Lecture 4

15th January 2021

Team 5:

SHANU GANDHI	2019201014
SRITEJA SUGOOR	20171032
TUSHAR JOSHI	2018101013

Presentation Scribe
Comparison between Client Server and P2P Architecture

1. **Introduction to Architecture Styles:** Before going into the details about client server and p2p architecture and the difference between them, lets first understand what an architecture style means.

"An architectural style is a general, reusable solution to a commonly occurring problem in software architecture within a given context. Architectural styles are often documented as software design patterns."

Or in simple words, "Architectural Styles is a pattern for subsystem decomposition"

IEEE defines architectural style as "the process of defining a collection of hardware and software components and their interfaces to establish the framework for the development of a computer system."

2. Types of architectural Styles:

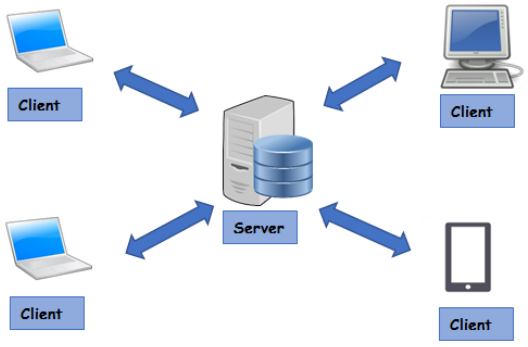
There are many recognized architectural patterns and styles, among them:

- Client/Server
- Peer-To-Peer
- Repository

- Model/View/Controller
- Three-tier, Four-tier
- Pipes and Filters

We will only cover client server architecture style and peer-to-peer architecture style in this scribe.

3. Client Server Architecture Style:



Lets first discuss the client -server architecture style. The Client-server architecture is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters called clients. In the client-server architecture, when the client computer sends a request for data to the server through the internet, the server accepts the requested process and delivers the data packets requested back to the client. Clients do not share any of their resources. Examples of Client-Server Model are Email, World Wide Web, etc

3.1 Some more insights about client server architecture is as follows:

- A client-server network involves multiple clients, or workstations, connecting to at least one central server.

- Most data and applications are installed on the server. When clients need access to these resources, they access them from the server.

- The clients are allowed to function as workstations without sharing any resources

- The response in general is immediate

- End users interact only with the client
-

3.2 Some details about Client and Server in the architecture:

- **CLIENT**
  - Clients have an active role and initiate a communication session by sending requests to servers
  - Clients can communicate with servers only; they cannot see each other.
- **SERVER**
  - Servers have a passive role and respond to their clients by acting on each request and returning results.
  - One server generally supports numerous clients

3.3 Advantages of Client Server Model:

- It is easier to upgrade software applications and files because they are held on one single computer.
- Security is enhanced on a client server network because the security is handled by the server.

3.4 Disadvantages of client server model:

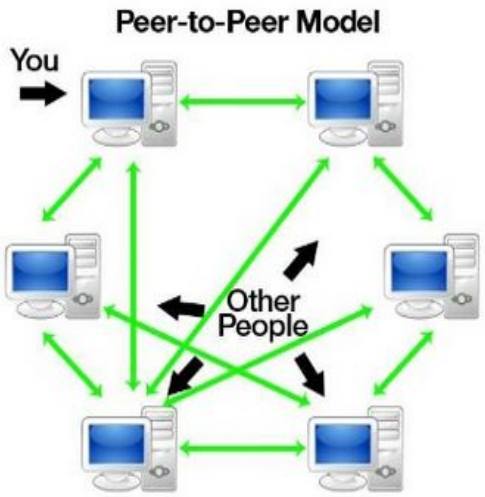
- If all the clients simultaneously request data from the server, it may get overloaded. This may lead to congestion in the network.
- If the server fails for any reason, then none of the requests of the clients can be fulfilled. This leads to failure of the client server network.

3.5 Applications of Client Server Model:

- Often used in the design of database systems
  - Front-end: User application (client)
  - Back end: Database access and manipulation (server)
- Functions performed by client:
  - Input from the user (Customized user interface)
  - Front-end processing of input data

- Functions performed by the database server:
    - Centralized data management
    - Data integrity and database consistency
    - Database security

4. Peer to Peer Architecture:



Peer-to-peer architecture (P2P architecture) is a commonly used computer networking architecture in which each workstation, or node, has the same capabilities and responsibilities. It is often compared and contrasted to the classic client/server architecture, in which some computers are dedicated to serving others. Here there does not exist some specific node which is called server or client , here each independent entity is called peer, which act as a server as well as client .

