CIS 630
Distributed Systems

Fall 2008

Instructor: Matthew Sottile

Logistics

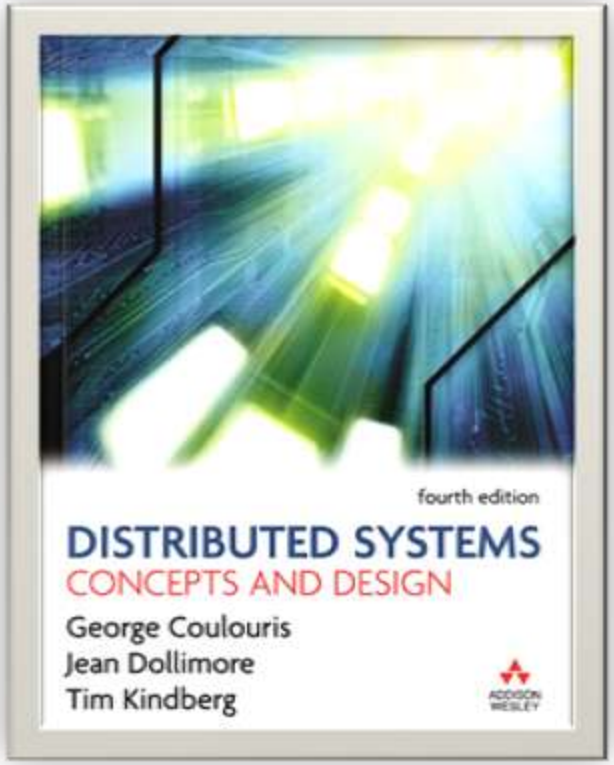
- Meeting time: Tues/Thurs, 1400-1520
- Final meeting: December 8, 1300, here.
- Course web page:
  - http://www.cs.uoregon.edu/classes/08F/cis630/
  - This will grow as the term goes on.

- Office hours:
  - Rm. 203 Deschutes
  - Monday/Wednesday, 3:00-4:30pm
    - Or by appointment – just e-mail me or drop by.
  - My e-mail: matt@cs.uoregon.edu

Textbook

- Distributed Systems: Concepts and Design (4th Edition)
- Coulouris/Dollimore/Kindberg

- You will want to read the chapters well. We will follow the text carefully.
  - Many figures used in slides are provided by book authors to be consistent with text.



Lectures

- Will be primarily slide based, slides posted weekly.
- Interactive discussion important too. Participation is a factor in your grade. Speak up, ask questions, participate.

- Some material will be pulled from sources other than the text book. I will post PDFs for papers, links for web pages, and put relevant books on reserve at the library.
- The course is based on past instances taught by Jan Cuny and Allen Malony. I will draw heavily on those prior instances of the class with updates based on changes to the text and the field at large.

Assignments

- Problem sets (5%)
  - Not graded.
  - Third and seventh week.
- Programming exercise(s) (10%)
  - Java RMI Application
  - Hands-on to get a feel for building a basic D.S.
- Reading summaries (10%)
- Term exam(in class) (25%)
- Term paper (25%)
- Term project (25%)

Reading Assignments / Summary

- You will be given four to five papers on topics in distributed systems.
  - Weekly, starting third week.
- Turn in two-page summaries.
- The goal is to give practice reading research papers.
- This will be useful in getting ready to write the term paper.
- We’ll have discussion time in class when summaries are due to talk about anything you found interesting.

Term Exam

- One and only one exam in the course.
- Will take place Tuesday before Thanksgiving.
- All content up to exam fair game.
  - Lectures
  - Assigned reading in the book
  - Papers that have been assigned

- This is not intended to be terribly painful. If you do the assignments and the readings, you'll be on track.

Term Paper

- One of the main assignments for the course.
- Give you an opportunity to explore some topic of interest.
  - Topics outside the scope of lectures/assignments are OK.
- Provide an experience digging through research literature to learn about a topic.
- Your paper will be presented in class during dead week (week 10).
- See course web page for paper requirements.
  - These will be posted soon.

