# Specification and Verification

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
- M0032004 Master of Mathematics: Financial and Applied Mathematics
- 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:** | 2001WETFSP |
| **Study Domain:** | Computer Science |
| **Semester:** | 1E SEM |

| | |
|---|---|
| **Contact Hours:** | 45 |
| **Credits:** | 6 |
| **Study Load (hours):** | 168 |
| **Contract Restrictions:** | No contract restriction |
| **Language of Instructions:** | ENG |
| **Lecturer(s):** | T Guillermo Alberto Perez |
| **Exameperiod:** | exam in the 1st semester |
| EVEN | |
| **Bi-annual course:** | Thought in academic years starting in an even year |

## 1. Prerequisites *

**speaking and writing of:**

- English

**reading and comprehending of:**

- English

**general notion of the basic concepts of**

**Discrete mathematics, models of software and hardware (such as Petri nets, graphs, etc.)**

**specific prerequisites for this course**

**You have successfully completed a number of more complex programming projects and you have a basic knowledge of logic, automata, and software engineering.**

## 2. Learning outcomes *

- Model reactive systems using mathematical formalisms
- Analyse these models using classical verification algorithms
- Understand and are able to convincingly explain the core mathematical concepts and algorithms for prominent techniques (almost to the level of a fully formal proof)
- Implement and use verification tools supporting the learned techniques

## 3. Course contents *

An important issue in the development of large software systems is the specification of their desired structural and dynamic properties. To avoid the impreciseness and ambiguities of specifications in natural language, a number of specialized formal languages, based on logic and mathematics, have been developed. Moreover, these languages are supported by algorithms and tools allowing various types of analysis and verification.
The course covers the following main topics:
* Formal verification
* Labelled transition systems as models of systems
* Linear temporal logic
* Computation tree logic
* Symbolic model checking
* Automatic controller synthesis

## 4. International dimension *

- This course stimulates international and intercultural competences.

## 5. Teaching method and planned learning activities

### 5.1 Used teaching methods *

### Class contact teaching

- Lectures
- Practice sessions
- Seminars/Tutorials

### Personal work

- Exercises

### Case studies

- Individually
- **Directed self-study**

### 5.2 Planned learning activities and teaching methods

The students will work on implementing a verification tool using binary decision diagrams (BDDs).

### 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
- - Multiple-choice
- - Exercises

**Continuous assessment**

- Assignments

**Other assessment methods**

- Project

**6.2 Assessment criteria ***

For the part of the course covered in the lectures, the students are expected to be able to specify formally desired behaviours of systems in linear temporal logic and/or computation tree logic. They should also be able to apply the algorithms learned during the lectures, and applied during the practicals, to determine whether a given model satisfies a particular specification. These will be evaluated during the exam which accounts for 60% of the final grade.
For the part about verification tools, the students are expected to implement a model checker using binary decision diagrams. This project will account for 40% of the grade (and can be turned in during the first session only).
The theory exam is closed book. However, students may bring **one cheat sheet (both sides)** with notes on the course contents. No cut-off point is used for multiple-choice questions and no negative points are assigned for incorrect answers. Up to 50% of the points from the exam can already be gained by completing assignments that are given during the practical sessions. This is not compulsory but highly encouraged as both the assistant and the professor provide feedback regarding the completed assignments and it is a great way to prepare for the exam (which is based on questions from the assignments).
Students may use generative AI tools for their assignment. In an oral exam of the assignment or during a contact moment, students are expected to elaborate on their assignment (and how they used generative AI).

# 7. Study material

### 7.1 Required reading *

Slides (via Blackboard)
Lecture notes (via Blackboard)

### 7.2 Optional reading

The following study material can be studied voluntarily :

Christel Baier and Joost-Pieter Katoen, Principles of model checking. MIT Press 2008, ISBN 978-0-262-02649-9

## 8. Contact information *

lecturer: guillermoalberto.perez@uantwerpen.be
teaching assistant: tim.leys@uantwerpen.be

## 9. Tutoring

The teacher is available for further questions