Peer-to-Peer Systems

Tales from the edges of the internet

Intro

Peer-to-peer systems are VERY different to standard IT within companies or between clients and companies – both technically and socially. Technically they ADD bandwidth and compute power per new node/participant. Socially they distribute control over IT infrastructures. By looking at p2p systems we can acquire new views on distributed systems: Extreme redundancy, anonymity and direct user enablement. And the possible downsides: reliability and security problems.

What makes p2p different? In the past most of the differences were a consequence of the peer not being part of an establised IT-system with full maintenance of security, systems etc. and with hosts that have a static identity. The peers typically lived at the „edge“ of the internet and this requires new and different answers to some well known distributed systems problems. Today p2p technology is used within companies as well (storage grids, key/value stores etc.)

Overview

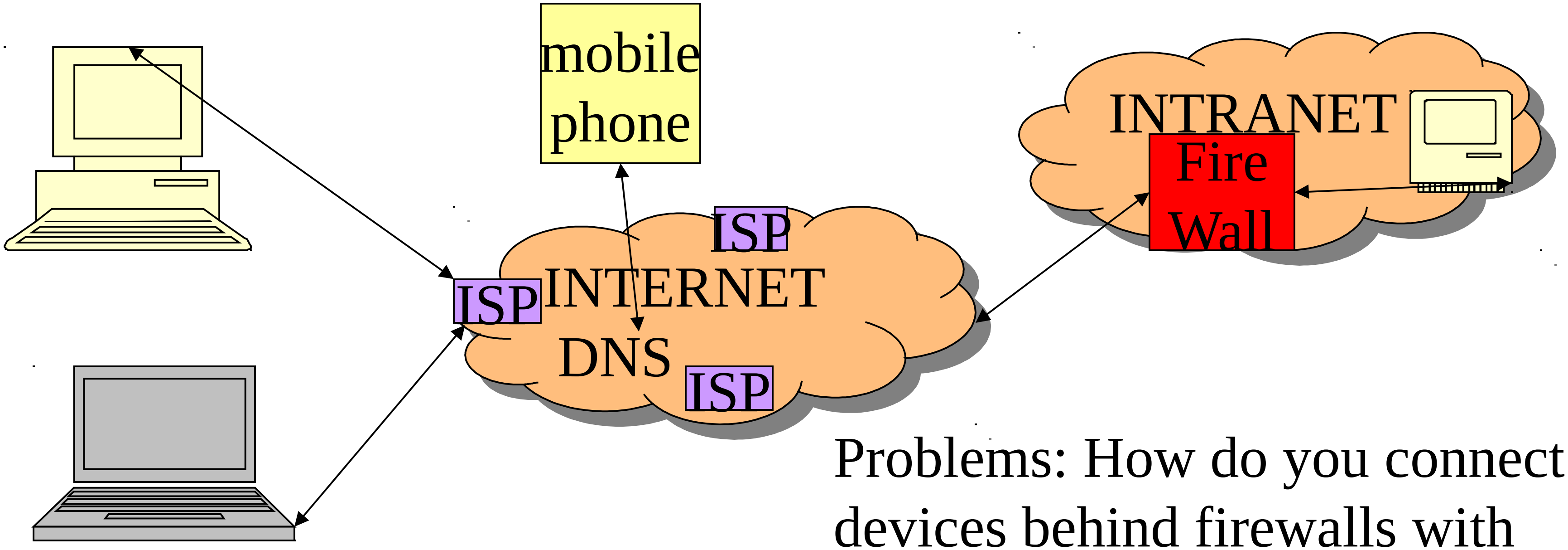
- We will first look at some p2p applications and try to come up with a classification.
- The next step is a systematic view at the basics of p2p technology with respect to environment, network structure, identity, naming and addressing etc.
- We won’t forget our typical questions about security, transactions etc.
- P2p also operates in a special social environment where free-riding and “the tragedy of the commons” pose problems which may not be SO different after all.
- And finally we take a look at the p2p framework jxta, its abstractions and goals
- The bittorrent p2p system is described and we show some empirical results about its effectivity and question some of its architecture.
- A glance at new uses of p2p technologies in storage grids, media grids etc.

Definition of Peer-to-Peer

„Peer-to-peer is a class of applications that takes advantage of resource-storage, cycles, content, human presence – available at the edges of the Internet.“ (Clay Shirkey, p2p –harnessing the power...page 22)

This definition leaves room for different distribution topologies, business models or political goals etc. as we will see.

The „edge“ of the Internet



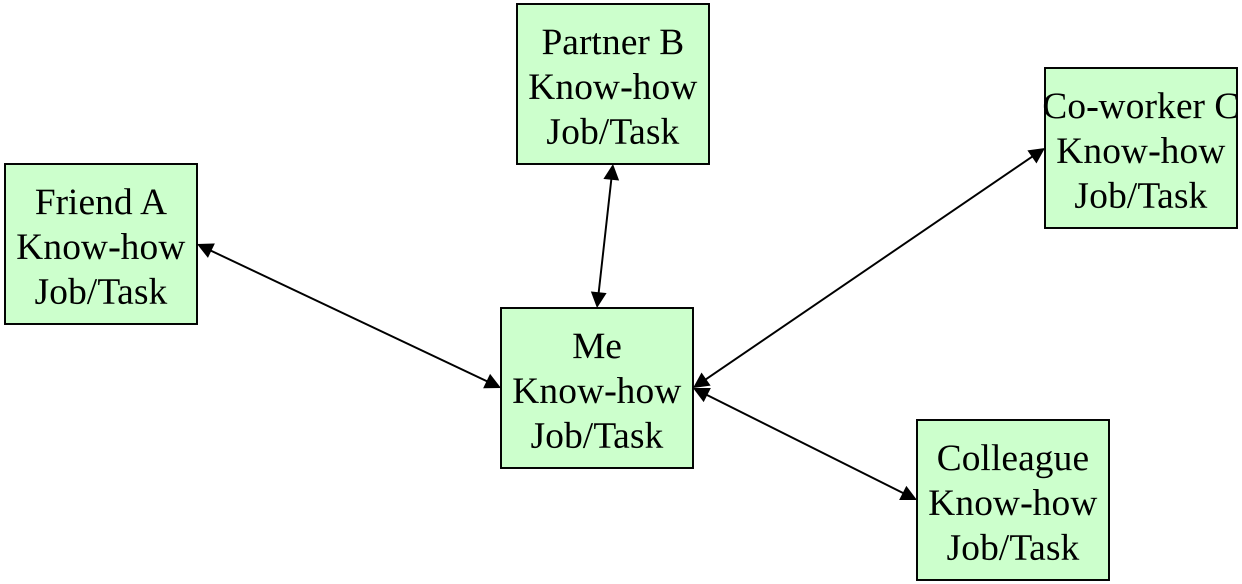
Problems: How do you connect devices behind firewalls with NAT into a p2p network?

Nodes on the edge have no fixed IP address and therefore no permanent identity (DNS name). They are also characterized by frequent off-line times and unreliable connections. But there is a huge number of them and there is a big opportunity to share computing resources, information or services. There is no central administration of those edge devices – an advantage as well as a disadvantage. And they are mostly privately used.

Consequences of being at the „edge“

- No permanent Identity: how do you create this in p2p?
- no standard addressing: how do you locate peers/resources in p2p? How do you search for something in p2p?
- Unreliable connections: how do you deal with disconnected peers?
- No central security: How do you avoid abuse of p2p systems? Who is responsible for what?
- Privately owned devices: how do you create a business model in a „share based“ community?
- No central control: how do you version resources? protect from unwanted changes?

Non-computer peer networks



Hiring personnel used to be a formal process in organization, traditionally conducted by a human-resource department. This has changed dramatically: in many cases it is the personal (peer) network of the people working on a project that is used to find new team members. The peer network has also turned into THE social safety net for professionals. See „It‘s not what you know, it‘s who you know..“

Examples of P2P Applications

- Napster: file sharing app turned into largest mp3 sharing tool
- freenet: generic file sharing app with censorship protection
- Groove: Collaboration tool
- Jabber: Instant messaging tool and generic messaging platform
- Gnutella: generic file sharing app.
- Open Cola: distributed searching.
- SETI@home: perform computations for a research project on edge machines
- Mixmaster Remailers: send mail anonymously
- Publius: publishing systems, tamper and censorship resistant. Support for anonymous publishing.
- Free Haven: anonymous storage

All these applications have some p2p features in common. The use of servers does NOT preclude an application from being a p2p app. SETI e.g. uses central servers to distribute work and collect results. But the work itself is performed on the edge.

Things to share in p2p applications

Resources

CPU cycles, file storage, routing services

Information

Music, Documents, Video Streams etc. Publishing in general

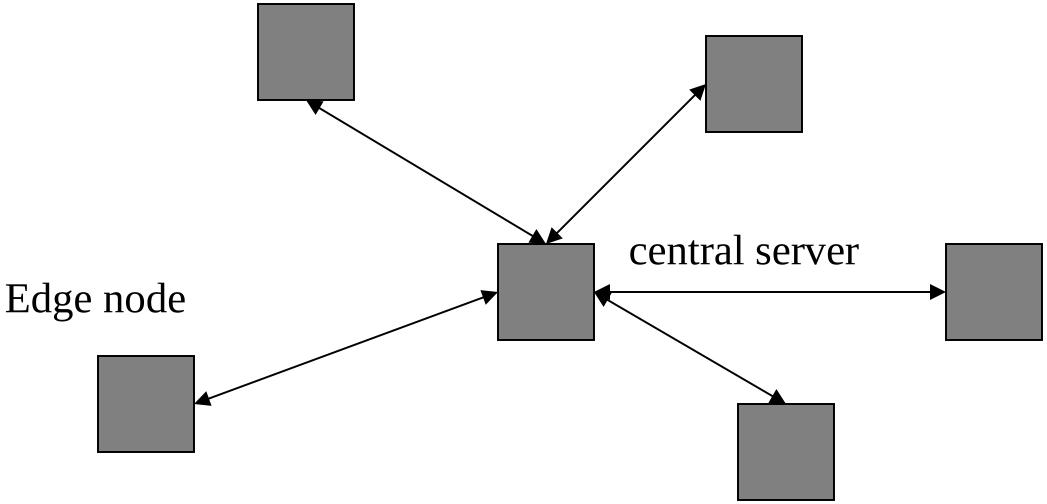
Services

Collaboration, instant messaging, presence.

Currently most p2p applications share resources in those areas. The resources differ considerably with respect to replication and copying: Information will scale easily by being copied closer to requesters. Copying services is much harder and hardware resources can‘t be copied at all

P2P Distribution Topology (1)

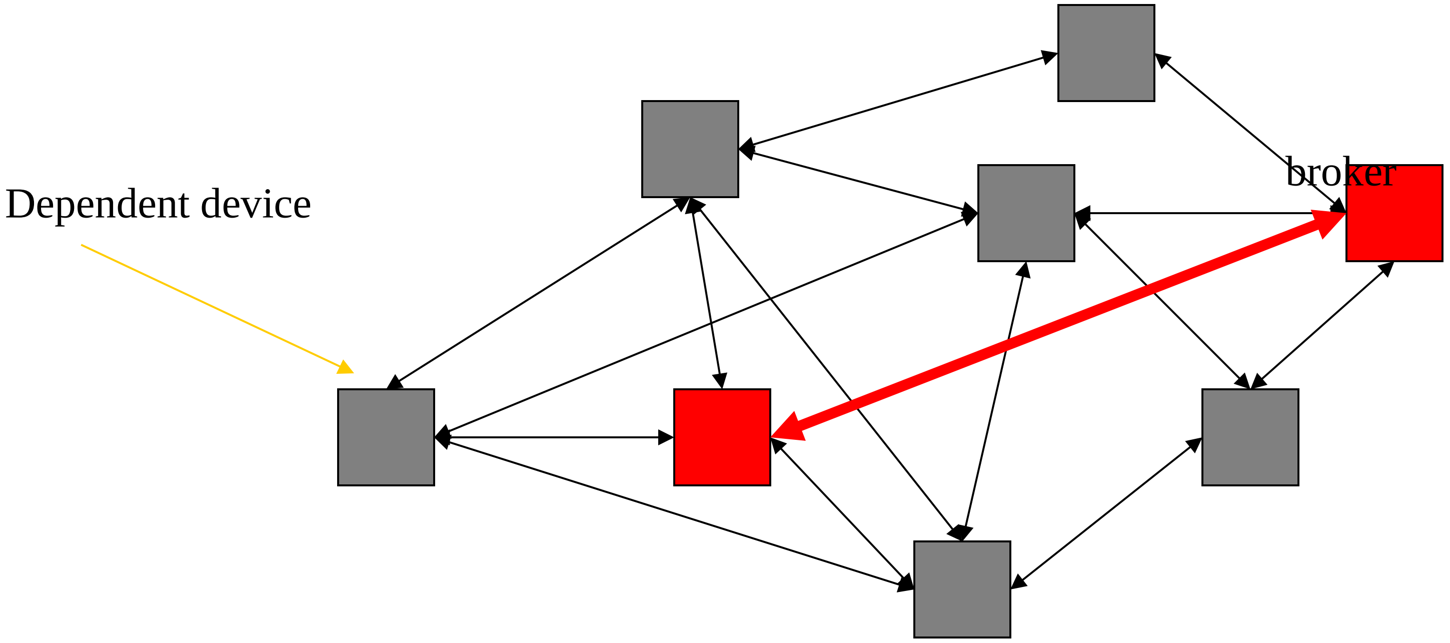
centralized



SETI@home and Jabber are p2p applications that use central servers. They are still p2p type apps because most of the work happens at the edge or because of their support for edge devices behind firewalls etc.

