Introduction on Peer to Peer systems

Georges Da Costa
dacosta@irit.fr

Goal of this Lecture

- What can P2P do, not only as a buzzword
- What it can't do
- Shows some examples & algorithms

A Survey and Comparison of Peer-to-Peer Overlay Network Schemes, by Eng Keong Lua and al.

in *IEEE Communications survey and tutorial* March 2004

Harnessing the Power of Disruptive Technologies

published by O'Reilly, 2001

Universal

What have in common

- Net Meeting, Skype, Ekiga
- Irc, Msn, Icq, Jabber
- Kazza, Freenet, Napster, Gnutella
- Seti@Home, Folding@Home
- Ebay, Flickr, MySpace

Definition

Philosophical one

Participants gathering their resources in order to achieve a common goal

Why ?

Available resources

- Large Hard Drives
- Powerful CPUs
- Correct connexion to Internet

Users want

- More freedom
- No link to commercial companies
- No infrastructure cost

A new (?) solution : Peer to Peer systems

Definition

Participant gathering their resources in order to achieve a common goal

- Computers are running the same code
- There is no global view of the system
- View is limited to neighboors
- Everyone has the same rights and duties

Peer-to-Peer: New name, old concept

An architecture already there

- Internet connects most of existing computers
- Most computers are not fully used
  - Idle time $> 75\%$ on personal computers
  - Storage systems are mostly empty

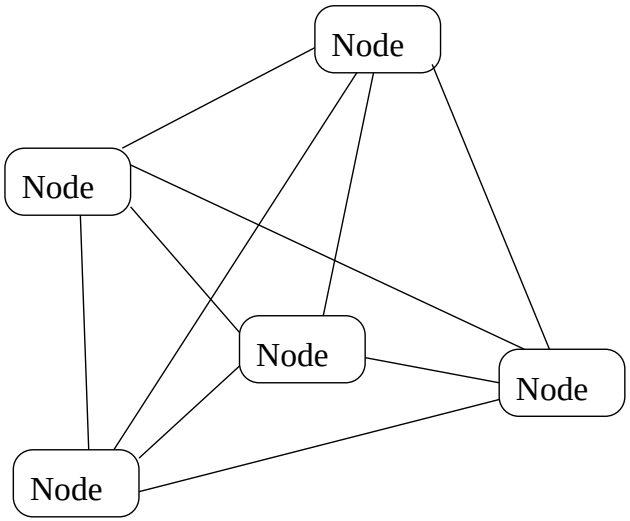
Already used between servers

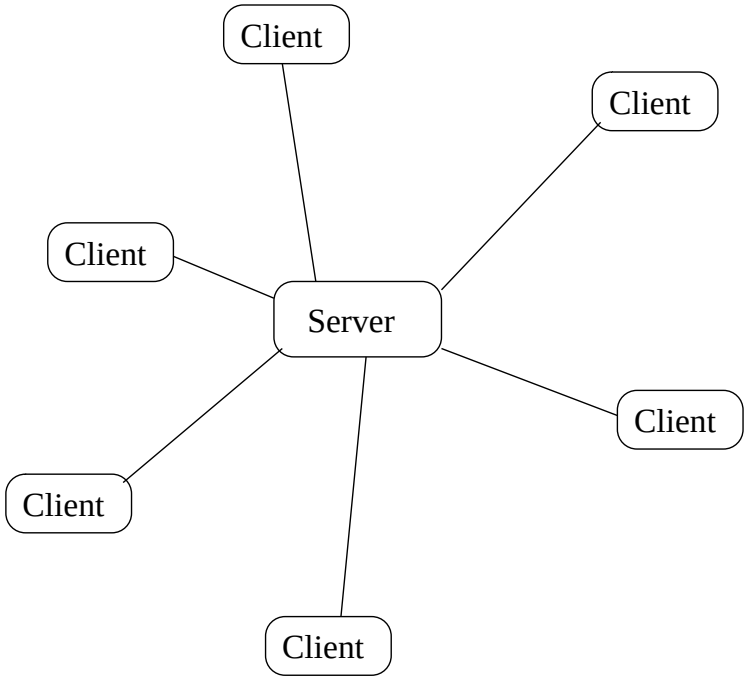
- Services: Usenet, DNS
- Routing: IP Routing

