Overview of P2P SIP Principles & Technologies

Dan Pascu

Future of VoIP II

The Hague, 15 March 2007

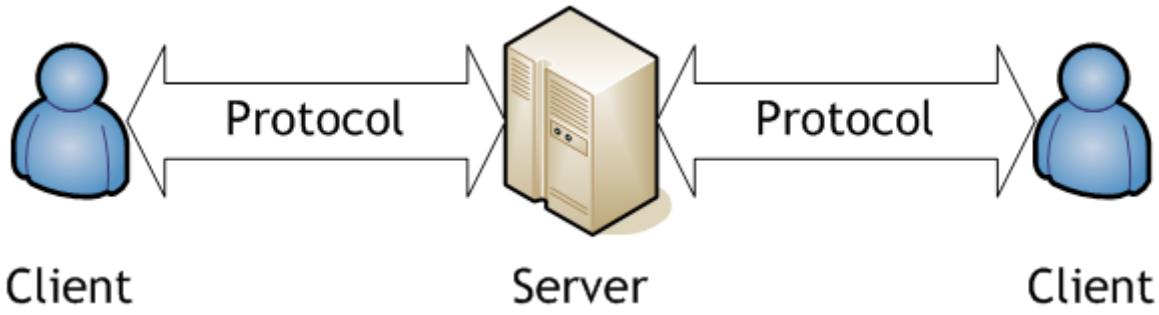
Understanding P2P

- How it came to life
- What it tries to solve
- What it is
- How it works

Life before P2P

- Client-Server (CS) based
- Centralized architectures
- Simple model

Client-Server model



- Client and server communicate using a well defined protocol
- Server stores all the service logic (centralized architecture)
- Clients (dumb) ask the server to perform tasks for them
- Clients can only use what the server provides and allows
- Communication between clients goes through the server
- Servers can and **do** exercise control of usage
- This made it a very popular model among businesses

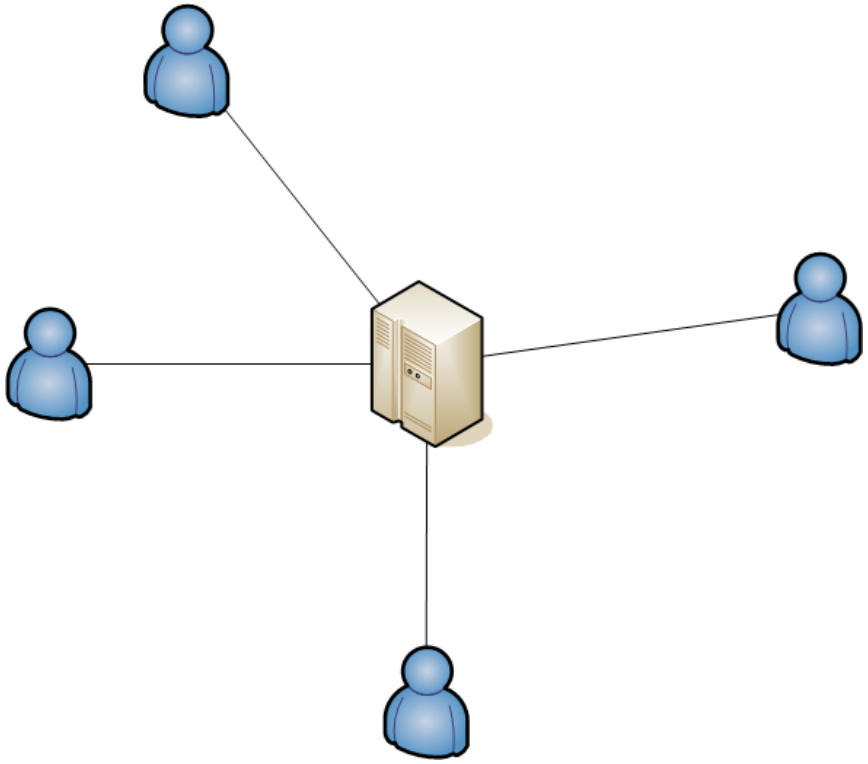
Client-Server model evaluation

- Simple architecture
- Works with simple clients
- Centralized logic is easy to maintain/upgrade
- Centralized data storage is easy to maintain

- Can impose restrictions on client communication
- Clients cannot communicate directly
- It can have privacy issues
- It has scalability issues
- The server is a single point of failure
- Acquiring high availability is hard and costly

Client-Server model evolution

It started with small servers holding a few resources, to which a small number of clients had access

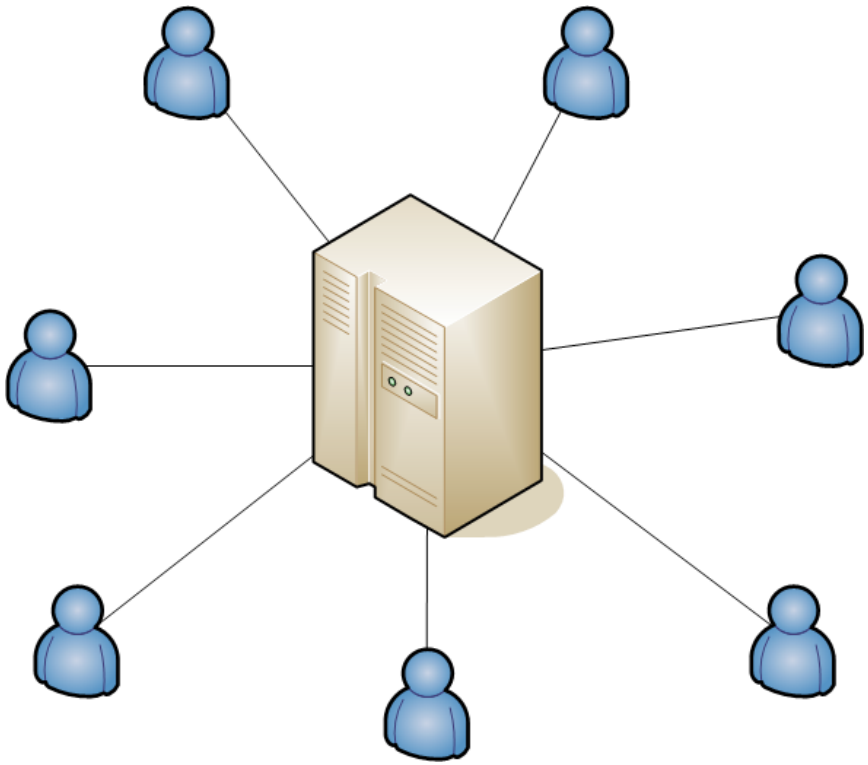


Client-Server model evolution

But other people liked the idea and they also wanted to have access to those resources...

Client-Server model evolution

... so larger servers have been built to handle more clients

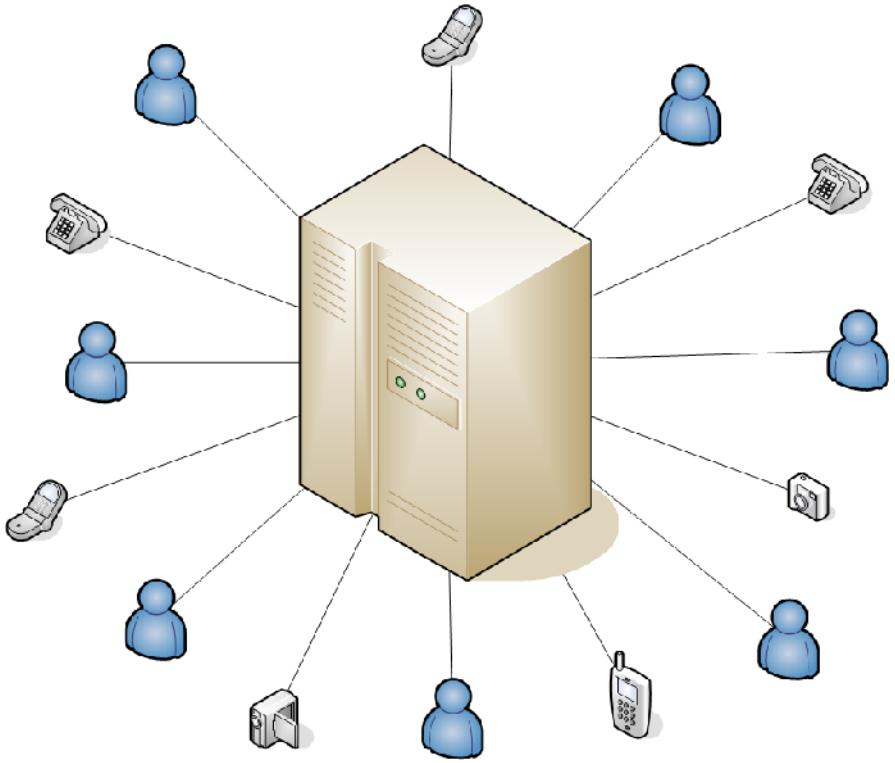


Client-Server model evolution

But more and more people wanted to join in, to be able to access the resources...

