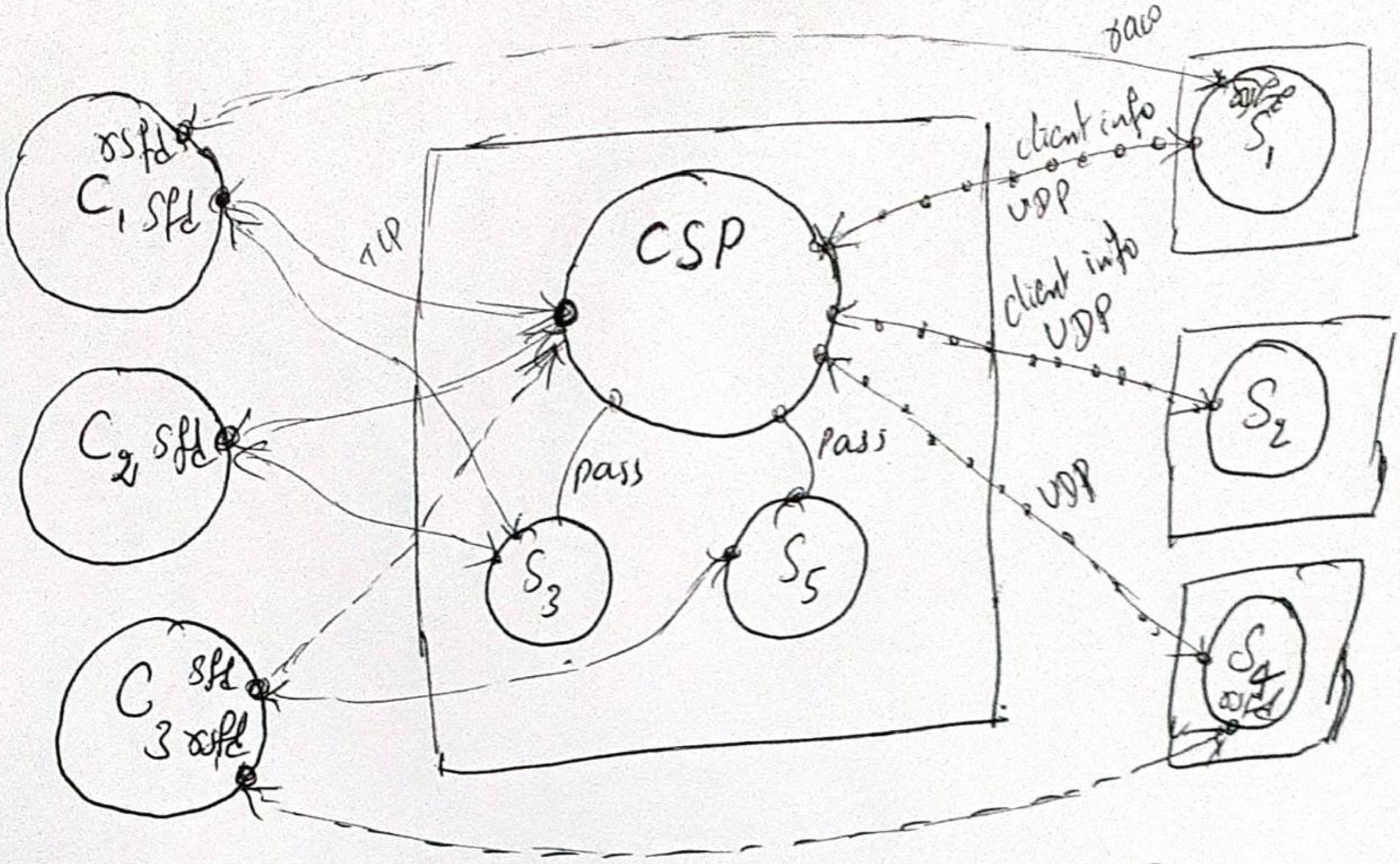
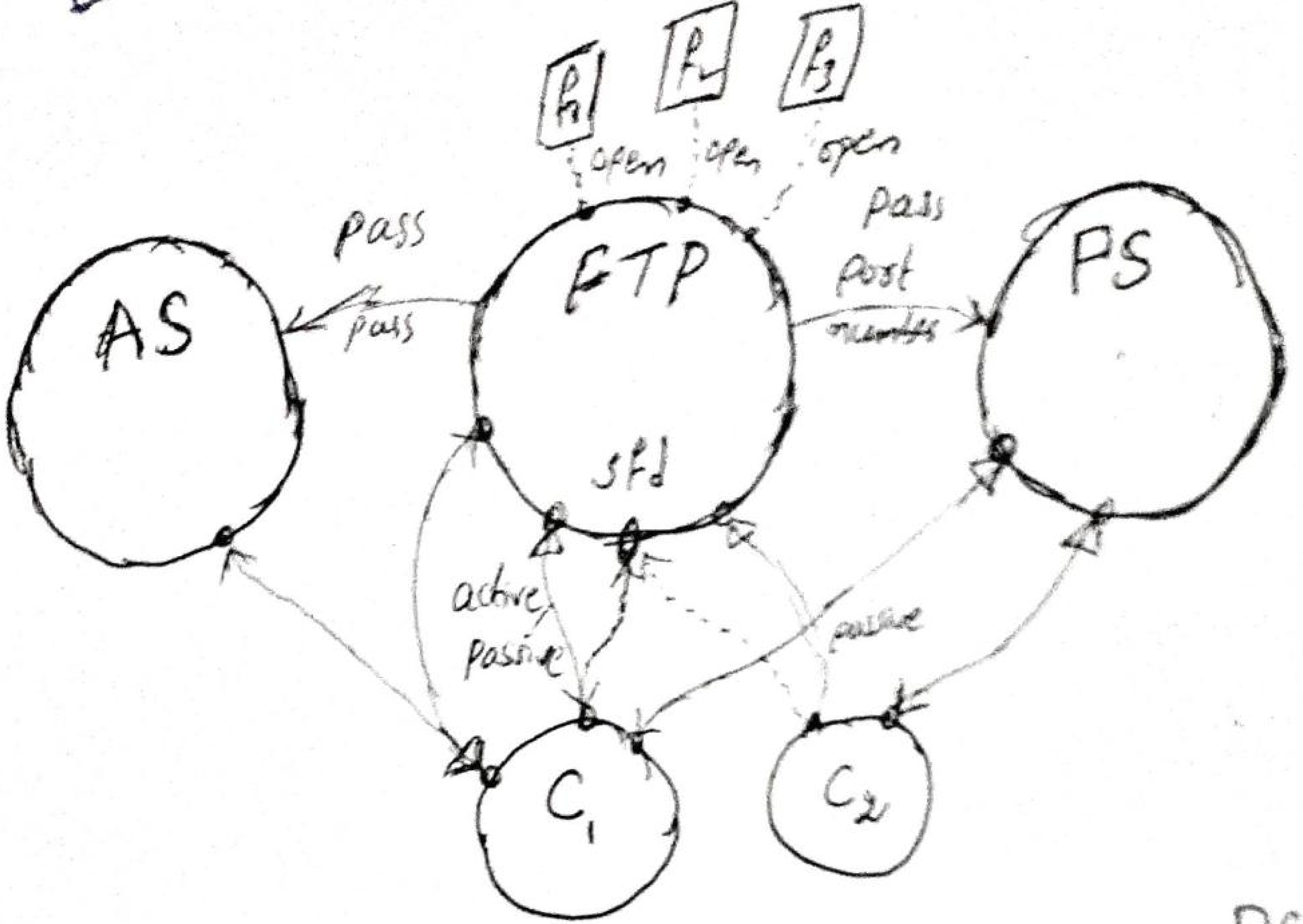
9.~~3~~ Central Service Provider _ Raw



