Computer Science MSc

VU University Amsterdam - Faculteit der Exacte Wetenschappen - M Computer Science - 2015-2016

Computer Science studies the technology that has become ubiquitous in our global, connected society. Traditionally, the computer had been the primary object of study. Nowadays, globally distributed information processing services have taken center stage, with the Internet connecting a wide variety of information processing devices, ranging from mobile phones to data centers operated by the world leadership companies.

The technical side of Computer Science deals with computer operations, like system software, computer networks, and programming environments. The theoretical foundations cover, for example, limits of what can be computed, computational efficiency, correctness, and the intricacies of concurrent execution. Software engineering studies construction and maintenance of large and often mission-critical software systems that need to be maintained by large teams of people. Because of its prominent role in everyday life, non-functional aspects of information technology are gaining importance in Computer Science, most notably the energy efficiency of data centers, and the security of computer systems.

Students aiming to enroll in the Master Computer Science are required to have a solid background on the level of a Bachelor of Science in Computer Science, or similar. The program is structured in multiple, focused specializations from which a student chooses one according to personal preferences. The program is organized in close collaboration with the University of Amsterdam.
From the academic year 2016 on, students will be awarded a joint degree from both universities. Classes are taught both at the VU campus, and at Science Park Amsterdam. Both locations are conveniently connected by both public transportation and bicycle paths.

Index

Research Variant Internet and Web Technology

The Internet and the World Wide Web play an ever more central role in our society. This specialisation is concerned with large-scale computer systems, especially computer networks and the Internet. Important topics are: Internet and Web protocols, distributed systems, network security, development tools for network applications, peer-to-peer technology, etc.

Note: Every programme, including the choice of optional courses, has to be discussed and agreed upon with the master coordinator or a personal mentor and approved by the Examination Board.

Master Coordinator:

dr. S. Voulgaris
K room R-447
T +31 (0) 20 598 3715
E spyros.voulgaris@vu.nl

Programme components:

- Constrained Choice Mathematics
- Limited offered course
- Suggested elective courses
- Pre-approved Elective courses
- Core track courses
- Mastercore

Constrained Choice Mathematics

Compulsory choice of at least one Mathematics course of 6 credits, recommended are the courses below.

Courses:

Name	Period	Credits	Code
Coding and Cryptography	Period 1	6.0	X_405041
Experimental Design and Data Analysis	Period 5	6.0	X_405078

Limited offered course

This course is taught this year only at our University. Philippe Kruchten is Professor of Software Engineering at the University of British Columbia in Vancouver. He is world-famous as the chief designer of the Rational Unified Process (RUP) and currently he is doing research on Agile Architectures. He will give this course specially for our Master Computer Science and Master Information Sciences students.

Courses:

Name	Period	Credits	Code
Watson Innovation	Period 2	6.0	X_405129

Suggested elective courses

Programme components:

- Suggested elective courses Foundations of Computing and Concurrency
- Suggested elective courses Programming
- Suggested elective courses Software Engineering

Suggested elective courses Foundations of Computing and Concurrency

Courses:

Name	Period	Credits	Code
Advanced Logic	Period 4	6.0	X_405048
Concurrency and Multithreading	Period 1	6.0	X_405064
Concurrency Theory	Period 1	6.0	X_418103
Logical Verification		6.0	X_400115
Protocol Validation	Period 5	6.0	X_400117

Suggested elective courses Programming

Courses:

Name	Period	Credits	Code
Computer Networks Practical	Period 5+6	6.0	X_405072
Individual Systems Practical	Ac. Year (September)	6.0	X_405088
Operating Systems Practical	Ac. Year (September)	6.0	X_405071
Parallel Programming Practical	Period 2+3	6.0	X_400162
Programming Concurrent Systems	Period 2	6.0	X_418109
Software Testing Practical	Period 6	6.0	X_405124

Suggested elective courses Software Engineering

Courses:

Name	Period	Credits	Code

Software Architecture	Period 2	6.0	X_400170
Software Asset Management	Period 1, Period 5	6.0	X_400412
Software Metrics	Period 4	6.0	X_405121
Software Testing	Period 5	6.0	X_400439

Pre-approved Elective courses

Courses:

Name	Period	Credits	Code
Advanced Selforganisation	Period 2	6.0	X_400434
Advances in Computer Architecture	Period 1	6.0	X_418047
Binary and Malware Analysis	Period 1	6.0	X_405100
Business Process Management	Period 1	6.0	X_405115
Data Mining Techniques	Period 5	6.0	X_400108
Developing Services for the Cloud	Period 3	6.0	X_405074
Evolutionary Computing	Period 1	6.0	X_400111
Green Lab	Period 1	6.0	X_418158
ICT4D: Information and communication technology for Development	Period 5	6.0	X_405101
Industrial Internship	Ac. Year (September)	6.0	X_405080
Introduction to Computational Science	Period 1	6.0	X_418111
Knowledge and Media	Period 1	6.0	X_405065
Knowledge Engineering	Period 2+3	6.0	X_405099
Lambda Calculus	Period 2	6.0	X_418108
Large Scale Data Engineering	Period 4	6.0	X_405116
Large-Scale Computing Infrastructures	Period 5	6.0	X_405106
Neural Networks	Period 1	6.0	X_400132
Parallel Programming for High-performance Applications	Period 1	6.0	X_400161
Performance of Networked Systems	Period 4	6.0	X_405105
Serious Games	Period 5	6.0	X_405097
Systems Security	Period 4	6.0	X_405108
The Social Web	Period 4	6.0	X_405086

Core track courses

Courses:

Name	Period	Credits	Code
Distributed Algorithms	Period 2	6.0	X_400211
Distributed Systems	Period 2	6.0	X_400130
Internet programming	Period 1	6.0	X_405082
Performance of Networked Systems	Period 4	6.0	X_405105
Service Oriented Design	Period 1	6.0	X_405061
Web Services and Cloud-based Systems	Period 5	6.0	X_418110

Mastercore

Courses:

Name	Period	Credits	Code
History of digital cultures	Period 3	6.0	X_418107
Literature Study and Seminar	Ac. Year (September)	6.0	X_405111
Master Project	Ac. Year (September)	36.0	X_400442

Research Variant High-Performance Computing

High performance computing seeks to solve computing problems as fast and as efficiently as possible. The most important approach is to use a (large) number of computers instead of one, and let these computers work together, in parallel. In clusters and grids, the computers are typically distributed across an organization (a university, for example), a country, or even the globe. This had led to the term "High Performance Computing."

Note: Every programme, including the choice of optional courses, has to be discussed and agreed upon with the master coordinator or a personal mentor and approved by the Examination Board.

Master Coordinator:

dr. S. Voulgaris
K room R-447
T +31 (0) 20 598 3715
E spyros.voulgaris@vu.nl

Programme components:

- Constrained choice Foundations of Computing and Concurrency (6 EC)

- Constrained Choice Mathematics
- Limited offered course
- Pre-approved Elective courses
- Constrained choice Programming (6 EC)
- Constrained Choice Software Engineering (6EC)
- Core Track Courses
- Mastercore

Constrained choice Foundations of Computing and Concurrency (6 EC)

Compulsory choice Theoretical Computer Science at least 6 credits, recommended are the courses below.

Courses:

Name	Period	Credits	Code
Advanced Logic	Period 4	6.0	X_405048
Concurrency and Multithreading	Period 1	6.0	X_405064
Concurrency Theory	Period 1	6.0	X_418103
Distributed Algorithms	Period 2	6.0	X_400211
Logical Verification		6.0	X_400115
Protocol Validation	Period 5	6.0	X_400117

Constrained Choice Mathematics

Compulsory choice of at least one Mathematics course of 6 credits, recommended are the courses below.

Courses:

Name	Period	Credits	Code
Coding and Cryptography	Period 1	6.0	X_405041
Experimental Design and Data Analysis	Period 5	6.0	X_405078

Limited offered course

This course is taught this year only at our University. Philippe Kruchten is Professor of Software Engineering at the University of British Columbia in Vancouver. He is world-famous as the chief designer of the Rational Unified Process (RUP) and currently he is doing research on Agile Architectures. He will give this course specially for our Master Computer Science and Master Information Sciences students.

Courses:

Name	Period	Credits	Code
Watson Innovation	Period 2	6.0	X_405129

Pre-approved Elective courses

Courses:

Name	Period	Credits	Code
Advanced Selforganisation	Period 2	6.0	X_400434
Advances in Computer Architecture	Period 1	6.0	X_418047
Binary and Malware Analysis	Period 1	6.0	X_405100
Business Process Management	Period 1	6.0	X_405115
Data Mining Techniques	Period 5	6.0	X_400108
Developing Services for the Cloud	Period 3	6.0	X_405074
Evolutionary Computing	Period 1	6.0	X_400111
Green Lab	Period 1	6.0	X_418158
ICT4D: Information and communication technology for Development	Period 5	6.0	X_405101
Industrial Internship	Ac. Year (September)	6.0	X_405080
Introduction to Computational Science	Period 1	6.0	X_418111
Knowledge and Media	Period 1	6.0	X_405065
Knowledge Engineering	Period 2+3	6.0	X_405099
Lambda Calculus	Period 2	6.0	X_418108
Large Scale Data Engineering	Period 4	6.0	X_405116
Large-Scale Computing Infrastructures	Period 5	6.0	X_405106
Neural Networks	Period 1	6.0	X_400132
Parallel Programming for High-performance Applications	Period 1	6.0	X_400161
Performance of Networked Systems	Period 4	6.0	X_405105
Serious Games	Period 5	6.0	X_405097
Systems Security	Period 4	6.0	X_405108
The Social Web	Period 4	6.0	X_405086

Constrained choice Programming (6 EC)

Courses:

Name	Period	Credits	Code

Computer Networks Practical	Period 5+6	6.0	X_405072
Individual Systems Practical	Ac. Year (September)	6.0	X_405088
Internet programming	Period 1	6.0	X_405082
Operating Systems Practical	Ac. Year (September)	6.0	X_405071
Programming Concurrent Systems	Period 2	6.0	X_418109
Software Testing Practical	Period 6	6.0	X_405124

Constrained Choice Software Engineering (6EC)

Courses:

Name	Period	Credits	Code
Service Oriented Design	Period 1	6.0	X_405061
Software Architecture	Period 2	6.0	X_400170
Software Asset Management	Period 1, Period 5	6.0	X_400412
Software Metrics	Period 4	6.0	X_405121
Software Testing	Period 5	6.0	X_400439

Core Track Courses

Courses:

Name	Period	Credits	Code
Advances in Computer Architecture	Period 1	6.0	X_418047
Distributed Systems	Period 2	6.0	X_400130
Large-Scale Computing Infrastructures	Period 5	6.0	X_405106
Parallel Programming for High-performance Applications	Period 1	6.0	X_400161
Parallel Programming Practical	Period 2+3	6.0	X_400162
Performance of Networked Systems	Period 4	6.0	X_405105

Mastercore

Courses:

Name	Period	Credits	Code

History of digital cultures	Period 3	6.0	X_418107
Literature Study and Seminar	Ac. Year (September)	6.0	X_405111
Master Project	Ac. Year (September)	36.0	X_400442

Research variant Software Engineering & Green IT

Some people define software engineering as: 'the application of a systematic, quantifiable approach to the development, execution and maintenance of software. It is a broad and comprehensive field, in which engineering plays an important part, next to psychological and managerial aspects. Keywords are evolution and complexity.
The field continually evolves, as the type of systems as well as the world at large changes. New developments such as outsourcing, global system development, service-orientation and the incorporation of off-the-shelf software profoundly influence the field.

Note: Every programme, including the choice of optional courses, has to be discussed and agreed upon with the master coordinator or a personal mentor and approved by the Examination Board.

Master Coordinator:

Prof.dr. Patricia Lago
K room WN-T422
T +31 (0) 20 598 82 (secr.)
E p.lago@vu.nl

Programme components:

- Constrained Choice Mathematics
- Limited offered course
- Pre-approved Elective courses
- Constrained choice Foundations of Computing and Concurrency (6 EC)
- Constrained choice Programming (6 EC)
- Compulsory Courses
- Mastercore

Constrained Choice Mathematics

Compulsory choice of at least one Mathematics course of 6 credits, recommended are the courses below.

Courses:

Name	Period	Credits	Code
Coding and Cryptography	Period 1	6.0	X_405041
Experimental Design and Data Analysis	Period 5	6.0	X_405078

Limited offered course

This course is taught this year only at our University. Philippe Kruchten is Professor of Software Engineering at the University of British Columbia in Vancouver. He is world-famous as the chief designer of the Rational Unified Process (RUP) and currently he is doing research on Agile Architectures. He will give this course specially for our Master Computer Science and Master Information Sciences students.

