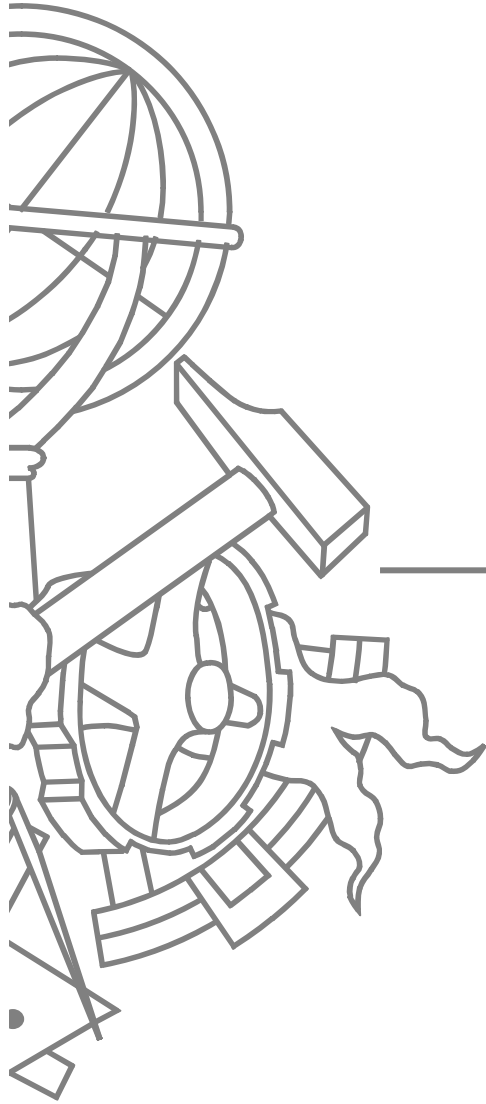


Padrões de Aplicações Empresariais

Paulo Sousa

Engenharia da Informação
Instituto Superior de Engenharia do Porto



Introdução aos Padrões

Parte 1

O que é um *Pattern*?



Each pattern describes a **problem** that **occurs over and over** again in our environment and then describes the **core of the solution** to that problem in such a way that you can **use this solution a million times over without ever doing it the same way twice.**

Christopher Alexander

O que é um *Pattern*?

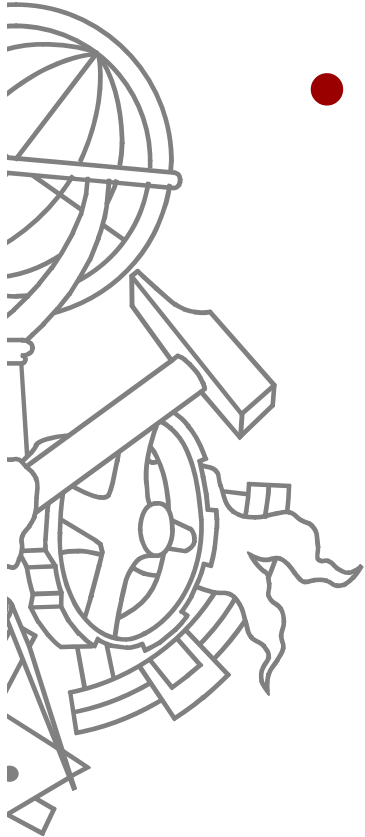


A design pattern names, abstracts, and identifies the key aspects of a common design structure that make it useful for creating a reusable object-oriented design.

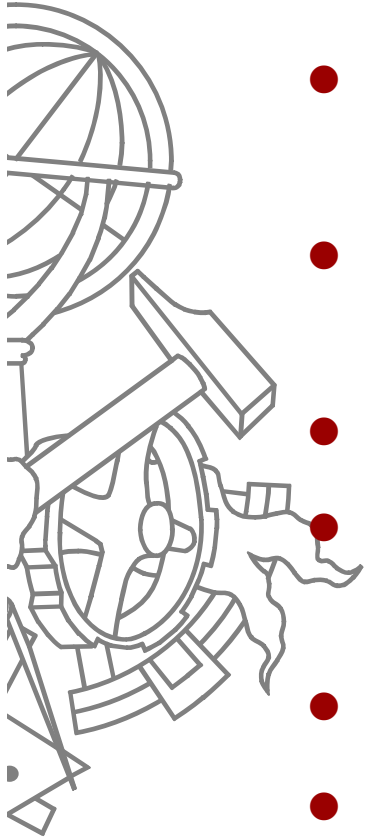
Design Patterns-Elements of Reusable
Object-oriented Software, Gamma et al.
(Gang of Four)