P2P Distribution Topology (2)

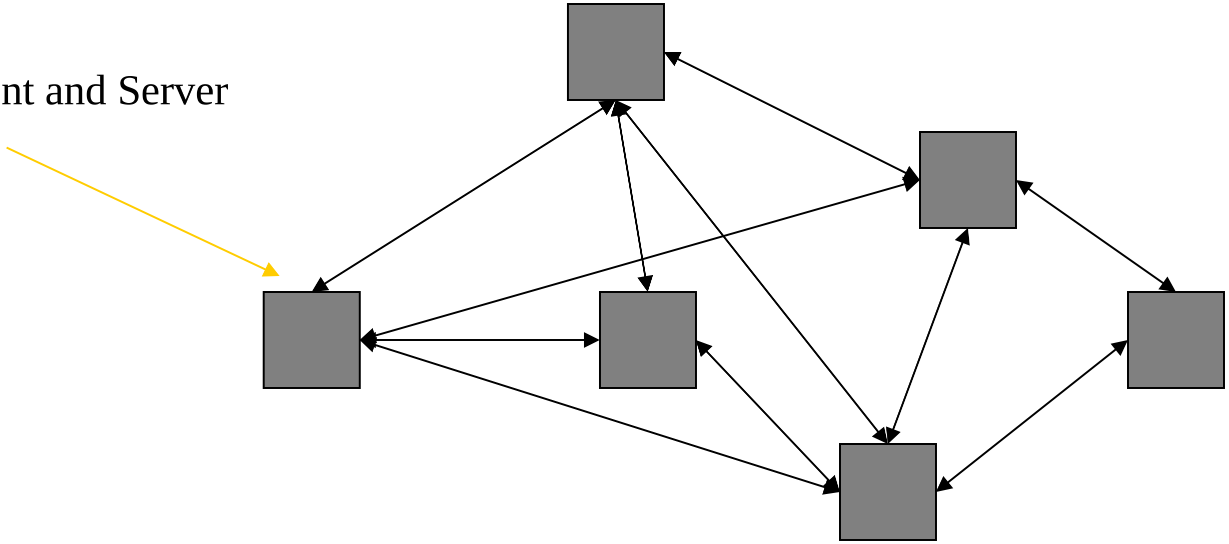
mediated



Some peers may take over a special role, e.g. serve as directories. They can either still function as regular peers or become dedicated servers (brokers) which mediate requests. The real question is: do they only mediate requests and then let the peers talk to each other directly or do they play central server? The special peers can form a hierarchy, e.g. like DNS. Broker peers hold meta-data on resources.

P2P Distribution Topology (3)

Totally distributed



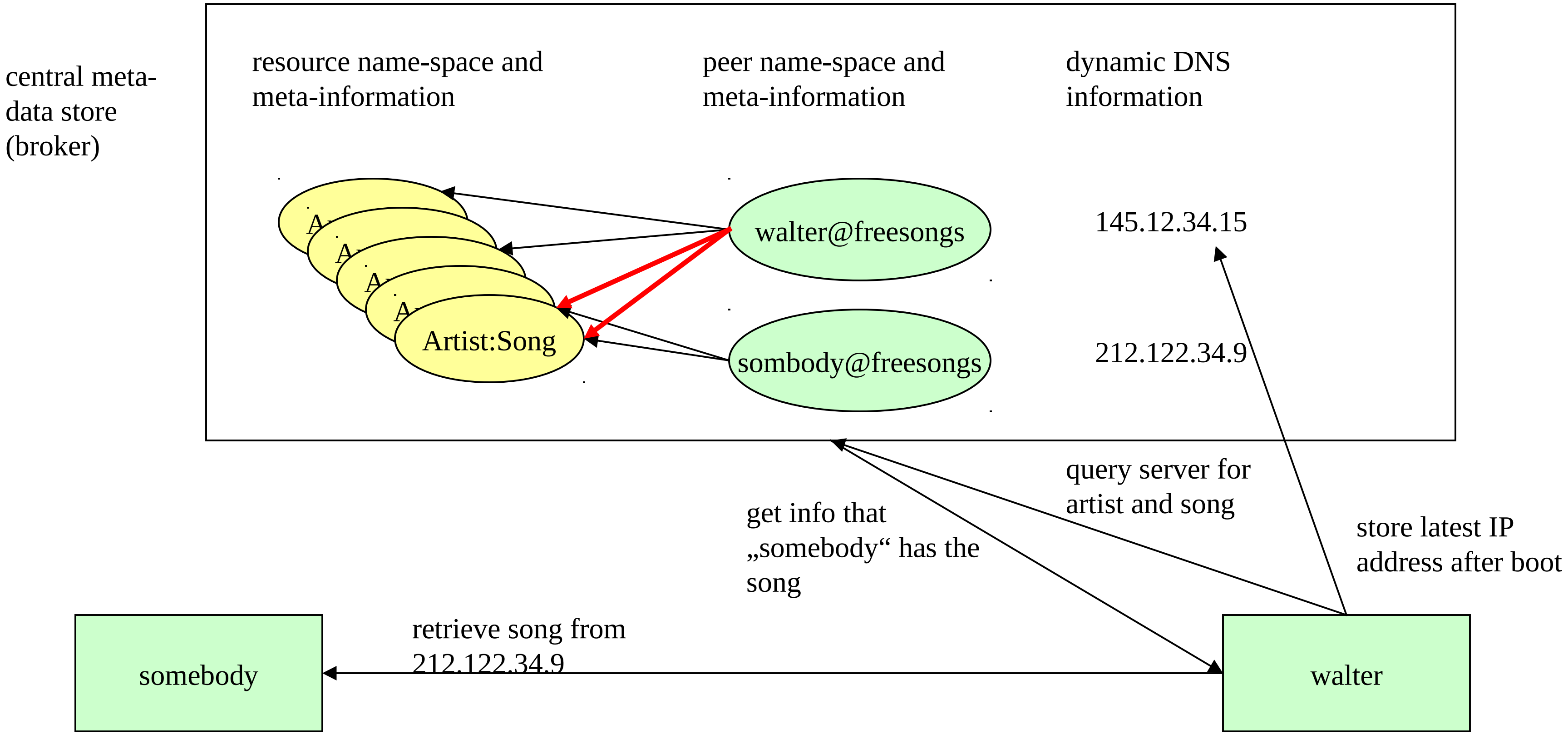
Client and Server

This architecture consists of identical peers only. It is very robust and hard to attack because it avoids special nodes. Its downside is scalability and effectivity (e.g. of searches). There is no broker peer that could store meta-data on requests or resources.

Vulnerability and distribution topology

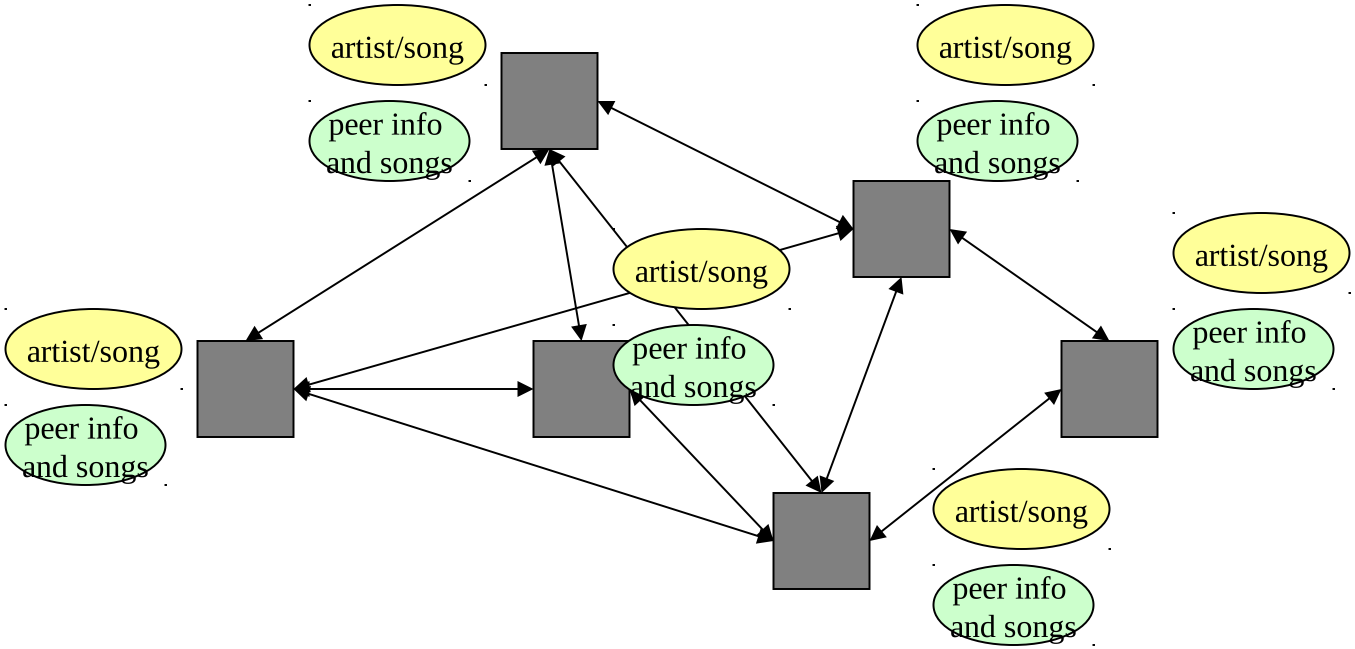
- Central servers: they concentrate all work at few machines. This makes them both dependent on the hardware of the cluster but also independent from all the other machines at the edge. But they are hurt by denial of service attacks.
- Broker/special peers in hierarchies: The p2p network can resist DOS attacks for a while but breaks down suddenly if a certain number of specialized peers is unavailable
- Totally distributed: DOS attacks do not threaten the whole p2p network.

Naming and Addressing: broker


central meta-data store (broker)
resource name-space and meta-information
peer name-space and meta-information
dynamic DNS information
A
A
A
A
Artist:Song
walter@freesongs
sombody@freesongs
145.12.34.15
212.122.34.9
query server for artist and song
get info that „somebody“ has the song
store latest IP address after boot
retrieve song from 212.122.34.9
somebody
walter

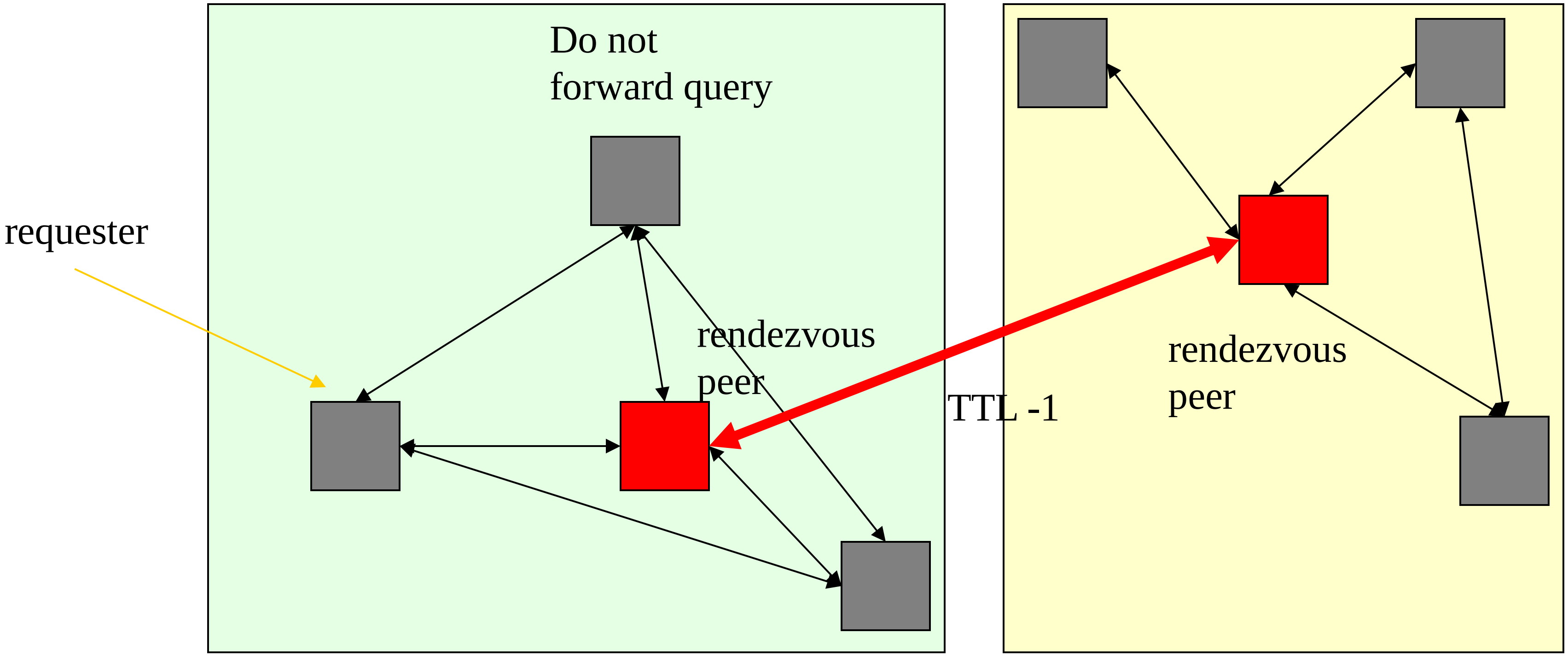
Several namespaces had to be created for this example: one for the artists and songs, another one for the peers. The peers get virtual identities that are only valid on this server. The server provides an interface where peers can register their latest (dynamically assigned) IP address. Walter queries the song server for a special artist and song and retrieves information where it can be found. After that walter retrieves the songs and the server updates its repository with the knowledge that walter now also has the requested songs.