* Clients connect to Central Service Provicer (CSP) and specifies the service number. Service servers S_1, S_2, S_4 are on another systems whereas service server processes S_3, S_5 are in the same system of CSP.
* ~~If~~ CSP accepts clients and if services are 3 or 5 it passes to S_3 or S_5. If services are 1, 2 or 4 then it ~~inf~~ sends client information to S_1, S_2 or S_4. CSP also sends a message "r" to clients if services are 1, 2, 4.
* If a client gets 'r' message from CSP, it opens a raw socket and it gets served on that raw socket by remote servers S_1, S_2 or S_3 according to its request.
* At a time a client can request many services.

~~10~~ Active/Passive FTP

$\boxed{10}$



* FTP - main Server, AS - Active Server, PS - Passive Server

* Client opens a Control connection with FTP and sends for active - a or passive - p mode. If active mode it sends message as $\boxed{\text{a port-number file name}}$

which means : on which port-number it listens, and it wants the contents of file name.

If passive mode a client sends message as

$\boxed{\text{P file-name}}$ means : it needs contents of file name.

* In 'a' mode client accepts connection from server and then gets contents of the file.

In 'p' mode it first ~~ge~~ receives port-number from server and gets connected to that port, then it starts receiving the contents of file.

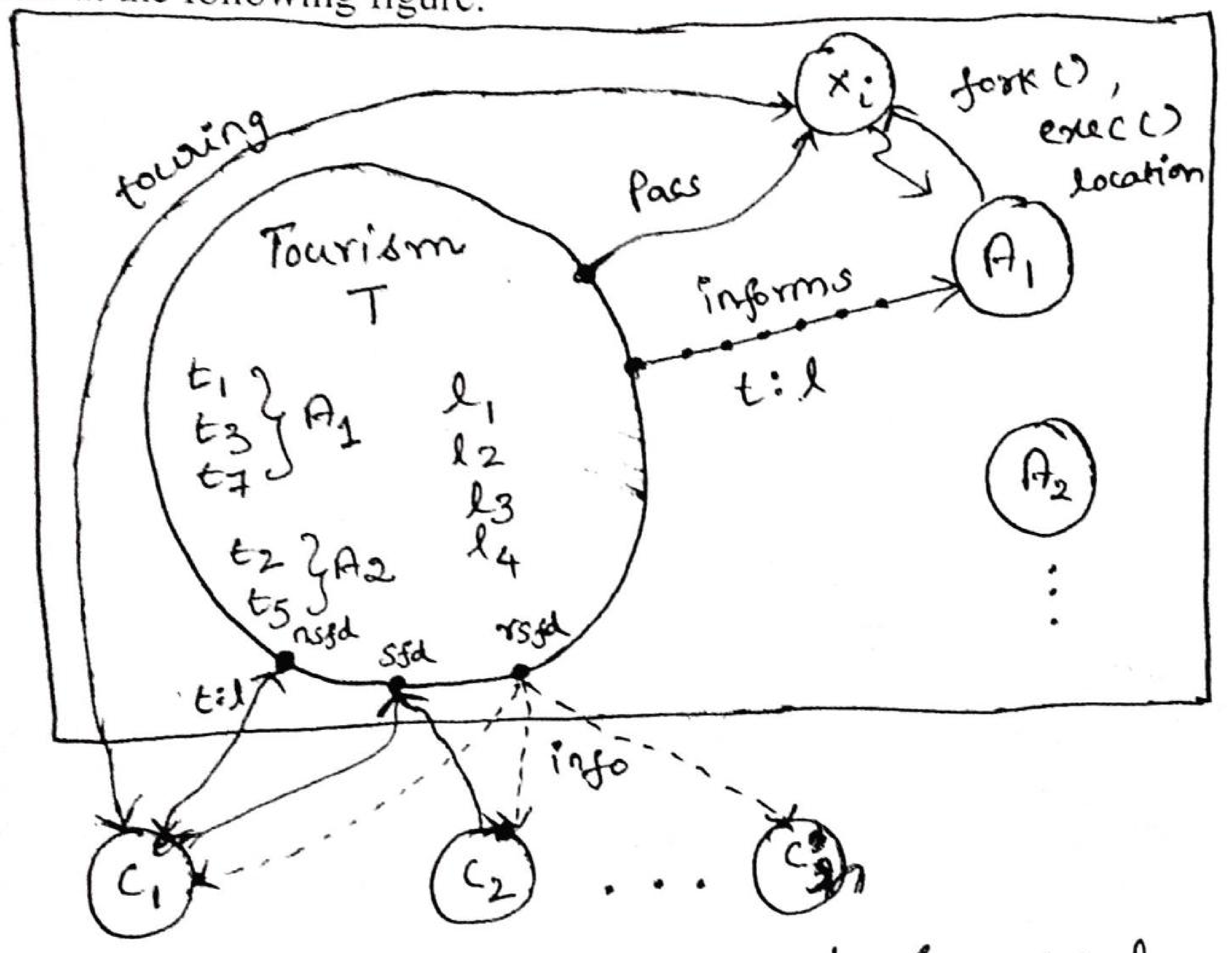
* FTP passes/sends required messages to AS, PS according to Client modes. FTP itself opens the data files.

$\boxed{11}$ *

Nitw-Tourism

(7.3)

Look at the following figure.