O que não é um *pattern*

- Um padrão **não** é uma receita milagrosa que resolve todos os problemas da aplicação ou empresa

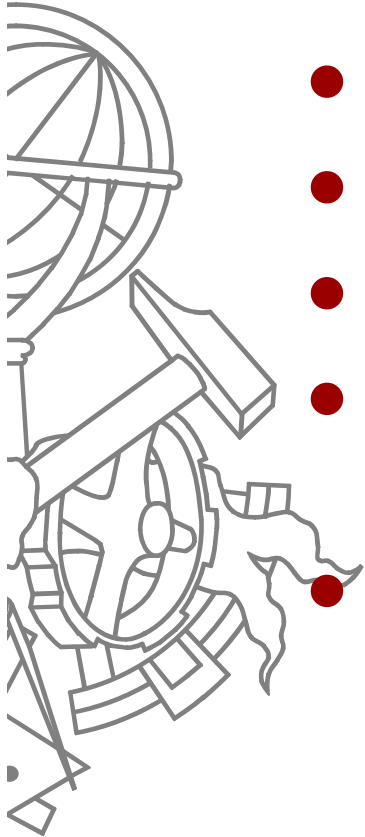


O que é um *Pattern*?



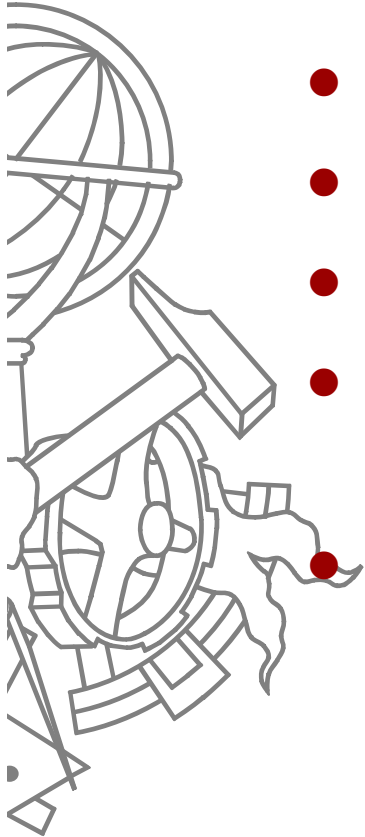
- Um padrão (*pattern*) é um conjunto de *best-practices*
- Representa uma solução tipificada para um problema usual num dado contexto
- Não reinventar a roda
- Facilita a comunicação ao criar um vocabulário comum
- Os padrões descobrem-se, não se inventam
- “Patterns are half-baked”
 - Martin Fowler

Vantagens de padrões



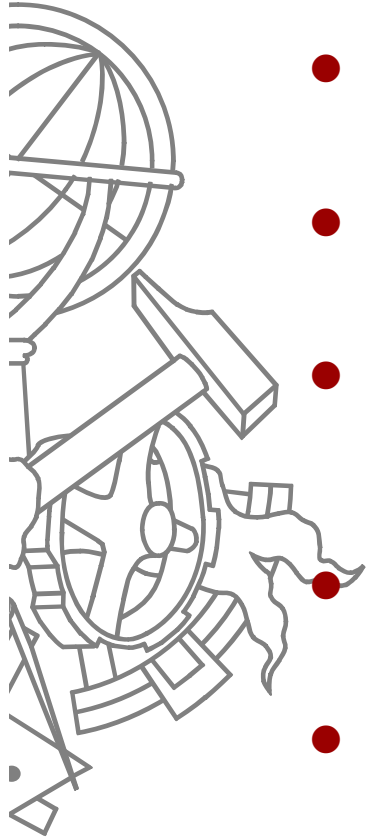
- Facilita a comunicação
- Descreve solução tipificada e “provada”
- Facilita a reutilização em larga-escala
- Capturam conhecimento dos peritos e *trade-offs* efectuados
- São maioritariamente agnósticos
 - Tecnologia
 - Linguagem de programação
 - Filosofia (*open source*, ...)

