CSE 552: Distributed and Parallel Systems

Instructor: Tom Anderson (tom@cs)
TA: Lisa Glendenning (lglenden@cs)

MW 10:30 – 11:50
(some weeks WF)
Savery 131 (pending)

Overview: Large scale distributed systems have become pervasive, underlying virtually all widely used services. In tandem, our understanding of how to build scalable, robust, efficient, and secure systems is increasingly well-founded. This class will attempt to bring students to the state of the art in distributed systems research and practice as well as to identify a set of open research problems. A particular focus will be "experiences": what can we learn from other people's experiences in building their own scalable distributed systems?

This iteration of the class will **NOT** cover parallel computing.

Pre-requisites: CSE 451 (or equivalent). CSE 550 recommended. The course listing has a pre-requisite of CSE 551; this will not be enforced (nor will it matter if you haven't had it). However, it is essential that you understand and have experience with how to program with threads before taking this class; at UW, that can be accomplished with CSE 451.

Further, we will assume basic knowledge of distributed systems topics, including: RPC, two-phase commit, distributed time, serializability, and mapreduce. This can be obtained by taking CSE 550 before CSE 552 (that is the normal order). A motivated student will be able to pick up these topics on her own. See **background** below for more information.

Students having taken an undergraduate distributed systems class will find about 25% of the material will overlap; in particular, it is permitted to take both CSE 452 and 552.

Course reading and discussion: A major part of the course will be a group discussion of the various papers. The goal will be to develop your ability to uncover the broader implications of research papers. Most systems research starts with this process: what can one conclude from a research result, beyond what is written by the authors?

Prior to each class, I will post a short list of discussion questions to get the discussion started; you will need to arrive to class with at least **two** interesting, novel points to make about the paper (in answer to one of the discussion questions, or on a separate topic).

In addition, a group of three students will lead the initial discussion of each paper, starting with a short recap of the principal results.

Problem Sets: Three to four problem sets will be assigned during the quarter to verify your understanding of the concepts being presented in the papers. A thorough understanding of the papers is a pre-requisite of being able to understand other work in the area, or to put your own research in an appropriate context.

Research Project and Presentation: An independent research project is required, on a topic of your choosing, with team members of your choosing. Projects can relate to any distributed systems topic, and they area encouraged to be related to your own research. I will hand out a list of possible topics for those needing a starting point. A strict requirement is that every project must have a quantitative result – in other words, purely paper designs are not sufficient.

The project will be due in five steps:

a. Initial one page project proposal, due Oct 11
b. Three page outline, including a complete introduction, due Nov 1
c. Five page version, including a complete discussion of related work, due Nov 22
d. Final research project papers, due Tuesday, December 10
e. Final research project presentation, due Wednesday, December 11

Final: The final will be **oral**, 15 minutes per student, scheduled on December 9-10. An oral final is an efficient way for me to determine a student's level of understanding of the material, as it allows me to direct questions to the frontier of a student's knowledge. Questions will be drawn from the required portions of the reading list.

Grading: Class preparation: 15%; problem sets: 15%; research project: 40%; oral exam: 30%.

Background: Please read these during the first week or two if you are unfamiliar with them. The first week's problem set refers to the material in these papers.

Birrell and Nelson, Implementing Remote Procedure Call, ACM TOCS 1984.

Google, Introduction to Distributed Systems Design
http://www.hpcs.cs.tsukuba.ac.jp/~tatebe/lecture/h23/dsys/dsd-tutorial.html

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

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

Dean and Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. OSDI 2004.

Mockapetris, The Domain Name System, SIGCOMM 1998

Schedule Anomalies: The class will meet on **Friday** during three weeks of the quarter when meeting on Monday is impossible for various reasons. Location TBD.

No class October 21; October 25 instead
No class November 4, 6; November 8 instead
No class November 11; November 15 instead

Syllabus:

1. (Sept 25) Introduction: SaaS/SOA/WSDL/SOAP/AJAX/REST

Yegge, Lessons for Google from Amazon's Service Oriented Architecture
https://plus.google.com/112678702228711889851/posts/eVeouesvaVX

Freedman. Experiences with CoralCDN. NSDI 2010.

2. (Sept 30) Programming frameworks (1): Hank to lead discussion

Jul et al., Fine-Grained Mobility in the Emerald System, ACM TOCS 1988.

Liskov, Distributed Programming in Argus, CACM 1988.

Optional:
Liu et al., Fabric: A Platform for Secure Distributed Computation and Storage, SOSP 2009.
Birrell et al., Network objects, SOSP 93

3. (Oct 2) Programming frameworks (2)

Eugster et al., The Many Faces of Publish/Subscribe, ACM Computing Surveys, 2003

Adya et al., Thialfi: A Client Notification Service for Internet Scale Applications, SOSP 2011.

4. (Oct 7) Cache Coherence (1): Mike Dahlin to lead discussion

J. Ousterhout, "The Role of Distributed State," CMU Computer Science: A 25th Anniversary Perspective, R. Rashid ed., ACM Press, 1991, pp. 199-217.

Anderson et al., Serverless Network File Systems, SOSP 1995.

Optional:
Ghemawat et al., The Google File System, SOSP 2003.