Client-Server model evolution

... so they have built even larger servers hosting multiple services and handling all kind of new fancy devices

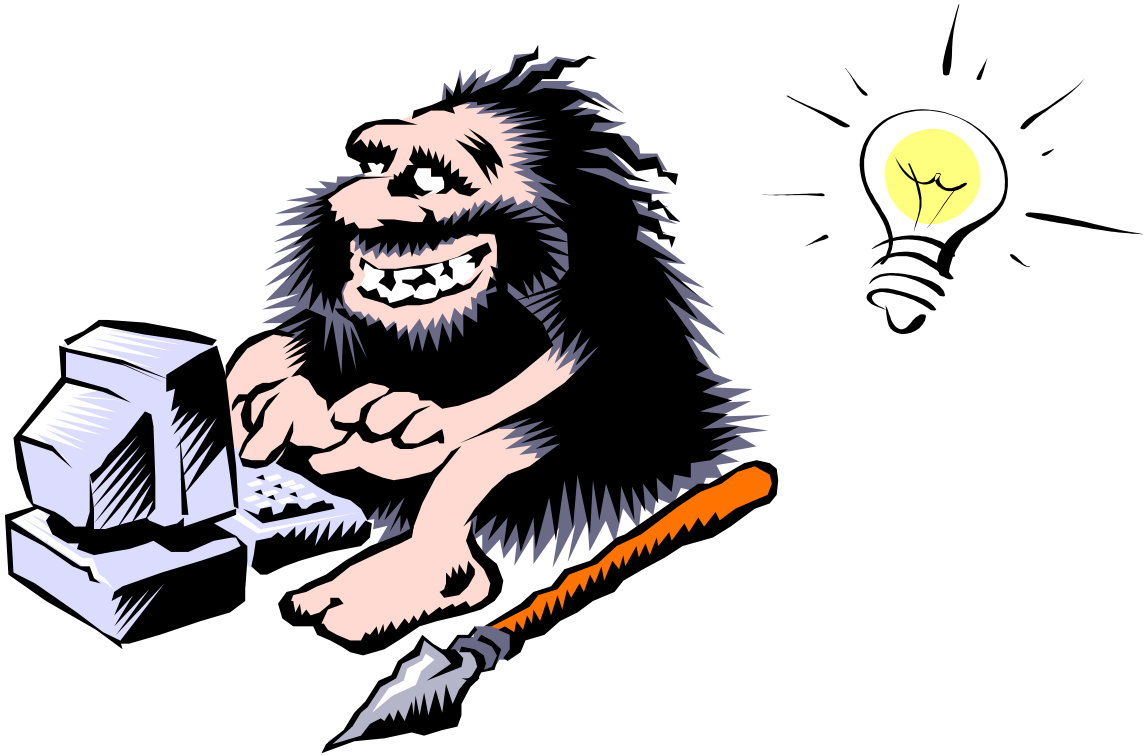


Client-Server model evolution

And all was well until scalability and availability problems started to raise their heads

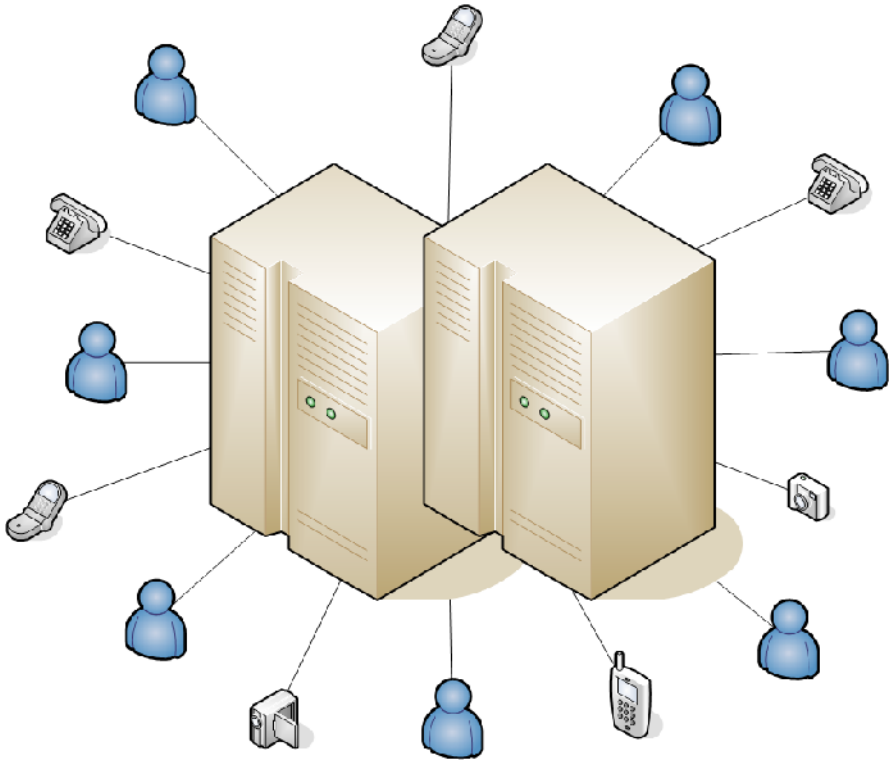
Client-Server model evolution

But some smart guys thought hard about them and came up with solutions...



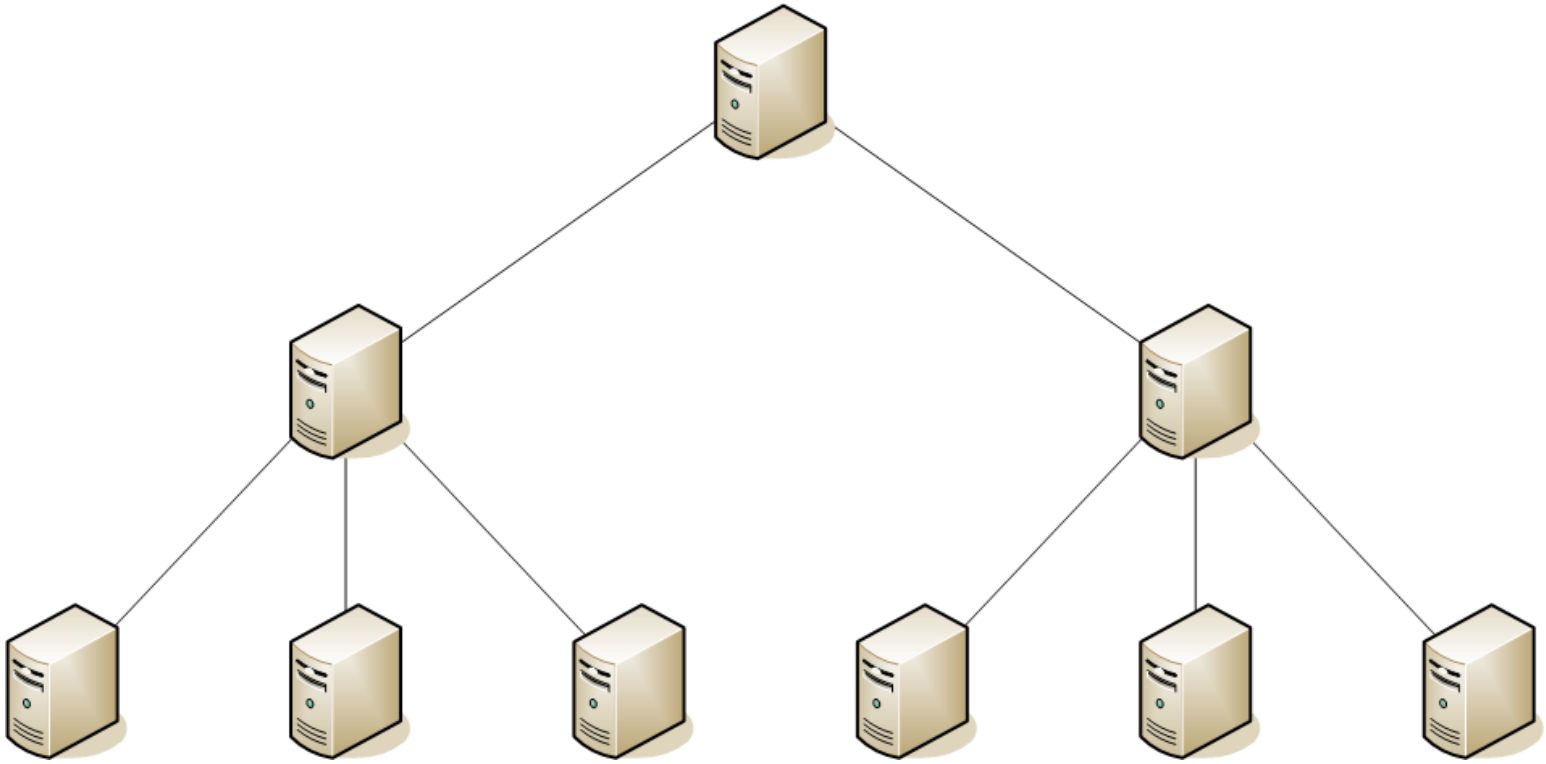
Client-Server model evolution

And so the clusters were born to address high availability issues and eliminate single points of failure...



