
ITnerante

Universo ESAF

Galáxia

2012 CGU Analista de Finanças e Controle

Desenvolvimento de Sistemas

Constelação Desenvolvimento de Sistemas

Desenvolvimento de sistemas

[Questão 7] Acoplamento é uma medida do número e da resistência

- [A] da extensão do procedimento.
- [B] das ligações entre classes.
- [C] das ligações entre procedimentos.
- [D] das relações entre atributos.
- [E] das ligações redundantes entre formas de processamento.

Desenvolvimento de sistemas

[Questão 7] Acoplamento é uma medida do número e da resistência

- [A] da extensão do procedimento.
- [B] das ligações entre classes.
- [C] das ligações entre procedimentos.
- [D] das relações entre atributos.
- [E] das ligações redundantes entre formas de processamento.
- A alternativa mais correta seria a **letra B**.

Acoplamento

- Craig Larman
  - É uma medida de quão fortemente um elemento em relação a outros:
    - Está conectado a eles
    - Tem conhecimento deles
    - Depende deles
- Pressman
  - É uma medida qualitativa do gram em que as **classes** são **conectadas entre si**
  - Cresce à medida que as **classes** (e componentes) tornam-se **interdependentes**
  - Um **objetivo importante** em **projeto no nível de componente** é preservar o **acoplamento tão baixo quanto possível**

Desenvolvimento de sistemas

[Questão 8] Polimorfismo é

- [A] a multiplicidade de atributos de determinada classe.
- [B] a propriedade de um diagrama de classes ter múltiplas classes possuidoras de atributos.
- [C] a habilidade de um atributo ou variável poder identificar instâncias de classes com atributos dependentes.
- [D] a propriedade de uma instrução poder apontar para múltiplos objetos de uma mesma classe sem implicações de desempenho.
- [E] a habilidade pela qual uma única operação ou nome de atributo pode ser definido em mais de uma classe e assumir implementações diferentes em cada uma dessas classes.

Desenvolvimento de sistemas

[Questão 8] Polimorfismo é

- [A] a multiplicidade de atributos de determinada classe.
- [B] a propriedade de um diagrama de classes ter múltiplas classes possuidoras de atributos.
- [C] a habilidade de um atributo ou variável poder identificar instâncias de classes com atributos dependentes.
- [D] a propriedade de uma instrução poder apontar para múltiplos objetos de uma mesma classe sem implicações de desempenho.
- **[E] a habilidade pela qual uma única operação ou nome de atributo pode ser definido em mais de uma classe e assumir implementações diferentes em cada uma dessas classes.**

Polimorfismo

- **Estático**
  - Métodos sobrecarregados (**overloading**)
    - Quando a classe possui métodos com o mesmo nome, porém com argumentos diferentes
    - A decisão de qual método chamar é tomada em **tempo de compilação**, baseada nos argumentos que foram passados
- **Dinâmico**
  - Método sobre-escritos (**overriding**)
    - Está associado com o **conceito de herança**
    - Ocorre quando uma subclasse redefine um método existente na superclasse
    - A decisão de qual método executar é tomada somente em **tempo de execução**

Polimorfismo

- Page-Jones
  - É a habilidade pela qual uma **única operação** ou **nome de atributo** pode ser definido em mais de uma classe e assumir implementações diferentes em cada uma dessas classes

Questão repetida

[2010 SUSEP - Analista de TI - Questão 23] Polimorfismo é a

- [A] utilização múltipla de programas em análise orientada a objetos.
- [B] habilidade de uma única operação ou nome de atributo ser definido em mais de uma classe e assumir diferentes implementações em cada uma dessas classes.
- [C] habilidade de um programador em desenvolver aplicações e caracterizar objetos com múltiplos atributos.
- [D] utilização de uma classe com diferentes formatos em programas com definição de objetos e atributos.
- [E] habilidade de uma única variável ser utilizada em diferentes programas orientados a objetos.

Questão repetida

[2010 SUSEP – Analista de TI – Questão 23] Polimorfismo é a

- [A] utilização múltipla de programas em análise orientada a objetos.
- **[B] habilidade de uma única operação ou nome de atributo ser definido em mais de uma classe e assumir diferentes implementações em cada uma dessas classes.**
- [C] habilidade de um programador em desenvolver aplicações e caracterizar objetos com múltiplos atributos.
- [D] utilização de uma classe com diferentes formatos em programas com definição de objetos e atributos.
- [E] habilidade de uma única variável ser utilizada em diferentes programas orientados a objetos.