Naming and Addressing: no central server



Without central repositories lookup of information tends to become very ineffective and sometimes resources will not be found because the time-to-live is not long enough. The www uses search engines as „in-the-net service“ to overcome this problem. An interesting question: would the web work without e.g. google and co.? Searching in totally distributed P2P systems like gnutella is an open research topic.

Scalability: Avoiding broadcast storms



Several techniques are combined to prevent request storms: Rules, special peers and grouping. Rules e.g. prevent simple peers from forwarding queries to other peers they know. Only special peers (in JXTA called „rendezvous“ peers forward queries, possibly also to other rendevous peers in different GROUPS. A group forms a scoping boarder e.g for queries, security or monitoring. But almost always a Time-To-Live (e.g. 7 hops) is used to let requests die after a number of hops. Gnutella and JXTA use 7.

URL, URI, URN

http://www.w3.org/index.html

An URL defines the ADDRESS of a resource. It is an error if the resource cannot be resolved at the given address

http://www.w3.org/1999/XMLSchema-instance

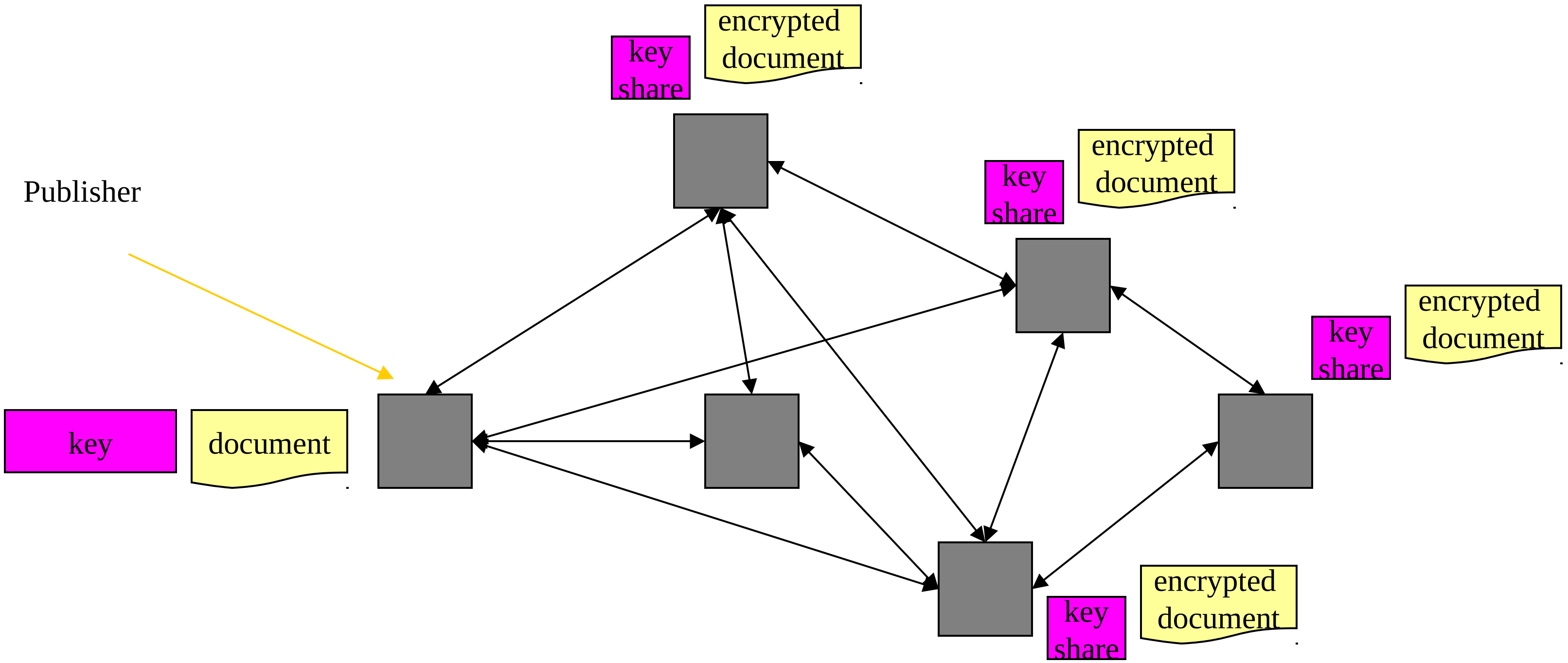
An URI can serve as a URL but it need not be electronically resolvable. XML Namespace definitions are such a case. There is NOTHING behind the namespace URI given.

urn:juxta:idform-something

An URN is ONLY a name which can be used to create unique identification spaces. The JXTA IDs are an example

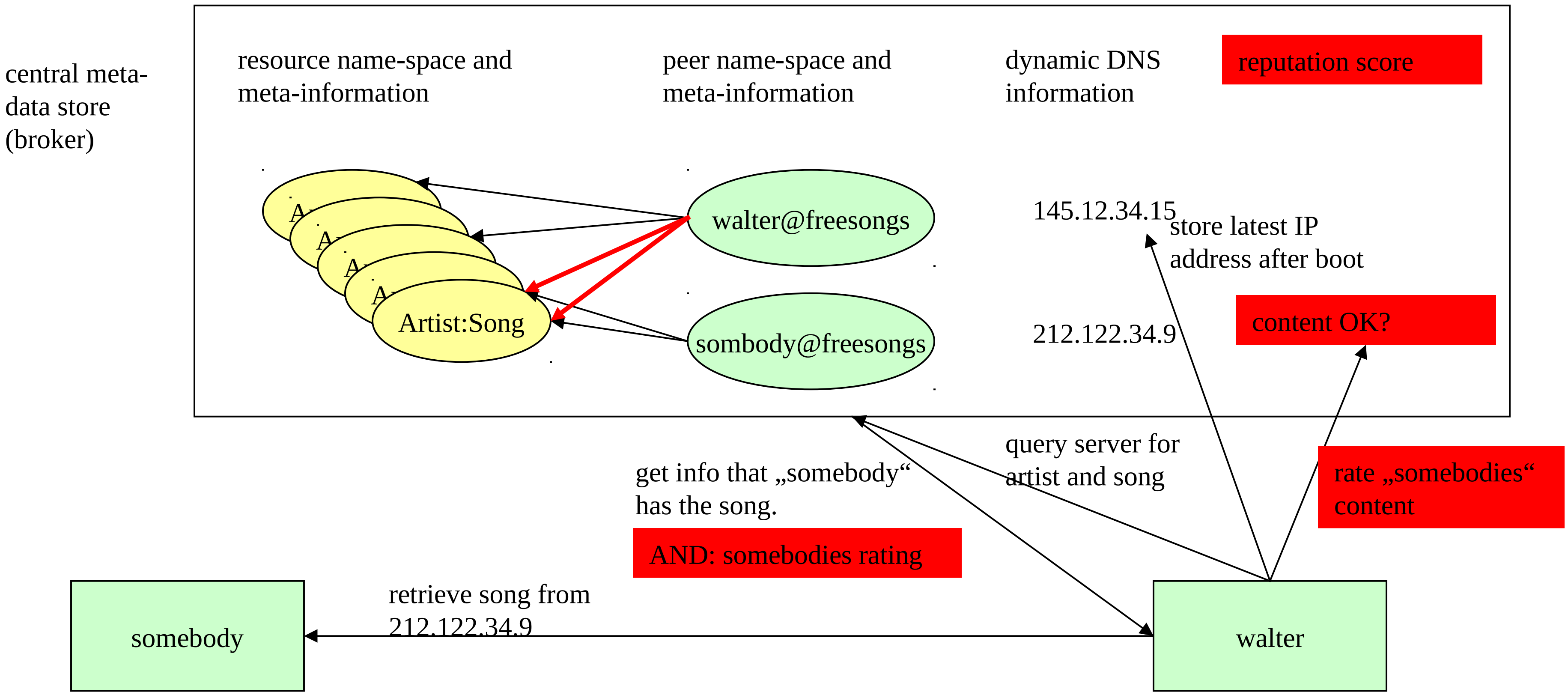
One of the P2P problems is that no central authority SHOULD exist which could hand out unique ID‘s. And DNS does not use permanent IDs (IP-addresses) at the edge of the internet.

Security (1): censorship resistant publishing



A publisher creates a key and uses it to encrypt the document. Then a key sharing algorithm is applied that creates n partial keys. A certain number of those keys are necessary to reconstruct the original key (and document). Then shares and keys are distributed on publishing servers. They can claim that they do not know the contents. Censorship is hard because documents are distributed over a nubmer of machines and a hostile person would need to destroy most of the key shares to make the document unreadable. Surprisingly updates and delete by the author still work. This is an example from the PUBLIUS system.

Security (2): Reputation System



central meta-data store (broker)

resource name-space and meta-information

peer name-space and meta-information

dynamic DNS information

reputation score

Artist:Song

walter@freesongs

sombody@freesongs

145.12.34.15

212.122.34.9

store latest IP address after boot

content OK?

query server for artist and song

get info that „somebody“ has the song.