Client-Server model evolution

... and multilevel bad balancing schemes were created to address scalability issues



Client-Server model evolution

But at this point the architecture was no longer simple...

The systems became hard to build and maintain. They became costly and required highly skilled individuals to keep them up and running

And more often than not, failures in such complex systems lead to frustration on all levels

Client-Server model conclusion

And yet the main issues are still unresolved...

- The single point of failure became a double point of failure
- A central database is still a single point of failure
- Load balancers are bottlenecks and single points of failure
- There are still privacy issues
- The service providers can create and manage artificial bottlenecks that control what a client can do, with the sole purpose of maintaining old revenue schemes

Client-Server model conclusion

No matter what enhancements were made to the model, there is one element that stays at the base of its principle that has never changed: the fact that there is a central entity that has all the logic and the resources under its command.

Keeping all logic in one place — the root of all problems

Client-Server model conclusion

The moral of the story...

Understanding P2P

- How it came to life
- What it tries to solve
- What it is
- How it works

Issues addressed by P2P

- Eliminate the control that can be imposed on clients
- High availability
- Scalability
- Privacy

How to address them

- By distributing the logic and the resources
- By expanding horizontally not vertically (collaboration)
- By adapting to network changes on the fly
- By using direct client-to-client communication

P2P model evaluation

- Scalable
- Distributed (no single point of failure)
- Intelligence moved to the network border (clients)
- No centralized control
- No server maintenance

- More complex
- Higher latency in routing
- Distributed data storage is hard to do

Understanding P2P

- How it came to life
- What it tries to solve
- What it is
- How it works

P2P definitions

- A self-organizing, distributed network of entities which contribute their individual resources and collaborate in order to reach the goal for which the network was built.

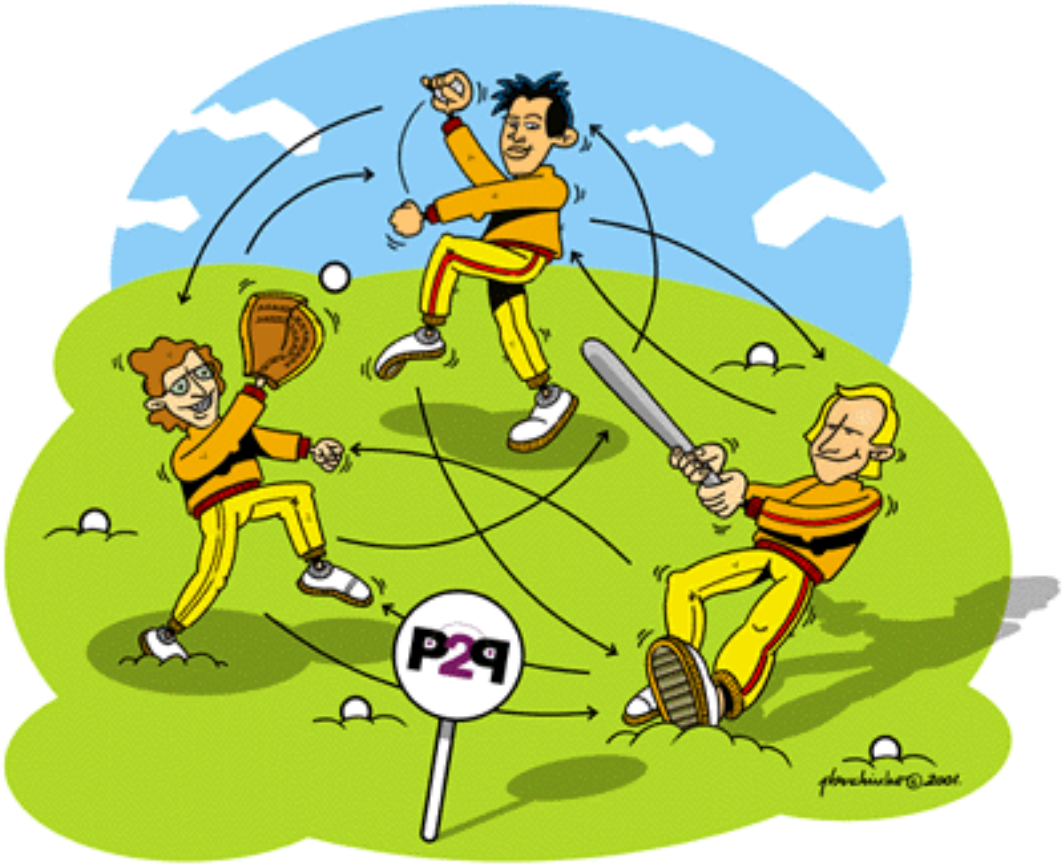
- P2P networks are those which exhibit 3 characteristics:
  - self-organization
  - distributed control/resources
  - symmetric communication

Overlay definition

An overlay network is a virtual network of nodes and logical links that is built on top of an existing network with the purpose to implement a network service that is not available in the existing network. *I. Stoica*

- A P2P network is an overlay itself (over TCP/IP)
- There can be overlays over a P2P network as well

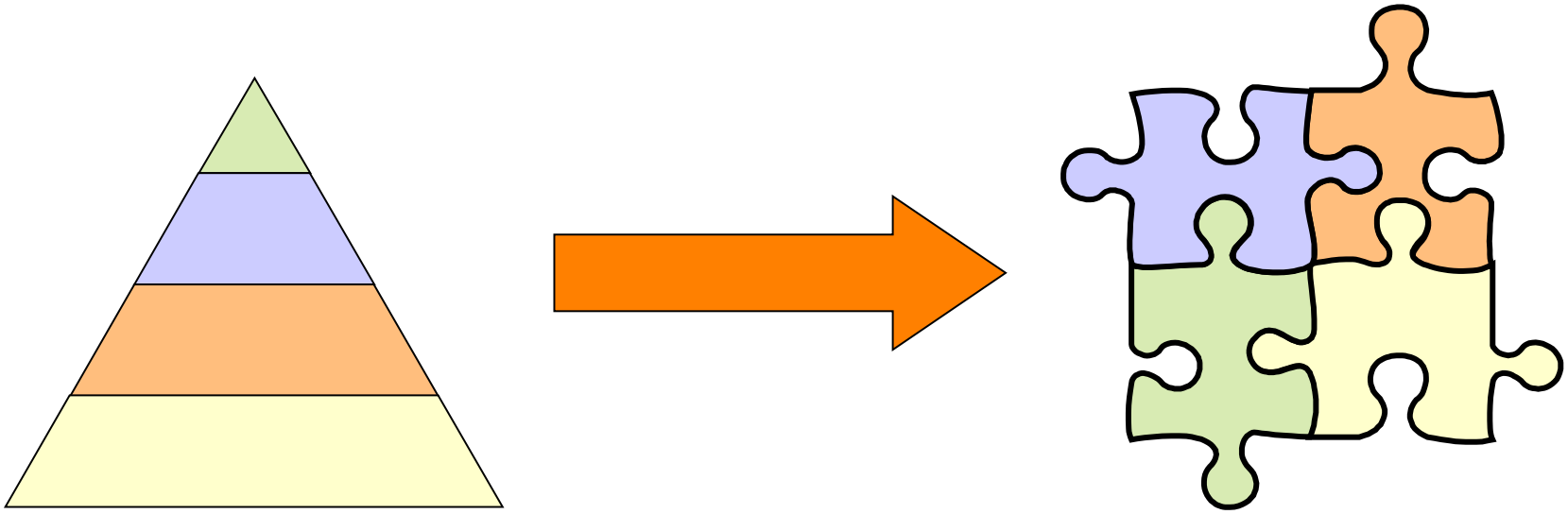
The P2P concept in images



The P2P concept in images

How P2P changes the picture

P2P networks replace centralization and hierarchy with distribution and collaboration. At a phibsophical level they replace centralized control with responsibility and freedom.