T – Tourism Server

A – Agents

X – Taxi

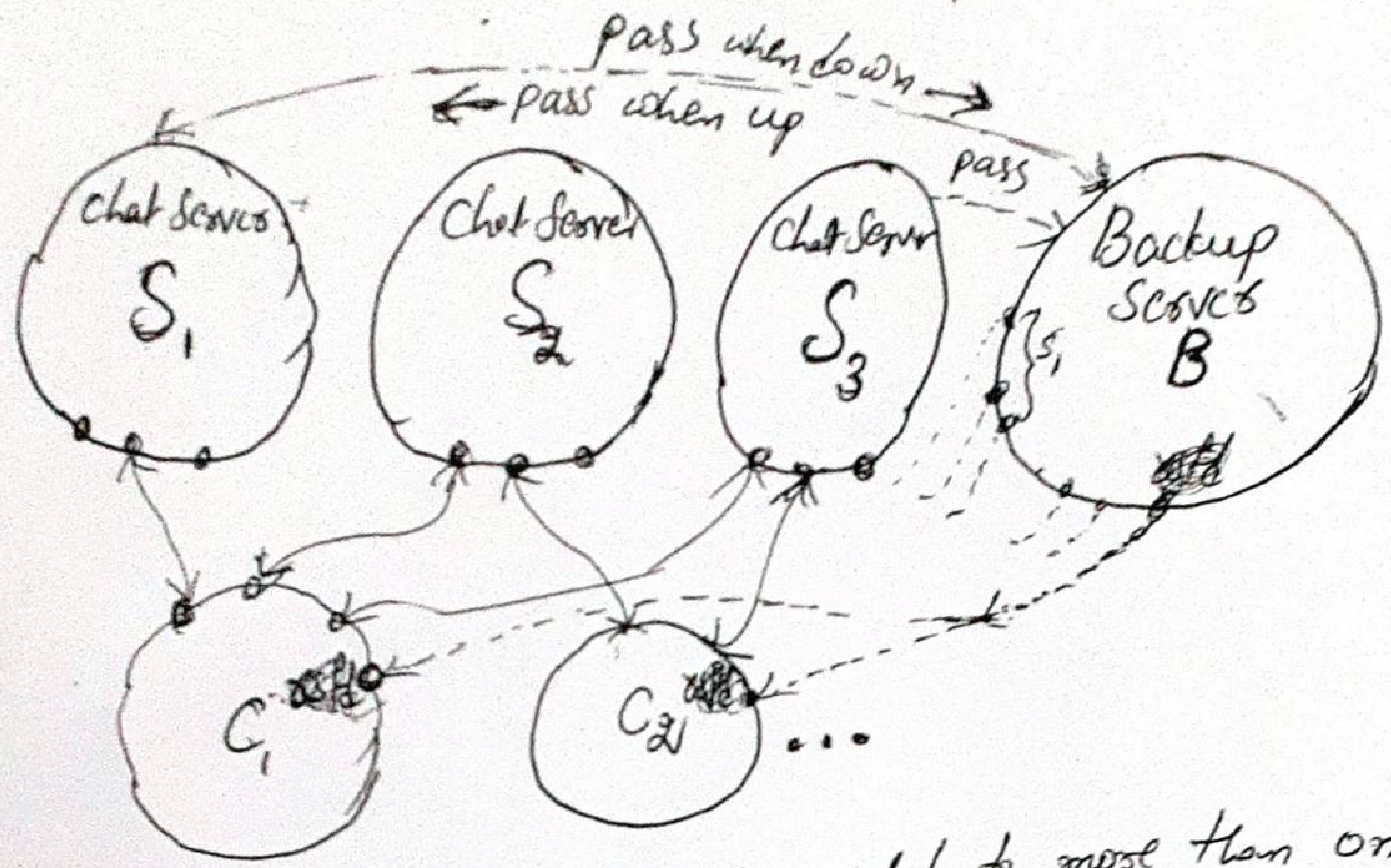
C – Customers

$t_1, t_2, t_3, t_5, t_7 \rightarrow$ Tourist places

$l_1, l_2, l_3, l_4 \rightarrow$ pick-up location for customers [not IP & Port numbers]

