
# Parallel and Distributed Computer Systems MSc

Vrije Universiteit Amsterdam - Faculteit der Exacte Wetenschappen - M Parallel and Distributed Computer Syst - 2017-2018

This master’s programme emphasizes experimental computer science, rather than theory, and typically requires design, implementation, testing, and performance analysis of software for advanced parallel and distributed systems. It contains a balance between classes and practical work: about one third of the first three semesters consists of practical projects. The final semester is a master's thesis, which will usually involve doing research in conjunction with one of the faculty members. In contrast to other master’s programmes, PDCS requires that students explicitly apply to be admitted. Up-to-date information can be found at the Beta-website.

**More information**

- All compulsory courses and electives you find in the year schedule;
- A complete description of the programme you find in the Teaching and Examination Regulations;
- For more information about the programma you can contact the academic advisor (VU students only);
- As a VU student you need to register for all courses via VUnet. Only after you completed your enrollment for the study programme you can register for courses;
- More information on all the courses you find through the links below.

Inhoudsopgave

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

Compulsory choice Theoretical Computer Science of 6 credits, at least one choice out of 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.

Vakken:

| Naam | Periode | Credits | Code |
|---|---|---|---|
| Coding and Cryptography | Periode 4 | 6.0 | X_405041 |
| Concurrency and Multithreading | Periode 2 | 6.0 | X_405064 |
| Distributed Algorithms | Periode 5 | 6.0 | X_400211 |
| Logical Verification | | 6.0 | X_400115 |
| Protocol Validation | Periode 1 | 6.0 | X_400117 |

## Constrained Choice Programming (6EC)

Compulsory choice Programming of 6 credits, at least one choice out of 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.

Vakken:

| Naam | Periode | Credits | Code |
|---|---|---|---|
| Android Lab | Periode 5+6 | 6.0 | XM_40011 |
| Kernel Programming | Periode 1 | 6.0 | XM_40014 |
| Parallel Programming Practical | Periode 2+3 | 6.0 | X_400162 |
| PDCS Programming Project | Ac. Jaar (september) | 12.0 | XM_405054 |

## Optional Courses

Vakken:

| Naam | Periode | Credits | Code |
|---|---|---|---|
| Advanced Logic | Periode 4 | 6.0 | X_405048 |
| Binary and Malware Analysis | Periode 5 | 6.0 | X_405100 |
| Evolutionary Computing | Periode 1 | 6.0 | X_400111 |

| | | | |
|---|---|---|---|
| Green Lab | Periode 1 | 6.0 | X_418158 |
| ICT4D in the field | Periode 6 | 6.0 | XM_0008 |
| Industrial Internship | Ac. Jaar (september) | 6.0 | XM_405080 |
| Internet programming | Periode 1 | 6.0 | X_405082 |
| Large Scale Data Engineering | Periode 1 | 6.0 | X_405116 |
| Parallel System Architectures | Periode 1 | 6.0 | XMU_40015 |
| Performance Engineering | Periode 5 | 6.0 | XMU_40016 |
| Performance of Networked Systems | Periode 4 | 6.0 | X_405105 |
| Programming Multi-core and Many-core Systems | Periode 4 | 6.0 | XMU_40018 |
| Selected Topics in Parallel and Distributed Computer Systems | Ac. Jaar (september) | 3.0 | XM_400426 |
| Selected Topics in PDCS | Ac. Jaar (september) | 6.0 | XM_400379 |
| Term Rewriting Systems | Periode 5 | 6.0 | XM_400121 |
| Web Data Processing Systems | Periode 2 | 6.0 | XM_40020 |

## Year 1

Vakken:

| Naam | Periode | Credits | Code |
|---|---|---|---|
| Computer and Network Security | Periode 1 | 6.0 | X_400127 |
| Distributed Systems | Periode 2 | 6.0 | X_400130 |
| Programming Large-scale Parallel Systems | Periode 1 | 6.0 | XM_40017 |
| Secure Software | Periode 2 | 6.0 | XM_40019 |

## Year 2

Vakken:

| Naam | Periode | Credits | Code |
|---|---|---|---|
| Advanced Topics in Distributed Systems | Periode 1 | 6.0 | X_405022 |
| Master Project | Ac. Jaar (september) | 36.0 | XM_400461 |
| Research Proposal Writing | Periode 2 | 6.0 | X_405023 |

## Advanced Logic

| | |
|---|---|
| **Vakcode** | X_405048 (405048) |

| | |
|---|---|
| **Periode** | Periode 4 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | dr. F. van Raamsdonk |
| **Examinator** | dr. F. van Raamsdonk |
| **Docent(en)** | dr. F. van Raamsdonk |
| **Lesmethode(n)** | Hoorcollege, Werkcollege, Deeltoets extra zaalcapaciteit |
| **Niveau** | 500 |

**Doel vak**
The objective of the course Advanced Logic is to obtain a good understanding of modal logic and its
use in computer science and artificial intelligence.

**Inhoud vak**
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: for example 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.

**Onderwijsvorm**
Weekly 2 lectures and 1 exercise class, for the duration of 7 weeks.

**Toetsvorm**
A written exam and assignments that can make half a point bonus.

**Literatuur**
Johan van Benthem, Modal Logics for Open Minds, CSLI Publications 2010.

**Aanbevolen voorkennis**
The bachelor course Logic and Modelling
or an equivalent introduction to first-order logic.

**Doelgroep**
mAI, mCS, mPDCS

**Uitleg in Blackboard/Canvas**
The information about Advanced Logic is shared via the webpage of the course http://www.cs.vu.nl/~tcs/al.