AND: somebodies rating

rate „somebodies“ content

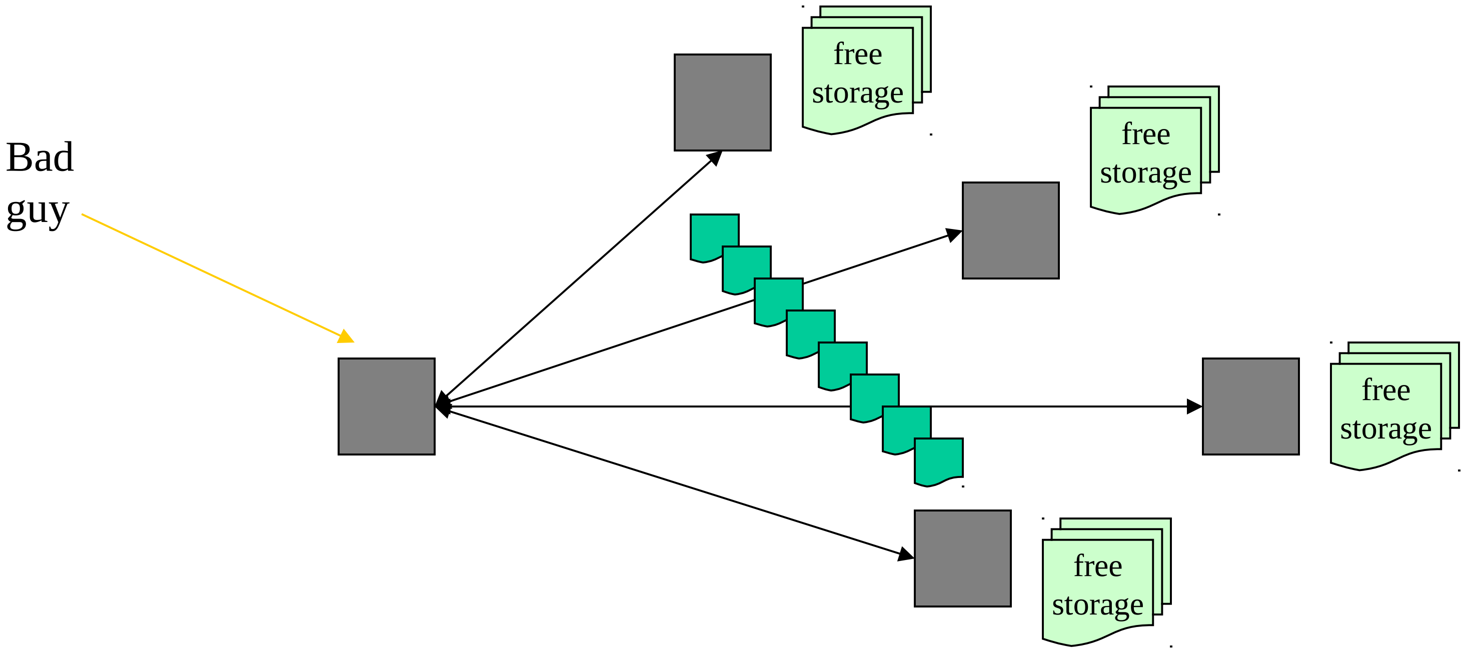
retrieve song from 212.122.34.9

somebody

walter

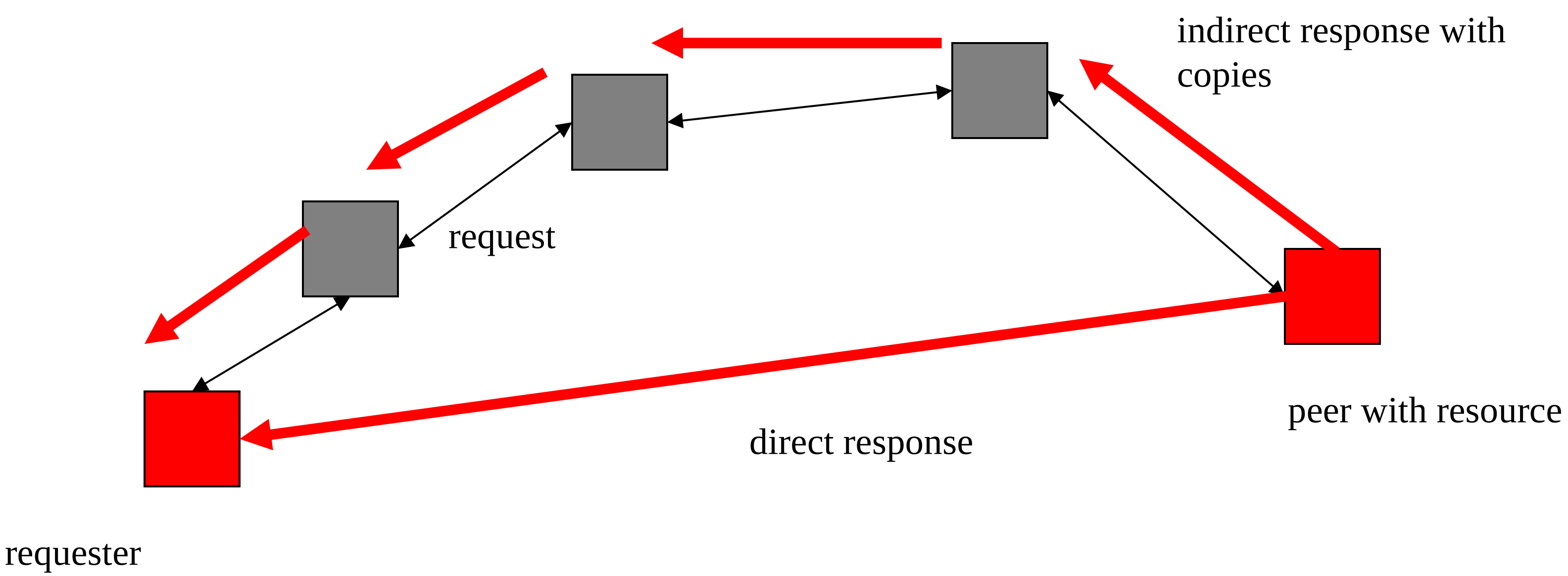
Using a central server a reputation system can be built rather easily. Peers rate others after each transaction. There are a number of problems with this approach: How do you prevent pseudo-spoofing (somebody creates lots of different pseudonyms). These pseudonyms can rate each other into a high reputation and then „cash-in“ (e-bay case). See: Accountability, chapter 17 in Harnessing the power...

Security (3): Micropayment



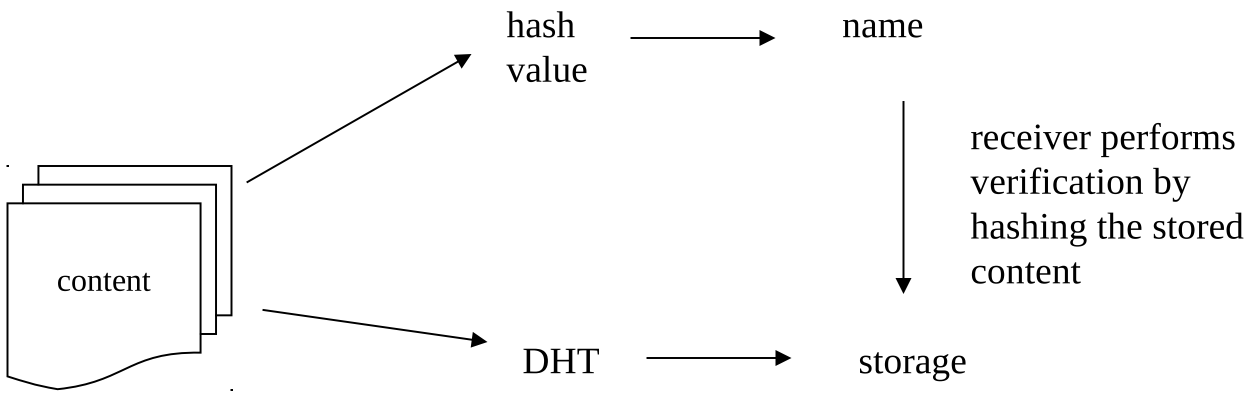
The bad guy tries to create a DOS attack by flooding the free storage peers with dummy documents. This could also be used to censor existing documents by flushing the caches of publishing servers (e.g. freenet caches frequently requested docs longer than others). One way to prevent this is to have the publisher solve a computationally intense piece of work for every entry. This will slow rogue publishing down. This mechanism is not much different from digital cash only that the work does not translate into re-usable cash.

Security (4): Routing and Anonymity



„Plausible Deniability“ is an important argument in legal disputes. A host that acts only as a transponder without really knowing the content can claim it to protect himself from legal actions. P2P networks can be designed to minimize the numbers of hosts that know each other directly. This is of course less efficient than e.g. having a source sending a response directly to the requester.

Security (5): Content Safety/Fakes


hash value
name
receiver performs verification by hashing the stored content
content
DHT
storage

Self-verifying content is achieved be using hash values for names. This allows easy validation on the receiving side. It does NOT provide a means to verify what a name really MEANS (denotes). So called fakes are planted to reduce the lookup success of p2p protocols and to waste bandwidth (time). A directory approach (collecting hashes and tagging them as „fake“) are not really a successful countermeasure as it also relies on unauthenticated notifications of fakes (the problem of „faked fakes“)

A related problem is versioning distributed content. Here an author needs to provide key material that proves authorization to make changes.

Security (6): Infrastructure Attacks

Routing Attacks: wrong lookup and node identification, false routing updates (poisioning routing tables), fake bootstrap host into a different (virtual) network, sybil attacks (one host posing as several different hosts)

Retrieval Attacks: hosts throwing away documents or incorrectly claiming responsibility for documents (needs regular checks)

DOS attacks on all replicas of a document thereby making it inaccessible

DHT balancing attack: rapidly joining and leaving a DHT can cause a large number of overhead requests caused by rebalancing the DHT.

See Emil Sit et.al, Security Considerations for Peer-to-Peer Distributed Hash Tables (resources) for a more detailed view on p2p infrastructure attacks.

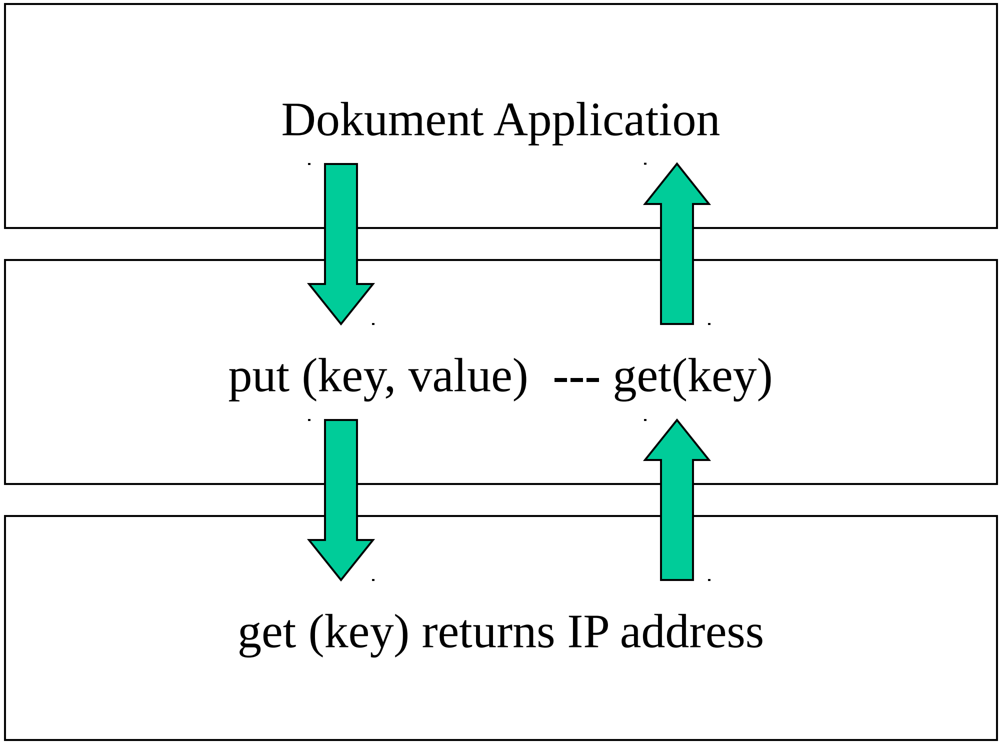
Social Problems in P2P Systems

- Free-riding is the use of system resources without letting the system use one‘s own resources (e.g disk space or songs)

- The tragedy of the commons is the fact that common goods that are not owned by somebody are usually destroyed by everybody using them.

In both cases the answer is the introduction of some kind of „payment“ or „credit“ into the system. This can be a computational „payment“ that leads to a certain delay in up-loads or virtual credit that is increased by offering services to others.

Distributed Hash Tables (DHT)



location independent storage layer

ID – Host mapping layer

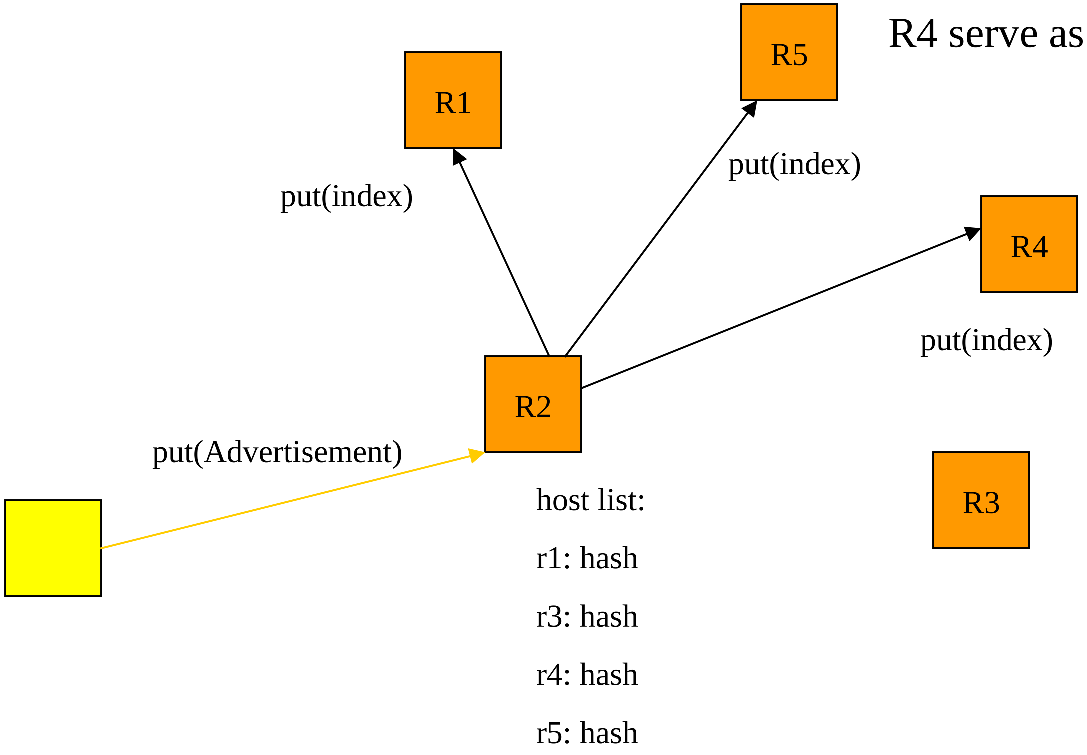
For an overview of different DHT approaches compare CAN, CHORD and e.g KADEMLIA. Look at how the routing algorithms deal with high rates of peers leaving/entering the network. The advantage of a DHT lies in its simple interface and location independence

DHT Problems

1. Content Integrity: this is mostly solved through hashing the content (self-verification).
2. Host lookup. Several algorithms are possible, from totally distributed (gnutella) to registry approaches (napster, edonkey) or hybrid models (JXTA)
3. Maintenance: A high churn rate invalidates many indices and can force a large number of maintenance messages to be exchanged which decreases lookup efficiency

Host lookup can be optimized by applying a distance function on the key – calculating how „far“ a certain host ID is away in virtual host space. In its most simple case this could be just subtracting the content hash from the host ID hash and chosing the host with the smallest difference. Other systems (kademlia e.g.) try to organise host ID hashes in trees to improve lookup speed