4.1 Some more features about Peer-to Peer architecture:

- Peer to Peer is a type of architecture in which nodes are interconnected with each other and share resources with each other without a central controlling server.
- Unlike client-server architecture, there is no central server for processing requests in a P2P architecture. The peers directly interact with one another without the requirement of a central server.

4.2 Advantages of Peer to Peer Architecture:

- No Central Dependency on server as it exists in case of client server architecture.

- Hence, it is more Reliable as **a single point of failure(server in case of client server arch.)** does not exist in this case.
- Here the relative cost is less as no expensive tools are required to set up a high end server.
- It is Highly Scalable as peers can simply be added in the system without having to worry much about load on a single server.

4.3 Disadvantages of Peer To Peer Architecture:

- Network cannot be administered as there is no central authority like a server.
- Security is compromised as some peers can act maliciously and no one can know as there is no central authority to check it.
- Backup is difficult as data is distributed across clients so a snapshot of the whole system with all peers needs to be taken if one wants to backup the system.

4.4 Applications of Peer to peer Architecture:

- The first adopted usage of P2P applications was Instant messengers.
- The peer-to-peer file sharing era started with Napster and continued with much more powerful applications such as **Torrent**.
- High performance computing is important for scientific research or for large companies. Sharing of resources like computation power, network bandwidth, and disk space will benefit from P2P.
- Another major usage of P2P is IP telephony. IP telephony revolutionaries the way we use the internet, enabling us to call anywhere in the world for free using our computers.

5. Differences between Client Server and Peer to Peer

a

Attribute	**Client Server**	**Peer To Peer**
Participants	Multiple clients and single server.	Multiple Peers
Active role (requester)	Client	Any Peer

Passive Role(provider)	Server	Any peer
Interaction	Clients with Server	Arbitrary
Data stored	Centralised Server	Distributed among Peers
Scalability	Hard to achieve	Relatively Easy
Cost	Relatively More Cost	Relatively Less Costlier
Reliability	Relatively Low	Relatively High
Security	Relatively High	Relatively Less

Questionnaire

Q1.How is the client server more secure than peer to peer?

Ans. In the client server there is a central authority that can monitor activities of each client, while in peer to peer each node has to take care of security by itself. Peer-to-peer networks are typically less secure than a client-server network because security is handled by the individual computers, not on the network as a whole.

Q2. Can Client Server have multiple servers too ?

Yeah there is a concept of **Multi-client/Multi-server split architecture**
That is there can be multiple servers in client server architecture in order to increase reliability and availability of the system. In that case scheduling needs to be done in order to evenly distribute load between servers and also it can handle failures occurring in one server.You can refer more about it in the paper(https://ieeexplore.ieee.org/document/6496712)

Q3.Client-server models are easier to maintain than p2p networks comment on this?

Yeah Client server models are easier to maintain than p2p. One major issue with maintaining a P2P system is updating the software on all the nodes of the network. This could be an automatic update, or a manual one

that is performed by the user. For the latter it cannot be assumed that all users would perform the upgrade and so the system may have to be able to cope with different versions of itself. Furthermore, the lack of updating the system could be particularly important should the update patch a security flaw within the system

---

Class Scribe

- **Message Ordering**
  - **FIFO** - If p1 is sending message to p2 , than message that p1 send first must reach p2 first
  - **CAUSAL** - When multiple messages reach a particular process , the receive order will be the same as send order. If two different processes send messages to a particular process , the message that was sent first will reach first.
  - **NON FIFO** - In non fifo the send order does not matter , messages can be received in any order.

Question : - Is it a strict requirement that the number of processes be greater than two in causal , can't fifo be seen as a special case of causal?

=> Yes, Fifo is a special case of causal. If only two processes are involved then it is FIFO , we can talk about causal only if more than two processes are involved otherwise it is equivalent to FIFO.

- **Matrix time**
  For a process Pi having matrix time M -
  - Entry M[i,i] is equivalent to the scaler clock of process Pi.
  - Row vector Mi , termed as principle vector for process Pi is equivalent to the vector clock of process Pi.

- Other entries Mjk , are what process Pi knows about what process Pj knows about process Pk.

- **Matrix time update after receiving a matrix T from another process Pj.**
  - M[i,i] = M[i,i]+1
  - M[i,k]= max(M[i,k] , T[j,k]) { for all k except i }
  - M[l,k]= max(M[l,k] , T[l,k]) { for all l except i }

Question :- Will the principle vector always be greater than other non principal vectors in the time matrix ?

- Pi has the most updated value about his own state in M[i,i] so it is more recent than any other entry in column i.
- For other entries in principle vectors , Any update in any other cell is also reflected in the principle vector.