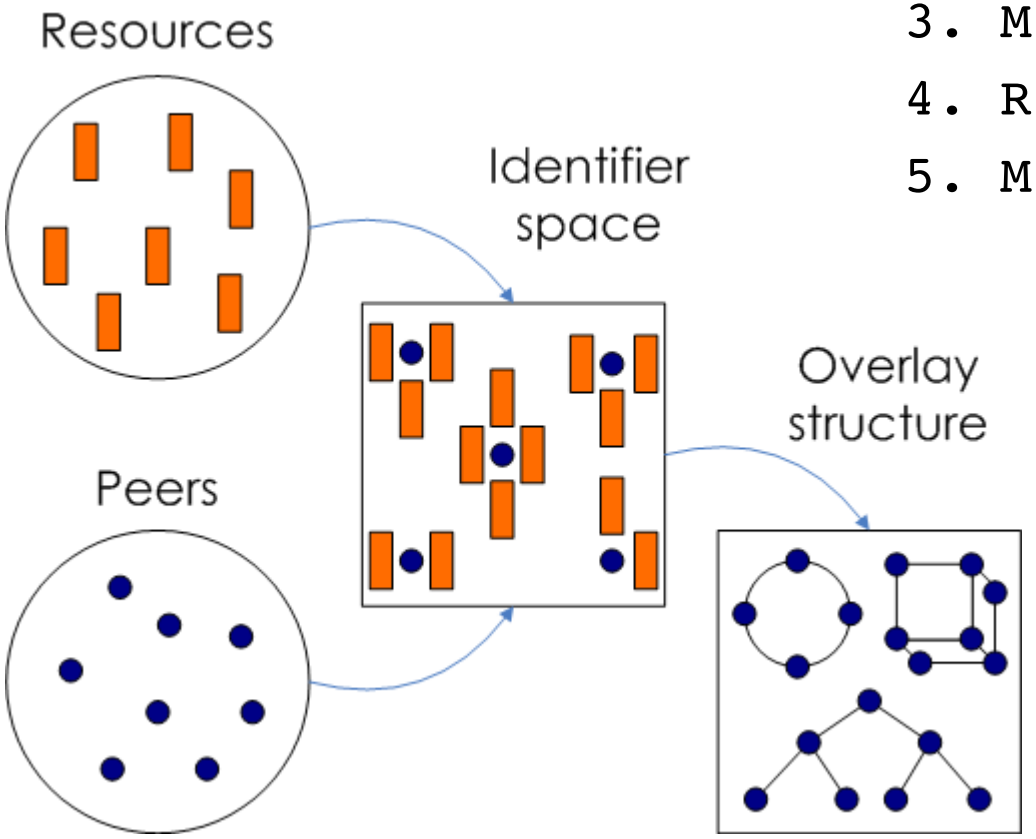


Understanding P2P

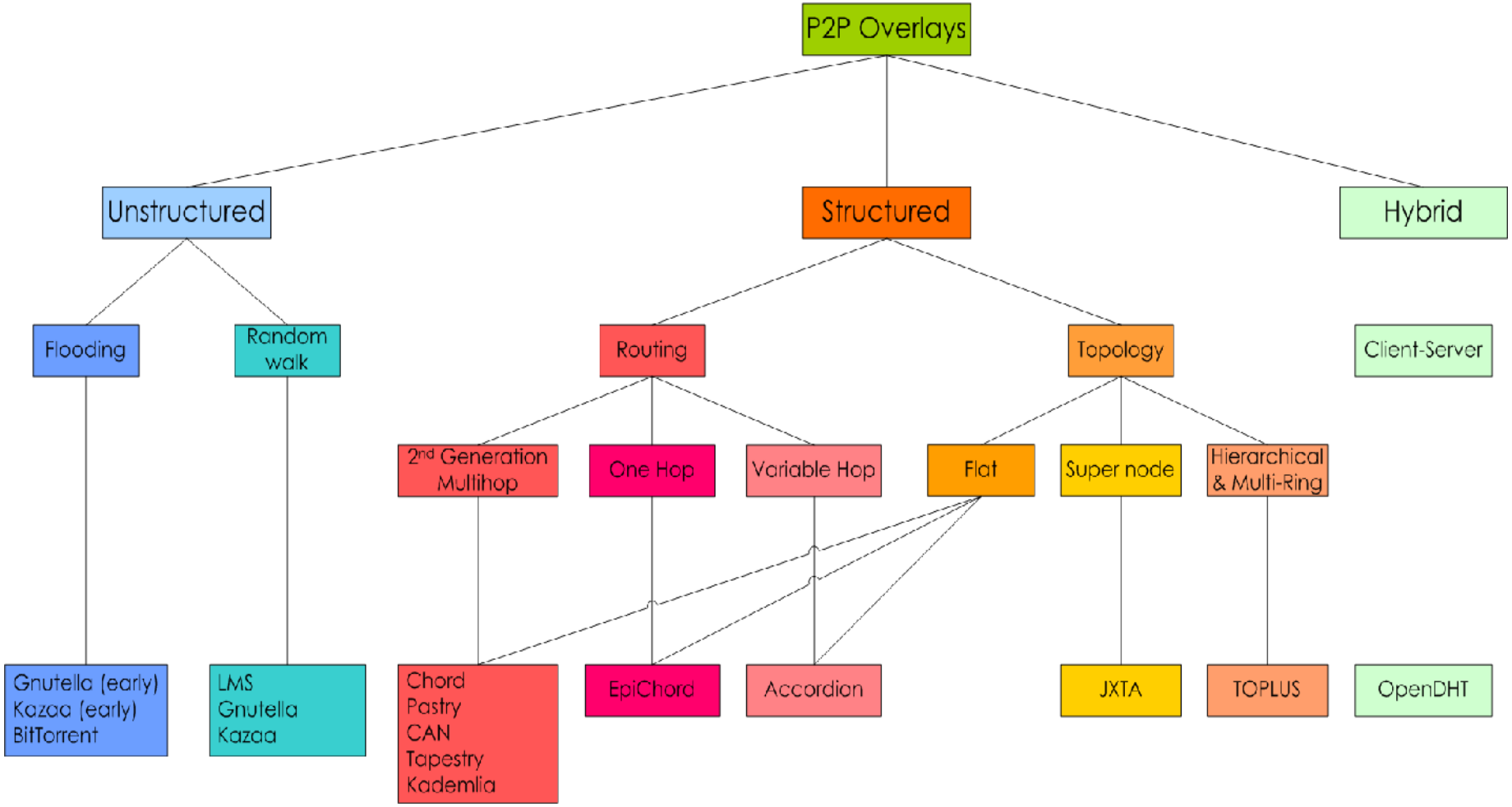
- How it came to life
- What it tries to solve
- What it is
- How it works

P2P overlay design

1. Choice of identifier space (IS)
2. Map resources and peers to IS
3. Management of the IS by peers
4. Routing strategy
5. Maintenance strategy



The P2P family

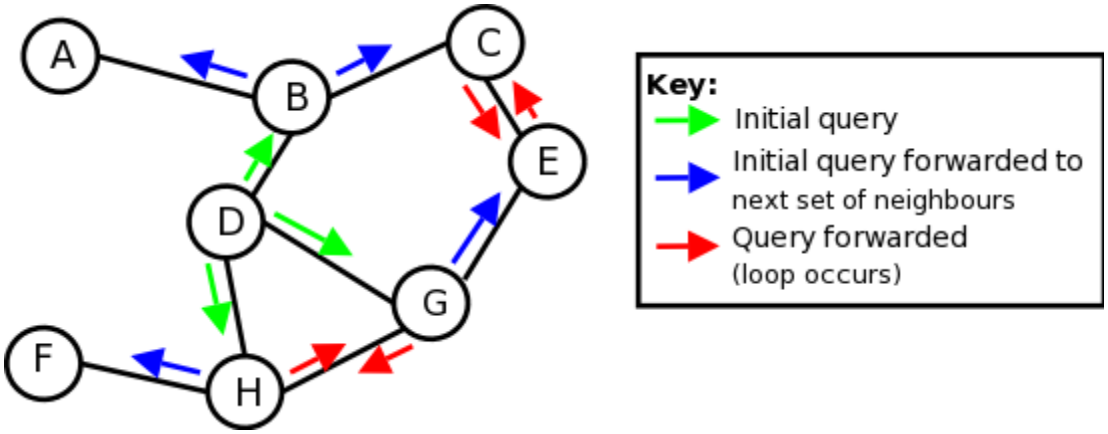


Index types in P2P networks

- Local
  - Each peer only indexes its own content and flood queries widely
  - Can perform complex searches (rich queries not just key lookups)
  - Are becoming rare (used mostly by unstructured P2P networks)
- Centralized
  - Hybrid systems: index centralized and data distributed
  - Can perform complex searches
  - Fast lookup, but single point of failure (e.g. Napster)
- Distributed
  - Also known as Distributed Hash Table or DHT
  - Most widely used nowadays (structured P2P networks)
  - Efficient key lookup / routing
  - Can perform only exact key lookups

1st generation P2P networks

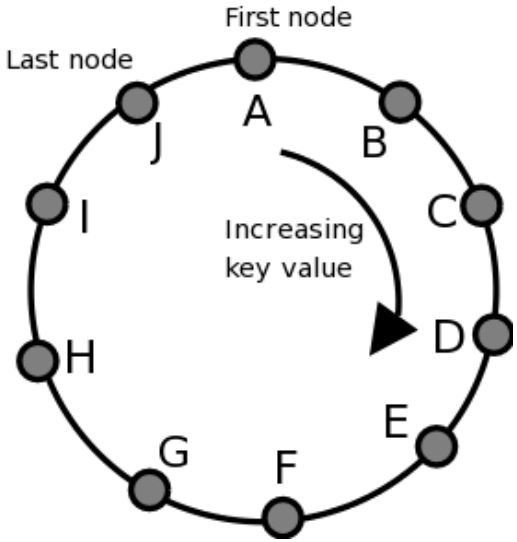
- Unstructured (had scalability issues)
- Mostly created and used by file sharing programs
- Use inefficient search strategies: flooding/random walk
- Have high routing costs (latency and bandwidth)
- May fail to find an available resource
- Can search by keywords



