CS 425 / ECE 428
Distributed Systems
Fall 2014

Indranil Gupta (Indy)

August 26 – December 9, 2014

Lecture 1-29

Web: courses.engr.illinois.edu/cs425/

Our First Goal in this Course was...

(First lecture slide)

To Define the Term Distributed System

Can you name some examples of Distributed Systems?

(First lecture slide)

- 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 are other examples you've seen in class?

What is a Distributed System?

(First lecture slide)

FOLDOC DEFINITION

(First lecture slide)

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

(First lecture slide)

- 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

(First lecture slide)

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

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

Problems we have seen since then

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

Basic Theoretical Concepts

Cloud Computing

What Lies Beneath

Problems we have seen since then (2)

- RPCs & Distributed Objects ← Basic Building Blocks
- Concurrency Control
- 2PC and Paxos
- Replication Control
- Key-value and NoSQL stores

} Distributed Services (e.g., storage)

- Stream Processing
- Graph processing

} Cloud Computing

- Self-stabilization
- Distributed File Systems
- Distributed Shared Memory
- Security

} Old but Important

Problems we have seen since then (3)

- Midterm
- HW's and MP's

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

  – You've built a new (emulated) cloud computing system from scratch!

Take it and build a real deployed key-value store!

Rejoinder: Typical Distributed Systems Design Goals

(First lecture slide)

- Common Goals:
  - Heterogeneity
  - Robustness
  - Availability
  - Transparency
  - Concurrency
  - Efficiency
  - Scalability
  - Security
  - Openness
  - (Also: consistency, CAP, partition-tolerance, ACID, BASE, and others …)

Do they make sense now?

Rejoinder: Typical Distributed Systems Design Goals

(First lecture slide)

- 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)

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

Core Material of this course

Related to **CS 525 (Advanced Distributed Systems Offered Spring 2015)**

Related to CS 438/439/538

Problems we have seen in Class

(AND THEIR RELATION TO OTHER COURSES)

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

Core Material of this course (RPCs & Distributed Objects, Concurrency Control, 2PC and Paxos)

Related to CS 411/CS 511 (Replication Control)

Related to **CS 525** (Key-value and NoSQL stores, Stream Processing, Graph processing, Self-stabilization, Distributed File Systems)

Related to CS 421/CS 433 (Distributed Shared Memory)

Related to CS 423/523 (Security)

CS525: Advanced Distributed Systems (taught by Indy)

CS 525, Spring 2015

- Looks at hot topics of research in distributed systems: clouds, p2p, distributed algorithms, sensor networks, and other distributed systems
- We 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
- Previous research projects published in journals and conferences, some great startup ideas too!

Questions?

A WORKING DEFINITION FOR US

(First lecture slide)

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 [Indy + All TAs] until Dec 12th (usual schedule).
- **Final Exam**
  - Final Exam, December 12 (Friday), 1.30 PM – 4.30 PM
    - DCL 1320: if your last name starts with A-M
    - Everitt 151: if your last name starts with N-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.
  - Cheat sheet: Allowed to bring a *cheat sheet* to the exam (A4 size, two sides only, at least 1 pt font). Need to turn it in with exam.
  - Can bring a calculator (but no other devices).
  - Structure: Final will be similar in structure to Midterm, only 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 questions 1 and 2 (you can skip #5)
- Please write your detailed feedback on the back – this is valuable for future versions of the course!
- After you've filled out:
  - 1) Hand survey to volunteer. 2) Pick up your HW4.
- Volunteer student:
  1. Please collect all reviews, and drop envelope in *campus mail box*
  2. Return the box of pencils to me (3112 SC)
  3. Return un-collected HW4s to me (3112 SC)