  So the principle vector is most updated and thus will be greater than other non principle vectors.

HW

- We know vector time tells what we know about other processes. Matrix time tells what a process knows about what another process knows about a third process. Can we extend it further, can we have 3D or 4D time matrix.

- Consider causal message ordering.Assume vector clocks being used. Consider an updation event at site D which was propagated to all sites at time Tsu; which A is aware of.
  - A wants to know if B got the update by looking at Bs vector sent to A. Can A know?
  - Now A wants to know if C knows about the update by looking at B's vector. That is A is interested in Vc[D] but he needs to figure this out looking at B's vector in the msg from B to A. Assume that C has sent B a msg recently where VB[C] >= Tsu.Then is

it true that C has received the update? Can A figure out if C has received the broadcast?

- If the messages passed used matrix clocks instead of vector clocks then can A have answers to both problems above?

Global States and Snapshots

Problem:

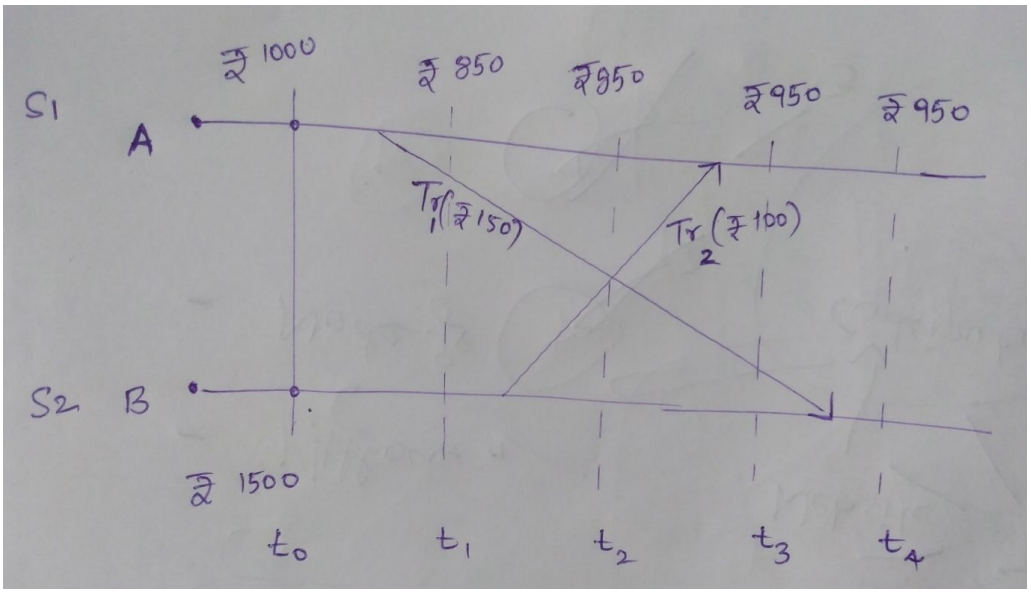


Figure 1

Consider the above distributed system. At time t_0, A has Rs. 1000 and B has Rs. 1500. Tr_i(Rs. x) denotes a transaction of Rs. x in the respective direction. Amount in transit (in Rs.) for the above distributed system at different times is shown in the table below.

C_{AB}	0	150	150	150	0
C_{BA}	0	0	100	0	0

So here we want to capture a global snapshot of the system at any specified time. If we have physical time and all the clocks of the processes

are synchronised, then the Global snapshot of this system is a union of local snapshot of the processes at that time and that of the transits.
But what if we don't have access to a physical clock, but only to a logical clock. In that case, how can we measure the global snapshot of the system?

The global snapshot that we take using logical time may not be equivalent to exact physical time. But we need to make sure that it is a consistent snapshot.

Definition:
Let say a distributed system with n processes where

GS denotes the global state of the distributed system.
P_i denote the i^{th} process where $i \in [0, n-1]$
LS^x_i denote the state of Process P_i at the instant x (result of sequence of all events executed by P_i upto that instant x)
m_{ij} denote the message from P_i to P_j
C_{ij} denotes the channel from P_i to P_j
SC_{ij} contains the messages in the channel C_{ij}

$$SC_{ij} = \{ m_{ij} \mid send(m_{ij}) \in LS_i \text{ and } rec(m_{ij}) \notin LS_j \}$$

The above line tells that a channel C_{ij} has a message m_{ij} if the sending of that message occurs within LS_i and receive of that message hasn't happened within LS_j. That message is called a message in transit.

