Architectural styles for software systems

Peer to Peer
Agenda

- P2P: overview
- Basic types of P2P systems
- Case study: Skype
- Case study: Bitcoin
Peer to peer computing

- a class of applications that takes advantage of resources—storage, cycles, content, humans—available at the edges of the Internet
- Because accessing these decentralized resources means operating in an environment of unstable connectivity and unpredictable IP addresses, peer-to-peer nodes must operate outside the DNS system and have autonomy from central servers
Layers in P2P

The overlay network layer is responsible for implementing an efficient routing algorithm: the nodes in the system are structured in order to decrease the search steps necessary to find the target identifier.

Each node maintains a local routing table, which holds the identifiers of other nodes in the system.
Distributed hash tables

A distributed hash table (DHT) provides a lookup service similar to a hash table: (key, value) pairs are stored in a DHT, and any participating node can efficiently retrieve the value associated with a given key.
Distributed ledgers and blockchains

Distributed ledger

Blockchain
Distributed ledgers vs blockchains

- e.g. Bitcoin
- e.g. Blockchain
- Distributed ledger technology

Implementation
A P2P system is a distributed collection of peer nodes.

Each node is able to provide services, as well as to make requests, to other nodes:
- Each node acts as both a server and a client.

The goal of this style:
- To share resources and services (data, CPU, disk,…).
Peer-to-peer systems

- File sharing systems based on BitTorrent
- Messaging systems such as Jabber
- Blockchains – Bitcoin and Ethereum
- Databases – Freenet is a decentralized database
- Phone systems – Viber or Skype
- Computation systems - SETI@home
P2P: requirements and drivers

- Typical functional characteristics of P2P systems:
  - File sharing system
  - File storage system
  - Distributed file system
  - Redundant storage
  - Distributed computation

- Typical non-functional requirements:
  - Availability
  - Reliability
  - Performance
  - Scalability
  - Anonymity
Peer: CRC

<table>
<thead>
<tr>
<th>Class</th>
<th>Peer</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Responsibilities</strong></td>
<td></td>
</tr>
<tr>
<td>• Component</td>
<td></td>
</tr>
<tr>
<td>• Handles User interaction</td>
<td></td>
</tr>
<tr>
<td>• Asks other Peers for searching some data</td>
<td></td>
</tr>
<tr>
<td>• Asks some Peers for obtaining some data</td>
<td></td>
</tr>
<tr>
<td><strong>Collaborators</strong></td>
<td></td>
</tr>
</tbody>
</table>
P2P: Brief History

- Although they were proposed several years ago, they mainly evolved in the last 20 years
- File sharing systems showed the power of the concept (Napster, 1999; Gnutella, 2000)
- In 2000, the Napster client was downloaded in few days by 50 millions users
  - Traffic peak of 7 TB in a day
- Gnutella followed Napster’s footprint
  - The first release was delivered in 2003
  - In June 2005, Gnutella's population was 1.81 million computers; in 2007, it was the most popular file sharing network with an estimated market share > 40%
  - Host servers are listed at gnutellahost.com
The phases of a P2P application

A P2P application is organized in three phases:

- **Boot**: a peer connects to the network and actually performs the connections (remark: P2P boot is rare)
- **Lookup**: a peer looks for a provider of a given service or information (generally providers are SuperPeers)
- **Resource sharing**: resources (requested and found) are delivered, usually in several segments
P2P resource sharing example: bitTorrent
There are three types of P2P architecture, different with respect to the lookup phase:

- **Centralized**
  - Centralized network architecture uses a centralized indexed server to maintain a database of all the content and users at any time
  - The database is updated whenever a peer logs on to the network

- **Decentralized (Pure P2P)**
  - Each peer acts as an index server, searches and holds its own local resources, and as a router, relaying queries between peers

- **Hybrid Architecture**
  - Deploys a hierarchical structure by establishing a backbone network of Super Nodes that take on the characteristics of a central index server
P2P: Centralized Index

- There is a centralized index used to search the information.
- When a peer connects, it informs the central server:
  - IP address
  - Content
- File transfer is decentralized, but locating content is highly centralized.
- Example: Napster
Centralized: Architecture

Components:
- Peer
  - An entity with capabilities similar to other entities in the system
  - Each node is both a server and a client
  - Autonomous: no administrative authority
  - Unreliable: nodes enter and leave the network “frequently”
- Index Server
  - An entity with special capabilities:
    - Allow peer to join the system
    - Allow the research of content
    - Maintain a database of all the content and users at any time, which is updated whenever a peer logs on to the network

Connectors:
- Network protocol
  - Often specialized for P2P communication
Centralized Index: Component Diagram
Centralized Index: Class Diagram

```
Directory Index
+ Search()
+ Login()

Peer
+ FileTransfer()

requester
+ *
provider
```

1
Centralized Index: Sequence Diagram
Centralized Index Example: Napster
Centralized Index: Pro & Cons

Benefits:
- Low per-node state
- Limited bandwidth usage
- High success rate
- Fast search response time
- Easy to implement and maintain

Pitfalls:
- Single point of failure
- Vulnerable to censorship
- Limited scale
- Possibly unbalanced load
- Database might be obsolete
Decentralized P2P organizes the overlay network as a random graph.

