1. INTRODUCTION

+ 1.1. DEFINITION OF A DISTRIBUTED SYSTEM

+ 1.2. GOALS
  + 1.2.1. Making Resources Accessible
  ++ 1.2.2. Distribution Transparency
  ++ Types of Transparency
  ++ Degree of Transparency
  o 1.2.3. Openness
    o Separating Policy from Mechanism
  ++ 1.2.4. Scalability
  ++ Scalability Problems
  ++ Scaling Techniques
  ++ 1.2.5. Pitfalls

+ 1.3. TYPES OF DISTRIBUTED SYSTEMS
  + 1.3.1. Distributed Computing Systems
    + Cluster Computing Systems
    + Grid Computing Systems
  + 1.3.2. Distributed Information Systems
    + Transaction Processing Systems
    + Enterprise Application Integration
  + 1.3.3. Distributed Pervasive Systems
    o Home Systems
    o Electronic Health Care Systems
    + Sensor Networks

o 1.4. SUMMARY

2. ARCHITECTURES

++ 2.1. ARCHITECTURAL STYLES

++ 2.2. SYSTEM ARCHITECTURES
  ++ 2.2.1. Centralized Architectures
  ++ Application Layering
  ++ Multitiered Architectures
  + 2.2.2. Decentralized Architectures
    ++ Structured Peer-to-Peer Architectures
    ++ Unstructured Peer-to-Peer Architectures
    o Topology Management of Overlay Networks
    o Superpeers
  + 2.2.3. Hybrid Architectures
    + Edge-Server Systems
    + Collaborative Distributed Systems

++ 2.3. ARCHITECTURES VERSUS MIDDLEWARE
  ++ 2.3.1. Interceptors
  + 2.3.2. General Approaches to Adaptive Software
2.3.3. Discussion

2.4. SELF-MANAGEMENT IN DISTRIBUTED SYSTEMS
++ 2.4.1. The Feedback Control Model
o 2.4.2. Example: Systems Monitoring with Astrolabe
+ 2.4.3. Example: Differentiating Replication Strategies in Globule
o 2.4.4. Example: Automatic Component Repair Management in Jade

2.5. SUMMARY

3. PROCESSES

3.1. THREADS
++ 3.1.1. Introduction to Threads
++ 3.1.2. Thread Usage in Nondistributed Systems
++ 3.1.3. Thread Implementation

3.2. VIRTUALIZATION
++ 3.2.1. The Role of Virtualization in Distributed Systems
++ 3.2.2. Architectures of Virtual Machines

3.3. CLIENTS
++ 3.3.1. Networked User Interfaces
++ 3.3.2. Thin-Client Network Computing
++ 3.3.3. Compound Documents
++ 3.3.4. Client-Side Software for Distribution Transparency

3.4. SERVERS
++ 3.4.1. General Design Issues
++ 3.4.2. Server Clusters
++ 3.4.3. General Organization
++ 3.4.4. Distributed Servers
++ 3.4.5. Managing Server Clusters
++ 3.4.6. Common Approaches
++ 3.4.7. Example: PlanetLab

3.5. CODE MIGRATION
++ 3.5.1. Approaches to Code Migration
++ 3.5.2. Reasons for Migrating Code
++ 3.5.3. Models for Code Migration
++ 3.5.4. Migration and Local Resources
++ 3.5.5. Migration in Heterogeneous Systems

3.6. SUMMARY
4. COMMUNICATION

4.1. FUNDAMENTALS

4.1.1. Layered Protocols
- Lower-Level Protocols
- Transport Protocols
- Higher-Level Protocols
- Middleware Protocols

4.1.2. Types of Communication

4.2. REMOTE PROCEDURE CALL

4.2.1. Basic RPC Operation
- Conventional Procedure Call
- Client and Server Stubs

4.2.2. Parameter Passing
- Passing Value Parameters
- Passing Reference Parameters

4.2.3. Asynchronous RPC

4.2.4. Example: DCE RPC
- Introduction to DCE
- Goals of DCE RPC
- Writing a Client and a Server
- Binding a Client to a Server
- Performing an RPC

4.3. MESSAGE-ORIENTED COMMUNICATION

4.3.1. Message-Oriented Transient Communication
- Berkeley Sockets
- The Message-Passing Interface (MPI)

4.3.2. Message-Oriented Persistent Communication
- Message-Queuing Model
- General Architecture of a Message-Queuing System
- Message Brokers
- A Note on Message-Queuing Systems

4.3.3. Example: IBM's WebSphere Message-Queuing System
- Overview
- Channels
- Message Transfer
- Managing Overlay Networks

