Communication in Distributed Systems

Sape J. Mullender
Huygens Systems Research Laboratory
Universiteit Twente
Enschede
Functions of Communication

- Transport data between processes, machines, sites
- Firewall against failures
- Force clean interfaces
- Hide distinction between “local” and “remote”
Desirable Properties of Protocols

- Low latency
- High throughput
- Minimum jitter (multimedia)
- Authenticated and Secure
- Hide as many failures as possible
- Report all other failures correctly
• Physical medium transmits a series of bits
• Groups of bits are transmitted as units known as *frames*
• Special patterns indicate beginning and end of frames
• Physical network is *point-to-point* or *broadcast*
• Physical medium (usually) *preserves order* and does *not duplicate* frames
• Most networks have hardware that uses frames to transmit addressed and checksummed *packets* of data from and to memory — this is the Media Access Control (MAC) layer
• Protocols are *layered*; each layer carries out some well-defined function — Data link: send packets over wires; Network: route through network; Transport: reliable end-to-end transport.
• Reliability is not bound to one specific layer.
• Error detection and retransmission is used to get reliable data transport.
• Routing differences and packet retransmission can cause *loss of order* or *duplication* of packets.
Datagrams vs. Virtual Circuits

Datagrams are used in connection-less communication: each packet (datagram) is sent (and routed) independently of all others.

Virtual circuits are used in connection-oriented communication: packets are sent through a virtual “wire” (no duplication, order preserving). Often the wire is made error free.

Each protocol layer can be organized as a datagram service or a virtual-circuit service independently of other layers: usually datagrams can be delivered out of order and packet loss or duplication is possible.
Types of Transport Service

Transport Layer transfers data between communication endpoints (processes, sockets, ports)

Usually offers a reliable service

1. Reliable virtual circuit — ordered packet stream, no loss or duplication.
2. Reliable byte stream — byte stream, as in Unix pipe
3. Reliable datagram — unordered, but reliable stream of datagrams
4. Reliable message stream — unordered reliable stream of arbitrary-size messages
Communication Types

- Remote operations
- Bulk data transfer
- One-to-many
- Continuous media
Remote Operation: A client requests a server to carry out an operation.

*Client* and *server* are roles that processes or machines take on.

One client’s server can be another server’s client.
Implementation — Reliable Network and Hosts

Printer server

Print file1

Printing

Print file2

File server

Read file1

Reading

Data

Read file2
Implementation — Unreliable Network
Need to keep state information:
Exactly Once — Host Crashes

One cannot distinguish *down* from *disconnected*.

Printer Server
Remote Operations

Obvious implementation: *reliable messages*

What is the meaning of an *ack*?

[Diagram of request and acknowledgement with processing in the middle]
Remote Operations

Request-response communication

Request

Ack

Request

Ack

Request

Ack
There is Fundamental Uncertainty

When hosts can crash (and forget), there is always fundamental uncertainty about what was and what was not done just before the crash.
When there is no feedback, we cannot know whether work was carried out or not.

The best we can do is make sure that, when there is feedback, the work was carried out correctly.

This is Saltzer’s *End-to-End Argument*.
Shared State

Reliable communication over unreliable media requires that communicating parties maintain *shared state*. Shared state is used for establishing packet order, and identify missing or duplicated packets. Packets must be labelled explicitly or implicitly with a *packet sequence number*. Reliable connections cannot survive crashes at the end points.
The correctness of a protocol is compromised by the arrival of the wrong packets with the right sequence number from another connection.

Packets, therefore, must also be labelled explicitly or implicitly with a connection identifier.
Establishing Shared State

If communication endpoints are reliable, shared state can be established implicitly at the beginning of time. Otherwise, one must establish connection identifiers and packet sequence numbers (on a per-connection basis).
Connection identifiers

- sequence number
- port pair plus time stamp
- port pair plus random number
- pair of disposable ports
- port pair plus time of day
- encryption key
Many networks have the property that packets have a bounded lifetime.

And even if they don’t, they can often be made to have one.

In such networks, the passage of time can be used for the establishment of a new session.
If the retransmission period is $t_r$, the maximum packet transmission time is $t_x$ and the maximum number of retransmissions is $k$, then

the maximum packet lifetime is

$$t_l = k \times t_r + t_x$$
In the $\Delta - t$ Protocol a new session automatically starts after a period of inactivity greater than $\Delta t = 2 \times t_l + t_p$, where $t_p$ is the maximum processing time between receiving a packet and generating a response.

After a crash, a process waits $\Delta t$ before responding to filter out packets from a previous session.
Can be viewed as a remote operation with a very large request or response message.

