Programme-specific Annex to the Teaching and Examination Regulations for the Bachelor's programme in Technical Computer Science

The rules in this Annex are part of the programme portion of the Student Charter, including the Teaching and Examination Regulations for the Bachelor's programme in Technical Computer Science offered by the Faculty of Electrical Engineering, Mathematics and Computer Science of the University of Twente.

1. CONTENTS AND STRUCTURE OF THE PROGRAMME

1.1 General objectives of the programme (Article 7.13, paragraph 2c of the Higher Education and Research Act)

The objective of the Bachelor's degree programme in Technical Computer Science is to train students at the Bachelor's level, instilling in them a solid foundation in mathematics and a thorough basic knowledge and understanding of the field of Computer Science. The programme is comprehensive and focuses not only on software and information systems, but also, and especially, on computer systems and communication networks. The programme emphasizes skills and the societal context, and offers students the opportunity to explore another field by taking a minor. Students complete the programme by conducting an individual research project and a group design project.

Most graduates will continue their education by enrolling in a Master's programme, although the expertise and skills they acquire during the Bachelor's programme will allow them to find gainful employment in the field.

1.2 The final attainment targets of the programme (Article 7.13, paragraph 2c of the Act)

Knowledge and experience as relevant to the domain of Technical Computer Science
The graduate has knowledge and understanding of the field of Technical Computer Science. This knowledge includes:

1. Software: programming languages, principles of software development, software engineering, formal methods
2. Computers: architecture and organization, management systems
3. Networks: networks and communications, principles of communication systems
4. Fundamentals of Computer Science: algorithms and complexity, discrete structures, parallel and distributed computing
5. Human Media Interaction: computational science, graphics and visualization, human-computer interaction, intelligent systems
6. Information management: databases
7. Information security: fundamentals of security, network security, cryptography
8. Mathematics: discrete mathematics, calculus, linear algebra, probability and statistics

Design

1. The graduate is capable of the integrated application of relevant, field-specific knowledge to systems design.
2. The graduate is capable of specifying a problem and devising a solution based on a general description of the problem.
3. The graduate is capable of devising solutions/designing systems by selecting and implementing methods, models and techniques.
4. The graduate is capable of evaluating the properties of solutions/systems and of making a substantiated choice between different solutions based on his/her evaluation.

Research

1. The graduate is capable of critically analysing field-specific problems.
2. The graduate is capable of systematically setting up and implementing a research project.
3. The graduate is capable of contributing to the further development of the field by working in a sub-field.

Organizational ability

1. The graduate is capable of independently acquiring and incorporating new knowledge and skills as required.
2. The graduate is capable of analysing and discussing ethical, social, cultural and societal aspects of problems, solutions and developments in the field.
3. The graduate understands team dynamics, and is capable of working in a team and with a variety of stakeholders such as the client and end-users.
4. The graduate is capable of communicating effectively with colleagues and non-specialists, both orally and in writing.
5. The graduate is capable of organizing his/her working processes and reflecting on their effectiveness.
6. The graduate is capable of taking a position on an issue and of substantiating this position with regard to a design or scientific argument.
7. The graduate has a multidisciplinary attitude.
8. The graduate has intercultural skills.

1.3 Content of the programme and related examinations (Article 7.13, paragraph 2a of the Act)

The table below shows the units of study comprising the Twente Education Model (TOM) curriculum. Section 1.3.3 of this Annex contains a curriculum that has been adjusted for the combined final degree audit for Technical Computer Science and Applied Mathematics. The Board of Examiners of the relevant programme is to publish details regarding the content of a unit of study in the course catalogue at least six weeks before the start of the teaching period (semester or quarter) in which the unit of study is offered.

1.3.1 The TOM curriculum (Cohorts from 2013 and later)

The table below shows the subjects in the order in which they are offered, the student's preferred prior knowledge and any additional prerequisites. The associated examination tables are included in Annex 1.

Tabel 1. Curriculum Technische Informatica

Course code	Course name	Q	Language	Prerequisites*
B1 Fase (Year 1)				
201700139	Pearls of Computer Science	1A	EN	
201700117	Software Systems	1B	EN	Desirable: Pearls of Computer Science
201600146	Network Systems	2A	EN	Desirable: Pearls of Computer Science
201300180	Data & Information	2B	EN	Desirable: Software Systems
B2 Fase (Year 2)				
201400210	Computer Systems	1A	EN	
201600105	Intelligent Interaction Design	1B	EN	Desirable: Software Systems + Data & Information
201400433	Discrete Structures & Efficient Algorithms	2A	EN	Desirable: Computer Systems
xxxxxxxxx	Minor / Elective module	2B	EN	Requirement for participation in minor module: the B1 phase must be complete upon registration in Osiris
B3 Fase (Year 3)				
xxxxxxxxx	Minor / Elective module	1A	EN	Requirement for participation in minor module: the B1 phase must be complete upon registration in Osiris
xxxxxxxxx	Minor / Elective module	1B	EN	Requirement for participation in minor module: the B1 phase must be complete upon registration in Osiris
201500121	Design Project	1A of 2A	EN	Required: 120 credits (excluding minor) upon registration in Osiris
201500120	Research Project	1B of 2B	EN	Required: 120 credits (excluding minor) upon registration in Osiris

*Desirable: some prior module-specific knowledge is advised, although this is not a prerequisite. Required: prerequisite must be met prior to starting the module.

1.3.2 Elective section

1. The elective section consists of an elective module and two minor modules;
2. One of the modules listed in Table 2 must be chosen;
3. Approved minors are listed on the minors site: www.utwente.nl/minor;
4. By way of derogation from the list of approved in-depth minors as listed on the minors site, the programme has designated the Serious Gaming and Smart Cities minors as contract minors.
5. In addition to (2), a maximum of one additional elective module may be taken as an in-depth minor;
6. Een student dient vooraf toestemming te vragen aan de examencommissie voor een vrije minor;
7. The Examination Board uses the following guidelines to assess the student's request:

a. The educational component of the minor must be at an academic level;
b. At least 15 of the 30 credits must involve a paradigm shift;
    i. The contents of the minor must not fall within the field of computer science; or
    ii. The contents of an exchange minor may fall within the field of computer science, business administration or industrial engineering and management, provided that the minor is taken at an institute of higher education abroad and the educational component of the minor is at an academic level.
c. The educational component of the minor may not overlap with the programme's compulsory units of study;
d. Up to five credits may be devoted to courses on the language and culture of the host country.