Desvantagens de padrões



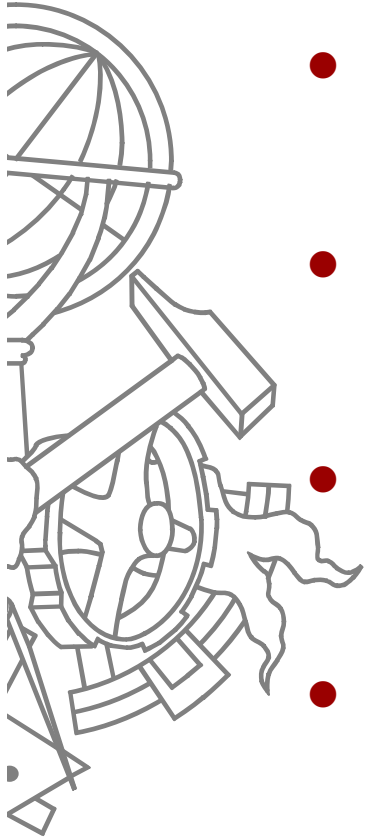
- Não permite reutilização directa de código
- São demasiado simples
- A equipa pode sofrer de *pattern overload*
- São validados através da experiência e da discussão em vez de testes automáticos
- A integração de padrões no processo de desenvolvimento de software é uma tarefa intensa do ponto de vista humano

Descrição de um padrão



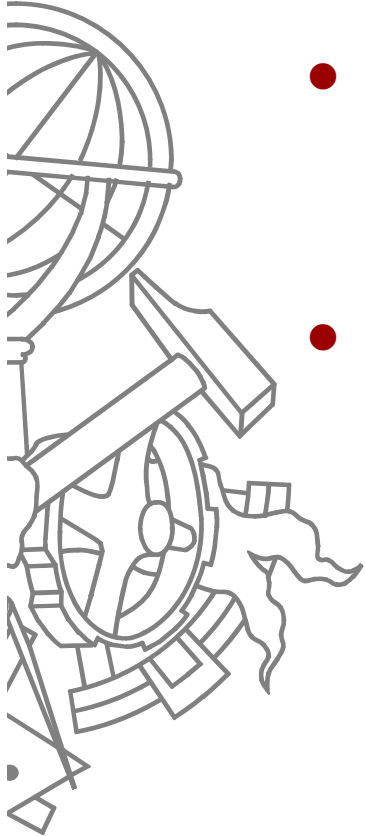
- **name**
 - Contributes to the pattern vocabulary
- **synopsis**
 - Short description of the problem the pattern will solve.
- **forces**
 - Requirements, considerations, or necessary conditions
- **solution**
 - The essence of the solution
- **counter forces**
 - Reasons for *not* using the pattern.
- **related patterns**
 - Possible alternatives in the design.

Catálogos de Patterns



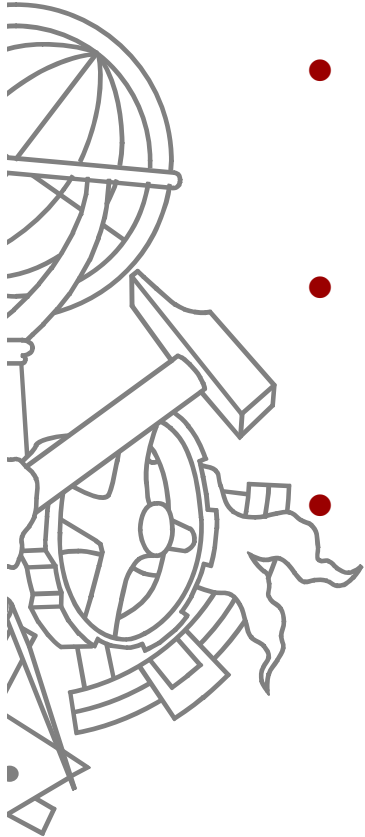
- *GoF Design patterns (em C#)*
<http://www.dofactory.com/Patterns/Patterns.aspx>
- *GoF & POSA Design patterns (em Java)*
<http://www.vico.org/pages/PatronsDisseny.html>
- *Patterns of Enterprise Application architecture*
<http://martinfowler.com/eaCatalog/>
- *PatternShare.org*
<http://patternshare.org/default.aspx/Home.EnterpriseArchitecturalSpaceOrganizingTable>