Given the very low packet-loss rate of modern networks, retransmission of whole messages is acceptable.
Broadcast — message to every process
Multicast — one message to a group of processes.
Various semantics of reliable multicast are possible.
Protocols may try to exploit hardware multicast facilities.
Important that protocols do not flood the network with acks.
Continuous Media

Very different from conventional data transport

- Minimizing latency is a primary concern
- Some data loss is often acceptable
- Bandwidth needed fairly constant over time
Components of latency

- Protocol overhead
- Packet size
- Media access
- Routing
- Transmission latency
- Buffering
- Scheduling
Ideal Network for Low Latency

- Small packets
- Virtual circuits
- Bandwidth reservation

ATM networks
Asynchronous Transfer Mode

- Mesh topology
- Cells with 48-byte payload
- Header contains virtual circuit identifier (VCI)
- Routing at VC creation time
- Bandwidth reservation possible
- Typical link bandwidths 100 Mbps to 10 Gbps
- No limit on aggregate bandwidth
Continuous Media Transport

Lightweight protocols
No retransmission
Sometimes need media-dependent resynchronization
RPC Transport Protocols

- Cannot do “Exactly Once”
- Must settle for “At-Most-Once”
- May need authentication

Setting up of necessary shared state can be amortized over many remote operations: request/reply protocol layered over a connection
• Encryption Keys — identifies association between communicating principals
• Session identifier — identifies current association between instances of principals
• Operation identifier — identifies current operation
• Packet identifier — identifies packet within request or response message
Establishing keys is a relatively expensive operation.

Keys can be cached (within bounds) and re-used in new sessions.

Connection for reliable message transport can be nested inside connection for at-most-once request/response protocol.
Remote operations are like function calls.
But invocation is different.
Remote Procedure Call gives remote operations approximately the syntax and semantics of procedure-call.
Remote Procedure Call

Client

Call

Marshall Call Parameters

Send Message

RPC Transport

Network

Server

Call

Unmarshall Call Parameters

Receive Message

RPC Transport

Network

Client

Return

Unmarshall Result Parameters

Receive Message

RPC Transport

Client Stub

Server Stub

Server Stub

RPC Transport

RPC Transport

RPC Transport

Client Stub

Server Stub

Marshall Result Parameters

Send Message

Marshall Call Parameters

Unmarshall Result Parameters

Return
Differences

Failure Semantics — More failures to consider
Parameter-passing restrictions — no pointers
Call-by-value, call-by-result are no problem.
Call-by-reference and all-by-name is a problem.
Global variables are a problem.
Pointers are a problem — call-back is possible, but ...
Function parameters are a problem.
Extra syntax can be used to indicate parameter direction (*in*, *out*, or *in/out*).

Extra syntax can be used to indicate how full a buffer is.
Interface Definition Language

Defines types and representation of those types on the wire.

```plaintext
interface file {
    const MAXBUF = 0x100000;
    int proc read(
        in unsigned long offset,
        inout unsigned int nbytes,
        out char data[nbytes:MAXBUF];
    );
    int proc write ...
}
...
refobj file foo;
char buffer[file.MAXBUF];

result = foo.read(0, 0x1000, buffer);
```
Marshalling

Convert between host's type system and network type system.

- Byte order
- Floating-point representation
- Character sets
- Non-linear structures
- Pointers
- Network representations
Interface Definition Languages

- Argus (MIT)
- Network Computing Protocol (Apollo)
- XDR, External Data Representation (SUN)
- Flume (DEC SRC)
- Amoeba Interface Language (CWI)
Using an IDL

1. **Client Source**
2. **Server Source**
3. **IDL**
4. **Stub Compiler**
5. **Client Stub**
6. **Server Stub**
7. **Compile & Link**
8. **Client Binary**
9. **Server Binary**
10. **Binding agent**
11. **RPC Transport Connection**
The mapping from a desired service to an *instance* of that service is called *binding*

*Binding* is mediated through a *binding agent*

Examples of binding agents are YP, DNS, GNS, X.500, the ANSA Trader, or even a *method table*
Early binding means that a binding information is built into an application.

Late binding means that binding information is obtained when it is needed; i.e., when a connection is made.

“Early binding is extremely wicked.” (Roger Needham)
Input to the binding agent must name a service
Some binding agents allow descriptions as names (ANSA Trader)
Output of the binding process can be a network address, a socket, a certificate, ...
The Domain Name Service maps \{pathname, type\} pairs to type-dependent information.

\{cs.utwente.nl, mx\}, for instance, will map to a prioritized set of mail hosts for the cs.utwente.nl domain.

> set q=mx
> cs.utwente.nl.

Name Server: artemis.cs.utwente.nl
Address: 130.89.10.7

cs.utwente.nl  preference = 0, mail exchanger = utrhcs.cs.utwente.nl
cs.utwente.nl  preference = 10, mail exchanger = utcvx.civ.utwente.nl
utrhcs.cs.utwente.nl internet address = 130.89.10.247
utcvx.civ.utwente.nl internet address = 130.89.1.10