4.4. STREAM-ORIENTED COMMUNICATION

4.4.1. Support for Continuous Media
- Data Stream

4.4.2. Streams and Quality of Service
- Enforcing QoS

4.4.3. Stream Synchronization
- Synchronization Mechanisms

4.5. MULTICAST COMMUNICATION

4.5.1. Application-Level Multicasting
4.5.2. Gossip-Based Data Dissemination

Information Dissemination Models
Removing Data
Applications

4.6. SUMMARY

5. NAMING

5.1. NAMES, IDENTIFIERS, AND ADDRESSES

5.2. FLAT NAMING

5.2.1. Simple Solutions
Broadcasting and Multicasting
Forwarding Pointers

5.2.2. Home-Based Approaches

5.2.3. Distributed Hash Tables
General Mechanism
Exploiting Network Proximity

5.2.4. Hierarchical Approaches

5.3. STRUCTURED NAMING

5.3.1. Name Spaces

5.3.2. Name Resolution
Closure Mechanism
Linking and Mounting
Implementation of Name Resolution

5.3.3. The Implementation of a Name Space
Name Space Distribution
Implementation of Name Resolution

5.3.4. Example: The Domain Name System
The DNS Name Space
DNS Implementation
Decentralized DNS Implementations

5.4. ATTRIBUTE-BASED NAMING

5.4.1. Directory Services

5.4.2. Hierarchical Implementations: LDAP
Decentralized Implementations
Mapping to Distributed Hash Tables
Semantic Overlay Networks

5.5. SUMMARY

6. SYNCHRONIZATION

6.1. CLOCK SYNCHRONIZATION

6.1.1. Physical Clocks

6.1.2. Global Positioning System
7.3.4. Read Your Writes
7.3.5. Writes Follow Reads

7.4. REPLICA MANAGEMENT
7.4.1. Replica-Server Placement
7.4.2. Content Replication and Placement
    Permanent Replicas
    Server-Initiated Replicas
    Client-Initiated Replicas
7.4.3. Content Distribution
    State versus Operations
    Pull versus Push Protocols
    Unicasting versus Multicasting

7.5. CONSISTENCY PROTOCOLS
7.5.1. Continuous Consistency
    Bounding Numerical Deviation
    Bounding Staleness Deviations
    Bounding Ordering Deviations
7.5.2. Primary-Based Protocols
    Remote-Write Protocols
    Local-Write Protocols
7.5.3. Replicated-Write Protocols
    Active Replication
    Quorum-Based Protocols
7.5.4. Cache-Coherence Protocols
7.5.5. Implementing Client-Centric Consistency
    A Naive Implementation
    Improving Efficiency

7.6. SUMMARY

8. FAULT TOLERANCE
8.1. INTRODUCTION TO FAULT TOLERANCE
8.1.1. Basic Concepts
8.1.2. Failure Models
8.1.3. Failure Masking by Redundancy

8.2. PROCESS RESILIENCE (SEE NEW SLIDES)
8.2.1. Design Issues
    Flat Groups versus Hierarchical Groups
    Group Membership
8.2.2. Failure Masking and Replication
8.2.3. Agreement in Faulty Systems
8.2.4. Failure Detection

8.3. RELIABLE CLIENT-SERVER COMMUNICATION
8.3.1. Point-to-Point Communication
8.3.2. RPC Semantics in the Presence of Failures
++ Client Cannot Locate the Server
++ Lost Request Messages
++ Server Crashes
++ Lost Reply Messages
++ Client Crashes

8.4. RELIABLE GROUP COMMUNICATION
8.4.1. Basic Reliable-Multicasting Schemes
8.4.2. Scalability in Reliable Multicasting
8.4.3. Atomic Multicast
++ Virtual Synchrony
++ Message Ordering
skip Implementing Virtual Synchrony

8.5. DISTRIBUTED COMMIT
8.5.1. Two-Phase Commit
skip 8.5.2. Three-Phase Commit

8.6. RECOVERY
8.6.1. Introduction
skip Stable Storage
8.6.2. Checkpointing
8.6.3. Message Logging
8.6.4. Characterizing Message-Logging Schemes
8.6.5. Recovery-Oriented Computing

8.7. SUMMARY

9. SECURITY
9.1. INTRODUCTION TO SECURITY
9.1.2. Example: The Globus Security Architecture
9.1.3. Design Issues
9.1.4. Focus of Control
9.1.5. Layering of Security Mechanisms
9.1.6. Distribution of Security Mechanisms
9.1.7. Simplicity
9.1.8. Cryptography
9.1.9. Symmetric Cryptosystems: DES
9.1.10. Public-Key Cryptosystems: RSA
9.1.11. Hash Functions: MD5