Search by flood-routing the request

2st generation P2P networks

- Structured
- Most of them use a Distributed Hash Table (DHT)
- Guarantee finding a target in a bounded number of hops
- Can only search for exact keys (no keywords)
- Highly transient peers are not well supported by DHTs

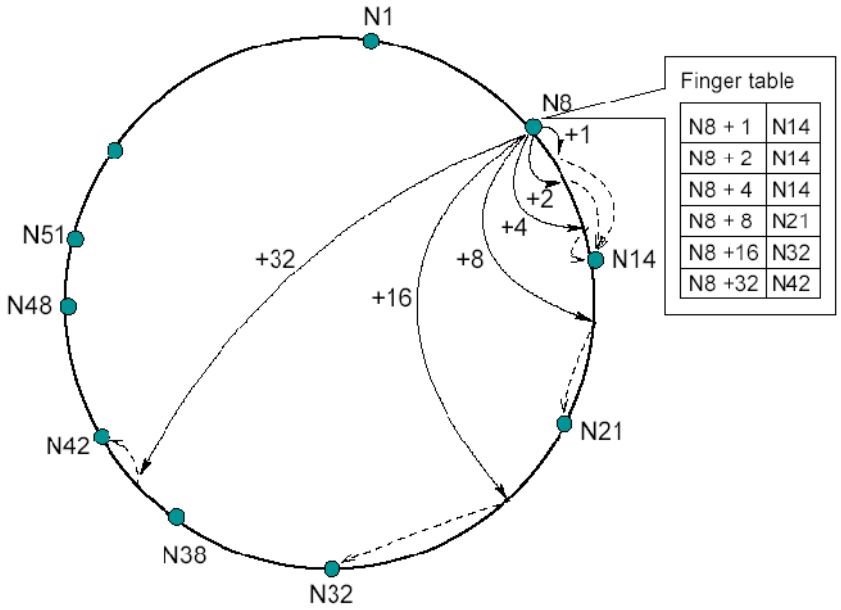


A structured node ring which uses a hashing function to order the nodes in ascending order. Network resources are also indexed using the hashing functions and each node handles the resources which have a hash in the node's neighborhood.

2st generation multihop overlays

	Pastry	CAN	Chord	Tapestry
Source	Microsoft	ICSI	MIT	UC Berkley
Overlay network	Yes	Yes	Yes	Yes
DHT	Yes	Yes	Yes	Yes
Topology	Flat	Flat	Flat	Flat
Routing	Prefix based multihop	Cartesian routing in N-dimensional space	Finger table	Longest prefix multihop
Routing performance	O(log N)	O(log N)	O(log N)	O(log N)
Routing table size	O(log N)	O(log N)	O(log N)	O(log N)

The Chord DHT overlay



- Uses SHA-1 hashes (160 bits)
- Maps nodes and keys to a ring
- O(log N) performance
- O(log N) routing table size
- Supports join and leave operations for maintaining the network
- It basically supports one operation: lookup a node for a given key

- Each node knows its predecessor, successor and keeps a list of successor nodes known as the finger table, which is used to improve lookup performance and increase fault tolerance
- Each node handles the resources which have their hashes mapped between the node itself and its predecessor
- If a lookup doesn't yield a local resource, it is forwarded to the node in the finger table which has the closest hash value preceding the hash of the queried resource

P2P networks today

- The structured vs. unstructured taxonomy is fading away
  - Unstructured networks have evolved and incorporated structure
  - There are emerging schema based P2P designs with super-node hierarchies and structure within documents which are quite distinct from the structured DHT proposals (Neidl, Siberski et al. 2003)
  - Most, if not all, P2P designs today assume some structure
- Exotic P2P designs are appearing to address specific needs that resulted from experience.
  - For example the MUTE project is using a routing algorithm inspired by the way ants track their food using pheromone trails, to implement a file sharing P2P network with complete anonymity. http://mute-net.sourceforge.net

P2P SIP Technologies

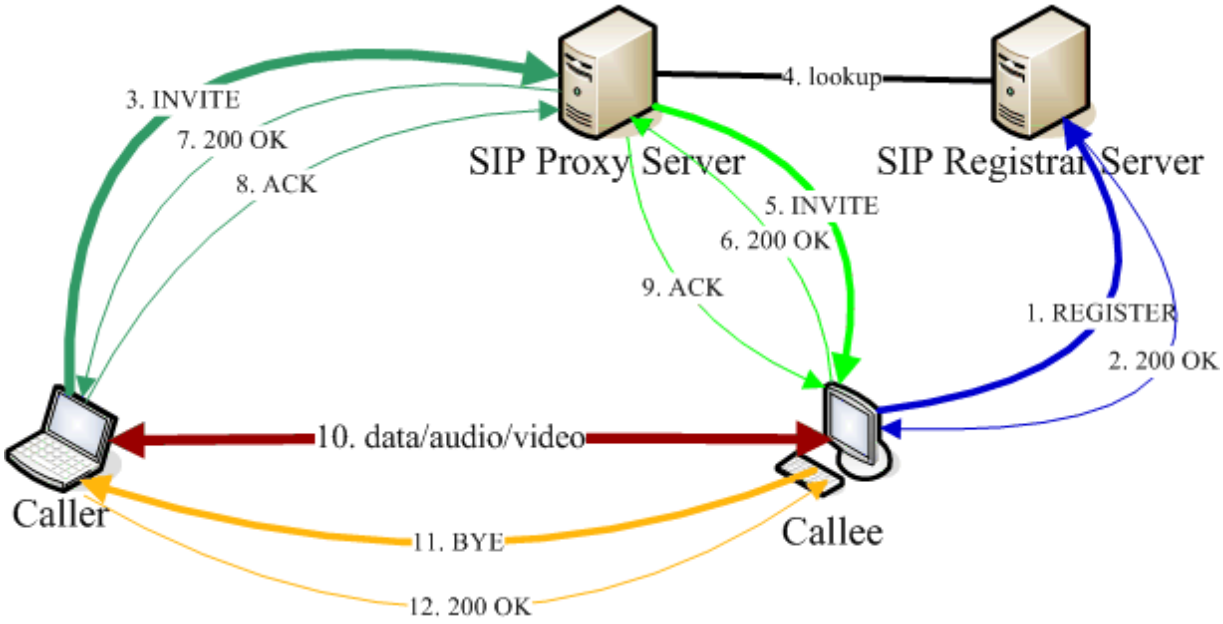
- SIP meets P2P
- P2P SIP models
- P2P SIP and NAT
- Conclusions

SIP technologies today

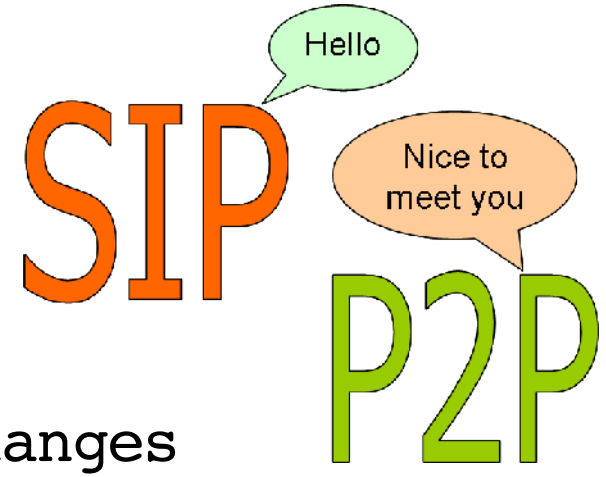
- Based around user agents, proxies and registrars
- No clear client server model
- User agents use direct symmetric communication
- Proxies and registrars are only a mechanism to find an AOR
- Proxies only route SIP messages (exercise almost no control)
- User agents have usually more intelligence than SIP proxies

SIP callsetup today

SIP



SIP meets P2P



SIP is already ready for P2P with little changes

- Uses symmetric, direct client-to-client communication
- Intelligence resides mostly on the network border in the user agents
- The proxies and the registrar only perform bokup and routing
- All that user agents lack to build a P2P network is bokup and routing

The bokup/routing functions of the proxies/registrar can be replaced by a DHT overlay built in the user agents. By adding **join**, **leave** and **bokup** capabilities, a SIP user agent can be transformed into a peer capable of operating in a P2P network

