UAlg
UNIVERSIDADE DO ALGARVE

English version at the end of this document

Ano Letivo 2019-20

Unidade Curricular LABORATÓRIO DE ENGENHARIA DE SOFTWARE

Cursos ENGENHARIA INFORMÁTICA (1.º ciclo)

Unidade Orgânica Faculdade de Ciências e Tecnologia

Código da Unidade Curricular 14781064

Área Científica CIÊNCIA DE COMPUTADORES

Sigla

Línguas de Aprendizagem Português

Modalidade de ensino Presencial

Docente Responsável Paula Cristina Negrão Ventura Martins

DOCENTE	TIPO DE AULA	TURMAS	TOTAL HORAS DE CONTACTO (*)
Paula Cristina Negrão Ventura Martins	PL; T	T1; PL1; PL2	15T; 90PL

* Para turmas lecionadas conjuntamente, apenas é contabilizada a carga horária de uma delas.

ANO	PERÍODO DE FUNCIONAMENTO*	HORAS DE CONTACTO	HORAS TOTAIS DE TRABALHO	ECTS
3º	S2	15T; 45PL	168	6

* A-Anual;S-Semestral;Q-Quadrimestral;T-Trimestral

Precedências

Sem precedências

Conhecimentos Prévios recomendados

Programação, Engenharia de Software, Análise e Modelação de Sistemas, Interfaces Pessoa-Máquina

Objetivos de aprendizagem (conhecimentos, aptidões e competências)

Aprofundar a cultura e sensibilidade acerca das temáticas da engenharia dos sistemas de informação, designadamente nos aspectos avançados da modelação, do processo de desenvolvimento e da utilização de ferramentas CASE de suporte. No final, os alunos devem:

- Descrever e aplicar as várias fases do processo de desenvolvimento de software
- Planear e gerir o desenvolvimento ágil de um sistema informático
- Conhecer e aplicar ferramentas no desenvolvimento de aplicações de acordo com os padrões da indústria
- Desenvolver e testar o sistema informático de acordo com as boas práticas do processo selecionado e da engenharia de software

Conteúdos programáticos

1. Metodologias de desenvolvimento de software tradicionais (Iconix, RUP, Catalysis, Nikel, Ferramentas EPF)
2. Metodologias de desenvolvimento de software ágeis (XP, Scrum, Crystal, DSDM)
3. Metamodelo SPEM
4. Metlhoria do Processo de Desenvolvimento de Software (CMMI, ISO/IEC TR 15504 SPICE, BOOTSTRAP)
5. Ferramentas Colaborativas de Gestão de Projectos
6. Ferramentas CASE (Evolução histórica, áreas de intervenção, classificações, avaliação)
7. Model-Driven Approach (MDA) e Model-Driven Development (MDD)
8. Testes de software
9. Gestão de Processos de Negócio

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

Os conteúdos programáticos estão em coerência com os objectivos da unidade curricular dado que o programa aborda de forma integrada o desenvolvimento de software, introduzindo os processos ágeis e ferramentas adequadas (ponto 1, 2, 5 e 6 dos conteúdos programáticos). A melhoria de processos foca a importância do aumento de produtividade das organizações face à qualidade dos produtos e redução dos custos e tempos dos projetos de software (ponto 4).

O MDD surge como novo paradigma de desenvolvimento de software. Neste contexto apresenta-se a abordagem MDA proposta pelo OMG (ponto 7). A transformação baseada em metamodelos é uma das técnicas mais usadas no domínio da MDA. O SPEM permite introduzir os conceitos e técnicas relacionados com metamodelação, usando como caso de estudo os processos de desenvolvimento de software (ponto 3).

Sendo uma unidade curricular do último ano, com o projeto pretende-se consolidar os conhecimentos adquiridos ao longo do curso.

Metodologias de ensino (avaliação incluída)

Aulas T(15h) As noções teóricas serão dadas por método predominantemente expositivo, com projeção e explicação dos objetivos e conteúdos correspondentes a cada tema, acompanhado de debate, colocação e esclarecimento de dúvidas.
Aulas P(45h):Os estudantes serão motivados para aplicar as competências adquiridas através de atividades práticas, incluindo a análise e implementação de problemas.
Desenvolvimento de sistema informático de média dimensão integrando os conhecimentos disciplinas da área científica de Sistemas de Informação e Bases de Dados e recorrendo a metodologias ágeis
Reuniões semanais para gestão do projeto

A aprovação depende da assiduidade: (75% aulas teóricas) e (90% aulas laboratoriais). Componentes de avaliação: Presentação (15%), Documentação (15%), Gestão Projecto (10%), Implementação e testes (60%). Classificação entre 0-20 valores.

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

As metodologias de ensino estão em coerência com os objectivos da unidade curricular dado que: 1) a exposição do programa associada à apresentação de casos práticos possibilita uma explicitação adequada dos conteúdos face ao público-alvo; 2) a exposição de evidência científica em conjunto com a análise de casos práticos permitem mostrar a importância dos processos de desenvolvimento de software e da melhoria de processos, bem como do desenvolvimento baseado em modelos; 3) a exposição dos problemas atuais, complementadas com a realização de um trabalho prático permite a aplicação do desenvolvimento baseado em modelos, bem como a realização de testes de software. O regime de avaliação foi concebido para medir até que ponto as competências foram desenvolvidas

