Course Title:

Advanced Topics in Distributed and Reactive Programming

Course Code:

202-1-6191

Department:

Faculty of Natural Sciences - Department of Computer Science

Semester:

'19

Required Courses:

1. Advanced Topics in Distributed and Reactive Programming

Prerequisites:

Computer Systems, Programming Principles

Course Description:

Modern applications based on the use of distributed resources and organizing complex workflow between virtual machines in the cloud, internet services, distributed storage, and more.

Programming in such scenarios requires specific treatment of parallelism and synchronization, in order to enable scalability and availability.

The course covers programming methods to meet these challenges.

Topics:

1. Parallelism
   - Processes and threads
   - Immutability
   - Deadlocks, Mutexes
   - Liveness and performances

2. Synchronization
   - Mutual Exclusion
   - Semaphores
   - Monitors
   - Deadlocks
   - State & Time - Distributed Mutual Exclusion

3. Concurrency Frameworks
   - Java executors, futures, futureTask, asynchronous services
   - C++ asynchronous programming: std::async, std::future
   - Python generators, coroutines (yield and send), async and await
   - More
Modern applications are increasingly executed over distributed resources which are orchestrated into complex workflows involving virtual and cloud computers, Web servers, distributed storage, database servers. Programming in such environments requires specific attention to concurrency and synchronization, to ensure scalability and responsiveness. This course covers programming techniques that help address these challenges.

**Topics**

1/ Concurrency

- Processes and Threads
- Thread Safety: Definition and Immutable Objects
- Liveness and Performance

2/ Synchronization

- Mutual_Exclusion
- Semaphores
- Monitors
- Deadlocks
• Distributed Synchronization - Time & State
• Distributed Synchronization - Distributed Mutual Exclusion

3/ Concurrency frameworks:

• Java executors, futures, futureTask, asynchronous services.
• C++ asynchronous programming: std::async, std::future
• Python generators, coroutines (yield and send), async and await.
• Javascript (ES2015) generators, async, await
• Error handling in concurrent code

4/ Asynchronous I/O and Event Loops

• Non blocking I/O in Java, Python, C++, Javascript
• Java Reactor Pattern
• Python Asyncio
• Node

5/ Reactive Programming

• Streams, observable sequences, stream combinators
• Reactive Java (RxJava)
• Reactive Javascript (RxJS)
• Functional Reactive Programming (Akka)

6/ Applications

• Reactive REST Server case studies (Node Express, Java Spring with HTTP Streaming over asynchronous servlet)
• Reactive Client of multiple REST servers (Javascript)

Course Structure

• 5 Credits
• 4 hours / week lectures
• 2 hours / week practical sessions
• 4 programming assignments
• Exam