Comparison with Client/Server

- In client/Server each node is either a Client or a Server. Usually there are a few Servers and lots of Clients.
- Client/Server systems suffer from *single point of failure*.
- Client/Server are mostly static, at least the Servers. Peer to Peer systems are dynamics.
- Client/Server systems need *human* administrators
- Client/Server does not scale

Comparison with Client/Server II





Comparison with Client/Server II

When a new participant joins a service, the service increase the resource consumption

- **Client/Server** : increases the server power/connectivity
- **Peer to Peer** : uses the resources given by the participant

Not so easy

Wanted

- Scalability (1K,100K,1M nodes)
- Dynamicity
- Security (user, task)
- Transparent
  - For the user (CPU,memory,disk)
  - For the network (NAT/Firewall)
- Heterogeneity
- Self-organization
- Participation (66% of Free riders)

Self-organization

Participants

High volatility & voluntary

- No central administration
- Resource discovery
- Heterogeneity
  - Hardware
  - Users (15% of users have 94% of files)
- Distribution of the resources
- Trust

What's not new

Partial solutions

- Scalability : Farm of web servers
- Dynamism : Cell phones
- Fault tolerance : Redundant servers

Current Peer to Peer systems

Available applications

- *File sharing*
- Distributed storage
- Content delivery
- Distributed computing
- Telephony/Chat
- Games

Current Peer to Peer systems (cont)

Widely used

2004: According to British Web analysis firm CacheLogic, BitTorrent accounts for 35 percent of all the traffic on the Internet (more than all other peer-to-peer programs combined) and dwarfs mainstream traffic like Web pages

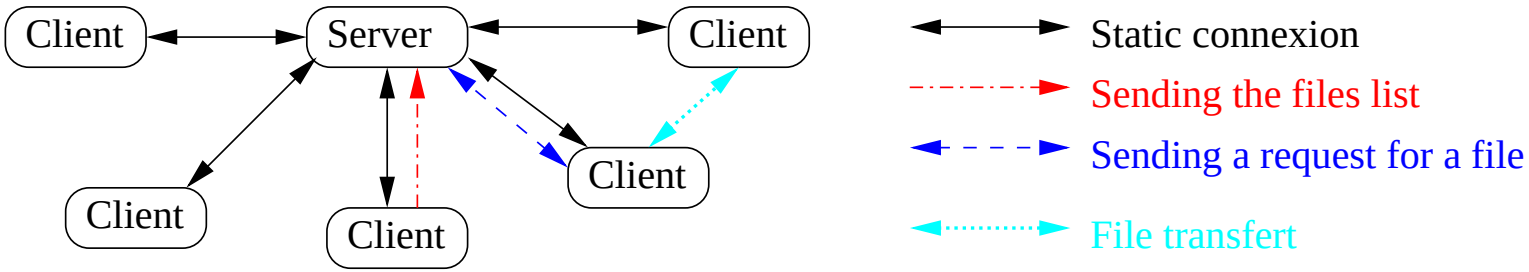
Start-ups

- Skype (ok, no more a small start-up)
- BitTorrent
- UbiStorage

Two worlds

- Internet Users
  - Problem of security
  - Large scale
  - No control
  - Motivation needed
- Private Area (Corp., Univ.)
  - Other mean of security
  - Medium to large scale
  - Total control

Index Method



- Users send the list of their files to a server
- To find a file, you send a request to the server
- It answers with the list of clients owning the file
- You directly contact the owners for the transfer

Index Method II

Systems

Napster, Mojonation, Yaga, Filetopia, Seti@Home

Problems

- Scaling
- Price
- HotSpot
- Attack
- Single point of failure

Useful when...