Table 2. Elective modules / Programme-specific in-depth minor modules

Course code	Course name	Q	Prerequisites
201500057	Smart Spaces	1A	Advisable: Software Systems
201500053	Cyber-Physical Systems	1B	Advisable: Software Systems
201500025	Web Science	1B	Advisable: Software Systems
201400537	Programming Paradigms	2B	Advisable: Software Systems

See www.utwente.nl/ti for further information regarding the Examination Board's procedure for approving the minor. Once approval has been granted, the Bureau of Educational Affairs (BOZ) is responsible for the administrative procedure involved in enrolling the student in the relevant minor.

1.3.3 Sequence requirements (Article 7.13, paragraph 2s of the Act)

1. A student may enrol in the minor through the Minor Bureau once he/she has completed the B1-fase upon registration in Osiris;
2. A student may only enrol in the final semester modules Design Project (201500121) and Research Project (201500120) once he/she has earned at least 120 credits, excluding minors;

1.3.4 Applied Mathematics and Computer Science double degree

An adjusted curriculum applies to students pursuing a double degree in Technical Computer Science and Applied Mathematics. This programme is detailed on the next page.

The chart on the next page shows the components of the units of study. In each quarter, the components listed under Applied Mathematics form a cohesive unit of study, as do the components under Technical Computer Science.

1.4 Programme format (Article 7.13, paragraph 2i of the Act)

The programme is only offered on a full-time basis.

2. Language of tuition (Article 3.3, paragraph 1 of the Teaching and Examination Regulations)

The programme is taught in English for the 2016 cohort (and later cohorts), and in Dutch for the 2015 cohort (and earlier cohorts).

Dubbelprogramma TI/TW, 2016-2017

1e jaar

Kwartiel 1	21 EC	**Kwartiel 2**	21 EC	**Kwartiel 3**	20 EC	**Kwartiel 4**	20 EC
gezamenlijk: *Math A*	1,5 EC	module 2 TW: *Lin.Struc II* *Analyse I* *Project: prooflab*	10 EC	module 3 TW: *Signalen en Transf.*	5 EC	module 4 TW: *Vector Calculus*	5 EC
module 1 TW: *Lineaire Structuren*	6 EC	module 2 TI: *Math B2*	3 EC	*deel Kansrekening*	3 EC	module 3 TW *deel Kansrekening*	2 EC
module 1 TI: *Math B1*	2,5 EC	*Programmeren theorie en project*	8 EC	module 3 TI: *Network Systems (excl Math C1)*	12 EC	module 4 TI: *Data & Informatie excl. Kansrekening*	12 EC
Parels	8 EC						
Project TI	3 EC						

2e jaar

Kwartiel 1	20 EC	**Kwartiel 2**	20	**Kwartiel 3**	21 EC	**Kwartiel 4**	15 EC
module 5 TW: *Wisk. Statistiek*	5 EC	module 6 TW: *Differentiaalvergl* *Systeemtheorie*	8 EC	gezamelijk: *Discrete Structuren & Efficiënte Algoritmen*	15 EC	module 8 TW: *Modelling and Analysis of stochastic processes for Math*	15 EC
module 5 TI: *Computer Systems*	15 EC	module 6 TI: *Intelligent Interaction Design (excl Statistiek)*	12 EC	uit mod 3 TW: Project *(inclusief intro Wisk. Mod.)*	6 EC		

3e jaar

Kwartiel 1	10 EC	**Kwartiel 2**	15 EC	**Kwartiel 3**	15 - 20 EC	**Kwartiel 4**	15 - 20 EC
module 5 TW: *Analyse II* *Project* *Presenteren*	10 EC	*minorruimte*	15 EC	*afstudeerfase*		*afstudeerfase*	

Totale omvang van dit programma:

tussen de 213 EC en 223 EC

3. TEACHING AND ASSESSMENT

3.1 Assessment and examination formats (Article 7.13, paragraph 2l of the Act)

Annex 1 details the examination format for each unit of study.

3.2 Registration of results

In addition to Article 4.1, Guideline TER:

1. Exemptions for examinations are indicated with the code 'VR'.
2. Exemptions are assigned a numerical value of 6.
3. The examination results of complete (V) and incomplete (NVD) have no numerical values.

3.3 Participation in tests (Art. 4.3(3), TER Guideline)

1. If attendance in designated educational activities is a prerequisite for participation in a test, then the module coordinator must decide on granting exemptions to students resitting the test or must define an alternative method to satisfy the attendance requirement.
2. If a module has been changed and the non-divisible component is no longer clearly identifiable, then the module coordinator must decide which tests must be passed in order to complete the former non-divisible component.
3. A substantiated request must be submitted to the Examination Board if a student wishes to participate in sessions that are not part of the regular module.

3.4 Third attempt

If a student requires more than two consecutive academic years to pass a module, then the student must agree on a study plan together with the Study Advisor at least two weeks prior to the start of the relevant module. The study plan must include agreements on time keeping, active participation in tutorials and other aspects.

3.5 Examination transparency

In addition to Article 4.4 (Guideline TER), the programme is to ensure that information is made available for each examination regarding its level, structure and marking norms, e.g. by providing a sample examination, an examination from a previous year or a collection of sample examination questions.

3.6 Period of validity (Art. 4.7(2) Guideline TER)

The module components are indicated by a Roman numeral in the module descriptions in Table 2. The results of these module components remain valid indefinitely.