The global state of the system is defined as

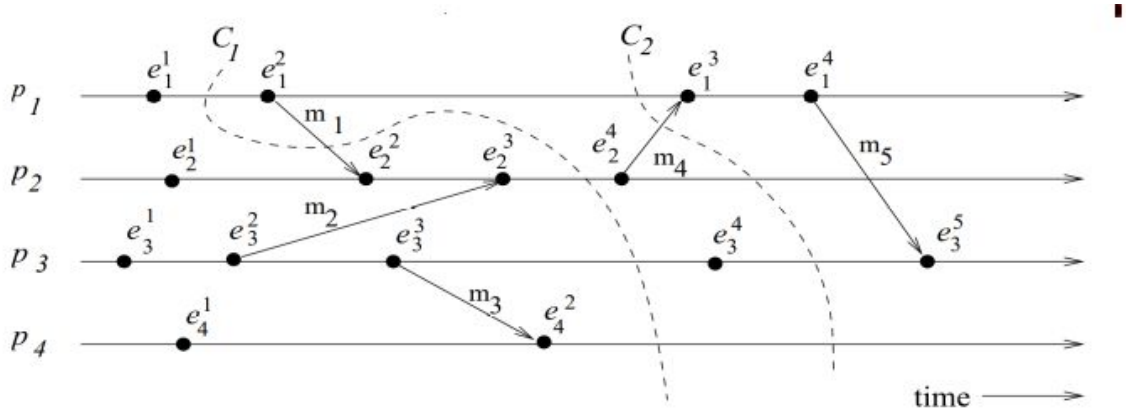
$$GS = \{ \cup_i LS_i, \cup_{i,j} SC_{ij} \}$$

The above global state is said to be consistent iff it satisfies the following two conditions:

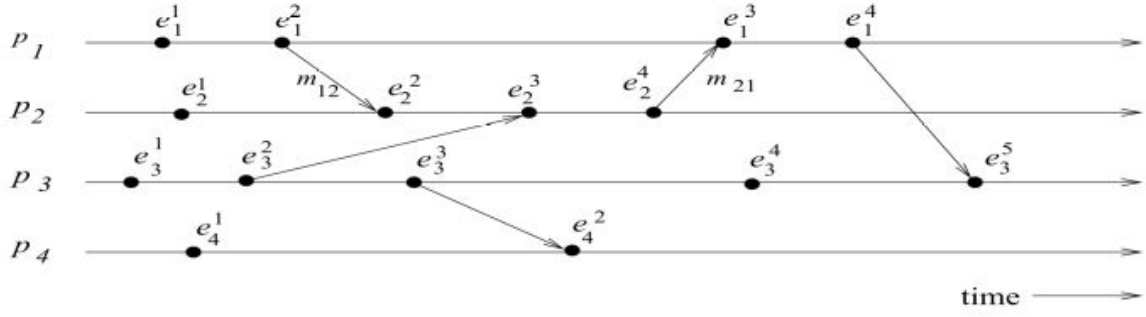
$$C_1 : send(m_{ij}) \in LS_i \Rightarrow m_{ij} \in SC_{ij} \oplus rec(m_{ij}) \in LS_j$$
$$C_2 : send(m_{ij}) \notin LS_i \Rightarrow m_{ij} \notin SC_{ij} \ \wedge \ rec(m_{ij}) \notin LS_j$$

C_1 says that if a message is sent within the local state of P_i, then either the message should be in transit or the message has been received at P_j but not both. C_2 says that if a message has not been sent by P_i then it neither should be in transit nor should it be received by P_j.

Interpretation in terms of cuts



A cut is a line joining an arbitrary point on each process line that slices the diagram into a past and a future. A global state is said to be consistent if every message received in the past of the cut was sent in the past of that cut. In the above example, C1 is inconsistent because the message m_1 is sent in the future in p_1 but received at p_2 in the future.



From the above diagram, answer the following questions:

1. $\{LS^1_1, LS^3_2, LS^3_3, LS^{24}\}$ - This is inconsistent because m_{12} has sent in the future of p_1 but received in the past of p_2.

2. $\{LS^2_1, LS^4_2, LS^4_3, LS^2_4\}$ - This is inconsistent because the message m_{21} is not accounted for in the transit.

3. {LS^2_1, LS^1_2, SC^{21}_{12}, SC^{31}_{32}, LS^3_3, LS^2_4} - This is a global consistent state.

HW

In Figure 1, if the state of the account A is recorded at time t_0 and the state of account B and that of the channels C_{12} and C_{21} are recorded at t_2, why did we land up in an inconsistent state?

Questions

Q: How do we represent a message in transit?

A: We write it as SC^{ij}_{xy} where m_{ij} is the message from state i of p_x to state j of p_y