Catálogos de Patterns



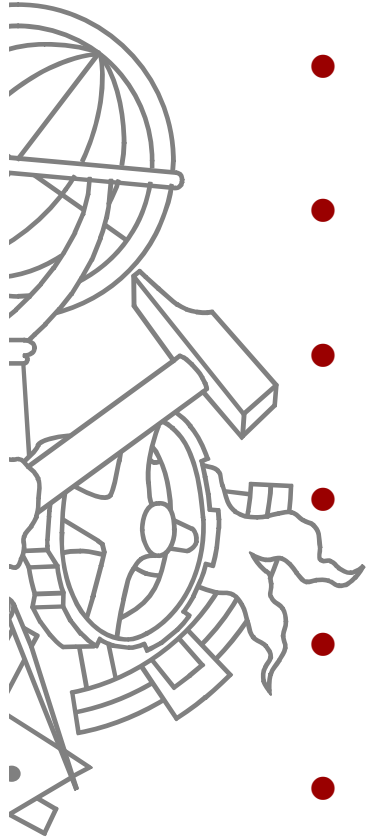
- *Core J2EE Patterns*
<http://www.corej2eepatterns.com>
- *Enterprise Solution Patterns Using Microsoft .NET.*
<http://msdn.microsoft.com/architecture/patterns/default.aspx?pull=/library/en-us/dnpatterns/html/Esp.asp>

Catálogos de Patterns

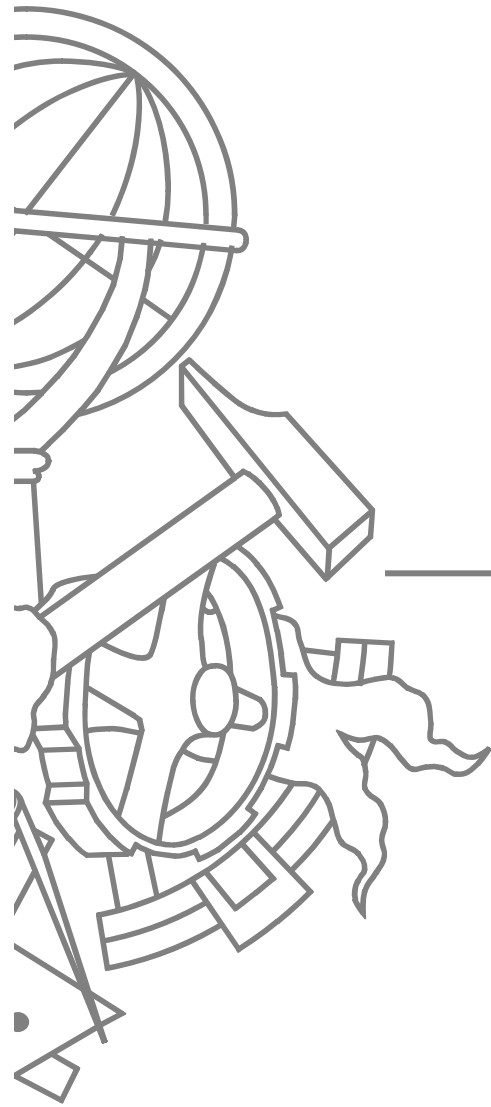


- *Patterns of Enterprise Application Integration*
<http://www.enterpriseintegrationpatterns.com/>
- *Enterprise Java Patterns @ The Server Side*
<http://www.theserverside.com/patterns/index.tss>
- *Microsoft Data Patterns*
<http://msdn.microsoft.com/architecture/patterns/default.aspx?pull=/library/en-us/dnpatterns/html/Dp.asp>