Each node knows about a subset of nodes, its “neighbors”:
- Neighbors are chosen in different ways:
  - physically close nodes, nodes that joined at about the same time, etc.

Example: Gnutella, Bitcoin
Decentralized : Class Diagram

Peer

service1()
service2()
...
serviceN()

requester
*

provider
*
Decentralized: Architecture

- **Components:**
  - **Peer**
    - An entity with capabilities similar to other entities in the system
    - Each node is both a server and a client
    - Autonomous: no administrative authority
    - Unreliable: nodes enter and leave the network “frequently”
    - Local knowledge: nodes only know a small set of other nodes

- **Connectors:**
  - **Network protocol**
    - Often specialized for P2P communication
Decentralized Example: Gnutella

query: "Baby Go Home.mp3"

6-7 levels depending on "time to live"

"I've got it!"

8,000 - 10,000 computers
Decentralized: Component Diagram

**Fig. 2.** Typical component-based structure of a Gnutella servant
Fig. 3. UML Sequence diagram of a search session
Fig. 4. UML Sequence diagram of a reply session
Decentralized: Pro & Cons

- **Benefits:**
  - Limited per-node state
  - Fault tolerant

- **Pitfalls:**
  - High bandwidth usage
  - Long time to locate item
  - No guarantee on success rate
  - Possibly unbalanced load
P2P: Hybrid Architecture

- Deploys a hierarchical structure by establishing a backbone network of Super Nodes that take on the characteristics of a central index server
- When a client logs on to the network, it makes a direct connection to a single **Super Peer**
- Example: **Skype**
Example: Skype
Hybrid: Architecture

- **Components:**
  - **Peer**
    - An entity with capabilities similar to other entities in the system
    - Each node is both a server and a client
    - Autonomous: no administrative authority
    - Unreliable: nodes enter and leave the network “frequently”
  - **Super Peer**
    - Gathers and stores information about peer and content available for sharing
    - Act as servers to regular peer nodes, peers to other super Peers
    - Maintain indexes to some or all nodes in the system
- **Connectors:**
  - **Network protocol**
    - Often specialized for P2P communication
Hybrid Architecture: Component Diagram
Hybrid Architecture: Class Diagram

Peer

FileTransfer()

Super Peer

Search()
Login()
DisseminateQuery()

Simple Peer

requester

provider

35
Hybrid Architecture: Sequence Diagram
Hybrid Architecture Example: Skype (1)
Hybrid Architecture Example: Skype (2)

- A mixed client-server and peer-to-peer architecture addresses the discovery problem.
- Replication and distribution of the directories, in the form of supernodes, addresses the scalability problem and robustness problem encountered in Napster.
- Promotion of ordinary peers to supernodes based upon network and processing capabilities addresses another aspect of system performance:
  - “not just any peer” is relied upon for important services.
Hybrid Architecture Example: Skype (3)

- A proprietary protocol employing encryption provides privacy for calls that are relayed through supernode intermediaries
- Restriction of participants to clients issued by Skype, and making those clients highly resistant to inspection or modification, prevents malicious clients from entering the network
Hybrid Architecture: Pro & Cons

- **Benefits:**
  - Manageable per-node state
  - Manageable bandwidth usage and time to locate item
  - Guaranteed success

- **Pitfalls:**
  - Possibly unbalanced load
  - Harder to support fault tolerance
Bitcoins

- Bitcoins are based on the idea of avoiding to let to spend twice the same digital coin using a chain of transactions recorded in a shared ledger.

- The Bitcoin system is the **blockchain**, a P2P architecture, and transactions take place between anonymous users directly, without an intermediary.

- These transactions are verified by network nodes and recorded in a public distributed ledger called a blockchain.
Bitcoin: dynamics

A wants to send money to B.

The transaction is represented online as a block.

The block is broadcast to every party in the network.

Those in the network approve the transaction is valid.

