

CS 425 / ECE 428  
Distributed Systems  
Fall 2023

Indranil Gupta (Indy) and Aishwarya  
Ganesan

*August 23 – December 6, 2023*

*Lecture 1-29*

All slides © IG

# 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
- ← 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)

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

# What This Course is About

(First lecture slide)

- Politics: 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! )

## **MPs: Amazing work, everyone!**

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

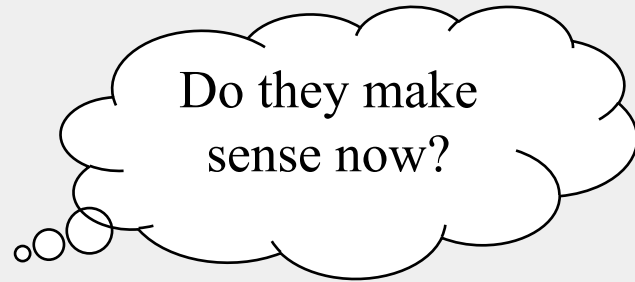
}  
How far is your design from a  
full-fledged system?  
What else do you need to do to  
make it competitive with open-source?

# Rejoinder: Typical Distributed Systems Design Goals

- Common Goals:

- Heterogeneity
- Robustness
- Availability
- Transparency
- Concurrency
- Efficiency
- Scalability
- Security
- Openness

(First lecture slide)



# Rejoinder: Typical Distributed Systems Design Goals

- Common Goals:

(First lecture slide)

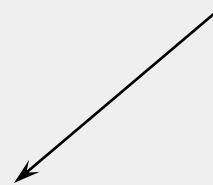
- **Heterogeneity** – can the system handle a large variety of types of PCs and devices?
- **Robustness** – is the system resilient to host crashes and failures, and to the network dropping messages?
- **Availability** – are data+services always there for clients?
- **Transparency** – can the system 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

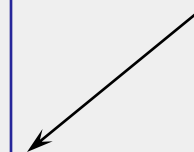
(and their relation to other courses)