Notação/Definições



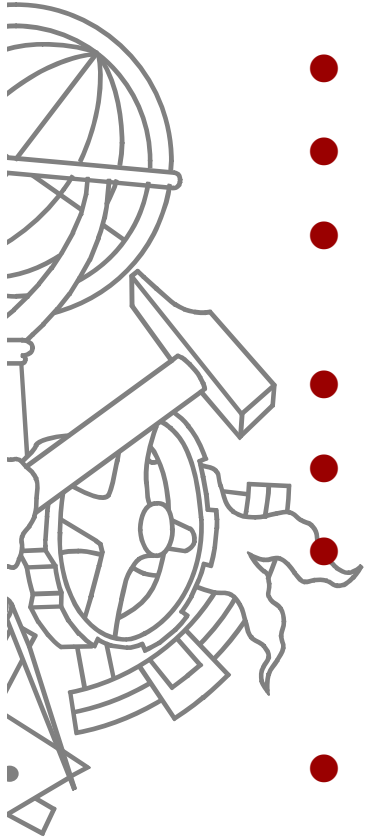
- [GoF]
 - Gang of four
- [PoEAA]
 - Patterns of Enterprise Application Architecture
- [CJP]
 - Core J2EE Patterns
- [ESP]
 - Enterprise Solution Patterns using Microsoft .NET
- [POSA]
 - Pattern-Oriented Software Architecture
- CRUD
 - Create, Read, Update, Delete



Padrões de Aplicações Empresariais

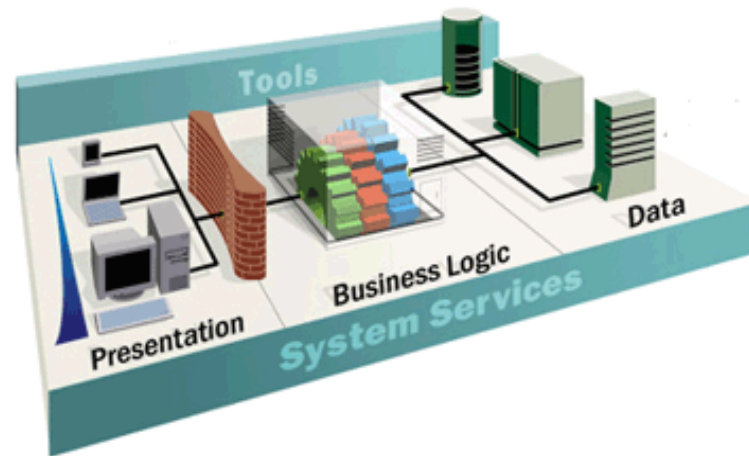
Parte 2

Aplicação empresarial



- Funcionalidade crítica para a empresa
- Dados persistentes e em grande quantidade
- Acessos simultâneos e concorrentes a esses dados
- Grande número de ecrãs
- Integração com outras aplicações
- Dissonância conceptual entre conceitos com mesmo nome entre departamentos/unidades (aplicações) diferentes
- Regras de negócio complexas e por vezes “ilógicas” (para contemplar situações particulares)