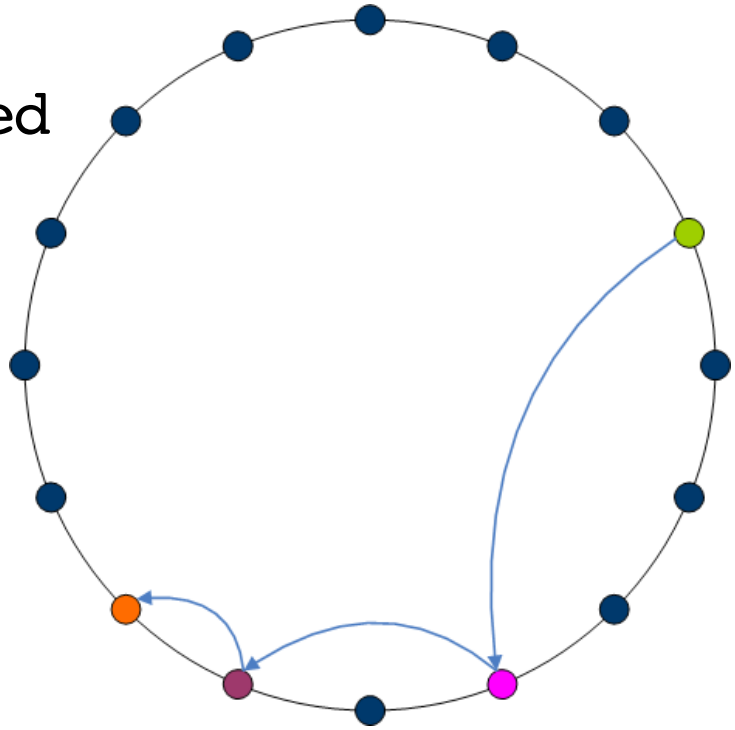
P2P SIP Technologies

- SIP meets P2P
- P2P SIP models
- P2P SIP and NAT
- Conclusions

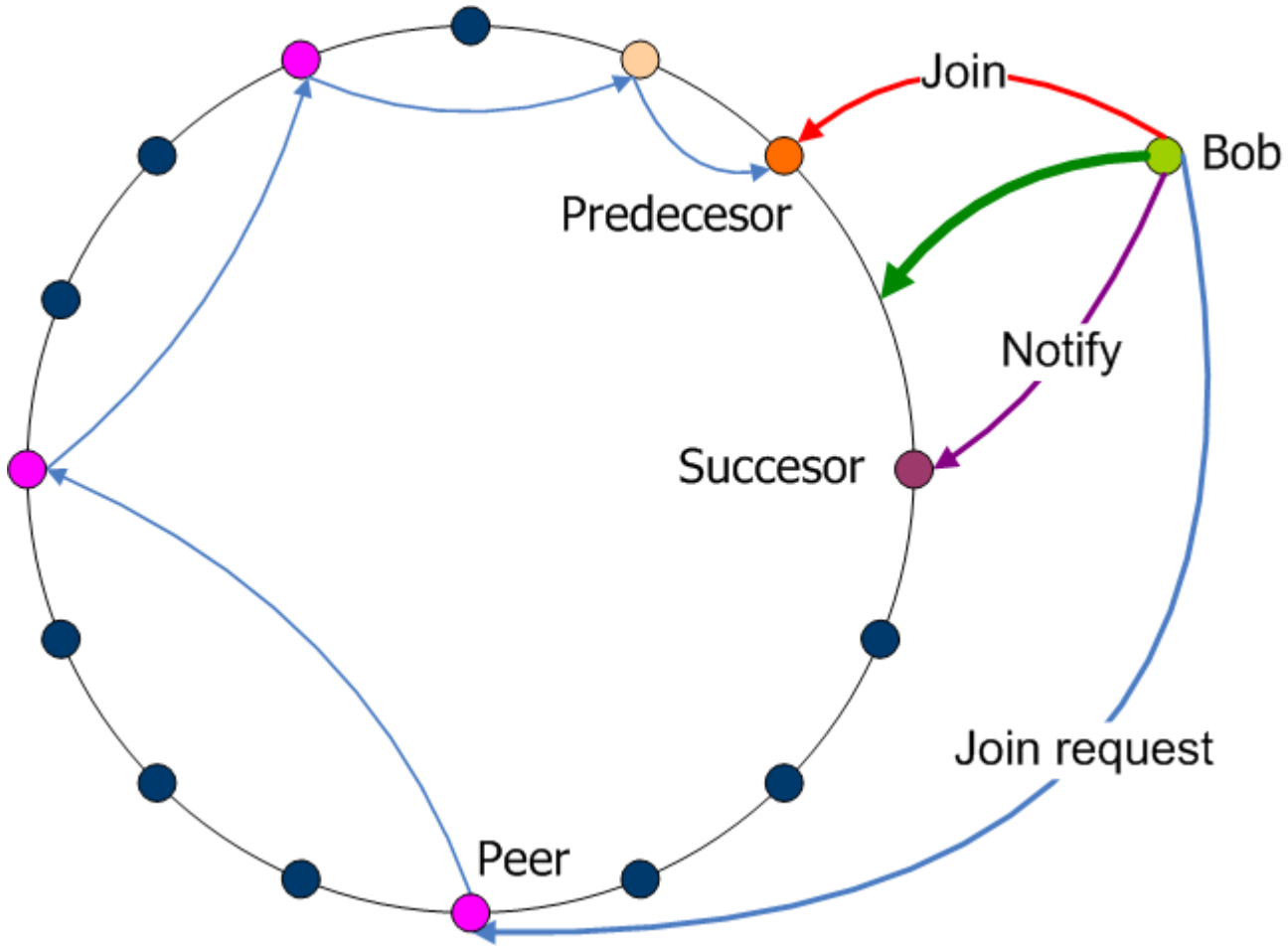
The pure P2P SIP model

- SIP AOR bokups use exact key matches, so we can use a 2nd generation DHT based overlay (like Chord)

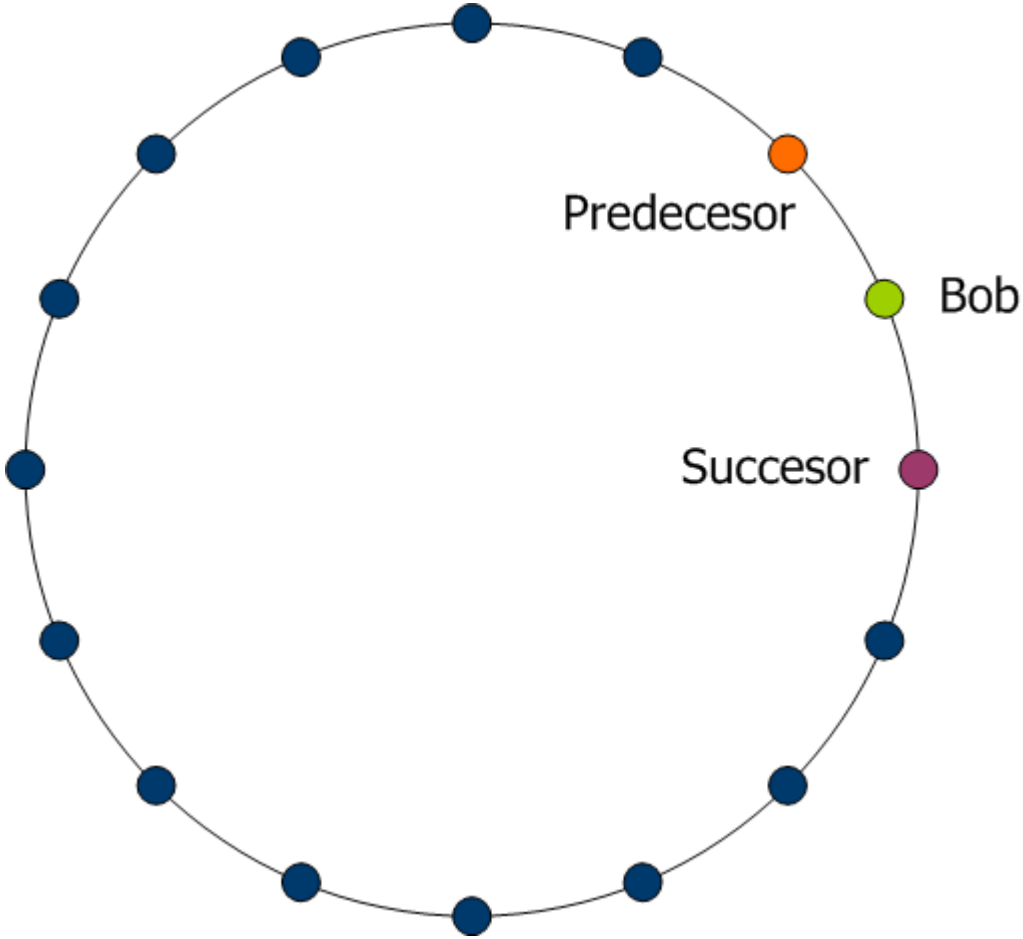
- A SIP user agent can be transformed into a P2P SIP client by adding **join**, **leave** and **bokup** capabilities to it