Desenvolvimento de sistemas

[Questão 9] São qualidades da orientação a objetos:

- [A] Recuperabilidade. Confiabilidade. Precisão. Portabilidade. Distributividade. Armazenabilidade.
- [B] Reutilização. Confidencialidade. Robustez. Extensibilidade. Comutabilidade. Consistência.
- [C] Baixo risco. Computabilidade. Robustez. Extensibilidade. Distributividade. Escalabilidade.
- [D] Reutilização. Confiabilidade. Robustez. Extensibilidade. Distributividade. Armazenabilidade.
- [E] Acessibilidade. Compartimentabilidade. Robustez. Homogeneidade terminológica. Distributividade. Armazenabilidade.

Desenvolvimento de sistemas

[Questão 9] São qualidades da orientação a objetos:

- [A] Recuperabilidade. Confiabilidade. Precisão. Portabilidade. Distributividade. Armazenabilidade.
- [B] Reutilização. Confidencialidade. Robustez. Extensibilidade. Comutabilidade. Consistência.
- [C] Baixo risco. Computabilidade. Robustez. Extensibilidade. Distributividade. Escalabilidade.
- **[D] Reutilização. Confiabilidade. Robustez. Extensibilidade. Distributividade. Armazenabilidade.**
- [E] Acessibilidade. Compartimentabilidade. Robustez. Homogeneidade terminológica. Distributividade. Armazenabilidade.

Qualidades de sistemas OO

- Page-Jones
  - As **qualidades** mais frequentes observadas em **sistemas** construídos no modo **orientado a objeto** são:
    - **Reutilização**
    - **Confiabilidade**
    - **Robustez**
    - **Extensibilidade**
    - **Distributividade**
    - **Armazenabilidade**

Desenvolvimento de sistemas

[Questão 10] Os domínios das classes de um Sistema Orientado a Objetos normal são:

- [A] Aplicação. Sistema. Arquitetura. Bloco.
- [B] Concepção. Unidade da estrutura. Interface. Base.
- [C] Aplicação. Negócio. Hierarquia. Usuário.
- [D] Agrupamento. Negócio. Ambiente. Base.
- [E] Aplicação. Negócio. Arquitetura. Base.

Desenvolvimento de sistemas

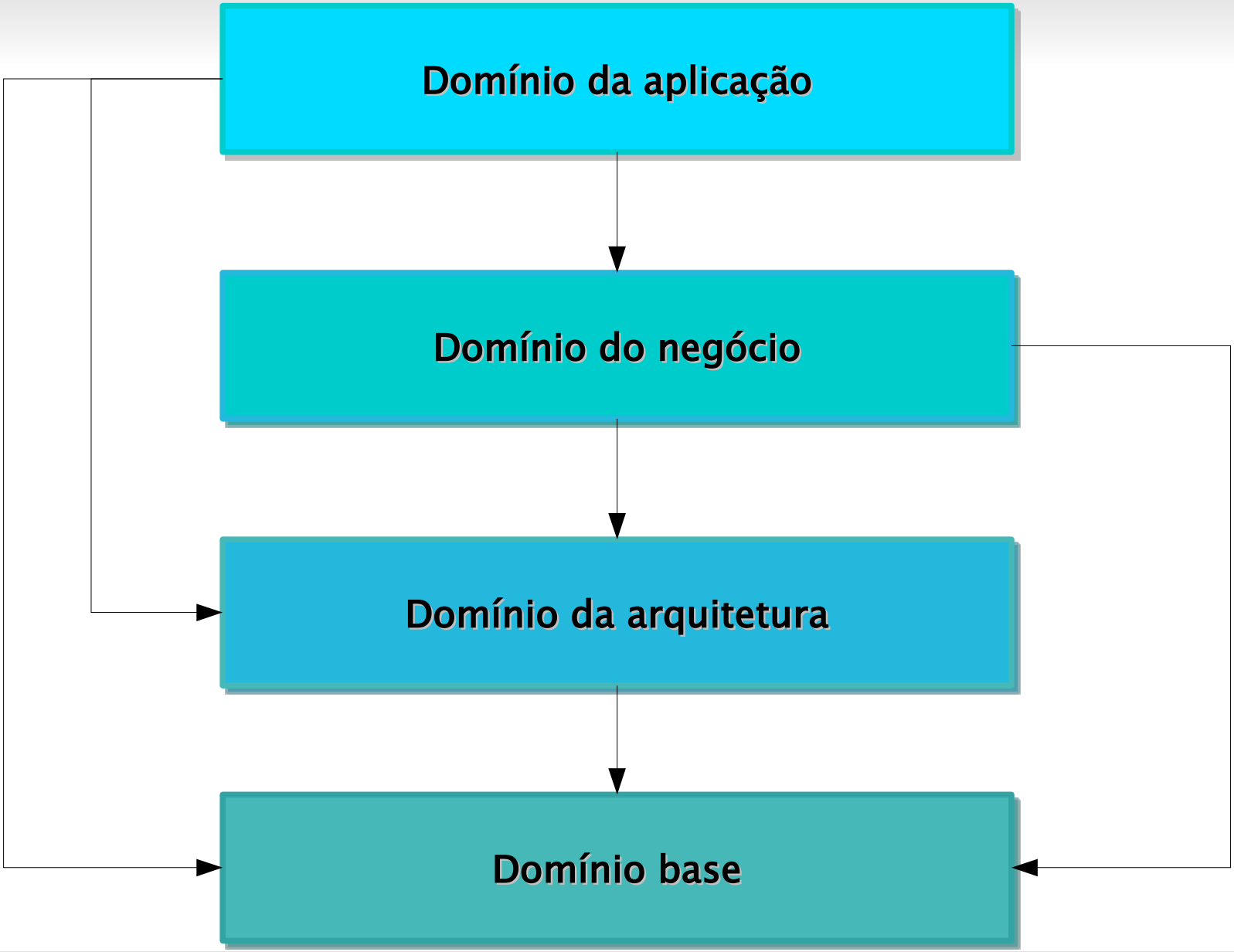
[Questão 10] Os domínios das classes de um Sistema Orientado a Objetos normal são:

- [A] Aplicação. Sistema. Arquitetura. Bloco.
- [B] Concepção. Unidade da estrutura. Interface. Base.
- [C] Aplicação. Negócio. Hierarquia. Usuário.
- [D] Agrupamento. Negócio. Ambiente. Base.
- **[E] Aplicação. Negócio. Arquitetura. Base.**

Camada de domínios das classes

- **Domínio** pode ser visto como uma estrutura de **classificação de elementos correlatos**
- Normalmente, sistemas OO tem suas classes em um dos seguintes **domínios**:
  - **Domínio de aplicação**
  - **Domínio de negócio**
  - **Domínio de arquitetura**
  - **Domínio de base**
- Cada classe de um sistema OO devem pertencer a um único domínio para ser coesa

Camada de domínios das classes



Camada de domínios das classes

- **Domínio da aplicação**
  - Contém **classes importantes** para uma **aplicação**
  - Por exemplo: classes de regras de negócios de uma aplicação
- **Domínio do negócio**
  - Contém **classes**:
    - **Importantes** para um **tipo de negócio**, tais como: Financeiro, Seguros e etc
    - Que têm um conjunto de regras válidas para todo o segmento

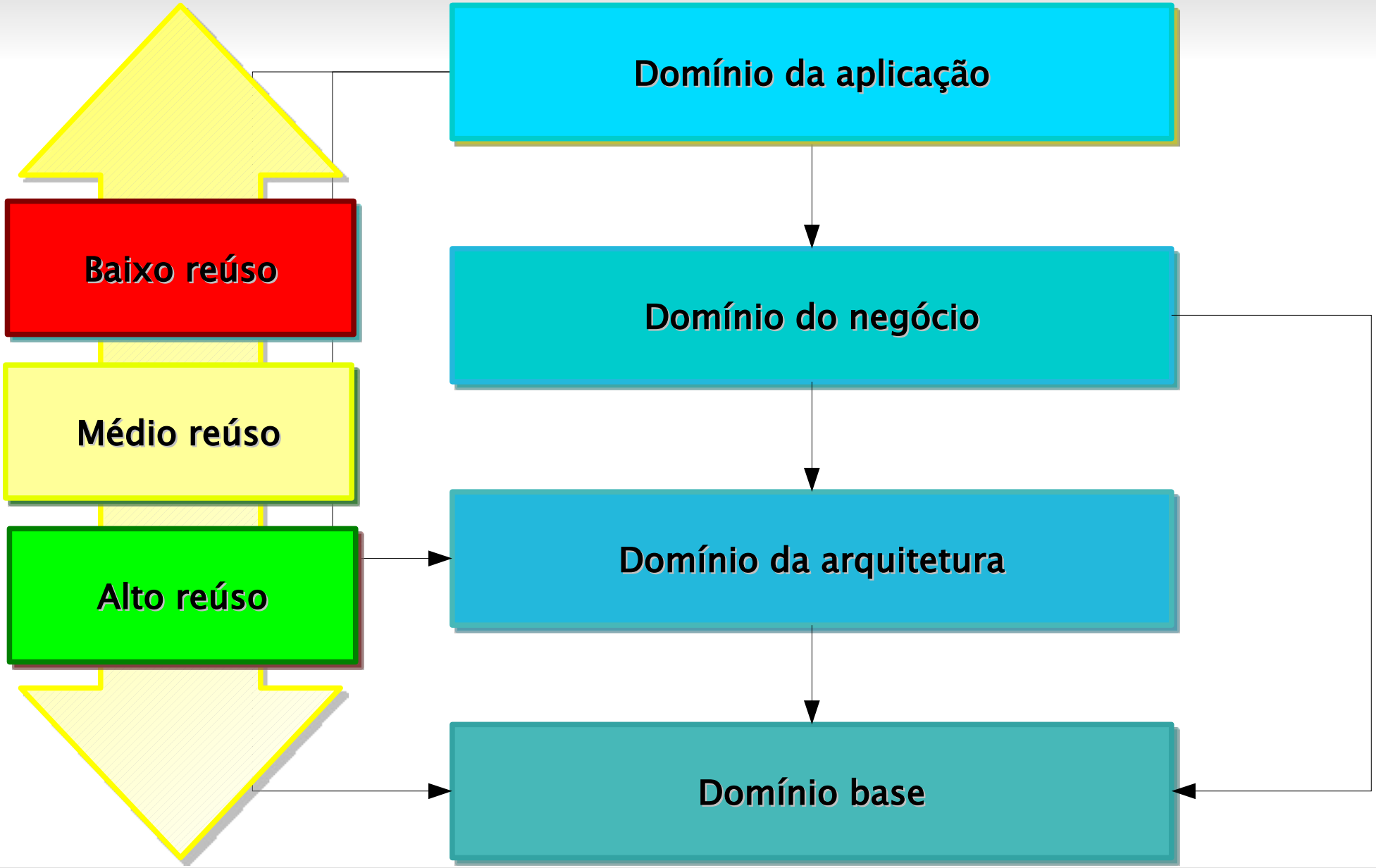
Camada de domínios das classes

- **Domínio da arquitetura**
  - Contém **classes importantes** para uma **arquitetura de implementação**
  - Por exemplo:
    - Classes de interface com usuário
    - Classes de manipulação de banco de dados
    - Classes de comunicação entre
      - Computadores
      - Outros dispositivos