Courses:

Name	Period	Credits	Code
Watson Innovation	Period 2	6.0	X_405129

Pre-approved Elective courses

Courses:

Name	Period	Credits	Code
Advanced Selforganisation	Period 2	6.0	X_400434
Advances in Computer Architecture	Period 1	6.0	X_418047
Binary and Malware Analysis	Period 1	6.0	X_405100
Business Process Management	Period 1	6.0	X_405115
Data Mining Techniques	Period 5	6.0	X_400108
Developing Services for the Cloud	Period 3	6.0	X_405074
Evolutionary Computing	Period 1	6.0	X_400111
Green Lab	Period 1	6.0	X_418158
ICT4D: Information and communication technology for Development	Period 5	6.0	X_405101
Industrial Internship	Ac. Year (September)	6.0	X_405080
Introduction to Computational Science	Period 1	6.0	X_418111
Knowledge and Media	Period 1	6.0	X_405065
Knowledge Engineering	Period 2+3	6.0	X_405099
Lambda Calculus	Period 2	6.0	X_418108
Large Scale Data Engineering	Period 4	6.0	X_405116
Large-Scale Computing Infrastructures	Period 5	6.0	X_405106
Neural Networks	Period 1	6.0	X_400132
Parallel Programming for High-performance Applications	Period 1	6.0	X_400161
Performance of Networked Systems	Period 4	6.0	X_405105

Serious Games	Period 5	6.0	X_405097
Systems Security	Period 4	6.0	X_405108
The Social Web	Period 4	6.0	X_405086

Constrained choice Foundations of Computing and Concurrency (6 EC)

Compulsory choice Theoretical Computer Science at least 6 credits, recommended are the courses below.

Note: Every programme, including the choice of optional courses, has to be discussed and agreed upon with the master coordinator or a personal mentor and approved by the Examination Board.

Courses:

Name	Period	Credits	Code
Advanced Logic	Period 4	6.0	X_405048
Concurrency and Multithreading	Period 1	6.0	X_405064
Concurrency Theory	Period 1	6.0	X_418103
Distributed Algorithms	Period 2	6.0	X_400211
Logical Verification		6.0	X_400115

Constrained choice Programming (6 EC)

Courses:

Name	Period	Credits	Code
Computer Networks Practical	Period 5+6	6.0	X_405072
Individual Systems Practical	Ac. Year (September)	6.0	X_405088
Internet programming	Period 1	6.0	X_405082
Operating Systems Practical	Ac. Year (September)	6.0	X_405071
Parallel Programming Practical	Period 2+3	6.0	X_400162
Programming Concurrent Systems	Period 2	6.0	X_418109
Software Testing Practical	Period 6	6.0	X_405124

Compulsory Courses

Courses:

Name	Period	Credits	Code
Distributed Systems	Period 2	6.0	X_400130

Green Lab	Period 1	6.0	X_418158
Service Oriented Design	Period 1	6.0	X_405061
Software Architecture	Period 2	6.0	X_400170
Software Asset Management	Period 1, Period 5	6.0	X_400412
Software Metrics	Period 4	6.0	X_405121
Software Testing	Period 5	6.0	X_400439

Mastercore

Courses:

Name	Period	Credits	Code
History of digital cultures	Period 3	6.0	X_418107
Literature Study and Seminar	Ac. Year (September)	6.0	X_405111
Master Project	Ac. Year (September)	36.0	X_400442

Research Variant Foundations of Computing and Concurrency

This track aims at Computer Science students with a general interest in the application of formal methods in computing, concurrency and the design and verification of software systems. Some theoretical disciplines that play a central role are term rewriting, process algebra, distributed algorithms and type theory. Foundational disciplines include logic, recursion theory and complexity.

All these topics have a wide range of applications, of which we mention just a few. Tools developed from process algebra are used in protocol validation. Term rewriting is used in the execution of equational specifications and lies at the basis of functional programming and the analysis of infinitary processes. Distributed algorithms are of central importance for the efficient use of concurrent systems. Logic and type theory form the basis of proof checking, used in software verification.

In all of the above areas courses are offered. To mention a few examples: Distributed Algorithms, Logical Verification, Protocol Validation, Term Rewriting Systems, Concurrent System Design by Abstraction. The programme can be enhanced by choosing one or more appropriate mathematics courses.

Note: Every programme, including the choice of optional courses, has to be discussed and agreed upon with the master coordinator or a personal mentor and approved by the Examination Board.

Master Coordinator:

dr. F. van Raamsdonk
K room T-446
T +31 (0) 20 598 7710
E f.van.raamsdonk@vu.nl

Programme components:

- Constrained Choice Mathematics
- Limited offered course
- Pre-approved Elective courses
- Constrained Choices
- Constrained Choice Programming
- Constrained Choice Software Engineering (6EC)
- Compulsory Courses
- Mastercore

Constrained Choice Mathematics

Compulsory choice of at least one Mathematics course of 6 credits, recommended are the courses below.

Courses:

Name	Period	Credits	Code
Coding and Cryptography	Period 1	6.0	X_405041
Experimental Design and Data Analysis	Period 5	6.0	X_405078

Limited offered course

This course is taught this year only at our University. Philippe Kruchten is Professor of Software Engineering at the University of British Columbia in Vancouver. He is world-famous as the chief designer of the Rational Unified Process (RUP) and currently he is doing research on Agile Architectures. He will give this course specially for our Master Computer Science and Master Information Sciences students.

Courses:

Name	Period	Credits	Code
Watson Innovation	Period 2	6.0	X_405129

Pre-approved Elective courses

Courses:

Name	Period	Credits	Code
Advanced Selforganisation	Period 2	6.0	X_400434
Advances in Computer Architecture	Period 1	6.0	X_418047
Binary and Malware Analysis	Period 1	6.0	X_405100

Business Process Management	Period 1	6.0	X_405115
Data Mining Techniques	Period 5	6.0	X_400108
Developing Services for the Cloud	Period 3	6.0	X_405074
Evolutionary Computing	Period 1	6.0	X_400111
Green Lab	Period 1	6.0	X_418158
ICT4D: Information and communication technology for Development	Period 5	6.0	X_405101
Industrial Internship	Ac. Year (September)	6.0	X_405080
Introduction to Computational Science	Period 1	6.0	X_418111
Knowledge and Media	Period 1	6.0	X_405065
Knowledge Engineering	Period 2+3	6.0	X_405099
Lambda Calculus	Period 2	6.0	X_418108
Large Scale Data Engineering	Period 4	6.0	X_405116
Large-Scale Computing Infrastructures	Period 5	6.0	X_405106
Neural Networks	Period 1	6.0	X_400132
Parallel Programming for High-performance Applications	Period 1	6.0	X_400161
Performance of Networked Systems	Period 4	6.0	X_405105
Serious Games	Period 5	6.0	X_405097
Systems Security	Period 4	6.0	X_405108
The Social Web	Period 4	6.0	X_405086

Constrained Choices

Programme components:

- Constrained choice Verification
- Constrained choice Models of Computation
- Constrained Choice Concurrency
- Constrained choice Foundations of Computation

Constrained choice Verification

Courses:

Name	Period	Credits	Code
Logical Verification		6.0	X_400115
Protocol Validation	Period 5	6.0	X_400117

Constrained choice Models of Computation

Courses:

Name	Period	Credits	Code
Lambda Calculus	Period 2	6.0	X_418108
Term Rewriting Systems	Period 2	6.0	X_400121

Constrained Choice Concurrency

Courses:

Name	Period	Credits	Code
Concurrency and Multithreading	Period 1	6.0	X_405064
Concurrency Theory	Period 1	6.0	X_418103

Constrained choice Foundations of Computation

Courses:

Name	Period	Credits	Code
Computational Complexity	Period 1	6.0	X_417017
Recursion Theory	Period 1	6.0	X_400534

Constrained Choice Programming

Compulsory choice Practical Work Computer Science at least 6 credits, recommended are the courses below.

Note: Every programme, including the choice of optional courses, has to be discussed and agreed upon with the master coordinator or a personal mentor and approved by the Examination Board.

Courses:

Name	Period	Credits	Code
Computer Networks Practical	Period 5+6	6.0	X_405072
Individual Systems Practical	Ac. Year (September)	6.0	X_405088
Internet programming	Period 1	6.0	X_405082
Operating Systems Practical	Ac. Year (September)	6.0	X_405071

Parallel Programming Practical	Period 2+3	6.0	X_400162
Programming Concurrent Systems	Period 2	6.0	X_418109
Software Testing Practical	Period 6	6.0	X_405124

Constrained Choice Software Engineering (6EC)

Courses:

Name	Period	Credits	Code
Service Oriented Design	Period 1	6.0	X_405061
Software Architecture	Period 2	6.0	X_400170
Software Asset Management	Period 1, Period 5	6.0	X_400412
Software Metrics	Period 4	6.0	X_405121
Software Testing	Period 5	6.0	X_400439

Compulsory Courses

Courses:

Name	Period	Credits	Code
Advanced Logic	Period 4	6.0	X_405048
Distributed Algorithms	Period 2	6.0	X_400211

Mastercore

Courses:

Name	Period	Credits	Code
History of digital cultures	Period 3	6.0	X_418107
Literature Study and Seminar	Ac. Year (September)	6.0	X_405111
Master Project	Ac. Year (September)	36.0	X_400442

Research Variant Technical Artificial Intelligence

In this specialization the realisation of intelligent computer programs is the central subject. Artificial intelligence uses a great number of techniques from computer science and also plays a part in the development of these techniques, often inspired by human cognition. In this programme the student can choose between the existing techniques. Analysing, modelling and implementing of human knowledge, leading to a computer program that can reason with symbolic representations of this

knowledge is the subject of Knowledge Technology. In Knowledge Discovery and Data Mining the computer is used for recognition of structures in raw data, from which conclusions can be drawn.

Note: Every programme, including the choice of optional courses, has to be discussed and agreed upon with the master coordinator or a personal mentor and approved by the Examination Board.

Master Coordinator:

dr. M. Hoogendoorn
K room T-333
T +31 (0) 20 598 7772
E m.hoogendoorn@few.vu.nl

Programme components:

- Constrained Choice Mathematics
- Limited offered course
- Pre-approved Elective courses
- Constrained choice Foundations of Computing and Concurrency (6 EC)
- Constrained Choice Programming
- Constrained Choice Software Engineering (6EC)
- Compulsory Courses
- Mastercore

Constrained Choice Mathematics

Compulsory choice of at least one Mathematics course of 6 credits, recommended are the courses below.

Courses:

Name	Period	Credits	Code
Coding and Cryptography	Period 1	6.0	X_405041
Experimental Design and Data Analysis	Period 5	6.0	X_405078

Limited offered course

This course is taught this year only at our University. Philippe Kruchten is Professor of Software Engineering at the University of British Columbia in Vancouver. He is world-famous as the chief designer of the Rational Unified Process (RUP) and currently he is doing research on Agile Architectures. He will give this course specially for our Master Computer Science and Master Information Sciences students.

Courses:

Name	Period	Credits	Code
Watson Innovation	Period 2	6.0	X_405129

Pre-approved Elective courses

Courses:

Name	Period	Credits	Code
Advanced Selforganisation	Period 2	6.0	X_400434
Advances in Computer Architecture	Period 1	6.0	X_418047
Binary and Malware Analysis	Period 1	6.0	X_405100
Business Process Management	Period 1	6.0	X_405115
Data Mining Techniques	Period 5	6.0	X_400108
Developing Services for the Cloud	Period 3	6.0	X_405074
Evolutionary Computing	Period 1	6.0	X_400111
Green Lab	Period 1	6.0	X_418158
ICT4D: Information and communication technology for Development	Period 5	6.0	X_405101
Industrial Internship	Ac. Year (September)	6.0	X_405080
Introduction to Computational Science	Period 1	6.0	X_418111
Knowledge and Media	Period 1	6.0	X_405065
Knowledge Engineering	Period 2+3	6.0	X_405099
Lambda Calculus	Period 2	6.0	X_418108
Large Scale Data Engineering	Period 4	6.0	X_405116
Large-Scale Computing Infrastructures	Period 5	6.0	X_405106
Neural Networks	Period 1	6.0	X_400132
Parallel Programming for High-performance Applications	Period 1	6.0	X_400161
Performance of Networked Systems	Period 4	6.0	X_405105
Serious Games	Period 5	6.0	X_405097
Systems Security	Period 4	6.0	X_405108
The Social Web	Period 4	6.0	X_405086

Constrained choice Foundations of Computing and Concurrency (6 EC)

Compulsory choice Theoretical Computer Science at least 6 credits, recommended are the courses below.