**Intekenprocedure**
Registration is organized in the standard way.

## Advanced Topics in Distributed Systems

| | |
|---|---|
| **Vakcode** | X_405022 (405022) |
| **Periode** | Periode 1 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |

| | |
|---|---|
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | dr. ir. M.X. Makkes |
| **Examinator** | dr. ir. M.X. Makkes |
| **Docent(en)** | J. Urbani |
| **Lesmethode(n)** | Hoorcollege |
| **Niveau** | 600 |

**Doel vak**
The students will be able to critically discuss advanced topics relevant for traditional and modern distributed systems.

**Inhoud vak**
The course takes the form of a seminar that is based on a selection of papers that either have had a strong impact on distributed systems today, or explore novel ideas that may be important in the future. Subjects will cover important aspects of distributed systems such as communication, data consistency, replication, fault tolerance, performance, scalability, etc. Also, modern distributed systems such as next-generation Web-based systems and wireless sensor networks will have their place. For this seminar we expect the students to actively participate by means of presentations and discussions.

**Onderwijsvorm**
Seminar.

**Toetsvorm**
Presentations, active participation at seminar, and a short (4-page) position paper.

**Literatuur**
A (selection of a) list of papers, reflecting recent developments.

**Vereiste voorkennis**
Distributed Systems (X_400130).

**Doelgroep**
mPDCS

**Overige informatie**
This course is only accessible for mPDCS students.

## Android Lab

| | |
|---|---|
| **Vakcode** | XM_40011 () |
| **Periode** | Periode 5+6 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | dr. S. Voulgaris |
| **Examinator** | dr. S. Voulgaris |
| **Lesmethode(n)** | Hoorcollege, Computerpracticum |
| **Niveau** | 400 |

**Doel vak**
The objective of this practical is to put concepts of Computer Networks and Operating Systems into practice, in the context of Android smartphones. It consists of low-level programming assignments on Android smartphones, requiring the thorough understanding of operating systems and network concepts.

**Inhoud vak**
This practical consists of two parts. The first is to build a TCP implementation for Android smartphones. The second is to develop a Chat application over the TCP stack of the first part. Both parts are to be written in Java using the Android operating system.

**Onderwijsvorm**
One project assignment.

**Literatuur**
Computer Networks, by Andrew S. Tanenbaum

**Vereiste voorkennis**
Good knowledge of Java.

**Aanbevolen voorkennis**
Good knowledge of Java.

**Doelgroep**
mCS, mPDCS

**Overige informatie**
Lecturer:
dr. S. Voulgaris

## Binary and Malware Analysis

| | |
|---|---|
| **Vakcode** | X_405100 () |
| **Periode** | Periode 5 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | C. Giuffrida |
| **Examinator** | C. Giuffrida |
| **Docent(en)** | prof. dr. ir. H.J. Bos, C. Giuffrida |
| **Lesmethode(n)** | Hoorcollege |
| **Niveau** | 500 |

**Doel vak**
Deepening insights in static and dynamic analysis, applied to binaries and malware

**Inhoud vak**
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.

**Onderwijsvorm**
Hoorcollege and practical

**Literatuur**
Slides and online material

**Doelgroep**
mCS-HPDC, mCS-IWT, mPDCS

## Coding and Cryptography

| | |
|---|---|
| **Vakcode** | X_405041 (405041) |
| **Periode** | Periode 4 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | prof. dr. R.M.H. de Jeu |
| **Examinator** | prof. dr. R.M.H. de Jeu |
| **Docent(en)** | prof. dr. R.M.H. de Jeu |
| **Lesmethode(n)** | Hoorcollege |
| **Niveau** | 500 |

**Doel vak**
The goal of the course is to give an elaborate introduction to the theory of error correcting codes, and to
discuss some algebraic background of cryptography. After taking this

course, the student:
* will know the definitions of basic notions in error correcting codes (Hamming distance, error detecting/correcting capability);
* will be able to decode certain 2-error correcting BCH-codes;
* will be able to perform calculations in Galois fields with 2^n elements;
* will be able to decode Reed-Solomon codes over such Galois fields;
* will be able to decrypt messages encrypted under some public key cryptography (RSA, ElGamal).

**Inhoud vak**
This course provides a thorough introduction to the theory of error correcting codes, and also, as a small part of it, treats the algebraic background of some protocols in 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. These are used in the error correcting codes underlying, for example, CD-ROM, audio CD, and QR-codes. For the small part on cryptography we discuss some modern public key cryptography (e.g., RSA, ElGamal, DSA), which form part of the protocol underlying https.

**Onderwijsvorm**
Lectures and exercise classes

**Toetsvorm**
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).

**Literatuur**
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).

**Aanbevolen voorkennis**
Some knowledge on linear algebra (vectors, matrices, nullspaces, basis, dimension, some determinants),
on the integers modulo n, and on polynomials. Although these will be reviewed, experience shows that the
course is difficult to follow without having seen this material before.

**Doelgroep**
XM_CS 1, XM_PDCS 1, XM_MAT_B 1, XM_MAT_E 1, XM_MAT_T 1, XM_MAT_S 1, XM_MAT_ADS 1, XM_MAT_AG 1

## Computer and Network Security

| | |
|---|---|
| **Vakcode** | X_400127 (400127) |
| **Periode** | Periode 1 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | prof. dr. ir. H.J. Bos |
| **Examinator** | prof. dr. ir. H.J. Bos |

| | |
|---|---|
| **Docent(en)** | prof. dr. ir. H.J. Bos |
| **Lesmethode(n)** | Hoorcollege |
| **Niveau** | 500 |

**Doel vak**
This is a 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 (such as spoofing, scannning, sniffing, DoS, and TCP hijacking), and web
attacks such as SQL injection, cross-site scripting, and other vectors used by computer hackers. Besides basic attacks, students will also learn
about state-of-the-art exploitation methods. The course is very(!) hands-on.