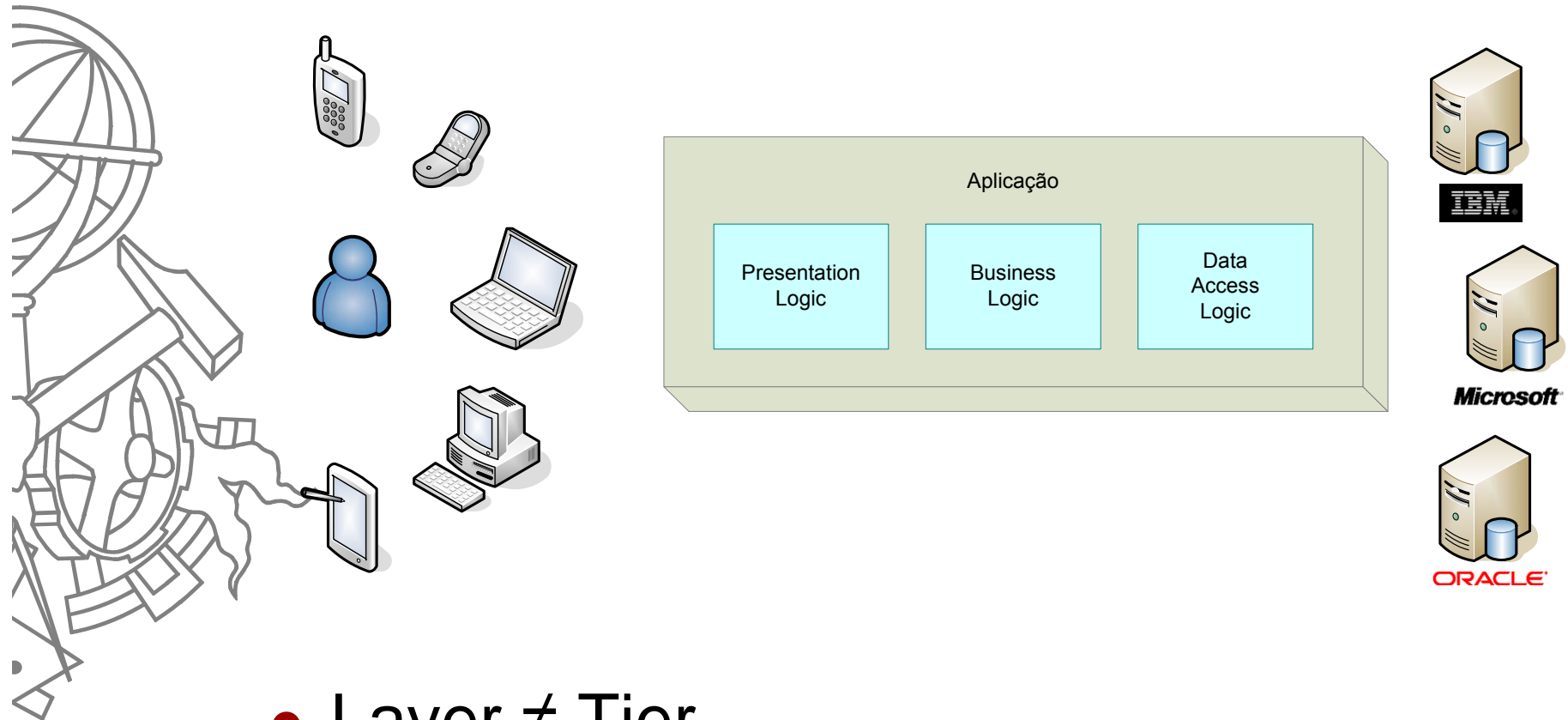
Arquitecturas por camadas



- Podem existir mais que 3 camadas

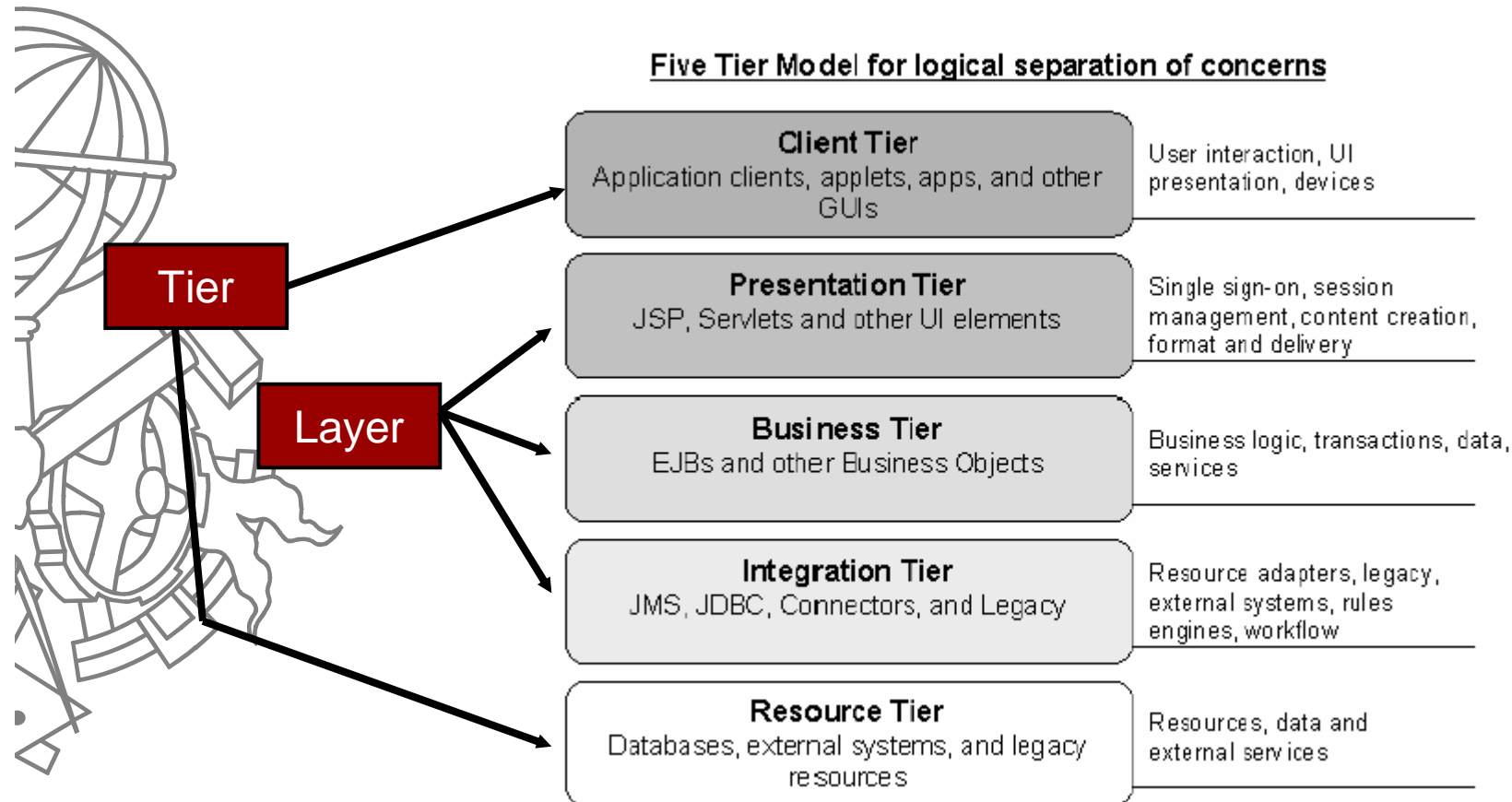
