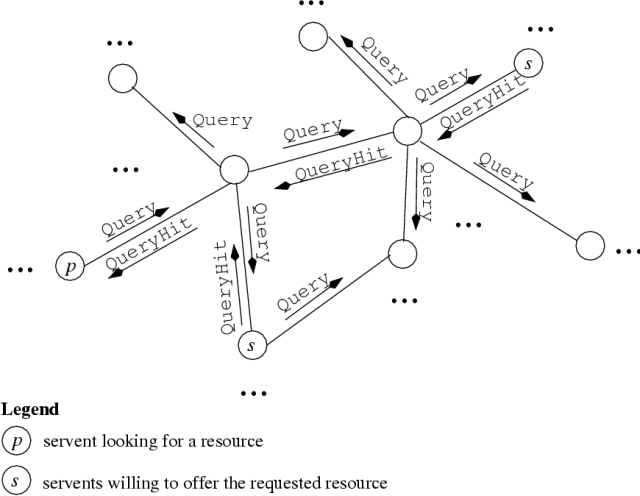


Figure 2 P2P ARCHITECTURE OF GNUTELLA

The basic operations of Gnutella include joining or leaving network, searching and downloading files. Here, when a peer joins the Gnutella Network it sends a “PING” message to all its neighbors and if it doesn’t receives any acknowledgement then it updates it list of neighbors. Now, when the peer needs a file it sends a “QUERY” message to all its neighbors and they in turn relay it to their neighbors. The peer having the desired file replies with a “QUERYHIT” message and it is routed back along the same path of “QUERY”. Now the original peer may have many replies at hand, and can choose some peers to connect and then download the desired files.

Structured P2P:

In a structured P2P system, there is a certain mechanism to determine the location of files in the network. For example, by applying a distributed hash function (e.g., SHA-1) on both files and peers’ name, files are placed on the peers whose hash values are numerically close to that of the files. Thus, a mapping is built up between files and peers. Given a query, the location of desired files can be decided quickly and deterministically, so it is unnecessary to aimlessly visit unrelated nodes to find the answers to the query. As a result, the efficiency of searching and routing can be improved greatly. Usually, peers need to maintain some data structures (e.g., distributed hash table) to guarantee the correctness and efficiency of query routing. When peers join and leave the network very frequently, the cost to maintain the routing information is quite high. Systems such as PAST, Chord and OceanStore belong to this category. [7]

PAST:

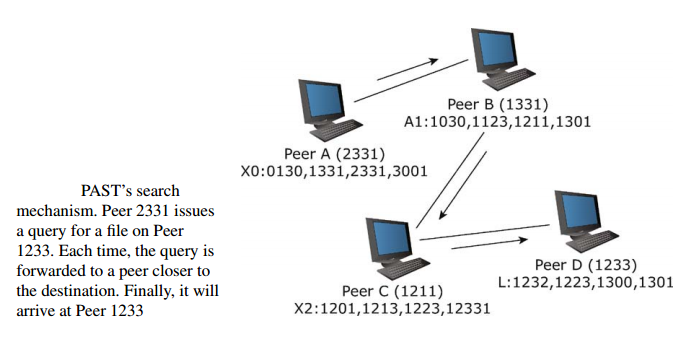


Figure 3 P2P ARCHITECTURE OF PAST

In the PAST system, each node is assigned a 128-bit node identifier that is obtained by hashing the node’s public key using a hash function such as SHA-1. Similarly, each file stored in the PAST is assigned a 160-bit file identifier that is derived from hashing the file name, the owner’s public key, and a randomly chosen salt. When a file is inserted into PAST, it is put on ‘k’ peers whose identifiers are numerically closest to the 128 most significant bits of the file identifier, among all live peers. Given a lookup for a file, a peer will forward the request to a peer whose identifier has a longer prefix match with the file identifier than itself. If such a peer cannot be found, the message will be forwarded to a peer whose identifier shares the same length with the file identifier as the present node, but is numerically closer to the file identifier. To achieve this, each peer needs to maintain a leaf set and a routing table, whose entry maps a peer identifier to its corresponding IP address. Finally, the request will be reliably routed to one of the k nodes that store the file and near to the node issuing the lookup. [8]

