


EEC-681/781

Distributed Computing Systems

Lecture 1


Wenbing Zhao
Department of Electrical and Computer Engineering
Cleveland State University
wenbing@ieee.org



Outline

- Syllabus
- Overview of Distributed Systems
 - Definition
 - Design Goals


22 January 2006 EEC686/785 Wenbing Zhao



Course Objectives

- Become familiar with principles of distributed systems
 - What is a distributed system
 - What are the basic principles of distributed systems
 - What are the challenges on building distributed systems
- Get hands-on experiences
 - Learn basic distributed computing techniques

22 January 2006 EEC686/785 Wenbing Zhao



Prerequisite

- Operating system principles
 - Processes, scheduling, file systems, etc.
- Computer networks
 - TCP, UDP, IP, Ethernet, etc.
- Java programming language
 - At least you should know how to write a Hello World program
 - You don't have to be a Java expert

22 January 2006 EEC686/785 Wenbing Zhao

Grading Policy

- Midterms 40% (20% each)
- 7 labs (approximate) (40%)
 - Mandatory attendance, lab report due at the end of each lab
 - Grade based on effort, instead of result
- Course project:
 - Implementation track (30%)
 - Theory track (20%)

Outline of Lectures

- Overview of distributed systems
- System models & design principles
- Communication protocols & methods
- Concurrency & multithreading
- Naming
- Synchronization
- Practical distributed systems

Outline of Labs

- Lab 0 – Getting familiar with Linux
- Lab 1 – Java RMI (Remote method invocation)
- Lab 2 – ActiveMQ (Java Message Service)
- Lab 3 – Java JNDI (Naming and Directory)
- Lab 4 – MySQL/JDBC (Transactions)
- Lab 5 – Apache Axis (Web services)
- Lab 6 – JXTA Shell (Peer to peer systems)

Outline of Course Project

- Two alternative tracks are available
 - **Implementation track:** build an interesting distributed system/application
 - **Theory track:** comprehensive survey of a particular topic of your interest

Implementation Track

- Team of up to two (2) persons
- You define the project you want to work on
 - Problem definition
 - Approach
 - Related work
- Deliverables
 - Progress report to help you keep good pace
 - Final project report
 - ◆ Design documentation
 - ◆ Source code of your system/application
 - ◆ Performance measurement and analysis
 - Demonstration and presentation

Theory Track

- Must be done individually
- You define the project you want to work on
- Deliverables
 - Progress report to help you keep good pace
 - Final project report
 - ◆ Problem definition
 - ◆ Survey of current state of the art
 - ◆ Open issues and future research directions
 - Presentation of your survey

What You Should Not Do

- Steal other's project and use it as yours
- Join a team but do not work on it at all
- Why it is not a good idea to do so?
 - If you can find it from the Internet, I can find it too => You get 0 on the course project
 - During presentation, I will ask you questions => Your grade on the project will be reduced significantly if I determine you don't know what you are talking about
 - You lose the chance of learning something practical and useful for your future career

What You Should Do

- Make your own design, code your own system
- Write in your own words and create your own power point slides
 - Don't copy and paste => I can detect it easily
- If you are on a team, make your best contribution to the project
 - Different grade might be assigned to different team members
- Start early and don't wait until the last week of the semester to start
- Communicate with me often and ask help

Why It Is a Good Idea to Take the Implementation Track 13

- **You can get extra credit worth 10% of the course**
- You learn useful technical skills for building distributed systems
- Put the project on your resume => makes it easier for you to find a good job
- **I will recommend the student who did the best project to the Department Chair for assistantship**
- It is not that hard – it can be extensions of the labs
- **I am here to help – talk to me whenever you are stuck or frustrated**

22 January 2006

EEEC686/785

Wenbing Zhao

Reference Texts 14

- Andrew S. Tanenbaum and Marten van Steen:
 - [“Distributed Systems: Principles and Paradigms”](#)
 - ◆ Prentice-Hall, 2002
- George Coulouris, Jean Dollimore and Tim Kindberg:
 - [“Distributed Systems: Concepts and Design”](#)
 - ◆ 3rd Edition, Addison-Wesley, 2001
- Sape J. Mullender:
 - [“Distributed Systems”](#)
 - ◆ 2nd Edition, ACM Press, 1993

22 January 2006

EEEC686/785

Wenbing Zhao

Instructor Information 15

- Instructor:
 - Dr. Wenbing Zhao
 - ◆ e-mail: wenbing@ieee.org
 - ◆ Lecture hours: T Th 4:00-5:50pm
 - ◆ Office hours: *MW 4:00-6:00pm* and by appointment
- Course Web site:
 - URL: http://academic.csuohio.edu/zhao_w/teaching/EEEC681-S06/eeec681.htm
 - Lecture nodes will be posted (with graphs removed)