9.2. SECURE CHANNELS
9.2.1. Authentication
9.2.2. Authentication Based on a Shared Secret Key
Implementation of Object References

10.3.2. Static versus Dynamic Remote Method Invocations

10.3.3. Parameter Passing

+ 10.3.4. Example: Java RMI
  + The Java Distributed-Object Model
  + Java Remote Object Invocation

skip 10.3.5. Object-based Messaging

10.4. NAMING

10.4.1. CORBA Object References

10.4.2. Globe Object References

skip 10.5. SYNCHRONIZATION

+ 10.6. CONSISTENCY AND REPLICATION
  + 10.6.1. Entry Consistency
  skip Replication Frameworks
  + 10.6.2. Replicated Invocations

skip 10.7. FAULT TOLERANCE
  skip 10.7.1. Example: Fault-Tolerant CORBA
  skip An Example Architecture
  skip 10.7.2. Example: Fault-Tolerant Java

skip 10.8. SECURITY
  skip 10.8.1. Example: Globe
  skip Overview
  skip Secure Method Invocation
  skip 10.8.2. Security for Remote Objects

skip 10.9. SUMMARY

11. DISTRIBUTED FILE SYSTEMS

11.1. ARCHITECTURE
  11.1.1. Client-Server Architectures
  11.1.2. File System Model
  11.1.3. Cluster-Based Distributed File Systems
  11.1.4. Symmetric Architectures

skip 11.2. PROCESSES

skip 11.3. COMMUNICATION
  skip RPCs in NFS
  skip The RPC2 Subsystem
  skip File-oriented Communication in Plan 9

skip 11.4. NAMING
  skip 11.4.1. Naming in NFS
  skip File Handles
skip          Automounting
skip  11.4.2. Constructing a Global Name space

11.5. SYNCHRONIZATION
o Semantics of File Sharing
skip File Locking
o Sharing Files in Coda

+  11.6. CONSISTENCY AND REPLICATION
+  11.6.1. Client-side Caching
+ Caching in NFS
+ Client-side caching in Coda
skip Client-side Caching for Portable Devices
+  11.6.2. Server-side Replication
+ Server Replication in Coda
skip  11.6.3. Replication in Peer-to-Peer File Systems
skip Unstructured Peer-to-Peer Systems
skip Structured Peer-to-Peer Systems
skip  11.6.4. File Replication in Grid Systems

skip  11.7. FAULT TOLERANCE
skip  11.7.1. Handling Byzantine Failures
skip  11.7.2. High Availability in Peer-to-Peer Systems

skip  11.8. SECURITY
skip  11.8.1. Security in NFS
skip Secure RPCs
skip Access Control
skip  11.8.2. Decentralized Authentication
skip  11.8.3. Secure Peer-to-Peer File-Sharing Systems
skip Secure Lookups in DHT-based Systems
skip Secure Collaborative Storage

skip  11.9. SUMMARY

+  12. DISTRIBUTED WEB-BASED SYSTEMS

+  12.1. ARCHITECTURE
+  12.1.1. Traditional Web-based Systems
+ Web Documents
++ Multitiered Architectures

12.1.2. Web Services
o Web Services Fundamentals
o Web Services Composition and Coordination

o  12.2. PROCESSES
skip  12.2.1. Clients
+  12.2.2. The Apache Web Server
+  12.2.3. Web Server Clusters
12.3. COMMUNICATION
  12.3.1. Hypertext Transfer Protocol
  12.3.1.1. HTTP Connections
  12.3.1.2. HTTP Methods
  12.3.1.3. HTTP Messages
  12.3.2. Simple Object Access Protocol

12.4. NAMING

12.5. SYNCHRONIZATION

++ 12.6. CONSISTENCY AND REPLICATION
++ 12.6.1. Web Proxy Caching
++ 12.6.2. Replication for Web Hosting Systems
++ 12.6.3. Replication of Web Applications

++ 12.7. FAULT TOLERANCE

12.8. SECURITY

12.9. SUMMARY

13. DISTRIBUTED COORDINATION-BASED SYSTEMS

13.1. INTRODUCTION TO COORDINATION MODELS

13.2. ARCHITECTURES
  13.2.1. Overall Approach
  13.2.2. Traditional Architectures
  13.2.3. Peer-to-Peer Architectures

13.2.4. Mobility and Coordination

13.3. PROCESSES

13.4. COMMUNICATION
  13.4.1. Content-based Routing
  13.4.2. Supporting Composite Subscriptions

13.5. NAMING
  13.5.1. Describing Composite Events
  13.5.2. Matching Events and Subscriptions