AC-04-12

Outline for a Course on Distributed Systems

COMP 512

Bettina Kemme

Introduction:

Owing to new technologies like the Internet and cluster computing, distributed systems have become reality and are widely applied in practice. Well known are the Web and distributed component infrastructures like CORBA or J2EE compliant application servers. Moving from a centralized to a distributed environment introduces new complexity: communication, asynchronous behavior of the different components in the system, architectural considerations, failures etc. Distributed systems handle these issues by providing tools and protocols for efficient and powerful coordination among the cooperating components.

The objective of this new course is to learn the state-of-the-art of practical distributed systems, to understand the typical problems and challenges encountered in distributed environments, and to discuss both sound and practical solutions for them.

Pre-requisites:

The formal pre-requisites of this course are COMP-251 *Data Structures and Algorithms* and COMP-310 *Computer Systems and Organization*. Students are assumed to know the material of these courses well. A networking course (e.g., COMP-435 *Basic Computer Networks*) is not required but might be useful for a better understanding. A short overview of network protocols will be given at the beginning of the course.

Description:

The course will discuss the most important tools, algorithms, and services provided by distributed system in a systematic manner. For each topic, we will look at the main problems and challenges to solve, then discuss basic algorithmic solutions, and then look how existing systems implement these algorithms. Each topic will be discussed in the context of several different system architectures and applications.

The weekly schedule is as follows:

Overview (1 week)

- Introduction,
- Models and architectures
- Fault, errors, and failures in a distributed environment
- Example distributed systems

Application-oriented communication paradigms (3 weeks)

- Basic network protocols
- Synchronous vs. asynchronous communication
- 2-3 of the following communication paradigms
    - client/server protocol
    - remote method invocation
    - group communication
    - persistent queues
    - publish/subscribe systems

Naming Services (1 week)

Synchronization (2 weeks)

- Physical and logical clocks
- Mutual exclusion
- Distributed transactions and concurrency control

Fault-tolerance (3 weeks)

- Process and data replication
- Agreement protocols
    - Example: distributed transactions and two-phase commit

Scalability (1 week)

- Data replication and caching
- Load balancing

Security (1 week)

- Encryption, public and private keys, authentication, privacy
- Access control

Discussion of 1-2 existing distributed system infrastructures, e.g., (1 week)

- CORBA
- J2EE
- Web
- Peer-to-Peer

Textbook will be either

- Distributed Systems - Principles and Paradigms, by A.S. Tanenbaum and M. van Steen, Prentice Hall, 2002. or
- Distributed Systems - Concepts and Design, 3rd ed., G. Coulouris, J. Dollimore, and T. Kindberg, Addison Wesley, 2001.

Additional course material will be provided, e.g., lecture slides, references to research papers and secondary textbooks, and links to web-pages with relevant information about existing systems.

Evaluation:

The evaluation scheme will be as follows:

Written assignments: 15%
Midterm: 10%
Final: 40%
Project: 35%

In the written assignments (probably three equally spread over the term), the students will analyze existing algorithms and techniques, develop their own algorithms, and propose architectural solutions to given problems. The purpose of the midterm is to provide students with an early indication whether they understand the course material. The final covers the entire course material. Since the course is accompanied by a large programming project, which goes beyond the workload of a typical 3-credit course, this course has 4 credits. The students will learn how to program using advanced component and communication technology. They will develop their own distributed system infrastructure providing advanced services (e.g., concurrency control or replication). The project will have several deliverables (probably three equally over the term). Each deliverable will require a project report, and a demonstration at the computer. Probably the first deliverables will only be milestones with little weight, while the last deliverable will present the project in its entirety with a large weight.

A note on academic integrity

McGill University values academic integrity. Therefore all students must understand the meaning and consequences of cheating, plagiarism and other academic offences under the Code of Student Conduct and Disciplinary Procedures (see http://www.mcgill.ca/integrity for more information).