**Inhoud vak**
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).

**Onderwijsvorm**
Lectures and (very challenging) practical assignments.

**Toetsvorm**
Written exam (30%) and practical assignments (70%).

**Literatuur**
No set book. All material will be made available during the course.

**Vereiste voorkennis**
Knowledge of C is highly recommended (and probably essential)

**Aanbevolen voorkennis**
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.

## Concurrency and Multithreading

| | |
|---|---|
| **Vakcode** | X_405064 (405064) |

| | |
|---|---|
| **Periode** | Periode 2 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | prof. dr. W.J. Fokkink |
| **Examinator** | prof. dr. W.J. Fokkink |
| **Docent(en)** | prof. dr. W.J. Fokkink |
| **Lesmethode(n)** | Hoorcollege, Werkcollege |
| **Niveau** | 400 |

**Doel vak**
This course provides a comprehensive presentation of the foundations and programming principles for multicore computing devices.

Specific learning objectives are:
* To provide insight into fundamental notions of multicore computing and their relation to practice: locks, read-modify-write operations, mutual exclusion, consensus, construction of atomic multi-reader-multi-writer registers, lost wakeups, ABA problem.
* To provide insight into algorithms and frameworks for multicore computing and their application in multi-threaded programs: mutual exclusion algorithms, spin locks, monitors, barriers, AtomicStampedReference class in Java, thread pools in Java, transactional memory.
* Analyzing algoritmes for multicore computing with regard to functionality and performance: linearizability, starvation- and wait-freeness, Amdahl's law, compute efficiency gain of parallelism.
* Mastering elementary datastructures in the context of multicore computing: lists, queues, stacks.
* Programming in multi-threaded Java, and performing experiments with such programs.

**Inhoud vak**
The course consists of the following topics: Shared memory, mutual exclusion, synchronization operations, concurrent data structures, scheduling, transactional memory, and a multithreaded programming assignment.

**Onderwijsvorm**
Lectures: 4 hours per week, exercise classes: 4 hours per week.

**Toetsvorm**
The written exam counts for 65% and the programming assignment for 35% of the final mark.

Both for the written exam and the programming assignment at least a 5.0 must be obtained (and the overall average mark should be at least 5.5).

Only students that achieved at least a 3.0 for their initial programming assignment are offered a resit opportunity for this assignment.

**Literatuur**
Maurice Herlihy, Nir Shavit, The Art of Multiprocessor Programming, Morgan Kaufmann, 2008.

**Aanbevolen voorkennis**
Datastructures & Algorithms

Programming in Java

**Doelgroep**
mAI, mCS, mPDCS

**Overige informatie**
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.

## Distributed Algorithms

| | |
|---|---|
| **Vakcode** | X_400211 (400211) |
| **Periode** | Periode 5 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | prof. dr. W.J. Fokkink |
| **Examinator** | prof. dr. W.J. Fokkink |
| **Docent(en)** | prof. dr. W.J. Fokkink |
| **Lesmethode(n)** | Hoorcollege, Werkcollege |
| **Niveau** | 500 |

**Doel vak**
The main objective is to provide students with an algorithmic frame of mind for solving fundamental problems in distributed computing. They obtain insight into concurrency concepts, and are offered a bird's-eye view on a wide range of algorithms for basic and important challenges in distributed systems.

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

**Inhoud vak**
The following topics are treated in the course: Logical clocks, 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.

**Onderwijsvorm**
4 hours per week HC
4 hours per week WC

**Toetsvorm**

Written examen (plus a take-home exercise sheet that can provide up to 0.5 bonus point, if a passing mark for the written exam is achieved).

**Literatuur**
W.J. Fokkink. Distributed Algorithms: An Intuitive Approach. MIT Press, 2013.

**Aanbevolen voorkennis**
Datastructures & Algorithms

**Doelgroep**
mAI, mCS, mPDCS

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

## Distributed Systems

| | |
|---|---|
| **Vakcode** | X_400130 (400130) |
| **Periode** | Periode 2 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | dr. A. Uta MSc |
| **Examinator** | prof. dr. ir. A. Iosup |
| **Docent(en)** | prof. dr. ir. A. Iosup |
| **Lesmethode(n)** | Hoorcollege, Werkcollege |
| **Niveau** | 400 |

**Doel vak**
After taking this course, students will be able to:
1. Explain the basic concepts, objectives, and functions of distributed computing systems, e.g., communication, resource management and scheduling, data consistency, fault-tolerance, performance.
2. Compare distributed computing with other computing paradigms (i.e., centralized, parallel).
3. Identify the different flavors of modern distributed systems (i.e., peer-to-peer systems, cluster computing, grid computing, cloud computing, datacenters, distributed HPC, SDN, Big Data systems, IoT systems).
4. Analyze the trade-offs inherent in the design of modern distributed systems.
5. Design your portfolio distributed-system, with many basic and some complex operations of modern distributed systems.
6. Implement your portfolio distributed-system.
7. Analyze your portfolio distributed-system.

**Inhoud vak**
This course focuses on distributed computing systems. In general, debugging and tuning existing systems, and designing, implementing, and analyzing new distributed computing systems remains vital and challenging.

Starting with the mid-1990s, computing is undergoing a revolution, in which collections of independent computers appear to users as a single,

albeit distributed, computing system. Motivated by the advent of the Internet, by the increase in the computation capacity of consumer computers, by the commoditization of server-grade machines, by energy constraints, etc., the distributed computing paradigm has permeated all fields using computers. Current distributed computing applications range from social networks to banking, from peer-to-peer file-sharing to high-performance computing used in research, from massively multiplayer online games to business-critical workloads, etc. Important advances have helped to fuse heterogeneous resources into truly global distributed systems, for example in scientific computing, where distributed computation is using Big Data and distributed sensors to produce meaningful progress for the humankind. We will focus in this course on a number of these modern examples of distributed computing systems.

