CS603: Distributed Systems

Lecture 1: Basic Communication Services

PURDUE UNIVERSITY

Reference Material

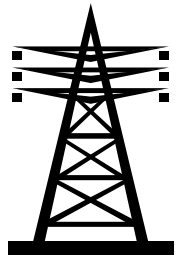
- Textbooks
  - Ken Birman: Reliable Distributed Systems
- Recommended reading
  - Research papers that will be specified for each lecture

What is a Distributed System?

A *distributed computing system* is a set of computer programs executing on one ore more computers and coordinating actions by exchanging messages.

Examples of Distributed Systems

- Air Traffic Control
- Space Shuttle
- Banking Systems
- Grid Power Systems
- Modern Data Centers



Distributed Systems Requirements

- Reliability: provide continuous service
- Availability: ready to use
- Safety: systems do what they are supposed to do, avoiding catastrophic consequences
- Security: withstands passive/active attacks from outsiders or insiders

…not easy to achieve because

- Computers and networks fail in many (often unpredictable) ways
- Computers get compromised
- Real-time constraints
- Performance requirements
- Complexity

Why Do Computer Systems Fail?

- 1985, Fault-tolerant system (Tandem)
  - System administration (operator actions, system configuration and maintenance)
  - Software faults, environmental failures
  - Hardware failures (disks and communication controllers)
  - Power outages
- 2004, Where are we now?! The Internet Age
  - Operator error (particularly configuration errors) is the leading cause of failures
  - Failures in custom-written front-end software
  - Not enough on-line testing

Why do Internet services fail, and what can be done about it? D. Oppenheimer, A.Ganapathi and D. A. Patterson, 2003.
Why Do Computers Stop and What can be done about it? Jim Gray, 1985

Why Do Computers Get Compromised?

- Software bugs
- Administration errors
- Lack of diversity, same vulnerability is exploited
- The explosion of the Internet facilitates the spread of malware

..how do computer system fail…

- Halting failures: no way to detect except by using timeout
- Fail-stop failures: accurately detectable halting failures
- Send-omission failures
- Receive-omission failures
- Network failures
- Network partitioning failures
- Timing failures: temporal property of the system is violated
- Byzantine failures: arbitrary failures, include both benign and malicious failures

Air Traffic Control: A Case Scenario

- Prepared with slides courtesy of Prof. Ken Birman and used in a similar course at Cornell University

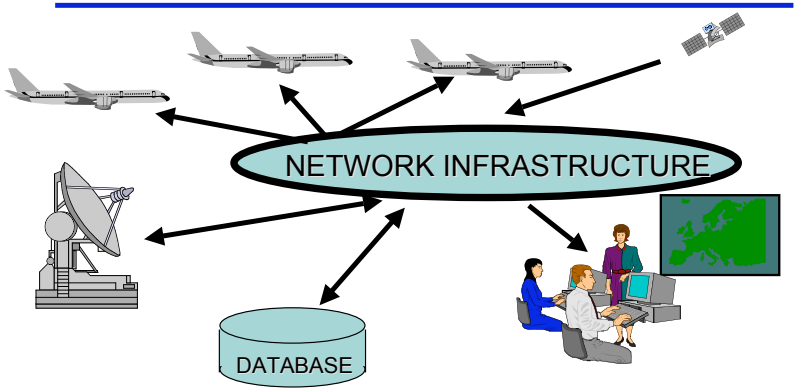
ATC and Its Role

- Assists planes in taking-off, landing and en route (during flying)
- Assigns trajectories making sure that planes fly at a safe distance
- Each ATC has a certain space assigned to it
- As planes move they enter the space controlled by different ATCs
- Planes are also equipped with a collision avoidance system TCAS

More Details on ATC

- Air space divided in sectors
- Each sector has a control center
- Centers may have few or many (50) controllers
  - In USA, controller works alone
  - In France, a “controller” is a team of 3-5 people
- Data comes from a radar system that broadcasts updates every 10 seconds
- Database keeps other flight data
- Controllers “owns” smaller sub-sectors
- Controllers make very quick decision(s) based on available data

ATC Architecture



THE SYSTEM MUST BE AVAILABLE ALL TIME and MAINTAIN CONSISTENCY OF THE INFORMATION

What Can Go Wrong?

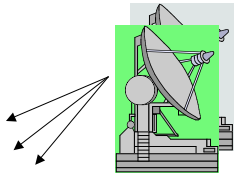
- Overloaded computers can often crash
- Systems may get slow as volume of air traffic rises
- Inconsistent displaying:
  - phantom planes
  - missing planes
  - stale information
- Scheduled maintenance going wrong
- Some major outages recently (and some near-miss stories associated with them), some very unfortunate events as recent as 2003.

