CS 425 / ECE 428 Distributed Systems Fall 2022

Indranil Gupta (Indy)

August 24 – December 7, 2022

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, AWS

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

- 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
- Spark
- ML
- Scheduling
- Distributed File Systems
- Distributed Shared Memory
- Security

$\leftarrow$ Basic Building Blocks

$\}$ Distributed Services (e.g., storage)

$\}$ New Emerging Distributed Systems

$\}$ Old but Important (Re-emerging)

What This Course is About

(First lecture slide)

- Sports
- Movies
- Travel to Saturn
- Interviews
- Company Acquisitions
- (Not Kidding)

What This Course is About

(First lecture slide)

- Sports: HW1
- Movies: HW2
- Travel to Saturn: HW3
- Interviews: HW4
- Company Acquisitions: MP1-4
- (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)
(& that standard devs are important!)

  - You've built a new distributed system from scratch!
  - And used some open-source distributed systems!

How far is your design from a full-fledged system?

Rejoinder: Typical Distributed Systems Design Goals

(First lecture slide)

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

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)

- 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 other graduate classes in department (e.g., CS523, CS525)

Problems we have seen in Class

(and their relation to other courses)

- RPCs & Distributed Objects
- Concurrency Control
- 2PC and Paxos

Core Material of this course

- Replication Control

Related to CS 411/CS 511

- Key-value and NoSQL stores
- Stream Processing
- Graph processing
- Spark
- ML
- Scheduling
- Distributed File Systems

Related to CS 525

- Distributed Shared Memory

Related to CS 421/CS 433

- Security

Related to CS 523/561

Other Related Grad Courses

- CS598AG – (Aishwarya Ganesan) ML+Systems, Distributed computing
- CS598RA – (Ram Alagappan) Storage systems
- CS598LR – (Ling Ren) Consensus, Blockchain
- CS523 – Tianyin Xu
- CS525 – Indy

- See also courses by Radhika Mittal (ECE, distributed storage), Andrew Miller (ECE, blockchain)

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 [All TAs and Indy] until Dec 10th 4 pm (usual schedule).
  - Exceptions posted on Piazza (check before heading out to an OH)
- **Final Exam Window: Online, timed. Dec 11 Sun 7 pm to Dec 13 Tue 10 pm (must finish before 10 pm)**
  - Please follow all instructions on Piazza and website
  - Syllabus: Includes all material since the start of the course. There may be more emphasis on material since midterm.
  - During final exam time (Dec 11 to Dec 13) there will be a Piazza blackout; you can only post private questions ONLY about the exam questions (no questions on material)
- Please check Piazza before (and during) finals: updates/corrections will be posted in a pinned post there

Course Evaluations (“ICES”)

- Please complete them online! (Search for mail from “ICES”)
- 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
- Answer all questions
- Please write your detailed feedback – this is valuable for future versions of the course!