Camada de domínios das classes

- **Domínio base**
  - Contém **classes importantes** para:
    - **Todas as arquiteturas**
    - **Áreas de negócios**
    - **Aplicação**
  - Geralmente estas classes estão atrelados a linguagem de programação
  - Por exemplo:
    - Classes bases
    - Classes estruturais e etc
  - Estas classes geralmente são tipos de dados, coleções e etc

Camada de domínios das classes



Desenvolvimento de sistemas

[Questão 11] Assinale a opção correta.

- [A] As classes podem formar heranças segmentadas em classes adjacentes.
- [B] *Overflow* é a redefinição do fluxo de uma classe, em uma de suas subclasses.
- [C] *Overriding* é a redefinição de um método, definido em uma classe, em uma de suas subclasses.
- [D] *Overriding* é a redefinição de uma classe através de métodos de objetos diferentes.
- [E] As classes não podem formar hierarquias de herança de superclasses e subclasses.

Desenvolvimento de sistemas

[Questão 11] Assinale a opção correta.

- [A] As classes podem formar heranças segmentadas em classes adjacentes.
- [B] *Overflow* é a redefinição do fluxo de uma classe, em uma de suas subclasses.
- **[C] *Overriding* é a redefinição de um método, definido em uma classe, em uma de suas subclasses.**
- [D] *Overriding* é a redefinição de uma classe através de métodos de objetos diferentes.
- [E] As classes não podem formar hierarquias de herança de superclasses e subclasses.