* **Hybrid P2P Systems**

Hybrid P2P systems draw advantages from the other types of P2P architectures: Centralized and Decentralized, while distinguish themselves from the other two types by their elegant auxiliary mechanisms that facilitate resource location. In some P2P systems of hybrid architecture, there are some peers possessing much more powerful capabilities and having more responsibilities than other peers, which are usually referred to as “super” peers. These peers form an “upper level” of a hybrid system, which provides similar services for the ordinary peers as the central server does in a Centralized P2P System. The common peers, on the other hand, can enjoy much more services from the peers in the “upper” layer, especially in the process of resource location. A “super” peer has to not only coordinate the operations among the peers under its supervision, but also perform the same operations by it and contribute its own resources as the common peers do. Hybrid P2P Systems have many advantages, such as optimizing network topology, improving response time and saving system resource consumption, and avoiding a single point of failure and corresponding applications of such P2P systems in real life are BestPeer, PeerDB, PeerIS and CQBuddy. [9]

BestPeer:

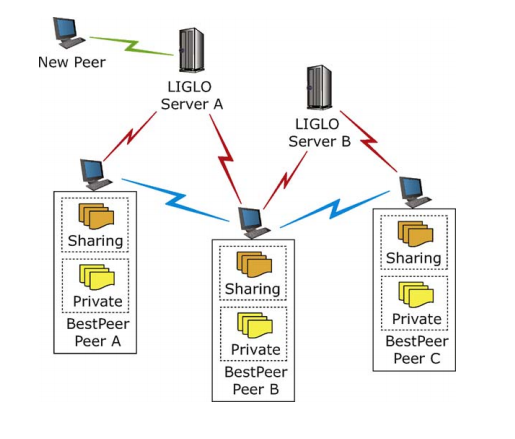


Figure 4 P2P ARCHITECTURE OF BESTPEER

For the first time when a peer joins the system, it needs to register itself with a LIGLO server. When a LIGLO server receives a registration request from a peer, it creates a global and unique identifier BPID (BestPeer ID) for that peer. This BPID is then returned to the new node. Additionally, the LIGLO server also returns to the new node a list of BPID and IP address of current online peers registered to the server. The new peer can communicate with peers in this list directly without going through LIGLO servers. However, since a peer can leave the system anytime without notifying the server, the IP address of a peer may be incorrect. In this case, the peer simply removes the unreachable peer from the list. To avoid this case, LIGLO servers often check IP address of their registered peers and discard offline peers from the list. Now, when a peer is in the system, it can query and access shared data from other peers. The basic idea of query processing in BestPeer is similar to that in Gnutella. In particular, when a node wants to issue a query, it simply broadcasts the query to its neighbor peers, which in turn forward the query to their neighbor peers, and so on. When a peer receives a query request, if it contains the queried data, it returns the result directly to the query initiator. Finally, when a node rejoins the system, i.e., it is not the first time the node joins the system, the node sends its IP address together with its BPID to the LIGLO server it has been registered before. If this IP address is different from the previous registered IP address, the LIGLO server updates the new IP address for the node. [10]

**References**

[1] <https://en.wikipedia.org/wiki/File_sharing>

[2] <https://en.wikipedia.org/wiki/Network_architecture>

[3] <http://www.britannica.com/technology/client-server-architecture>

[4] <http://www.britannica.com/technology/P2P>

[5], [6], [7], [8], [9], [10] <http://www.springer.com/cda/content/document/cda_downloaddocument/9783642035135-c2.pdf?SGWID=0-0-45-855488-p173920223>

Figure 1 <http://computer.howstuffworks.com/kazaa3.htm>

Figure 2 <http://www2002.org/CDROM/refereed/68/p68-cornelli01.png>

Figure 3, Figure 4 <http://www.springer.com/cda/content/document/cda_downloaddocument/9783642035135-c2.pdf?SGWID=0-0-45-855488-p173920223>