Although so many distributed systems already exist, the list of conceptual and technical challenges they pose is long. Depending on requirements, even trivial communication between nodes of the distributed system can be challenging. The failure of a single node, or sometimes even a performance hiccup, can bring an entire system down; with it, other nodes or entire other systems may also crash, experiencing correlated and catastrophic failures. Data consistency and coordinating nodes remain important challenges made worse by the large-scale of real-world deployments. Poor resource management and naive scheduling can lead to orders-of-magnitude higher operational costs and consumption of energy that we simply cannot spare. It is not uncommon for a modern distributed system to quickly rise and then fall in popularity, as exemplified by the 2016 example of Pokemon Go. We will present in this course real-world situations where modern distributed systems have behaved poorly.

Addressing these challenges requires unique approaches and concepts. Separating concerns and breaking down problems into smaller cases often lead to limited success, because many properties of distributed systems can only be achieved end-to-end. Can anyone imagine a perfectly reliable production pipeline, if even one of its key stages can suffer failures? Building capability by adding resources is often offset by the distributed nature of the system. Can anyone ignore the physical limitations of communication around the globe? In this course, we will focus on the unique approaches and principles of distributed systems, from specific architectures and communication protocols, to specific concepts in resource management and scheduling, data consistency, fault-tolerance, and performance.

**Onderwijsvorm**
The course is taught as a series of lectures, in combination with self-study and with a large practical assignment.

**Toetsvorm**
Written exam. Depending on enrollment, an oral exam may also be available.
Report on the practical assignment.

**Literatuur**
The course uses as textbook the book:
Maarten van Steen and Andrew S. Tanenbaum, Distributed Systems, 3rd. Ed., online edition, 2017. (free for all) [Online] Available: https://www.distributed-systems.net/index.php/books/distributed-systems-

The lecture slides also recommend additional literature. The study guide available on Canvas also indicates other worthwhile sources of information.

**Vereiste voorkennis**
Ability to work in teams for the practical assignment.
Ability to develop code using modern software engineering practices, e.g., setting up your own GitHub repository, co-editing using tools such as Overleaf or Sharelatex, etc., is a big plus.

**Aanbevolen voorkennis**
Students should have taken standard courses on:
- Computer networks.
- Programming paradigms, in particular OOP and/or actor-based approaches.
- Software engineering.
Prior experience with Internet, web, distributed, or parallel programming is helpful.
Prior experience with operating systems development and analysis, and in general experience with computer systems courses, is a big plus.

**Doelgroep**
mCS, mPDCS, mSNE (UvA)

**Overige informatie**
This course uses gamification.

## Evolutionary Computing

| | |
|---|---|
| **Vakcode** | X_400111 (400111) |
| **Periode** | Periode 1 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | J.V. Heinerman MSc |
| **Examinator** | J.V. Heinerman MSc |
| **Docent(en)** | prof. dr. A.E. Eiben, J.V. Heinerman MSc |
| **Lesmethode(n)** | Hoorcollege |
| **Niveau** | 400 |

**Doel vak**
This course has a threefold objective: 1) To learn about computational methods based on Darwinian principles of evolution. 2) To illustrate the usage of such methods as problem solvers and as simulation tools. 3) To gain hands-on experience in performing computational experiments with evolutionary algorithms.

**Inhoud vak**
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, evolution strategies, evolutionary programming, genetic programming). 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.

**Onderwijsvorm**
Oral lectures and compulsory Java programming assignment (in teams of 3). Highly motivated students can replace the programming assignment by a special research track under the personal supervision of the lecturer(s). These research projects aim at publications.

**Toetsvorm**
Written exam and programming assignment (weighted average). To pass the course as a whole, you must pass both the exam and the programming assignment.

**Literatuur**
Eiben, A.E., Smith, J.E., Introduction to Evolutionary Computing. Springer, 2015, 2nd edition, ISBN 978-3-662-44873-1.

**Vereiste voorkennis**
Java programming skills are necessary to do the practical assignment.

**Doelgroep**
mBA, mAI, mCS, mPDCS

## Green Lab

| | |
|---|---|
| **Vakcode** | X_418158 () |
| **Periode** | Periode 1 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | dr. I. Malavolta |
| **Examinator** | dr. I. Malavolta |
| **Docent(en)** | dr. I. Malavolta |
| **Lesmethode(n)** | Hoorcollege, Practicum |
| **Niveau** | 400 |

**Doel vak**
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.

**Inhoud vak**
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.

**Onderwijsvorm**
Lectures (H). Lab sessions (pra).

**Toetsvorm**
Teamwork. Project assignments (pro).

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

**Aanbevolen voorkennis**
Basic statistical analysis techniques (descriptive statistics and most common tests).

**Doelgroep**
mCS, PDCS, mAI

## ICT4D in the field

| | |
|---|---|
| **Vakcode** | XM_0008 () |
| **Periode** | Periode 6 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | dr. J. Gordijn |
| **Examinator** | drs. A. Bon |
| **Niveau** | 400 |

## Industrial Internship

| | |
|---|---|
| **Vakcode** | XM_405080 () |
| **Periode** | Ac. Jaar (september) |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | prof. dr. W.J. Fokkink |
| **Examinator** | prof. dr. W.J. Fokkink |
| **Niveau** | 400 |

**Doel vak**
The student applies study contents in an industrial setting. After completion of the internship, the student will be able to reflect on the

peculiarities of industrial practice, in relation to knowledge and skills obtained at university.

