Distributed Systems - Architecture

CS239
Advanced Operating Systems
October 26, 1999
Dr. Richard Guy

Lecture Source Materials

Singhal, **Chapter 4: Architectures of Distributed Systems**

Tanenbaum95, Distributed OS, Chapter 1-2.

Goscinski91, Distributed OS, Chapter 1-3.

Galli00, Distributed OS, Chapter 1.

Outline

- Definition: What is a distributed system?
- Motivation: Why do we care?
- Taxonomy: Is it a NOS? a DOS? a MOS?
- Design Issues: What are the fun problems?
- Communications: How to organize things?
- Communications: How to talk?

What is a distributed system?

Assorted definitions:

"a term used to define a wide range of computer systems, from weakly-coupled systems such as wide area networks to strongly-coupled systems such as local area networks to very strongly-coupled systems such as multiprocessor systems." (Goscinski, p. 3)

or...

" ... a collection of heterogenous computers and processors connected via a network."(Galli, p. 2)

more...

" ... consists of several computers that do not share a memory or clock; the computers communicate with each other by exchanging messages over a communications network; and each computer has its own memory and runs its own operating system." (Singhal, p. 71)

and...

"a collection of independent computers that appear to the users of the system as a single computer." (Tanenbaum, p. 2)

- Key point: **AUTONOMY**

Examples

- CSD workstation network
- Robotic factory
- Bank branches

Outline

- Definition: What is a distributed system?
- **Motivation: Why do we care?**
- Taxonomy: Is it a NOS? a DOS? a MOS?
- Design Issues: What are the fun problems?
- Communications: How to organize things?
- Communications: How to talk?

Motivation for Dist. Systems

- Performance Economics + Communications
  - Grosch' s Law (MIPS : $\$^2$) debunked
- Inherently distributed tasks

+ Increased reliability
+ Incremental expandability
+ Resource sharing

and the downside...

- Software complexity
- Network reliability
- Security
- Supply/Demand location mismatch
- Non-decomposable computation

Outline

- Definition: What is a distributed system?
- Motivation: Why do we care?
- **Taxonomy: Is it a NOS? a DOS? a MOS?**
- Design Issues: What are the fun problems?
- Communications: How to organize things?
- Communications: How to talk?

Distributed System Taxonomy

- Hardware
  - Flynn's SISD/SIMD/MISD/MIMD
  - Tanenbaum's MIMD taxonomy

Flynn's Taxonomy [ca. 1966]

Data Stream

Instruction Stream

Flynn's Taxonomy

Data Stream

Instruction Stream		Single	Multiple
	Single		
	Multiple		

Flynn's Taxonomy

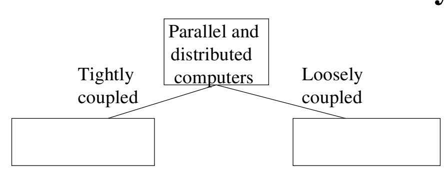
Data Stream

Instruction Stream		Single	Multiple
	Single	SISD	SIMD
	Multiple	MISD	MIMD

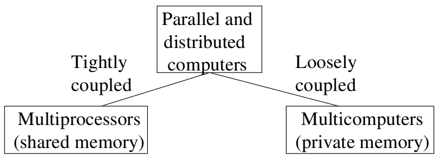
Tanenbaum MIMD Taxonomy

Parallel and distributed computers

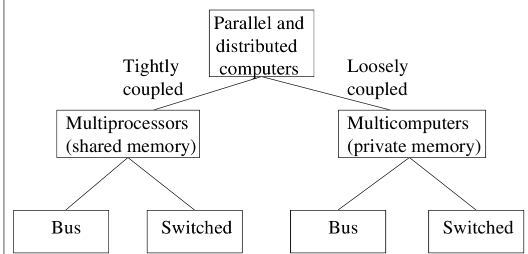
Tanenbaum MIMD Taxonomy



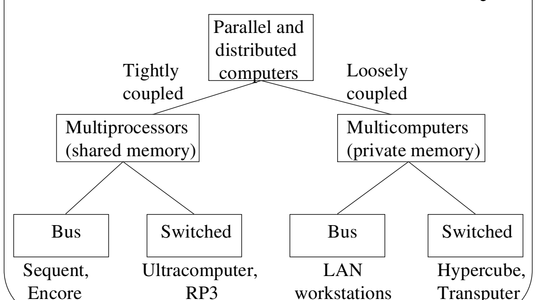
Tanenbaum MIMD Taxonomy



Tanenbaum MIMD Taxonomy



Tanenbaum MIMD Taxonomy



Software Taxonomy

- Tightly vs loosely coupled
- Crossproduct of coupling and HW/SW

		HARDWARE	
		loosely coupled	tightly coupled
SOFTWARE	loosely coupled		
	tightly coupled		