Note: Every programme, including the choice of optional courses, has to be discussed and agreed upon with the master coordinator or a personal mentor and approved by the Examination Board.

Courses:

Name	Period	Credits	Code
Advanced Logic	Period 4	6.0	X_405048
Concurrency and Multithreading	Period 1	6.0	X_405064
Concurrency Theory	Period 1	6.0	X_418103
Distributed Algorithms	Period 2	6.0	X_400211
Logical Verification		6.0	X_400115

Constrained Choice Programming

Compulsory choice Practical Work Computer Science at least 6 credits, recommended are the courses below.

Note: Every programme, including the choice of optional courses, has to be discussed and agreed upon with the master coordinator or a personal mentor and approved by the Examination Board.

Courses:

Name	Period	Credits	Code
Computer Networks Practical	Period 5+6	6.0	X_405072
Individual Systems Practical	Ac. Year (September)	6.0	X_405088
Internet programming	Period 1	6.0	X_405082
Operating Systems Practical	Ac. Year (September)	6.0	X_405071
Parallel Programming Practical	Period 2+3	6.0	X_400162
Programming Concurrent Systems	Period 2	6.0	X_418109
Software Testing Practical	Period 6	6.0	X_405124

Constrained Choice Software Engineering (6EC)

Courses:

Name	Period	Credits	Code
Service Oriented Design	Period 1	6.0	X_405061
Software Architecture	Period 2	6.0	X_400170
Software Asset Management	Period 1, Period 5	6.0	X_400412
Software Metrics	Period 4	6.0	X_405121
Software Testing	Period 5	6.0	X_400439

Compulsory Courses

Courses:

Name	Period	Credits	Code
Distributed Systems	Period 2	6.0	X_400130
Evolutionary Computing	Period 1	6.0	X_400111
Knowledge Engineering	Period 2+3	6.0	X_405099
Model-based Intelligent Environments	Period 1	6.0	X_405056
Neural Networks	Period 1	6.0	X_400132

Mastercore

Courses:

Name	Period	Credits	Code
History of digital cultures	Period 3	6.0	X_418107
Literature Study and Seminar	Ac. Year (September)	6.0	X_405111
Master Project	Ac. Year (September)	36.0	X_400442

Research Variant Computer Systems and Security

Programme components:

- Constrained choice Foundations of Computing and Concurrency (6 EC)
- Constrained Choice Mathematics
- Constrained choice Programming (6EC)
- Constrained choice Software Engineering (6 EC)
- Limited offered course
- Pre-approved Elective courses
- Compulsory Courses
- Mastercore

Constrained choice Foundations of Computing and Concurrency (6 EC)

Compulsory choice Theoretical Computer Science at least 6 credits,
recommended are the courses below.

Courses:

Name	Period	Credits	Code
Advanced Logic	Period 4	6.0	X_405048
Concurrency and Multithreading	Period 1	6.0	X_405064

Concurrency Theory	Period 1	6.0	X_418103
Distributed Algorithms	Period 2	6.0	X_400211
Logical Verification		6.0	X_400115
Protocol Validation	Period 5	6.0	X_400117

Constrained Choice Mathematics

Compulsory choice of at least one Mathematics course of 6 credits, recommended are the courses below.

Courses:

Name	Period	Credits	Code
Coding and Cryptography	Period 1	6.0	X_405041
Experimental Design and Data Analysis	Period 5	6.0	X_405078

Constrained choice Programming (6EC)

Each student must follow at least one of the following programming-related courses

Courses:

Name	Period	Credits	Code
Computer Networks Practical	Period 5+6	6.0	X_405072
Individual Systems Practical	Ac. Year (September)	6.0	X_405088
Internet programming	Period 1	6.0	X_405082
Operating Systems Practical	Ac. Year (September)	6.0	X_405071
Parallel Programming Practical	Period 2+3	6.0	X_400162
Software Testing Practical	Period 6	6.0	X_405124

Constrained choice Software Engineering (6 EC)

Each student must follow at least 6EC from the core track courses of Track Software Engineering

Courses:

Name	Period	Credits	Code
Service Oriented Design	Period 1	6.0	X_405061
Software Architecture	Period 2	6.0	X_400170
Software Asset Management	Period 1, Period 5	6.0	X_400412
Software Metrics	Period 4	6.0	X_405121

Software Testing	Period 5	6.0	X_400439

Limited offered course

This course is taught this year only at our University. Philippe Kruchten is Professor of Software Engineering at the University of British Columbia in Vancouver. He is world-famous as the chief designer of the Rational Unified Process (RUP) and currently he is doing research on Agile Architectures. He will give this course specially for our Master Computer Science and Master Information Sciences students.

Courses:

Name	Period	Credits	Code
Watson Innovation	Period 2	6.0	X_405129

Pre-approved Elective courses

Courses:

Name	Period	Credits	Code
Advanced Selforganisation	Period 2	6.0	X_400434
Advances in Computer Architecture	Period 1	6.0	X_418047
Binary and Malware Analysis	Period 1	6.0	X_405100
Business Process Management	Period 1	6.0	X_405115
Data Mining Techniques	Period 5	6.0	X_400108
Developing Services for the Cloud	Period 3	6.0	X_405074
Evolutionary Computing	Period 1	6.0	X_400111
Green Lab	Period 1	6.0	X_418158
ICT4D: Information and communication technology for Development	Period 5	6.0	X_405101
Industrial Internship	Ac. Year (September)	6.0	X_405080
Introduction to Computational Science	Period 1	6.0	X_418111
Knowledge and Media	Period 1	6.0	X_405065
Knowledge Engineering	Period 2+3	6.0	X_405099
Lambda Calculus	Period 2	6.0	X_418108
Large Scale Data Engineering	Period 4	6.0	X_405116
Large-Scale Computing Infrastructures	Period 5	6.0	X_405106
Neural Networks	Period 1	6.0	X_400132

Parallel Programming for High-performance Applications	Period 1	6.0	X_400161
Performance of Networked Systems	Period 4	6.0	X_405105
Serious Games	Period 5	6.0	X_405097
Systems Security	Period 4	6.0	X_405108
The Social Web	Period 4	6.0	X_405086

Compulsory Courses

Courses:

Name	Period	Credits	Code
Advanced Networking	Period 5	6.0	X_417018
Advances in Computer Architecture	Period 1	6.0	X_418047
Binary and Malware Analysis	Period 1	6.0	X_405100
Distributed Systems	Period 2	6.0	X_400130
Programming Concurrent Systems	Period 2	6.0	X_418109
Systems Security	Period 4	6.0	X_405108

Mastercore

Courses:

Name	Period	Credits	Code
History of digital cultures	Period 3	6.0	X_418107
Literature Study and Seminar	Ac. Year (September)	6.0	X_405111
Master Project	Ac. Year (September)	36.0	X_400442

Expired Courses

The course modules presented in the list below will no longer be offered in academic year 2015-2016.

Advanced Logic

Course code	X_405048 (405048)
Period	Period 4
Credits	6.0
Language of tuition	English

Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. F. van Raamsdonk
Examinator	dr. F. van Raamsdonk
Teaching staff	dr. F. van Raamsdonk
Teaching method(s)	Lecture, Seminar
Level	500

Course objective
The objective is to obtain a good understanding of modal logic and its use in computer science and artificial intelligence.

Course content
A thorough introduction to modal logics, and its applications in computer science and artificial intelligence. We will select some themes from the book Modal Logics for Open Minds, by Johan van Benthem: basic modal logic and possible world semantics, bisimulation and invariance, modal definability, decidability, ... In particular we treat the modal logics most relevant to computer science and AI: temporal, dynamic and epistemic logic.

Form of tuition
Weekly 2 lectures and 1 exercise class, for the duration of 7 weeks.

Type of assessment
A written exam and assignments that can make half a point bonus.

Course reading
Johan van Benthem, Modal Logics for Open Minds, CSLI Publications 2010.

Recommended background knowledge
The bachelor course Logica en Modelleren (previously Inleiding Logica), or an equivalent introduction to first-order logic.

Target group
mAI, mCS, mPDCS

Advanced Networking

Course code	X_417018 ()
Period	Period 5
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Level	500

Course content
http://coursecatalogue.uva.nl/xmlpages/page/2015-2016-en/search-course/course/18147

Registration procedure
This course is offered at the UvA. For more information contact: FNWI Education Service Centre, Science Park 904, servicedesk-esc-science@uva.nl, +31 (0)20 525 7100.
Enrolment via https://m.sis.uva.nl/vakaanmelden is required.

Advanced Selforganisation

Course code	X_400434 (400434)
Period	Period 2
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Teaching method(s)	Lecture
Level	400

Course objective
To understand, simulate and analyse the behaviour and self-organization of complex systems. The student is able to explain, implement and recognize basic principles and properties of such systems.

Course content
This course is about the understanding of the behavior and self-organization of complex systems: systems in which the interaction of the components is not simply reducible to the properties of the components. The general question the we address is: how should systems of very many independent computational (e.g. robotic or software) agents cooperate in order to process information and achieve their goals, in a way that is efficient, self- optimizing, adaptive, and robust in the face of damage or attack? We will look at natural systems that solve some of the same problems that we want to solve, e.g. adaptive path minimization by ants, wasp and termite nest building, army ant raiding, fish schooling and bird flocking, coordinated cooperation in slime molds, synchronized firefly flashing, evolution by natural selection, game theory and the evolution of cooperation. The course includes a practical part in which students implement a simulation of a self-organizing complex system and conduct structured experimental analysis with this simulation.

Form of tuition
Theory in lectures and practice in labs.

Type of assessment
Report including description of simulation and experimental analysis.

Course reading
Schut M.C., Scientific Handbook for Simulation of Collective Intelligence, 2007. Will be distributed in class.

Target group
mAI, mBA, mCS, mPDCS

Remarks
More information available on BlackBoard. This is a project- oriented course and therefore students will be expected to have basic programming skills.

Advances in Computer Architecture

Course code	X_418047 ()
Period	Period 1

Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Teaching method(s)	Lecture, Computer lab, Seminar
Level	500

Course content
http://studiegids.uva.nl/xmlpages/page/2015-2016/zoek-vak/vak/15582

Target group
mCS

Remarks
This course is offered at the UvA. For more information contact: FNWI Education Service Centre, Science Park 904, servicedesk-esc-science@uva.nl, +31 (0)20 525 7100.
Enrolment via https://m.sis.uva.nl/vakaanmelden is required.

Binary and Malware Analysis

Course code	X_405100 ()
Period	Period 1
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	C. Giuffrida
Examinator	prof. dr. ir. H.J. Bos
Teaching staff	prof. dr. ir. H.J. Bos, C. Giuffrida
Teaching method(s)	Lecture
Level	600

Course objective
Deepening insights in static and dynamic analysis, applied to binaries and malware

Course content
Binaries in general, and malware in particular, are very hard to analyse. Unlike with source code, you have no idea what the binary does, or even what the data structures look like - let alone what they mean!. Security analysts, forensic experts, and reverse engineers often have to dig their way through such programs to figure out what the code is all about, and where the interesting pieces of information are.

How do they do this? What techniques and tools can they fall back on, and, conversely, what techniques do the malware authors use to prevent this?

This is a (tough) hands-on specialisation course for a small group of motivated students, who will learn essential analysis techniques and methods in both static and dynamic analysis. Not only will they pick apart real malware, they will also be working on a set of cool and very complicated challenges to find a secret burried deep inside a binary program.

For static analysis, we will look in depth at the generation of control flow graphs, and complications that may arise due to indirect calls and jumps (as well as deliberate obfuscation). For dynamic analysis, we will look at data and control flow tracking (dynamic information flow tracking)

Binary patching will be used to circumvent the binary's defenses. To do so, students need to know details about popular binary formats (ELF, PE, etc.), and work with all manner of state-of-art system tools to analyse the binaries (think IDA Pro, OllyDbg, taint analysis tools, etc.).

In addition, students will be exposed to programs that actively fight static and dynamic analysis.

Form of tuition
Hoorcollege and practical

Course reading
Slides and online material

Target group
mCS-HPDC, mCS-IWT, mPDCS

Business Process Management

Course code	X_405115 ()
Period	Period 1
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	prof. dr. ir. H.A. Reijers
Examinator	prof. dr. ir. H.A. Reijers
Teaching staff	prof. dr. ir. H.A. Reijers
Teaching method(s)	Lecture, Practical, Seminar
Level	400

Course objective
Business Process Management is a rapidly growing field, both in practice and academia. Evidence from the effectiveness of process-oriented approaches is accumulating. Process-aware technologies are used by organizations in all areas of the world, in all sectors.