**Inhoud vak**
Individual project work by which the student applies the study contents in an industrial setting.

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. The project must aim at applying state-of-the-art methods and technology and must require an original contribution by the student.

Before the start of the internship, the student has to get approval for the internship project by a VU or UvA Computer Science lecturer. 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.

The report must contain both a description of the project and a reflection on study contents being applied in an industrial context. For the project, the problem, the student's solution, and an evaluation of the results must be presented. For the reflection part, (at least) the following question need to be addressed: 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?

The overall grade is a weighted average of the project contents and the student's solution (50%), the quality of the written report (35%), and the quality of the reflection of study vs. industrial context (15%).

Internship assignments that either do not give the student sufficient freedom to devise his or her own original solution or that can not be reported upon due to disclosure limitations by the company are not eligible for this course.

**Onderwijsvorm**
individual project work in an industrial setting

**Toetsvorm**
written report

**Aanbevolen voorkennis**
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.

**Doelgroep**
mCS, mPDCS

**Overige informatie**
Various lecturers

## Internet programming

| | |
|---|---|
| **Vakcode** | X_405082 () |
| **Periode** | Periode 1 |

| | |
|---|---|
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | dr. ir. M.X. Makkes |
| **Examinator** | dr. ir. M.X. Makkes |
| **Docent(en)** | dr. ir. M.X. Makkes |
| **Lesmethode(n)** | Hoorcollege |
| **Niveau** | 400 |

**Doel vak**
Guide the student through the design and development of Network and Web applications.

**Inhoud vak**
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.

**Onderwijsvorm**
Lectures combined with lab assignments

**Toetsvorm**
Final exam plus lab assignments

**Literatuur**
Course slides

**Vereiste voorkennis**
Knowledge of C, Java

**Aanbevolen voorkennis**
Good knowledge of both C and Java

**Doelgroep**
mAI, mCS, mPDCS

## Kernel Programming

| | |
|---|---|
| **Vakcode** | XM_40014 () |
| **Periode** | Periode 1 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | dr. K. Razavi |
| **Examinator** | dr. K. Razavi |
| **Docent(en)** | C. Giuffrida |
| **Lesmethode(n)** | Hoorcollege |
| **Niveau** | 400 |

**Doel vak**
Gaining a deep understanding of kernel
programming and of major OS subsystems design. This course is very
tough and time-consuming.

**Inhoud vak**
The course will feature a number of hands-on assignments accompanied by lectures on advanced operating system kernel design and programming concepts. In each assignment, students will be expected to start with a minimal kernel implementation and exercise their kernel hacking skills on one of the major operating subsystems (i.e., memory management, drivers, etc.). This will involve programming in both C and assembly as well as directly interfacing with the hardware. The course will also link lectures and assignments to modern operating system features and offer insights into state-of-the-art OS research efforts.

**Onderwijsvorm**
Hoorcollege and practical

**Literatuur**
Slides and online material

## Large Scale Data Engineering

| | |
|---|---|
| **Vakcode** | X_405116 () |
| **Periode** | Periode 1 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | prof. dr. P.A. Boncz |
| **Examinator** | prof. dr. P.A. Boncz |
| **Docent(en)** | prof. dr. P.A. Boncz |
| **Lesmethode(n)** | Hoorcollege |
| **Niveau** | 500 |

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

More information is found on http://event.cwi.nl/lsde

**Inhoud vak**
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 (data access patterns, hardware latency/bandwidth), as well as possess certain
skills and experience in managing large-scale computing infrastructure.

**Onderwijsvorm**
There are two lectures per week, and the course 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.

**Toetsvorm**
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 exam; the grade is based on the two assignments grades, the grade for the in-class presentation and attendance/participation.

**Literatuur**
scientific papers provided in the course

**Vereiste voorkennis**
Hadoop environments 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.

**Aanbevolen voorkennis**
Programming proficiency in C/C++ or Java

**Doelgroep**
mCS, mPDCS

## Logical Verification

| | |
|---|---|
| **Vakcode** | X_400115 (400115) |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | dr. J.C. Blanchette |
| **Examinator** | dr. J.C. Blanchette |
| **Docent(en)** | dr. F. van Raamsdonk |
| **Lesmethode(n)** | Hoorcollege, Practicum |
| **Niveau** | 500 |

**Doel vak**
Introduction to the proof assistant Coq and its type-theoretic foundations.

**Inhoud vak**
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.

**Onderwijsvorm**
2 times 2 hours theory class, 2 times 2 hours practical work

**Toetsvorm**
Written exam,
obligatory Coq-exercises,
obligatory hand-in theory exercises,
possibly presentations of papers.

**Literatuur**
Course notes possible with some recent papers as addition.

**Vereiste voorkennis**
An introduction course in logic.

**Doelgroep**
mCS, mAI, mMath, mPDCS

**Overige informatie**
The course will not be taught in 2017-2018. The next opportunity will be in study year 2018-2019 in period 2.

## Master Project

| | |
|---|---|
| **Vakcode** | XM_400461 (400461) |
| **Periode** | Ac. Jaar (september) |
| **Credits** | 36.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | prof. dr. W.J. Fokkink |
| **Examinator** | prof. dr. W.J. Fokkink |
| **Niveau** | 600 |

**Doel vak**
With the Master project, the student is to demonstrate the ability to integrate knowledge, insights, and skills gained so far in the Master programme, and to apply them to a new or otherwise unknown subject.

**Inhoud vak**
The student is offered a research question that needs to be answered following a systematic approach. This approach includes steps such as exploring relevant literature, and will, in general, consist of setting up and carrying out experiments by means of simulations, emulations, or actual systems software. The results and findings will be described in a thesis conforming to the academic standards in the field. An oral presentation of the project results concludes the project.