Software Taxonomy

- Tightly vs loosely coupled
- Crossproduct of coupling and HW/SW

		HARDWARE	
		loosely coupled	tightly coupled
SOFTWARE	loosely coupled	Network Operating System (NOS)	
	tightly coupled		

Software Taxonomy

- Tightly vs loosely coupled
- Crossproduct of coupling and HW/SW

		HARDWARE	
		loosely coupled	tightly coupled
SOFTWARE	loosely coupled	Network Operating System (NOS)	
	tightly coupled	Distributed Operating System (DOS)	

Software Taxonomy

- Tightly vs loosely coupled
- Crossproduct of coupling and HW/SW

	HARDWARE: loosely coupled	HARDWARE: tightly coupled
SOFTWARE: loosely coupled	Network Operating System (NOS)	
SOFTWARE: tightly coupled	Distributed Operating System (DOS)	Multiprocessor Timesharing (MOS)

Software Taxonomy

Item	NOS	DOS	MOS
Appear to be virtual uniprocessor?			
All CPUs run same OS?			
How many OS copies?			
How is communication achieved?			
A priori n/w protocols agreement?			
Single run queue for processes?			
Well-defined file sharing semantics?			

Software Taxonomy

Item	NOS	DOS	MOS
Appear to be virtual uniprocessor?	No		
All CPUs run same OS?	No		
How many OS copies?	***n***		
How is communication achieved?	Shared files		
A priori n/w protocols agreement?	Yes		
Single run queue for processes?	No		
Well-defined file sharing semantics?	Usually No		

Software Taxonomy

Item	NOS	DOS	MOS
Appear to be virtual uniprocessor?		Yes	
All CPUs run same OS?		Yes	
How many OS copies?		***n***	
How is communication achieved?		Messages	
A priori n/w protocols agreement?		Yes	
Single run queue for processes?		No	
Well-defined file sharing semantics?		Yes	

Software Taxonomy

Item	NOS	DOS	MOS
Appear to be virtual uniprocessor?			Yes
All CPUs run same OS?			Yes
How many OS copies?			1
How is communication achieved?			Shared memory
A priori n/w protocols agreement?			No
Single run queue for processes?			Yes
Well-defined file sharing semantics?			Yes

Software Taxonomy

Item	NOS	DOS	MOS
Appear to be virtual uniprocessor?	No	Yes	Yes
All CPUs run same OS?	No	Yes	Yes
How many OS copies?	***n***	***n***	1
How is communication achieved?	Shared files	Messages	Shared memory
A priori n/w protocols agreement?	Yes	Yes	No
Single run queue for processes?	No	No	Yes
Well-defined file sharing semantics?	Usually No	Yes	Yes

Outline

- Definition: What is a distributed system?
- Motivation: Why do we care?
- Taxonomy: Is it a NOS? a DOS? a MOS?
- **Design Issues: What are the fun problems?**
- Communications: How to organize things?
- Communications: How to talk?

Distributed OS Design Issues

- Transparency
- Computational model
- Flexibility
- Reliability
- Performance
- Scalability

Distributed OS Design Issues, cont.

- Process Synchronization
- Compatibility
- Resource Management
- Realt-time

Transparency: Single-system image

- Name transparency
- Location transparency
- Access transparency
- Migration transparency
- Replication transparency
- Concurrency/parallelism transparency
- Failure transparency

Computational model

- client/server
- peer-to-peer

Flexibility/Extensibility

- monolothic
- μ-kernel
- layered

Reliability

- Availability
- Integrity
  - storage: correct, consistent
  - computation
  - security
- Fault Tolerance
  - richer failure modes
  - client-server model

Performance

- fine- vs. coarse-grained parallelism
- metrics
- measurement methods/tools

Scalability

- avoid centralized
  - components
  - tables
  - algorithms
    - no machine has complete info about system state
    - machines make desicions only on local info
    - failure of one machine doesn't ruin whole algorithm
    - no implicit assumption of global clock

Compatibility

- Binary
- Execution
- Protocol

Resource management

- Data migration
- Computation migration
- Distributed scheduling
- Security

Real-time

- hard vs soft vs firm
- synchronous vs asynchronous vs isochronous
- new problems
  - network latency
  - bandwidth vs cost
  - routing optimization
  - micronetwork behavior

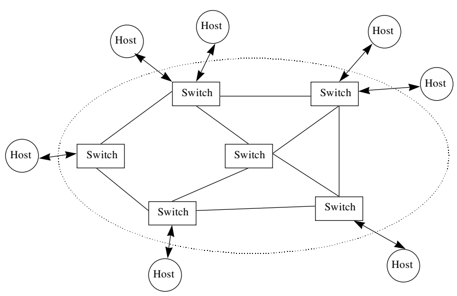
Outline

- Definition: What is a distributed system?
- Motivation: Why do we care?
- Taxonomy: Is it a NOS? a DOS? a MOS?
- Design Issues: What are the fun problems?
- **Communications: How to organize things?**
- Communications: How to talk?