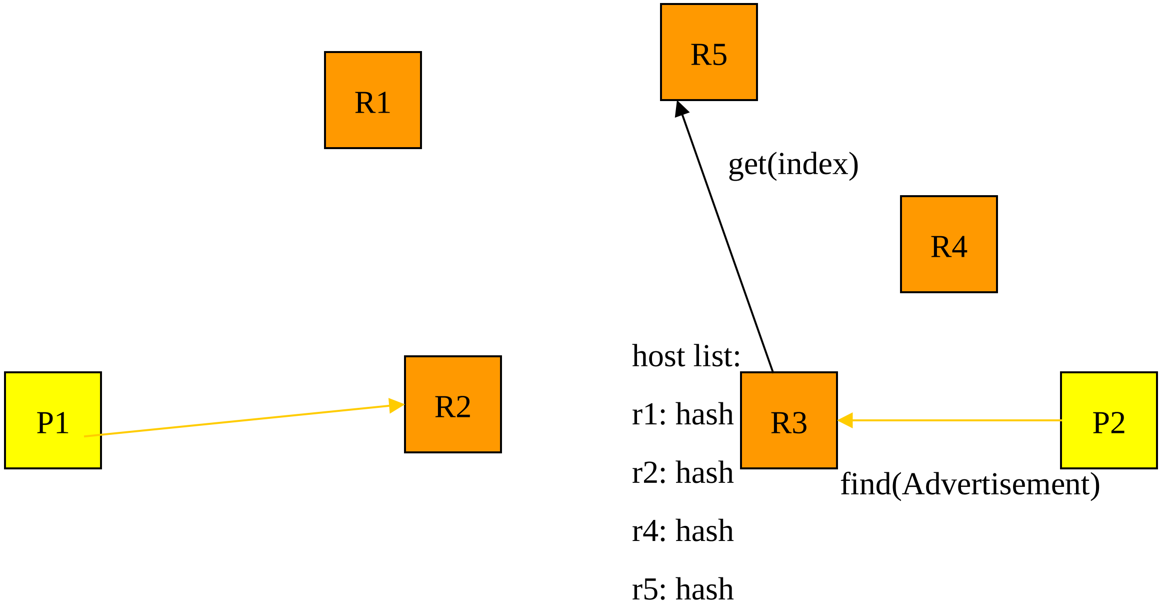
A loosely consistent tree-walker (Store)



R5 is used to store the index of the advertisement, R1 and R4 serve as backups

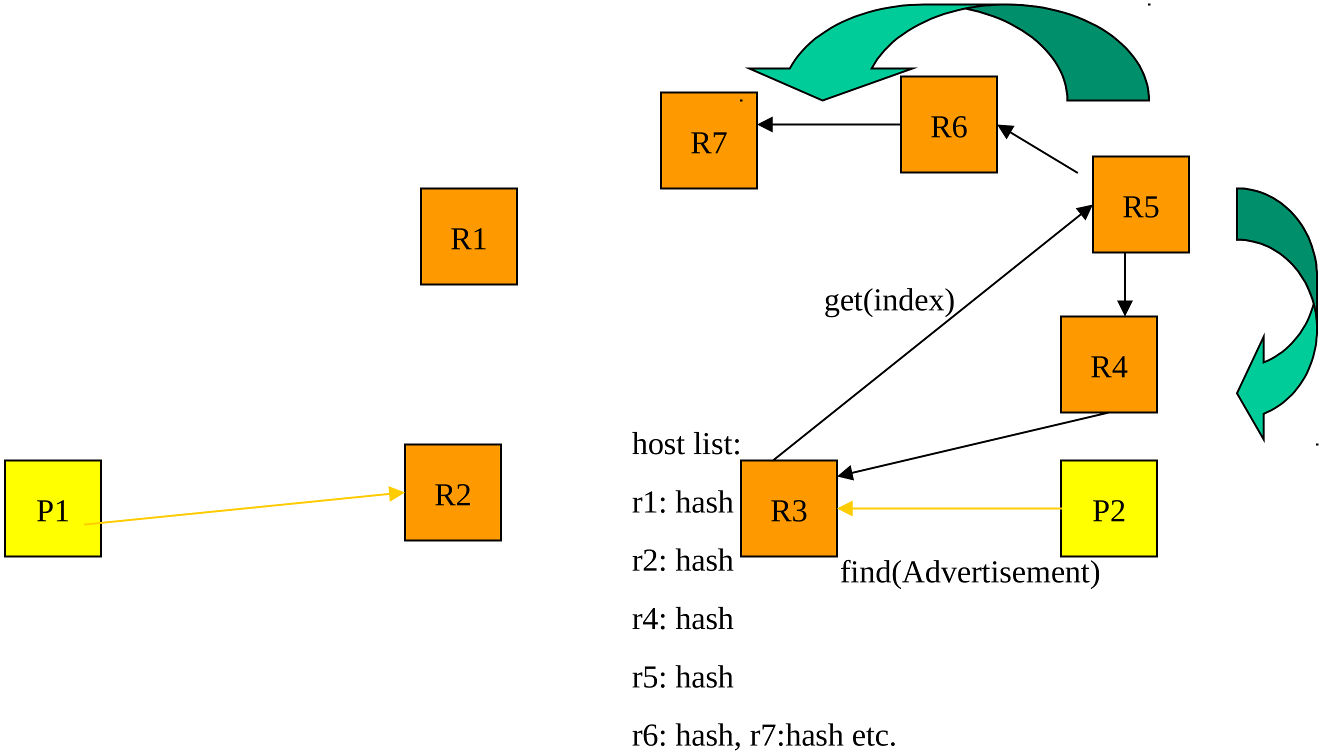
The Rendevous peer R2 calculates the hash of the advertisement, applies the distance function and finds R5 as best storage location for the indexed advertisement. It also stores the content at „nearby“ hosts (hosts which are close to R5 in R2‘s routing table. On a random base the rendevous peers exchange routing tables and detect dead hosts. (See: „a loosely consistent DHT Rendevous Walker, B. Traversat et.al.)

A loosely consistent tree-walker (Lookup)



When a peer does a lookup the rendevous peer calculates the distance function and picks the proper host where the content was supposedly stored. If the DHT configuration has not changed the first lookup will already succeed. If e.g. R5 is down either R1 or R4 would be picked as backup

A loosely consistent tree-walker (Walking)



In case of a high churn rate the routing tables have changed a lot. In case a query fails at one host the host will start a tree-walk in both directions (up and down the ID space) and search for the requested content. This allows content lookup even if the rendezvous peer structure changed beyond our initial backup copies.

Research Issues in DHT‘s

Lookup optimization without maintenance (crawling/walking has costs but those are NOT maintenance)

Delivery guarantees (storage reliability)

Censorship (avoid registry problems)

Routing optimizations for mobile systems (minimize overhead messages)

Consistent hashing to allow for scalability (e.g. Dynamo/Amazon)

Constant lookup times

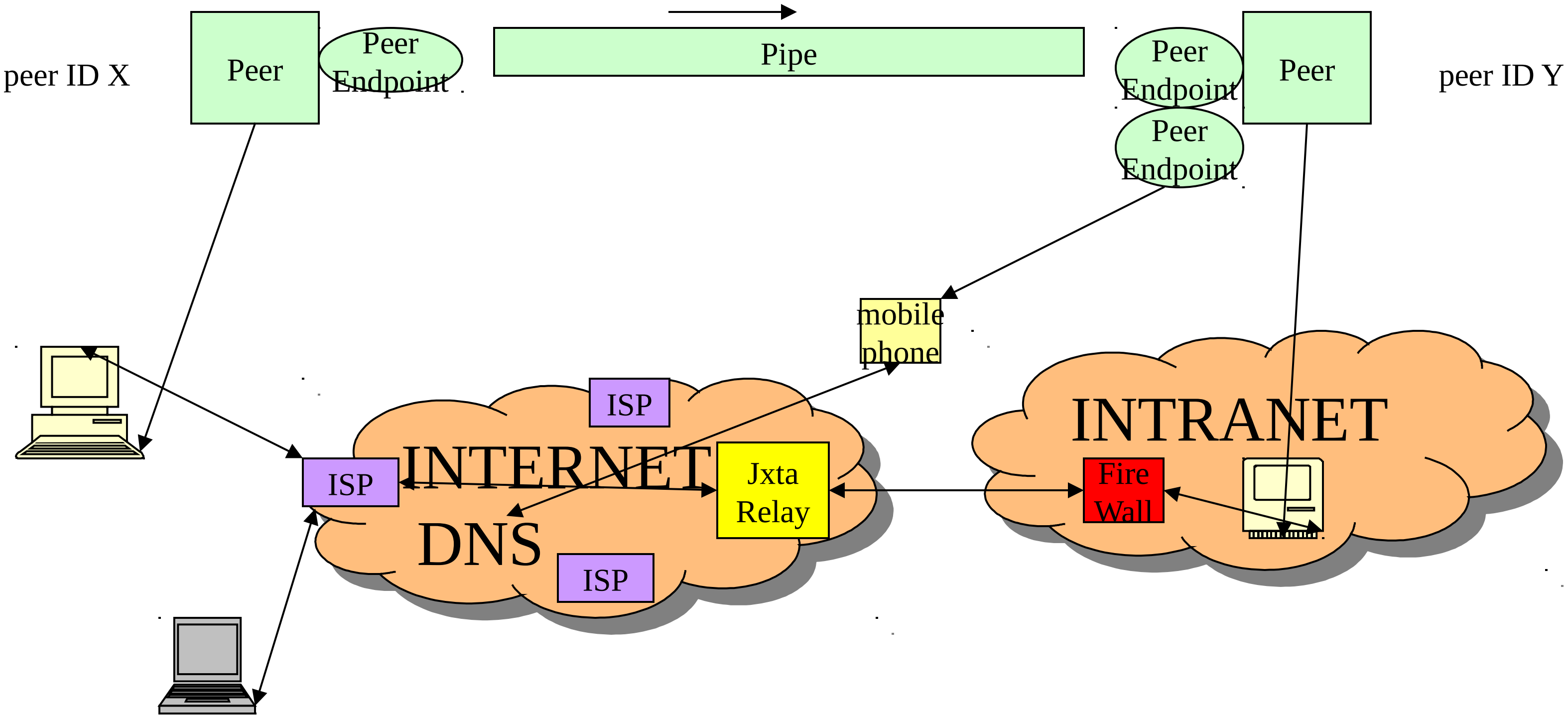
See Gurmeet Singh Manku, Routing Networks for DHT‘s

Why a framework for P2P systems?

- Rationale behind JXTA (from „Juxtaposition“)
- Architecture
- Concepts and Abstractions

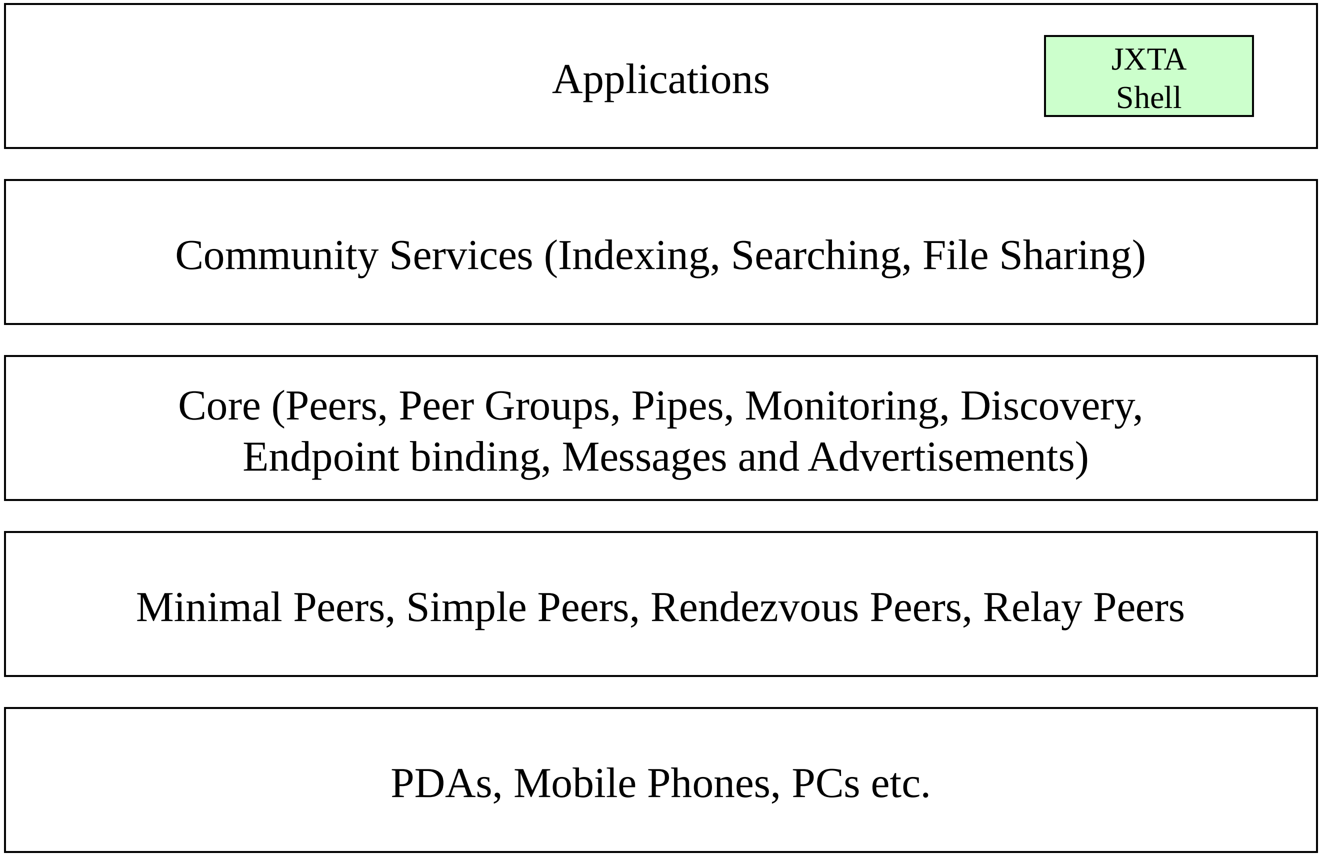
Currently p2p suffers from two deficits: There seems to be a new application for every kind of purpose or resource: the sharing of mp3 files, collaboration spaces, other file-sharing systems etc. They all use different protocols and meta-data and cannot re-use each others services. The other deficit is that every p2p application needs to solve the same low level problems of network structure, peer identity etc. This could easily performed by a common framework

Abstracting away the physical differences



Nodes on the edge use all kinds of identities, naming and addressing modes. They are disconnected frequently. They are behind firewalls with NAT. JXTA puts an abstraction layer above the physical infrastructure that allows programmers to program without worrying about the physical differences in latency etc.

