Syllabus **CS 603 Advanced Topics in Distributed Systems** **Spring 1998**

Instructor : Kihong Park
Class : Tue. & Thu. 1:30–2:45 (CS G066)
Office Hours : Tue. & Thu. 2:45–3:45, Wed. 11–noon (CS 220)
E-mail : `park@cs.purdue.edu` (Tel. : 494–7821)

Teaching Assistant : Saumil Shah
Office Hours : TBA (MATH 405)
E-mail : `saumil@cs.purdue.edu` (Tel. : 494–5006)

Course Content : Advanced graduate course in distributed system design and analysis.

Prerequisites : Operating systems (CS 503); computer networks (CS 536); solid background in C, C++ and UNIX (experience with projects and system programming); sound mathematical preparation (calculus, discrete math, differential equations, probability and statistics).

Text Books : Required - *Distributed Operating Systems.* Tanenbaum, Prentice-Hall, 1995. Recommended - *Internetworking with TCP/IP*, Vol. III. Comer & Stevens, Prentice-Hall, 1996; *Distributed Algorithms.* Lynch, Morgan Kaufmann Publ., 1996. The textbooks, including Tanenbaum, are to be viewed as reference materials; the lectures are the main source of information. Attendance is crucial for the successful completion of the course.

Work Load and Grading Policy : Homework assignments (40 %), midterm (10 %), final (15 %), research project (25 %). The research project is due at the end of the semester; project topics will be determined at the halfway point in the semester. Several topics will be suggested, however, students are encouraged to seek out and explore interesting subject matters on their own.

Homework assignments are given out weekly (in total 10); they are due in class the following week. One-day-late assignments are accepted at 20 % penalty; later homeworks will not be accepted. Discussion of initial ideas and approaches regarding homework assignments is fine. *Collaboration* on homeworks (i.e., jointly exploring the specifics of a solution), however, is not allowed. Academic dishonesty is a serious matter and is dealt with in accordance with University policies.

Lab Space : Our main programming and implementation platform will be the Xinu Lab. Accounts, if not already set up, should be applied for with Candy Walters (`clw@cs`, 494–9206, CS 210).

Goal of the Course :

The primary goal of this course is to study the design principles underlying distributed systems—distributed operating systems, network of workstations (NOWs), network computing (e.g., PVM, Java), and other distributed computing environments. The first part of the course will be devoted to understanding the fundamentals of distributed systems design—synchronization/concurrency control, group communication,

scheduling/load balancing, threads, replication/cache consistency, fault-tolerance, distributed shared memory, naming, distributed file systems, and security. Emphasis will be placed on performance issues including trade-offs in design (e.g., computation/communication ratio, user space vs. kernel space implementation), and their analysis.

In the second part of the course, we will study several state-of-the-art research topics including parallel distributed computing, real-time scheduling, cooperative/noncooperative resource allocation algorithms, on-line algorithms for load balancing and communication management, O.S. support for high-speed networking, adaptive algorithms for server selection and caching, fault-tolerance in Internet-like large-scale environments, middleware and network computing environments, and case studies of distributed systems.

Course Outline :

- Introduction.
- Networking overview and client/server design.
- Group communication.
- Physical/logical clocks and their application.
- Mutual exclusion.
- Leader election and fault models.
- Distributed transactions.
- Concurrency control.
- Distributed deadlock detection/prevention.
- Theoretical aspects of consensus problems.
- —*Midterm*—
- Advanced fault models.
- Fault-tolerant distributed computing.
- Load balancing and process migration.
- Noncooperative vs. cooperative resource allocation.
- Distributed scheduling, real-time scheduling.
- Network computing, middleware, and the WWW.
- Distributed caching.
- Distributed file systems and naming.
- Distributed shared memory.
- Case studies.
- —*Final*—