- 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

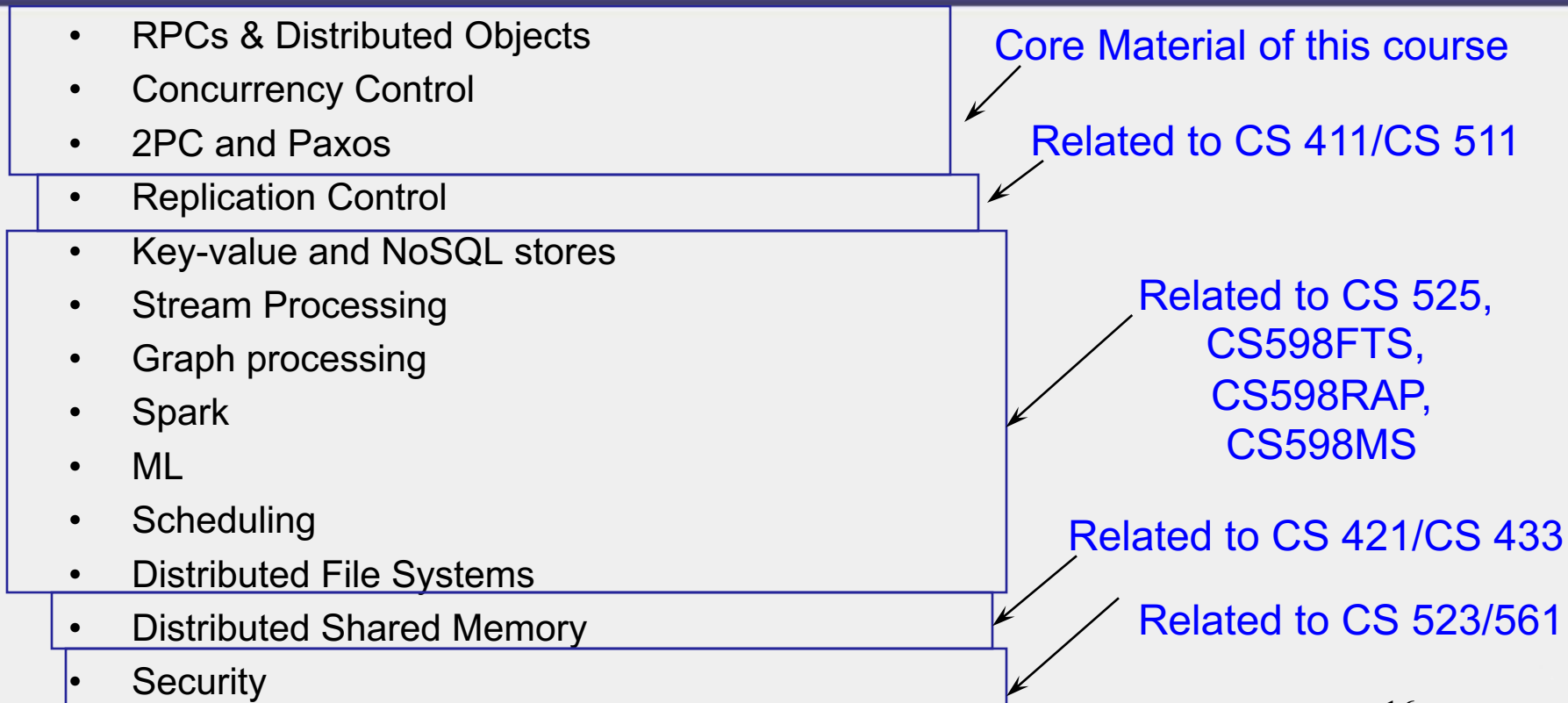


Related to other graduate  
classes in  
department (e.g., CS523, CS525)



# Problems we have seen in Class

(and their relation to other courses)





# CS525: Advanced Distributed Systems (taught by Indy)

## CS 525, next offered Spring 2024

- Looks at hot topics of research in distributed systems: clouds, p2p, distributed algorithms, ML, sensor networks, and other distributed systems
- We will read many papers and webpages for classical and 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 50-80
- Previous research projects published in journals and conferences, some great startup ideas too!

# CS598 FTS (taught by Aishwarya)

## CS598 RAP (taught by Ram)

### **CS598 FTS - Fault-tolerant and consistent data center systems, next Spring 2025**

- Deep dive deep into replication and consensus protocols, geo-replication, distributed transactions, and various consistency models and how to implement them.
- Designing distributed systems for emerging hardware (e.g., persistent memory, programmable network) and emerging trends in data center (e.g., rack-scale, RDMA)

### **CS 598 RAP - Storage systems, next Spring 2025**

- Covers a set of topics in storage systems both local (e.g., local key-value stores, file systems) and distributed (e.g., disaggregation, control-plane storage).
- Some topics covered in recent offerings: write-optimized storage systems, reliability and performance in local storage systems, crash consistency techniques, shared-log systems, and storage and memory disaggregation.
- Read, review, and discuss research papers; case studies from production systems.
- Semester-long research project

# Other Related Grad Courses

- CS525 – Indy
- CS598FTS – (Aishwarya Ganesan) ML+Systems, Distributed computing
- CS598RAP – (Ram Alagappan) Storage systems
- CS598LR – (Ling Ren) Consensus, Blockchain
- CS598MS – (Daniel Kang) Machine Learning and Data Systems
- CS523 – Tianyin Xu
  
- See also courses by Radhika Mittal (ECE, distributed storage), Andrew Miller (ECE, blockchain), Mingjia Zhang (new in SP24)

# 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] until Dec 11<sup>th</sup> (usual schedule).
  - Exceptions posted on Piazza (check before heading out to an OH)
- **Final Exam: In person. Dec 12<sup>th</sup> Tue at 7 pm to 10 pm**
  - There will be a Piazza blackout Dec 8 (Fri) afternoon to Dec 11 (Mon) morning, as Coursera students will be taking the final exam (their final exam is different than yours)
  - Syllabus: Includes all material since the start of the course. There may be more emphasis on material since midterm.

# 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!