Communications Networks

- Wide area networks (WANs)
- Local area networks (LANs)
- ISO OSI reference model

Wide area network diagram



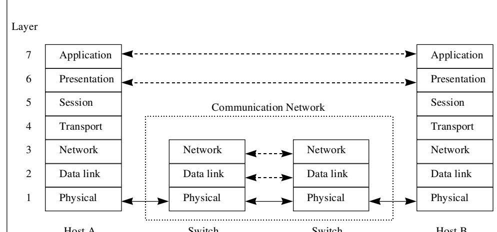
Wide area networks (WANs)

- Links are leased circuits from telcos
  - Arpanet was 56Kbps
  - Internet backbone today is at least T3 (45Mbs)
    - typical link is T1 (1.5Mbps)
    - new backbone at 1Gbs (ATM SONET)
    - typical ATM link is 155Mbps
- Heterogeneous hosts
- Circuit vs Packet switching
  - Store-and-forward

Local area Networks (LANs)

- 10Mbps-1Gbps transmission rate
- Small geographic scope (<10km)
- Topologies
  - Bus/tree: CSMA/CD, token bus
  - Ring: token ring, slotted ring
- Low transmission error rate

ISO OSI Reference model



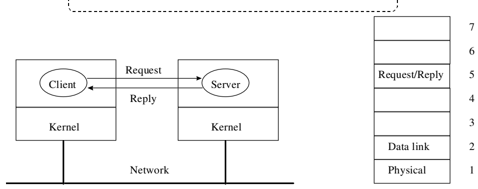
Outline

- Definition: What is a distributed system?
- Motivation: Why do we care?
- Taxonomy: Is it a NOS? a DOS? a MOS?
- Design Issues: What are the fun problems?
- Communications: How to organize things?
- **Communications: How to talk?**

Communications Primitives

- Client-Server Model
- Remote Procedure Call
- Group Communication

Client-Server Model



- Request/reply message passing protocol
- Blocking vs. non-blocking primitives
- Reliable vs. unreliable primitives
- Synchronous vs. asynchronous primitives

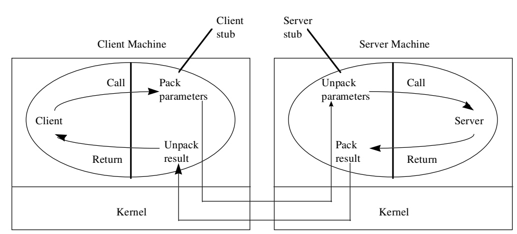
Message passing problems

- Pairing replies with requests
- Heterogeneous data representation
- Addressing remote partner
- Handling system/communication failures

Remote Procedure Call

- Basic RPC model
- Detailed RPC issues

Basic RPC Model



RPC details

- Client/server stubs
- Dynamic binding to server
- Parameter/result marshaling
- Failure semantics

Parameter/result marshaling

- Call-by-value/reference/…
- Heterogeneous hardware architectures
  – ASCII vs EBCDIC
  – big vs little endian
  – strings vs integers
- Canonical form vs dynamic conversion
- Pointer dereferencing

Failure semantics

- Lost replies & idempotency
- Client crash & orphans
- Server crash semantics
  – *At least once* (partials possible)
  – *At most once* (partials possible)
  – *Exactly once*
  – *No promises*

Group Communication

- Basics
- Group management
- Correctness Issues

Basics of Group Communication

- one-to-one vs. one-to-many
- dynamic groups
- multicast/broadcast/unicast

Group Management

- closed vs. open groups (exclusivity)
- peer vs. hierarchical (internal)
- group membership (dynamic mgt)
- group addressing (single/multiple/predicate)

Correctness of Group Communication

- Send/Receive primitives (merge RPC/group?)
- Atomicity
- Message ordering (global/consistent time)
- Overlapping groups
- Scalability (volume/gateway)

Summary

- Definition: A distributed system is …
- Motivation: We care because ...
- Taxonomy: An OS is an xOS because ...
- Design Issues: The fun problems are ...
- Communications: We organize things ...
- Communications: Pieces talk by ...