5. (Oct 9) Cache Coherence (2)

Nishtala et al., Scaling Memcache at Facebook, NSDI 2013
Ports et al., Transactional Consistency and Automatic Management in an Application Data Cache, OSDI 2010

Optional: Adve and Gharacherloo, Shared Memory Consistency Models: A Tutorial.

6. (Oct 14) Distributed vs. Global State

Chandy and Lamport. Distributed Snapshots: Determining Global States of Distributed Systems, TOCS 1985.

Optional:
Fagin et al., Common Knowledge Revisited.
Schneider, Implementing Fault Tolerant Replicated Services: A Tutorial, Computing Surveys

7. (Oct 16) Paxos

Lamport, Paxos Made Simple

NO CLASS OCT 21

8. (Oct 23) Paxos Implemented

Liskov, Viewstamped Replication Revisited
Bolosky et al., Paxos RSM as the Basis of a High Performance Data Store, NSDI 2011

Optional:
Mazieres, Paxos Made Practical
van Renesse, Paxos Made Moderately Complex
Chandra et al., Paxos Made Live: An Engineering Perspective

9. (Oct 25) Byzantine Fault Tolerance

Castro and Liskov, Practical Byzantine Fault Tolerance, SOSP 1999.
Clement et al., UpRight Cluster Services, SOSP 2009.

Optional:
Liskov, From Viewstamped Replication to Byzantine Fault Tolerance
http://www.pmg.lcs.mit.edu/papers/vr-to-bft.pdf
Oceanstore/Pond
Zyzzyva

10. (Oct 28) Serializability

Brewer, CAP Twelve Years Later: How the "Rules" Have Changed, IEEE Computer

Gun Sirer, Consistency Alphabet Soup
http://hackingdistributed.com/2013/03/23/consistency-alphabet-soup/

Gun Sirer, Broken By Design: MongoDB Fault Tolerance
http://hackingdistributed.com/2013/01/29/mongo-ft/

Optional: Belaramani et al., PRACTI Replication, NSDI 2006.

11. (Oct 30) Storage and Lookup (1)

Schroeder et al., Experience with Grapevine: The Growth of a Distributed System, TOCS 1984.

Chang et al., BigTable: A Distributed Storage System for Structured Data, TOCS 2008.

NO CLASS NOV 4, 6

12. (Nov 8) Storage and Lookup (2)

Escriva et al., Hyperdex: A Distributed, Searchable Key-Value Store, SIGCOMM 2012

NO CLASS NOV 11

13. (Nov 13) Storage and Lookup (3): Bill Bolosky/Jon Howell guest

Bolosky et al., The Farsite Project: A retrospective, OSR 2007

14. (Nov 15) SQL or not

DeWitt, Mapreduce is a major step backwards.
http://homes.cs.washington.edu/~billhowe/mapreduce_a_major_step_backwards.html

Power and Li. Piccolo: Building Fast, Distributed Programs with Partitioned Tables, OSDI 2010.

Others: Gonzalez et al., PowerGraph: Distributed Graph-Parallel Computation on Natural Graphs, OSDI 2012.
Yu et al., Dryad-LINQ: DryadLINQ: A System for General-Purpose Distributed Data-Parallel Computing Using a High-Level Language, OSDI 2008.

15. (Nov 18) Beyond the data center (1)

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

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

Optional:
Megastore
PNUTs

16. (Nov 20) Beyond the data center (2)

Calder et al., Windows Azure Storage, SOSP 2011.

Glendenning et al., Scalable Consistency in Scatter, SOSP 2009

17. (Nov 25) Weakly connected systems

Terry et al., Managing Update Conflicts in Bayou, a Weakly Connected Replicated Storage System, SOSP 1995

Linus Torvalds, Tech Talk: Linus Torvalds on git, 2007.

Optional:
Petersen et al., Flexible Update Propagation for Weakly Consistent Replication, SOSP 97.
Garcia-Molina, Ullman, Widom, 18.1-18.3, 18.8-18.9 (database consistency levels)

18. (Nov 27) Resource Allocation

Fu et al., SHARP: An Architecture for Secure Resource Peering, SOSP 2003

Hindman et al., Mesos: A Platform for Fine-Grained Resource Sharing in the Data Center, NSDI 2011.

Others: Faircloud

19. (Dec 2) Authentication

Steiner et al., Kerberos: An Authentication Service for Open Network Systems, USENIX 1988.
Burrows et al., A Logic of Authentication, ACM TOCS, 1990.

20. (Dec 9) Distributed security

Lampson, Practical Principles for Computer Security, 2006

Optional:
Lampson et al., Authentication in Distributed Systems: Theory and Practice, 1992.

Other optional reading:

Protocol buffers.
https://developers.google.com/protocol-buffers/docs/tutorials
Fawn
OneHop is Enough
Killian et al., Life and Death and the Critical Transition, NSDI 2007.
Reliability and security in Codeen
Anderson and Roscoe. Learning from PlanetLab. Worlds 2006.
RDDs/Spark
Privad
Tor
Dissent
Taos security
Alvisi et al., SOK: The Evolution of Sybil Defense via Social Networks
http://www.mpi-sws.org/~aclement/papers/Alvisi13SoK.pdf
Karma
SybilGuard
Maymounkov and Mazieres, Kademlia: A peer-to-peer information system based on the XOR metric, IPTPS 2002