Poznan University of Technology
Faculty of Electrical Engineering

European **C**redit **T**ransfer **S**ystem

STUDY MODULE DESCRIPTION FORM

Name of the module/subject **Distributed programming**	Code **1010335511010335196**

Field of study **Information Engineering**	Profile of study (general academic, practical) **(brak)**	Year /Semester **1 / 1**
Elective path/specialty **-**	Subject offered in: **Polish**	Course (compulsory, elective) **obligatory**

Cycle of study: **Second-cycle studies**	Form of study (full-time,part-time) **part-time**

No. of hours Lecture: **16** Classes: **-** Laboratory: **8** Project/seminars: **8**	No. of credits **6**

Status of the course in the study program (Basic, major, other) **(brak)**	(university-wide, from another field) **(brak)**

Education areas and fields of science and art	ECTS distribution (number and %**)**
technical sciences	**6 100%**
Technical sciences	**6 100%**

Responsible for subject / lecturer:

Ph.D. Eng. Adam Meissner
email: Adam.Meissner@put.poznan.pl
tel. 61 665 37 24
Faculty of Electrical Engineering
ul. Piotrowo 3A 60-965 Poznań

Responsible for subject / lecturer:

Ph.D. Eng. Krzysztof Zwierzyński
email: Krzysztof.Zwierzynski@put.poznan.pl
tel. 61 665 37 55
Faculty of Electrical Engineering
ul. Piotrowo 3A 60-965 Poznań

Prerequisites in terms of knowledge, skills and social competencies:

1	**Knowledge**	Student has theoretical and practical knowledge on algorithm design and analysis, on abstract data structures and their implementation and on computationally hard problems; he/she has theoretical and practical knowledge on computer system architectures, on operating systems and on popular information engineering technologies.
2	**Skills**	Student is able to design algorithms using basic algorithmic techniques and analyse the algorithm complexity; he/she knows how to apply programming environments and platforms to develop, execute and test simple programs implemented in imperative, object-oriented and declarative languages.
3	**Social competencies**	Student understands the need of permanent learning and improving the professional, personal and social competencies; a student realises the responsibility for his/her work done individually or in a team; he/she is also ready to accept the rules of group work.

Assumptions and objectives of the course:

providing students with basic models of distributed programs and with general methods of communication and synchronization in programs of this type; presentation of selected problems in design of distributed programs.

Study outcomes and reference to the educational results for a field of study

Knowledge:

1. Student has theoretical and practical knowledge on algorithm design and analysis, on abstract data structures and their implementation and on computationally hard problems - [K_W04]

2. Student has theoretical and practical knowledge on network technologies - [K_W07]

3. Student has theoretical and practical knowledge on internet technologies - [K_W11]

Skills:

1. Student is able to work individually and in a team; he/she can estimate a time for the given task and construct a schedule for it - [K_U02]

2. Student is able to plan and perform experiments and to apply mathematical methods and models in order to test, analyse and evaluate information systems and their parts - [K_U07]

3. Student is able to analyse a functioning of a computer system and also a functioning of operating systems and computer networks or their parts - [K_U11]

Social competencies:

1. Student understands the need of permanent learning and improving the professional, personal and social competencies - [K_K01]

2. Student understands the importance of a thorough design of a given project, respecting notation standards, using a proper language and keeping deadlines - [K_K07]

Assessment methods of study outcomes

Lecture. Written exam consisting of theoretical questions and simple problems to solve.

Labs. Oral or written tests for preparation of a student to exercises, rating a student's activity during exercises, evaluation of reports including their punctual delivery.

Project. Keeping all milestone deadlines of the project; evaluation of the final report.

More than 50% points are necessary for passing the exam, project and labs.

Course description

Lecture. Distributed programming vs. parallel programming, a distributed model of a parallel program, network transparency, client-server model, MPI standard, Open CL environment, synchronisation of threads and processes, efficiency measures of distributed systems, design of distributed algorithms, elements of programming in the client-server model, problems of security and fault-tolerance in distributed systems, distributed programming in the Erlang language.

Course update 2017: distributed programming in the Erlang language.

Labs. Distributed programming using the MPI standard and the GPGPU technology. Task queuing in supercomputer systems (optional).

Project. The project illustrates capabilities of distributed programming of a given software or hardware platform.

Teaching methods:

- lectures supported by slides and examples presented on the table
- laboratories - writing programs by individual students and running them in distributed hardware environments also accessible from home, performing computational experiments
- projects - work in a team, multimedia presentation of the work progres by the team, discussion of proposed solutions, reviewing project documentation.

Basic bibliography:

1. Concepts, Techniques, and Models of Computer Programming, Roy P. van, Haridi S., MIT Press, Cambridge, 2004
2. Learn You Some Erlang for Great Good! A beginer?s Guide, Herbert F., http://learnyousomeerlang.com/
3. Programowanie współbieżne i rozproszone, Weiss Z., Gruźlewski T., WNT, Warszawa, 1993
4. Systemy rozproszone. Zasady i paradygmaty, Tanenbaum A.S., Steen M. van, WNT, Warszawa, 2006

Additional bibliography:

1. A User's Guide to MPI, Pacheco P.S., http://www.wellesley.edu/CS/courses/CS331/notes/mpi.guide.pdf
2. Ericcson AB, Erlang/OTP System Documentation, http://erlang.org/doc/pdf/otp-systemdocumentation.pdf
3. Introduction to Parallel Computing, Barney B., https://computing.llnl.gov/tutorials/parallel_comp/
4. Sztuka programowania wieloprocesorowego, Herlihy M.,Shavit N., PWN, Warszawa, 2008

Result of average student's workload

Activity	Time (working hours)
1. Lectures	16
2. Labs	8
3. Project	8
4. Consultations and the exam	5
5. Preparation to labs, preparing the reports	15
6. Design of the project	15
7. Preparation to the exam	30

Student’s workload

Source of workload	hours	ECTS
Total workload	97	6
Contact hours	37	3
Practical activities	46	3