3.7 Confidentiality

In addition to Article 4.9(2) (Guideline TER):

1. Reports of final assignments are public documents except in the following cases.
2. The Programme Board may deem a report to be confidential for a specific period based on a detailed request:
    a. The first supervisor must submit a request to the Programme Board prior to the start of the final assignment.
    b. The confidential report must be accessible/available to the committee responsible for assessing the final assignment, the Programme Board, and representatives of bodies that have a statutory duty of overseeing the quality of the assessment or the programme as a whole.
    c. The parties mentioned above are required to observe confidentiality with regard to the report.
3. In the case of a confidential report as referred to in point 2, the public presentation of the report may be amended to ensure that no confidential information is made public.

3.8 Teaching evaluations (Art. 4.10(3) Guideline TER)

1. The online Student Experience Questionnaire (SEQ) is used for evaluation purposes at the conclusion of each module;
2. Additionally, the module coordinator may initiate supplementary evaluations, such as additional surveys and panel discussions during the module or at its conclusion;
3. If the SEQ results and/or student complaints give reason for concern, then the programme director is to discuss the matter with the module coordinator either during the module or at its conclusion;
4. They are to use this discussion to develop a plan for improving the remainder of the module or for the subsequent module, including a strategy for evaluating the improvements.

4. FINAL DEGREE AUDIT

4.1 Pass/Fail Regulation

1. Students who meet the following requirements will pass the Bachelor's final degree audit for the TCS programme:
   a. The student has received an assessment for all units of study of the Bachelor's final degree audit;
   b. The student's final results are 6 or higher for all units of study;

   In all other cases not specified under (1), the student will not pass the final degree audit and will not receive a Bachelor's degree.

4.2 Cum Laude

1. A student may pass the Bachelor's final degree audit with distinction (cum laude) upon meeting the following requirements:
   a. The student passes the Bachelor's final degree audit within four years of initial enrolment (performance requirement);
   b. The student's average mark is 8.0 or higher (non-numeric assessments and exemptions not included). This is a weighted average based on the relative number of credits per unit of study.
   c. No more than one unit of study may have a final result of 6.
   d. The mark for the module part Research Project of the module Research Project (201500120) is 8.0 or higher.
2. In exceptional cases and at the student's request, the Examination Board may award the distinction of cum laude if the student has met all requirements with the exception of the performance requirement, due to extenuating circumstances. These circumstances may involve delays recognized and provided for by the institution. It should be noted that the distinction of cum laude is never awarded automatically, but only following individual assessment of the student's academic achievements.

5. BINDING RECOMMENDATION (BSA)

A student will receive a positive BSA upon satisfying one of the following conditions (Article 6.3, Guideline TER):

1. Successful completion of three complete modules;
2. Successful completion of 45 credits of module components, including at least three mathematics modules (Math A+B1, Math B2, Math C1, Probability Theory);
3. In addition to the stipulations in (2), a module component has been successfully completed if it is part of a fully completed module, or if the test results of the completed module component is a 5.5 or higher in the case of a module that has not yet been completed;
4. Students pursuing a double degree in Technical Computer Science and Applied Mathematics are subject to an additional BSA provision: the BSA may involve removal from the Applied Mathematics programme if the student fails to earn 15 or more credits from the units of study associated with the Technical Computer Science programme.

6. ADMISSION

6.1 Admission to a Master's programme

A student with a Bachelor's degree in Technical Computer Science will gain automatic admission to the following Master's programmes at the University of Twente:

- Computer Science
- Business Information Technology*
- Embedded Systems (3TU)*
- Human Media Interaction
- Internet Science & Technology

* Additional requirements apply for admission to this Master's programme for graduates of the University of Twente's Technical Computer Science Bachelor's programme.

7. STUDY MATERIALS

Students who started on the programme in September 2013 or later must obtain a 'budget notebook' from the Notebook Service Centre (or acquire a similar or better device).

Annex 1: Assessment Tables modules

Final testing schedules are to be published on the module's Blackboard course page at least two weeks prior to module commencement.

1. Pearls of Computer Science (201700139)

The Pearls of Computer Science module consists of two indivisible module components:

I. Math A+B1 (4 EC)
II. Pearls of Computer Science (11 EC)

Figure 1. Assessment Table Pearls of Computer Science

Module parts	Name Assessment	Type of Assessment	Individual / Group	Weight assessment (%)	Minumum grade	Weight result assessment (%)
I	Math A+B1	Written test	I	100	5.5*	27
		Math case	G	Pass		
II	Pearls	8x written test	I	12.5% each	5.5 each*	53
		8x assignment	G	PASS for access to written test		
				Weighted sub-average	**5.5**	
	Project	Project- assessment	G	100	5.5	20
	Academic Skills	Assignments	I	PASS	PASS	0
				Weighted average	**5.5**	

*OR 'Math A & B1' 5.0=<grade<5.5 OF TWO Pearls 5.0=<grades<5.5 is allowed IF the sub-weighted average is 5.5 or higher (grade =>5.5).

1.1 Learning goals module part I

- Clearly express formulations;
  - Work with elementary properties of sets and logic;
  - Construct elementary proofs using basic techniques;
  - Work with elementary properties of combinatorics.
- Work with vectors and elementary properties of functions, especially with the rules of differentiability;
  - Apply elementary vector operations;
  - Calculate dot product and cross product;
  - Apply elementary properties of functions;
  - Calculate derivatives using differentiation rules and the derivatives of elementary functions;
- Work with limits and the definitions of continuity and differentiability and applications, for functions of 1 variable;
  - Calculate limits
  - State and apply the definition of (left, right) continuity;
  - Work with limits involving infinity;
  - State and apply the definition of differentiability;
  - Calculate and apply linear approximations and differentials;
  - Calculate the absolute extreme values on a close bounded interval;
  - Apply l'Hôpital's rule to indeterminate forms of limits;

- Investigate functions in two veriables;
    - Plot graphs and contour lines;
    - Investigate continuity and differentiability;
    - Calculate partial derivatives;
    - Calculate the tangent plane and linearization.