Term project

- Second main assignment for course.
- Real hands on experience with distributed systems.
- Performed in teams of 3-4 people.
  - Individual efforts must be identified when project turned in.
  - Skills survey and team preferences.
- Deliverables
  - Written report of accomplishments.
  - Demonstration during finals week.
  - Project presentations during final exam period.

Term project (2)

- You will be able to use any of the usual CS department resources. These include:
  - Your office workstations
  - The CS dept. cluster “mist”
  - Workstations in rm. 100 (don’t be disruptive to users if you’re running in the background though…)
- Languages and technologies are your choice.

Experience survey.

- Please fill out the experience survey.
- This will be used for me to gauge the level of experience in the class for programming and project assignments.
- I will use this to assign you to teams for the project.

Tentative Schedule

- **Note**: *One class will be rescheduled or someone else will give it due to travel.*

Week	Date	Topic	Chapters
1	Sep 30, 2008	Intro, Models, Networking	1, 2, 3
	Oct 2, 2008	Intro, Models, Networking	1, 2, 3
2	Oct 7, 2008	IPC, RPC, RMI	4, 5
	Oct 9, 2008	IPC, RPC, RMI	4, 5
3	Oct 14, 2008	Time and global states	11
	Oct 16, 2008	Coordination and agreement	12
4	Oct 21, 2008	Transactions and concurrency	13
	Oct 23, 2008	Transactions and concurrency	13
5	Oct 28, 2008	Distributed transactions	14
	Oct 30, 2008	Distributed transactions	14
6	Nov 4, 2008	Distribution and replication	15
	Nov 6, 2008	Distributed file systems	8
7	Nov 11, 2008	Name services	9
	Nov 13, 2008	Web services	19
8	Nov 18, 2008	Peer-to-peer systems	10
	Nov 20, 2008	Distributed shared memory	18
9	Nov 25, 2008	Term Exam	
	Nov 27, 2008	HOLIDAY	
10	Dec 2, 2008	Paper presentations	
	Dec 4, 2008	Paper presentations	
11	Dec 8, 2008	Project Presentations	

This week

- We’ll start with the fundamentals.

- Reading assignment this week: chapters 1, 2, and 3.

  - Characterization of distributed systems
  - System models
  - Networking and internetworking

Objectives

- Basics of distributed systems with examples.
- Challenges: What issues arise that make distributed systems an interesting topic?
- Common distributed system architecture models.
- Requirements: What requirements drive the design of distributed systems?
- Fundamental models: What submodels describe the fundamental properties of distributed systems?
- Refresh our understanding of the networking infrastructure upon which distributed systems are built.

What is a distributed system?

- A distributed system is one in which hardware and/or software components located at networked computers communicate and coordinate their actions by exchanging messages.

- What are the consequences of this?
  - Concurrency
  - No global clock
  - Independent failures

Consequences

- Concurrency
  - Operations can execute at the same time at each computer. Coordination of these concurrently executing activities is a core topic of this class.
- No global clock
  - Programs communicate by passing messages and occasionally close coordination requires a shared idea of what time things occur. How to deal with time is a fundamental part of distributed systems.
- Independent failures
  - Computers in the system will fail for a variety of reasons. How to deal with this is critical in real distributed systems.