As an expert in Business Information Systems, it is inevitable that you will get involved in process improvement projects. In your career, you may find yourself in the role of a professional working in a process that is being analyzed, redesigned, or supported by information technology. Alternatively, you may be managing such a process. Even more likely, you may play the role of intermediary, standing between the operational professionals executing a process and higher management that wishes organizational improvement. The knowledge and especially the skills taught in this course provide you with the basic instruments to carry out and understand BPM projects.

This course also gives a view on the scientific challenges that the BPM

field is concerned with. This may stimulate you to contribute to the solutions for these challenges, for example as a scientist in this area.

After taking this course, the student will be able to:
• explain the organizational merits of process thinking, in particular in contrast to traditional management thinking;
• identify the different phases in the management of business processes;
• model complex business processes with a formal modeling technique, taking (partly) informal requirements into account;
• communicate process designs to both end-users and IT specialists;
• use process design theory to develop alternatives to existing processes;
• analyze the conformance and performance of process designs before they are put into production;
• understand how business processes can be analyzed on the basis of analyzing event logs;
• describe and understand the main features of process-aware information systems (workflow technology).

Course content
As a response to increasing competition and more demanding customers, various researchers, practitioners, and management gurus have suggested companies to put less emphasis on hierarchical and functional structures, but instead focus on and improve entire chains of business operations, ranging often from client to client. The orientation on such business processes to manage and improve organizational effectiveness is at the core of this course.
Within this course, there is an emphasis on the role of models and information technology to manage business processes. This means that there will be a focus on the creation and analysis of design artifacts, in particular process models. Also, the role of IT as an enabling and support technology for process improvement will receive a wide share of attention.
The course on Business Process Management builds on the idea that business processes go through a life-cycle, with different phases:
• Identification: the problem to distinguish which processes in organizations require priority to be actively managed;
• Discovery: the elicitation and specification of the way that operational processes are carried out;
• Analysis: the understanding of a process' structural ability to fulfill the requirements it must meet;
• Redesign: the planned actions to increase the performance and/or conformance of business processes by changing its elements;
• Implementation: the execution of business processes using advanced IT, such as workflow management systems;
• Monitoring/control: the day-to-day monitoring of a business process to detect operational problems and violations of regulations.
The various lectures and instructions will be devoted to these phases.

Form of tuition
Three hours of lectures per week (h) and two hours of work instructions (w).

Type of assessment
Assignments (O) and a closed-book exam (T). The resit is one integrated, closed-book exam (T).

Course reading
Fundamentals of Business Process Management. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A. Springer, 2013. ISBN: 978-3-642-33142-8 (Print) 978-3-642-33143-5 (Online).

Coding and Cryptography

Course code	X_405041 (405041)
Period	Period 1
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	prof. dr. R.M.H. de Jeu
Examinator	prof. dr. R.M.H. de Jeu
Teaching staff	prof. dr. R.M.H. de Jeu
Teaching method(s)	Lecture
Level	500

Course objective
To give an introduction the theory of error correcting codes and to cryptography.

Course content
This course provides a thorough introduction to the theory of error correcting codes, and to cryptography. It is aimed especially at students of Computer Science. For error correcting codes we shall include cyclic codes, BCH codes, Reed-Solomon codes and burst error correction. For cryptography we discuss some modern public key cryptography (e.g., RSA, ElGamal, DSA).

Form of tuition
Lectures and exercise classes

Type of assessment
Written exam and homework. The written exam will count for 80 percent of the grade, the homework will count for 20 percent of the grade. If not both the written exam and the homework are at least 55 percent each, then the maximum score will be 54 percent (which constitutes a fail).

Course reading
We shall be working from "Coding theory and cryptography, the essentials" by Hankerson, Hoffman, Leonard, Lindner, Phelps, Rodger and Wall (second edition, revised and expanded).

Recommended background knowledge
Some knowledge on linear algebra, on the integers modulo n, and on polynomials.

Target group
mAI, mCS, mMath, mPDCS

Computational Complexity

Course code	X_417017 ()

Period	Period 1
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Level	500

Course content
http://studiegids.uva.nl/xmlpages/page/2015-2016/zoek-vak/vak/20246

Remarks
This course is offered at the UvA. For more information contact: FNWI Education Service Centre, Science Park 904, servicedesk-esc-science@uva.nl, +31 (0)20 525 7100.
Enrolment via https://m.sis.uva.nl/vakaanmelden is required.

Computer Networks Practical

Course code	X_405072 (405072)
Period	Period 5+6
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. S. Voulgaris
Examinator	dr. S. Voulgaris
Teaching method(s)	Lecture
Level	500

Course objective
Put concepts of Computer Networks and Operating Systems into practice, in the context of smartphones.

Course content
This is a (tough) lab course, that involves low-level programming on Android smartphones. It requires very thorough understanding of operating systems and network concepts. It is done either individually or in groups of two.

Form of tuition
Practical computer work

Type of assessment
Practical computer work

Recommended background knowledge
Computer Networks (400487)
Operating Systems (400011)
Good knowledge of Java!

Target group
mCS, mPDCS

Concurrency and Multithreading

Course code	X_405064 (405064)
Period	Period 1
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	prof. dr. W.J. Fokkink
Examinator	prof. dr. W.J. Fokkink
Teaching staff	prof. dr. W.J. Fokkink
Teaching method(s)	Lecture, Seminar
Level	400

Course objective
This course provides a comprehensive presentation of the foundations and programming principles for multicore computing devices.

Learning objectives are: fundamental insight into multicore computing; algorithms for multicore computing; analyzing such algorithms; concurrent datastructures; multicore programming.

Course content
Shared memory, mutual exclusion, synchronization operations, concurrent data structures, scheduling, transactional memory, multithreaded programming assignment.

Form of tuition
Lectures: 4 hours per week, exercise classes: 4 hours per week.

Type of assessment
Written exam (which counts for 70% of the final mark) and one programming assignment (which counts for 30% of the final mark).

Course reading
Maurice Herlihy, Nir Shavit, The Art of Multiprocessor Programming, Morgan Kaufmann, 2008.

Target group
mAI, mCS, mPDCS

Remarks
The homepage of the course is at http://www.cs.vu.nl/~tcs/cm/

The lectures and written exam of the BSc and MSc variant of Concurrency and Multithreading coincide. The difference is that the BSc variant has a smaller programming assignment than the MSc variant.

The MSc variant of this course cannot be followed by students that included the BSc variant in their BSc program.

Concurrency Theory

Course code	X_418103 ()
Period	Period 1
Credits	6.0
Language of tuition	English

Faculty	Faculteit der Exacte Wetenschappen
Coordinator	O.W. Schrofer
Examinator	O.W. Schrofer
Teaching staff	dr. A. Ponse
Teaching method(s)	Lecture, Seminar, Computer lab
Level	600

Course content
http://studiegids.uva.nl/xmlpages/page/2015-2016/zoek-vak/vak/16283

Remarks
This course is offered at the UvA. For more information contact: FNWI Education Service Centre, Science Park 904, servicedesk-esc-science@uva.nl, +31 (0)20 525 7100.
Enrolment via https://m.sis.uva.nl/vakaanmelden is required.

Data Mining Techniques

Course code	X_400108 (400108)
Period	Period 5
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. M. Hoogendoorn
Examinator	dr. M. Hoogendoorn
Teaching staff	dr. M. Hoogendoorn
Teaching method(s)	Lecture
Level	500

Course objective
The aim of the course is that students acquire data mining knowledge and skills that they can apply in a business environment. How the aims are to be achieved: Students will acquire knowledge and skills mainly through the following: an overview of the most common data mining algorithms and techniques (in lectures), a survey of typical and interesting data mining applications, and practical assignments to gain "hands on" experience. The application of skills in a business environment will be simulated through various assignments of the course.

Course content
The course will provide a survey of basic data mining techniques and their applications for solving real life problems. After a general introduction to Data Mining we will discuss some "classical" algorithms like Naive Bayes, Decision Trees, Association Rules, etc., and some recently discovered methods such as boosting, Support Vector Machines, and co-learning. A number of successful applications of data mining will also be discussed: marketing, fraud detection, text and Web mining, possibly bioinformatics. In addition to lectures, there will be an extensive practical part, where students will experiment with various data mining algorithms and data sets. The grade for the course will be based on these practical assignments (i.e., there will be no final examination).

Form of tuition
Lectures (h) and compulsory practical work (pra). Lectures are planned to be interactive: there will be small questions, one-minute discussions, etc.

Type of assessment
Practical assignments (i.e. there is no exam). There will be two assignments done in groups of three. There is a possibility to get a grade without doing these assignments: to do a real research project instead (which will most likely to involve more work, but it can also be more rewarding). For the regular assignments the first assignment counts for 40% and the second for 60%. The grade of both assignments needs to be sufficient to pass the course.

Course reading
Ian H. Witten, Eibe Frank, Mark A. Hall, Data Mining: Practical Machine Learning Tools and Techniques (Third Edition). Morgan Kaufmann, January 2011
ISBN 978-0-12-374856-0

Recommended background knowledge
Kansrekening and Statistiek or Algemene Statistiek (knowledge of statistics and probabilities) or equivalent. Recommended: Machine Learning.

Target group
mBA, mCS, mAI, mBio

Developing Services for the Cloud

Course code	X_405074 ()
Period	Period 3
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	prof. dr. P. Lago
Examinator	prof. dr. P. Lago
Teaching staff	prof. dr. P. Lago
Teaching method(s)	Lecture
Level	500

Course objective
Learn how to design and implement software services by using modern technologies and development environments; transforming SoaML models into code, experience implementation challenges of software services; troubleshooting in teams; deployment in the cloud.

Course content
Both service oriented computing and cloud computing are booming areas of rapid development and adoption. This module addresses major related technical aspects including how to transform a service oriented design into service implementations, how to deploy them in a remote cloud environment, how to effectively and critically use development tools. Group work and active participation is an essential part of the classes.

form of tuition: Lectures and group work.

Form of tuition
HC, WC, PR

Type of assessment
Final presentation with demo. Teamwork.

Course reading
Material distributed by lecturers.

Entry requirements
Service Oriented Design (X_405061).

Recommended background knowledge
Knowledge of UML and SoaML; software engineering and Java programming.
Preferred: module Service Oriented Design (405061).

Target group
mCS

Remarks
Registration is compulsory at least 4 weeks before course starts.
Attendance is compulsory (all students bring their laptop!).
The students attending the module will be instructed to prepare required input material before the module starts.

Distributed Algorithms

Course code	X_400211 (400211)
Period	Period 2
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	prof. dr. W.J. Fokkink
Examinator	prof. dr. W.J. Fokkink
Teaching staff	prof. dr. W.J. Fokkink
Teaching method(s)	Lecture, Seminar
Level	500

Course objective
To obtain a good understanding of concurrency concepts and a large range of distributed algorithms.

To offer a bird's-eye view on a wide range of algorithms for basic challenges in distributed systems.

To provide students with an algorithmic frame of mind for solving fundamental problems in distributed computing.

Course content
Snapshots, graph traversal, termination detection, garbage collection, deadlock detection, routing, election, minimal spanning trees, anonymous networks, fault tolerance, failure detection, synchronization, consensus, mutual exclusion, self-stabilization.

Characteristic of the course is that correctness arguments and complexity calculations of distributed algorithms are provided in an intuitive fashion.

Form of tuition
4 hours per week HC
4 hours per week WC

Type of assessment
Written examen (plus a take-home exercise sheet that can provide up to 0.5 bonus point).

Course reading
W.J Fokkink. Distributed Algorithms: An Intuitive Approach. MIT Press, 2013.

Target group
mAI, mCS, mPDCS

Remarks
The homepage of the course is at http://www.cs.vu.nl/~tcs/da/

Distributed Systems

Course code	X_400130 (400130)
Period	Period 2
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. ing. T. Kielmann
Examinator	dr. ing. T. Kielmann
Teaching staff	dr. ing. T. Kielmann
Teaching method(s)	Lecture, Seminar
Level	400

Course objective
After taking this course, students will be able to:

- understand to a large extent the intricacies related to designing and developing a distributed computer system.

- understand the tradeoffs between centralized, distributed, and fully decentralized solutions.

- be capable of successfully studying research papers on (advanced) distributed systems.

Course content
It is difficult to imagine a standalone modern computer system: every such system is one way or the other connected through a communication network with other computer systems. A collection of networked computer systems is generally referred to as a distributed (computer) system. As with any computer system, we expect a distributed system to simply work, and often even behave as if it were a single computer

system. In other words, we would generally like to see all the issues related to the fact that data, processes, and control are actually distributed across a network hidden behind well-defined and properly implemented interfaces. Unfortunately, life is not that easy.

