II.3502 – Distributed Architectures and Programming

Module title: Distributed Architectures and Programming
Module code: II.3502
Person in charge : Alexandra LEVCHENKO
ECTS : 5 credits
Workload : 100h to 150h including 42h face-to-face
Teamwork : yes
Keywords : distributed, synchronization, load distribution, peer-to-peer, Fault tolerance, Elixir

Presentation

Modern software systems and services must now be integrated in increasingly distributed and parallel environments.

This evolution stems from several factors. Firstly, current processors are parallel by default (multi core). Secondly, numerous high margin service rely on processing large amounts of data in parallel, thus needing distribution of applications on several systems.

This functioning provides the performance and availability of applications on the modern deployment platforms such as cloud and virtualized infrastructures.

Academic Objectives

The aim of this module is to explain the design principles and constraints specific to distributed applications.

- Design and model a distributed algorithm/program
- Use communication libraries for distributed systems in Java
- Use a programming language dedicated to distributed systems (Elixir)
- Evaluate levels of coherence and availability required by a system

Prerequisites

- Programming in Java
- Graph theory (notions of graphs, nodes, trees, ..)
- Concurrent Programming (notion of thread/ process/locks/IPC)
- Networks: HTTP, TCP/IP, model OSI.

Content/Program

Concepts

- Communication in distributed systems: RPCs, message queues, shared memory, etc.
- Properties of distributed systems: Availability, consistency, partition tolerance, fault tolerance, scaling
- Typology of distributed systems: storage systems, computing systems, peer-to-peer systems, etc.
- Models of concurrent systems
- Java development environment
- Elixir development environment
- Concurrent system models
- Java development environment
- Elixir development environment

Pedagogical Methods

Learning methods

The module is organized into 7 lecture sessions (2h) and 7 tutorials/labs (3h).

The first tutorials will be devoted to implementing theoretical concepts through algorithmic exercises or case studies. Practical labs will follow on specific subjects, with the aim of building simple systems based on the examples seen in class.

Classes must be prepared by individual readings of scientific articles related to the forthcoming course (flipped classroom).

Assessment methods

- A tutorial assessed by a test of theoretical knowledges (20%)
- Assessments of lectures (quiz 20 %)
- Collective assessment of one lab (20%)
- Written final examination (Theory and a study case, 40 %)

Language

English

Bibliography, Webography, Other resources

- Distributed Systems: Principles and Paradigms Paperback by Andrew S. Tanenbaum, Maarten van Steen, 2013 Pearson
- Distributed Systems: Concepts and Design, 5th edition, George Coulouris, Jean Dollimore, Tim Kindberg, 2011, Pearson
- Distributed Algorithms An intuitive approach, Wan Fokkink, 2013, MIT Press
- http://book.mixu.net/distsys/single-page.html
- https://github.com/aphyr/distsys-class