# 1. Distributed systems

Define distributed systems. Explain the different kinds of communicating entities and communication paradigms. Give an overview of different distributed system architectures.

[Coulouris et al. chapter 2]

1.A distributed system is one in which components located at networked computers communicate and coordinate their actions only by-passing messages.

2.. In some primitive environments the entities that communicate are nodes.

. In most distributed system environments, processes are supplemented by threads that are the endpoints of communication.

3.Communication paradigms:

• interprocess communication, refers to the relatively low-level support for communication between processes in distributed systems including message-passing primitives, direct access to the API offered by Internet protocols (socket programming)

• remote invocation : two-way exchange between communicating entities

Request-reply protocols: a pattern imposed on an underlying message-passing service to support client-server computing

Remote procedure calls: procedures in processes on remote computers can be called as if they are procedures in the local address space

Remote method invocation: : Remote method invocation (RMI) strongly resembles remote procedure calls but in a world of distributed objects

NOTE

communication represents a two-way relationship between a sender and a receiver with senders explicitly directing messages/invocations to the associated receivers. Receivers are also generally aware of the identity of senders, and in most cases both parties must exist at the same time.

• indirect communication(through a third entity)

• Senders do not need to know who they are sending to (space uncoupling).

• Senders and receivers do not need to exist at the same time (time uncoupling).

1.Group communication: Group communication is concerned with the delivery of messages to a set of recipients (one-to-many communication)

2.Publish-subscribe systems: A publish-subscribe system is a system where publishers publish structured events to an event service and subscribers express interest in particular events through

subscriptions which can be arbitrary patterns over the structured events. (one-to-many)

3.Message queues: is point-to-point service using the concept of a message queue as an indirection, thus achieving the desired properties of space and time uncoupling.

4.Distributed shared memory (DSM) is an abstraction used for sharing data between

computers that do not share physical memory. Processes access DSM by reads and

updates to what appears to be ordinary memory within their address space.(one-to-many)

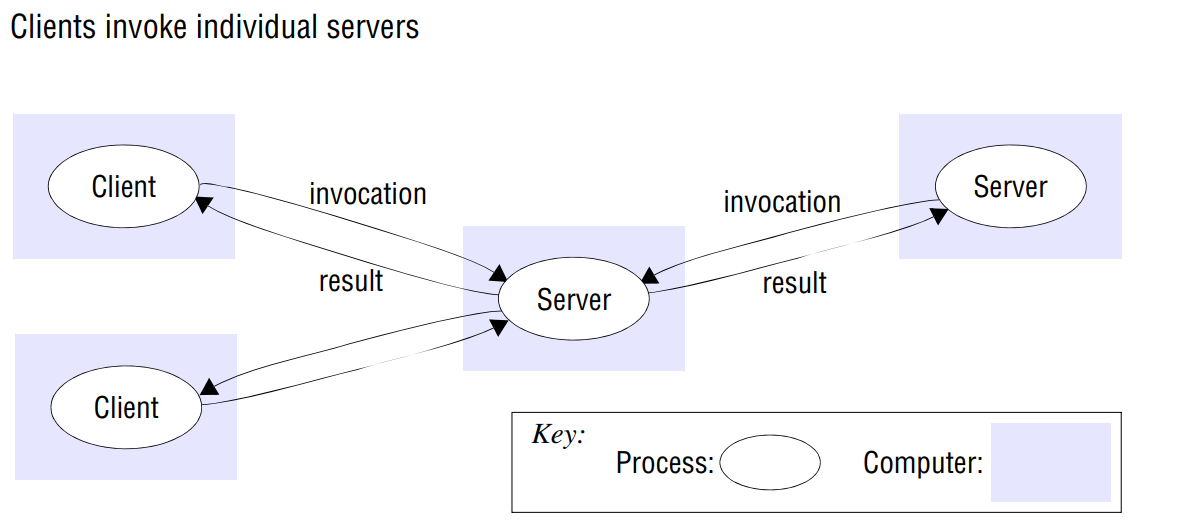
5.Tuple spaces In this approach, processes communicate indirectly by placing tuples

in a tuple space, from which other processes can read or remove them. Tuples do not