As it turns out, distributed systems time and again exhibit emergent behavior that is difficult to understand by simply looking at individual components. In fact, many aspects of a distributed system cannot even be confined to a few components, as is easily seen by just considering security.

In this course, we pay attention to the pillars on which modern distributed systems are built. Unfortunately, these pillars cannot be viewed independently from each other: each one is equally important for understanding why a distributed system behaves the way it does, and depends on the way that other pillars have been constructed. In this sense, pillars form principles, in turn offering a view that one can take when studying distributed systems. We will consider the following principles:

- architectures
- processes
- communication
- naming
- coordination
- consistency and replication
- fault tolerance

These principles will be discussed in the context of a few simplifying concepts that have been used to master the complexity of developing distributed systems: objects, files, documents, and events.

Form of tuition
The course is taught as a series of lectures, in combination with small exercises.

Type of assessment
Written exam.

Course reading
This year, we will use a reader. Details about its distribution will be announced via blackboard in due time.

Entry requirements
Students should have taken a standard course on computer networks. Experience with (distributed) programming will be helpful.

Target group
mCS, mPDCS, mAI

Evolutionary Computing

Course code	X_400111 (400111)
Period	Period 1
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen

Coordinator	prof. dr. A.E. Eiben
Examinator	prof. dr. A.E. Eiben
Teaching staff	prof. dr. A.E. Eiben, J.V. Heinerman MSc
Teaching method(s)	Lecture
Level	400

Course objective
To learn about computational methods based on Darwinian principles of evolution. To illustrate the usage of such methods as problem solvers and as simulation, respectively modelling tools.To gain hands-on experience in performing experiments.

Course content
The course is treating various algorithms based on the Darwinian evolution theory. Driven by natural selection (survival of the fittest), an evolution process is being emulated and solutions for a given problem are being "bred". During this course all "dialects" within evolutionary computing are treated (genetic algorithms, evolutiestrategieën, evolutionary programming, genetic programming, and classifier systems). Applications in optimisation, constraint handling, machine learning, and robotics are discussed. Specific subjects handled include:
various genetic structures (representations), selection techniques, sexual and asexual variation operators, (self-)adaptivity. Special attention is paid to methodological aspects, such as algorithm design and tuning. If time
permits, subjects in Artificial Life will be handled. Hands-on-experience is gained by a
compulsory programming assignment.

Form of tuition
Oral lectures and compulsory programming assignment. Highly motivated students can replace the programming assignment by a special research track under the personal supervision of the lecturer(s).

Type of assessment
Written exam and pogramming assignment (weighted average).

Course reading
Eiben, A.E., Smith, J.E., Introduction to Evolutionary Computing.
Springer, 2003 ISBN 3-540-40184-9.
Slides available from http://www.cs.vu.nl/~gusz/ecbook/ecbook.html .

Target group
mBA, mAI, mCS, mPDCS

Experimental Design and Data Analysis

Course code	X_405078 ()
Period	Period 5
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. E.N. Belitser

Examinator	dr. E.N. Belitser
Teaching staff	dr. E.N. Belitser
Teaching method(s)	Lecture, Practical
Level	400

Course objective
In this course the student is acquainted with the most common experimental designs and regression models. Furthermore, nonparametric tests and bootstrap methods are discussed. On completion of this course the student should be able to:
- design experiments and analyse the results according to the design,
- analyse data using the common ANOVA designs,
- analyse data using linear regression or a generalized linear regression model,
- perform basic nonparametric tests,
- perform bootstrap and permutation tests.

Course content
Regression models try to explain or predict a dependent variable using measured independent variables. Statistical methods are needed if there is random variation in the dependent variables. We will discuss multiple linear regression, analyses of variance (ANOVA), generalized linear regression models. All methods will be illustrated with practical examples. Especially in the case of ANOVA it is necessary that the study is well designed in order to draw sound conclusions from an experiment or survey. In this course a few well known designs (completely randomized, randomized block etc.) and the associated analyses of variance are discussed. The remainder of the course is be dedicated to non-parametric testing methods and bootstrap methods:
- Wilcoxon test for (one and two samples)
- Kolmogorov-Smirnov test (two samples)
- rank correlation tests
- permutation and bootstrap tests
All analyses are carried out by a computer package, for which we need to know code but no formulas.

Form of tuition
Lectures, computer classes, discussions of the computer assignments.

Type of assessment
Weekly computer assignments and final assignment. The final grade is based on the written reports of all these assignments.

Course reading
- Slides of the lectures,
- R manual,
- assignments.

An introductory book on statistics (containing the prerequisite knowledge for this course) is for example
- Statistical reasoning for everyday life, J.O. Bennett, W. Briggs, M.F. Triola.

For more background on the topics in this course, the following books are recommended:
- Linear models with R, by J.J. Faraway (emphasis on the implementation in R);

- Extending the linear model with R, by J.J. Faraway (emphasis on the implementation in R);
- A first course in the design of experiments; a linear models approach, by D.C. Weber and J.H. Skillings (emphasis on the designs, also implementation in SAS).

Entry requirements
Introductory statistics, e.g. Empirical Methods

Recommended background knowledge
Probability and statistics courses.

Target group
mAI, mCS

Remarks
All assignments are to be solved using the statistical package R (http://www.r-project.org/)

Green Lab

Course code	X_418158 ()
Period	Period 1
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	prof. dr. P. Lago
Examinator	prof. dr. P. Lago
Teaching staff	dr. ing. G. Procaccianti
Teaching method(s)	Lecture, Practical
Level	400

Course objective
Learn the basics of empirical experimentation in the field of Software Engineering.
Be able to operate in a lab environment and build a successful experiment for software energy consumption.
Become familiar with the research problems in the field of green software engineering.
Understand and measure the impact of software over energy consumption.

Course content
Students will work in teams to perform experiments on software energy consumption in a controlled environment. They will have to carry out all the phases of empirical experimentation, from experiment design to operation, data analysis and reporting. They will be provided with examples of previous experiments, but they will have to choose by themselves the experimental subjects and hypotheses to test. During the lab sessions, students will be assisted for technical operation of the lab equipment as regards measurement and data gathering. Students will also receive the required training for data analysis and visualization (i.e. graphs, dashboards) using specialized software.

Form of tuition

Lectures. Lab sessions.

Type of assessment
Teamwork, project assignments.

Course reading
Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., & Wesslén, A. (2012). Experimentation in software engineering. Springer.
Material distributed on Blackboard.

Recommended background knowledge
Basic statistical analysis techniques (descriptive statistics and most common tests).

Target group
mCS, PDCS, mAI

History of digital cultures

Course code	X_418107 ()
Period	Period 3
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	O.W. Schrofer
Examinator	O.W. Schrofer
Level	400

Course content
http://studiegids.uva.nl/xmlpages/page/2015-2016/zoek-vak/vak/21188

Remarks
This course is offered at the UvA. For more information contact: FNWI Education Service Centre, Science Park 904, servicedesk-esc-science@uva.nl, +31 (0)20 525 7100.
Enrolment via https://m.sis.uva.nl/vakaanmelden is required.

ICT4D: Information and communication technology for Development

Course code	X_405101 ()
Period	Period 5
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. K.S. Schlobach
Examinator	dr. K.S. Schlobach
Teaching staff	dr. K.S. Schlobach
Teaching method(s)	Lecture, Seminar
Level	400

Course objective
In the developed world Computers are ubiquitous, and ICT has rapidly grown into a critical asset for economic, technological, scientific and societal progress. The main objectives of this course are:
1) to make the next generation of Computer Scientists aware of:
a) The importance of ICTs for the developing world and the unexpected way developing countries are leapfrogging into the information age
b) The opportunities and challenges that exist for an information scientist in the area of 'development4development'
c) The influence of context in a typical ICT4D project
d) The complexity of deploying an ICT project within a development context, and how to tackle this.
2) to equip the students with some initial project management, technological and programming skills specific to an ICT deployment in a developing country.
Positioned at the heart of the VU's vision of social relevance as one of the guiding principles, the core aim of the course is to raise the awareness that we as Computer Scientists can make a significant difference by sharing our expertise according to well established principles of international development.

Course content
The course will be given jointly by the Department of Computer Science and the Center for International Cooperation, and will consist of 4 modules: two practical ones, and two theoretical ones.
1) Analysing a development problem (CIS): this theoretical module will introduce the analytical methods required for an indepth understanding of a potential development support project. A number of invited speakers will introduce general requirements and strategies, as well as more focused on a particular potential project.
2) Developing a deployment plan (CIS): in this practical module the students will have to produce a specific deployment plan for an ICT project in a developing country.
3) From plan to project (CS): this theoretical module will provide some initial technological knowledge required for running an ICT project in a developing country. It will give an overview over technology already applied, such as specific networks, connection types, hardware as well as specific software environments, but also introduce basic concepts in project management for ICT projects.
4) Turn projects into tools (CS): In this practical module the students will actually build a set of deployment tools according to the conditions specified in their deployment plan, including building the required infrastructure, setting up hardware, writing and installing required software, including appropriate documentation and user guidance.
Depending on current actual collaborations of CIS and the CS department a specific type of deployment will be chosen. Examination will be via 2 projects related to those concrete deployment activities of ICT in the development context

Form of tuition
The course will be a combination of lectures (first 4 weeks) and project work (weeks 5-8).

Course reading
Collection of papers.

Target group
mAI, mCS

Individual Systems Practical

Course code	X_405088 ()
Period	Ac. Year (September)
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. S. Voulgaris
Examinator	dr. S. Voulgaris
Level	500

Course objective
Give very motivated students the opportunity to work on challenging, research-oriented projects, with a strong focus on serious systems work.

Course content
The content is to be negotiated individually with the professor supervising and grading it. Agreeing with a professor at the VU is absolutely necessary *before* the project starts. Showing up in retrospect, trying to earn some credits for work you did last summer or on a side-job, will certainly not qualify.

Only a few students (the ones that come up with a convincing project proposal) will be allowed to do it. Proposals that are not challenging enough, are not deep into systems work, or their nature prevents the clear assessment of the student's own contribution, will be declined.

Type of assessment
Supervised systems work.

Target group
mCS, mPDCS

Industrial Internship

Course code	X_405080 ()
Period	Ac. Year (September)
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. ing. T. Kielmann
Examinator	dr. ing. T. Kielmann
Level	500

Course objective
Deepening insights by applying study contents in an industrial setting

Course content

Individual project work by which the student applies the study contents in an industrial setting. Before the start of the internship, the student has to get approval for the internship project by a VU Computer Science lecturer. The project has to focus on research or development aspects, by which the student can apply and validate the study contents within the specific constraints of an industrial setting. At the end of the internship, the student submits a written report to the lecturer, in which the work, the lessons learned, and the insights from applying study contents in an industrial setting are described.

For the grading of the report, most important are the student's reflections
on study contents vs. "industrial reality": What did you learn during your
studies that was particularly helpful for your internship? What is different
in an industrial environment, compared to university? What did you learn during your internship that you were not told at university?

Form of tuition
individual project work in an industrial setting

Type of assessment
written report

Recommended background knowledge
The student should have completed at least 48 credits of his or her Master programme such that there are sufficient study contents to be applied in an industrial setting.

Target group
mCS, mPDCS

Remarks
Various lecturers

Internet programming

Course code	X_405082 ()
Period	Period 1
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. S. Voulgaris
Examinator	dr. S. Voulgaris
Teaching staff	dr. S. Voulgaris
Teaching method(s)	Lecture
Level	500

Course objective
Guide the student through the design and development of Network and Web applications.

Course content

The course discusses the principles for understanding, designing, and developing Internet applications. This includes programming the network (sockets, threads, RPC, RMI), programming the web interface (servlets, PHP, Javascript, AJAX), and setting up secure communication channels. Throughout the course, as well as in the context of the lab assignments, attention is paid to practical issues of applying these concepts.

Form of tuition
Lectures combined with lab assignments

Type of assessment
Final exam plus lab assignments

Course reading
Course slides

Entry requirements
Knowledge of C, Java

Recommended background knowledge
Good knowledge of both C and Java

Target group
mAI, mCS, mPDCS

Introduction to Computational Science

Course code	X_418111 ()
Period	Period 1
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Level	400

Course content
http://studiegids.uva.nl/xmlpages/page/2015-2016/zoek-vak/vak/18229

Remarks
This course is offered at the UvA. For more information contact: FNWI Education Service Centre, Science Park 904, servicedesk-esc-science@uva.nl, +31 (0)20 525 7100.
Enrolment via https://m.sis.uva.nl/vakaanmelden is required.

