



Ambientes de Desenvolvimento Avançados

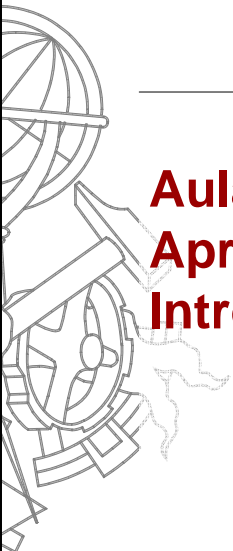
<http://www.dei.isep.ipp.pt/~jtavares/ADAV>

Aula 1 Engenharia Informática

2005/2006

José António Tavares
jrt@isep.ipp.pt

1



Aula de Apresentação e de Introdução

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

2

Corpo Docente

Docente	Sigla	Gabine	Aulas	Turmas	Home Page
José Tavares	JRT	Gabine G15, piso 3, Edifício G	T, T/P e P		http://www.dei.isep.ipp.pt/~jtavares/
Paulo Sousa	PAG		P		www.dei.isep.ipp.pt/~psousa/
Teófilo Matos	TBM		P		www.dei.isep.ipp.pt/~tmatos/
Nuno Ferreira	NCF		P		www.dei.isep.ipp.pt/~nacf/
Nuno Malheiro	NTM		P		www.dei.isep.ipp.pt/~ntm/

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

3

Objectivos da disciplina

- Utilização e domínio de tecnologias actuais de desenvolvimento de aplicações:
 - Projectar e desenvolver componentes de *software*.
 - Desenvolvimento de aplicações por composição de componentes de *software*.
 - Programação com componentes e ligação a bases de dados;
- Novas *frameworks* e arquitecturas para o desenvolvimento de aplicações baseadas em ambientes de execução e máquinas virtuais. O caso do .Net. e Java.

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

4



Programa da Disciplina

1. Introdução ao desenvolvimento de aplicações em arquitecturas avançadas.
2. Introdução ao desenvolvimento de aplicações baseadas em componentes de *software*.
Motivação. Definição de componentes de software. Conceitos relacionados com componentes de software. Modelos para o desenvolvimento de aplicações baseadas em componentes de software. Arquitecturas cliente-servidor e distribuídas. Exemplos. Discussão de casos. Desenvolvimento de componentes em .NET.
3. Introdução à utilização de ambientes de execução de *código gerido* (*managed code*).
O principio das máquinas virtuais para execução controlada de código. O caso da máquina virtual de Java e do CLR. Diferença face a código que não executa em máquinas virtuais. A *Framework* .NET.

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

5



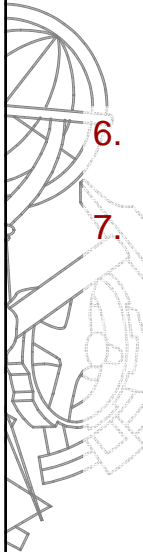
Programa da Disciplina

4. Acesso a bases de dados em .NET.
ADO.NET. Desenvolvimento de aplicações cliente-servidor com acesso a bases de dados.
5. Introdução a tecnologia COM.
Desenvolvimento de componentes COM simples. Automação, *scripting* e bibliotecas de tipos (*type libraries*) no COM. O OLE DB. O ADO como API para aplicações com acesso e manipulação de dados no COM. Interoperacionalidade entre a tecnologia COM e .NET.

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

6



Programa da Disciplina

6. Ferramentas de desenvolvimento de aplicações.
7. Introdução aos Serviços WEB (*Web Services*).

Serviços WEB como uma plataforma distribuída de computação na WEB. A *Framework* dos serviços WEB em termos dos formatos e protocolos (SOAP, XML e HTTP), linguagens de descrição (WSDL) e descoberta de serviços (UDDI). Desenvolvimento de serviços WEB em .NET. Exemplos simples de descoberta e invocação de serviços WEB.



Programa da Disciplina

Aulas Práticas

A parte prática da disciplina destina à resolução de problemas propostos de acordo com alguns conceitos teóricos e tecnologias apresentadas nas aulas teóricas.

Os alunos terão liberdade de escolha na linguagem de programação. A *framework* .NET será usada para o desenvolvimento de componentes de *software*.



