Distributed Systems

ID2201

outline

Johan Montelius

First thing

- Course home page: KTH Social
- register by signing the paper now or next week
  - if you haven't done *course selection* arrange for doing so, talk to your study coordinator
  - make sure that you have done semester registration (terminsregistrering)
  - If you're not on the list, you're not in the course!

You should after the course be able to...

- explain important characteristics of distributed systems
- describe architectural and fundamental models of distributed systems
- explain and compare strategies for interprocess communication
- explain and compare middleware models
- explain and compare name services
- explain the concept of logical time
- use logical time to implement distributed algorithms

examination

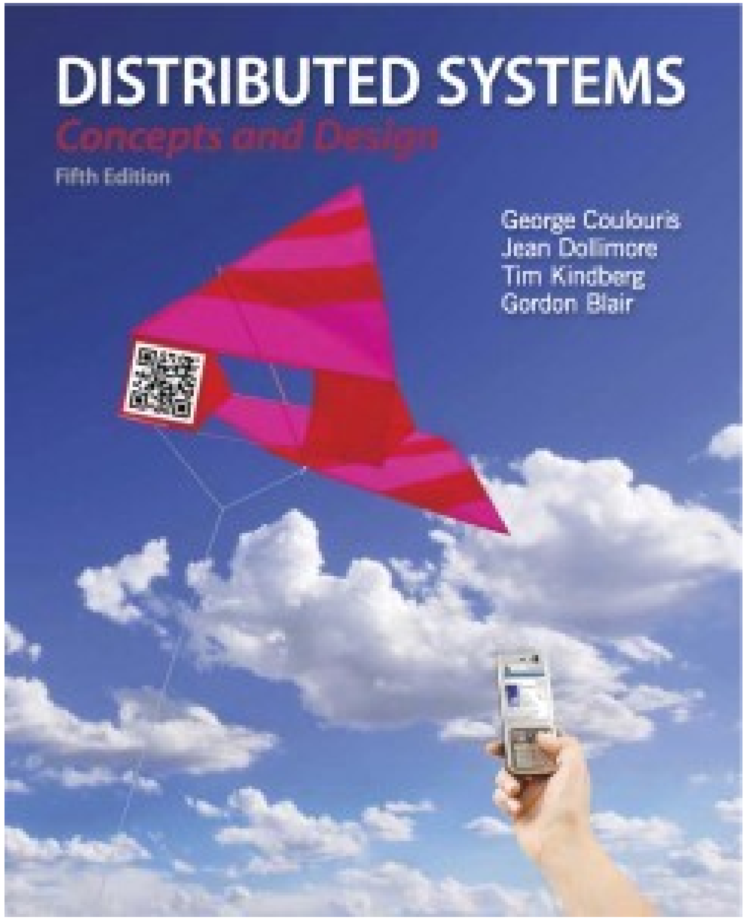
- compulsory lab session / seminars
  - complete tasks in advance
  - signing the list is “yes I've done it”
  - don't turn up unprepared
  - if you can not attend, email before the seminar
- written examination
  - A : declarative (multiple choice questions)
  - B : compare, describe (multiple choice questions)
  - C : analytic, reflect (essay answers)

Grading

- Written exam
  - E: 16 out of 22 in part A
  - D: 18 out of 22 in part A
  - C: … and 6 out of 12 in part B
  - B: … and 8 out of 12 in part B
  - A: … and 8 out of 8 in part C
- Seminars
  - Pass / Fail
- Final grade
  - Grade of exam possibly raised depending on seminar