Knowledge and Media

Course code	X_405065 (405065)
Period	Period 1
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. T. Kuhn MSc
Examinator	dr. T. Kuhn MSc
Teaching staff	prof. dr. A.T. Schreiber, dr. T. Kuhn MSc

Teaching method(s)	Seminar
Level	500

Course objective
The goal of the course is to provide insight in the concepts of information organization,
knowledge, ontologies and knowledge processes in relation to various ICT-based media.

Course content
This course treats the principles and theories that form the foundation of information organization and knowledge-intensive processes in relation to various multi-media
applications. Knowledge processes are those processes that use knowledge (reasoning), document knowledge (representation), acquire knowledge or transfer knowledge (teaching). The relation between knowledge processes and (interactive) media will be explored. Various types of applications will be discussed, such as special purpose search engines, educational systems, serious gaming and mind tools.

Form of tuition
Working lectures

Type of assessment
Portfolio

Course reading
Articles distributed through Blackboard

We will use The Discipline of Organizing Edited by Robert J. Glushko as a text.

Target group
UvA students and optional course for mCS, mAI and mIS

Knowledge Engineering

Course code	X_405099 ()
Period	Period 2+3
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. A.C.M. ten Teije
Examinator	dr. A.C.M. ten Teije
Teaching staff	dr. A.C.M. ten Teije
Teaching method(s)	Lecture
Level	400

Course objective
goals:
1) to be able to elicitate knowledge from experts by using several elicitation techniques
2) to be able to build all CommonKads models that play a role in the development of a knowledge based system, this includes the context of

the KBS and the expertise model based
3) to be able to implement the expertise model as a prototype
4) to be able to reflect on your own process of modelling and building a knowledge based system, and to reflect on your product (=which are the models and the implementation)

Course content
Knowledge Engineering is a discipline that involves integrating knowledge into a program for solving a complex problem, which requires human expertise. Typical tasks are classification, diagnosis, planning etc. In the course we use CommonKADS as the methodology for the process of modeling the organisation, the context and the knowledge intensive tasks.
This methodology give clear guidelines and concrete templates for modeling the organisational aspects and the expertise model, which is the core model of knowledge based system. The notion of pattern-based knowledge modeling is a key issue in the knowledge modelling process.
The goal of the final project is to perform the entire knowledge technology process for a knowledge intensive problem of your own choosing, starting with context analysis, up to a (partial) implementation of the knowledge based system.

Form of tuition
Lectures, assignments, group project

Type of assessment
Assignment, project reports.

Course reading
Schreiber, Akkermans, Anjewierden, de Hoog, Shadbolt, van de Velde, Wielinga: Knowledge Engineering & Management. The MIT Press, Cambridge MA, 2000, ISBN 0-262-19300-0.

Target group
mAI, mIS, mCS-TAI

Lambda Calculus

Course code	X_418108 ()
Period	Period 2
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen

Course content
http://studiegids.uva.nl/xmlpages/page/2015-2016/zoek-vak/vak/21590

Remarks
This course is offered at the UvA. For more information contact: FNWI Education Service Centre, Science Park 904, servicedesk-esc-science@uva.nl, +31 (0)20 525 7100.
Enrolment via https://m.sis.uva.nl/vakaanmelden is required.

Large Scale Data Engineering

Course code	X_405116 ()

Period	Period 4
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	prof. dr. P.A. Boncz
Examinator	prof. dr. P.A. Boncz
Teaching staff	prof. dr. P.A. Boncz
Teaching method(s)	Lecture, Seminar
Level	500

Course objective
The goal of the course is to gain insight into and experience with algorithms and infrastructures for managing big data.

Course content
This course confronts the students with some data management tasks, where the challenge is that the mere size of this data causes naive solutions, and/or solutions that work only on a single machine, to stop being practical. Solving such tasks requires the computer scientist to have insight in the main factors that underlie algorithm performance (access pattern, hardware latency/bandwidth), as well as possess certain skills and experience in managing large-scale computing infrastructure.

Form of tuition
There are two lectures per week, and requires significant practical work. The practicals are done outside lecture hours, at the discretion of the students who are supported remotely through Skype screen sharing.

Type of assessment
In the first assignment the students can work either on their own laptops via a prepared VM, or in the cloud using an Amazon EC2 Micro Instance; and there is an online competition between practicum teams for the best result. The second assignment, using a Hadoop Cluster, are done on the SurfSARA Hadoop cluster (90 machines, 720 cores, 1.2PB storage). For this assignment, a report of 5-8 pages must be written. The students also need to read two scientific papers of choice, related to the second assignment, and present these in class. There is no written examn; the grade is based on the two assignments grades, the grade for the in-class presentation and attendance/participation.

Course reading
scientific papers provided in the course

Entry requirements
Hadoop environments are consist of Linux machines, so some basic ability in working with these comes in handy. Also, you must have some programming skills in C,C++ or Java.

Recommended background knowledge
Programming proficiency in C/C++ or Java

Target group
mCS, mPDCS

Large-Scale Computing Infrastructures

Course code	X_405106 ()
Period	Period 5
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. ing. T. Kielmann
Examinator	dr. ing. T. Kielmann
Teaching staff	dr. ing. T. Kielmann
Teaching method(s)	Lecture
Level	500

Course objective
Students explore the field of large-scale computing infrastructures. They study its technological frontiers from scientific publications and get hands-on experience via programming assignments.

Course content
Cloud infrastructures like Amazon's EC2 or Microsoft's Azure provide seemingly limitless compute and storage capacity. The technology underlying these systems strongly relies on decades of work on high-performance, distributed computing platforms, such as cluster computing, computing grids, and supercomputers. We study aspects of computing in large scale, such as resource management and scheduling, remote data access, energy efficiency, failure resilience, performance of large systems, as well as suitable software architecture and programming models such as MapReduce.

Form of tuition
Introductory lectures, followed by a seminar part and practical programming assignments. In the seminar part, students explore the technological frontiers of large-scale computing, published in top-quality scientific venues of the field. Students present their findings and write position papers about topics presented by other students in the class. With the practical programming assignments, students get hands-on experience with large-scale computing infrastructures.

Type of assessment
Both parts contribute 50% each to the grade:

- seminar presentation and position paper

- programming assignments

Course reading
Various scientific articles as available online

Recommended background knowledge
Students should have basic knowledge about distributed systems and parallel application programming.
Students must be able to program in Java and Python (or be able to get the needed skills on the fly).

Target group
mPDCS, mCS

Literature Study and Seminar

Course code	X_405111 ()
Period	Ac. Year (September)
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	prof. dr. P. Lago
Examinator	prof. dr. P. Lago
Teaching method(s)	Lecture
Level	500

Course objective
After taking this course, students will be able to:
- design a sound desk research based on scientific literature and identify the relevant literature related to the target research; or perform an in-depth study of a selected set of literature sources;
- criticize, analyze, and discuss scientific literature;
- reflect on the in-depth knowledge gained during the course on the selected literature study topic.

Course content
The course consists of carrying out a literature study on a topic chosen in agreement with a selected tutor.

The ‘theoretical preparation’ of the course consists of studying the provided material on literature study design, where the student learns how to go from a research question to a well-structured analysis of the literature. This step can be carried out by translating the research question into a sound search query, identifying adequate on-line literature search engines, and performing a motivated selection of the literature for further analysis; or alternatively by selecting, together with the tutor, a set of relevant literature and performing an in-depth study.

The actual literature study starts with the 'exploration phase' in which the student must identify a topic of interest or a course particularly appreciated. He or she will then contact the person in charge of the identified research area/course and discuss with him/her the possibility to carry out a literature study under his/her supervision.

In general the literature study will be either predominantly broad, leading to a literature survey on the selected theme, or deep, carving out the intricacies of a specific topic. The outcome of the literature study is a final report, which must include: study design; overview of selected literature; analysis of the literature; discussion and conclusions. At the end of this phase, the student gives one final presentation to the research group of the tutor. This presentation includes research questions, study design, study execution, and discussion of analysis.

Form of tuition
Individual assignment.

Type of assessment
Final report. Final presentation.

Grading criteria: quality of study design; rationale for literature selection (if applicable); quality of results (quality of writing, scientific quality of the analysis, discussion of the findings, reflection in the drawn conclusions, clear answer to main research question); quality of technical report (style, clarity, organization); correctness and completeness of references and citations; final presentation to the research group where he/she carries out the literature study.

Course reading
Material on literature study design (distributed by the coordinator).

Target group
mCS

Logical Verification

Course code	X_400115 (400115)
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. F. van Raamsdonk
Examinator	dr. F. van Raamsdonk
Teaching staff	dr. F. van Raamsdonk
Teaching method(s)	Lecture, Practical
Level	500

Course objective
Introduction to the proof assistant Coq and its type-theoretic foundations.

Course content
A proof-assistant is used to check the correctness of a specification of a program or the proof of a theorem. The course is concerned with the proof-assistant Coq which is based on typed lambda-calculus. In the practical work, we learn to use Coq. One of the exercises is concerned with the correctness proof of the specification of a sorting algorithm, from which a functional program is extracted. In the course, we focus on the Curry-Howard-De Bruijn isomorphism between proofs on the one hand and lambda-terms (which can be seen as functional programs) on the other hand. This is the basis of proof-assistants like Coq. We study various typed lambda calculi and the corresponding logics.

Form of tuition
2 times 2 hours theory class, 2 times 2 hours practical work

Type of assessment
Written exam, obligatory Coq-exercises, obligatory hand-in theory exercises.

Course reading
Course notes

Entry requirements
An introduction course in logic.

Target group
mCS, mAI, mMath, mPDCS

Remarks
The course is taught once every two years, the next opportunity will be in study year 2016-2017

Master Project

Course code	X_400442 (400442)
Period	Ac. Year (September)
Credits	36.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. ing. T. Kielmann
Level	500

Course objective
The Master Project is the place in the study where scientific and professional skills are trained most extensively. The Master Project will always involve an element of originality or creativity, for example in performing a design task or in contributing to the solution or the analysis of a scientific problem. Other important elements of the Master Project are the cooperation with professionals and possibly with other students, planning the project, and documenting and presenting the final results.

Course content
The Master project concludes the Master programme. It is either in the form of a graduation project in one of the research groups of the Department of Computer Science of VU University or the Informatics Institute of the University of Amsterdam, or as an internship in a company. In most cases, it will be performed as an individual project but it can be a group project as well.

Form of tuition
The Master Project has always to be supervised by a staff member of the Department of Computer Science of VU University or the Informatics Institute of the University of Amsterdam. In the case of an internship, supervision will be in cooperation with a supervisor at the company. Internships proposed by the student him- or herself need advance approval from a member of staff, who will become the project supervisor.

Type of assessment
The final grade will be based on the quality of the research, the written thesis and an oral presentation. In the case of a group project, students have to write individual theses and give individual oral presentations, demonstrating their individual proficiency.

Entry requirements
The student must have completed (almost) all other courses of the Master programme.

Target group
mCS

Remarks
For additional information and procedural rules we refer to the website of Faculty of Sciences of VU University.
There, you will also find links to the web pages of the research groups of the Department of Computer Science (VU) and the Informatics Institute (UvA), with options for master projects.
For company internships, useful documentation can be found on the website of the Internship Office.

Model-based Intelligent Environments

Course code	X_405056 (405056)
Period	Period 1
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. M.C.A. Klein
Examinator	dr. M.C.A. Klein
Teaching staff	prof. dr. J. Treur, dr. M.C.A. Klein, dr. T. Bosse
Teaching method(s)	Lecture
Level	500

Course objective
The student will understand different ways in which computerized models can be used in intelligent support systems, and will develop a prototype of such a system based on approaches described in the literature.

Course content
During their bachelor and first year of the master, students have learned to model human processes using different techniques and at different levels of abstraction. In addition, they have learned to use such models for analysis of situations and reasoning about effective support. In this
course, the modeling knowledge will be further deepened and applied to a specific domain or scenario. Scientific literature and applications of model-based reasoning will be studied. The student will develop a prototype of an application based on models relevant for a scenario chosen by the student. By building this prototype, the student shows that he/she masters the modeling approaches and is able to apply this in a specific domain or scenario.

Form of tuition
Lectures and project.

Type of assessment
Assignments.

Course reading
Papers

Recommended background knowledge
Introduction to Modeling and Simulation, Integrative Modeling

Neural Networks

Course code	X_400132 (400132)
Period	Period 1
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. M. Hoogendoorn
Examinator	dr. M. Hoogendoorn
Teaching staff	dr. M. Hoogendoorn
Teaching method(s)	Lecture, Practical
Level	500

Course objective
The course provides an introduction to key concepts and algorithms for pattern recognition and neural networks. It strives towards providing insight both from a theoretical perspective as well as more practical settings. In the end, the student should be able to confidently apply the aforementioned techniques in real-life settings and understand their theoretical basis.