22 January 2006

EEEC686/785

Wenbing Zhao

Your Feedback Is Appreciated 16

- You are welcome to send me your feedback regarding my teaching
 - Constructive criticism, or
 - Just let out your frustration
- Anonymous email account
 - teachingsu@gmail.com
 - Password:

22 January 2006

EEEC686/785

Wenbing Zhao

Distributed Computing Systems Lab

- Location: SH 306
- The lab consists of 8 computers each with dual Intel Xeon processor running Suse 10.0 Linux
- Accessible from the Internet through the server
 - Externally visible domain name: dcs.csuohio.edu
- The lab itself runs many distributed systems, e.g.,
 - NFS (networked file systems)
 - NIS (network information service)
- Avoid using the lab during the following hours
 - Th 9:00am-12:00pm

Overview of Distributed Systems

- Definition
- Design Goals
- Hardware Concepts
- Software Concepts

Distributed Application Examples

- The World Wide Web
- Automated banking systems
- Global positioning systems
- Retail point-of-sale terminals
- Air-traffic control
- Peer-to-peer file sharing systems

Motivation for Distribution

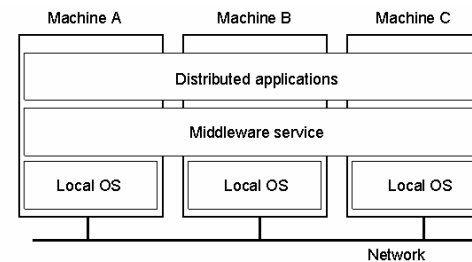
- Share resources
- Personalize environments
- Location independence
- People & information are distributed
- Performance & cost
- Modularity & expandability
- Availability & reliability
- Scalability

Definition of a Distributed System

- **Original:** A collection of independent computers that appear to the users as a single coherent system
- **Modified:** A piece of **software** that ensures a collection of autonomous computers to appear as a single coherent system
 - Autonomous computers connected by a network
 - Software specifically designed to provide an integrated computing facility

Definition of a Distributed System

- Two aspects (1) independent computers and (2) single system ⇒ **middleware**



Definition of a Distributed System

- “You know you have a distributed system when the crash of a computer you’ve never heard of stops you from getting any work done.” – Leslie Lamport
 - Inter-dependencies
 - Shared state
 - Failure handling is difficult

Design Goals

- Connecting users and resources:
 - Reduce cost, increased connectivity and sharing
- Transparency:
 - Users feel like they are using a single-user system
- Openness:
 - Services provided are based on standards
- Flexibility:
 - Separation of policy and mechanisms

Design Goals

- Scalability:
 - Able to support large number nodes, etc.
- Availability:
 - Fault tolerant
- Security:
 - Confidentiality and Integrity

Distribution Transparency

- Access transparency
- Location transparency
- Migration transparency
- Relocation transparency
- Replication transparency
- Concurrency transparency
- Persistency transparency

Access Transparency

- Hide differences in data representation and how a resource is accessed
 - Byte ordering of integer types
 - ◆ Sun Sparc processor: big endian (low order byte is transmitted first)
 - ◆ Intel processor: little endian (high order byte is transmitted first)
 - File systems
 - ◆ Windows uses NTFS, FAT32, FAT16 etc. A file is accessed using the form C:\a folder\file.ext
 - ◆ Linux uses Ext3, Reiser, XFS, Ext2 etc. A file is accessed using the form /home/username/directory/file.ext
- Example: Web application
 - You fetch a Web page, it is displayed in your browser momentarily. It doesn't matter if your system is the same as that of the Web server

Location Transparency

- Location transparency - Hide where a resource is located
 - Logical name, instead of physical name, is used to refer to resources
- Example: WWW
 - You use domain name to refer to a resource, e.g., http://academic.csuohio.edu/zhao_w/teaching/EEEC681-S06/eec681.htm points to the main Web page for this course. You do not need to know where it is physically located
 - In fact, even an IP address is a logical name



Migration Transparency

- Migration transparency - Hide that a resource may move to another location
- This is closely related to location transparency and is enabled by using logical names for resources
- Example: WWW
 - A domain name can be mapped to different IP addresses
 - The same IP address can be reassigned to a server located in a different place



Relocation Transparency

- Relocation transparency - Hide that a resource may be moved to another location while in use
- Example
 - A mobile user can keep using his/her laptop without losing the Internet connection while moving across different cells