JXTA Architecture



Applications

JXTA Shell

Community Services (Indexing, Searching, File Sharing)

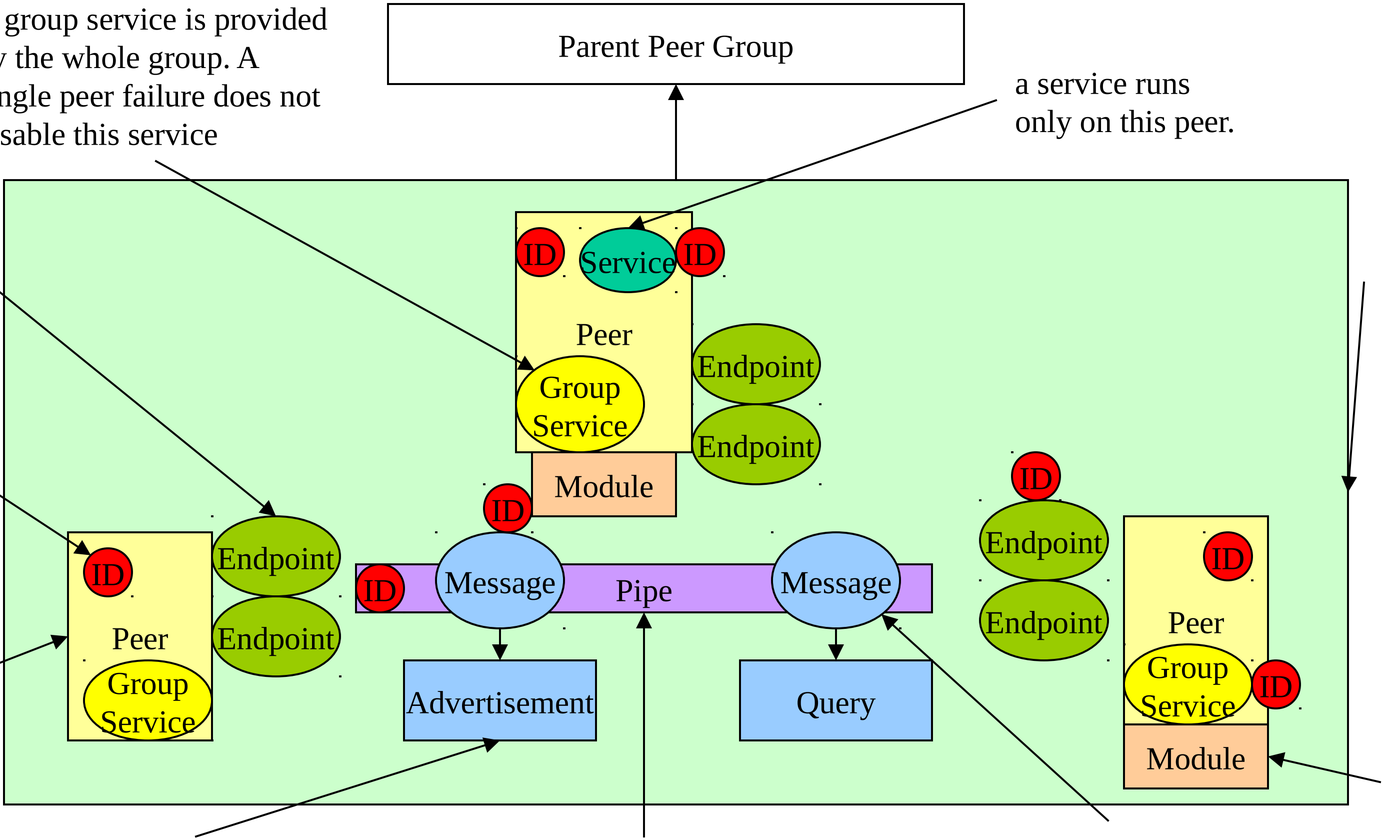
Core (Peers, Peer Groups, Pipes, Monitoring, Discovery, Endpoint binding, Messages and Advertisements)

Minimal Peers, Simple Peers, Rendezvous Peers, Relay Peers

PDAs, Mobile Phones, PCs etc.

By providing core services and interfaces, JXTA allows the building of interoperable applications and relieves the applications from implementing the same core functionality again and again.

JXTA Abstractions and Concepts



A group service is provided by the whole group. A single peer failure does not disable this service

a service runs only on this peer.

A peer-group provides a scoping boarder for queries, security and monitoring. It has a parent peer group where it is advertised

endpoints bind peers to transports

Most everything has an ID

Peers are networked devices

A module is a piece of pluggable behavior.

Advertisements are meta-data about resources. They are described (as everything in JXTA) using XML

A pipe is a virtual communication channel: uni/bi-directional, secured etc. Some can propagate to more peers

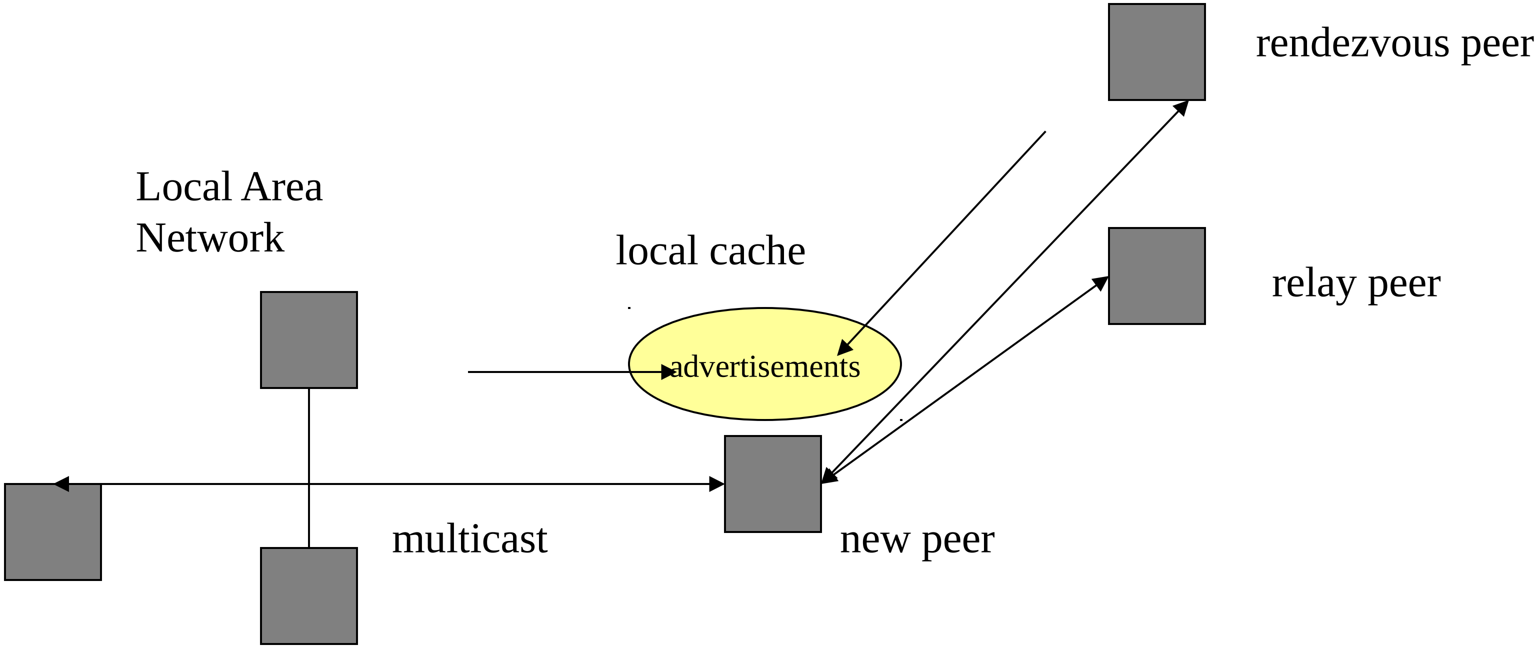
A message is the basic request/reply format. It is an XML structure of name/value pairs

JXTA Protocols

- Peer Discovery Protocol: How to advertise and find resources
- Peer Information Protocol: How to get status information about other peers (connectivity, uptime etc.)
- Peer Resolver Protocol: generic query mechanism allows exchange of arbitrary defined queries.
- Pipe Binding Protocol: to establish a virtual communication channel between peers. The peers need not be able to talk directly to each other (Relay peers are used)
- Endpoint Routing Protocol: to find routes to destination ports on other peers. Multi-Hop is possible.
- Rendezvous Protocol: Special peers serve as message propagators within peer groups.

See: JXTA programming guide pg. 19 ff. for a detailed description of the core protocols. JXTA defines a message format for each of those so that applications can interoperate more easily.

Bootstrapping a JXTA peer



If a rendezvous or relay peer are pre-configured at the new peer, those will be contacted through the discovery service. Else a multicast on the lcoal are network is performed by the discovery service. The local cache of the new peer will be filled with advertisements about peers or peer groups. All messages are performed ASYNCHRONOUSLY! TTL at the creating peer is 1 year, otherwise 2 hours. The new peer itself piggybacks its advertisement on the discovery.

Asynchronous Discovery and Use

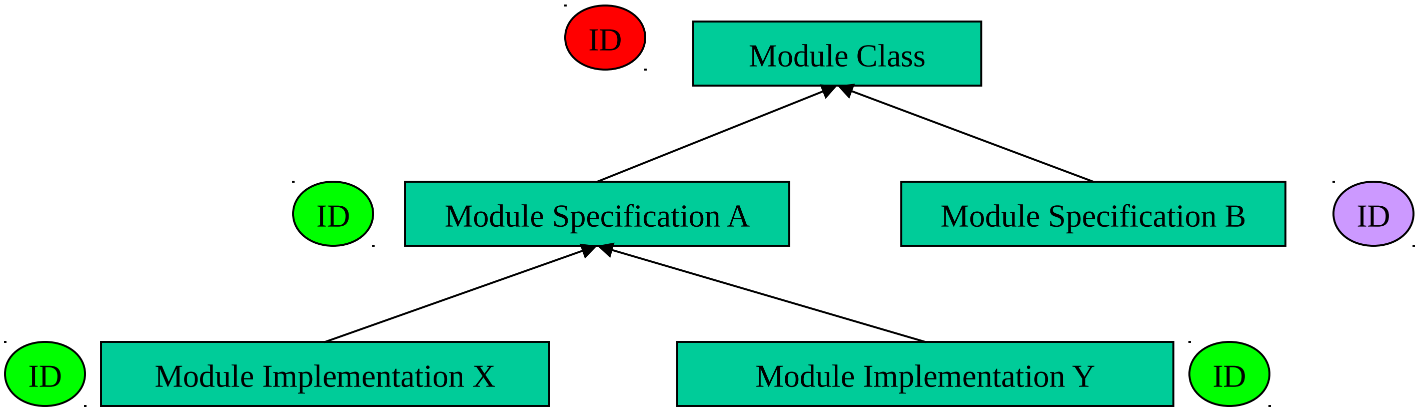
1. Create advertisement for pipe and publish it
2. Create resource, e.g. pipe
3. Register listener for pipe events.

---

4. Handle asynchrounous callbacks in listener

This is basically the same pattern used as in Java Beans or Swing.

