Announcements

- Please email me your presentation preferences by Thursday
- So far received only 17 of 30
- Need 2 students for 9/9
- This Thursday:
  - Globus paper
  - End-to-end paper
- Evaluations due before class

Recap

- Case for distributed systems
- Some common forms
  - Client-server, peer-2-peer, clusters
  - rlogin.cs.vt.edu (uses random DNS)
- Types of OS: NOS, DOS, Middleware
- Goals for distributed systems

Comparison Between Systems

<table>
<thead>
<tr>
<th>Item</th>
<th>Distributed OS</th>
<th>Network OS</th>
<th>Middleware-based DB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transparency</td>
<td>Very High</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Same OS?</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td># OS copies</td>
<td>1</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>Communication</td>
<td>Shared Memory</td>
<td>Messages</td>
<td>Files</td>
</tr>
<tr>
<td>Resource Management</td>
<td>Global, central</td>
<td>Global, distributed</td>
<td>Per node (?)</td>
</tr>
<tr>
<td>Scalability</td>
<td>No</td>
<td>Moderately</td>
<td>Yes</td>
</tr>
<tr>
<td>Openness</td>
<td>Closed</td>
<td>Closed</td>
<td>Open</td>
</tr>
</tbody>
</table>

Outline for Today

- Goals for distributed systems (cont’d)
- Scaling techniques & design principles for distributed systems

Goals for Distributed Systems

- Transparency
- Consistency
- Robustness
- Scalability
- Openness
- Flexibility
Types of Scalability

- Size
- Geography
- Administration

- What are challenges for each of these?
- What causes poor scalability?

Centralization Pitfalls

- Centralized services
  - Single point of failure
- Centralized data
  - Bottlenecks → high latency
- Centralized algorithms
  - Requiring global knowledge
- Examples?

Decentralized Algorithms

- Incomplete information about global state
- Decide based on local state
- Tolerate failure of individual nodes
- No global clock

Scaling Techniques

- Hide communication latencies
  - Use asynchronous communication whenever possible

Asynchronous RPC

- Uses synchronous, transient, receipt-based messaging.
Deferred synchronous RPC

- Combines two asynchronous RPC.

![Diagram showing Deferred synchronous RPC process flow](image)

Scaling Techniques (cont’d)

- Minimize communication
  - Caching & replication
  - Piggybacking
  - Placement of computation
  - Other examples:

![Diagram showing Scaling Techniques](image)

Scaling Techniques (cont’d)

- Minimize synchronization requirements
  - Example: Consistency in caching

- Distribute workload
  - Example: DNS

DNS Example: fbox.cs.vt.edu

- DNS Zones

![Diagram showing DNS example](image)

Openness

- Interface specifications (IDL)
- Syntax:
  - Names of methods
  - Types of parameters
  - Exceptions raised etc.
- Examples:
  - MS MIDL, OMG IDL, ONG RPC (SunRPC)

Example: OMG IDL (CORBA)

```plaintext
module data {
    typedef string<200> name;
    typedef string<20>   phone;
    struct entry { name n; phone p; };
    exception duplicate { phone p; }
    exception notfound {};
    interface phonebook {
        void  add(in entry e)   raises (duplicate);
        void  remove(in name n) raises (notfound);
        phone find(in name n)   raises (notfound);
    };
}
```
Language Bindings

• Describe how IDL maps to implementation language, e.g. CORBA-to-C++, CORBA-to-C, CORBA-to-Java

• Related example: Java RMI

Openness (cont’d)

• Interface specifications should be semantically
  – Complete
  – Neutral wrt implementation

• Your experience?

Summary

• Goals for distributed systems
  – Transparency, Consistency, Robustness, Scalability, Openness, Flexibility
• Scaling techniques
• Types of messaging
  – Synchronous vs. asynchronous
• Design principles
  – Layered protocols, end-to-end argument