- Algumas plataformas definem modelo de camadas padrão a seguir pelas aplicações
 - J2EE
 - .Net

3-Layer

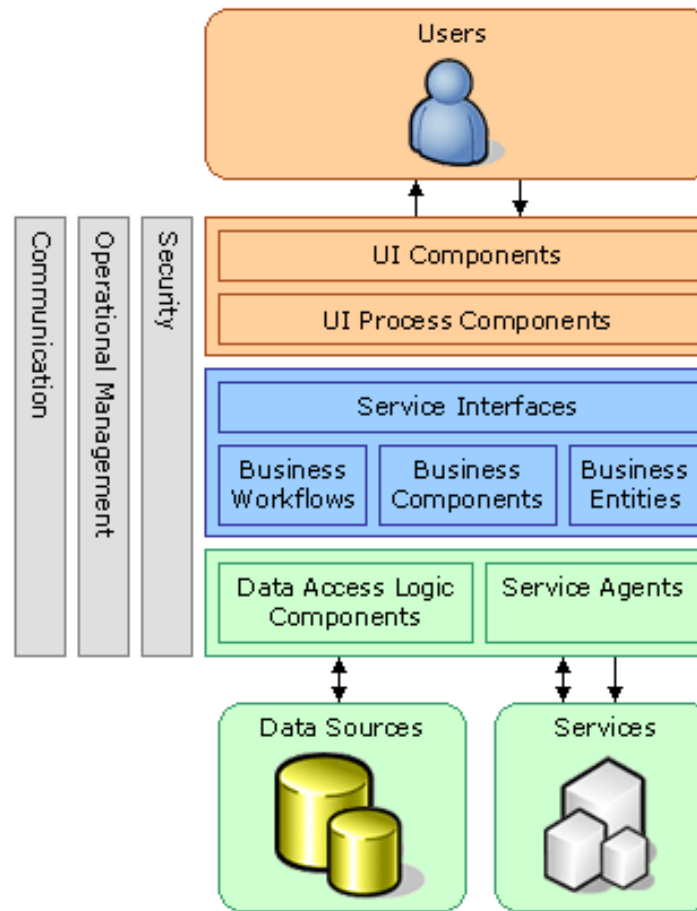
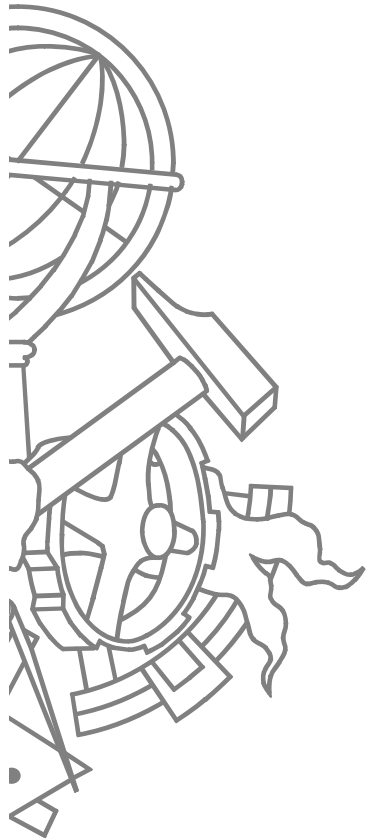