1.2 Learning goals module part II

After absorbing the pearl “Computer Architecture”

- The student can work with binary and hexadecimal number representations, binary logic and boolean algebra.
- The student knows the basic architecture of a computer and its concepts register, memory, address, ALU, clock, program, program counter, instruction and mnemonic.
- The student can write simple programs for a microcomputer in machine language using arithmetic, I/O, and (conditional) jump instructions.

After absorbing the pearl “Algoritmics”

- The student can explain the importance of searching and sorting algorithms;
- The student can explain the principle of and differences between linear and binary search methods, as well as between bubble sort and merge sort;
- The student understands the complexity arguments behind the aforementioned algorithms and can analyse which is the best solution in what context;
- The student can apply simple imperative programming concepts: if/then, while, integer variables and arrays;
- The student can program the above algorithms in Python.

After absorbing the pearl “Databases”

- The student knows the basic concepts of databases
- The student can design a databaseschema for a simple case using ER-modeling.
- The student can realize such a design in a relational DBMS using SQL.
- The student can query and update a relational DBMS with SQL.

After absorbing the pearl “Functional Programming”

- The student knows the basic concepts of the chosen functional language,
- The student is able to explain the concept of function application,
- The student understands the principles of recursion and their relationship with induction,
- The student is able to express simple algorithms in the chosen functional language.

After absorbing the pearl “Intelligent Interaction”

- The student knows the basic concepts of artificial intelligence and can design a simple rule-based socially intelligent system.
- The student knows the basic principles of machine learning and can design and execute a classification task with a (black-box) classifier.

After absorbing the pearl “Computer Networks and Operating Systems”

- The student can identify and explain the most important responsibilities of an operating system.
- The student understands the working and layered construction of packet-switched computer networks,
- and can reason about the therein occurring delays.
- The student knows the basic working of the internet and internet applications as well as the protocols like TCP, IP, and HTTP.

After absorbing the pearl “Cryptography”

- The student understands symmetric-key encryption: block ciphers and their modes of operation, stream ciphers, and the one-time-pad. They know some basic design techniques of such ciphers, such as linear feedback shift registers and Feistel networks. Also, they know about the existence of DES and AES.
- The student understands asymmetric-key encryption: the RSA cryptosystem and the RSA signature scheme. They know how to use it for key exchange (hybrid encryption).
- The student has some necessary background knowledge of elementary number theory (modular arithmetic, Euclidean algorithm) and basic probability theory, for a proper understanding of the above-mentioned cryptosystems.

After absorbing the pearl “Requirements Engineering”

- The student can explain the importance of controlled and predictable realisation of software and project artifacts
- The student knows a few techniques for project management
- The student can derive and formulate requirements as well as acceptance criteria for them
- Besides reaching these learning goals, it is an explicit additional goal of this pearl to formulate requirements and set up a structure for the execution of the project of this module. Project After carrying out the project
- The student can coherently apply and integrate knowledge and skills in a team and for a project that is based on real-world aspects.
- The student has experienced going through all phases of realizing a software artifact

After absorbing and carrying out the exercises of Academic Skills

- The student can explain the importance of working together in a team
- The student can effectively give and receive feedback
- The student understands and can apply the core quality quadrant model of Daniel Ofman
- The student understands and can apply the Belbin team role model
- The student can effectively resolve team conflicts
- The student can evaluate a project
- The student understands the concepts of fraud and plagiarism, and knows how to behave responsibly as a professional concerning these aspects

2. Software Systems (201700117)

The Software Systems module consists of two indivisible module components:

I. Math B2 (3 EC)
II. Software Systems (12 EC)

Figure 2. Assessment Table Software Systems

Module parts	Name Assessment	Type of Assessment	Individual / Group	Weight assessment (%)	Minumum grade	Weight result assessment (%)
I	Math B2	Written test	I	100	5.5*	20
		Math Case	G	Pass		
II	Design	Written test	I	100	5.5*	20
		Assignments	I	Pass		
	Programming	Written test	I	100	5.5*	20
		Assignments	I	Pass		
	Weighted sub-average				**5.5***	
	Design Project	Report	G	100	5.5	20
	Programming Project	Product	G	100	5.5	20
		Report	G			
	Academic Skills	Assignments	I	PASS	PASS	0
	Weighted average				**5.5**	

*Out of the marked (*) module component grades ONE mark lower than 5.5, but at least 5.0 (5.0 = <rate <5.5) is allowed IF it's sub-weighted average is 5.5 or higher.

2.1 Learning goals module part I

- Work with elementary properties of integrals and calculate integrals using different techniques, for functions of 1 variable;
  - formulate Riemann sums
  - formulate and use the Fundamental Theorem of Calculus
  - calculate integrals using anti-derivatives
  - calculate integrals using the substitution method
  - calculate integrals using the technique of integration by parts
  - calculate improper integrals using limits
- Work with power series and Taylor series, for functions of 1 variable;
  - calculate the convergence radius by the ratio test
  - calculate Taylor series and polynomials
- Solve linear differential equations;
  - solve first order equations using integrating factor
  - solve second order homogeneous equations with constant coefficients using the characteristic equation
  - solve first and second order equations with constant coefficients using the method of undetermined coefficients
  - solve initial / boundary value problems
- Work with complex numbers;
  - plot (sets of) complex numbers in the plane
  - calculate absolute value and argument of a complex number to express the complex number in polar form
  - apply the complex arithmetic operations
  - find roots of a complex number and solve binomial equations

2.2 Learning goals module part II

Concerning Software Design, after successfully finishing this module a student is capable of:

- Specifying an existing software system or a software system under design in terms of UML models (including class diagrams, activity diagrams and state machines);
- Interpreting these models, explaining the relation between different models, and between each model and the software code, and the usefulness of defining models in addition to writing software code;
- Explaining the commonly recognised phases of software development;
- Applying version management in software development projects;
- Explaining basic software metrics and using them to assess quality characteristics of a code base.

