**MASTER**

# Computer Science and Engineering (CSE)

**TU/e**

**EINDHOVEN UNIVERSITY OF TECHNOLOGY**

**TUE.NL**

Software systems play an often unseen yet highly important role in our society. Consider, for example, the systems of banks or insurance companies, or the software in a television set, internet search engines or software that controls the electricity network. We cannot manage without them.

As a graduate of the Master Computer Science and Engineering program, you will be able to develop algorithms to solve computational problems, design quality software and analyze complex systems. You will also combine a scientific perspective with the approach of an engineer. You will systematically produce inventories of requirements and incorporate them into your design. You will also be able to make qualitative and quantitative judgments regarding the extent to which those requirements have been met.

The CSE curriculum is structured in three **focus areas**, which are described below. These focus areas ensure a comprehensive and broad set of courses and provide guidance for the directions in which to specialize.

| Program structure | Total ≥ 120 EC |
|---|---|
| **Foundational courses*** | 15 EC |
| **Specialization**** | 15 EC |
| **CSE electives**** | 30 EC |
| **Free electives**** | 15 EC |
| **Seminar** | 5 EC |
| **Graduation project***** | 10+30 EC |

\* One course per focus area
\*\* Three courses from the same focus area, in addition to a foundational course.
\*\*\* Electives may include homologation courses, an internship and/or courses from other programs or universities, depending on relevance and permission.
\*\*\*\* The graduation project consists of a preparation phase (10 EC) and a Master's Thesis project (30 EC).

## ALGORITHMS & THEORY

A deep mathematical understanding of computation and semantics is indispensable for reasoning about the quality and efficiency of algorithms, data structures and (concurrent) systems. This focus area thus encompasses, for example, improving and understanding trade-offs between algorithm efficiency and quality, exploring and pushing the limits of computation, modelling and (manually, mechanically or fully automatically) verifying computational and/or concurrent systems.

| Quarter | Code | Course |
|---------|------|--------|
| **Foundational courses** | | |
| 1 | 2IMA10 | Advanced Algorithms |
| 1 | 2IMF25 | Automated Reasoning |
| 4 | 2IMF10 | Process Algebra |
| **Deepening courses** | | |
| 1 | 2IMA20 | Algorithms for Geographic Data |
| 2 | 2IMA15 | Geometric Algorithms |
| 2 | 2IMA35 | Massively Parallel Algorithms |
| 3 | 2IMA25 | Exact Algorithms for NP-hard Problems |
| 3 | 2IMF15 | Proving with computer assistance |
| 4 | 2IMA30 | Topological Data Analysis |

## ARCHITECTURE & SYSTEMS

Modern digital systems involve complex interactions between various hardware and software components operating under functional and non-functional requirements. This focus area addresses the understanding and management of the architecture, interactions, behavior, and trade-offs in such systems. In particular, it focuses on the theory and practice for the modeling, design, implementation, analysis and verification of complex networked, embedded, large-scale and data-intensive systems.

| Quarter | Code | Course |
|---------|------|--------|
| **Foundational courses** | | |
| 1 | 2IMN10 | Architecture of Distributed Systems |
| 1 | 2IMF30 | System Validation |
| 2 | 2IMD10 | Engineering of data systems |
| **Deepening courses** | | |
| 2 | 2IMN15 | Internet of Things |
| 2 | 2IMN20 | Real-time systems |
| 2 | 2IMF35 | Algorithms for model checking |
| 3 | 2IMN25 | Quantitative evaluation of ES |
| 4 | 2IMN35 | VLSI programming |

## SOFTWARE & ANALYTICS

Software is a key enabler in Computer Science. The development of software should be efficient and result in high-quality software. This focus area addresses the development of high-quality software in an efficient way. It does so by providing knowledge on developing correct software by construction and by combining principles and methodology of software development with analysis of information sources, specifically by mining software repositories to understand the effects of software evolution.

| Quarter | Code | Course |
|---------|------|--------|
| **Foundational courses** | | |
| 1 | 2AMM20 | Foundations of Data Mining |
| 2 | 2IMP10 | Program Verification Techniques |
| 3 | 2IMP25 | Software Evolution |
| **Deepening courses** | | |
| 1 | 2IMN30 | Machine Learning for Industry |
| 2 | 2IMP40 | Applications of Data Science for Software Engineering |
| 3 | 2AMV10 | Visual Analytics |
| 3 | 2IMD15 | Big Data Management |
| 4 | 2IMP20 | Domain Specific Language Design |
| 4 | 2IMP30 | System Design Engineering |