GHENT UNIVERSITY

Course Specifications

Valid as from the academic year 2019-2020

Parallel and Distributed Software Systems (E017930)

Course size (nominal values; actual values may depend on programme)

Credits 6.0	Study time 180 h	Contact hrs 60.0 h

Course offerings and teaching methods in academic year 2020-2021

A (semester 1)	English	lecture	30.0 h
		seminar: coached exercises	10.0 h
		self-reliant study activities	15.0 h
		practicum	5.0 h
B (semester 1)	Dutch	seminar: coached exercises	10.0 h
		practicum	5.0 h
		guided self-study	30.0 h
		self-reliant study activities	15.0 h

Lecturers in academic year 2020-2021

Fostier, Jan	TW05	lecturer-in-charge
De Turck, Filip	TW05	co-lecturer

Offered in the following programmes in 2020-2021	crdts	offering
Master of Science in Teaching in Science and Technology (main subject Computer Science)	6	A
Brugprogramma Master of Science in Bioinformatics (main subject Engineering)	6	A
Bridging Programme Master of Science in Computer Science Engineering	6	B
Bridging Programme Master of Science in Computer Science Engineering	6	A
Master of Science in Electrical Engineering (main subject Communication and Information Technology)	6	A
Master of Science in Electromechanical Engineering (main subject Control Engineering and Automation)	6	A
Master of Science in Electromechanical Engineering (main subject Electrical Power Engineering)	6	A
Master of Science in Bioinformatics (main subject Engineering)	6	A
Master of Science in Electromechanical Engineering (main subject Maritime Engineering)	6	A
Master of Science in Electromechanical Engineering (main subject Mechanical Construction)	6	A
Master of Science in Electromechanical Engineering (main subject Mechanical Energy Engineering)	6	A
Master of Science in Computer Science	6	A
Master of Science in Computer Science Engineering	6	B
Master of Science in Computer Science Engineering	6	A
Exchange Programme in Bioinformatics (master's level)	6	A
Exchange Programme in Computer Science (master's level)	6	A

Teaching languages

Dutch, English

Keywords

Parallel and distributed software, communication, coordination, synchronization, efficiency, programming models, high performance computing, cloud computing, fault tolerance.

Position of the course

This course will teach students the concepts regarding the different aspects of the design and implementation of distributed software. The course will provide the students with a state-of-the-art overview of parallel and cloud computing systems, design of parallel algorithms, software engineering specifically for these applications, and management of high performance and cloud-based systems. The emphasis is on the software side and on the different programming models. Hardware/architecture aspects are assumed to be covered in other courses and are only used to the extent necessary to understand the impact on the software performance.

Contents

- Fundamentals, definitions & terminology, classification.
- Performance metrics & limiting factors: speedup, efficiency, scalability (strong & weak), high availability, Amdahl's and Gustafson's law, CAP theorem, network cost modeling, failure.
- Distributed software: message passing, remote procedure call (RPC), distributed objects, remote message invocation (RMI), request-reply protocol, marshalling. Message oriented middleware and services.
- Cloud computing models: service models, deployment models, payment models; cloud platforms and programming models.
- Data-driven architectures: scale out vs. scale up, move processing power to data, avoid random access, scalability; MapReduce; NoSQL; Big data for enterprise applications.
- High performance distributed-memory computing: MPI, point-to-point communication, collective communication, problem decomposition, case studies.
- Shared-memory programming: data protection, achieving concurrency, mutexes, semaphores, condition variables, deadlocks, false-sharing, thread-safety, programming models, case studies.
- Monitoring and management of large-scale parallel and distributed software systems: large-scale measurements and monitoring, autonomic and control theory based management.
- GPU programming: kernels, thread hierarchy, memory hierarchy, control flow.

Initial competences

Basic programming skills in C/C++ and Java. Basic knowledge of datastructures and algorithms. Basic knowledge of operating systems.

Final competences

1 To know and understand the principle algorithmic problems associated with parallel and distributed systems and the standard strategies to solve them.
2 To know the different functions of middleware, the principle architectures for realizing parallel and distributed systems, and the important software technologies for realizing parallel and distributed applications.
3 To be able to explain the differences between different parallel and distributed programming models.
4 To apply the basic strategies for solving algorithmic problems associated with parallel and distributed systems.
5 To be able to deliver a basic design for a parallel and distributed application, realize a parallel and distributed application, and estimate performances of different implementation alternatives.
6 To pay attention to scalability and performance issues at design time.
7 To evaluate algorithms for standard problems and applying them in the most appropriate way.
8 To pay sufficient time to evaluate different design alternatives prior to implementation.

Conditions for credit contract

Access to this course unit via a credit contract is determined after successful competences assessment

Conditions for exam contract

This course unit cannot be taken via an exam contract

Teaching methods

Guided self-study, lecture, practicum, self-reliant study activities, seminar: coached exercises

Learning materials and price

- Syllabus via VTK
- Slides on the electronic learning environment
- Source code examples via the electronic learning environment

References

- George Coulouris, Jean Dollimore, Tim Kindberg, Gordon Blair, “Distributed Systems: Concepts and Design (5th Edition)”, Pearson Publishers.
- Rajkumar Buyya, James Broberg, Andrzej M. Goscinski, “Cloud Computing: Principles and Paradigms”, Wiley Publishers.
- Cloud programming (online references)
- MPI, The complete reference (online)
- Peter Pacheco: “An Introduction to Parallel Programming”, Morgan Kaufmann.
- Ian Fostier, “Designing and Building Parallel Programs”, Addison-Wesley Inc.
- Jimmy Lin, Chris Dyer, “Data-intensive Text Processing with MapReduce”.

Course content-related study coaching

- Practicals are supervised by assistents.
- Additional information via the electronic learning environment.

Evaluation methods

end-of-term evaluation and continuous assessment

Examination methods in case of periodic evaluation during the first examination period

Written examination

Examination methods in case of periodic evaluation during the second examination period

Written examination

Examination methods in case of permanent evaluation

Participation, skills test, report

Possibilities of retake in case of permanent evaluation

examination during the second examination period is not possible

Extra information on the examination methods

- During examination period: written closed-book exam; written open-book exam.
- During semester: graded practicals and homework assignments.

Calculation of the examination mark

- 75% exam
- 25% practicals and homework assignment

In case the score for the examination is less than 8/20, the final score for this course will be limited to 8/20.