Bibliografia principal

1. Software Engineering: A Practitioner's Approach, 8th Edition, 2014, Roger S Pressman, MacGraw-Hill Higher Education
2. Software Engineering: Theory and Practice, 4th Edition, 2019
3. Shari Lawrence Pfleeger, Joanne M. Atlee, Prentice Hall
4. CMMI(R): Guidelines for Process Integration and Product Improvement, 3rd Edition, 2011
5. Mary Beth Chrissis, Mike Konrad, Sandy Shrum, Addison-Wesley
6. Model Driven Architecture (OMG): Applying MDA to Enterprise Computing, David S. Frankel, Wiley, 2010
7. UML-Metodologias e Ferramentas CASE, Alberto Silva e Carlos Videira, 2ª Edição, 2005, Centro Atlântico

Academic Year	2019-20
Course unit	SOFTWARE ENGINEERING LABORATORY
Courses	INFORMATICS (COMPUTER SCIENCE) (1st Cycle)
Faculty / School	FACULTY OF SCIENCES AND TECHNOLOGY
Main Scientific Area	CIÊNCIA DE COMPUTADORES
Acronym	
Language of instruction	Portuguese
Teaching/Learning modality	Presential
Coordinating teacher	Paula Cristina Negrão Ventura Martins

Teaching staff	Type	Classes	Hours (*)
Paula Cristina Negrão Ventura Martins	PL; T	T1; PL1; PL2	15T; 90PL

* For classes taught jointly, it is only accounted the workload of one.

Contact hours

T	TP	PL	TC	S	E	OT	O	Total
15	0	45	0	0	0	0	0	168

T - Theoretical; TP - Theoretical and practical ; PL - Practical and laboratorial; TC - Field Work; S - Seminar; E - Training; OT - Tutorial; O - Other

Pre-requisites

no pre-requisites

Prior knowledge and skills

Programming, Software Engineering, Systems Analysis and Modeling, Human-Computer Interaction

The students intended learning outcomes (knowledge, skills and competences)

This curricular unit aims to deepen and explore the culture and sensibility about software engineering issues, particularly advanced aspects of modelling, software development processes and CASE tools. In the end, students should be able to:
1. Describe and apply the several software development phases
2. Plan and manage a software development project

3. Know and use appropriate software development tools and management tools to develop applications conforming to industry standards 4. Develop and test a software system according to the software development process best practices.

Syllabus

1. Traditional Software Development Methodologies (Iconix, RUP, Catalysis, Nikel, Ferramentas EPF)
2. Agile Software Development Methodologies (XP, Scrum, Crystal, DSDM)
3. SPEM Meta-model
4. Software Development Process Improvement (CMMI, ISO/IEC TR 15504 SPICE, BOOTSTRAP)
5. Collaborative Project Management Tools
6. CASE Tools (Evolution, intervention areas, classification, evaluation)
7. Model-Driven Approach (MDA) e Model-Driven Development (MDD)
8. Software Tests
9. Business Process Management

Demonstration of the syllabus coherence with the curricular unit's learning objectives

The syllabus is consistent with the objectives of the curricular unit since the syllabus was designed to address, in an integrated way, Software Development, introducing agile processes and appropriate tools (point 1, 2, 5 and 6 of the syllabus). Software Process Improvement focuses on the importance of increasing organizations productivity, looking particularly at the effect of product quality and software projects costs and schedule (point 4).

MDD is a new software development paradigm. In this context, we present the MDA approach proposed by the OMG (point 7). In the domain of MDA, metamodel transformation is one of the most used techniques. SPEM allows introducing metamodeling concepts and techniques, using software development processes as a case study (point 3).
As a final course, the project aims to consolidate all of the theoretical and practical knowledge acquired throughout the course.

Teaching methodologies (including evaluation)

Theoretical lessons (15h). Theoretical notions are predominantly given by expository-style lectures, projection and explanation objectives and contents relevant to each theme, followed by debate and questions.

Practical lessons (45h). Students are encouraged to apply the competences acquired through practical activities, including the analysis and development of problems. Development of a medium-sized software system, in conjunction with skills and concepts from previous courses in the scientific area of Information Systems and Databases using agile methodologies.Weekly meetings for project management.

Course approval depends on the following attendance conditions: 75% of theoretical lessons and 90% of lab lessons. Assessment components: Presentation (15%), Documentation (15%), Project Management (10%), Implementation and tests (60%). Classification from 0-20 values.

Demonstration of the coherence between the teaching methodologies and the learning outcomes

The teaching methodologies are consistent with the objectives of the curricular unit because 1) the exposition of the syllabus associated with the presentation of practical cases allows an adequate explanation of the contents over the target public; 2) the exposition of scientific evidence together with the analysis of practical cases allows to show the importance of software development processes and software process improvement, as well as model-driven software development; 3) the exposition of current challenges, complemented with a practical work allows to apply model-driven software development approaches, as well as software testing. The assessment scheme was designed to measure the extent to which competences were developed.

Main Bibliography

1. Software Engineering: A Practitioner's Approach, 8th Edition, 2014, Roger S Pressman, MacGraw-Hill Higher Education
2. Software Engineering: Theory and Practice, 4th Edition, 2019
3. Shari Lawrence Pfleeger, Joanne M. Atlee, Prentice Hall
4. CMMI(R): Guidelines for Process Integration and Product Improvement, 3rd Edition, 2011
5. Mary Beth Chrissis, Mike Konrad, Sandy Shrum, Addison-Wesley
6. Model Driven Architecture (OMG): Applying MDA to Enterprise Computing, David S. Frankel, Wiley, 2010
7. UML-Metodologias e Ferramentas CASE, Alberto Silva e Carlos Videira, 2ª Edição, 2005, Centro Atlântico