Ficha de Unidade Curricular – (Versão A3ES 2018-2023)

1. Caracterização da Unidade Curricular.

1.1. Designação da unidade curricular ***(1.000 carateres).***
Programação Concorrente / Concurrent Programming

1.2. Sigla da área científica em que se insere ***(100 carateres).***
IC

1.3. Duração[1] ***(100 carateres).***
Semestral

1.4. Horas de trabalho[2] ***(100 carateres).***
162 h

1.5. Horas de contacto[3] ***(100 carateres).***
Total – 67,5 h
TP - 67,5 h

1.6. ECTS ***(100 carateres).***
6

1.7. Observações[4] ***(1.000 carateres).***

1.7. Remarks ***(1.000 carateres).***

2. Docente responsável e respetiva carga letiva na Unidade Curricular ***(preencher o nome completo) (1.000 carateres).***
Pedro Miguel Henriques dos Santos Félix

3. Outros docentes e respetivas cargas letivas na unidade curricular ***(1.000 carateres).***

4. Objetivos de aprendizagem (conhecimentos, aptidões e competências a desenvolver pelos estudantes). ***(1.000 carateres).***

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, corotinas e *reactive streams*.

4. Intended learning outcomes (knowledge, skills and competences to be developed by the students). ***(1.000 characters).***

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 ***(1.000 carateres).***

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 ***(1.000 characters).***

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 ***(1.000 carateres).***

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 ***(1.000 characters).***

Items (1) to (5) of the syllabus provides the knowledge required to correctly develop programs on multi-threaded contexts, including proper data synchronization and thread coordination, therefore addressing the first learning outcome.
Items (6) and (7) of the syllabus provides the additional knowledge for the development of asynchronous programs, addressing learning outcome (2), including the interaction between asynchronous models and *multi-threading*.

7. Metodologias de ensino (avaliação incluída) ***(1.000 carateres).***

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). Cada estudante resolve, em laboratório aberto, três séries de exercícios práticos, abordando todos os tópicos abordados na UC. Realiza-se avaliação escrita cobrindo todos os objetivos de aprendizagem. Adicionalmente, esses objetivos de aprendizagem são também avaliados com base nas resoluções das séries de exercícios práticos.

7. Teaching methodologies (including assessment) ***(1.000 characters).***

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, in open laboratory, three series of practical exercises, addressing all topics covered in the UC. A written evaluation covering all learning objectives is carried out. In addition, those learning objectives are also evaluated based on the resolutions of the practical exercise series.

8. Demonstração da coerência das metodologias de ensino com os objetivos de aprendizagem da unidade curricular ***(3.000 carateres).***

As aulas teórico-práticas são utilizadas para abordar os principais conceitos e técnicas da programação concorrente, assíncrona e a forma como os mesmos estão presentes nos ambientes virtuais de execução tomados como referência. Através desta metodologia, os estudantes são confrontados com problemas reais e com as soluções consideradas aceitáveis. Através da resolução das séries de exercícios, realizada individualmente em laboratório aberto e com apoio do docente, sempre que solicitado, cada estudante poderá, ao seu próprio ritmo, exercitar e consolidar os conceitos e técnicas abordadas nesta UC.

8. Evidence of the teaching methodologies coherence with the curricular unit's intended learning outcomes ***(3.000 characters).***

Theoretical-practical classes are used to address the main concepts and techniques of concurrent, asynchronous and programming and how they are present in virtual execution environments taken as a reference. Through this methodology, students are faced with real problems and with solutions considered acceptable. Through the resolution of the series of exercises, carried out individually in an open laboratory and with the support of the teacher, each student can, at his own pace, exercise and consolidate the concepts and techniques covered in this UC.

9. Bibliografia de consulta/existência obrigatória ***(1.000 carateres).***[1]

B. Goetz, *Java Concurrency in Practice*, Addison-Wesley Professional, 2006. ISBN 9780321349606

J. Duffy, *Concurrent Programming on Windows,* Addison-Wesley Professional, 2008. ISBN 9780321434821

R. Blewett, A. Clymer, *Pro Asynchronous Programming with .NET*, APress, 2013. ISBN 9781430259206

C. Champbel, R. Johnson, A. Miller, S. Toub, *Parallel Programming with Microsoft .NET*, Microsoft Press, 2010. ISBN 9780735651593

C. Martins, *Sincronização com Monitores na CLI e na Infraestrutura Java*, 3ª edição, ISEL, 2009.

---

[1] Anual, semestral, trimestral, …

[2] Número total de horas de trabalho.

[3] Discriminadas por tipo de metodologia adotado (T - Ensino teórico; TP - Ensino teórico-prático; PL - Ensino prático e laboratorial; TC - Trabalho de campo; S - Seminário; E - Estágio; OT - Orientação tutorial; O - Outro).

[4] Assinalar sempre que a unidade curricular seja optativa.