Introduction to Distributed Systems and Distributed Shared Memory

Jeong-Hyon Hwang (jhh@cs.albany.edu)
What is a Distributed System?
What is a Distributed System?

A *distributed system* is:

A collection of independent computers that appears to its users as a single coherent system.
Very Brief History of Computing

• Computer

- ENIAC (1946)
  - 500 flops
  - $500K (now $5M-$25M)

- Dell T100 (2009)
  - 2x10^9 flops
  - $300

- Cray XT5 Jaguar (2009)
  - 1.7x10^15 flops
  - >$100M

10^{11} performance/cost ➲

• Internet

- The ARPA Network
  - DEC 1969

Wednesday, March 17, 2010
Very Brief History of Computing

- **Computer**

  ENIAC (1946)
  - 500 flops
  - $500K (now $5M-$25M)

  10¹¹ performance/cost ↑

  Dell T100 (2009)
  - 2x10⁹ flops
  - $300

  Cray XT5 Jaguar (2009)
  - 1.7x10¹⁵ flops
  - >$100M

- **Internet**

  THE ARPA NETWORK
  DEC 1969

  600M nodes (2009) [https://www.isc.org/solutions/survey/history]
Middleware

A distributed system organized as middleware. Note that the middleware layer extends over multiple machines.
Benefits of Distributed Systems
Benefits of Distributed Systems

• conquer geographic separation
  - Web
Benefits of Distributed Systems

• conquer geographic separation
  - Web

• build reliable systems out of unreliable components
  - Chord, CAN, RON
Benefits of Distributed Systems

• conquer geographic separation
  - Web

• build reliable systems out of unreliable components
  - Chord, CAN, RON

• aggregate many computers for high capacity
  - aggregate cycles: MapReduce, Dryad
  - aggregate bandwidth: Bullet, BitTorrent
  - aggregate disks: BigTable, Dynamo
Design/Implementation Goals

• Access to Resources
  - economics of resource sharing

• Openness
  - interoperability
  - portability
  - extensibility

• Distribution Transparency

• Scalability
In an open middleware-based distributed system, the protocols used by each middleware layer should be the same, as well as the interfaces they offer to applications.
## Transparency in a Distributed System

<table>
<thead>
<tr>
<th>Transparency</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>Hide differences in data representation and how a resource is accessed (e.g., distributed file systems)</td>
</tr>
<tr>
<td>Location</td>
<td>Hide where a resource is located (e.g., naming)</td>
</tr>
<tr>
<td>Migration</td>
<td>Hide that a resource may move to another location</td>
</tr>
<tr>
<td>Relocation</td>
<td>Hide that a resource may be moved to another location while in use</td>
</tr>
<tr>
<td>Replication</td>
<td>Hide that a resource may be shared by several competitive users</td>
</tr>
<tr>
<td>Concurrency</td>
<td>Hide that a resource may be shared by several competitive users (e.g., isolation in ACID)</td>
</tr>
<tr>
<td>Failure</td>
<td>Hide the failure and recovery of a resource</td>
</tr>
<tr>
<td>Persistence</td>
<td>Hide whether a (software) resource is in memory or on disk</td>
</tr>
</tbody>
</table>
Communications

• How remote processes can communicate with each other?
Communications

• How remote processes can communicate with each other?

• using communication primitives (e.g., TCP socket, MPI)
Communications

- How remote processes can communicate with each other?
- using communication primitives (e.g., TCP socket, MPI)
- using distributed shared memory
Distributed Shared Memory

- What is virtual memory?
- Where are the page frames stored when paged (swapped) out?
- Distributed shared memory (DSM) is a way of making a non-inherently shared memory machine look like it has shared memory.
Distributed Shared Memory

- software presents the abstraction of shared memory
- performance goal: minimize latency
- either an OS or a language run-time manages the shared memory
- there are many different DSM granularities
  - page-based DSM: like regular virtual memory
  - shared-variable DSM and object-based DSM: managed by a language run-time system
Page-based DSM

Pages of address space distributed among four machines

Situation after CPU 1 references page 10

Situation if page 10 is read only and replication is used
Page-based DSM

Pages of address space distributed among four machines

Situation after CPU 1 references page 10

Situation if page 10 is read only and replication is used

What kinds of inefficiency can happen?
False Sharing

- False sharing of a page between two independent processes.
- How big should a page be?
Implementation

• Think of how you would implement such a “virtual” shared memory among workstations
  - Software: at the OS level
  - Software: at a language runtime level (e.g., Java VM)
  - Hardware level support
    • most efficient, but costly
  - Hybrid
Reducing Latency

• migration
  - migrate shared unit (page, object, etc.) to node which has current access -> need state to determine current location

• replication
  - keep multiple copies -> need per shared unit state for keeping track of replicas, and types of access at each replica
Access Algorithm

• Single Reader – Single Writer
  - simplest, migration can suffice, not very efficient

• Multiple Readers – Single Writer
  - more general, need to track current/most recent writer (aka owner)

• Multiple Readers – Multiple Writers
  - maximum concurrency, need to resolve write conflicts through consistency protocols
Coherence Policy

• write update
  • allows multiple readers and writers
  • multicast message to update peer copies
  • processes have to agree on a total order for multicast writes for SC

• write invalidate
  • first invalidate peers
  • write to local cache
  • subsequent reads will get this new value
Distributed Objects

- Common organization of a remote object with client-side proxy.