CSE 452/CSE M552: Distributed Systems

Instructor: Tom Anderson (tom@cs)
TAs: Umar Javed (ujaved@cs)
Will Scott (wrs@cs)

Class meetings: MWF 2:30 – 3:20, MGH 241
Section: Thursdays 8:30-9:20 (MGH 235), 9:30 – 10:20 (MGH 235)

Instructor Office Hours: Mondays, 3:30 – 4:30, Allen Center 646
http://www.cs.washington.edu/452/

Motivation: Distributed systems have become central to many aspects of how computers are used, from web applications to e-commerce to content distribution. This senior-level course will cover abstractions and implementation techniques for the construction of distributed systems, including client server computing, the web, cloud computing, peer-to-peer systems, and distributed storage systems. Topics will include remote procedure call, preventing and finding errors in distributed programs, maintaining consistency of distributed state, fault tolerance, high availability, distributed lookup, and distributed security. We believe the best way to learn the material is to implement the ideas presented in the course, and so there will be a substantial programming project. Two versions of the course will be offered simultaneously: CSE 452 to undergraduates, and CSE M552 to students in the fifth year masters program.

Student Deliverables:

Course project to be done in teams of 2-3, with 3 parts (4 for M552); each project assignment includes: design outline and short demo
    Foundational readings in distributed systems (15-20 papers)
Blog entries answering questions about reading (5 for 452, 10 for M552)
Two problem sets
Final exam

Textbook, Papers and Blogs: There is no textbook for this course. Instead, we will assign 15-20 research papers tied to specific lectures; these readings are to be done **before** class, as we will take the papers as a starting point, not the end point of the class discussion.

For each assigned paper, we will post a discussion question, linked off the course web page. Students in both 452 and M552 are required to post a short, **unique** comment or observation to the discussion by 1pm on the day of the class; students can pick which questions they respond to, provided you do 5 (452) or 10 (M552) over the quarter. Valid posts can be to take a position on a design choice in the paper, to discuss the applicability of the paper to an important problem, to point out a problem in some other student's argument, etc. Students are welcome to post additional comments for (small) extra credit. Everyone should read the posted comments before class.

Project: The core of the course is the project: to design and build a fault tolerant peer-to-peer Twitter. The project is to be done in groups of 2-3 people. The project has three pieces (for M552, four pieces), each building on the previous ones, due roughly every three weeks. We need every group to get an early start on the assignments. We will provide you some basic message passing code as well as a framework for simulating and emulating a distributed system to aid in debugging.

The first project assignment is to build a simple client-server Twitter system. We then add crash recovery and high availability to this basic system. M552 students will extend the basic design in some interesting way, e.g., to add support for disconnected operation. CSE 452 students may do so for extra credit.

Approximately 1.5 weeks before each project is due, we will ask you to complete a one page design sketch of your intended solution to that portion of the project; this will be graded credit/no credit, but we will use it to give you timely feedback on your design. We have found that some groups make the assignments more difficult than what is intended, and this is an attempt to catch those problems early.

To help the TA's grade the project, we will hold 15 minute demos with each project group shortly after each turning deadline. This will take place in place of section.

To help provide you time to complete the project, we set aside week 10 as a "**hack week**" with no lectures or sections. We will also automatically grant each group three slip days for the project assignments, for you to use at your discretion.

Regardless of slip days, all assignments must be turned by 5pm, **June 7**. Final project demonstrations will be held at various times on June 7.

Details about the project assignments will be provided in a separate handout.

Problem Sets: There will be two problem sets handed out during the quarter. The problem sets are to be done **individually** and are intended to reinforce the basic material covered in the class, and to help prepare students for the final exam. Problem sets are due in section.

Collaboration/Cheating: This course uses CSE's Academic Misconduct Policy: http://www.cs.washington.edu/students/policies/misconduct/

Final: The final for this class will be held Tuesday, June 11 at 2:30pm.

Grading: Project: 45%; design docs: 5%; problem sets 10%; blogs: 5%; final: 35%

Tentative Syllabus: Readings to be done **before** class

4/1: Introduction

Part 1: Client/Server

4/3 (Guest lecture: Dan Ports), 4/5: Remote procedure call

Google. Introduction to Distributed System Design.

Sandberg et al., Design and Implementation of the SUN Network File System, 1985.

4/4: NO SECTION

4/8, 4/10: Distributed debugging

Lamport, Time, Clocks and the Ordering of Events in a Distributed System, CACM 1978.
(up to, not including, the section on physical clocks)

4/11: One page design document due: client-server

Part 2: Memory consistency and cache coherence

4/12, 4/15: Cache coherence

Adve and Gharacherloo. Shared Memory Consistency Models: A Tutorial. DEC WRL TR, 1995.

4/17, 4/19: Eventual consistency

Terry et al. Managing Update Conflicts in Bayou. SOSP 1995.

4/23: Project part 1 due: client-server

4/25: NO SECTION; project demos

Part 3: Failure recovery and Fault Tolerance

4/22: Transactions

Robert Hagmann. Reimplementing the Cedar File System Using Logging and Group Commit. SOSP 1987.

4/24, 4/26 (guest lecture TBD): Distributed Transactions: 2PC and MVCC

Bernstein, Hadzilacos, and Goodman. Distributed Recovery. Chapter 7 in Concurrency Control and Recovery in Database Systems. (up to, and not including, three phase commit)

5/2: Problem set 1 due: memory coherence
5/2: One page design document due: distributed transactions

4/29, 5/1, 5/3: Paxos

Lamport, Paxos Made Simple, ACM SIGACT News, 2001.

Part 4: The Cloud Software Stack

5/6: Chubby (configuration)

Burrows. The Chubby Lock Service. OSDI 2006.

5/8: GFS (storage)

Ghemawat et al. The Google File System. SOSP 2003.

5/10: BigTable (indexing)

Chang et al. BigTable: Distributed Storage System for Structured Data. OSDI 2006.

5/13 (guest lecture: TBD) MapReduce (computation)

Zaharia et al., Resilient Distributed Datasets, NSDI 2012.

5/14: Project part 2 due: distributed transactions

5/16: NO SECTION; project demos

5/15: Spanner (multiple data centers)

Corbett et al. Spanner: Google's Globally Distributed Database. OSDI 2012.

5/17: Distributed hash tables: Dynamo

DeCandia et al. Dynamo: Amazon's Highly Available Key-Value Store. SOSP 2007.

Part 5: Beyond the Cloud

5/20, 5/22: Peer to Peer

Piatek et al. Do Incentives Build Robustness in BitTorrent? NSDI 2007.

Cheng et al., Serverless Web Services. UW TR.

5/24, 5/29: Distributed security

5/31: Wrapup

Lampson. Hints for Computer System Design. ACM SIGOPS OSR 1983.

5/28: Problem set #2 due: distributed semantics
5/28: Project part 3 design document due: Paxos
5/28: (M552 only) Project part 4 design document due

Hack Week: 6/3 – 6/7

6/6: Project part 3 due: Paxos

6/6: M552 project 4 due

6/7: Project demonstrations

6/11: Final Exam, 2:30pm – 4:20