Bibliografia

- **“Component Software: Beyond Object-Oriented Programming”**, Clemens Szyperski, 2nd Edition, The Component Software Series, Addison Wesley Professional. ISBN: 0201745720.
- **“Designing Data Tier Components and Passing Data Through Tiers - patterns & practices”**, Microsoft, <http://www.microsoft.com/downloads/details.aspx?displaylang=en&FamilyID=A8381E9C-884D-4CB2-9DBE-255C2790634B>.
- Outro material que se poderá revelar importante no decurso do semestre.



Métodos de Avaliação

A Nota Final da disciplina resulta de duas componentes:

- Avaliação contínua é constituída pela elaboração de um trabalho prático obrigatório pesando 45% na nota final e por uma apreciação individual que poderá envolver alguns questionários de carácter teórico com peso de 5% na nota final.
- PROVA DE EXAME pesando 50% na nota final.



Métodos de Avaliação

Trabalho Prático

- O trabalho é obrigatório.
- O trabalho deve ser desenvolvido em grupo com número limitado a 2 (duas) pessoas e será classificado após a sua apresentação oral.
- As notas dos trabalhos são individuais.
- O trabalho prático será desenvolvido em 2 (duas) etapas. A primeira etapa é a uma especificação da solução para o problema que será proposto e a segunda etapa consiste na implementação da solução proposta.
- Estas duas etapas serão avaliadas ao longo do semestre lectivo (período de aulas) tendo a 1ª etapa um peso de 30% na nota de frequência e a 2ª um peso de 60%.
- Os restantes 10% correspondem aos questionários de carácter teórico.



Métodos de Avaliação

- No caso de alunos com dispensa de avaliação contínua, o trabalho poderá ser desenvolvido fora das aulas práticas e avaliados na última semana de aulas.
- Neste as 2 etapas do trabalho terão um peso de 40% e 60%, respectivamente.



Métodos de Avaliação

- A Prova de Exame é dividida em duas partes:
 1. Parte teórica que vale 60% da nota da prova e
 2. Parte prática que vale 40% da nota da prova.
- Existe nota mínima de 8 valores nas duas partes da prova.

- **Classificação final da disciplina**

$$\left(\frac{xNFREQ + yPE}{x + y} \right)$$

$$x = 0.5$$

$$y = 0.5$$

$$\text{Min NFREQ} = 10$$

$$\text{Min PE} = 8$$

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

13



Métodos de Avaliação

Melhoria de Nota (avaliação)

- A melhoria de nota exige a realização de uma prova prática (PP) especial realizada em computador além da prova escrita da época de avaliação.
- O aluno pode optar por manter a nota de frequência dispensando a realização dessa prova prática.
- **Nota de Melhoria = 50% PE + 50% PP (ou NFREQ).**
- Aplicam-se as mesmas notas mínimas entre PE e PP relativamente às que são definidas para cálculo da classificação da disciplina.

2005/2006

ADAV
Ambientes de Desenvolvimento Avançados

14



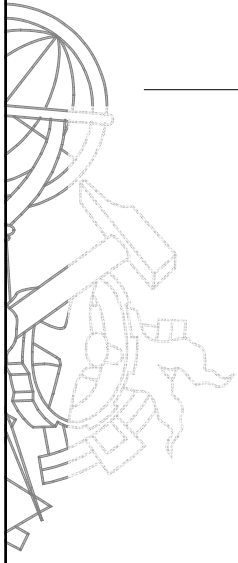
O que é um componente?

- **“A software package which offers *service through interfaces*”**
[Peter Herzum and Oliver Sims, “Business Components Factory: A Comprehensive Overview of Component-Based Development for the Enterprise”, John Wiley & Sons, Incorporated, 1999].
- **“A coherent package of software artifacts that can be independently developed and delivered as a unit and that can be composed, unchanged, with other components to build something larger”**
[D.F. D’Souza and A.C. Wills, “Objects, Components, And Frameworks with UML – The Catalysis Approach” Addison-Wesley, 1998].
- **“A component is a unit of composition with contractually specified interfaces and *explicit context dependencies* only. A software component can be deployed independently and is subject to composition by third parties.”**
[C. Szyperski, “Component Software: Beyond Object-Oriented Programming” Addison-Wesley, 1998].



O que não é um componente?

Component isn't an object, not in sense of simply being an object in a Java or C++ program, although it is true at runtime.



Questões