Course content
The course provides an introduction to key concepts and algorithms for pattern recognition and neural networks. It covers the following topics:

- classification, regression, and clustering problems,
- elements of statistical pattern recognition,
- methods for estimation of probability distributions,
- linear classifiers, including Support Vector Machines,
- single-layer and multi-layer networks,
- RBF-networks and kernel methods
- methods for dimensionality reduction
- methods for feature extraction and selection

Moreover, several real-life applications of pattern recognition, including recognition of speech, handwritten characters, images, etc., will be discussed in depth.

Form of tuition
Lectures (h) and practical (pra).

Type of assessment
Practical assignments and written examination. Both count for 50% of the final grade and both grades should be sufficient in order to pass the course.

Course reading
Simon Haykin, Neural Networks and Learning Machines, Pearson Education, 3rd international edition, 2008

Target group
mAI mBio, mBA, mCS

Remarks
More information will be available via Blackboard.

Operating Systems Practical

Course code	X_405071 (405071)
Period	Ac. Year (September)
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	ir. M.P.H. Huntjens
Examinator	ir. M.P.H. Huntjens
Teaching staff	ir. M.P.H. Huntjens
Teaching method(s)	Lecture
Level	500

Course objective
Gain practical experience with the contents of the Computer Networks and Operating Systems courses.

Course content
Three problems covering (as much as possible) the content of the courses.

Form of tuition
Practical computer work

Type of assessment
Practical computer work.

Target group
mCS, mPDCS

Remarks
The course is not taught in 2014-2015, the next opportunity will be probably in 2015-2016

Parallel Programming for High-performance Applications

Course code	X_400161 (400161)
Period	Period 1
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	prof. dr. ir. H.E. Bal
Examinator	prof. dr. ir. H.E. Bal
Teaching staff	prof. dr. ir. H.E. Bal
Teaching method(s)	Lecture

Level	400

Course objective
You will learn
- the backgrounds of High Performance Computing (HPC)
- apply design methods for parallel algorithms
- compare different parallel computer architectures
- analyze performance of network topologies
- compare different parallel programming constructs and programming environments
- get insight in some selected parallel applications
- understand many-cores (e.g. GPUs), many-core algorithms, many-core optimizations

Course content
This lecture discusses how programs can be written that run in parallel on a large number of processors, with the goal of reducing execution time. The class has a brief introduction into parallel computing systems (architectures). The focus of the class, however, is on programming methods, languages, and applications. Both traditional techniques (like message passing) and more advanced techniques (like parallel object-oriented languages) will be discussed. Several parallel applications are discussed, including N-body simulations and search algorithms. About 4 lectures are devoted to an important new development: programming many-core machines such as Graphical Processing Units (GPUs). The class fits well with existing research projects within the department of Computing Systems. It is a good basis for M.Sc. projects in the area of parallel programming, which use the parallel computing systems of the department.

Form of tuition
Lectures (4 hours per week), given by prof.dr.ir. Henri Bal (VU) and dr. Ana Varbanescu (UvA). There is a separate Parallel Programming Practicum (6 ECTS).

Type of assessment
Written exam.

Course reading
To be announced.

Target group
mAI, mBIO, mCS, mPDCS, m Computational Science

Parallel Programming Practical

Course code	X_400162 (400162)
Period	Period 2+3
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	prof. dr. ir. H.E. Bal
Examinator	prof. dr. ir. H.E. Bal
Teaching staff	prof. dr. ir. H.E. Bal
Teaching method(s)	Lecture
Level	500

Course content
With this practicum, several parallel programs have to be written, using different programming environments, including Java, MPI, and CUDA/OpenCL (for GPUs). The
programs must be tested on a parallel machine of the faculty (see http://www.cs.vu.nl/das4) and the performance (speedups) of the programs must be measured, analyzed, and, whenever necessary, optimized. A brief report must be written that explains the approach and discusses the measurements.

Form of tuition
Practical computer work.

Type of assessment
Practical computer work.

Entry requirements
Knowledge of parallel programming in Java, MPI, and CUDA/OpenCL (as taught in the Parallel Programming course) is required.

Target group
Masters Computer Science, PDCS, AI, and Computational Science

Remarks
Students can do this course either in Period 2 or in Period 3.
It is not possible to submit assignments in both periods.

Lecturers:
prof. dr. ir. H.E. Bal
Dr. C. Grelck

Performance of Networked Systems

Course code	X_405105 ()
Period	Period 4
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	prof. dr. R.D. van der Mei
Examinator	prof. dr. R.D. van der Mei
Teaching staff	dr. ing. T. Kielmann, prof. dr. R.D. van der Mei
Teaching method(s)	Lecture
Level	400

Course objective
Students will acquire basic knowledge of:
• performance aspects of networked systems, consisting of servers, services, and clients
• performance engineering principles and methods,
• quantitative models for predicting and optimizing the performance

of networked systems,
• quantitative models for planning capacity of networked systems.
Students will gain experience in engineering and planning performance of networked systems, and will learn how to tackle practical performance problems arising in the ICT industry.

Course content
Over the past few decades, information and communication technology (ICT) has become ubiquitous and globally interconnected. As a consequence, our information and communication systems are expected to process huge amounts of (digital) information, which puts a tremendous burden on our ICT infrastructure. At the same time, our modern society has become largely dependent on the well-functioning of our ICT systems; large-scale system failures and perceivable Quality of Service (QoS) degradation may completely disrupt our daily lives and have huge impact on our economy.

Motivated by this, the course will focus on performance-related issues of networked systems. In the first part, we study capacity planning and modeling for server systems and networks. In the second part, we study the client side of performance while focusing on web applications for both desktop and mobile devices. We address questions like:

- How can we design and engineer networked systems for performance?
- How can we plan server capacity in networked systems?
- How can web applications improve performance across wired and wireless networks?

Form of tuition
Classroom lectures and practical homework assignments.

Type of assessment
The assessment will be based on both homework assignments and a written exam.

Course reading
Textbook, supplemented with a reader on Stochastic Performance Modelling.

High Performance Browser Networking, Ilya Grigorik, O-Reilly, 2013.

Entry requirements
The students should have basic knowledge of computer networks.

Target group
mBA, mCS, mPDCS, mEct

Programming Concurrent Systems

Course code	X_418109 ()
Period	Period 2
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen

Course content
http://studiegids.uva.nl/xmlpages/page/2015-2016/zoek-vak/vak/18739

Remarks
This course is offered at the UvA. For more information contact: FNWI Education Service Centre, Science Park 904, servicedesk-esc-science@uva.nl, +31 (0)20 525 7100.
Enrolment via https://m.sis.uva.nl/vakaanmelden is required.

Protocol Validation

Course code	X_400117 (400117)
Period	Period 5
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	prof. dr. W.J. Fokkink
Examinator	prof. dr. W.J. Fokkink
Teaching method(s)	Lecture, Practical, Seminar
Level	500

Course objective
Learning to use formal techniques for specification and validation of communication protocols.

Course content
This course is concerned with the specification and validation of protocols, using formal methods. The course is based on a specification language based on process algebra combined with abstract data types, called mCRL. This language and its toolset can be used for the specification of parallel, communicating processes with data. Model checking is a method for expressing properties of concurrent finite-state systems, which can be checked automatically. Interesting properties of a specification are: "something bad will never happen" (safety), and "something good will eventually happen" (liveness). In the lab we will teach the use of a tool for automated verification of the required properties of a specification.

Form of tuition
4 hours per week HC
4 hours per week WC/PR (mixed)

During the practicum the mCRL tool and the CADP model checker will be used for the validation of protocols discussed during lectures.

Type of assessment
Written exam, together with a practical homework assignment. The overall mark of the course is (H+2W)/3, where H is the mark for the homework assignment, and W is the mark for the written exam.

Course reading
Wan Fokkink, Modelling Distributed Systems, Springer 2007.

Recommended background knowledge

Logica en Modelleren

Target group
mAI, mCS, mPDCS

Remarks
The course is taught once every two years, the next opportunity will be in study year 2015-2016

Recursion Theory

Course code	X_400534 (400534)
Period	Period 1
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Level	400

Course content
The course description is available on:
http://studiegids.uva.nl/xmlpages/page/2015-2016/zoek-vak/vak/2050

Remarks
This course is offered at the UvA. For more information contact: FNWI Education Service Centre, Science Park 904, servicedesk-esc-science@uva.nl, +31 (0)20 525 7100.
Enrolment via https://m.sis.uva.nl/vakaanmelden is required.

Serious Games

Course code	X_405097 ()
Period	Period 5
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	prof. dr. A.P.W. Eliens
Examinator	prof. dr. A.P.W. Eliens
Teaching staff	prof. dr. A.P.W. Eliens
Teaching method(s)	Lecture
Level	400

Course objective
Serious games are more and more considered to be an effective means to bring about awareness, acquire skills, change behavior, and influence social patterns. With elementary game development technology, the students will explore the potential of serious games in a social context, using casual game mechanics, and what recently has been identified as the dynamics of gamification.

Course content
The course will cover the following topics:
* an introduction to game design

* practical skills in game development
* game mechanics and scoring mechanisms
* elementary game and utility theory
* media & communication theory
* game interaction patterns
* practical applications of serious games
Students are required to work in teams of 2-4 people, with as a goal the actual development of a serious game, with social network support.

Form of tuition
lectures and practicum

Type of assessment
essay and practicum assignment(s)

Course reading
online reference material(s)

Recommended background knowledge
preferably, but not obligatory, project interactive multimedia and multimedia authoring

Target group
choice for master students CS, IS, and others, with an interest in multimedia and game development

Remarks
For information and registration, see: www.cs.vu.nl/~eliens/serious

Service Oriented Design

Course code	X_405061 (405061)
Period	Period 1
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	prof. dr. P. Lago
Examinator	prof. dr. P. Lago
Teaching staff	prof. dr. P. Lago
Teaching method(s)	Lecture, Seminar
Level	400

Course objective
Learn advanced design techniques applicable to large service-oriented software systems. Be able to select among them and apply them for a specific system. Be able to reason about and assess the design decisions.

Course content
The lectures explain the concepts related to the Service Orientation software paradigm and Service Oriented Architecture (SOA).
The lectures provide the students with knowledge about how to identify the requirements for a service-oriented software system, how to map them on business services and transform them into complex networks of software services. Special emphasis is given to the design reasoning

techniques for crucial decision making, service identification, SOA design and migration. Each year experts from academia and industry are invited to give guest lectures.

The students participate in small teams to piecemeal develop understanding of various service-oriented aspects, and work on an assigned SOA design project.

Form of tuition
Lectures and group work.

Type of assessment
Written reports of the assignments. Teamwork.

Course reading
Material handed out by the lecturer and on Blackboard.

Recommended background knowledge
Software modeling experience (knowledge of UML and SoaML preferred). Programming.

Target group
mAI, mCS, mIS

Remarks
Registration for this course is compulsory four weeks prior to the start. Further information on this module will be made available on the Blackboard system http://bb.vu.nl.

Software Architecture

Course code	X_400170 (400170)
Period	Period 2
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	prof. dr. P. Lago
Examinator	prof. dr. P. Lago
Teaching method(s)	Lecture
Level	400

Course objective
Get acquainted with the field of software and information architecture. Understand the drivers behind architectural decisions. Be able to develop and reason about an architecture of a non-trivial software system.

Course content
Students work in groups to develop an architecture for a fictitious system. They have to develop different representations (called views) of the architecture. These different representations emphasize different concerns of people that have a stake in the system. Each group will also be asked to assess ("test") the architecture of another group for certain quality attributes.

Form of tuition
Group work with a number of assignments.

Type of assessment
Project work. Written exam

Course reading
Len Bass et al, Software Architecture in Practice, 3rd Edition, 2012

Target group
mCS, mIS

Registration procedure
Registration is compulsory at least 4 weeks before course starts.

Software Asset Management

Course code	X_400412 (400412)
Period	Period 1, Period 5
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	prof. dr. C. Verhoef
Examinator	prof. dr. C. Verhoef
Teaching staff	prof. dr. C. Verhoef
Teaching method(s)	Lecture
Level	500

Course objective
Gain insight in IT costs, benefits, risks, and returns on information technology.

Course content
In this class we treat several techniques to aid in software cost estimation. We discuss how IT migrates from cost issue to strategic asset, and how to come to grips with this important production factor. We provide insight in how to support decision making on IT investment issues. Examples from the Financial and Insurance industry, and independent software vendors are discussed. With IT benchmarks we obtain insight in the risks of IT developmtn and the operational costs of IT. We introduce the notion of an IT portfolio, and how to perform quantitative analyses with it, to aid in justifying IT investments.

