DISTRIBUTED SYSTEMS AND CLOUD COMPUTING

COURSE OUTLINE

Responsible: Charalampos Papadakis

1. GENERAL			
SCHOOL	School of Engineering		
ACADEMIC UNIT	Department of Electrical and Computer Engineering		
LEVEL OF STUDIES	Undergraduate		
COURSE CODE	0811.8.027.0	**SEMESTER**	2nd
COURSE TITLE	Distributed Systems and Cloud Computing		

INDEPENDENT TEACHING ACTIVITIES if credits are awarded for separate components of the course	**WEEKLY TEACHING HOURS**	**CREDITS**
	5	4
Total	**5**	**4**

COURSE TYPE general background, special background, specialised general knowledge, skills development	Special background / Core
PREREQUISITE COURSES	None
LANGUAGE OF INSTRUCTION and EXAMINATIONS	English
OFFERED TO ERASMUS STUDENTS	Yes (in English) — Spring Semester
COURSE WEBSITE (URL)	https://eclass.hmu.gr/courses/ECE196/

2. LEARNING OUTCOMES

Learning outcomes

The aim of the course is to provide the necessary knowledge to understand the specifics involved in the design of Distributed Algorithms and Systems. The course will cover theoretical and practical issues of Distributed Systems in terms of their requirements, design approaches and available technologies. Upon successful completion of the course, the student will be able to:

Know the basic principles governing the operation of Distributed Systems.

Understand the structure, capabilities and limitations of Cloud Systems.

Design Distributed Algorithms to solve practical problems.

Evaluate the performance of Distributed Algorithms.

Propose solutions for the development of integrated Distributed Systems.

General Competences

Search, analysis and synthesis of data and information, using the necessary technologies

Adaptation to new situations

Autonomous work

Work in an interdisciplinary environment

Generation of new research ideas

Promotion of free, creative and inductive thinking

3. SYLLABUS
Lectures 1. Introduction 1.1. What is a distributed system. 1.2. Basic characteristics of distributed systems (Segmentation, Heterogeneity, Errors, Trust, Synchronization) 1.3. Categories of Distributed Systems 1.4. Distributed Computing Systems Distributed Information Systems Distributed Pervasive Systems 1.5. Comparison with related systems 1.6. Parallel Processing, Cluster Computing, Grid Computing 1.7. Basic Design Principles - Objectives 1.8. Transparency (of access, location, migration, migration, replication, concurrency, failure). Open mode. System composition from heterogeneous pieces of software and hardware from different sources. It includes the concepts of interoperability, portability, flexibility. Scalability – Scaling (Size, geographic, management). Techniques for achieving scaling (hiding delays, distribution, etc.), reliability. 1.9. Concepts of Distributed Operating Systems 1.10. Distributed Operating Systems (tightly coupled systems). Multiprocessor and Multicomputer Distributed Operating Systems. Networked Operating Systems (loosely coupled systems). Heterogeneity, reduced transparency. Conceptual transition to the concept of Middleware. 1.11. Middleware 1.12. Definition – Examples of middleware services (high-level communication, remoting, names, files, objects, distributed transactions, replication, security, process synchronization, resource locking). Description of examples. Analysis of how these paradigms obey and promote the desired characteristics of Distributed Systems described in the Introduction. 2. Models of Distributed Systems 2.1. Communication Models 2.2. Client/Server Model. Description, request/response protocol, server classes, model advantages and disadvantages. Three-Tier Client/Server Model Description - Layered Architectures. Model of Peer-to-Peer Systems. Basic concepts, model advantages and disadvantages, comparison with the Client/Server model, Examples of Peer-to-Peer Systems. How they achieve the goals of Distributed Systems. 2.3. Theoretical Models of Distributed Systems Modern Systems. Processes, states, execution steps, calculation and receipt events, security and liveness conditions. Asynchronous systems. Model description, clocks and delivery events not synchronized. 3. Basic Distributed Algorithms 3.1. Introduction to Distributed Algorithms. 3.2. Evaluation methods (time complexity and number of messages) 3.3. Message Propagation in a Spanning Tree. 3.4. Description and analysis of algorithm complexity in synchronous and asynchronous systems. 3.5. Collection of information in a Spanning Tree. 3.6. Description and analysis of algorithm complexity in synchronous and asynchronous systems. 3.7. Creating a Spanning Tree. 3.8. Description and complexity analysis of a Spanning Tree structure creation algorithm in synchronous and asynchronous systems with the flooding method. 3.9. Chief Selection in Distributed Systems. 3.10. Problem description. Description and analysis of named, uniform and non-uniform algorithms for Leader Selection in synchronous and asynchronous systems (use of ring topology, general topology, etc.). 4. Causality 4.1. Introduction to the concept of causality. 4.2. Definition of the happens-before relation. Definition of the concept of casual shuffles. 4.3. Logical Clocks. 4.4. Introduction and description of the concept of logic clocks. Definition, advantages and disadvantages. 4.5. Vector clocks. 4.6. Introduction and description of the concept of vector clocks. Their definition and function. Usage examples and conceptual transition to the message layout concept. 5. Arrangement of Messages 5.1. Introduction to the concept of message layout. 5.2. Description of reliable message propagation and requirements (Integrity, Validity, Agreement). 5.3. FIFO order of messages. 5.4. Description of the concept. Description and analysis of a layout FIFO algorithm using a message disbursement queue. 5.5. Causal arrangement of messages. 5.6. Description of the concept. Description and analysis of a causal message ordering algorithm using vector clocks. 5.7. Ultimate message layout. 5.8. Description of the concept. Description and analysis of a causal message ordering algorithm using Successor. Description and analysis of a causal message ordering algorithm using timestamps and a message buffer. 6. Fault Tolerance Algorithms 6.1. Introduction to Fault tolerance. 6.2. Analysis of fault-tolerant algorithms in the context of the Consensus problem. Introduction to error types. Description of it

4. TEACHING and LEARNING METHODS - EVALUATION	
DELIVERY Face-to-face, Distance learning, etc.	Face to Face in classroom
USE OF INFORMATION AND COMMUNICATIONS TECHNOLOGY Use of ICT in teaching, laboratory education, communication with students	Use of ICT in laboratory training Use of ICT in communication with students through the electronic platform e-class
TEACHING METHODS The manner and methods of teaching are described in detail.	(see table below)
STUDENT PERFORMANCE EVALUATION Description of the evaluation procedure	1. Written exam with Short Answer Questions (60%) 2. Oral exam of laboratory exercises (40%) The evaluation criteria are announced to students at the beginning of the semester and are posted on the course website in eClass.

Activity	**Semester workload**
Lectures	52
Εργαστήριο	13
Independent study	52
Laboratory Exercises	33
Course total	**150**

5. ATTACHED BIBLIOGRAPHY

- Distributed Systems: Principles and Paradigms, Andrew S. Tanenbaum, Maarten van Steen, Publisher: Prentice Hall (2003), ISBN: 0131217860
- Journal of Parallel and Distributed Computing - Elsevier
- IEEE Transactions on Parallel and Distributed Systems - IEEE
- Journal of Parallel and Distributed Computing - ACM