DISTRIBUTED ADAPTIVE SYSTEMS (EMaCS-02-01)				
DEGREE PROGRAM:		Master in Computer Science for the Human-Centric and Sustainable Industry		
SEMESTER: Second	**TYPE:** Elective	**CREDITS:** 5 ECTS	**WORKLOAD:** 125 hours	**MENTORING:** 1 hours/week
LANGUAGE: English				

OBJECTIVES	
General	The module focuses on the modelling and implementation of adaptive features in distributed software systems composed of autonomous components.
Specific	• Autonomy, Nonlinearity, Equilibrium points, Synergy. • Game theory and network effects, Market mechanisms, Auctions, Voting systems. • Adaptation, Self-organization, Evolutionary algorithms, Meta-heuristics, Swarm intelligence. • Software agents, Agent architectures, and Multi-agent systems. • Distributed knowledge and distributed planning. • Frameworks for the development and simulation of multi-agent systems. • Additional topics based on current relevance.

SUSTAINABILITY

The Distributed Adaptive Systems course significantly contributes to sustainability by focusing on the modelling and implementation of adaptive features in distributed software systems composed of autonomous components. Students gain knowledge about various aspects of autonomy and adaptation, allowing them to design and implement systems that can dynamically respond to changing requirements. The emphasis on autonomy, nonlinearity, and equilibrium points aligns with sustainable practices as it enables systems to self-organize and evolve, reducing the need for constant manual intervention. By introducing concepts like game theory, market mechanisms, and evolutionary algorithms, the course equips students to develop applications that can adapt to complex and dynamic environments. This adaptability is crucial for sustainable software systems that need to evolve over time to meet new challenges and requirements.

RESILIENCE AND HUMAN-CENTRIC DEVELOPMENT

The Distributed Adaptive Systems course plays a vital role in fostering resilience and human-centric development by focusing on principles such as adaptation, self-organization, and swarm intelligence. Students acquire skills in creating models for autonomous and adaptive systems, enabling them to design applications and architectures that incorporate flexibility and adaptability. This emphasis on adaptability aligns with the principles of resilience, preparing students to develop systems that can withstand uncertainties and unforeseen challenges. Additionally, the course cultivates attitudes such as curiosity and openness to exploring new approaches, fostering a mindset that is crucial for addressing real-world challenges in software development and beyond. The recognition of the importance of adaptability and flexibility further contributes to human-centric development by prioritizing the needs and experiences of end-users in the face of evolving technological landscapes.

SUBJECT MATTER

-

COMPETENCES

C4. INTEGRATING AND RE-ELABORATING INFORMATION and DIGITAL CONTENT
C5. PROGRAMMING
C6. USING MACHINE LEARNING AND A.I. TECHNIQUES
C10. EXPLORATORY AND CRITICAL THINKING
C13. CREATIVELY USING DIGITAL TECHNOLOGIES
C16. WORKING WITH OTHERS
C18. COLLABORATING THROUGH DIGITAL TECHNOLOGIES

LEARNING OUTCOMES

Knowledge	• Know about various aspects of autonomy and adaptation in distributed systems. • Know about the principles of modelling and analysing autonomous adaptive systems. • Know about different types of algorithms, mechanisms, and approaches used in adaptive systems.
Skills	• Be able to create models for autonomous and adaptive systems. • Gain proficiency in analysing and evaluating the performance of adaptive systems. • Be able to design applications and architectures that incorporate flexibility and adaptability.
Attitudes/values	• Cultivate an appreciation for the significance of autonomous and adaptive technologies in modern software systems. • Develop a sense of curiosity and openness to exploring new approaches and techniques in the field of distributed adaptive systems. • Recognize the importance of adaptability and flexibility in addressing real-world challenges and problems in software development and other domains.
TEACHING METHODS	
• Seminar-style teaching methods: Work in small groups, board work, multimedia presentations, voluntary exercise tasks, academic work with publications, application-oriented work using online materials and current tools. • Practical work: Task processing in small groups with a concluding acceptance discussion, presentations, and written assignments.	
EVALUATION	
• Regular examination format: Graded presentation • Alternative examination formats: Graded oral examination or graded written exam. In cases where multiple examination formats are possible for the module, the responsible lecturer will announce the required format at the beginning of the course. Prerequisite (PVL): Successful completion of the exercise tasks.	
PRECONDITIONS	
None	
DEPARTMENT	Computer Science
LECTURERS	Zhen Ru Dai Jan Sudeikat: https://www.researchgate.net/profile/Jan-Sudeikat Thomas C. Schmidt: https://inet.haw-hamburg.de/members/schmidt Michael Köhler-Bußmeier: https://orcid.org/0000-0002-3074-4145
LITERATURE	• Michael Wooldridge. An introduction to multiagent systems. Wiley • Yoav Shoham and Kevin Leyton-Brown. Multiagent Systems Algorithmic, Game-Theoretic, and Logical Foundations. Cambridge University Press • David Easley and Jon Kleinberg. Networks, Crowds, and Markets: Reasoning About a Highly Connected World. Cambridge University Press. • Gerhard Weiss (ed.) Multiagent Systems. MIT Press.