Concept of IBM's 1994 System

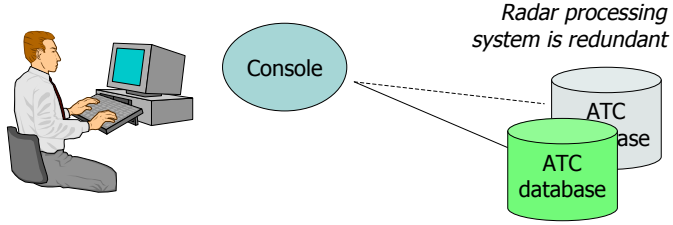
- Replace video terminals with workstations
- Build a highly available real-time system guaranteeing no more than 3 seconds downtime per year
- Offer much better user interface to ATC controllers, with intelligent course recommendations and warnings about future course changes that will be needed
- IBM approach was based on lock-step replication
  - Replace every major component of the system with a fault-tolerant component set
  - Replicate entire programs ("state machine" approach)

IBM ATC System Architecture

Independent consoles… backed by ultra-reliable components



Radar processing system is redundant



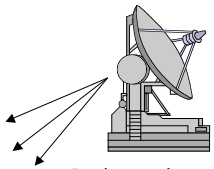
ATC database is really a high-availability cluster

French ATC Project Concept

- French project used replication selectively.
- Some specific and critical data was replicated, for example “list of planes currently in sector A.17”
  - E.g. controller interface programs could maintain replicas of certain data structures or variables with system-wide value
  - Programs did computing on their own helped by databases
  - Program “hosts” a data replica but isn’t itself replicated

French ATC System Architecture

Multiple consoles… but in some ways they function like one



Radar updates sent with hardware broadcasts

ATC database only sees one connection

Other technologies used

- Both used standard off-the-shelf workstations (easier to maintain, upgrade, manage)
  - IBM proposed their own software for fault-tolerance and consistent system implementation
  - French used Isis software developed at Cornell
- Both developed fancy graphical user interface much like the Web, pop-up menus for control decisions, etc.

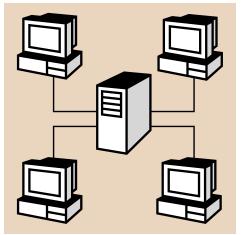
IBM Project Was a Fiasco!!

- IBM was unable to implement their fault-tolerant software architecture! Problem was much harder than they expected.
  - Even a non-distributed interface turned out to be very hard, major delays, scaled back goals
  - And performance of the replication scheme turned out to be terrible for reasons they didn’t anticipate
- The French project was a success and never even missed a deadline… In use today.

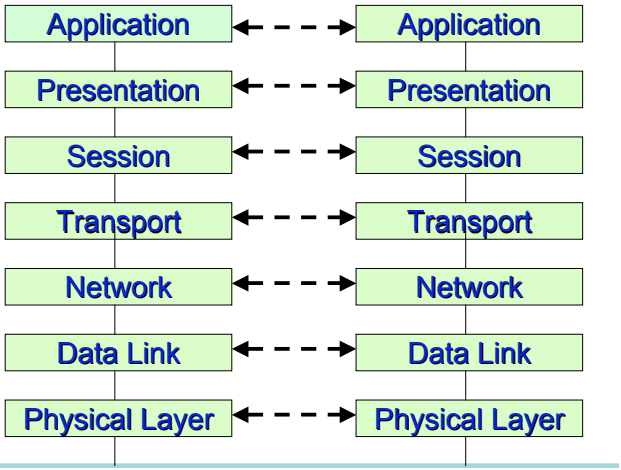
Where did IBM go wrong?

- Their software “worked” correctly
  - The replication mechanism wasn’t flawed, although it was much slower than expected
- But somehow it didn’t fit into a comfortable development methodology
  - Developers need to find a good match between their goals and the tools they use
  - IBM never reached this point
- The French approach matched a more standard way of developing applications

Basic Communication Services



OSI/ISO Model



Internet Protocol - IP

- IP is the current delivery protocol on the Internet, between hosts.
- IP provides ‘best effort’, unreliable delivery of packets.
- There are two versions:
  - IPv4 is the current routing protocol on the Internet
  - IPv6, a newer version, still not totally embraced by the community

Transport Protocols

- Provides communication between processes running on hosts
- The most common transport protocols are UDP and TCP.
- OS provides support for developing applications on top of UDP and TCP.

User Datagram Protocol - UDP

- Connectionless protocol for a user process:
  - No connection established
  - Unreliable transmission: no guarantee that the packets reach their destination.
  - Error detection.
- Runs on top of IP.

