Peer-to-peer Architectural Styles

& BIT TORRENT

Leonardo Serilli, Gabriele Colapelle, Alessandro d'Orazio, Pietro Ciammaricone

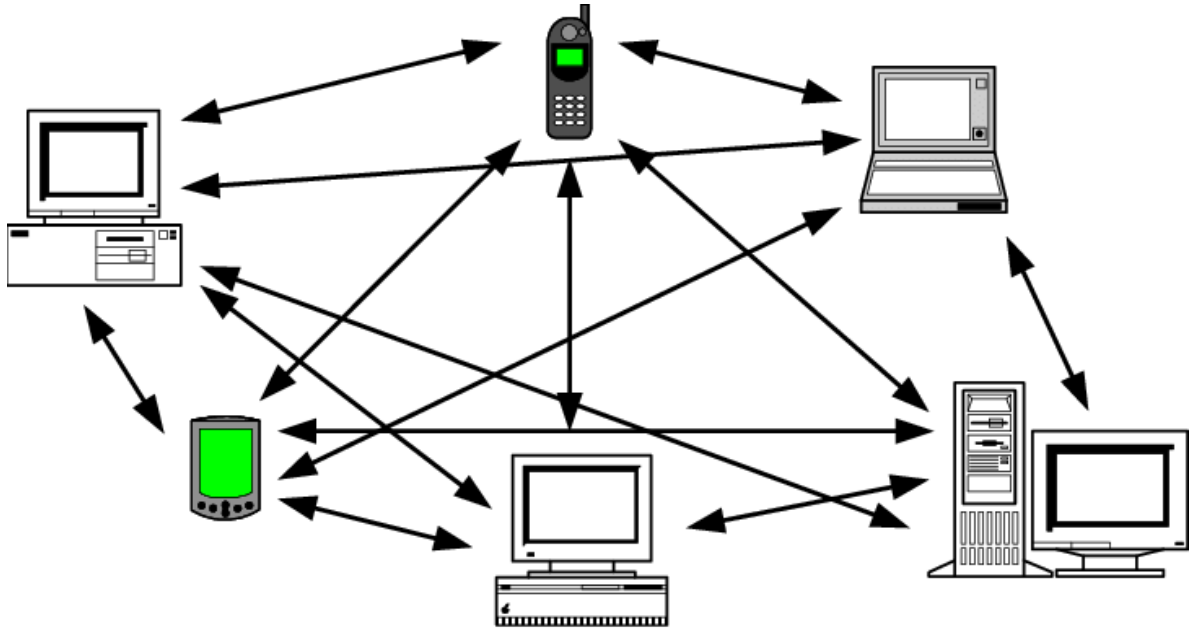
Summary

Introduction

- In the **peer-to-peer** architectural style, **components** directly interact as peers by exchanging services.

- Peer-to-peer communication as **request/reply interaction** without the asymmetry of client-server style.

- **Connectors** involve complex **bidirectional protocols** of interaction, reflecting the two-way communication between peer-to-peer components.



Peer-to-peer style

Application Context

- Distributed computational entities need to cooperate and collaborate:
  - provide a service to a distributed community of users
- Considered **equally important** in terms of:
  - initiating an interaction
  - provides own resources
- Large systems which requires **Scalability**, so without loss in:
  - **Performance**
  - **Availability**

Sources: [1], [2]

Peer-to-peer style Faced Problem

- How can a set of “equal” distributed computational entities be connected to each other via a common protocol?

- How can they organize and share their services with high availability and scalability?

Sources: [1]

Peer-to-peer style
Adopted Solution

Components

- **Peer**: indipendent component runnining on a node of the network
- **Super Peer**: special peer components can provide routing, indexing, and peer search capability.

Connectors

- **Reply/Request** connectors: connect to the peer network, search for other peers, and invoke services from other peers

Sources: [1]

Peer-to-peer style

Adopted Solution

All peer components are **equals**

Connectors in peer-to-peer systems involve **bidirectional interactions**

Peers first connect to the peer-to-peer network on which they discover other **neighbors peers**

Any peer component can interact with any other by requesting or offering his services, so each is both a client and a server.

Sources: [1]

Peer-to-peer style

Advantages

Scalability

- Peers can be **added and removed** from the network with no significant impact.
- Fexibility for deploying the system across a **highly distributed platform.**

Availability

- No peer can be **critical for the health** of the system.
- Multiple peers can offer the same services.
- If one of these peers becomes unavailable, the others can still provide the services to complete the task.

Sources: [1]

Peer-to-peer style

Weakness

Managing of services

- Managing **security, data consistency, data/service availability, backup, and recovery** are all more complex.

System's size

- **Small peer-to-peer systems** may not be able to consistently achieve quality goals such as performance and availability.

Sources: [1]

Real world P2P architectures

BitTorrent

- Rather than downloading a file from a single source server, the BitTorrent protocol allows users to join a **swarm** of hosts to **upload to/download** from each other **simultaneously.**

- This **lower bandwidth usage** also helps prevent large spikes in internet traffic in a given area, keeping **internet speeds higher for all users** in general, regardless of whether or not they use the BitTorrent protocol.