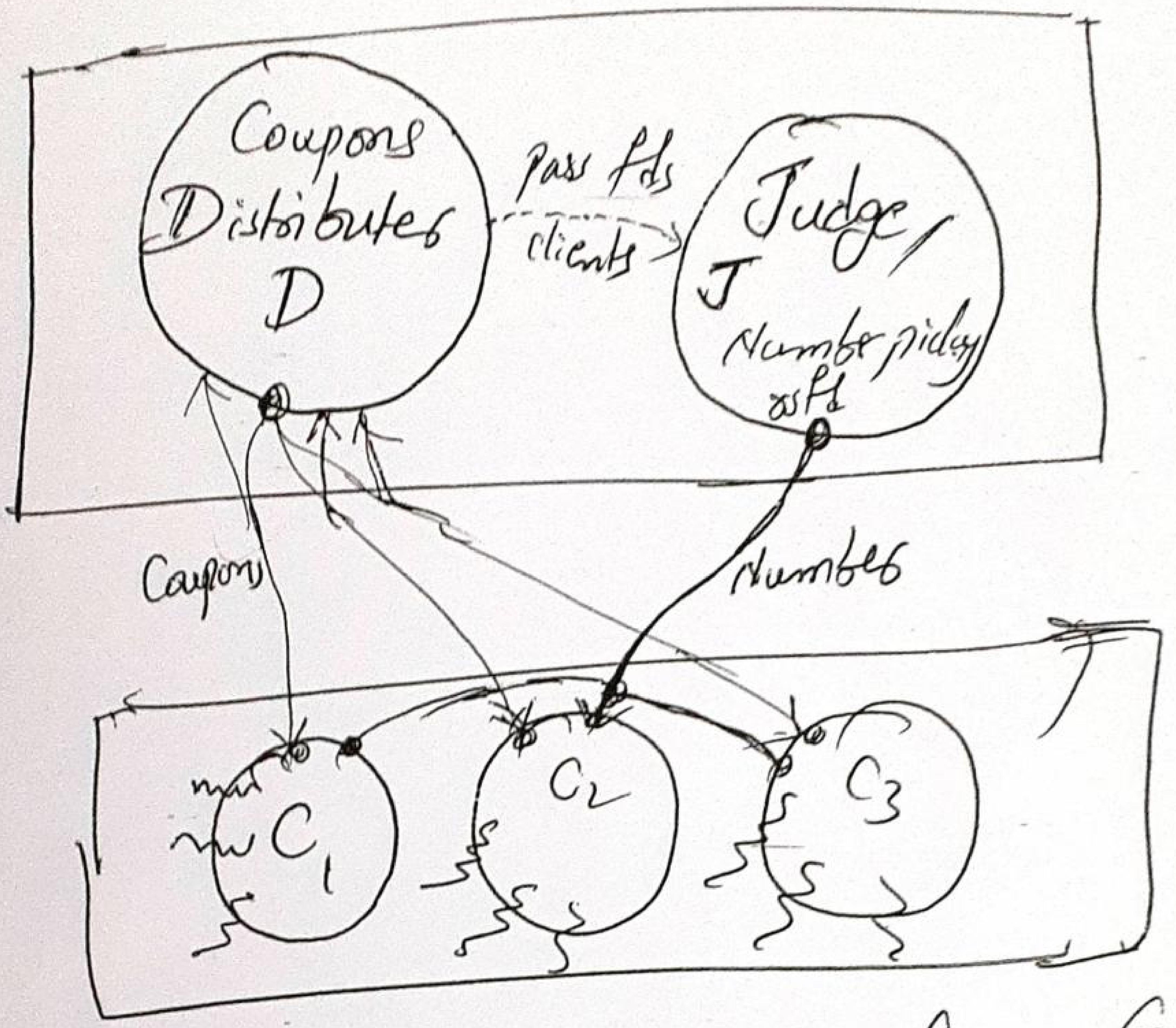
* C connects to T and requests for t_i, l_i.
* T accepts C connection and looks for A who provides t_i.
* T sends message to A_i for t_i, l_i.
* A fork(), exec() a taxi X_i by giving details of l_i.
* T passes customer to X_i.
* X_i now serves C_i.
* All clients who used service of T, gets tour packages/offers information from T, time to time.
* If any breakdown of vehicle occurs, X_i sends Signal to A_i.

$\boxed{12}$ Multichat Servers_Backup Servers



* A client can get connected to more than one chat server.
* If a chat server S_i is going to down, then it passes all its client connections to Backup Server B.
* B just informs the clients that they are getting served by it and then allows the chatting.
* More than one chat server can be down at a time.
* As soon as ~~the~~ a chat server S_i passes client chat connections to Backup Server, it notifies the clients that it has taken over the chat serving duty on behalf of that particular server S_i.
* clients can exit from a chat by sending "x".
* When a chat ~~g~~ server S_i gets ready again, the Backup server B has to send it back the latest clients ~~&~~ in that group.

13 ~~3~~ Housy - Multiple Coupons



* Customers/clients get coupons from Coupon Distributor D. Each C can get multiple coupons. Assume a Coupon consists four numbers between 0 & 99. Ex: $\boxed{9\ 41\ 18\ 22}$. All client processes are at same syst.
* After coupon distribution is over, process D sends client/customers information to Judge process J. The Judge process starts picking a random number and sends the numbers to clients site.
* C_1, C_2, C_3 compares the number within their coupons in each thread (one coupon per thread). If any coupon gets fully matched, the C_i informs to Judge J. J announces the Winner.

[illegible]