Distributed Systems

Bulletin Description

Prerequisite, COMP 530. Permission of the instructor for students lacking the prerequisite. Design and implementation of distributed computing systems. Inter-process communication, performance, replication, consistency, group communications, example applications.

General Course Info

Term:	Fall 2015
Department:	COMP
Course Number:	734
Section Number:	001
Time:	TR 2:00 – 3:15
Location:	Room SN 150
Website:	http://www.cs.unc.edu/~dewan/734/current/index.html

Instructor Info

Name:	Prof. Prasun Dewan
Office:	FB150
Email:	dewan@cs.unc.edu
Phone:	5906123
Web:	http://www.cs.unc.edu/~dewan
Office Hours:	TR 15:30 – 16:30

Teaching Assistants: None

Textbooks and Resources

I plan to provide notes, PPT slides, and videos on the material I cover accessible from the course home page. These should suffice for the course. They will not be posted on Sakai, which will be used however for submitting programs.

Course Description

This course will provide a practical study of distributed systems. It will be driven by a series of projects, which will demoed on the first day of class.

We will cover the design and implementation of blocking and non-blocking byte communication, object communication, and remote procedure call, relayer-based and p2p communication, causal and atomic broadcast, and

fault tolerance. These are foundational concepts, which are becoming particularly relevant with the emerging areas of cloud computing and distributed games. These concepts will be introduced as layers in a general distributed infrastructure.

At the end of the course, you will have a basic understanding of how distributed software works, the potential uses of this software, and the design and implementation space of distributed abstractions. As a distributed program is also a parallel program, you will also learn and use basic concepts in threads and thread synchronization. Because of the emphasis on implementation, you will gain practice with the use and implementation of advanced software engineering concepts such as layers, generic types, factories, and abstract factories.

The main difference between this course and a distributed theory course is that it will address the use, design and implementation rather than theoretical foundations of distributed programming abstractions. For instance, you will learn how to use, design and implement remote procedure call rather than define its semantics formally. Thus, the difference between this course and a theory course is similar to the difference between a course on programming languages/compilers and courses on models of languages/computation. The course will focus on implementation of distributed abstractions, but before we get to that, we will address their programming and design.

Target Audience

The target audience is students wishing to learn in depth the nature of practical distributed systems.

Prerequisites

The pre-requisites are knowledge of object-oriented programming, data structures, and threads. UNC Comp 401, 410, and 530 cover these topics, respectively UNC Comp 431 (Internet Services and Protocols) is not a prerequisite.

Goals and Key Learning Objectives

As mentioned in the course description, at the end of the course, you will have a basic understanding of how distributed software works, the potential uses of this software, and the design and implementation space of distributed abstractions. As a distributed program is also a parallel program, you will also learn and use basic concepts in threads and thread synchronization.

Because of the emphasis on implementation, you will gain practice with the use and implementation of advanced software engineering concepts such as layers, generic types, factories, and abstract factories. Finally, you will create a layered system that is more sophisticated in many ways than the state of the art, and you are likely to use it for programming future distributed programs

Course Requirements

The students must attend lectures, implement a semester-wide project, and take a midterm and a final exam.

As there is a cumulative project, there is no final exam in this course.

You must submit the source code of your program (with pledge signed) and screens showing executions of the program on test data. You may also do demos at certain stages of your project

Examinations are closed book, notes and program listings; computers and collaboration are not allowed either.

Key Dates

Midterm 1: Tuesday Oct 13th, 2015 (in class)
Final: 12pm on Saturday Dec. 5 (in class)

Grading Criteria

A grade will be assigned based on performance on homework programming assignments, written assignments class work, and exams. Exams will constitute 40% of the grade, homework assignments 50%, and class-work, 10%. There will be a midterm and a final. I reserve the right to apply a 10% fudge factor to give consideration to things such as good class participation, stellar programs, and early submission and extra credit

Course Policies

Students are requires to attend each class unless there are extenuating circumstances. If such circumstances occur, you should access the class material posted for missed classes, and contact classmates to become aware of the announcements that were made.

Assignments are due at 11:59pm on each specified due date. Programs and homework assignments will be penalized 5% for each class session late.

Honor Code

You are encouraged to discuss the assignments with fellow students but required to write/code the solutions/programs individually. Also you cannot use solutions from previous offerings of the course. Not following these rules is a violation of the honor code policy

Course Schedule

If possible, a schedule of topics covered by the course organized by course date or week number.

1. Course Information,
2. Java Non Blocking I/O (NIO)
3. Java Remote Method Invocation
4. General Model of Inter-Process Communication (IPC) Design
5. Overview of Generic IPC Implementation Architecture
6. Byte Communication Drivers
7. Object Communication Layer
8. Remote Procedure Layer
9. P2P and Relayed Communication Layer
10. Broadcast consistent guarantees
11. Fault-Tolerant Broadcast

Disclaimer

The professor reserves to right to make changes to the syllabus, including project due dates and test dates. These changes will be announced as early as possible.