Ficha de Unidade Curricular A3ES
Programação Concorrente
Licenciatura em Engenharia Informática e de Computadores
2024-25

---

1. Caracterização da Unidade Curricular

1.1 Designação
[1637] Programação Concorrente / Concurrent Programming

1.2 Sigla da área científica em que se insere
IC

1.3 Duração
Unidade Curricular Semestral

1.4 Horas de trabalho
162h 00m

1.5 Horas de contacto
Total: 67h 30m das quais TP: 67h 30m

1.6 ECTS
6

1.7 Observações
Unidade Curricular Obrigatória

---

2. Docente responsável [1207] Pedro Miguel Henriques Santos Félix

---

3. Docentes e respetivas cargas letivas na unidade curricular Não existem docentes definidos para esta unidade curricular

---

4. Objetivos de aprendizagem (conhecimentos, aptidões e competências a desenvolver pelos estudantes)

Os estudantes que concluírem com sucesso esta unidade curricular serão capazes de:

1. Desenvolver de forma correta programas para ambientes multi-threaded, com ênfase em plataformas como a JVM (Java Virtual Machine) e a plataforma .NET. Identificar e tratar de forma correta situações de concorrência, incluindo o desenho de sincronizadores e a utilização das garantias fornecidas pelo modelo de memória.

2. Usar corretamente modelos de programação assíncrona, nomeadamente *futures* , métodos assíncronos, corrotinas e *reactive streams* .

4. Intended learning outcomes (knowledge, skills and competences to be developed by the students)

Students who successfully complete this course unit will be able to:

1. Correctly develop application-level programs for execution in multi-threaded environments, with emphasis on managed platforms such as the JVM (Java Virtual Machine) and the .NET platform. Identify and correctly handle concurrency scenarios, including the design of custom synchronizers and the guarantees provided by memory models.

2. Correctly use the common application-level asynchronous programming models available, namely futures, asynchronous methods, coroutines, and reactive streams.

5. Conteúdos programáticos

1. Conceitos fundamentais: *thread* e comutação de contexto de execução, partilha de dados e problemas associados.

2. Caracterização do modelo de *threading* para tipos de aplicação mais comuns, nomeadamente aplicações com *user-interface* e servidores HTTP.

3. Identificação dos problemas associados à partilha de dados entre *threads* , e as técnicas para a sua resolução, nomeadamente imutabilidade, afinidade a *threads* , exclusão mútua e correta publicação de dados.

4. Coordenação entre *threads* , o conceito de sincronizador e sua implementação usando monitores.

5. O conceito de modelo de memória e a sua utilização para a correta implementação de programas concorrentes.

6. Técnicas para gestão de *threads* e *thread* pools.

7. O conceito de I/O assíncrono e a motivação para modelos de programação assíncronos. Modelos e mecanismos para programação assíncrona: *futures* , métodos assíncronos, corotinas, e *reactive streams* . Coordenação assíncrona entre *threads.*

5. Syllabus

1. Fundamental concepts: threads and execution context switching, data sharing concurrency and associated hazards.

2. Characterization of the threading model for typical application types, namely UI-based applications and HTTP servers.

3. Identification of data sharing concurrency hazards and techniques for their elimination, including immutability, thread affinity, mutual-exclusion, and proper object publication.

4. Coordination between threads, the synchronizer concept and their implementation using monitors.

5. The concept of memory model and its use to correctly implement concurrency scenarios.

6. Thread management techniques and thread pools.

7. The concept of asynchronous I/O and the motivation for asynchronous programming models. Models and mechanisms for asynchronous programming: futures, asynchronous methods, coroutines and reactive streams. Asynchronous coordination between threads.

6. Demonstração da coerência dos conteúdos programáticos com os objetivos de aprendizagem da unidade curricular

Os items (1) a (5) do programa fornecem o conhecimento necessário para o desenvolvimento de programas em contextos com múltiplas *threads* , incluindo adequada sincronização e coordenação, garantindo dessa forma o primeiro objetivo de aprendizagem.