The block then can be added to the chain, which provides an indelible and transparent record of transaction.
Bitcoins are finite, and some are lost forever

http://www.coinbuzz.com/2015/03/31/23-bitcoins-mined-13-may-lost/
Bitcoin wallet interface
Bitcoin Developer API's

Payment Processing

**Receive Payments** (My Wallet Account Required)
An incredibly easy method for websites to receive bitcoin payments. This service is completely free and secure. Perfect for business or personal use.

View Documentation

Blockchain Wallet APIs

**Blockchain Wallet API** (My Wallet Account Required)
API to send and receive payment from a Blockchain Wallet Account.

View Documentation

**Bitcoin-Qt Compatible JSON RPC** (My Wallet Account Required)
Blockchain.info is able to function as a bitcoind RPC server for merchants to interact with their My Wallet Account.

View Documentation

**Create Wallets**
Programmatically create wallets for your users with the ability to load and redeem funds.

View Documentation

Data on Transactions & Blocks

**Blockchain Data API**

Welcome Developers!
Here you will find everything you need to get started coding for bitcoin. You can use Blockchain.info’s own APIs at no cost, to help you create anything bitcoin related! 3rd Party API’s provided by other services.

Blockchain.info provides official API libraries for a number of languages

<table>
<thead>
<tr>
<th>Python</th>
<th>Java</th>
<th>.NET (C#)</th>
<th>Ruby</th>
<th>PHP</th>
<th>Node</th>
</tr>
</thead>
</table>

Installation via pip:

```
$ pip install blockchain
```

Go to GitHub for documentation and instructions
1. To initiate a wire transfer using Bitcoin, the user needs a BTC Wallet (software installed on a computer or mobile device), a BTC address & a private key.

2. Using the above, Person A initiates a cross-border money transfer in Bitcoins. The transaction created by the wallet tells the Blockchain network about Person A, the amount & Person B.

3. The Money Transfer Transaction is created with the digital signature of Person A. This can be publically verified. The transaction consists of a Debit, a credit & an implied (and very low) transaction fee like an entry in a double entry ledger. Transactions can also be moved in a 'chain' from one input to another.

4. The transaction is propagated into Bitcoin's peer-to-peer network.

5. Global Network of Miners compete to process the transaction.

6. Ownership of blocks that comprise the transaction is transferred into the target account (digital address) of Person B.

7. Person B receives the funds.

8. The transaction is recorded in the Blockchain for posterity.

The process of mining:
- New transactions are propagated & advertised to all nodes.
- Every miner node collects new transactions into a block.
- Nodes look for proof of work.
- When a node finds a proof-of-work, it broadcasts the block to all nodes.
- Nodes accept the block only if all transactions in it are valid and not already spent.
- Nodes accept the block by working on creating the next block in the chain, using the hash of the accepted block as the previous hash.
Bitcoin nodes

**Bitcoin node: main functions**

- **Wallet**: Full Blockchain
- **Miner**: Routing Node
- **Network**: Full Blockchain
- **Reference Client (Bitcoin Core)**: Contains a Wallet, Miner, full Blockchain database, and Network routing node on the bitcoin P2P network.
- **Full Block Chain Node**: Contains a full Blockchain database, and Network routing node on the bitcoin P2P network.
- **Solo Miner**: Contains a mining function with a full copy of the blockchain and a bitcoin P2P network routing node.
- **Lightweight (SPV) wallet**: Contains a Wallet and a Network node on the bitcoin P2P protocol, without a blockchain.
- **Pool Protocol Servers**: Gateway routers connecting the bitcoin P2P network to nodes running other protocols such as pool mining nodes or Stratum nodes.
- **Mining Nodes**: Contain a mining function, without a blockchain, with the Stratum protocol node (S) or other pool (P) mining protocol node.
- **Lightweight (SPV) Stratum wallet**: Contains a Wallet and a Network node on the Stratum protocol, without a blockchain.

http://chimera.labs.oreilly.com/books/1234000001802/ch06.html
The Blockchain is the global decentralized ledger which is the overall technology platform. The Blockchain is shared by all nodes & is updated by the miners. The BC maintains an ordered and timestamped ledger of all transactions. Cryptography ensures the constant integrity of the Blockchain.

The Blockchain explorer and other tools provide a way to explore the contents of different blocks and to query & search them.
GLOBAL BITCOIN NODES DISTRIBUTION
Reachable nodes as of Tue Nov 28 2017
18:55:10 GMT+0100 (CET).

11311 NODES
24-hour charts »

Top 10 countries with their respective number of reachable nodes are as follow.

