Distributed Systems

https://dse.in.tum.de/

Chair of Decentralized Systems Engineering (DSE)
Department of Computer Science

WS 2022/23

Course staff members

Chair of Decentralized Systems Engineering (DSE)

- Established in **September 2020 @ TUM**

- Chair website: https://dse.in.tum.de/

- Research areas:
  - Distributed systems
  - Operating systems and virtualization
  - Cloud computing and scalable data analytics
  - Storage and networked systems
  - Reliability and security

Lecturers

IT is Navid's Postman

Prof. Pramod Bhatotia

Dr. Martin Kleppmann

Prof. Pramod Bhatotia

- Professor at TU Munich
  - Chair of Decentralized Systems Engineering (DSE)
- Research interests
  - Distributed systems and operating systems

IIT KANPUR — MAX PLANCK INSTITUTE FOR SOFTWARE SYSTEMS — TECHNISCHE UNIVERSITÄT DRESDEN — THE UNIVERSITY of EDINBURGH

2008 2009 2015 2016 2020-

IBM Research — YAHOO! RESEARCH — Microsoft Research — NOKIA Bell Labs — HUAWEI

Dr. Martin Kleppmann

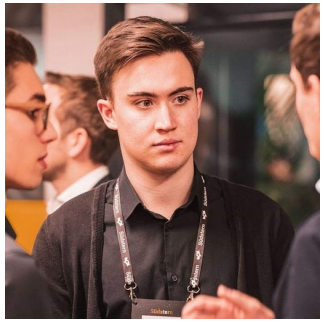
- Research fellow at TU Munich
  - Previously: University of Cambridge
  - Previous life: Silicon Valley Internet startups
- Book: Designing Data-Intensive Applications (2017)
- Current research: Decentralised collaboration software
  - More on this in week 7 lecture on eventual consistency

Teaching assistants (TAs)

Emmanouil (Manos) Giortamis

Pezhman Nasirifard

Harshavardhan Unnibhavi



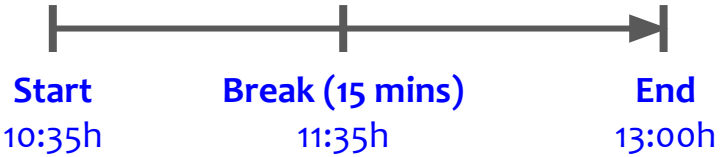
Julian Pritzl

Nathaniel Tornow

Format

Format

- Lectures
  - **Time:** Wednesday, 10:30 - 13:00 hrs
  - **Frequency:** 12 in-person lectures
  - **Venue:** Hörsaal 1 "Interims II" (5416.01.004)
  - **Dates:** See TUM Online



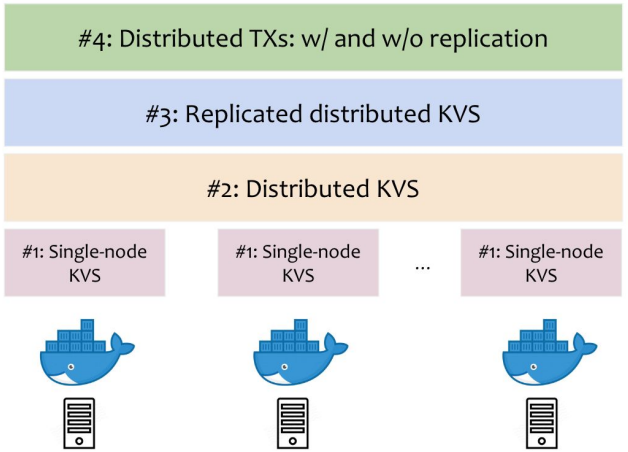
- Office hours
  - **Time:** Friday, 15:00-17:00 hrs
  - **Frequency:** Same as lectures
  - **Venue:** Online on Zoom (see TUM Online)
  - **Dates:** See TUM Online

Exam

- Single exam
  - No repeat exam
- Expected exam date: Feb 2023
  - Exact date and time will be finalized by the central administration
- Exam format
  - More details in the last lecture (25.01.2023)

Cloud systems engineering (“Cloud lab”)

- A complimentary practical lab
  - Learn by building distributed systems
  - https://github.com/TUM-DSE/cloud-lab
- Runs in parallel with DS lectures
  - Limited to 60 students
  - Either take it in parallel or next WS 23/24
  -
- An end-to-end system architecture:
  - KV-store, containers, kubernetes
  - Scaling distributed systems
  - Fault-tolerance with replication
  - Distributed transactions/algorithms (2PC)



Course material

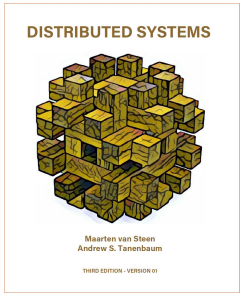
- Lecture slides and online notes
  - Available on Moodle
  - Lecture video recording (not guaranteed)

