Dr. Vishwanath Karad
MIT WORLD PEACE UNIVERSITY | PUNE
TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

Module 2

Communication in Distributed Systems,Introduction to middleware,Fundamentals of Communication, Basic RPC Operation, Parameter Passing, Asynchronous RPC, Message Oriented Communication, Stream Oriented Communication, Multicast Communication RMI - Introduction ,Request-reply protocols,Remote Method Invocation

Communication

Inter-process communication :-

- There is a requirement for communication and synchronization between co-operating processes.

Two communication patterns that are commonly used in distributed programs:-

- Client-Server communication [request –reply]

- Group communication [same message is sent to several processes]

Protocols for Distributed Systems

- Versatile Message Transport Protocol (VMTP)

provides group communication facility,efficient Client-Server communication.

Fast Local Internet Protocol (FLIP),transparency,client-server based protocols,easy network management.

Middleware Layers

- Applications,Services
- RMI,RPC
- Request-Reply protocol
- Marshalling and External Data Representation
- UDP and TCP

Discussion Focus

- Characteristics of protocols for communication between processes to model distributed computing architecture
  - Effective means for communicating objects among processes at language level
- Representation of objects
  - providing a common interface for object references
- Protocol construction
  - Two communication patterns for distributed programming: C-S using RMI/RPC and Group communication using ‘broadcasting’

Characteristics of IPC

- synchronous and asynchronous communication.
- Message destinations.
- Reliability.
- Ordering.

Characteristics of IPC

synchronous and asynchronous communication.

- Synchronous
  - Queues at remote sites are established for message placement by clients (sender). The local process (at remote site) dequeues the message on arrival
  - If synchronous, both the sender and receiver must ‘rendezvous’ on each message, i.e., both *send* and *receive* invocations are *blocking-until*

Characteristics of IPC

synchronous and asynchronous communication.

- Asynchronous communication
  - *Send* from client is non-blocking and proceeds in parallel with local operations
  - *Receive* could be non-blocking (requiring a background buffer for when message finally arrives, with notification – using interrupts or polling) AND if blocking, perhaps, remote process needs the message, then the process must wait on it
  - Having both sync/async is advantageous, e.g., one thread of a process can do blocked-receive while other thread of same process perform non-block receive or are active – simplifies synchronization. In general non-blocking-receive is simple but complex to implement due to messages arriving out-of-order in the background buffer

Characteristics of IPC

Message destinations

- Typically: send(IP, port#, buffer) – a many-to-one (many senders to a single receiving port), except multicast, which is many-to-group.

- Possibility: receiving process can have many ports for different message types
- Server processes usually publish their service-ports for clients

- Clients can use static IP to access service-ports on servers (limiting, sometimes), but could use location-independent IP by
  - using name server or binder to bind names to servers at run-time – for relocation
  - Mapping location-independent identifiers onto lower-level address to deliver/send messages – supporting service migration and relocation

Characteristics of IPC

Reliability

- Validity: transmission is reliable if packets are delivered despite some drops/losses, and unreliable even if there is a single drop/loss
- Integrity: message must be delivered uncorrupted and no duplicates

Ordering

- Message packets, even if sent out-of-order, must be reordered and delivered otherwise it is a failure of protocol

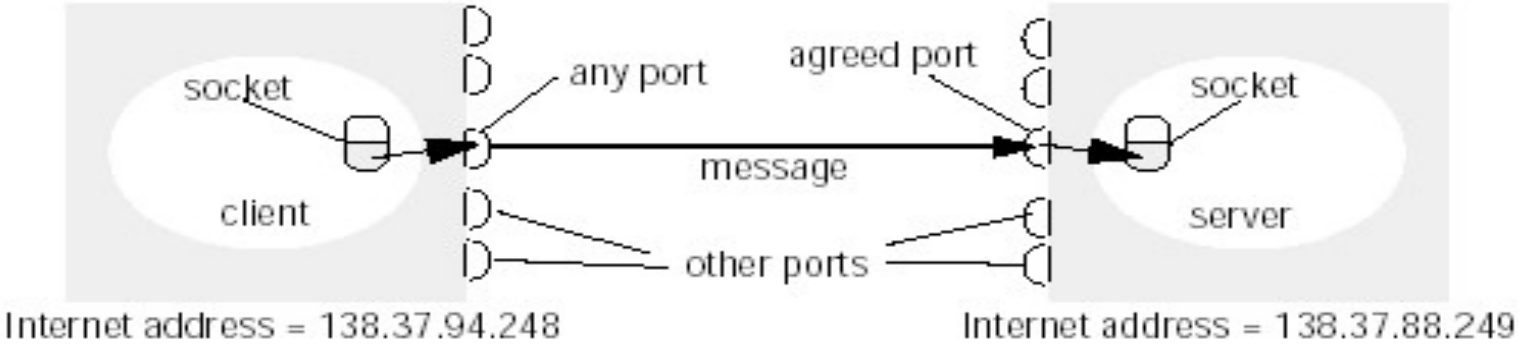
Means of IPC

Sockets

- Provide an abstraction of endpoints for both TCP and UDP communication
- Sockets are bound to ports on given computers (via the computer's IP address)
- Each computer has 2^{16} possible ports available to local processes for receiving messages
- Many processes in the same computer can deliver to the same port (many-to-one), however
- Sockets are typed/associated with either TCP or UDP

Means of IPC

Figure 4.2 Sockets and ports



UDP Datagram communication

- No acknowledgements/retries.
- Message size:The receiving process specifies an array of bytes,messages are truncated on arrival.
- IP allows packets lengths of 2^16.
- Larger messages are truncated
- Non-blocking sends but blocking receives
- Setting Time-outs on receives.

UDP Datagram communication..Failure Model

- Omission Failures : Messages may be dropped,either due to checksum error or because no buffer space available.
- Ordering : Messages can sometimes be delivered out of order.

UDP uses.

- DNS is implemented over UDP.

  These 3 sources of overhead are avoided.
- Need to store state information at source and destination
- The transmission of extra messages
- Latency of sender.

TCP stream communication

- Data is a stream of bytes
- Message sizes..streams,one or more packets
- Lost messages: Acknowledgement scheme
- Flow control..if writer is too fast for reader(blocking)
- Message duplication/ordering
- Message destinations

Issues related to stream communication

- Matching of data items

- Blocking

- Threads

Issues related to stream communication

- Failure Model

- To satisfy the integrity property of reliable communication TCP uses checksums to detect and reject corrupt packets

Use of TCP

- HTTP ,80
- FTP,20,21
- Telnet 23
- SMTP,25

A list of all TCP/UDP well known services.

TCP and UDP comparison

TCP protocol

- It is a connection oriented protocol
-
- It has flow control and error correction

- Not very fast,primary use is data transmission

- Schemes requiring delivery acknowledgement use TCP

TCP/IP and UDP comparison

UDP protocol

- It is connectionless protocol which means it can send packets without establishing connection with the receiver at first.
- It is error prone during transmission.
- It is fast and used mostly for audio and video streaming
- UDP ports are commonly used by services or programs that don't require the confirmation of delivery of packets. Most commonly used is DNS queries using UDP port 53.

Client – Server Communication

The RPC exchange protocols

R- Request protocol

RR- Request Reply (itself is like an ack)

(HTTP)(different status codes)

RRA – Request Reply Acknowledgement

Group communication

Multicast messages : single message to a group of processes.

Fault tolerance and replicated servers

Discovery servers for spontaneous networking

Better performance through replicated data

Propagation through event notifications