Formal Methods

Course description sheet

Basic information

Field of study Computer Science **Major** Systems Modelling and Intelligent Data Analysis **Organisational unit** Faculty of Electrical Engineering, Automatics, Computer Science and Biomedical Engineering **Study level** Second-cycle (engineer) programme **Form of study** Full-time studies **Profile** General academic	**Didactic cycle** 2020/2021 **Course code** EAIiIBINFMSS.IIi1O.c16b27e40c89f8a46358b7431d6e22fd.20 **Lecture languages** english **Mandatoriness** Obligatory **Block** General Modules **Course related to scientific research** Yes

Course coordinator	Tomasz Szmuc
Lecturer	Tomasz Szmuc, Wojciech Szmuc

Period Semester 1	**Method of verification of the learning outcomes** Exam **Activities and hours** Lectures: 28 Laboratory classes: 28	**Number of ECTS credits** 4

Goals

C1	The goal of the subject is to provide knowledge and to train skills in using formal methods for modelling and verification of hybrid systems and software. Petri nets, process algebras are used for modelling and temporal logics for verification - proving of the correctness.

Course's learning outcomes

Code	Outcomes in terms of	Learning outcomes prescribed to a field of study	Methods of verification
Knowledge – Student knows and understands:			
W1	Student has a deepened knowledge of the popular formal modelling languages.	INF2A_W01, INF2A_W04, INF2A_W07	Activity during classes, Test, Examination
W2	Student has a basic knowledge of formal verification of IT systems.	INF2A_W01, INF2A_W04, INF2A_W07	Activity during classes, Test, Examination
W3	Student has a basic knowledge of software tools for modelling and verification of IT systems with formal methods.	INF2A_W01, INF2A_W04, INF2A_W07	Activity during classes, Test, Examination
Skills – Student can:			
U1	Student is able to use formal methods the the analysis and design of IT systems.	INF2A_U01, INF2A_U05	Execution of laboratory classes, Test, Examination, Completion of laboratory classes
U2	Student is able to use software tools supporting the formal modelling and verification	INF2A_U01, INF2A_U05	Execution of laboratory classes, Test, Examination, Completion of laboratory classes
Social competences – Student is ready to:			
K1	Student understands understands the need for providing the public with information and opinions on the role of formal methods and their impact on the software quality.	INF2A_K02	Test, Examination

Program content ensuring the achievement of the learning outcomes prescribed to the module

The aim the subject to provide knowledge and train skills in using formal methods for modelling and verification of systems/software. Petri nets, process algebras and temporal logics are used.

Student workload

Activity form	Average amount of hours* needed to complete each activity form
Lectures	28
Laboratory classes	28
Preparation for classes	24
Realization of independently performed tasks	20
Examination or final test/colloquium	2
Preparation of project, presentation, essay, report	10

Student workload	Hours 112
Workload involving teacher	Hours 56

* hour means 45 minutes

Program content

No.	Program content	Course's learning outcomes	Activities
1.	1. Formal methods in computer science: formal modelling, types of formal languages, application of formal methods. 2. Formal methods in software engineering: the role of formal methods, verification, validation, norms. 3. Petri nets: place and transition nets (definition, properties, analysis methods), coloured Petri nets, hierarchical Petri nets. Modelling of concurrent systems with Petri nets. 4. Process algebras: CCS (Calculus of Communicating Systems), equivalence relations and their applications, LOTOS language, modelling with LOTOS. 5. Modelling time properties in process algebras. 6. Specification of models properties with temporal logics (LTL, CTL).	W1, W2, W3, K1	Lectures
2.	Practical classes using software modelling tool for formal modelling and verification. 1. Place-Transition Petri nets - TINA 2. High-Level Coloured Petri nets - CPN Tools 3. Process algebras - CADP software.	U1, U2, K1	Laboratory classes

Extended information/Additional elements

Teaching methods and techniques:

Lectures, Case study, Group work, Design thinking, Problem based learning

Activities	Methods of verification	Credit conditions
Lectures	Test, Examination	Final positive grade of the laboratory classes and positive grade received from exam are required to complete the subject.
Lab. classes	Activity during classes, Execution of laboratory classes, Completion of laboratory classes	Positive grades of all laboratory tasks.

Conditions and the manner of completing each form of classes, including the rules of making retakes, as well as the conditions for admission to the exam

Presence on laboratory classes is obligatory according to the general study regulation. If the requirement is not met the chief lecturer may specify additional conditions for the positive final grade or issue the negative grade (negative assessment). The student can take the exam only after obtaining a positive assessment of laboratory classes.

Method of determining the final grade

Course grade: The course grade is the average of laboratory and exam grades. In case of a correction exam, grades of all terms are taken into consideration. Laboratory classes: The final grade will be awarded based on the result of two tests tests. The credit will be based on a verification of both the theoretic knowledge and abilities, and the more practical skills of students. The activity during laboratory classes will be also taken into consideration as an additional evaluation criterion.

Manner and mode of making up for the backlog caused by a student justified absence from classes

All absences without leave may be made up by an individual agreement with the class instructor. The preferable form is a discussion on the topic, which should be made up by the student.

Prerequisites and additional requirements

Basic knowledge of:

1. Discrete mathematics, in particular relations and graphs.
2. Finite automata theory.
3. Propositional logic, first-order logic.

Rules of participation in given classes, indicating whether student presence at the lecture is obligatory

Lectures:

1. Attendance: Not obligatory.
2. Students are encouraged for active participation, asking questions and discussion. Video recording requires permission from the lecturer.