**Toetsvorm**
The grade will be determined based on the quality of the performed project work, a written thesis, and an oral presentation.

**Aanbevolen voorkennis**
The student must have completed (almost) the complete study program before starting the Master Project.

**Doelgroep**
mPDCS

**Overige informatie**
Various lecturers

## Parallel Programming Practical

| | |
|---|---|
| **Vakcode** | X_400162 (400162) |
| **Periode** | Periode 2+3 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | prof. dr. ir. H.E. Bal |
| **Examinator** | prof. dr. ir. H.E. Bal |
| **Docent(en)** | prof. dr. ir. H.E. Bal |
| **Lesmethode(n)** | Hoorcollege |
| **Niveau** | 500 |

**Doel vak**
Obtain practical experience with parallel programming using different programming systems.

**Inhoud vak**
During this practical, several parallel programs have to be written, using different programming environments, including Java, MPI, and Chapel. 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 presents for each problem the implementation approach taken and discusses the outcomes of the experiments conducted.

**Onderwijsvorm**
Practical computer work; students work on their own; there is one (kickoff) meeting and supervision from assistants.

**Toetsvorm**
Practical computer work, final report.

**Vereiste voorkennis**
Knowledge of parallel programming in Java/Ibis, MPI, and Chapel (as taught in the Parallel Programming course) is required, as well as practical experience with C and Java.

**Doelgroep**
Masters Computer Science, PDCS, AI, and Computational Science

**Overige informatie**

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

## Parallel System Architectures

| | |
|---|---|
| **Vakcode** | XMU_40015 () |
| **Periode** | Periode 1 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Docent(en)** | dr. A.D. Pimentel |
| **Lesmethode(n)** | Hoorcollege, Computerpracticum |
| **Niveau** | 400 |

**Inhoud vak**
http://studiegids.uva.nl/xmlpages/page/2017-2018/zoek-vak/vak/39272

**Intekenprocedure**
Registration is required via https://www.sis.uva.nl before the start of the semester.
Please visit the website of your programme through http://student.uva.nl and check the A-Z list 'Course and Exam Registration' for more information.
Please note that in 2017-18 this course will be given at the VU University Amsterdam.

**Overige informatie**
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.
For courses taught in period 1 and period 2, enrolment via https://datanose.nl/#specialenrol is required.

## PDCS Programming Project

| | |
|---|---|
| **Vakcode** | XM_405054 (405054) |
| **Periode** | Ac. Jaar (september) |
| **Credits** | 12.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | prof. dr. W.J. Fokkink |
| **Examinator** | prof. dr. W.J. Fokkink |
| **Niveau** | 600 |

**Doel vak**
Let the student get hands-on experience with developing systems-level software in the context of ongoing research projects

**Inhoud vak**
PDCS programming projects are related to existing research programs in computer systems. There is no set course description as each project is negotiated individually with the permanent staff member supervising and grading it.The assignment aims to offer students a challenging project that is research-oriented by nature. Students are supposed to talk to staff members individually to see whether they have a project that matches the student's interests.

Next to the computer program, a written report must be produced in which the idea behind the program is described, as well as the novelty and how it fits in the context of the overall research project. The student should
also describe lessons learned, and reflect how the project builds on knowledge acquired in earlier Bachelor and especially Master courses.

The final mark is based on the quality of the programming work (50%), the written report (30%), and the academic excellence shown and effort invested by the student during the project (20%).

**Onderwijsvorm**
Individual programming project

**Toetsvorm**
To be decided by the 1st and 2nd supervisor. There is a special evaluation form.

**Literatuur**
To be decided by the supervisor.

**Doelgroep**
mPDCS

**Overige informatie**
Various lecturers

## Performance Engineering

| | |
|---|---|
| **Vakcode** | XMU_40016 () |
| **Periode** | Periode 5 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Niveau** | 500 |

**Inhoud vak**
http://studiegids.uva.nl/xmlpages/page/2017-2018/zoek-vak/vak/39560

**Overige informatie**
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.

# Performance of Networked Systems

| | |
|---|---|
| **Vakcode** | X_405105 () |
| **Periode** | Periode 4 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | prof. dr. R.D. van der Mei |
| **Examinator** | prof. dr. R.D. van der Mei |
| **Docent(en)** | dr. ing. T. Kielmann, prof. dr. R.D. van der Mei |
| **Lesmethode(n)** | Hoorcollege |
| **Niveau** | 400 |

**Doel vak**
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.

**Inhoud vak**
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?

**Onderwijsvorm**
Classroom lectures and practical homework assignments.

**Toetsvorm**

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

**Literatuur**
Textbook, supplemented with a reader on Stochastic Performance Modelling.

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

**Vereiste voorkennis**
The students should have basic knowledge of computer networks.

**Doelgroep**
mBA, mCS, mPDCS, mEct

## Programming Large-scale Parallel Systems

| | |
|---|---|
| **Vakcode** | XM_40017 () |
| **Periode** | Periode 1 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | prof. dr. ir. H.E. Bal |
| **Examinator** | prof. dr. ir. H.E. Bal |
| **Docent(en)** | prof. dr. ir. H.E. Bal |
| **Lesmethode(n)** | Hoorcollege |
| **Niveau** | 400 |

**Doel vak**
You will
- be introduced to the domain of High Performance Computing (HPC)
- learn about design methods for parallel algorithms
- compare different parallel computer architectures
- analyze performance of network topologies
- develop basic familiarity with a range of parallel programming constructs,
environments and languages
- gain insight into some selected parallel applications

**Inhoud vak**
This lecture discusses how programs can be written that run in parallel on a large number of processors, with the main 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 MPI message passing) and more advanced techniques like parallel object-oriented approaches from the Java ecosystem or dedicated HPC programming languages will be
discussed. In particular, Cray’s high productivity language Chapel is discussed in much more detail
(about 4 lectures).
Several parallel applications are
discussed, including nearest-neighbor stencil computations, N-body simulations and search algorithms.