Motivation to build dist. systems

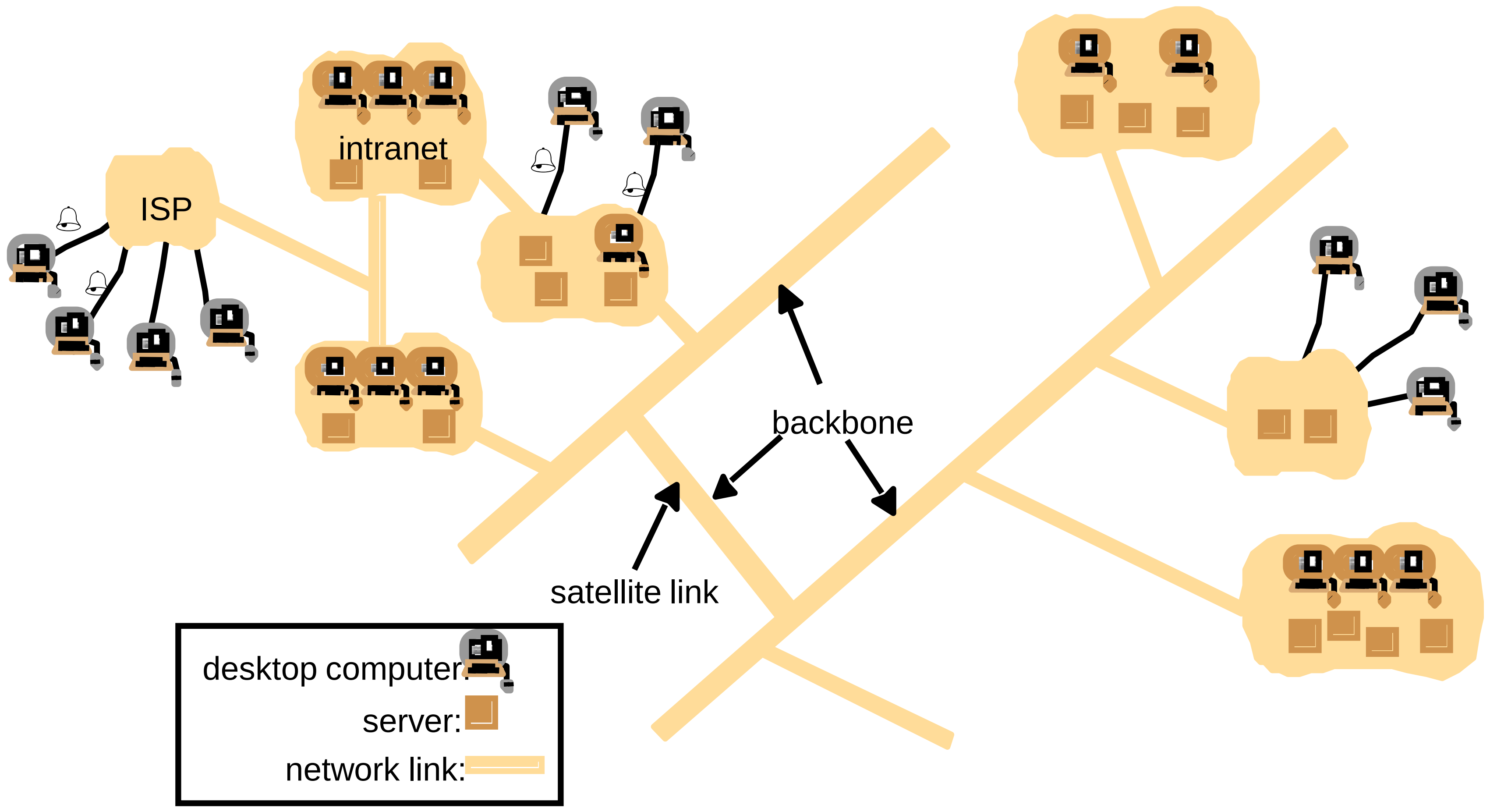
- The key motivation behind distributed systems is resource sharing, where *resource* is a generic term.
  - Examples: CPU time, storage space, bandwidth, software, services, etc…

- Various reasons are behind this sharing.
  - Redundancy and robustness
  - Performance
  - Scalability
  - Economic factors