Laboratory classes:

1. Attendance: obligatory.
2. Students perform tasks according to guidance and tutorials delivered by teaching assistant. Interim verifications of knowledge by colloquiums are applied.

Literature

Obligatory

1. Murata, T.: Petri Nets: Properties, Analysis and Applications, Proceedings of the IEEE, vol. 77, no 4, pp. 541-580, 1989
2. Jensen K, Kristensen L.: Coloured Petri nets. Modelling and Validation of Concurrent Systems. Springer: Heidelberg, 2009.
3. Turner K.J.: The formal specification language LOTOS. A course for users.
4. Behrmann G., David A., Larsen K.G.: A tutorial on UPPALL 4.0

Optional

1. Aceto L., Ingofsdottir A., Larsen K.G., Srba, J.: Reactive Systems: Modelling, Specification and Verification. Cambridge University Press, Cambridge, UK, 2007.

Scientific research and publications

Research

1. Rigorous Development of Cyber-Physical Systems (CPS) supported by mathematical modelling and formal verification.

Publications

1. Szmuc T., Szpyrka M.: Formal Methods - Support or Scientific Decoration in Software Development. Proceedings of the 22nd International Conference "Mixed Design of Integrated Circuits and Systems". June 27, 2015, Toruń, Poland. pp. 24-32 (Invited Paper)
2. Szmuc W., Szmuc T.: Modelling UML Object Event Handling with Petri Nets. Towards improvement of embedded systems analysis and design. Proceedings of the 23rd International Conference "Mixed Design of Integrated Circuits and Systems". June 23-25, 2016 Łódź. pp.454-457
3. Szmuc W., Szmuc T.: Modeling UML object event handling with Petri nets. Towards improvement of embedded systems analysis and design. Proceedings of the 23rd International Conference “Mixed Design of Integrated Circuits and Systems” MIXDES 2016, ed. Andrzej Napieralski. — Łódź : Lodz University of Technology. Department of Microelectronics and Computer Science, cop. 2016. — S. 4. — Full text on CD-ROM — ISBN 978-83-63578-08-4 — pp. 454-457. — Wymagania systemowe: Adobe Reader ; napęd CD-ROM. — Bibliogr. s. 457, Abstr.
4. Mrówka R., Szmuc T.: UML Statecharts compositional semantics in LOTOS. In: ISPDC 2008 Proceedings of the 7th International Symposium on Parallel and Distributed Computing : 1–5 July 2008, Krakow, Poland / eds. Marek Tudruj. — Los Alamitos, California; Washington; Tokyo : CPS Conference Publishing Services, IEEE Computer Society, cop. 2008. — ISBN 978-0-7695-3472-5. — S. 459–463. — Bibliogr. s. 463, Abstr
5. Samolej S., Szmuc T.: HTCPNs–based Modelling and Evaluation of Dynamic Computer Cluster Reconfiguration. Accepted, in the Proceedings of the Forth IFIP TC Central and East Europe Conference on Software Engineering Techniques, CEE-SET 2009 – also in Lecture Notes in Computer Science, vol. 7054, 2011, pp. 131-142Szmuc W., Szmuc T.: Towards Embedded Systems Formal Verification. Translation from SysML into Petri Nets. Proceedings of the 25th International Conference "Mixed Design of Integrated Circuits and Systems". June 21-23, 2018. Gdynia. pp. 420-423
6. Szmuc W., Szmuc T.: Towards Embedded Systems Formal Verification. Translation from SysML into Petri Nets.Proceedings of the 25th International Conference "Mixed Design of Integrated Circuits and Systems". June 21-23, 2018. Gdynia. pp. 420-423

Learning outcomes prescribed to a field of study

Code	Content
INF2A_K02	ma świadomość roli społecznej absolwenta uczelni technicznej, rozumie potrzebę formułowania i przekazywania społeczeństwu informacji i opinii dotyczących osiągnięć informatyki, wagi profesjonalnego zachowania i przestrzegania zasad etyki zawodowej, prawidłowo identyfikuje i rozstrzyga dylematy związane z wykonywaniem zawodu
INF2A_U01	potrafi pozyskiwać informacje z literatury, baz danych i innych źródeł, integrować uzyskane informacje, dokonywać ich interpretacji i krytycznej oceny, wyciągać wnioski oraz formułować i wyczerpująco uzasadniać opinie, a także określić kierunki dalszego uczenia się i realizować proces samokształcenia
INF2A_U05	potrafi wykorzystać poznane metody i modele do tworzenia różnego rodzaju programów o charakterze użytkowym i naukowym, z uwzględnieniem specyfiki specjalności
INF2A_W01	ma pogłębioną wiedzę w zakresie przedmiotów ścisłych, pozwalającą na formułowanie i rozwiązywanie złożonych zadań z zakresu informatyki
INF2A_W04	ma podbudowaną teoretycznie wiedzę w zakresie inżynierii oprogramowania z uwzględnieniem specyfiki specjalności, w szczególności w zakresie budowy narzędzi i systemów informatycznych, etapów i metod projektowania, rozwoju i analizy oprogramowania, oraz stosowanych modeli procesu wytwarzania oprogramowania z zakresu specjalności
INF2A_W07	orientuje się w obecnym stanie oraz najnowszych osiągnięciach i trendach rozwojowych informatyki i dziedzin pokrewnych oraz ma wiedzę niezbędną do rozumienia pozatechnicznych uwarunkowań działalności inżynierskiej