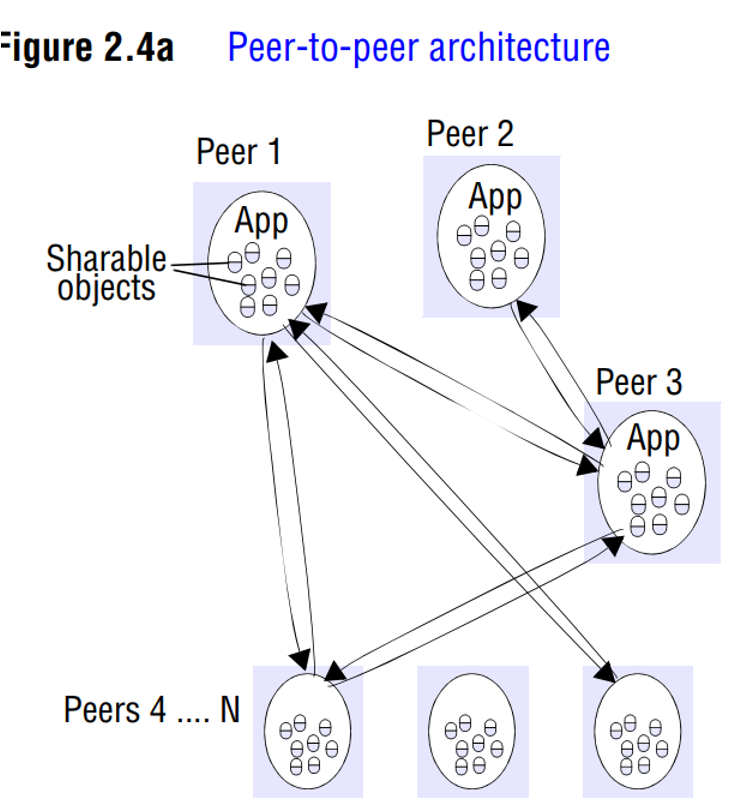
have an address but are accessed by pattern matching on content.(one-to-many) or .(one-to-one)