Transmission Control Protocol - TCP

- Connection oriented protocol for a user process:
  - Reliable, full-duplex channel: acknowledgements, retransmissions, timeouts, flow-control
  - The packets are delivered in the same order in which they were sent.
  - Flow Control: Max allowed window size
  - Congestion control:
    - Slow-start phase – exponential increase (until the slow-start threshold is hit)
    - Congestion Avoidance phase – additive increase
    - Multiplicative Decrease on timeout.

Hardware Addresses

- Hosts access the physical medium via network cards.
- Each network card is uniquely identified by a 48 bit (6 bytes) number, called hardware address, or Ethernet address.
- Ethernet addresses are hardwired into the electronics of the network device.

$$\underbrace{b_1\ b_2\ b_3}_{\text{unique to the manufacturer of the card.}}\ \underbrace{b_4\ b_5\ b_6}_{\text{assigned by the manufacturer.}}$$

- ARP/RARP protocols map IP addresses to hardware addresses and vice versa.

IP Addresses

- Hosts are identified in the network by IP addresses
- Two different network addresses:
  - IPv4 addresses: 32 bits addresses, most used
  - IPv6 addresses: 128 bits addresses.
- Each decimal number represents eight bits of binary data (value between 0 and 255).
- Divided in classes.
  - Network addresses with first byte between 1 and 126 are class A
  - Network addresses with first byte between 128 and 191 are class B
  - Network addresses with first byte between 192 and 223 are class C
  - All other networks are class D, used for special functions or class E which is reserved.

Naming services: DNS

- People prefer names for hosts (hostnames):
  - Name: ugrad1
  - Fully qualified name: ugrad1.cs.jhu.edu
- DNS (Domain Name System) maps hostnames to IP addresses.
- Example:

  ugrad1.cs.jhu.edu has the IP 128.220.224.76

NATs and their implications

- There are not enough IP addresses
- Solutions: IPv6 or ….Network Address Translation (NAT)
- NAT allows a single device, to act as an agent between the Internet (or "public network") and a local (or "private") network: only a single, unique IP address is required to represent an entire group of computers
- Computers can not communicate directly, STUN client-server protocol allows computers to discover each other behind a NAT (learn their public addresses), but requires presence of STUN server

Problems with NATs

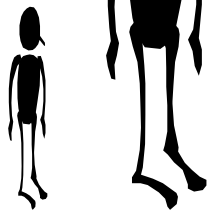
- Break end-to-end control
- Hosts depend on same trusted point (the STUN server)
- Add complexity
- Prevent IP security deployment

IP Multicast

- Provides support for group communication: send to multiple parties
- Groups are specified by reserved IP multicast addresses 224.0.0.0 to 239.255.255.255.
- Unreliable communication
- IGMP is used to dynamically register individual hosts in a multicast group on a particular LAN.
- Network cards recognize IP multicast addresses: hosts that did not subscribe to a particular group will not process those packets (unlike broadcast that is processed by all hosts in a network segment)
- Issues with IP multicast: can be used to cause DOS, many ISP and enterprise network block IP multicast communication

Byte Order

- Different systems store multibyte values (for example int) in different ways.
  - HP, Motorola 68000, and SUN systems store multibyte values in Big Endian order: stores the high-order byte at the starting address
  - Intel 80x86 systems store them in Little Endian order: stores the low-order byte at the starting address.
- Why is this a problem for network applications? Data is interpreted differently on hosts with different architectures.



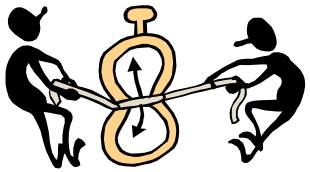
Buffering and Fragmentation

- Buffering: OS maintains a set of buffers used to temporary store incoming and outgoing messages
- THE BUFFERING SPACE is LIMITED
- Fragmentation: IP datagrams are fragmented, they can travel on different paths
- When processes send very fast, packets can be dropped by the OS without any notification
- On sending: no OS memory can be obtained for one or several fragments
- On receiving: one or several fragments did not make it to the destination, entire datagram is dropped

Why these protocols do not provide better support for distributed applications?

The End-to-End Argument

- End to end arguments in System Design. Saltzer, Reed, Clark TOCS 1990.



What is all about?

- Analyzes what services should be provided at low levels and what should be provided by the application
- Commonly cited as a justification for not addressing reliability at low levels and let application handle it
- Example: how to transfer a file: hop-by-hop or end-to-end
- Low-level mechanisms should focus on speed, not reliability
- The application should worry about “properties” it needs

References

- Chapter 1 and 2 from Reliable Distributed Systems
- Why do Internet services fail, and what can be done about it? D. Oppenheimer, A.Ganapathi and D. A. Patterson, 2003.
- Why Do Computers Stop and What can be done about it? Jim Gray, 1985.
- End to end arguments in System Design. Saltzer, Reed, Clark TOCS 1990.