Real world P2P architectures

BitTorrent

The first uploader acted as a **Seed**, and downloaders would initially connect as Peers

Those who wish to download the file connect to **Tracker** which had a list of the IP addresses of other seeds and peers in the swarm.

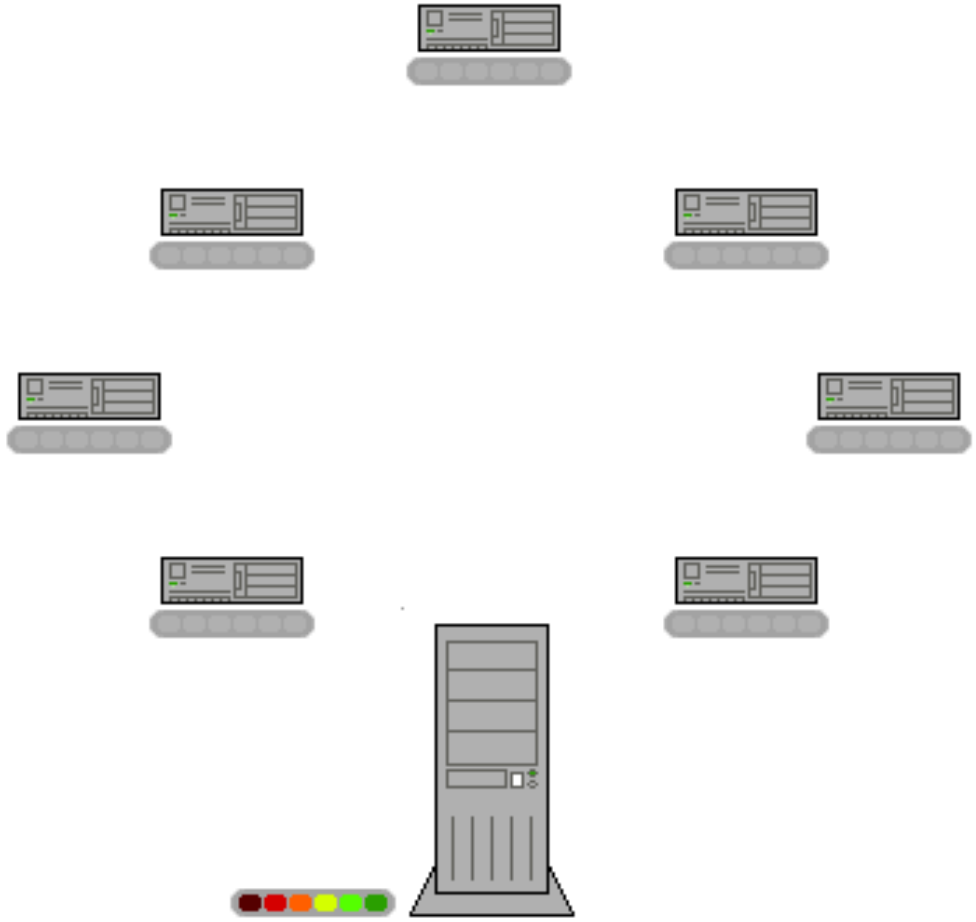
Once a peer completed a download of the complete file, it could in turn function as a seed.

Sources: [3]

Real world P2P architectures

BitTorrent

- The file is divided into segments of equal size called **pieces.**
- Peer which receives a new piece of the file becomes a **seeds**
- The **torrent descriptor (.torrent)** is a file which contains the **description of every piece** of the file and the associated **hash.**
- **Hybrid architecture**: Descriptors also contains the **Traker's URL** wich is a **server** in which are located all locations of pieces.



Sources: [3]

BitTorrent

Advantages and uses

Sharing:

- Distributing large files like Linux iso images.
- Distributing Software patches and updates.
- As being done by Blizzard Entertainment Inc, to distribute updates for the world of Warcraft
- Distributing popular files which have high traffic for relatively short periods

Performance

- Unlike traditional server/client downloads, high traffic leads to more efficient file sharing via BitTorrent.

Sources: [3]

BitTorrent

Disadvantages and Security issues

Security issues:

- The IP of all peers and info of files they are downloading are publicly available on trackers
- The tracker is a critical component and if it fails it can disrupt the distribution of all the files it has tracking.

Disadvantages:

- An easy distribution method for pirated/illegal content
- Cannot modify/update the file to newer versions once the torrent has been distributed
- Leeches who download more than they share. As BitTorrent is a collaborative distributed platform, there is a section of the community that wants solutions to punish and discourage such behaviour.

Sources: [3]

Bibliography

- [1] Len Bass, Paul Clements, Rick kazman. Software Architecture in Practice.
- [2] Amrit Tiwana. Platform Ecosystem.
- [3] Tarun Vellishetty. Understanding bit Torrent Protocol, https://www.beautifulcode.co/blog/58-understanding-bittorrent-protocol
- [4]Peter Backx, Tim Wauter, Bart DHoedt, Piet Deemester. A comparison of peer-to-peer architectures.