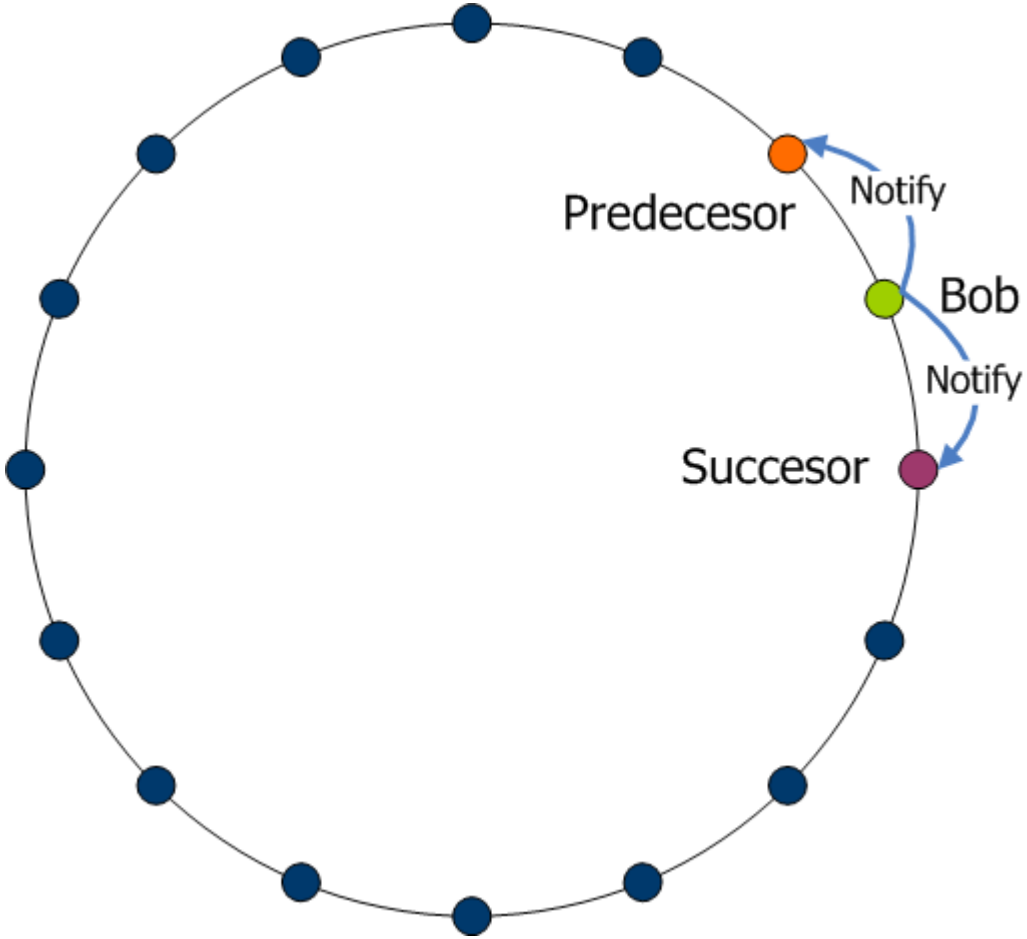
Joining a P2P SIP overlay



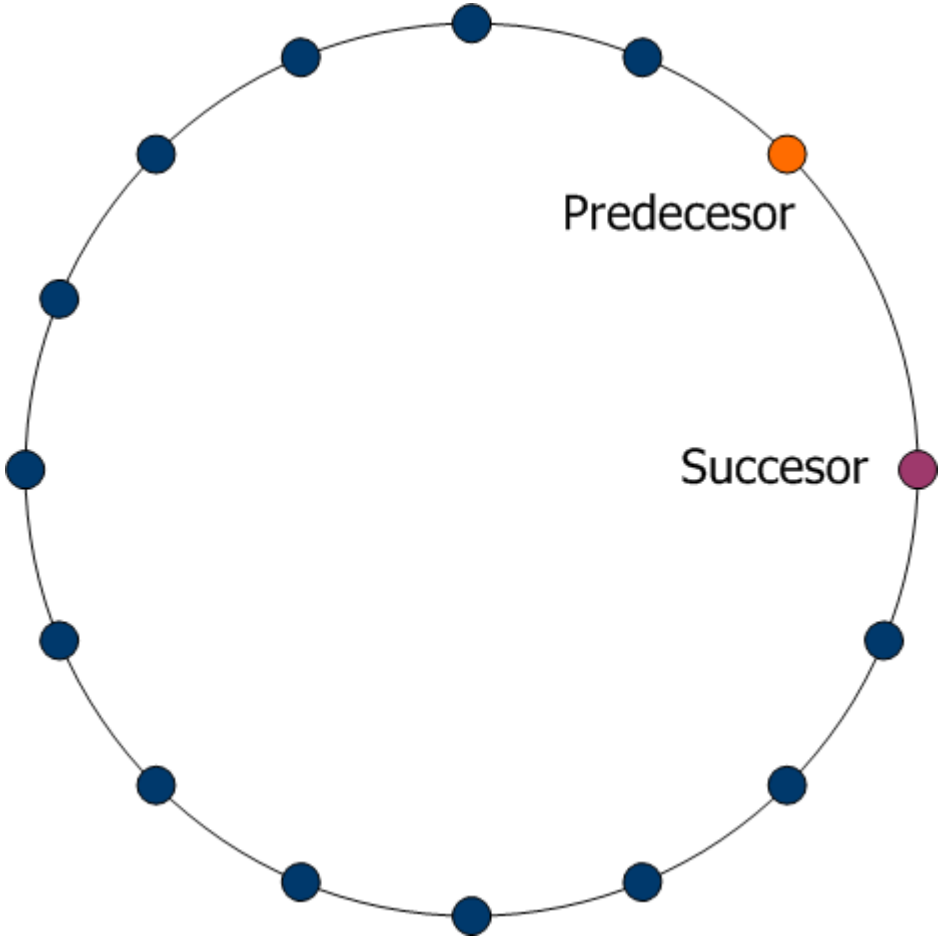
Joining a P2P SIP overlay



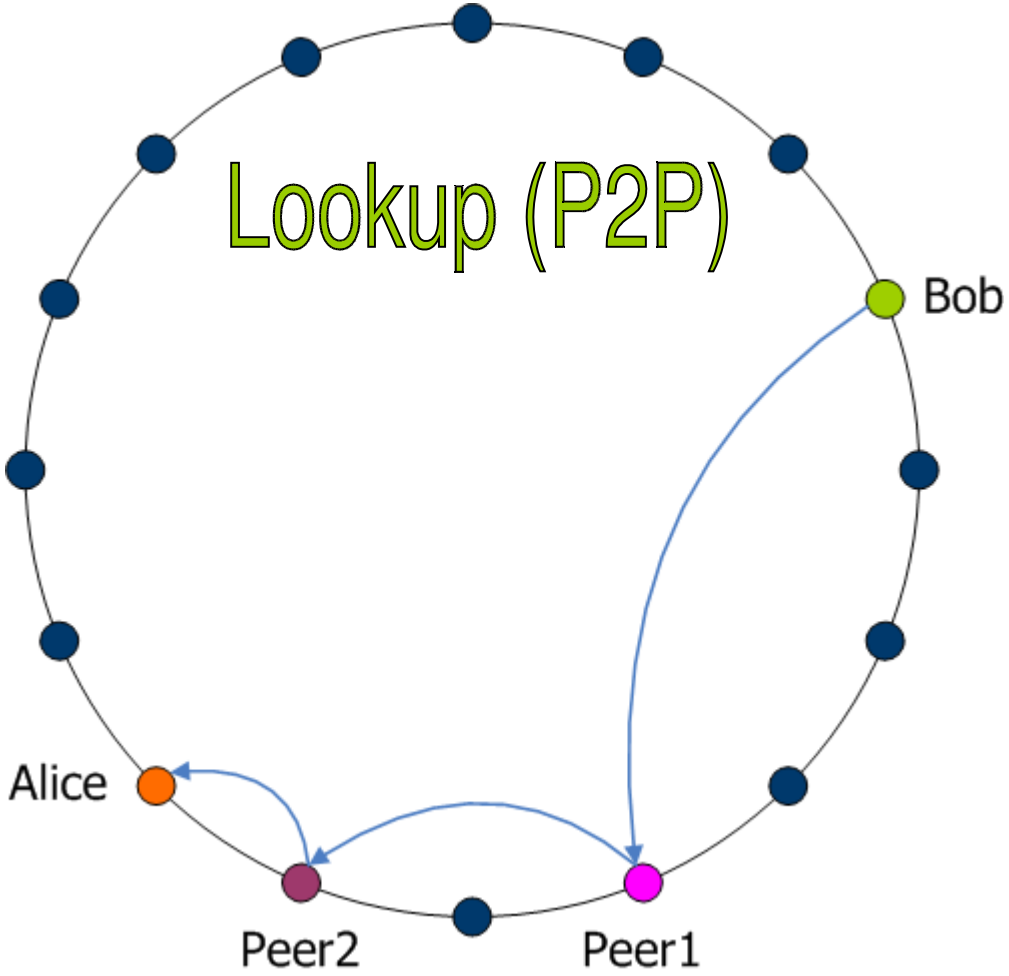
Leaving a P2P SIP overlay



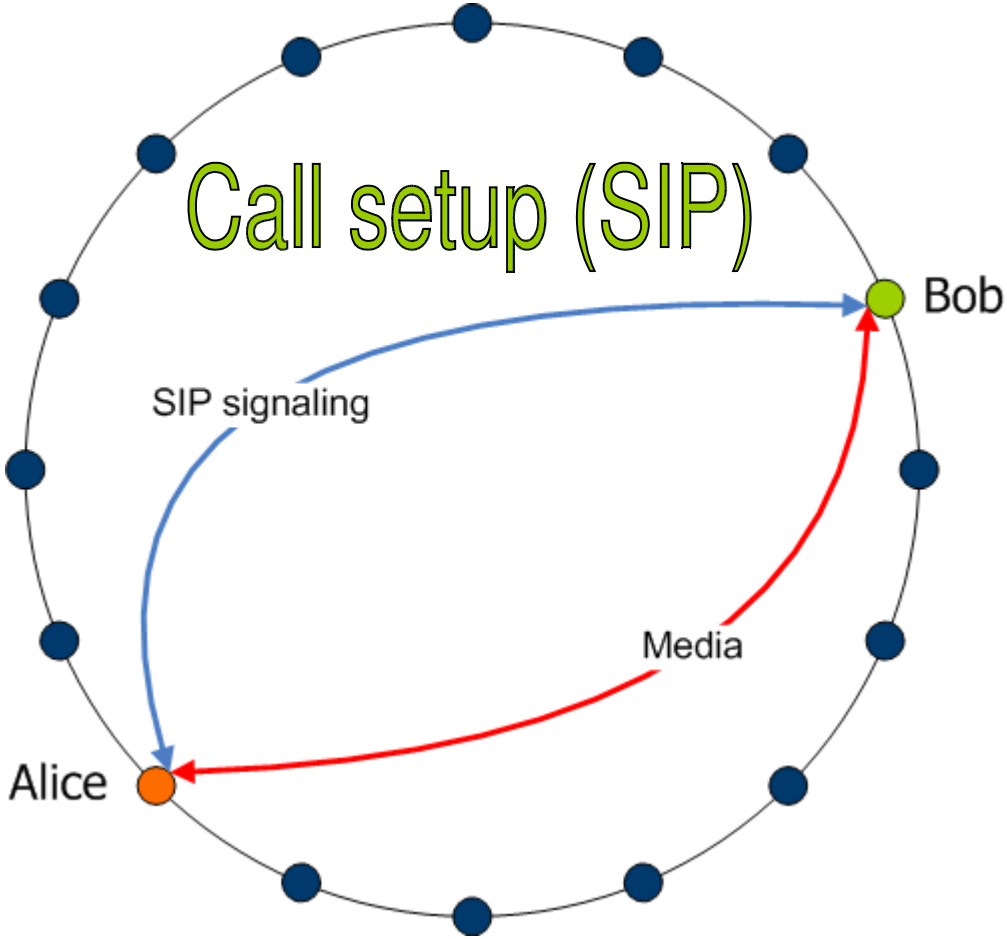
Leaving a P2P SIP overlay



Callsetup in pure P2P SIP

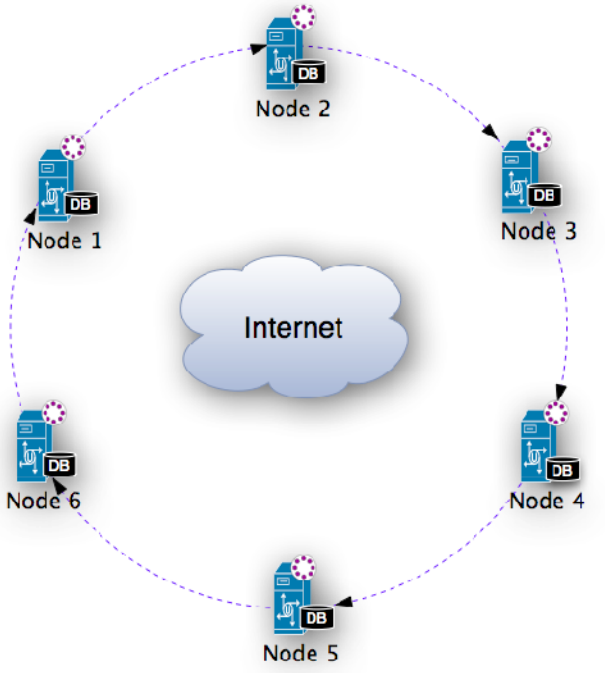


Callsetup in pure P2P SIP



The hybrid P2P SIP model

- Create a P2P overlay of SIP proxies, media relays, ...
- The overlay updates DNS with the list of available members
- Load is distributed evenly among members
- Works with unmodified user agents



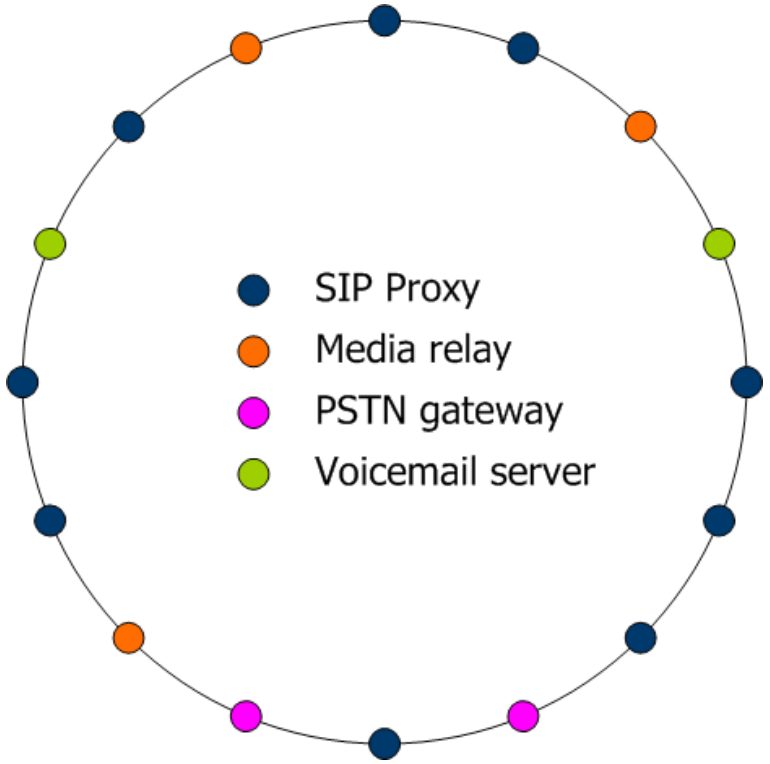
The hybrid P2P SIP model

- The overlay is smaller and more robust than pure P2P SIP
- Lookup is faster (O(1)) compared with pure P2P SIP
- Eliminates single points of failure and bottlenecks
- Addresses both scalability and availability
- Better than clusters or bad balancing schemes
- Works with existing user agents

Good transitional model

Logical overlays

- Each peer joining the overlay can declare its abilities
- All peers sharing the same abilities form a logical overlay
- Allows for more refined searches
- Logical overlays are cheap



P2P SIP Technologies

- SIP meets P2P
- P2P SIP models
- P2P SIP and NAT
- Conclusions

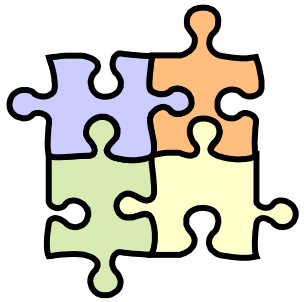
P2P SIP and NAT

- Nowadays NAT is almost a given
- Clients that are behind NAT must use another peer as proxy