- Layer \neq Tier

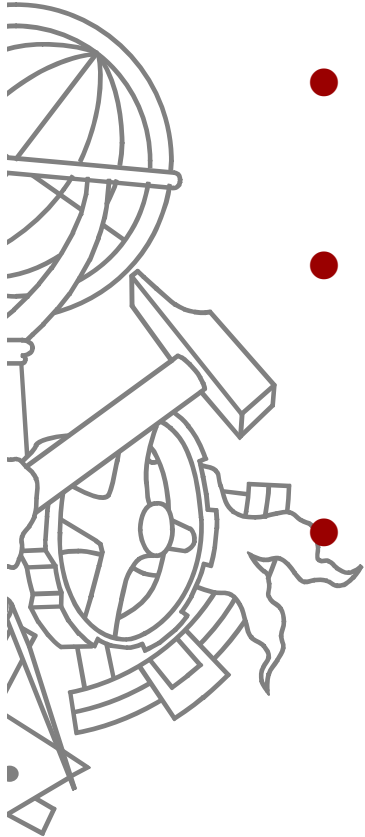
Arquitetura modelo (J2EE)



Arquitectura modelo (.net)

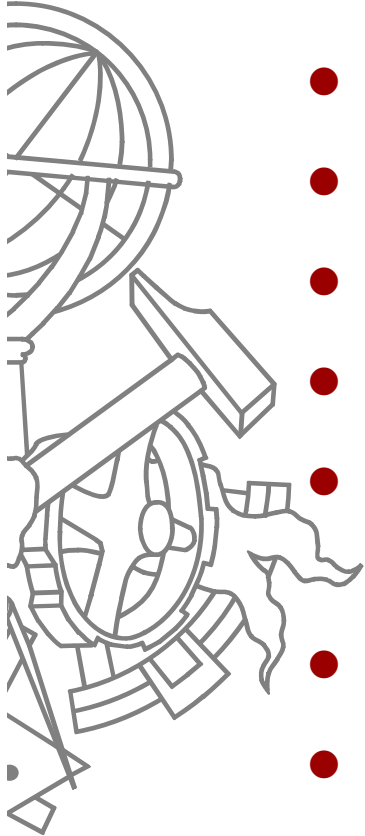


E então?...



- Que tipo de questões se colocam sobre cada uma das camadas apresentadas?
- Que tipo de padrões são conhecidos para cada camada no âmbito das aplicações empresariais?
- Nas próximas secções vamos falar de padrões para:
 - Entidades de negócio
 - Lógica de negócio
 - Lógica de acesso a dados
 - Lógica de apresentação
 - Concorrência na BD

Questões



- Como dividir a aplicação?
- Como representar as entidades de negócio?
- Como representar a lógica de negócio?
- Como persistir as entidades de negócio?
- Como garantir a consistência dos dados da base de dados?
- Como tratar a interacção com o utilizador?
- Como resolver questões relacionadas com a distribuição física da aplicação?