IN.5022 — Concurrent and Distributed Computing

Introduction

Prof. P. Felber

pascal.felber@unine.ch

Teaching assistant: Loris Witschard

Course objectives

- Covers the principles of concurrent and distributed computing
  - Combines theory and practical, hands-on exercises targeting distributed settings
- Expected results
  - Understand the major challenges and complexity inherent to concurrency and distribution
  - Know some of the major algorithms, programming paradigms, and theoretical results
  - Learn how to develop and deploy a “basic” distributed application (mini-project)

Lectures and labs

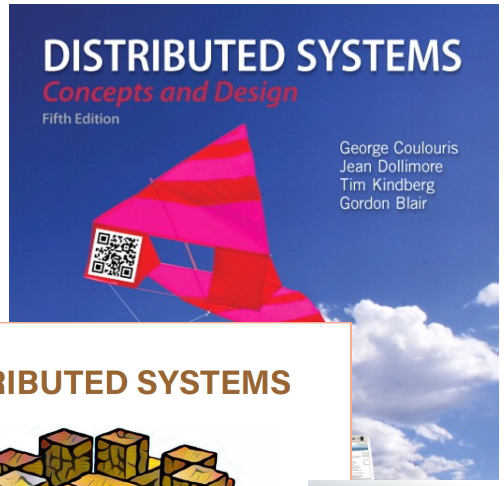
- Weekly lecture
  - Presentation of the “theoretical” material
- Weekly exercises sessions
  - Q&A, presentation of the assignments
- Assignments and homework
  - Assignments given on Moodle
    - Can be done at home or during exercise sessions
  - Results from assignments must be submitted via Moodle
    - Deadline generally before the following session
  - Assignments will be corrected but not graded (“pass” or “fail”)
    - Feedback will be given to students

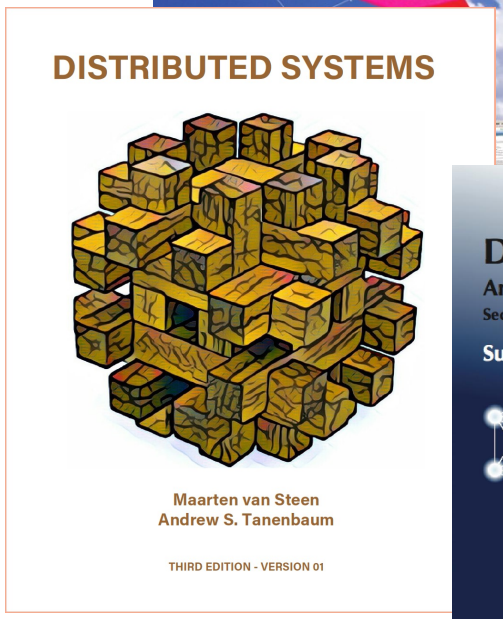
Evaluation

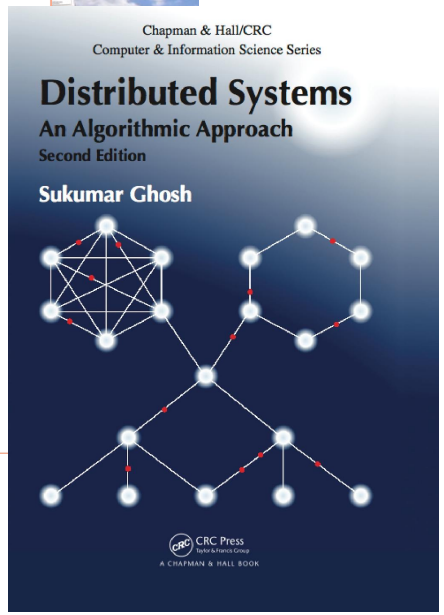
- Weekly exercises are mandatory and must be completed
  - Not graded but students must “pass” 70% of the exercises (6 of 8)
- Mini-project is mandatory and will be graded
  - Report and presentation
  - 30% of the final grade
- Exam
  - Written exam at the end of the term
  - 70% of the final grade

Some textbooks (optional)

- Distributed Systems:
  Concepts and Design (5e)
  George Coulouris, Jean Dollimore, Tim Kindberg, Gordon Blair
  `www.cdk5.net`
- Distributed Systems (3e)
  Maarten van Steen, Andrew S. Tanenbaum
  `www.distributed-systems.net`
- Distributed Systems:
  An Algorithmic Approach(2e)
  Sukumar Ghosh
  `homepage.divms.uiowa.edu/~ghosh`



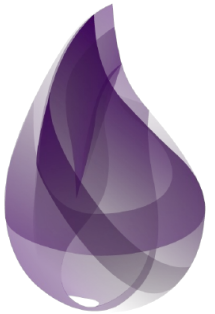




Exercises and mini-project

- Some exercises “on paper”
- Other exercises as small coding challenges
  - Language: **Erlang** *or* **Elixir**
  - Online development environment: `replit.com`
  - Distributed instances using Docker containers
  - Use your own laptop (computers available upon request)

- Mini-project
  - Develop a functional application
  - By teams (groups of 2 students)
  - Language: **Erlang** *or* **Elixir**



Agenda of the course (subject to changes)

1. Foundations of CDC
2. Representation + Erlang/Elixir (part 1)
3. Erlang/Elixir (part 2)
4. Shortest path
5. Logical time & clocks
6. Multicast communication
7. Coordination
8. Storage and lookup: DHTs
9. Replicated data: CRDTs + mini-project — Start
10. Mini-project — Idea
11. Mini-project
12. Mini-project
13. Mini-project
14. Project presentations