How to build a flexible P2P framework



e.g. NullMembershipService

e.g. PasswordMembershipService

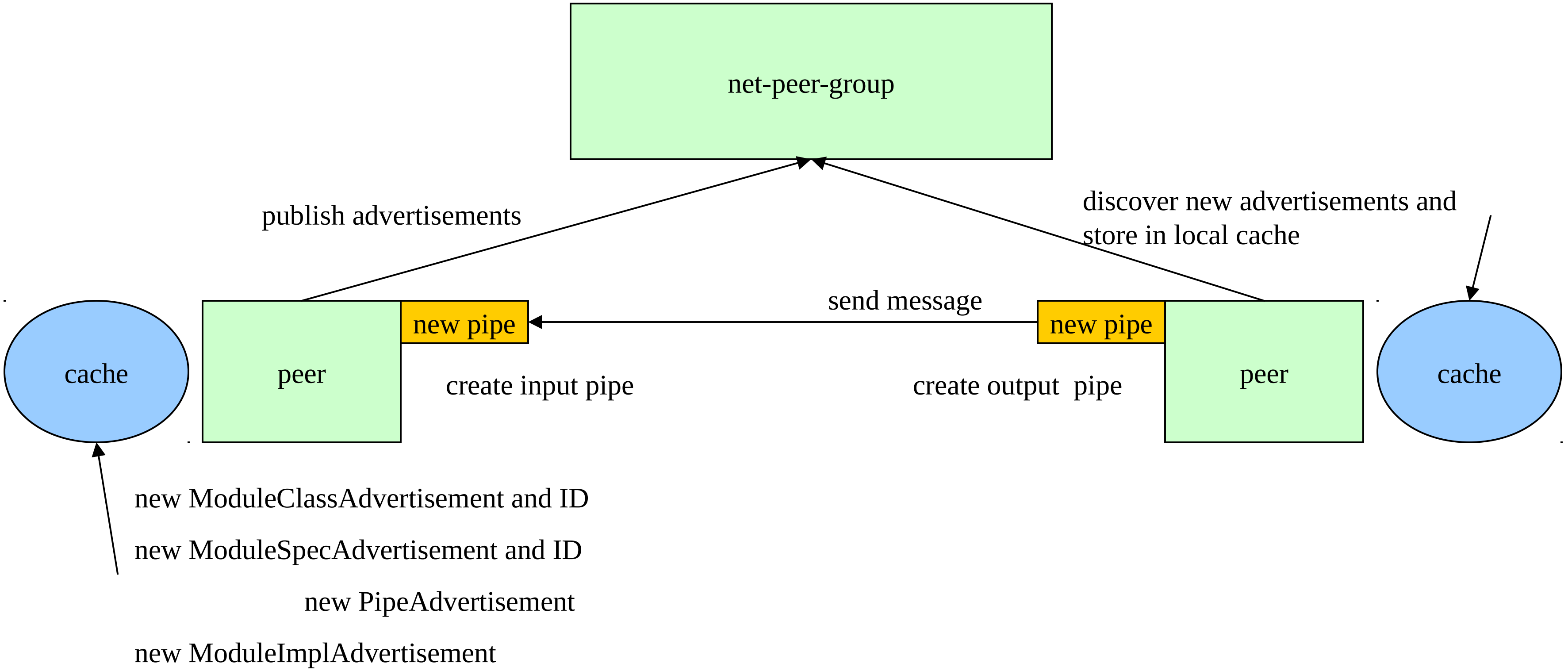
Define Interfaces for everything. This allows different implementations and strategies for the core concepts. Use Factories for object creation.

Define core services that belong to the platform and define a WIRE FORMAT for all core messages. This is a precondition for interoperability.

Define GENERIC types for messages, queries etc. to allow extensions

Define a plug-in or module concept that allows creation and dynamic loading of new behavior

Creating a new service in JXTA



A module specification defines a service and a pipe where the service can be reached. Other peers discover the published advertisements, find the pipe definition (ID), create the proper output pipe with same ID and can then send messages to the service. Also existing services can be replaced using this pattern. Of course, the peers need to understand the messages used by the new service.

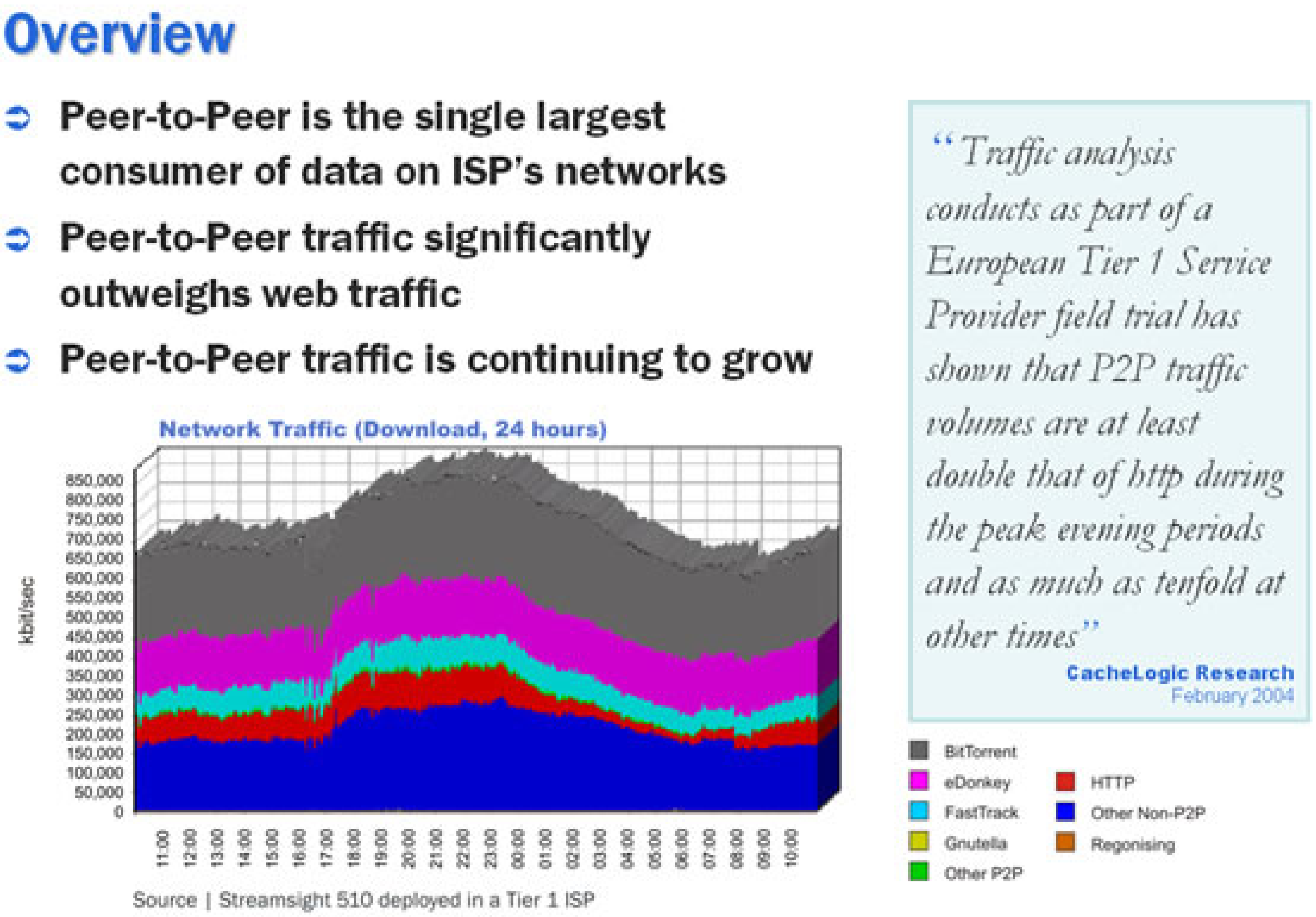
JXTA Security

All Implementations of the core services need to provide:

- Confidentiality
- Authentication
- Authorization
- Data Integrity
- Non-Repudiation

Other P2P security requirements e.g. to assure anonymity of publishers, hosts etc. and to avoid censorship are not mentioned at all. They will have to be implemented on top of the basic services.

Bittorrent: a successful p2p file-sharing system



Overview

- Peer-to-Peer is the single largest consumer of data on ISP's networks
- Peer-to-Peer traffic significantly outweighs web traffic
- Peer-to-Peer traffic is continuing to grow

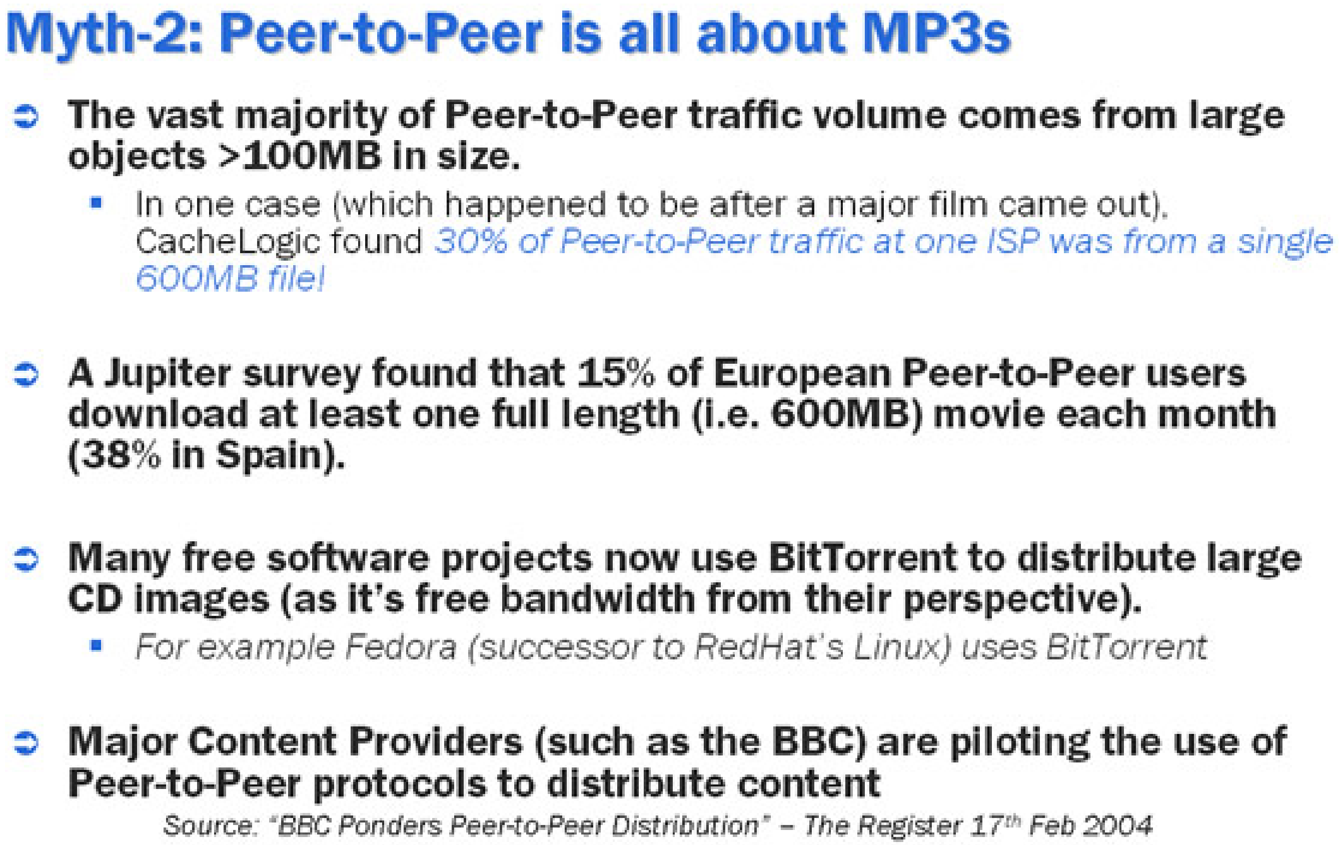
Network Traffic (Download, 24 hours)

Source | Streamsight 510 deployed in a Tier 1 ISP

> *"Traffic analysis conducts as part of a European Tier 1 Service Provider field trial has shown that P2P traffic volumes are at least double that of http during the peak evening periods and as much as tenfold at other times"*
>
> **CacheLogic Research**
> February 2004

from: www.cachelogic.com which provides very interesting empirical research on usage of p2p networks.