<table>
<thead>
<tr>
<th>RANK</th>
<th>COUNTRY</th>
<th>NODES</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>United States</td>
<td>3202</td>
</tr>
<tr>
<td>2</td>
<td>Germany</td>
<td>1875</td>
</tr>
<tr>
<td>3</td>
<td>France</td>
<td>767</td>
</tr>
<tr>
<td>4</td>
<td>China</td>
<td>687</td>
</tr>
<tr>
<td>5</td>
<td>Netherlands</td>
<td>537</td>
</tr>
<tr>
<td>6</td>
<td>Canada</td>
<td>475</td>
</tr>
<tr>
<td>7</td>
<td>United Kingdom</td>
<td>446</td>
</tr>
<tr>
<td>8</td>
<td>n/a</td>
<td>388</td>
</tr>
<tr>
<td>9</td>
<td>Russian Federation</td>
<td>376</td>
</tr>
<tr>
<td>10</td>
<td>Singapore</td>
<td>237</td>
</tr>
</tbody>
</table>

More (100) »

Map shows concentration of reachable Bitcoin nodes found in countries around the world.
Bitcoin architecture
Anonimity

- Bitcoin addresses are not tied to people
- Transactions are not tied to people
- Transaction data is transmitted to a random subset of nodes

- However, there are some (expensive) methods to de-anonymize a user, so Bitcoin is not perfectly anonymous
Moving bitcoins between wallets

Diagram: A visual representation of moving bitcoins between wallets, including a flow from My PC to My iPhone and then to Coinbase Server, with currency values of $13.27 USD and $1.00 USD.
Bitcoin

https://bulldozer00.com/2015/10/25/bitcoin-in-uml/
The Blockchain is the global decentralized ledger which is the overall technology platform. The Blockchain is shared by all nodes & is updated by the miners. The BC maintains an ordered and timestamped ledger of all transactions. Cryptography ensures the constant integrity of the Blockchain.

This layer provides the given Currency's (e.g. Bitcoin) p2p protocol along with the consensus rules & APIs that describe the semantics of the currency to the Blockchain layer.

The Blockchain explorer and other tools provide a way to explore the contents of different blocks and to query & search them.

- Provides Shared Ledger that stores "chains of blocks"
- Stores all processed transactions in chronological order
  - Organized group of Miners create
Blockchain
Implementing bitcoins: blockchain

- Each bitcoin (BTC) node retains a copy of the global, publicly shared blockchain.
- The Blockchain has 380K+ Blocks.
- Each Block has one or more validated BTC Transactions embedded within it.
- Via the interface facilities provided by a BTC Node, a User composes a Transaction and submits it to the network for validation and execution.
- Each instance of a BTC Transaction contains a source address, destination address, the BTC amount to be transacted, and the source address owner’s signature.
Bitcoin vs Ethereum

- Since the sw infrastructure is open source, it is easy to develop new cryptocurrencies.
- The difference between Bitcoin and Ethereum is that Bitcoin is a currency, whereas Ethereum is a ledger technology.
- Both Bitcoin and Ethereum operate on the “blockchain”, however Ethereum is more robust.
- Ethereum supports the building of decentralized applications – *smart contracts*
Smart contracts are agents

- A smart contract is a set of promises, specified in digital form, including protocols within which the parties perform on these promises.
- A smart contract is a general purpose computation that takes place on a blockchain.
- The bitcoin is limited to the currency use case; ethereum replaces bitcoin's restrictive language (a scripting language of a hundred scripts) and replaces allowing developers to write their own programs.
- Smart contracts are similar to autonomous agents.
Conclusions

- P2P networks are quite old: the Internet is a P2P
- Several new applications are implemented as p2p
- Blockchains are a powerful architecture for innovative financial applications
Self test

- Which architectural drives support P2P applications?
- What is an overlay network?
- What is the relationship between p2p and C/S?
- What is a hybrid p2p system?
- What is a blockchain?
References

- Raval, *Decentralized applications*, O’Reilly 2016
- Grolimund, A Pattern Language for Overlay Networks in Peer-to-Peer Systems, EuroPloP 2006
- Ripeanu, Peer-to-peer architecture case study: Gnutella network, 2001
- Wang, Skype VoIP service-architecture and comparison, 2005
- Amoretti e Zanichelli, P2P-PL: A Pattern Language to Design Efficient and Robust Peer-to-Peer Systems, 2016
Useful resources

- http://www.vamsitalkstech.com/?cat=2
- https://www.theverge.com/2013/12/19/5183356/how-to-steal-bitcoin-in-three-easy-steps
Questions?