Concerning Programming, after successfully finishing this module a student is capable of:

- Explaining and applying the core concepts of imperative programming, such as variables, data types, structured programming statements, recursion, lists, arrays, methods, parameters, and exceptions.
- Explaining and applying the core concepts of object-orientation, such as object, class, value, type, object reference, interface, specialisation / inheritance, and composition.
- Using the Model/View/Controller pattern when developing applications.
- Writing simple multi-threaded programs, and explaining the operation and problems (race-conditions) of concurrent threads, and using synchronisation mechanisms, such as monitors, locks and wait sets.
- Writing programs using basic network mechanisms, based on sockets.
- Explaining and applying the basic concepts of security engineering and applying them to Java programs.
- Writing software of average size (around ten classes) in Java, by using the concepts mentioned above, including the use of algorithms for searching and sorting data
- Documenting software of this size, by using (informal)preconditions, post conditions and (class) invariants, and (informally) justifying the correctness of the implemented software.
- Explaining how this software can be tested, defining and executing a test plan, and measuring and improving test coverage.

Concerning Academic Skills, after successfully finishing this module a student is capable of:

- Describing the major principles of effective time management.
- Applying these principles to make a personal planning for a medium long term period, e.g., a study semester, and for a medium-sized project.
- Formulating personal strengths and weaknesses with regard to time management, study behaviour and project work.
- Describing the major principles for defining a general project planning.
- Applying these principles when reflecting on some previous project planning.
- Giving and receiving peer feedback.
- Identifying major personal pitfalls concerning procrastination behaviour.

3. Network Systems (201600146)

The Network Systems module consists of two indivisible module components:

I. Math C1 (3 EC)
II. Network Systems (12 EC)

Figure 3. Assessment Table Network Systems

Module parts	Name Assessment	Type of Assessment	Individual / Group	Weight assessment (%)	Minumum grade	Weight result assessment (%)
I	Math C1	Written test	I	100	5.5*	20
		Math Case	G	Pass		
II	Network Systems Theory	4x written tests	I	100	5.5 each*	50
	Observation lab	Assignments	I	Pass	Pass	0
				Weighted sub-average	**5.5***	
	Network Systems Project	Challenges	G	50	5.5	30
		Project examination	G	50	5.5	
	Academic Skills	Assignments	I	Pass	Pass	0
				Weighted average	**5.5**	

*Out of the marked (*) module component grades ONE mark lower than 5.5, but at least 5.0 (5.0 = <rate <5.5) is allowed IF it's sub-weighted average is 5.5 or higher.

3.1 Learning goals module part I

After following this course students should be able to:

- work with subspaces of Rn and determinants and connect them with the previous concepts;
- write a solid line of argument, based on a clear question and understand the basic principles of effective presentations.

3.2 Learning goals module part II

- understand basic principles in communication systems, networks, and networked applications;
- describe and understand key protocols underlying the operation of the Internet;
- make simple quantitative models of network systems, and use them to evaluate these systems;
- analyze the behavior of common networking systems using network monitoring tools;
- design and implement basic networking protocols and applications;
- work with systems of linear equations, vectors, matrices, linear transformations and explain the connections between these concepts.

4. Data & Information (201300180)

The Data & Information module consists of two indivisible module components:

I. Probability Theory (3 EC)
II. Data & Information (12 EC)

Figure 4. Assessment Table Data & Information

Module parts	Name Assessment	Type of Assessment	Individual / Group	Weight assessment (%)	Minumum grade	Weight result assessment (%)
I	Probability Theory	Written test	I	100	5.5*	20
		Assignments	I	Pass		
II	5 Themes	4x Written test	I	100 each	5.5 each*	4x 10
				Weighted sub-average	**5.5***	
	Project	Product	G	100	5.5	40
		Report	G			
		Presentation	G			
	Academic Skills	Assignments	I	Pass	Pass	0
				Weighted average	**5.5**	

*Out of the marked (*) module component grades ONE mark lower than 5.5, but at least 5.0 (5.0 = <rate <5.5) is allowed IF it's sub-weighted average is 5.5 or higher.

4.1 Learning goals module part I

After successful completion of the module student is able to …

- explain and apply the use of elementary probability theory, such as combinatoric probability theory, conditional probability, independence;
- explain and apply probability distributions of one or more random variables, (discrete) conditional probabilities, and compute expectation, variance, and correlation coefficient;
- explain and apply basic discrete and continuous distributions, including binomial, geometric, Poisson, uniform, exponential and normal distributions.

4.2 Learning goals module part II

After successful completion of the module student is able to …

Agile software engineering

- develop software following Agile principles: SCRUM meetings, task boards, burn-down charts, frameworks, etc.;
- apply requirements-based testing;

Requirements engineering

- identify business requirements and translate these to user stories;
- specify functional and non-functional requirements;
- prioritize requirements in collaboration with various stakeholders;
- design a UML class diagram;
- systematically design web based applications using UML;

Structured data

- derive a logical database schema from a UML class diagram;
- identify functional dependencies and use these to systematically normalize a database to BCNF;
- formulate questions and translate these to SQL queries;
- apply SQL triggers in simple cases;
- identify transactions and explain the effect of different isolation levels on concurrency;

Web programming

- design and implement complex multi-tier web applications;
- use repositories and version management;
- integrate web applications with existing (REST-ful) services;
- build user interfaces with frameworks for HTML, CSS, and javascript;
- explain the consequences of server-side vs. client-side scripting, servlets, Ajax, JSP, Web frameworks, etc.;

Semi- and unstructured data

- apply basic techniques for handling XML/JSON data in an XML database using XML standards such as XPath and XQuery
- apply basic techniques for handling XML/JSON data in a relational database using extensions to the SQL standard, such as SQL/XML and JSON types, functions and operators
- understand the basic theoretical principles behind tree data structures and indexing of tree data
- understand the basic theoretical principles behind information retrieval and apply basic full text querying support in XML and relational databases