Os items (6) e (7) do programa fornecem o conhecimento adicional para o desenvolvimento de programas assíncronos, garantindo dessa forma o segundo objetivo de aprendizagem, com especial ênfase entre a interação entre os modelos de assincronismo e *multi-threading* .

6. Evidence of the syllabus coherence with the curricular unit´s intended learning outcomes

Syllabus items (1) to (5) provide the required knowledge for the development of programs using multiple threads, including proper synchronization and coordination, ensuring the first learning outcome.

Syllabus items (6) and (7) provide the additiona knowledge required for the use of asynchronous programming modesl, ensuring the second learning outcome, with a special emphasis on the interactions between the asynchronous programming models and multi-threading.

7. Metodologias de ensino (avaliação incluída)

Ensino teórico-prático com 15 aulas de 1,5 horas e 15 aulas de 3 horas. O tempo total de trabalho do estudante é de 162 horas. As aulas teórico-práticas destinam-se à apresentação dos conceitos e técnicas e de exemplos práticos de aplicação (aprendizagem baseada em casos).

A unidade curricular usa avaliação distribuída com exame final. Cada estudante resolve três trabalhos práticos, em grupos de 2 a 3 elementos, pedagogicamente fundamentais e abordando todos os tópicos do programa da UC. Realiza-se avaliação individual escrita em exame cobrindo todos os objetivos de aprendizagem. A classificação final é a média ponderada das classificações do exame (60%) e dos trabalhos práticos e respectiva discussão de validação (40%), ambas com a classificação mínima de 9,5 valores.

7. Teaching methodologies (including assessment)

Theoretical-practical teaching, with 15 classes of 1.5 hours and 15 classes of 3 hours. The total work time of the student is 162 hours. Theoretical-practical classes are devoted to the presentation of concepts and techniques and practical examples of application (case-based learning). Each student solves three sets of practical exercises (projects), in groups of 2 to 3 elements, considered pedagogical fundamental and addressing all topics covered in the course. A written individual evaluation (exam) covering all learning objectives is carried out at the end. The final grade is the weighted average of the written exam (60%) and project work and discussion (40%), both requiring 9.5 as the minimal grade.

8. Demonstração da coerência das metodologias de ensino com os objetivos de aprendizagem da unidade curricular

As aulas teórico-práticas são utilizadas para abordar os principais conceitos e técnicas da programação concorrente, e assíncrona, e a forma como os mesmos estão presentes nos ambientes de execução tomados como referência. Através desta metodologia, os estudantes são confrontados com problemas reais e com as soluções consideradas aceitáveis. Através da resolução dos trabalhos práticos, realizadas em laboratório aberto e com apoio do docente, cada estudante exercita e consolida os conceitos e técnicas abordadas nesta UC.

8. Evidence of the teaching methodologies coherence with the curricular unit´s intended learning outcomes

Theoretical-practical classes are used to address the main concurrent programming concepts and techniques, including asynchronous programming, as well as the way these concepts and techniques are present in the considered execution environments. Using this methodology, students deal with real concurrent programming challenges and their solutions. Through practical projects, done in an open laboratory model and assisted by the teaching staff, each student practices the course's main concepts and techniques.

9. Bibliografia de consulta/existência obrigatória	B. Goetz, Java Concurrency in Practice, Addison-Wesley Professional, 2006. ISBN 9780321349606 C. Martins, Sincronização com Monitores na CLI e na Infraestrutura Java, 3ª edição, ISEL, 2009 Links to multiple public information, dependent on the adopted frameworks, and lecture notes, available at the public course repository (e.g. https://github.com/isel-leic-pc/s2324v-li41d-li41n).
10. Data de aprovação em CTC	2024-07-17
11. Data de aprovação em CP	2024-06-26