**Onderwijsvorm**
Lectures (4 hours per week), given by prof.dr.ir. Henri Bal (VU) and Dr Clemens Grelck (UvA).
Interested students may also obtain an account on our DAS research cluster and do simple experiments with MPI.
There is also a separate Parallel Programming
Practical (6 ECTS) in P2 and P3 whose aim is to complement the contents of this course with practical skills and experience. That course makes heavy use of DAS.

**Toetsvorm**
Written exam

**Literatuur**
Papers will be made available on Canvas

**Doelgroep**
mAI, mBIO, mCS, mPDCS, m Computational Science

**Overige informatie**
Lecturers:
prof.dr.ir. Henri Bal (VU)
Dr. Clemens Grelck (UvA)

## Programming Multi-core and Many-core Systems

| | |
|---|---|
| **Vakcode** | XMU_40018 () |
| **Periode** | Periode 4 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Niveau** | 400 |

**Inhoud vak**
http://studiegids.uva.nl/xmlpages/page/2017-2018/zoek-vak/vak/38101

**Overige informatie**
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

| | |
|---|---|
| **Vakcode** | X_400117 (400117) |
| **Periode** | Periode 1 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | dr. A. Ponse |
| **Examinator** | dr. A. Ponse |
| **Lesmethode(n)** | Hoorcollege, Practicum, Werkcollege |

| | |
|---|---|
| **Niveau** | 500 |

**Doel vak**
Learning to use formal techniques for specification and validation of communication protocols.

**Inhoud vak**
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.

**Onderwijsvorm**
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.

**Toetsvorm**
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.

**Literatuur**
Wan Fokkink, Modelling Distributed Systems, Springer 2007. An online version of this book (2nd edition) will be available.

**Aanbevolen voorkennis**
Logica en Modelleren

**Doelgroep**
mAI, mCS, mPDCS, master of Logic

## Research Proposal Writing

| | |
|---|---|
| **Vakcode** | X_405023 (405023) |
| **Periode** | Periode 2 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | prof. dr. ir. H.E. Bal |
| **Examinator** | prof. dr. ir. H.E. Bal |
| **Docent(en)** | prof. dr. ir. H.E. Bal |
| **Lesmethode(n)** | Hoorcollege |
| **Niveau** | 600 |

**Doel vak**
The aim of this course is to learn to read papers at a research level, organize the material for the framework of a seminar talk, and practice presentation skills for such talk. At the end of this course, the student will be able to acquire the prerequisites for reading and understanding a paper by researching the literature on his own, understand the logic of a paper, and be able to critically evaluate a paper. He or she will be able to extract and condense the material for a talk of a fixed length, and give a captivating and interesting talk to fellow students.

**Inhoud vak**
This course has the single main aim to teach students the first steps of writing a research proposal. Students are asked to evaluate a number of existing proposals that were submitted by staff members in recent years. In addition, each student will also have to write his or her own proposal, which is then evaluated by fellow students following a procedure very similar to what happens in real life. This class may only be attended by PDCS students.

**Onderwijsvorm**
There will be two meetings (in working group style), one to discuss the existing proposals and one to discuss the students proposals.

**Toetsvorm**
The grading is based on
- Your reviews of the existing proposals
- Your own proposal
- Your reviews of the other students' proposals
- Class participation

**Doelgroep**
mPDCS

## Secure Software

| | |
|---|---|
| **Vakcode** | XM_40019 () |
| **Periode** | Periode 2 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | dr. K. Razavi |
| **Examinator** | dr. K. Razavi |
| **Docent(en)** | prof. dr. ir. H.J. Bos |
| **Lesmethode(n)** | Hoorcollege |
| **Niveau** | 500 |

**Doel vak**
This course covers the security of system software and the low-level hardware (e.g., CPU and the memory subsystem). This is not a generic course about computer security, but rather a focused course on cutting-edge topics in computer systems security.

This will be a highly research-oriented course with the aim of preparing the students for their literature study and thesis work in the area of computer systems security. The course introduces the students to critical thinking by writing reviews for a selected number of papers and discussing them in the class with other students. Finally, the students will learn how research work is carried out by reproducing some of the most high-profile attacks that the VUSec group has recently carried out. The practical part of the course is highly demanding and helps the students put their reviews in perspective and advances their knowledge in the area of operating systems and computer architecture.

By the end of the course the students can critically think about a research paper and have a meaningful discussion about it with others, useful for their literature study and master thesis. Further, they will be familiar with the current state of research in various computer security subjects and can decide which area they would like to follow for their thesis. The practical part of the course will show the students the efforts involved in doing practical research and helps them develop the skill set necessary for their master thesis work.

**Inhoud vak**
The students will be given a selected number of papers (around 30-40) to choose from for their reviews (a few papers per student). The students are expected to write reviews on a weekly basis. One day per week, the reviewers will discuss a few of these papers and their reviews in the class.

The course will feature a number of guest lectures aimed to prepare the students with the practical part of the course. For the practical part, the students will team up to reproduce one of the three sophisticated attacks that we have recently developed in the group using a well-defined structure. Depending on the attack, the groups will compromise ASLR from the browser, look at the traces of the memory management unit in the cache, or escalate their privilege using a Rowhammer attack. One day per week, the students will discuss their issues and how they addressed them in the class.

