

# Concurrent and distributed programming - opis przedmiotu

Informacje ogólne	
Nazwa przedmiotu	Concurrent and distributed programming
Kod przedmiotu	11.3-WE-INFP-CaDP-Er
Wydział	Wydział Nauk Inżynieryjno-Technicznych
Kierunek	Informatyka
Profil	ogólnoakademicki
Rodzaj studiów	Program Erasmus pierwszego stopnia
Semestr rozpoczęcia	semestr zimowy 2021/2022

Informacje o przedmiocie	
Semestr	4
Liczba punktów ECTS do zdobycia	6
Typ przedmiotu	obowiązkowy
Język nauczania	angielski
Sylabus opracował	

Formy zajęć					
Forma zajęć	Liczba godzin w semestrze (stacjonarne)	Liczba godzin w tygodniu (stacjonarne)	Liczba godzin w semestrze (niestacjonarne)	Liczba godzin w tygodniu (niestacjonarne)	Forma zaliczenia
Wykład	30	2	-	-	Egzamin
Laboratorium	30	2	-	-	Zaliczenie na ocenę

## Cel przedmiotu

- Familiarize students with basic techniques of concurrent programming
- Familiarize students with basic techniques of distributed programming
- Teach students the fundamental skills of using concurrent and distributed programming techniques
- Learn basic skills in developing software used balanced and distributed architecture

## Wymagania wstępne

Principles of programming, Java programming, Computer architectures I and II.

## Zakres tematyczny

**Concurrent programming** – basic concept: process, shared resources, critical section, mutual exclusion, synchronization, deadlock, starvation.

Aims of concurrent programming. Advantages and disadvantages of concurrent programming.

Semaphores: general semaphore, binary semaphore, synchronization of processes with usage of semaphores.

Concurrent programming in Java. Monitors. Additional methods of threads synchronization: blocking queued, barriers, countdown of latch and exchanger.

Classical problems of concurrent programming: dining philosophers problem, producer-consumer problem, readers-writers problems.

Characterization of **Distributed Systems**. Inter-process communication. Guidelines for design of inter-process communication.

Remote procedure call (RPC). Remote method invocation (RMI). How to build of distributed applications in Java RMI. Integration different distributed environments.

Time and coordination in distributed systems. Logical clock. Election algorithm. Transactions and concurrency control in distributed systems. Algorithms for deadlock detection in distributed systems.

## Metody kształcenia

Lecture: conventional lecture

Laboratory: laboratory exercises, group work

## Efekty uczenia się i metody weryfikacji osiągnięcia efektów uczenia się

Opis efektu	Symbole efektów	Metody weryfikacji	Forma zajęć
Can describe the mechanism of communication layer design and the issues connected with data exchange in distributed systems		<ul style="list-style-type: none"><li>• sprawdzian</li><li>• test</li></ul>	<ul style="list-style-type: none"><li>• Laboratorium</li></ul>
Can distinguish basic architectural models used for the design of distributed systems		<ul style="list-style-type: none"><li>• egzamin - ustny, opisowy, testowy i inne</li></ul>	<ul style="list-style-type: none"><li>• Wykład</li></ul>

Opis efektu	Symbole efektów Metody weryfikacji	Forma zajęć
Is aware of the need to use distributed systems and programs	<ul style="list-style-type: none"> <li>egzamin - ustny, opisowy, testowy i inne</li> </ul>	<ul style="list-style-type: none"> <li>Wykład</li> </ul>
Can explain the need for application of concurrent programming	<ul style="list-style-type: none"> <li>egzamin - ustny, opisowy, testowy i inne</li> </ul>	<ul style="list-style-type: none"> <li>Wykład</li> </ul>
Can design and create object-oriented software employing concurrent and distributed programming mechanisms	<ul style="list-style-type: none"> <li>bieżąca kontrola na zajęciach</li> </ul>	<ul style="list-style-type: none"> <li>Laboratorium</li> </ul>
Can explain the mechanisms of coordination of activities in distributed systems	<ul style="list-style-type: none"> <li>sprawdzian</li> <li>test</li> </ul>	<ul style="list-style-type: none"> <li>Laboratorium</li> </ul>

## Warunki zaliczenia

Lecture - obtaining a positive grade in written exam.

Laboratory - the main condition to get a pass are sufficient marks for all exercises and tests conducted during the semester.

Calculation of the final grade: = lecture 50% + laboratory 50%.

## Literatura podstawowa

1. Ben-Ari M.: Principles of Concurrent and Distributed Programming, Addison-Wesley, 2006.
2. Foster I.: Designing and Building Parallel Programs, (2020) <https://www.mcs.anl.gov/~itf/dbpp/>
3. Coulouris G. et al.: Distributed Systems. Concepts and Design, Addison Wesley, 2005.
4. Tanenbaum S., Maarten van Steen: Distributed Systems. Principles and Paradigms, Prentice Hall, 2016 (2020) - Distributed systems <https://www.distributed-systems.net/index.php/books/ds3/ds3-sneak-preview/>
5. Garg V. K.: Concurrent and Distributed Computing in Java. Wiley-IEEE Press, 2004.
6. Cay S. Horstmann, Gary Cornell: Core Java, Vol. 1: Fundamentals, Prentice Hall PTR, 2018
7. Cay S. Horstmann: Core Java, Vol. 2: Advanced Features, Prentice Hall PTR, 2019
8. Kathy Sierra, Bert Bates: Head First Java, 2nd Edition, O'Reilly Media, 2009
9. Goetz B., Peierls T., Bloch J., Bowbeer J., Holmes D., Lea D.: Java Concurrency in Practice, Addison-Wesley Professional, 2006.
10. Burns B.: Designing Distributed Systems: Patterns and Paradigms for Scalable, Reliable Services, O'Reilly, 2018

## Literatura uzupełniająca

1. Roger Wattenhofer: Principles of Distributed Computing, Spring 2016, (2020) [https://disco.ethz.ch/courses/podc\\_allstars/lecture/podc.pdf](https://disco.ethz.ch/courses/podc_allstars/lecture/podc.pdf)
2. Distributed Systems for fun and profit (2020) <http://book.mixu.net/distsys/single-page.html>

## Uwagi

Zmodyfikowane przez dr inż. Tomasz Gratkowski (ostatnia modyfikacja: 21-07-2021 10:03)

Wygenerowano automatycznie z systemu SylabUZ