linfo1131

2020

Paradigms of concurrent programming

Due to the COVID-19 crisis, the information below is subject to change, in particular that concerning the teaching mode (presential, distance or in a comodal or hybrid format).

5 credits	30.0 h + 30.0 h	Q1

Teacher(s)	Van Roy Peter ;
Language :	English
Place of the course	Louvain-la-Neuve
Main themes	• Concurrent programming paradigms (shared state, message passing, and declarative competition) • Major programming concepts that include function, object, class, abstraction, instantiation, inheritance, state, encapsulation, competition, dataflow, lazy evaluation, non-determinism, agent (active object), lock, monitor, transaction, deadlock, higher order programming, compositionality, etc. • Techniques of reasoning and design with different paradigms to design correct programs • Practical applications in several areas (eg logic circuits simulation, elevator simulation, transaction manager).
Aims	Given the learning outcomes of the "Bachelor in Computer science" program, this course contributes to the development, acquisition and evaluation of the following learning outcomes: • S1.I5 • S2.2-4 • S5.2, S5.4-5 Students completing successfully this course will be able to 1 • define precisely and use wisely in medium-sized programs the key concepts of programming; • define the main paradigms of concurrent programming (shared state, message passing and declarative competition), with the concepts they contain and the properties they give to programs; • explain the relationships (similarities, differences) between these different paradigms; • explain the link between programming languages and the main paradigms of concurrent programming; • write medium-sized programs in these concurrent programming paradigms. • think using abstractions (reason correctly on a system that includes several layers of abstractions, and define new abstractions to simplify the resolution of a problem) - - - - *The contribution of this Teaching Unit to the development and command of the skills and learning outcomes of the programme(s) can be accessed at the end of this sheet, in the section entitled “Programmes/courses offering this Teaching Unit”.*
Evaluation methods	**Due to the COVID-19 crisis, the information in this section is particularly likely to change.** • Dispensatory test 25% (around week 7) • Project 25% • Final exam 50% (or 75% if redoing test part) The project is mandatory and is done during the quadrimester. It can only be done only once and it counts for the whole academic year. The optional dispensatory test and the final exam may be done in auditorium or online, depending on university requirements.
Teaching methods	**Due to the COVID-19 crisis, the information in this section is particularly likely to change.** • Weekly lectures (in auditorium or online, according to university requirements). • Practical lab sessions in the computer room every week, to solve simplified problems using concepts explained during the lectures. • One major design and programming project to apply these concepts to a more complex application.
Content	The goal of this course is to present advanced concepts in programming languages. Among others, the course presents: • functional programming and higher-order programming, lazy evaluation, • concurrent programming in its three principal forms, namely deterministic dataflow, message-passing concurrency, and shared-state concurrency, • a rigorous presentation of the concept of declarativeness, • advanced techniques for designing declarative algorithms, • security as applied to data abstraction, • the design of multi-agent systems including with the Erlang language,

	- a rigorous presentation of mutable state and its utility, - the concepts of lock, monitor, and transaction. All these concepts are accompanied with practical exercises.
Inline resources	LINFO1131 Moodle.
Bibliography	Peter Van Roy and Seif Haridi, *Concepts, Techniques, and Models of Computer Programming,* MIT Press, 2004.
Faculty or entity in charge	INFO

Programmes containing this learning unit (UE)

Program title	Acronym	Credits	Prerequisite	Aims
Additional module in computer science	APPSINF	5		🔍