**Onderwijsvorm**
Lectures, (very challenging) practical assignments, paper reviews and group discussions.

**Toetsvorm**
Written exam (30%), practical assignments (40%), reviewing and participation in group discussions (30%).

**Literatuur**
No set book. All material will be made available during the course.

**Vereiste voorkennis**
Knowledge of operating systems, C, and computer architecture is essential. The skill set gained during Kernel Programming or Computer and Network Security offered in the previous period will be very helpful.

**Aanbevolen voorkennis**
No formal requirements, except a keen interest and a lot of time. Familiarity with core operating system and computer architecture concepts such as virtual address translation or TLBs will make it easier to follow this course.

**Doelgroep**
mCSS, mPDCS.

## Selected Topics in Parallel and Distributed Computer Systems

| | |
|---|---|
| **Vakcode** | XM_400426 (400426) |
| **Periode** | Ac. Jaar (september) |
| **Credits** | 3.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | prof. dr. W.J. Fokkink |
| **Examinator** | prof. dr. W.J. Fokkink |
| **Niveau** | 500 |

**Doel vak**
The goal of this course is to treat special topics, in the field of parallel and distributed computer systems, that are otherwise not included in the regular curriculum, to individual students as part of further preparation for their master's degree.

**Inhoud vak**
The actual content of the course is to be decided after consultation of one the PDCS staff members, who will act as project supervisor.

**Onderwijsvorm**
Individual study under guidance of the supervisor.

**Toetsvorm**
To be decided by the supervisor.

**Literatuur**
To be decided by the supervisor.

**Doelgroep**
mPDCS

**Overige informatie**
Lecturers:
Various lecturers.

## Selected Topics in PDCS

| | |
|---|---|
| **Vakcode** | XM_400379 (400379) |
| **Periode** | Ac. Jaar (september) |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | prof. dr. W.J. Fokkink |
| **Examinator** | prof. dr. W.J. Fokkink |
| **Niveau** | 500 |

**Doel vak**
The goal of this course is to treat special topics, in the field of parallel and distributed computer systems, that are otherwise not included in the regular curriculum, to individual students as part of further preparation for their master's degree.

**Inhoud vak**
The actual content of the course is to be decided after consultation of one the PDCS staff members, who will act as project supervisor.

**Onderwijsvorm**
Individual study under guidance of the supervisor.

**Toetsvorm**
To be decided by the supervisor.

**Literatuur**
To be decided by the supervisor.

**Doelgroep**
mPDCS

## Term Rewriting Systems

| | |
|---|---|
| **Vakcode** | XM_400121 (400121) |
| **Periode** | Periode 5 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | drs. J. Endrullis |
| **Examinator** | drs. J. Endrullis |
| **Docent(en)** | drs. J. Endrullis |
| **Lesmethode(n)** | Hoorcollege, Werkcollege |
| **Niveau** | 400 |

**Doel vak**
Learning the fundamental notions of term rewriting and getting acquainted with some more advanced topics in the field.

**Inhoud vak**
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
- term rewriting
- combinatory logic
- termination (rpo's, monotone algebras)
- confluence, critical pairs, orthogonality
- Knuth-Bendix completion
- strategies

- modularity
- decidability issues
- infinitary rewriting

**Onderwijsvorm**
Lectures and practice sessions

**Toetsvorm**
Written examination

**Literatuur**
Course notes will be provided

**Doelgroep**
mCS, mPDCS, mAI, mMath

## Web Data Processing Systems

| | |
|---|---|
| **Vakcode** | XM_40020 () |
| **Periode** | Periode 2 |
| **Credits** | 6.0 |
| **Voertaal** | Engels |
| **Faculteit** | Faculteit der Exacte Wetenschappen |
| **Coördinator** | J. Urbani |
| **Examinator** | J. Urbani |
| **Docent(en)** | J. Urbani |
| **Lesmethode(n)** | Hoorcollege |
| **Niveau** | 400 |

**Doel vak**
After taking this course, you will be able to:

- Understand the fundamentals of the most important problems that modern Web companies face daily;
- Process large amounts of Web data efficiently using state-of-the-art tools that are currently used in the Web industry;
- Extract useful insights from raw data available on the Web;
- Adapt or reuse techniques used on the Web to other fields (e.g. Data Mining, Artificial Intelligence) where similar problems might occur.

**Inhoud vak**
The Web constitutes the largest repository of knowledge that is available to mankind, and its impact on modern society is unprecedented at many levels. Many Web companies are valued with billion dollar quotations and are now central to our modern life.

The key players in the Web industry must face numerous challenges that are concerned with the size, distribution, heterogeneity, and the uncontrolled nature of the Web. Systems to process Web data require the application of a combination of techniques spanning databases, distributed systems, data mining, and artificial intelligence.

The goal of this course is to introduce the student to the most advanced systems and techniques which deal with Web data. Important classes of problems concern:

- the storage and retrieval of Web data (How can we store and retrieve information from large social networks, graphs, or large volumes of text?)
- efficient entity disambiguation (What is a particular web page talking about?)
- large-scale knowledge extraction (What sort of knowledge can we extract from web documents -- e.g. Wikipedia?)
- effective link prediction (Is there a connection between two users/events/concepts?)
- expressive ontological inference (Can current knowledge lead to more implicit knowledge?)
- trust (Can we trust the content on a certain blog post?)

This course will describe techniques to perform these tasks with a particular emphasis on scalability, which is a crucial aspect in this domain. In order to better understand the challenges and effectiveness of current solutions, the student will be called to implement practical assignments on realistic Web data. These assignments will be part of the final evaluation of the course.

**Onderwijsvorm**
The course takes the form of lectures and practical assignments.

**Toetsvorm**
A combination of exams and group homework assessments.

**Literatuur**
A mixture of scientific publications and other material available on the Web.

**Doelgroep**
XM_CS