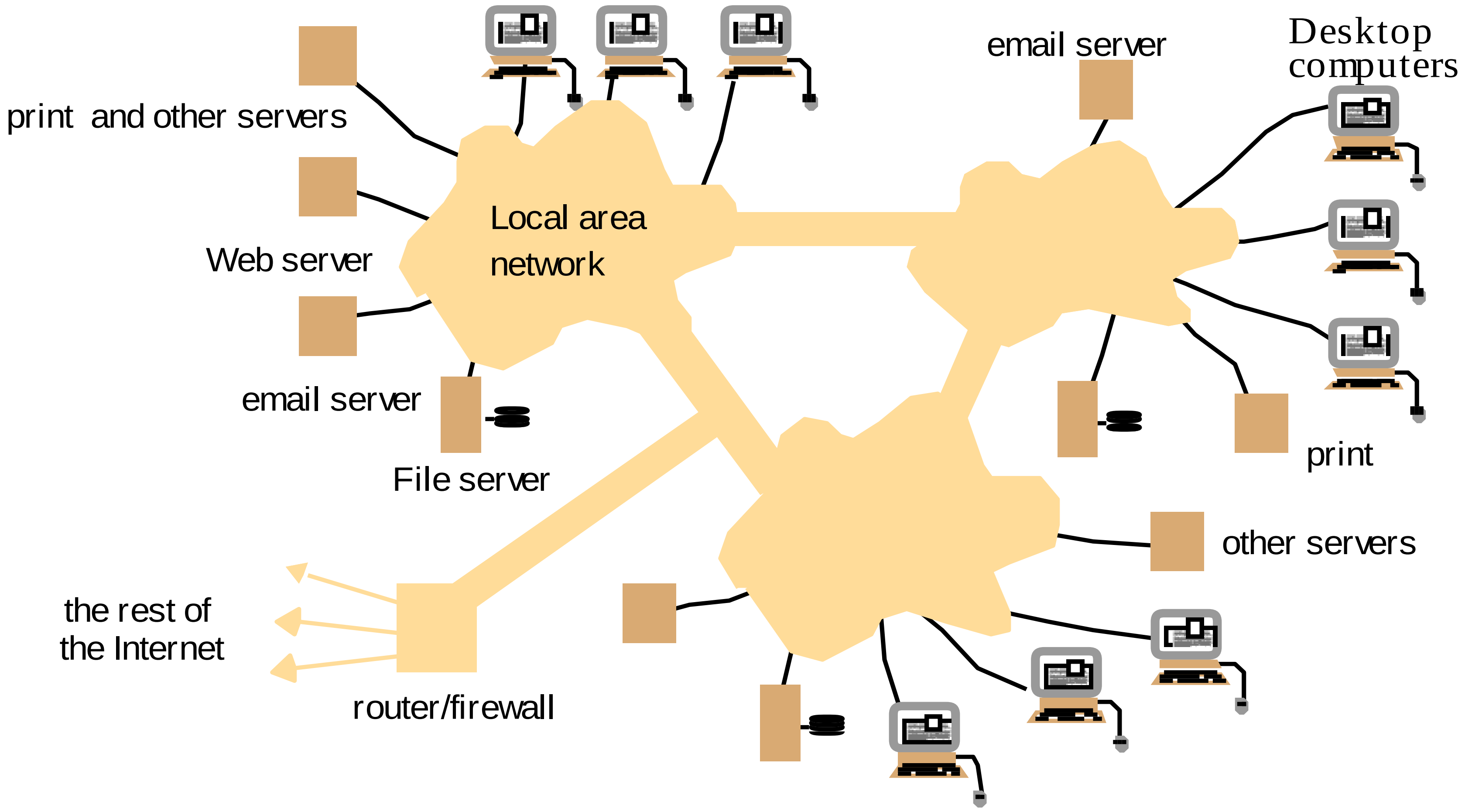
Examples

- The internet
- Intranets
- Mobile and ubiquitous computing

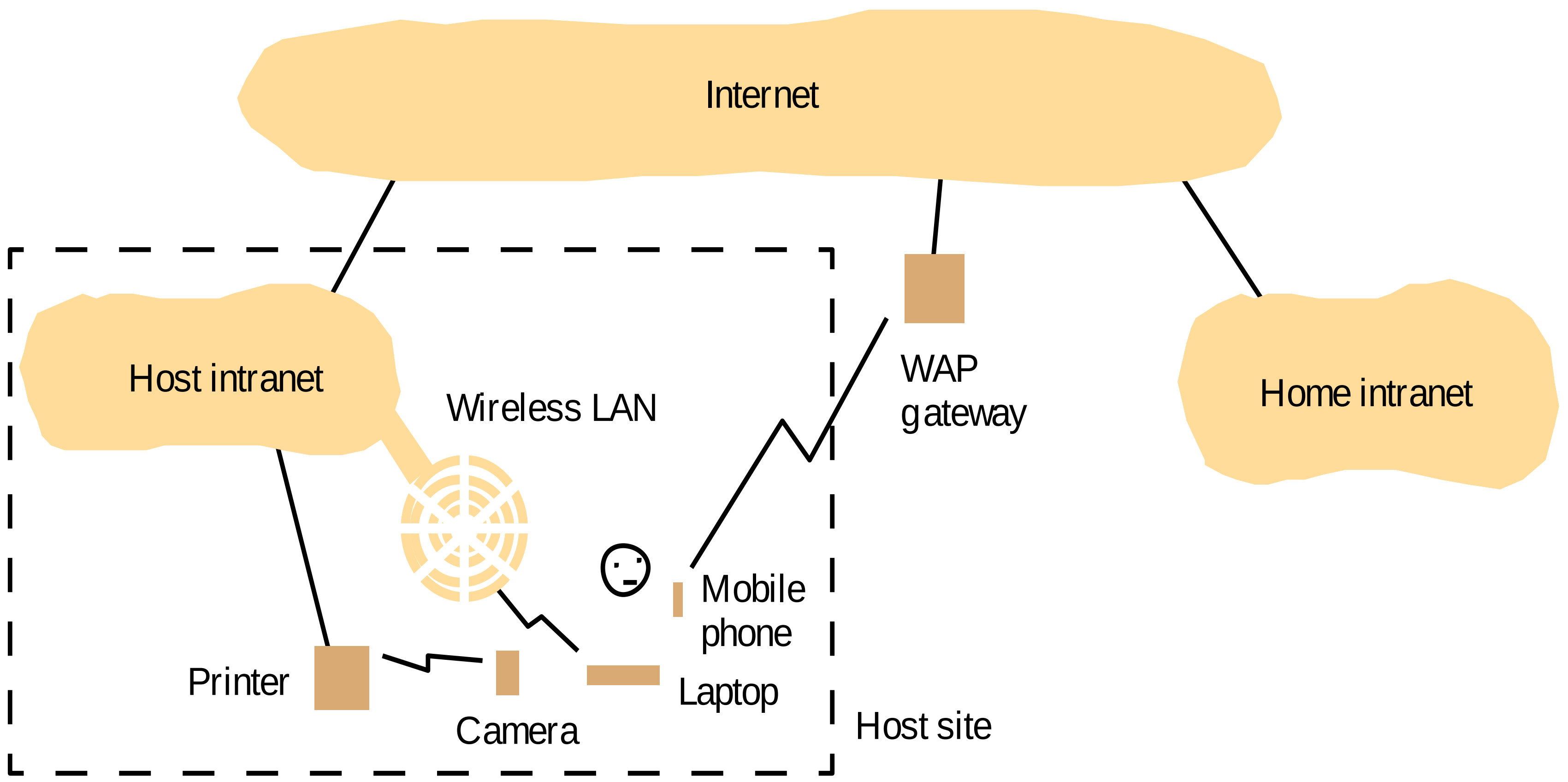
The Internet



Intranets


print and other servers
Web server
email server
File server
Local area network
email server
Desktop computers
print
other servers
the rest of the Internet
router/firewall

Mobile and Ubiquitous computing



Terminology

- **Service**: A distinct part of a computer system that manages a collection of related resources and presents their functionality to users and applications.
  - E.g.: File service, print service.

- **Server**: A running program on a networked computer that accepts requests for services and returns the result of the service (if any) to the client. The client is the computer that submits the request to the server.

Challenges

- Heterogeneity
  - More than one type of hardware or software.
- Openness
  - Extensibility through defined standards and APIs.
- Security
- Scalability
- Failure handling
  - Gracefully deal with components failing.
- Concurrency
- Transparency

Transparencies

- Access transparency
  - Local and remote resources accessed with identical operations.
- Location transparency
  - Resources accessible without knowledge of physical (room number) or network location (IP address).
- Concurrency transparency
  - Multiple processes can operate on resources without interfering with each other.
- Replication transparency
  - Multiple instances of a resource can be accessed for performance or reliability reasons without knowledge of the replicas by users or application programmers.

Transparencies (2)

- Failure transparency
  - Hide faults and allow users and application programs to complete their tasks despite hardware/software failures.
- Mobility transparency
  - Allow movement of resources and clients in a system without affecting the operation of users and programs.
- Performance transparency
  - Allows the system to be reconfigured for performance reasons as loads vary.
- Scaling transparency
  - Allows the system and applications to expand in scale without change to the system structure or application algorithms.