- Small number of client
- Need a total control of transfers (video game industry)
- Performance is more important than cost

BitTorrent

Same approach as Napster, but :

- Downloads are done in parallel
- One server per file
- Server manages all the details of transfers
- Server enforces the rule *The more you share, the more you get*

Differences

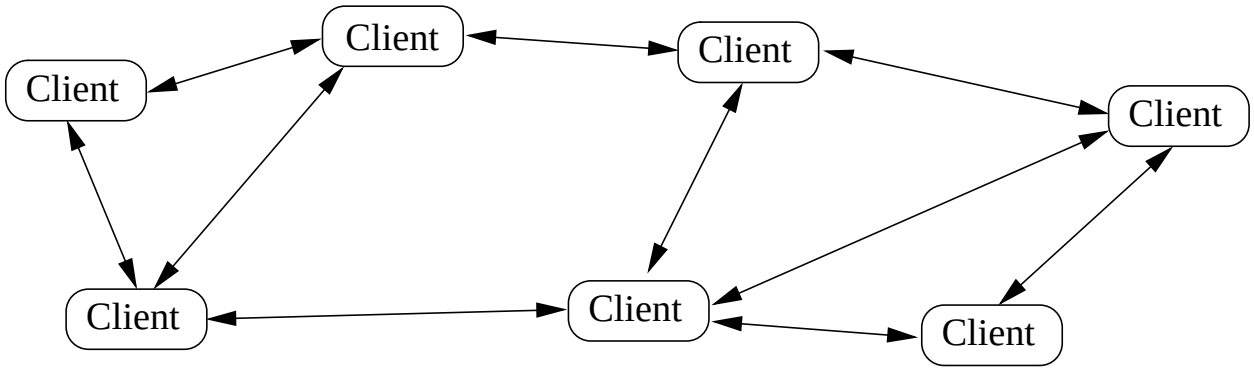
- Specialized for large files
- Distributed due to the *One server per file* rule

Privacy

No privacy

- Napster : The server knows all transfers
- BitTorrent : For each file, a server knows all transferts

Flooding



- You send your request to your neighbors
- They forward it to their neighbors, and so on until reaching the *Time To Live* depth
- Users with files corresponding to the request answer

Flooding II

Systems

Gnutella, Direct Connect

Characteristics

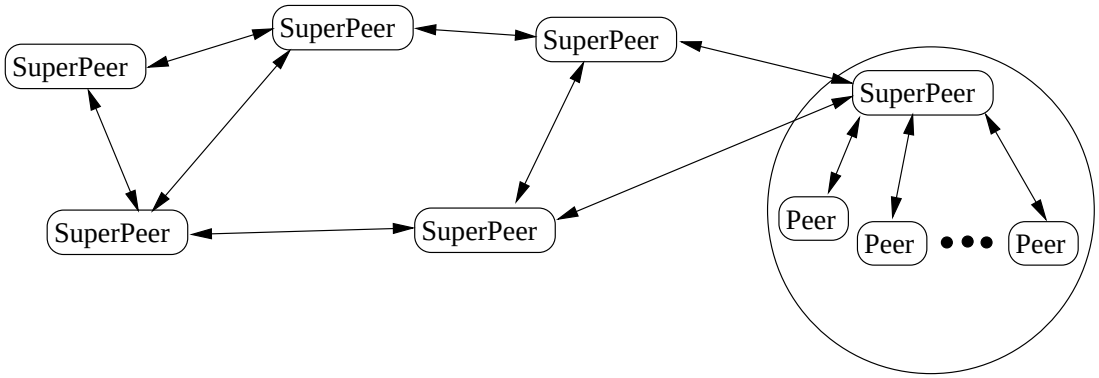
- Distributed structure
  - No single point of failure
  - Denial of service difficult (but possible)
- Not scalable
  - Resource consumption (network)
  - Not complete answers

Privacy

Average to good privacy

- Onion routing (good privacy)
- No global view of the system
- Usually easy to obtain the shared list of a node
- Difficult to have a global impact

Super Peers



Super Peers act as local servers

- Some reliable nodes act as super peers
- Super peers communicate in a gnutella way
- Each super peer acts as a local server for several peers

Super Peers II

Systems

Gnutella2, Kazaa

Characteristics

- Less distributed structure
  - Some nodes are more loaded
  - Some nodes are more important
- Scalable
  - Less resource consumption due to limits of number of answers