Bittorrent Usage Patterns: Disruptive



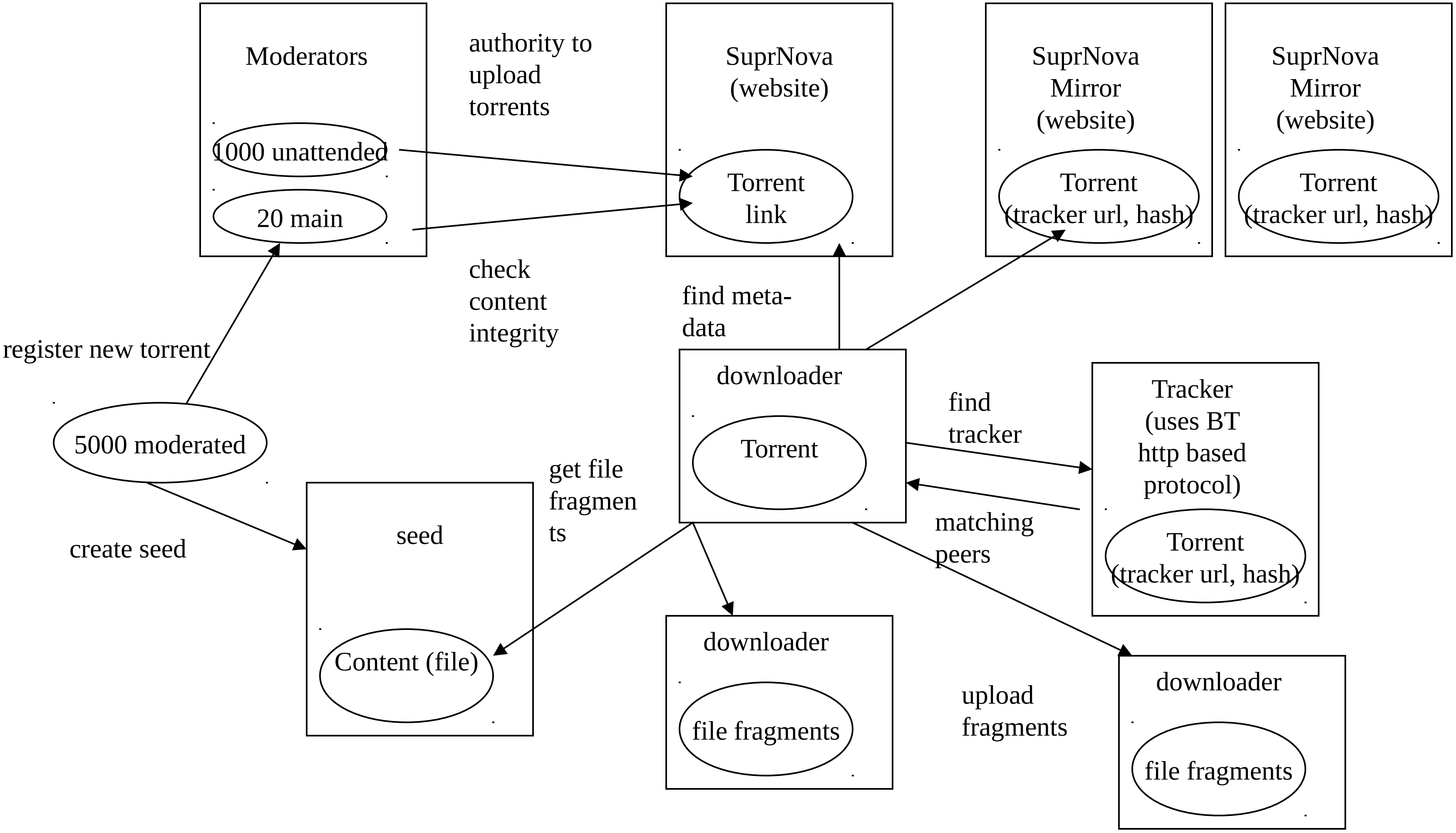
Myth-2: Peer-to-Peer is all about MP3s

- **The vast majority of Peer-to-Peer traffic volume comes from large objects >100MB in size.**
  - In one case (which happened to be after a major film came out), CacheLogic found *30% of Peer-to-Peer traffic at one ISP was from a single 600MB file!*
- **A Jupiter survey found that 15% of European Peer-to-Peer users download at least one full length (i.e. 600MB) movie each month (38% in Spain).**
- **Many free software projects now use BitTorrent to distribute large CD images (as it's free bandwidth from their perspective).**
  - *For example Fedora (successor to RedHat's Linux) uses BitTorrent*
- **Major Content Providers (such as the BBC) are piloting the use of Peer-to-Peer protocols to distribute content**

Source: "BBC Ponders Peer-to-Peer Distribution" – The Register 17th Feb 2004

from: www.cachelogic.com. Note the rising „serious“ use of bittorrent by software and media companies.

Bittorrent Architecture



Bittorrent relies on web services for finding torrents. It is a pure download network. Mirrors do load-balancing. Trackers match peers and seeders provide initial file upload.

Bittorrent Analysis

1. Availability: BT relies on centralized components which are hard to distribute and create SPOFs.
2. Integrity: At the same time they guarantee high content quality. The low number of moderators is a surprise. Trackers seem to suffer huge bandwidth costs.
3. Flashcrowds: Even content that is in high demand can be downloaded quickly. Seeders suffer high bandwidth costs for an extended period of time (design failure?)
4. Performance: around 250 kbit/s. Theoretical limit: overall upload bandwidth. Practical limit: much higher as not all BT users download permanently.

Taken from: Johan Pouwelse, The Bittorrent P2P file-sharing system. (see resources). The study provides empirical data on BT usage and outages and comes to some surprising results. See also peer-2-peer.org for diagrams. Improvements to the protocal would have to solve the problem of introducing further distribution (e.g. trackers) while preserving content integrity.

The Future

1. Wireless Mesh Networking. A p2p application that uses ad-hoc, wireless networking to create a networking infrastructure without dedicated servers and without the high investments and monopolies that usually accompany centralized approaches. Interesting for its technical (routing) and social (3rd world) aspects. See Tomas Krag and Sebastian Buettrich, Wireless Mesh Networking, presented at the O‘Reilly Emerging Conferences Show.2003
2. Mobile Device Integration: Use JXTA to tie J2ME clients into a general JMX enterprise infrastructure. See Faheem Khan, Wireless Messaging with JXTA part 1 and 2 (developerworks)

P2P computing puts the emphasis back on individual machines, compared to the server centric WWW world. This fits nicely with upcoming topics like mobile computing and autonomous systems.

What‘s next in our distributed systems series?

We have skipped some very interesting and important parts of distributed systems technology in this lecture: group communication, consensus and election algorithms, time in distributed systems etc.

But if we want to implement distributed systems we will need this know-how: I am therefore planning a seminar on advanced distributed algorithms where we will tackle all the above problems. And in addition to that we will develop a concept of failures, failure modes and how we can improve the reliability and stability of distributed systems – be they centralized or of peer-to-peer nature.

Resources (1)

- Peer-to-Peer, Harnessing the Power of Disruptive Technologies, Edited by Andy Oram, 2001, O‘Reilly. Contains good articles on different p2p applications (freenet, Mixmaster Remailers, Gnutella, Publius, Free Haven etc). And also from Clay Shirkey: Listening to Napster. Recommended.
- Peer-to-Peer, Building Secure, Scalable and Manageable Networks, Dana Moore and John Hebeler. Definitely lighter stuff then Andy Oram‘s collection. Missing depth. Covers a lot of p2p applications but few base technology.
- www.openp2p.org , the portal to p2p technology. You can find excellent articles e.g. by Nelson Minar on Distributed Systems Topologies there.
- www.jxta.org, home of the jxta framework from Sun.
- JXTA v1.0 Protocols Specification. Covers abstractions and protocols used in jxta.

Resources (2)

- Project JXTA: Java Programmer‘s Guide. First 20 pages are also a good technical overview on p2p issues.
- Upcoming: 2001 P2P Networking Overview, The emergent p2p platform of presence, identity and edge resources. Clay Shirkey et.al. I‘ve only read the preview chapter but Shirkey is definitely worth reading.
- It‘s not what you know, it‘s who you know: work in the information age, B.A.Nardi et.al., http://www.firstmonday.org/issues/issue5_5/nardi/index.html
- Freeriding on gnutella, E.Adar et.al., http://www.firstmonday.org/issues/issue5_10/adar/index.html, claims that over 70% of all gnutella users do not share at all and that most shared resources come from only 1% of peers.
- Why gnutella can‘t possibly scale, no really, by Jordan Ritter. http://www.monkey.org/~dugsong/mirror/gnutella.html. An empirical study on scalability in gnutelly.

Resources (3)

- A Modest Proposal: Gnutella and the Tragedy of the Commons, Ian Kaplan. Good article on several p2p topics, including the problem of the common goods (abuse) http://www.bearcave.com/misl/misl_tech/gnutella.html
- Clay Shirky, File-sharing goes social. Bad news for the RIAA because Shirky shows that prosecution will only result in cryptographically secured darknets. There are many more people then songs which makes sure that you will mostly get the songs you want in your darknet. Also: do your friends share your music taste? quite likely. http://www.shirky.com/writings/file-sharing_social.html Don‘t forget to subscribe to his newsletter – you won‘t find better stuff on networks, social things and the latest in p2p.
- Bram Cohen, Incentives Build Robustness in Bit Torrent. Explains why the bit torrent protocol is what it is. Bit torrent tries to achieve „pareto efficiency“ between partners. Again a beautiful example how social and economic ideas mix with technical possibilites in p2p protocol design: why is it good to download the rarest fragments first? etc.

Resources (4) DHT designs

- Bob Loblaw et.al, Building Content-Based Publish/Subscribe Systems with Distributed Hash Tables. Nice paper on DHT design with a content based focus (not topic based as usually done). Experimental, good resource section.
- M.Frans Kaashoek, Distributed Hash Tables: simplifying building robust Internet-scale applications (http://www.project-iris.net) . Very good slide-set on DHT design. You need to understand DHT if you want to understand p2p.
- Ion Stoica (CD 268), Peer-to-Peer Networks and Distributed Hash Tables. Another very detailed and good slide set on DHT designs. (CAN/Choord/freenet/gnutella etc.). Very good.

Resources (5) Security in P2P

- Emit Sit, Robert Morris, Security Considerations for Peer-to-Peer Distributed Hash Tables. A must read. Goes through all possible attack scenarios against p2p systems. Good classification of attacks (routing, storage, general). Suggests using verifyable system invariants to ensure security.
- Moni Naor, Udi Wieder, A simple fault tolerant Distributed Hash Table. Several models of faulty node behavior are investigated.
- Distributed Hash Tables: Architecture and Implementation. A usenix paper which discusses transactional capabilities of a DHT based DDS.
- www.emule-project.net/faq/ports.htm shows the ports in use by emule-related protocols. Shows that several emule-users behind a NAT/router/firewall need individual redirects established at the firewall to allow incoming connections to be redirected to a specific client.

Resources (6)

- OCB Maurice, Some thoughts about the edonkey network. the author explains how lookup is done in edonkey nets and what hurts the network. Interesting details on message formats and sizes.
- John R. Douceur et.al (Microsoft Research), A secure Directory Service based on Exclusive Encryption. One of many articles from Microsoft research which try to use P2p technologies as a substitute for the typical server infrastructure in companies.
- John Douceur, The Sybil Attack, Can you detect that somebody is using multiple identities in a p2p network. John claims you can’t without a logicall central authority.
- Atul Adya et.al (Micr.Res.), Farsite: Federated, Available and Reliable Storage for an Incompletely Trusted Environment. very good article with security etc. in a distributed p2p storage system. How to enable caching of encrypted content etc.
- W.J. Bolosky et.al, Feasibility of a Serverless Distributed Filesystem deployed on an Existing Set of PCs. Belongs to the topics above. Interesting crypto tech (convergent encryption) which allows detection of identical but encrypted files.

Resources (7)

- Ashwin R.Bharambe et.al, Mercury: A scalable Publish-Subscribe System for Internet Games. Very interesting approach but does not scale yet. Good resource list at end.
- Matthew Harren et.al, Complex Queries in DHT-based Peer-to-Peer Networks. How do you create a complex query if hashing means “exact match”? E.g. by splitting the meta-data in many separate hash values. Interesting ideas for search in p2p.
- Josh Cates, Robust and Efficient Data management fo a Distributed hash table, MIT master thesis.
- Peter Druschel at.al, PAST: a large-scale, persistent peer-to-peer storage utility.Excellent discussion of system design issues in p2p.
- Bernard Traversat et.al, Project JXTA: A loosely-consistent DHT Rendezvous walker. Read this to get the idea of DHT in an unreliable environment. Very good.
- John Noll, Walt Scacchi, Repository Support for the Virtual Software Enterprise. Use of DHT for software engineering support in distributed teams/projects.

Resources (8)

- Petar Maymounkov et.al. Kademlia: A peer-to-peer Information System based on the XOR metric. http://kademlia.scs.cs.nyu.edu/ An improvement on DHT technology through better organization of the node space. Interestingly, edonkey nets want to use it in the future.
- Zhiyong Xu et.al. HIERAS: A DHT based hierarchical P2P routing algorithm. Shows that one can win through a layered routing approach which e.g. allows optimization through proximity.
- Todd Sundsted, The practice of peer-to-peer computing. A series of entry level articles from www.ibm.com/developerworks (e.g. trust and security in p2p)
- http://konspire.sourceforge.net A comparison with bittorrent technology. Interesting. What limits the download in a p2p filesharing app? Also get the overview paper on konspire from that site.
- NS2 – the network simulator. A discrete event simulator targeted at network research. Use it to simulate your p2p networks. (from http://www.isi.edu/nsnam
- Zhiyong Xu et.al, Reducing Maintenance Overhead in DHT based peer-to-peer algorithms.

Resources (9)

- Bernard Traversat et.al., Project JXTA 2.0 Super-Peer Virtual network. Describes the changes to JXTA 2.0 which introduced “super-peers” for performance reasons – though they are dynamic and every peer can become one. Good overview on JXTA.
- Ken Birman et.al, Kelips: Building an Efficient and Stable P2P DHT Through increased Memory and Background Overhead. I read it simply because of Birman. Shows the cost if one wants to make p2p predictable.
- Krishna Gummadi et.al, The impact of DHT Routing Geometry on Resilience and Proximity. Compares several DHT designs. Quite good. Findings are that neighbour flexibility is more important than route selection flexibility. Proximity selection techniques perform well.
- Mark Spencer, Distributed Universal Number Discovery (DUNDi) and the General Peering Agreement, www.dundi.com/dundi.pdf
- http://www.theregister.com/2004/12/18/bittorrent_measurements_analysis/print.html An analysis of the bittorrent sharing system.

Resources (10)

- Ian G.Gosling, eDonkey/ed2k: Study of a young file sharing protocol. Covers security aspects.
- Heckmann, Schmitt, Steinmetz, Peer-to-Peer Tauschbörsen, eine Protokollübersicht. www.kom.e-technik.tu-darmstadt.de
- www.selfman.org Portal for EU sponsored research on media distribution (peerTV), transactional key/value stores (scalaris) and self-management.
- Security Issues and Solutions in Peer-to-peer Systems for Realtime Communications draft-schulzrinne-p2prg-rtc-security-00 (Internet RFC proposal) Feb. 2009
- Bittorrent tracker: http://chaosradio.ccc.de/cre057.html
- Vortrag
  http://events.ccc.de/congress/2007/Fahrplan/events/2355.en.html
  http://chaosradio.ccc.de/24c3_m4v_2355.html