Comp 533: Distributed Systems

Bulletin Description

Prerequisite, COMP 431/530. Permission of the instructor for students lacking the prerequisite. Applications, design, implementation, and performance of distributed abstractions – software abstractions that are layered above the network and are supported by the operating system, programming language, and middleware.

General Course Info
Term: Spring 2017
Department: COMP
Course Number: 533
Section Number: 001

Time: TR 12:30 – 1:45
Location: Room SN 011
Website: http://www.cs.unc.edu/~dewan/533/current/index.html

Instructor Info
Name: Prasun Dewan
Office: FB150
Email: dewan@cs.unc.edu
Phone: 5906123
Web: http://www.cs.unc.edu/~dewan
Office Hours: TR 16:15 – 17:15

Teaching Assistants: None

Textbooks and Resources

There is no text book covering the topics of this course. Notes, PPT slides, and videos on the covered material will be accessible from the course home page. They will not be posted on Sakai, which will be used however for submitting programs and quizzes.

Course Description

This course will provide a practical overview distributed systems. It will be driven by a series of implementation-based projects
We will cover the design, implementation, performance, and applications of abstractions for sharing information among distributed processes. These sit between the network layer and application and include: distributed shared memory, byte and object communication, remote procedure call, and broadcast/multicast sessions. Some of the general issues we will address are naming; synchronization; extensibility; and routing and consistency of broadcast/multicast messages.

We will overview but not focus on any of the specific applications of distributed abstractions such as the Web, distributed data mining, distributed data bases, file systems and distributed collaborative systems, which are covered in other courses. Our focus will be on foundational concepts applicable to all of these application areas. These concepts will be introduced as layers that sit above the OS and networks. The assignments and lectures will have two related goals. The first is to expose you to standard implementation of the covered abstractions in Java environment and other distributed systems and discuss how these abstractions can be used in a realistic programs. The assignments addressing this goal will involve the use of only Java abstractions. The second is to expose you to the implementation of these abstractions. I have developed a Java-based teaching/research testbed, called GIPC (for Generalized Interprocess communication), that provides implementation of these abstractions. In the assignments addressing this goal, you will replace parts of this extendible system to understand implementation concepts. The result of the replacement exercised will be a 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.

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, threads, operating systems, and networking. UNC Comp 401, 410, 530 and 431 cover these topics, respectively.

Goals and Key Learning Objectives

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

Course Requirements

The students must attend lectures and participate in class discussions, take quizzes, implement assignments, and take a midterm and a final exam.

You must submit the source code of your program (with pledge signed) and videos demoing executions of the program on test data.

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

Key Dates

Midterm: Class time, Tuesday Mar 7th, 2017 (in class)
Final: 12pm on Friday May 5 (in class)

Grading Criteria

A grade will be assigned based on performance on homework programming assignments, quizzes, class participation, and exams. Exams will constitute 45% of the grade (midterm 20%, final 25%), homework assignments 40%, and class work (quizzes, class participation) 15%. There will be a midterm and a final. I reserve the right to apply a 5% fudge factor to give consideration to things such as good class participation and stellar programs.

Course Policies

Students are required 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 11:55 pm on each specified due date. Homework assignments will be penalized 5% for each class session late. Quiz submissions will not be accepted after the due date.

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 and Introduction to Distributed Systems
2. Byte Communication: Xinu IPC (Interprocess Communication) Design and Implementation
3. Byte Communication: Pipes, Sockets, Non blocking I/O
4. Object Communication: Java, C# Serializable Objects
5. Synchronous Remote Method Invocation: Java RMI
6. Asynchronous Remote Method Invocation: GIPC RPC
7. IPC Design Space
8. Issues in Serializable Objects: Atomic and Structured Objects
9. Issues in Serializable Objects: DAGs, Graphs, Extensibility
10. issues Remote Method Invocation: Activation Record vs Messages, RPC awareness in applications, RPC vs RMI, Proxy Generation,
11. Issues in Remote Method Invocation: Marshalling vs Serialization, Dispatchers, Name Servers, Synchronous vs. Asynchronous,
12. Group Communication: Relayer-based and P2P Sessions
13. Group Communication: Causal Broadcast
14. Overview of Fault Tolerance

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.