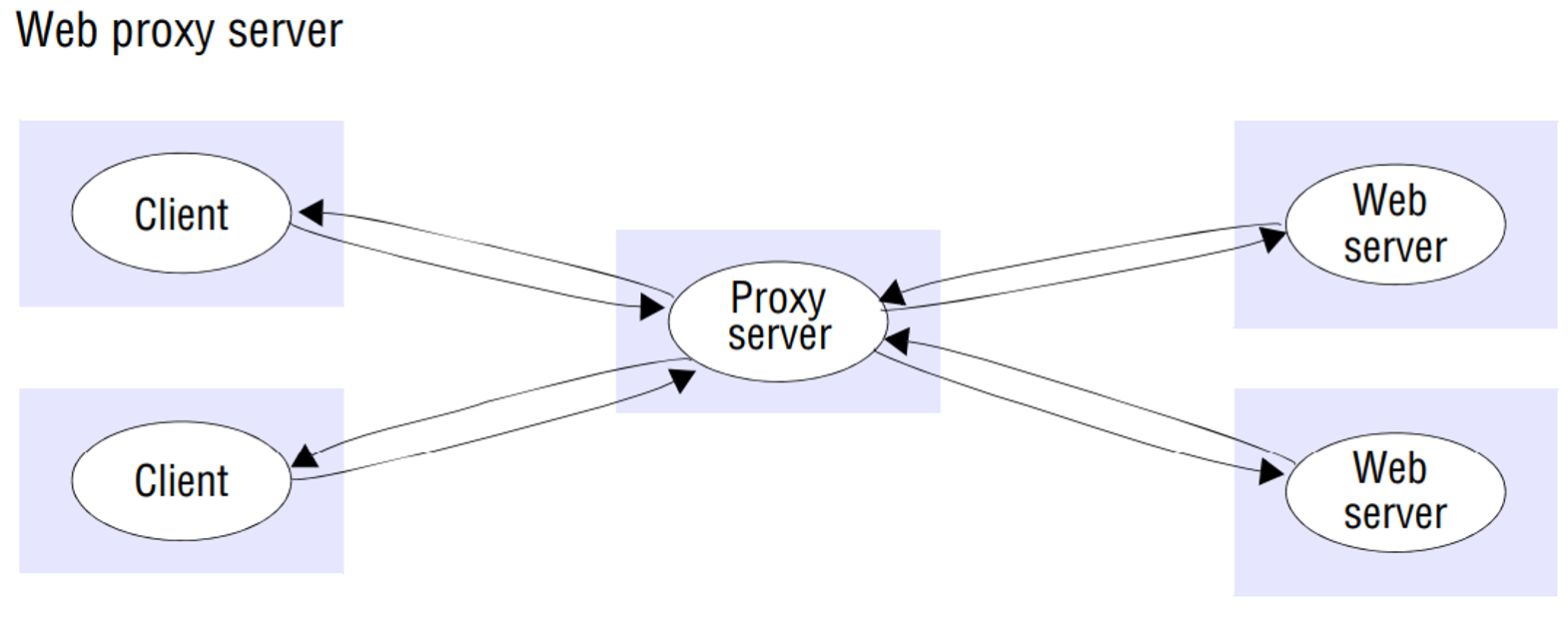
‘------------------------

distributed system architectures.

Client-server: Client processes interact with individual server processes in potentially separate host computers in order to access the shared resources that they manage (it scales poorly)

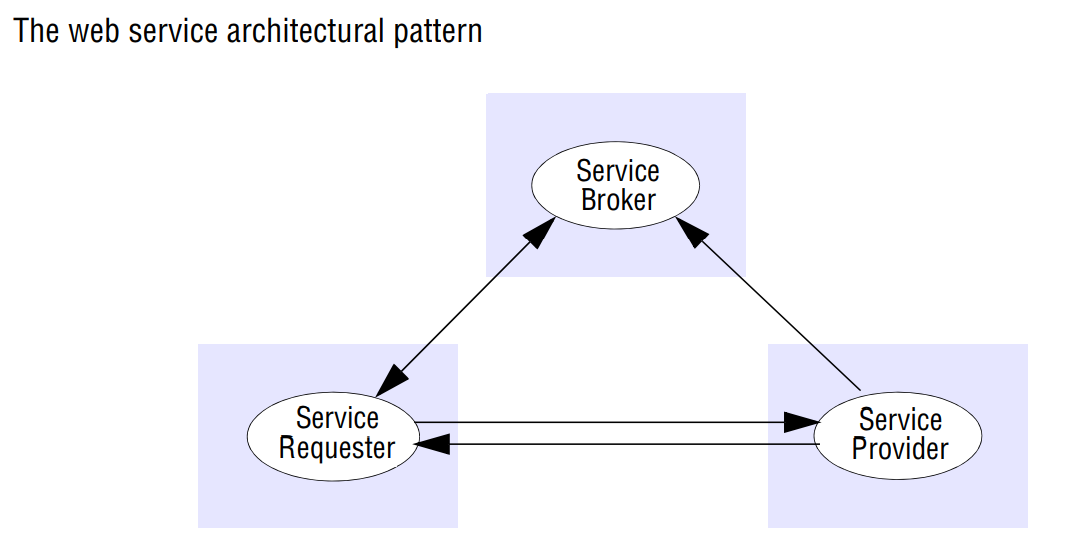
Peer-to-peer: In this architecture all of the processes involved in a task or activity play similar roles, interacting cooperatively as peers without any distinction between client and server





The proxy pattern is a commonly recurring pattern in distributed systems designed particularly to support location transparency in remote procedure calls or remote method invocation. With this

approach, a proxy is created in the local address space to represent the remote object.



• The use of brokerage this pattern consists of the trio of service provider, service requester and service broker. This brokerage pattern is replicated in many areas of distributed systems, for example with the registry in Java RMI