Introduction

Ashvin Goel

Distributed Systems, ECE 1746, Fall 2003

Topics

- Class format
- Characteristics of distributed systems
- Design issues in distributed systems
- Should distributed systems behave like centralized systems?

Class Format

- Class website available from my home page
  - `http://www.eecg.toronto.edu/~ashvin`
- Please sign up on class website
  - Enrollment key: ece1746
- Seminar style course
  - Lots of advanced material
- Assignments, final exams - nope
- Quizzes - 4
- Project

What is a Distributed System?

- An integrated computing or information facility, that is
  - Built out of many computers
  - That operate concurrently
  - That are physically distributed
    - Have their own failure modes
  - Have independent clocks
  - But are linked by a network

Why do we have them?

- People are distributed but need to work together
- Hardware needs to be close to people
- Information is distributed, but needs to be shared
    - Issue of trust
- Hardware is distributed, but needs to be shared (resource sharing)

Examples

- Distributed Unix
- Commercial Transaction Systems
- WWW
- NetNews (Unix newsgroups)

Characteristics of Distributed Systems

Heterogeneity in Resources, Env

- Resource = hardware, networks
- Environment = OS, languages, implementation
- Data type representation, byte ordering, API, protocols
- Middleware - software layer that tries to hide these differences with uniform computational model
- Virtual machines - making code executable on any hardware, provide uniform environment

Openness

- System can be extended and re-implemented
- Interfaces published
- Uniform mechanism to access resources
- Heterogeneous h/w and software, provided that they conform to the specification

Concurrency

- Many users, hence many concurrent pieces of work
- Servers need to be responsible for many clients, so they need to be concurrent
- Many computers, hence many resources available for parallelism
- Concurrency allows parallelism

Scalability

- Resources
  - Adding physical resources
  - Adding software resources
- Users
  - Performance degradation
- Challenge between distributing versus centralization of physical and software resources

Security

- Confidentiality: protection against disclosure to unauthorized individuals
  - Secrecy
  - Authentication
- Integrity: protection against alteration
- Availability: protection against interference with accessing resources
- Mobile code execution

Fault Tolerance and Availability

- Partial failures
  - Detect failure
  - Mask failure
  - Tolerate failure
    - Hardware redundancy
- Software recovery (rollback)

Design Issues in Distributed Systems

Naming

- We need names in order to use resources
- User-friendly names vs. internal (machine) names
- Issues in designing a namespace
  - Scale
  - Implementation of name lookup

Communication

- How does one computer "talk to" another?
- Latency vs. bandwidth
- Software Overhead
  - High-level programming model with low overhead!

Caching and Replication

- Alternative to remote access
- Distribution vs. centralization issue

Workload Allocation

- Load sharing
- Load balancing

Consistency Maintenance

- Updates
  - Immutable objects!
- Clocks
- Replicas

Exception and failure

- Exception: within specification of object
- Failure: outside specification of object

Transparency

Something hidden from the client

- Access Transparency
  - NFS vs ftp
- Location Transparency
  - Domain name vs IP address
- Concurrency Transparency
  - From whom is concurrency concealed? From the programmer? Or from the client?
- Replication Transparency

Transparency (2)

- Failure Transparency
  - Hide faults
- Migration or Mobility Transparency
  - Conceals movement of resources
- Performance Transparency
  - Allows system to be reconfigured to improve performance
- Scaling Transparency
  - Users

Quality of Service

- Performance
  - Throughput
  - Latency
  - Jitter
- Reliability
- Security

Should distributed systems behave like centralized systems?

Differences

- Latency
  - 4-5 orders magnitude
  - *Object migration?*
- Memory access
  - Local vs. remote address space
  - *DSM? References instead of pointers?*
- Partial failure
  - Impossible to mask!
  - No global state
  - Independent failures possible
  - *Interfaces should expose failure!*