
🔄 Peer-to-Peer (P2P) Model

---

1. What is Peer-to-Peer?

The **Peer-to-Peer (P2P) model** is a network architecture where **each computer (peer) has equal status**.

- Unlike the **Client–Server model**, there's no central server.
- Each peer can act as both a **client (requesting services)** and a **server (providing services)**.

👉 In simple words:
Every peer is both a **consumer and provider** of resources (files, computing power, data).

---

2. Characteristics

- **Decentralized**: No single controlling server.
- **Shared resources**: Peers share files, storage, processing power.
- **Scalable**: Adding more peers can increase capacity.
- **Dynamic roles**: A peer can be client in one request and server in another.

---

3. How P2P Works (Step-by-Step Example: File Sharing)

1. You want to download a file.
2. Your computer (peer) connects to multiple other peers that already have the file.
3. The file is broken into chunks.
4. Each chunk is downloaded from different peers simultaneously.
5. At the same time, your peer may upload chunks of the file to others.

👉 This makes downloading **faster and more fault-tolerant**.

---

4. Types of P2P Networks

1. **Unstructured P2P**
    - Random connections between peers.
    - Example: Early file-sharing apps like Napster.
    - Problem: Inefficient searching.
2. **Structured P2P**
    - Organized connections, often using Distributed Hash Tables (DHT).
    - Example: BitTorrent, Kademlia.
    - Efficient lookup and file sharing.
3. **Hybrid P2P**

    - Uses both central servers (for indexing/coordination) + P2P sharing.
    - Example: Skype (before Microsoft moved it to cloud servers).

---

5. Advantages of P2P

✅ **No single point of failure** → Network doesn't rely on one server.
✅ **Scalability** → More peers = more resources.
✅ **Cost-effective** → No central infrastructure needed.
✅ **Efficient file sharing** → Parallel downloads from multiple peers.

---

6. Disadvantages of P2P

⚠️ **Security risks** → No central authority, so malware can spread.
⚠️ **Data consistency** → Hard to ensure latest/valid version.
⚠️ **Resource usage** → High CPU/bandwidth usage on peers.
⚠️ **Reliability** → If peers go offline, resources may disappear.

---

7. Real-World Examples

- **File Sharing**:
    - BitTorrent, LimeWire, eMule.
- **Communication**:
    - Early Skype (peer-to-peer calling).

- **Blockchain / Cryptocurrencies**:
    - Bitcoin, Ethereum → P2P ledger synchronization.
- **Distributed Storage/Compute**:
    - IPFS (InterPlanetary File System).
    - SETI@home, Folding@home (scientific distributed computing).

---

8. Diagram (Conceptual)

Client–Server (Centralized):

```
 Clients ---> [ Server ] ---> Clients
```

Peer-to-Peer (Decentralized):

```
 [Peer] <--> [Peer] <--> [Peer] <--> [Peer]
     \______________________________/
```

---

9. P2P vs Client–Server (Quick Comparison)

Feature	Client–Server	Peer-to-Peer (P2P)
Control	Centralized (server)	Decentralized (peers)
Scalability	Limited by server capacity	High (more peers = more power)

Reliability	Server crash = failure	Many peers = redundancy
Security	Easier to enforce	Harder to control
Examples	Web apps, banking, APIs	Torrents, blockchain, VoIP

✅ **In short**:
The **Peer-to-Peer model** is decentralized, scalable, and fault-tolerant. It’s excellent for file sharing and blockchain systems but harder to secure and manage than the Client–Server model.