- Research papers and open-source projects
  - Provided as references

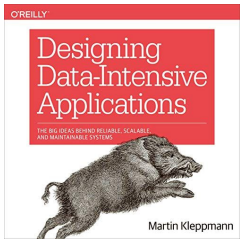
- Q&A/Discussion on slack (optional)
  - Workspace URL: https://ls1-courses-tum.slack.com
  - Course channel: #ws-22-ds
  - Feel free to join with @tum.de email address

Recommended books

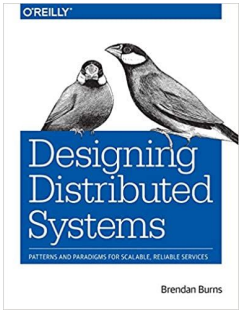
“Distributed Systems”
— Maarten van Steen and Andrew s. Tanenbaum



“Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems”
— Martin Kleppmann



“Designing Distributed Systems: Patterns and Paradigms for Scalable, Reliable Services”
— Brendan Burns



Code of conduct

- University plagiarism policy
  - https://www.in.tum.de/en/current-students/administrative-matters/student-code-of-conduct/

- Decorum
  - Promote freedom of thoughts and open exchange of ideas
  - Cultivate dignity, understanding and mutual respect, and embrace diversity
  - Racism and bullying will not be tolerated

Course overview

A Distributed system

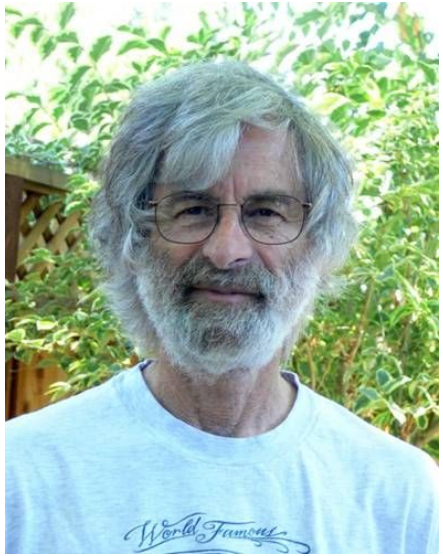
IT is Neil's Postman

"... a system in which the failure of a computer you didn't even know existed can render your own computer unusable."

–Lamport

"OR" a distributed system is an application that executes a collection of protocols over

- .. multiple computers communicating via a network.
- .. trying to achieve some task together
- Consists of "nodes" (computer, phone, car, robot, ...)



Leslie Lamport
Turing Award Winner, 2014

Why make a system distributed?

- It's inherently distributed:
  - e.g. sending a message from your mobile phone to your friend's phone
- For better reliability:
  - even if one node fails, the system as a whole keeps functioning
- For better performance:
  - get data from a nearby node rather than one halfway round the world
- To solve bigger problems:
  - e.g. huge amounts of data, can't fit on one machine

Powerful and ubiquitous “distributed” systems

- A distributed system can be much larger and more powerful given that the combined capabilities of distributed components

- Even a stand-alone application (e.g., your phone app) is most likely using a distributed system component (e.g., client-server)

- However, great capabilities come with great challenges!
  - This course will prepare you to solve these challenges!

Challenges of a distributed system

IT is Nandos Postman

- Fault tolerance
  - How to recover from failures without performing incorrect actions
- Availability
  - 24x7 operations, even in the presence of failures!
- Recovery
  - Failed system component can restart and rejoin the system in a correct state
- Consistency or correctness
  - System invariants are preserved in presence of concurrency, asynchrony, and failures!
- Scalability
  - If you can’t scale, you not gonna survive in the market!
- Performance
  - Achieve predictable performance, response in timely manner
- Security
  - System is secure/authenticated: data and code!

Learning goals

- Understand **architectures of distributed systems**, their building blocks and applications

- Apply **foundational principles** in the development of distributed systems

- Understand properties of **common building blocks** applicable for systems design

- Understand the **complexities involved in developing a distributed system** (e.g., machine and network failures, concurrency, etc.)

- Study **advanced distributed systems** topics

Tentative topics

Week	Topics
1	Overview + Distributed data analytics (MapReduce/Spark)
2	Distributed filesystem (HDFS)
3	Modelling distributed systems
4	Logical and physical time, broadcast protocols
5	Replication, 2PC, consistency models
6	Consensus and the Raft algorithm
7	Eventual consistency and CRDTs
8	Scalable systems: KV storage systems + sharding
9	Distributed synchronization: Apache Zookeeper
10	Distributed database (BigTable)
11	Distributed TXs (Spanner)
12	Exam prep.

References

- Introduction to Distributed System Design
  - https://www.hpcs.cs.tsukuba.ac.jp/~tatebe/lecture/h23/dsys/dsd-tutorial.html