Our First Goal in this Course was...

To Define the Term **Distributed System**
Can you name some examples of Distributed Systems?

- Client-Server (NFS)
- The Web
- The Internet
- A wireless network
- DNS
- Gnutella or BitTorrent (peer to peer overlays)
- A “cloud”, e.g., Amazon EC2/S3, Microsoft Azure
- A datacenter, e.g., NCSA, a Google datacenter, The Planet
What is a Distributed System?
A collection of (probably heterogeneous) automata whose distribution is transparent to the user so that the system appears as one local machine. This is in contrast to a network, where the user is aware that there are several machines, and their location, storage replication, load balancing and functionality is not transparent. Distributed systems usually use some kind of client-server organization.
Textbook definitions

- A distributed system is a collection of independent computers that appear to the users of the system as a single computer.
  [Andrew Tanenbaum]

- A distributed system is several computers doing something together. Thus, a distributed system has three primary characteristics: multiple computers, interconnections, and shared state.
  [Michael Schroeder]
A working definition for us

A distributed system is a collection of entities, each of which is autonomous, programmable, asynchronous and failure-prone, and which communicate through an unreliable communication medium.

- Entity=a process on a device (PC, PDA)
- Communication Medium=Wired or wireless network
- Our interest in distributed systems involves
  - design and implementation, maintenance, algorithmics

What Evidence/Examples have we seen?
Problems we have seen since then

- Time and Synchronization
- Global States and Snapshots
- Failure Detectors
- Multicast
- Mutual Exclusion
- Leader Election
- Consensus and Paxos
- Gossiping
- Peer to peer systems – Napster, Gnutella, Chord, BitTorrent
- Cloud Computing and Hadoop
- Sensor Networks
- Structure of Networks
- Datacenter Disaster Case Studies

Basic Theoretical Concepts

Cloud Computing

What Lies Beneath
Problems we have seen since then (2)

- RPCs & Distributed Objects
- Concurrency Control
- 2PC and Paxos
- Replication Control
- Key-value and NoSQL stores
- Stream Processing
- Graph processing
- Scheduling
- Distributed File Systems
- Distributed Shared Memory
- Security

Basic Building Blocks

Distributed Services (e.g., storage)

Cloud Computing

Old but Important (Re-emerging)
What This Course is About

• Sports
• Movies
• Travel to Mars
• Job Interviews
• (Not Kidding)
What This Course is About

• Sports: HW1
• Movies: HW2
• Travel to Mars: HW3
• Job Interviews: HW4
• (Not Kidding)
What This Course is About (2)

• Midterm
• HW’s and MP’s

{ How to get good grades (and regrades, and jobs in some cases) }

– You’ve built a new cloud computing system from scratch!
– And beaten a state of the art system!

How far is your design from a full-fledged system?
Can you convince developers to use your Crane instead of Storm/Spark Streaming...?
Rejoinder: Typical Distributed Systems Design Goals

• Common Goals:
  – Heterogeneity
  – Robustness
  – Availability
  – Transparency
  – Concurrency
  – Efficiency
  – Scalability
  – Security
  – Openness

Do they make sense now?
Rejoinder: Typical Distributed Systems Design Goals

- **Common Goals:**
  - **Heterogeneity** – can the system handle a large variety of types of PCs and devices?
  - **Robustness** – is the system resilient to host crashes and failures, and to the network dropping messages?
  - **Availability** – are data+services always there for clients?
  - **Transparency** – can the system hide its internal workings from the users?
  - **Concurrency** – can the server handle multiple clients simultaneously?
  - **Efficiency** – is the service fast enough? Does it utilize 100% of all resources?
  - **Scalability** – can it handle 100 million nodes without degrading service? (nodes=clients and/or servers) How about 6 B? More?
  - **Security** – can the system withstand hacker attacks?
  - **Openness** – is the system extensible?
  - (Also: consistency, CAP, partition-tolerance, ACID, BASE, and others … )
Problems we have seen in Class
(and their relation to other courses)

- Time and Synchronization
- Global States and Snapshots
- Failure Detectors
- Multicast Communications
- Mutual Exclusion
- Leader Election
- Consensus and Paxos
- Gossiping
- Peer to peer systems – Napster, Gnutella, Chord
- Cloud Computing
- Sensor Networks
- Structure of Networks
- Datacenter Disaster Case Studies

Core Material of this course
Related to CS 525 (Advanced Distributed Systems Offered Spring 2020)
<table>
<thead>
<tr>
<th>Topics</th>
<th>Core Material of this course</th>
<th>Related to CS 411/CS 511</th>
<th>Related to CS 525</th>
<th>Related to CS 421/CS 433</th>
<th>Related to CS 523/561</th>
</tr>
</thead>
<tbody>
<tr>
<td>• RPCs &amp; Distributed Objects</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Concurrency Control</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• 2PC and Paxos</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Replication Control</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Key-value and NoSQL stores</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Stream Processing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Graph processing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Scheduling</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Distributed File Systems</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Distributed Shared Memory</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Security</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CS525: Advanced Distributed Systems (taught by Indy)

CS 525, next offered Spring 2020

- Looks at hot topics of research in distributed systems: clouds, p2p, distributed algorithms, sensor networks, and other distributed systems
- We will read many papers and webpages for cutting-edge systems (research and production)
- If you liked CS425’s material, it’s likely you’ll enjoy CS525
- Project: Choose between Research project or Entrepreneurial project
  - Your project will build a cutting edge research distributed system, and write and publish a paper on it
  - Your project will build a distributed system for a new startup company idea (your own!) and perform associated research with it
- Both graduates and undergraduates welcome! (let me know if you need my consent).
- Class size is around 70-100
- Previous research projects published in journals and conferences, some great startup ideas too!
Questions?
A distributed system is a collection of entities, each of which is autonomous, programmable, asynchronous and failure-prone, and which communicate through an unreliable communication medium.

[Is this definition still ok, or would you want to change it?] Think about it!
Final Exam

• **Office Hours**: Regular [All TAs and Indy] until and including Dec 13\textsuperscript{th} (usual schedule).
  – Exceptions posted on Piazza (check before heading out to an OH)

• **Final Exam**: December 14 (Friday), 8.00 AM – 11.00 AM
  – Locations (also on Course Schedule)
    • **114 DKH (David Kinley Hall)**: if your last name starting letter is A-L
    • **1320 DCL (Here)**: if your last name starting letter is M-Z
    • Please go to your assigned classroom only!
  – Syllabus: Includes all material since the start of the course. There may be more emphasis on material since midterm.

• Please check Piazza before finals: updates will be posted there
Cheat sheet: Allowed to bring a cheat sheet to the exam (US letter size, two sides only, at least 1 pt font). Need to turn it in with exam. Physical copy only, no online access during exam.

Can bring a calculator (but no other devices).

Structure: Final will be similar in structure to Midterm, only proportionally longer. More detailed answers to long questions (partial credit).

Preparing: HW problems, and midterm problems (and textbook problems).
Course Evaluations

- Main purpose: to give us feedback on how useful this course was to you (and to improve future versions of the course)
- I won’t see these evaluations until after you see your grades
- Use pencil only
- Answer all questions
- Please write your detailed feedback on the back – this is valuable for future versions of the course!
- After you’ve filled out, hand survey to volunteer, and return pencil to box
- Volunteer student:
  1. Please collect all reviews, and drop envelope in campus mail box
  2. Return the box of pencils to me (3112 SC)