- Such clients are 2nd class citizens (they're not really peers)
- Such clients must use both SIP and media relays
- To address such issues some designs use super-nodes
- A pure P2P SIP overlay where most clients are behind NAT becomes very much equivalent with a hybrid overlay
- The hybrid model doesn't suffer from this problem

P2P SIP Technologies

- SIP meets P2P
- P2P SIP models
- P2P SIP and NAT
- Conclusions

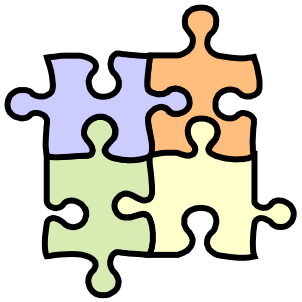
P2P conclusions



P2P has the makings of a disruptive technology – it can aggregate enormous storage and processing resources while minimizing entry and scaling costs.

The key to realizing this potential in applications other than file sharing is robustness. Many P2P structured designs have implemented strategies to increase fault tolerance in the presence of highly volatile peers and high churn rates. However the assumptions they make may not always hold true in real life (for example Chord assumes independent node leave, while in practice node leave may be correlated).

P2P conclusions



The P2P concept brings challenges not only on a technical level, but on a phibsophical level as well, as it encourages us to review our models and paradigms, as well as our way of thinking. P2P brings into equation new concepts that can change completely the way we view or do things, bringing them on a whole new level.

Questions?

Thank you,

Dan Pascu
dan@ag-projects.com
http://ag-projects.com