Form of tuition
Seminar with presentations of staff and students.

Type of assessment
Essay on selected topics from articles. You can do this individually or in a group of 3 persons max.

Course reading
Articles and chapters from books.

Recommended background knowledge
Proficiency in software engineering and statistics.

Target group
mCS, mIS, mBA, mAI

Software Metrics

Course code	X_405121 ()
Period	Period 4
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	O.N. Condori-Fernandez
Examinator	dr. O.N. Condori Fernandez
Teaching method(s)	Lecture, Seminar, Computer lab,
Level	500

Course objective
At the end of the course, students will be able to:
- Identify and explain the main elements of a viewpoint-based framework for software measurement.
- Understand the metrology concepts from science and engineering, using them as criteria to:
-- analyze strengths and weaknesses of some of the software metrics most often quoted for software systems.
-- design new software metrics for software systems.
- Validate the software metrics
- Synthesize and critically discuss research papers on software metrics

Course content
A multitude of software metrics have been proposed by researchers over the past decades. Many of them do, however, present design limitations that hinder their applicability in practice. How can we recognize which metrics are sound, and useful? What fundamental concepts and principles must be considered to build sound new software metrics?

This course introduces a general framework for software measurement. It shows how measurement can be used to understand, analyze, control, predict, and improve important properties of software products, services, processes, and projects. The basics on software metrology and software measurement theory are presented. The design of some of the most popular software metrics is analyzed and discussed in terms of weakness and strengths. Experts from green IT industry are also invited to give guest lectures.

Students work in groups to develop a catalogue of sound and useful software metrics for assessing selected software quality properties. They also apply selected energy efficiency metrics in a specific measurement context.

Form of tuition
The course will be taught as a series of lectures, seminar with presentations, and group-work with supervised assignments.

Type of assessment
Written exam, presentation of final project (student groups).

Course reading
- Scientific articles (provided by the teacher).
- Alain Abran, Software Metrics and Software Metrology. IEEE Computer Society and John Wiley & Sons. ISBN 9780470597200
- Norman Fenton, James Bieman. Software Metrics: A Rigorous and Practical Approach, Third Edition by CRC Press. ISBN 9781439838228
- Scott A. Whitmire, Object oriented design measurement. John Wiley & Sons. ISBN 0471134171

Entry requirements
Software Modeling (X_401016) or equivalent software engineering course.

Recommended background knowledge
Proficiency in discrete mathematics and empirical methods

Target group
mCS, PDCS

Remarks
Further information on this module will be made available on the Blackboard system http://bb.vu.nl.

Software Testing

Course code	X_400439 (400439)
Period	Period 5
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. N. Silvis-Cividjian
Examinator	dr. N. Silvis-Cividjian
Teaching staff	dr. N. Silvis-Cividjian
Teaching method(s)	Lecture, Practical
Level	400

Course objective
- Familiarization with basic terminology in software testing.
- Familiarization with techniques and tools used for test generation, execution and adequacy measument.
- Familiarization with software testing literature in a specific area by independent reading of selected research publications.

Course content
Testing is a method to improve software quality. Realistically, software testing is a trade-off between budget, time and quality. It is impossible to test everything so choices have to be made. Students learn how to make these choices and systematically test a software product based only on its requirements or when the code is also available.
This course provides an introduction to software testing with an emphasis on technical activities like test generation, selection,

execution and assessment. The course tries to answer a few questions like: How to design test cases? When to stop testing? What to test when a new version of the product is ready? How to test a safety critical software? How to predict how many faults are in a program? During their practical assignments the students have to test small and large object-oriented software using the techniques learned in class and a set of testing tools.
A few guest lectures showing examples of testing in industry are also planned.

Topics: boundary value analysis, equivalence partitioning, model based test generation, control-flow testing, data-flow testing, mutation testing, regression testing, inspections, automated testing.

Form of tuition
Lectures and compulsory homework assignments.

Type of assessment
Practical assignments and written exam.

Course reading
A. Mathur, Foundations of software testing, Pearson Education, Addison-Wesley, 2008, *ISBN: * 978-8131716601

Recommended background knowledge
Programming skills in Java

Target group
mCS, mAI

Remarks
All material is available in Blackboard.

Software Testing Practical

Course code	X_405124 ()
Period	Period 6
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. N. Silvis-Cividjian
Examinator	dr. N. Silvis-Cividjian
Teaching method(s)	Lecture
Level	400

Course objective
After completing this practical, the students will be able to:
• Develop and test software for different embedded systems, and produce artifacts such as: requirements specifications, safety and risk analysis, design documents, behavioral models, code, test plans and test reports.
• Monitor and analyze their development and testing process in order to asses or compare different software testing techniques

Course content
During the Software Testing course in period 5, students have been exposed to a wide range of traditional and novel testing techniques. During their practical homework assignments, they could apply some of these techniques on small software artifacts. However, there was not enough opportunity to exercise all these techniques in a realistic, industrial-like setting. Moreover, since there is no one silver-bullet, good-for-all cases testing technique, the students never investigated which technique or approach is better for a certain type of software-under-test (SUT). Therefore, it seems very useful to create in period 6 a "playground" where students can experiment with different testing techniques they learned about during the Software Testing course.
The purpose of this practical is to involve students in a realistic software development and testing process, that starts from requirements specification and ends up with high quality code. The SUT benchmark consists of software for embedded systems, such as: software for a heart monitoring (ECG) microcontroller, software agents for Lego Mindstorms NXT robotic kits, and smartphone apps.
The students will work in groups of 3-4. Each group will develop and test one of these SUTs, while continuously gathering test progress information. By analyzing these data, they will end up by drawing conclusions on the strengths and weaknesses of different testing techniques. Examples of possible experiments are: test-driven-development (TDD) vs. waterfall development model, formal specifications vs. natural language specifications, random testing vs. systematic testing, the strengths and weaknesses of model-based testing, combinatorial testing, static analysis, defect prediction, etc. Each group will write a report and will present their findings in class.
Testing expertise from software industry will be available to steer the student groups, and thus bring the academic setup closer to a realistic, industrial setting.

Form of tuition
Practical lab sessions, regular meetings with a steering group

Type of assessment
Written report and presentation

Entry requirements
A Software Testing course. Programming skills in Java, Matlab, C or Python. Familiarization with various testing tools.

Target group
mCS

Remarks
All material is available in Blackboard. The number of participants is limited to 24.

Systems Security

Course code	X_405108 ()
Period	Period 4
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen

Coordinator	prof. dr. ir. H.J. Bos
Examinator	prof. dr. ir. H.J. Bos
Teaching staff	prof. dr. ir. H.J. Bos
Teaching method(s)	Lecture, Seminar
Level	500

Course objective
Very tough course on security with a focus on systems work. At the end of the course students will deeply understand the basic notion of memory corruption attacks (buffer overflows, format strings, etc), network attacks, SQL injection, cross-site scripting attacks, and other vectors used by computer hackers. Besides basic attack, students will learn about state-of-the-art exploitation methods. The course is very(!) hands-on.

Course content
The course covers a wide spectrum of security issues. We explicitly focus on systems security rather than (say) cryptography, as we want to show students how attackers penetrate systems.

Specifically, the course focuses on (1) network security (sniffing, spoofing, hijacking, exploiting network protocols, DDoS, DNS attacks, etc.), (2) memory corruption and application security (buffer overflows, format string bugs, dangling pointers, shellcode, return oriented programming, ASLR/DEP/canaries, control flow integrity and cool new ways of exploitation), (3) web security (XSS, SQL injection, CSRF, http cache poisoning, SOP, authentication, etc.), (4) botnets (centralised/P2P, fast flux, double flux), (4) crypto (basics, systems aspects).

Much of the course will be hands-on and challenge-based. In assignments, student will carry out and investigate attacks in a controlled environment. This involves programming at the both the highest and lowest levels (say SQL and assembly).

Form of tuition
Lectures and (very challenging) practical assignments.

Type of assessment
Written exam (30%) and practical assignments (70%).

Course reading
No set book. All material will be made available during the course.

Entry requirements
Knowledge of C is probably essential

Recommended background knowledge
No formal requirements, except a keen interest and a lot of time. Programming experience
in C very strongly recommended. Knowledge of assembly and computer architecture helps too.

Target group
mCS, mPDCS.

Term Rewriting Systems

Course code	X_400121 (400121)
Period	Period 2
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	drs. J. Endrullis
Examinator	drs. J. Endrullis
Teaching staff	drs. J. Endrullis
Teaching method(s)	Lecture, Seminar
Level	600

Course objective
Learning the fundamental notions of term rewriting and getting acquainted with some more advanced topics in the field

Course content
Term rewriting systems (TRSs) provide for a natural formalism for specifying rules of computation and investigating their properties. TRSs are of basic importance for functional programming and for the implementation of abstract data types. Applications can also be found in theorem proving, proof checking and logic programming. Some topics that will be covered in the course are:
- abstract reduction systems
- critical pairs and Knuth-Bendix completion
- orthogonality and reduction strategies
- termination (rpo's, monotone algebras)
- combinatory logic
- decidability issues
- infinitary rewriting

Form of tuition
Lectures and practice sessions

Type of assessment
Written examination

Course reading
Course notes will be provided

Target group
mCS, mPDCS, mAI, mMath

Remarks
The course is taught once every two years, the next opportunity will be in study year 2016-2017

The Social Web

Course code	X_405086 ()
Period	Period 4
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen

Coordinator	dr. D. Ceolin
Examinator	dr. D. Ceolin
Teaching staff	dr. L.M. Aroyo, V. Maccatrozzo MSc
Teaching method(s)	Lecture
Level	400

Course objective
In this course the students will learn theory and methods concerning communication and interaction in a Web context. The focus is on distributed user data and devices in the context of the Social Web.

Course content
This course will cover theory, methods and techniques for:

- personalization for Web applications
- Web user & context modelling
- user-generated content and metadata
- multi-device interaction
- usage of social-web data

Form of tuition
- lectures
- practical sessions
- assignments including final paper

Type of assessment
Weighted average of assignments and final paper

Course reading
- course lecture slides
- selected articles, videos and Web links for each lecture

Target group
VU: mIS
UvA: master Information Studies - Human-Centered Multimedia
mCS
mAI

Watson Innovation

Course code	X_405129 ()
Period	Period 2
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Coordinator	dr. L.M. Aroyo
Examinator	dr. L.M. Aroyo
Teaching staff	dr. L.M. Aroyo, A. Dumitrache MSc, B.F.L. Timmermans MSc
Teaching method(s)	Lecture
Level	400

Course objective
The Watson Innovation course is a collaboration between VU Amsterdam and IBM. In this course you will learn the basics and challenges of Cognitive Computing and how to train Cognitive Computing Systems. You will have the unique opportunity to work with multidisciplinary teams on real prototypes of IBM Watson, and explore its potential for answering questions about the city of Amsterdam. You will also have a chance to showcase developed applications and plans to real clients.

Course content
- Basics of Cognitive Computing & IBM Watson
- How to train IBM Watson Instance
- Develop ideas for Cognitive Computing apps
- Build real IBM Watson prototype apps
- Showcase your ideas to real clients

Form of tuition
Lectures & practical sessions at locations of the VU Amsterdam and IBM Netherlands.

Type of assessment
Evaluation of group projects and individual peer-reviews

Course reading
Course lecture slides and related articles:
- What is IBM Watson? (http://www.ibm.com/smarterplanet/us/en/ibmwatson/what-is-watson.html)
- Building Watson: An overview of the DeepQA project (http://www.aaai.org/ojs/index.php/aimagazine/article/download/2303/2165)
- CrowdTruth papers (http://crowdtruth.org/papers/)

Target group
A balanced mix of Computer Science and Business & Economics students (from VU as well as UvA) in their bachelor or master level.

Registration procedure
Sign up through VUnet and http://crowdtruth.org/course.
For more information contact b.timmermans@vu.nl.

Places are limited, so sign up as soon as possible.

Remarks
There will be no lectures through the Christmas period. The period from 18 December till 10 January is reserved for students individual and group work. Office hours will be provided for additional feedback and questions.

Web Services and Cloud-based Systems

Course code	X_418110 ()
Period	Period 5
Credits	6.0
Language of tuition	English
Faculty	Faculteit der Exacte Wetenschappen
Level	400

Course content
http://studiegids.uva.nl/xmlpages/page/2015-2016/zoek-vak/vak/20645

Target group
mCS

Remarks
This course is offered at the UvA. For more information contact: FNWI Education Service Centre, Science Park 904, servicedesk-esc-science@uva.nl, +31 (0)20 525 7100.
Enrolment via https://m.sis.uva.nl/vakaanmelden is required.