Modelling of Software-intensive Systems

2025 - 2026

Is part of the next programmes:

- M0012004 Master of Computer Science: Software Engineering
- M0012005 Master of Computer Science: Data Science and Artificial Intelligence
- M0012006 Master of Computer Science: Computer Networks
- M0048004 Master of Computer Science: Software Engineering
- M0048005 Master of Computer Science: Data Science and Artificial Intelligence
- M0048006 Master of Computer Science: Computer Networks
- M0090004 Master of Teaching in Science and Technology: Computer Science
- U0001008 Courses open to exchange students in Sciences

Course Code:	2001WETMSI
Study Domain:	Computer Science
Semester:	1E SEM
Contact Hours:	52

Credits:	6
Study Load (hours):	168
Contract Restrictions:	Exam contract not possible
Language of Instructions:	ENG
Lecturer(s):	T Hans Vangheluwe
Examperiod:	exam in the 1st semester

1. Prerequisites *

speaking and writing of:

- English

specific prerequisites for this course

- **Object-oriented programming. Both theory and assignment parts of the course use the object-oriented scripting language Python. If Python is not yet known, it is advisible to prepare for the course using a Python tutorial.**
- **Basics of object-oriented design (notions of UML, design patterns)**
- **Basic linear algebra and calculus**

2. Learning outcomes *

- Knowledge of different formalisms: various UML diagrams, Modelica, Causal Block Diagrams, Petri Nets, Statecharts, Event Scheduling, DEVS, Forrester System Dynamics. Understanding the similarities and differences between different formalisms. Be able to choose between (and explain why) and use appropriate formalisms for modelling, analysis, simulation and synthesis of diverse (software-intensive) applications. The above forms a starting point for more advanced topics. In particular, the combination of different formalisms.

3. Course contents *

This course will introduce you to the different kinds of complexity we have to deal with when designing large software-intensive systems. This complexity will be tackled using different modelling formalisms, each appropriate for specific problems/aspects: various UML diagrams, Modelica, Causal Block Diagrams (aka Synchronous Data Flow), Petri Nets, Statecharts, Event Scheduling/Activity Scanning/Process Interaction Discrete-Event, DEVS, Forrester System Dynamics. Control Theory will also be briefly introduced with focus on the development of an optimal (embedded, software) controller.

The goal of the course is to gain understanding of the similarities and differences between different formalisms. Modelling formalisms vary in the level of detail in which they consider time (e.g., partial order, discrete-time, continuous-time), whether they allow modelling of sequential or concurrent behaviour, whether they are deterministic (mostly suited for system/software synthesis) or non-deterministic (mostly suited for modelling system environment effects, with subsequent safety analysis), whether they support a notion of spatial distribution, ...

At the end of the course, you should be able to choose between (and explain why) and use appropriate formalisms for modelling, analysis, simulation and synthesis of diverse (software-intensive) applications.

The above forms a starting point for more advanced topics. In particular, the combination of different formalisms and the development of Domain-Specific Modelling Languages. The latter is one of the topics of the course Model Driven Engineering.

Course topics:

1. Causes of Complexity
2. Unified Modelling Language (UML) notations
3. Requirements vs. Design
4. System Specification
5. Equation-based Object-Oriented Languages (in particular, Modelica)
6. Basics of Control Theory (bang-bang, PID controller)
7. Causal Block Diagrams (CBDs): discrete-time en continuous-time
8. Petri Nets
9. Statecharts
10. Event-Scheduling Discrete-Event

11. Discrete-EVent System Specification (DEVS)
12. (Forrester) System Dynamics
13. Cellular Automata

4. International dimension *

- This course stimulates international and intercultural competences.
- Students use course materials in a foreign language.
- The lecturer invites international guest lecturers.
- Students give presentations in a foreign language.
- Students write papers in a foreign language.
- The lecturer collaborates with an international partner (fe. joint course materials, joint case studies).

5. Teaching method and planned learning activities

5.1 Used teaching methods *

Class contact teaching

- Lectures
- Practice sessions

Personal work

- Exercises

Assignments

- Individually
- In group

5.2 Planned learning activities and teaching methods

The 10 topics each take at least 1 week (Statecharts and DEVS each two weeks).

An assignment is given about each important formalism.

There are two types of assignments: (1) the implementation of the operational semantics of a formalism and (2) the use of a formalisms for the modelling/analysis/simulation/synthesis of a complex problem.

5.3 Facilities for working students *

Classroom activities

- no specific facilities

6. Assessment method and criteria *

6.1 Used assessment methods *

Examination

- Written examination without oral presentation
- - Closed book

Continuous assessment

- Exercises
- Assignments

6.2 Assessment criteria *

Evaluation consists of on the one hand a closed-book exam and on the other hand of a series of assignments during the semester. Assignments are made in teams of one ore two students. The exact weight of exam and assignments depends on concrete discussions during the first class of the semester. It is however typically 25% (exam) and 75% (assignments).

To pass the course, you need to attend/submit and orally defend every part (theory exam and each and every assignment) of the course. If not, your grade will be "AFW" - absent. If you do attend/submit every part, you still need an overall score of 50% to pass the course. Additionally, if for at least one part (theory exam, or any assignment) your score is strictly below 40%, your overall grade will be *min(7, your_score)*. *your_score* is the score you would get when applying the weights given above.

7. Study material

7.1 Required reading *

The material (articles and notes) are available on the public course website
    http://msdl.uantwerpen.be/people/hv/teaching/MoSIS/
The ultimate reference are notes on the blackboard during the classes.

7.2 Optional reading

The following study material can be studied voluntarily :

Open Access textbook:
Paulo Carreira, Vasco Amaral, Hans Vangheluwe. Foundations of Multi-Paradigm Modelling for Cyber-Physical Systems. Springer, 2020.
https://link.springer.com/book/10.1007/978-3-030-43946-0

8. Contact information *

- lecturer: Prof. Hans Vangheluwe
- assistants: Randy Paredis, Joeri Exelmans, Rakshit Mittal, Lucas Albertins

9. Tutoring