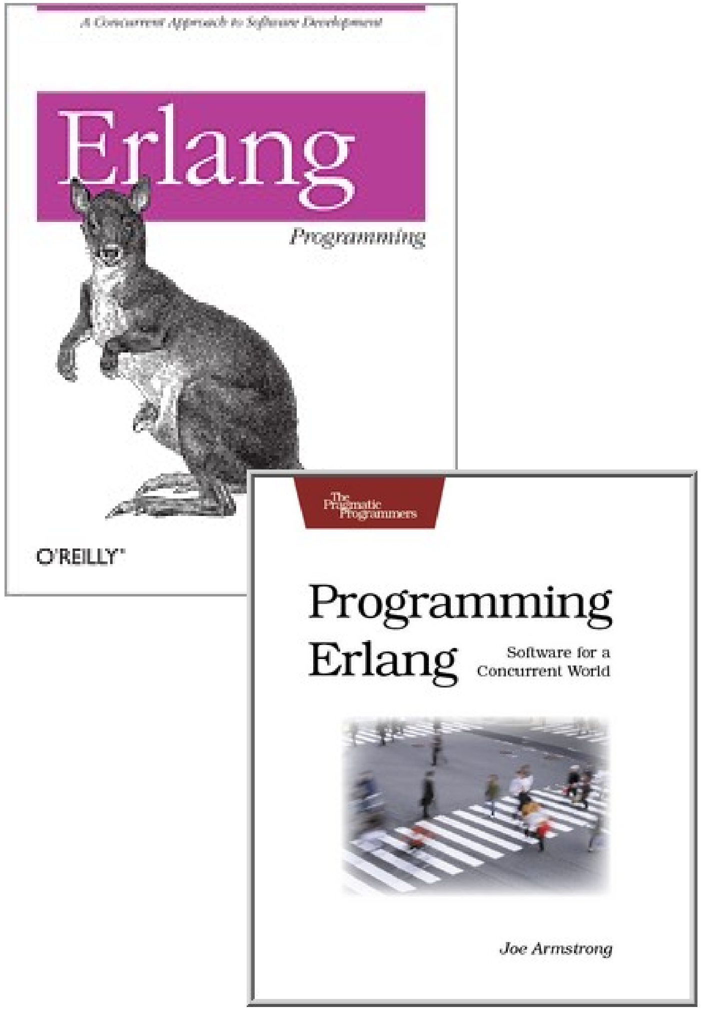
course literature

- Distributed System – concepts and design
  - Couloris, Dollimore, Kindberg
  - fifth edition (fourth is ok)
  - Addison Wesley
  - http://www.cdk5.net/



Erlang

- Programming Erlang
  - Francesco Cesarini and Simon Thompson
  - O'Reilly
- Programming Erlang – software for a Concurrent world
  - Joe Armstrong
  - Pragmatic Programmer
- Online material
  - www.erlang.org


A Concurrent Approach to Software Development
Erlang
Programming
O'REILLY
The Pragmatic Programmers
Programming Erlang
Software for a Concurrent World
Joe Armstrong

Lectures

- Fourteen lectures that will mostly follow the course book.
  - do read in advance
- Erlang is only given one lecture, you're expected to pick up a new language on your own.
- Slides will be available on the web.

Lectures

- 1: Introduction
  - What is a distributed systems and why is it different. *chapter 1 and 2*.
- 2: Erlang
  - Concurrent and distributed programming in Erlang.
- 3: Networks and process communication
  - Things you should know but we'll go through them again. *chapter 3 and 4*

Lectures

- 4: Remote invocation
  - Language constructs to program distributed systems. *Chapter 5*
- 5: Indirect Communication
  - Group communication, publish/subscribe and message queue systems. *Chapter 6*
- 6:File systems and Name services
  - The problems of a distributed file system, performance, consistency *chapter 12*

Lectures

- 7: Time
  - Time, a simple thing that turns out to be very complex. *Chapter 14.1-4*
- 8: Global state
  - Can we describe the state of a distributed system and what can we determine. *Chapter 14.5*
- 9: Coordination and agreement
  - How do we agree and how do we know that we do agree? *chapter 15*

Lectures

- 10: Transactions
  - How can we make a set of operations behave as an atomic operation? *chapter 16*
- 11: Distributed transactions
  - Now how do we solve it if we have multiple servers. *chapter 17*
- 12: Replication
  - building fault tolerant systems, *chapter 18*

Lectures

- 13: Distributed Hash Tables
  - Why do hashing? *chapter 16*
- 14: Peer-to-peer
  - A bit of the pros and cons of peer-to-peer systems. *Chapter 10*

Seminars / lab sessions

- First session
  - help with completing the tasks
  - TA: Mats Jansson
- Second session
  - hand in written report on how you solved the problem
  - be prepared to present your solution
  - connect the systems and do some experiments
- Select which group to join in Daisy.

Erlang

- The basics of Erlang
  - not compulsory, only if you need help getting started with Erlang
  - You should after this seminar be able to write, compile and execute distributed Erlang programs and know how Erlang handles process communication.

Rudy – a small web server

- Before the seminar:
  - complete a limited HTTP parser in Erlang
- At the seminar
  - Implement a web server that can return canned answers.
  - Discuss extensions and alternative implementations.

Routy – a routing network

- Before the seminar
  - Complete the Dijkstra algorithm (this is tricky) and implement a router.
- At the seminar:
  - Set up routers and start connecting them.

Loggy – a logical time logger

- Before the seminar
  - implement and do experiments with the different loggers given
- At the seminar
  - run distributed experiments of the loggers
  - discuss pros and cons of the loggers

Groupy – a group membership service

- Before the seminar:
  - Complete the assignment up to, and including, section 3
  - Write up your observations in a short report and be prepared to connect the nodes in a class room wide group.
- At the seminar:
  - On the seminars we will discuss the questions under section 4.

Chordy – a distributed hash table

- Before the seminar
  - Implement the distributed hash table u
  - write up a short report
- At the seminar:
  - build a larger ring and perform some measurements
  - discuss how to proceed with handling of failures