Security

- protect applications against unauthorized access;
- project applications against (SQL-)injection and cross-site scripting;

Academic skills: Project skills

- apply the Belbin team role model
- apply the core quality quadrant model of Daniel Ofman
- effectively give and receive feedback
- effectively resolve team conflicts
- explain the concepts of fraud and plagiarism and behave responsibly as a professional concerning these aspects

5. Computer Systems (201400210)

The Computer Systems module is an indivisible module worth 15 credits

Figure 5. Assessment Table Computer Systems

Module parts	Name Assessment	Type of Assessment	Individual / Group	Weight assessment (%)	Minumum grade	Weight result assessment (%)
I	Discrete Mathematics	Written test	I	100	5.5*	20
	Operating Systems	Assignments	I	100	5.5*	26
		Mondeling	I	PASS		
		Written test	I	Indien resultaat assignment lager is dan 5.5		
	Computer Architecture & Organisation	Written test	I	100	5.5*	20
		Assignment	I	PASS		
				Weighted sub-average	**5.5***	
	Project	Projectplan	G	30	5.5	27
		Dagelijkse verslagen	G	PASS		
		Video	G	30		
		Demo	G	40		
		Reflectierapport samenwerking	I	PASS		
	ICT & Law	Participatie	I	PASS	5.5	7
		Written test	I	50		
		Versalg	G	50		
				Weighted average	**5.5**	

*Out of the marked (*) module component grades ONE mark lower than 5.5, but at least 5.0 (5.0 = <rate <5.5) is allowed IF it's sub-weighted average is 5.5 or higher.

5.1 Learning goals module Computer Systems

Computer Architecture and Organisation

- Design circuits using basic logic gates
- Calculate with different number representations
- Understand mechanisms within a processor
- Program a processor
- Indicate the elements of a computer system and explain their functionality
- Design specific parts of a computer system

Project:

- Integrate the knowledge and skills that are taught in the CAO and DH parts for EE and in the CAO and OS parts for CS.
- Have students from EE and CS cooperate within a project.

Operating Systems:

- Understand the major mechanisms of current general-purpose operating systems exemplified by Linux.
- Appreciate the design space and trade-offs involved in implementing an operating system.
- Be capable of basic system-oriented programming and providing simple extensions to an operating system.

- Understand the exploitation of vulnerabilities and privilege escalation

ICT and Law:
- Signaling of relevant IT-juridical aspects in the execution of the work of computer scientists and in their task to communicate on these aspects with legal professionals
- Understand the possibilities and limitations w.r.t. the legal protection of software, databases and domainnames.
- Integrate requirements concerning the law on privacy during design and realisation of ICT systems and processes.
- Have insight in the criminal regime w.r.t. computercrime.

Discrete Mathematics:
- Apply logic and set theory
- Apply formal concepts of function and operation
- Understand relations and their properties

6. Intelligent Interaction Design (201600105)

The Intelligent Interaction Design module is an indivisible module worth 15 credits.

Figure 6. Assessment Table Intelligent Interaction Design

Module parts	Name Assessment	Type of Assessment	Individual / Group	Weight assessment (%)	Minumum grade	Weight result assessment (%)
I	Design & Evaluation of HCI	Written test	I	100	5.5*	20
	Statistical Techniques	Written test	I	100 – bonus x 5	5.5*	20
		4 assignments voor bonuspuntent	I	5% each**		
	AI Theory	Written test	I	100	5.5*	20
				Weighted sub-average	**5.5***	
	AI Practical	Practicumtoets	G	100	5.5	15
	HCI project	Project toetsing	G	100	5.5	25
				Weighted average	**5.5**	

* Out of the marked (*) module component grades ONE mark lower than 5.5, but at least 5.0 (5.0 = <rate <5.5) is allowed IF it's sub-weighted average is 5.5 or higher.
** IF assignment grade > written test grade

6.1 Learning goals module Intelligent Interaction Design

- The student can design, develop and evaluate low fidelity and high fidelity prototypes of an intelligent interactive system that is well justified in context.
- The student is able to take real users into account in the analysis, design, and evaluation of interactive systems with respect to both usability and user experience.
- The student can formulate a research question and answer it by choosing and applying various research methods, collecting data, analysing the data using the appropriate statistical or other methods, and drawing conclusions from this.
- The student can explain and apply the main AI-techniques concerning search, Bayesian networks and machine learning.

7. Discrete Structures & Efficient Algorithms (201600270)

The Discrete Structures & Efficient Algorithms module is an indivisible module worth 15 credits.

Figure 7. Assessment Table Discrete Structures & Efficient Algorithms

Module parts	Name Assessment	Type of Assessment	Individual / Group	Weight assessment (%)	Minumum grade	Weight result assessment (%)
I	Discrete Structures & Algorithms	Written test	I	100	5.5*	35
	Algebra & Finite Automata	Written test	I	100	5.5*	35
	Homework bonus	Assignments	I	Max +1.0 cijferpunt bovenop het weighted sub-average		
	Gewogen sub-gemiddeld (exclusief bonus)				**5.5***	
	Research	Product	G	Pass	5.5	30
		Paper	G	100		
		Presentation	G			
		Programmeercompetitie Bonus	G	Max +2.0 cijferpunt		
	Weighted average (exclusief bonus)				**5.5**	

* Out of the marked (*) module component grades ONE mark lower than 5.5, but at least 5.0 (5.0 = <rate <5.5) is allowed IF it's sub-weighted average is 5.5 or higher.

7.1 Learning goals module Discrete Structures & Efficient Algorithms

The teaching goals are a mix of theoretical understanding of important concepts in discrete mathematics and theoretical computer science, and acquiring practical ability of effectively using discrete structures and techniques while working on several implementation projects of different levels of complexity. Upon completion, the student is able to:

- understand and use discrete structures for modelling and problem solving, more specifically, work with and argue formally about data structures, graphs and networks, regular expressions, grammars, groups, rings, integral domains, fields, vector spaces over arbitrary fields, pushdown automata and Turing machines,
- work with a basic toolbox of techniques such as mathematical induction, asymptotic analysis, solution of recurrence relations, generating functions, transformations, Euclidean algorithm (also for polynomials), the classification of finite fields,
- practically realize and test algorithm designs, more specifically, implement algorithms for discrete structures (e.g. graphs, automata), use adequate data structures to achieve efficient implementations, perform computational experiments with algorithms, document computational results,
- document a research project in the form of a written research paper and presentation, more specifically, locate the studied problem in mathematical and literature context, describe the achieved research results in context of the state-of-the-art, describe the scope and impact of the achieved results, describe limitations and future developments.

8. Programming Paradigms (201400537)

The Programming Paradigms module is an indivisible module worth 15 credits.

Figure 8. Assessment Table Programming Paradigms

Module parts	Name Assessment	Type of Assessment	Individual / Group	Weight assessment (%)	Minumum grade	Weight result assessment (%)
I	Functional Programming	Written test	I	50 each	5.5*	15
	Concurrent Programming	Written test	I	100	5.5*	20
		Bonus	I	+1 cijferpunt		
	Compiler Construction	Take home toets	I	50 each	5.5	20
				Weighted sub-average	**5.5***	
	Functional & Logic Programming Projects	Project-assessment	G	50 each	5.5	15
	Integration Project	Project assessment	G	100	5.5	30
				Weighted average	**5.5**	

* Out of the marked (*) module component grades ONE mark lower than 5.5, but at least 5.0 (5.0 = <rate <5.5) is allowed IF it's sub-weighted average is 5.5 or higher.

8.1 Learning goals module Programming Paradigms

After successful completion of this module, the student is able to:

- Describe the major programming paradigms (FP, LP and CP) and their essential characteristics and differences
- Write basic programs in all major programming paradigms
- Solve non-trivial programming problems in FP and CP
- Explain the concepts and importance of typing, in terms of FP and CC
- Explain and use the typical types and data structures in FP and CP
- Explain and take advantage of the evaluation and execution mechanisms of FP (lazy evaluation) and CP (hardware-related aspects, concurrency models)
- Explain and use the following concepts of FP: recursion, list comprehension, higher order functions, parameter accumulation, function composition, lazy evaluation.
- Explain and use the following concepts of CP: interleaving, fairness, deadlock, memory models, synchronisation, locking.
- Explain and use the following concepts of CC: syntactic and semantic analysis, scanning, parsing, run-time organisation, code generation, optimisation.
- Write a compiler for a non-trivial imperative language with concurrency features generating a given (dedicated) instruction set.

9. Cyber-Pysical Systems (201500053)

The Cyber-Physical Systems module is an indivisible module worth 15 credits.

Figure 9. Assessment Table Cyber-Physical Systems

Module parts	Name Assessment	Type of Assessment	Individual / Group	Weight assessment (%)	Minumum grade	Weight result assessment (%)
I	Formal specification and hybrid systems	Take home toets	I/G	100	5.5*	14
	Dependable systems and networks	Take home toets	I/G	50	5.5*	14
		Presentation	I/G	50		
	Sensor and actuator systems	Written test	I	50	5.5*	14
		Mini-practica	G	50		
	Real-time operating systems	Written test	I	50	5.5*	14
		Practica rapport	G	50		
	Physical-systems modeling and controller design	6 verslagen over assignments en practicumverslagen	G	16,66 each	5.5*	14
				Weighted sub-average	**5.5***	
	Project	Project artefacts	G	25	5.5	30
		Presentaties	G	25		
		6-pagina's onderzoekspaper	G	25		
		Youtube filmpje	G	25		
				Weighted average	**5.5**	

* Out of the marked (*) module component grades ONE mark lower than 5.5, but at least 5.0 (5.0 = <rate <5.5) is allowed IF it's sub-weighted average is 5.5 or higher.

9.1 Learning goals module Cyber-Physical Systems

The learning goals for the various mini-tracks are as follows.

Formal specification and hybrid systems:
After this module the student is able to:

- understand the fundamentals of Timed Automata (TA)
- understand the use of model checking in the tool UPPAAL
- use TA and UPPAAL in the analysis and design of real-time systems
- understand the fundamentals of Statistical Model Checking (SMC)
- use UPPAAL SMC for analyzing simple hybrid systems

Sensor and actuator systems
After this module the student is able to:

- Describe and explain main design principles behind WSN systems, protocols and algorithms.
- Describe and explain mechanisms to be used to achieve distributed and self-organizing capabilities at various layers of an WSN architecture.
- Describe in detail how some well-established WSN systems, protocols and algorithms function, and describe their strengths and weaknesses.
- Use critical thinking skills to develop alternative strategies for solving WSN problems.

Physical-Systems Modeling and Controller Design (for CS).
After successful completion, the student knows essentials of:

- modeling of the dynamic behavior of physical systems, using a Domain-Specific Modeling Formalism;
- the design of loop controllers for these physical systems, to adapt / influence the dynamic behavior of those;
- testing and implementing these loop controllers on processors running hard real-time operating systems;
- and can apply the above to basic CPSes, using modern, model-based tools.

Dependable system and network design and evaluation:
After this module, the student is able to:

- Explain the concepts of dependability and the basic principles of dependable system design;
- Explain and apply simple redundancy mechanisms to improve system dependability;
- Explain and apply basic techniques to evaluate system dependability, such as reliability block diagrams, fault-trees and Markov chains;
- Explain and apply the basic principles of highly-dependable storage systems;
- Discuss key developments (from an historical perspective) in fault-tolerant system design;
- Use state-of-the-art tools for evaluation system dependability (for example Möbius or PRISM).

Real-time operating systems:
After successful completion, the student can:

- Model and analyze simple real-time systems using data flow techniques
- Apply these in the context of a small lab set-up (using raspberry Pi's)

Project part.

- Propose and defend a plan for a simple cyber-physical system
- Apply different modeling and analysis techniques (as learned in the first 6 weeks of the module) to support design decisions
- Design, implement and document a simple cyber-physical system
- Report about it in writing (short paper), presenting (powerpoint and video)

10. Smart Spaces (201500057)

The Smart Spaces module is an indivisible module worth 15 credits.

Figure 10. Assessment Table Smart Spaces

Module parts	Name Assessment	Type of Assessment	Individual / Group	Weight assessment (%)	Minumum grade	Weight result assessment (%)
I	Smart Spaces	2*Written test	I	100	5.5*	20
		4*Challenges	G	100	5.5	30
		Integrated project	G	100	5.5	50
		Peer-review	I	0	Pass	0
				Weighted average	5.5	

* 5.5 is minimum mark for each challenge but the average of 4 challenges should be at least 6

10.1 Learning goals module Smart Spaces

In this module students will learn the principles, concepts and techniques required to create and evaluate smart spaces. After the module, students are able to:

- explain and characterize principles of smart spaces and underlying methods and technologies
- develop creative, useful, and efficient smart space solutions and services
- explain basics of context awareness, service design and engineering
- explain and design (distributed) algorithms for context awareness, reasoning, and recognition
- design smart interaction methods based on context
- design solutions for and using technologies for user interaction, localization and other areas similarly relevant for smart spaces
- structurally evaluate and analysis of complex interactive smart spaces
- project planning and management
- perform fair self-assessment and reflection based on peer reviewing

11. Web Science (201500025)

The Web Science module is an indivisible module worth 15 credits.

Figure 11. Assessment Table Web Sciece

Module parts	Name Assessment	Type of Assessment	Individual / Group	Weight assessment (%)	Minumum grade	Weight result assessment (%)
I	Web Science	2x written test	I	50 each	5.5	50
	Implementation Projects	5x Rapport	G	100*	5.5	50
		Presentation	G	PASS		
				Weighted average	**5.5**	

*weight 2wk project is 2x weight 1wk project

11.1 Learning goals module Web Science

Learning goal is to be able to recognize and explain network phenomena. Social networks such as Facebook, information networks such as the Web, institutions such as voting are all IT-enabled. The student will learn:

- how to recognize and explain structural and dynamic phenomena in these networks, such as cascading behavior and power laws, and
- how to model and analyze using graph theory and game theory.

After following this module, the student is able to:

- Recognize these phenomena in practice;
- Apply mathematical models from graph theory, probability, and game theory to describe and analyze them;
- Explain and predict network phenomena in terms of network structure and behavior;
- Operationalize and apply these models to existing network data.

12. Design Project (201500121)

The Design Project module is an indivisible module worth 15 credits.

Figure 12. Assessment Table Design Project

Module parts	Name Assessment	Type of Assessment	Individual / Group	Weight assessment (%)	Minumum grade	Weight result assessment (%)
I	Design Project	Project examination	G	100	5.5	67
	Reflection Component	Assignments	G	100	5.5	33
				Weighted average	**5.5**	

12.1 Learning goals module Design Project

Learning objectives Design Component: the student should be able to

- Collect functional and quality requirements in cooperation with a client, and prioritize them
- Methodically design a system that meets the requirements, using relevant knowledge, techniques and tools
- Turn the design into a working prototype
- Formulate a testplan according to which the prototype is tested
- Document all phases in the design trajectory
- Justify choices and coordinate them with the client
- Write a project proposal and a project plan, and organize the project accordingly
- Work in a team: plan activities, distribute responsibilities, interact in a constructive way
- Indicate consequences of system and design choices (ethical, societal, organizational)
- Indicate follow up steps for the development of the system

The learning objectives Reflection Component: to improve students'

- ability to apply critical reasoning and ethical deliberation in order to assess and anticipate the impact of design choices.
- ability to understand how Information technology design may affect core moral values, user well-being, and societal change
- knowledge of the most fundamental discussions, theories and controversies in computer ethics, in particular related to the design of information technologies.
- knowledge of professional codes of ethics and Dutch/European legislation related to computer science.

13. Research Project (201500120)

The Research Project module is an indivisible module worth 15 credits.

Figure 13. Assessment Table Research Project

Module parts	Name Assessment	Type of Assessment	Individual / Group	Weight assessment (%)	Minumum grade	Weight result assessment (%)
I	Research Project	Project toetsing	I	100	5.5	67
	Reflection component	Assignments	I	100	5.5	33
				Weighted average	**5.5**	

13.1 Learning goals module Research Project

Learning goals Research component

- Onderzoek van beperkte omvang uitvoeren en daar een paper over schrijven. Dit leerdoel heeft subdoelen:
  - Wetenschappelijke literatuur zoeken, lezen en beoordelen op kwaliteit en relevantie voor het eigen uit te voeren onderzoek.
  - Een beargumenteerde keuze maken voor een onderzoeksmethode.
  - Een research proposal schrijven volgens een gegeven template.
  - Een paper volgens wetenschappelijke maatstaven schrijven en daarbij gebruik te maken van een gegeven template.
- Een review schrijven van research proposals van collega-studenten.
- Een review schrijven van papers van collega-studenten.
- Een presentation over het uitgevoerde onderzoek geven.

Learning goals Reflectiecomponent: De Learning goals zijn om het volgende te verbeteren

- het vermogen kritisch te redenen en het maken van ethische overwegingen toe te passen op informatica-onderzoek
- het vermogen te begrijpen hoe informatica onderzoek van invloed is op morele waarden, welzijn van gebruikers en maatschappelijke verandering
- kennis van fundamentele discussies, theorieën en controverses in computerethiek, in het bijzonder gerelateerd aan informaticaonderwijs
- kennis van professionele codes van ethiek en Nederlands/Europese wetgeving gerelateerd aan informatica-onderzoek.