Desenvolvimento de sistemas

[Questão 12] Em linguagem Java

- [A] == significa atribuição. & significa “E” lógico. || significa “OU” lógico.
- [B] == significa igualdade. && significa atribuição lógica. || significa “+” lógico.
- [C] == significa igualdade. && significa “E” lógico. || significa “OU” lógico.
- [D] <> significa igualdade. &+ significa “E” lógico. | significa “OU” lógico.
- [E] =+ significa igualdade superior. && significa “E” lógico. |=| significa “OU” lógico.

Desenvolvimento de sistemas

[Questão 12] Em linguagem Java

- [A] == significa atribuição. & significa "E" lógico. || significa "OU" lógico.
- [B] == significa igualdade. && significa atribuição lógica. || significa "+" lógico.
- **[C] == significa igualdade. && significa "E" lógico. || significa "OU" lógico.**
- [D] <> significa igualdade. &+ significa "E" lógico. | significa "OU" lógico.
- [E] =+ significa igualdade superior. && significa "E" lógico. |=| significa "OU" lógico.

Operadores

Tipo	Operadores
Aritmético	+, -, *, /, %
Atribuição	=, +=, -=, *=, /=, %=, &=, ^=, |=, <<=, >>=, >>>=
Relacional	<, <=, >, >=, instanceof
Igualdade	==, !=
Lógico	&&, ||
Bitwise	&, I, ^
Postfix	var++, var--
Unário	++var, --var, +var, -var, ~, !
Shift aritmético	<<, >>
Shift lógico	>>>

Desenvolvimento de sistemas

[Questão 13] Os tipos primitivos da linguagem Java são

- [A] boolean, byte, narrow, int, wide, fixed, double, char.
- [B] boolean, byte, short, int, long, float, double, char.
- [C] buffered, byte, double-byte, single, long, float, double, char.
- [D] logical, boolean, short, local, extended, float, double, cast.
- [E] boolean, byte, short, integral, partial, long, float, char.

Desenvolvimento de sistemas

[Questão 13] Os tipos primitivos da linguagem Java são

- [A] boolean, byte, narrow, int, wide, fixed, double, char.
- **[B] boolean, byte, short, int, long, float, double, char.**
- [C] buffered, byte, double-byte, single, long, float, double, char.
- [D] logical, boolean, short, local, extended, float, double, cast.
- [E] boolean, byte, short, integral, partial, long, float, char.

Tipos, valores e variáveis

- Tipos primitivos

Tipo	Tamanho em bytes	Faixa	Valor padrão
byte	1	Inteiro: -128 a 127	0
short	2	Inteiro: -32.768 a 32.767	0
int	4	Inteiro: -2.147.483.648 a 2.147.483.647	0
long	8	Inteiro: -9.223.372.036.854.775.808 a 9.223.372.036.854.775.807	0L
float	4	Ponto flutuante: +/-3.4E-38 a +/-3.4E+38	0.0f
double	8	Ponto flutuante: +/-1.7E-308 a +/-1.7E+308	0.0d
boolean	1	true ou false	false
char	2	Caracteres UNICODE	'u\0000'

Desenvolvimento de sistemas

[Questão 14] Na linguagem Java, o comando **continue** tem a função de

- [A] fazer com que o comando de seleção seja inicializado.
- [b] permitir realçar a posição de determinados comandos.
- [C] modificar a estrutura do loop, realçando procedimentos.
- [D] fazer com que a continuidade da execução de um loop fique condicionada a um teste de condição de continuidade.
- [E] fazer com que a condição do comando de loop seja novamente testada, mesmo antes de alcançar o fim do comando.

Desenvolvimento de sistemas

[Questão 14] Na linguagem Java, o comando **continue** tem a função de

- [A] fazer com que o comando de seleção seja inicializado.
- [b] permitir realçar a posição de determinados comandos.
- [C] modificar a estrutura do loop, realçando procedimentos.
- [D] fazer com que a continuidade da execução de um loop fique condicionada a um teste de condição de continuidade.
- **[E] fazer com que a condição do comando de loop seja novamente testada, mesmo antes de alcançar o fim do comando.**

Desenvolvimento de sistemas

[Questão 15] O comando **break** tem a função de

- [A] interromper a execução de um loop.
- [B] condicionar a execução de um comando de atribuição a um operador lógico.
- [C] segmentar a execução de um loop em duas ou mais partes aninhadas.
- [D] estabelecer um intervalo de depuração durante a execução de um loop.
- [E] impossibilitar o aninhamento de loops não lógicos.

Desenvolvimento de sistemas

[Questão 15] O comando **break** tem a função de

- **[A] interromper a execução de um loop.**
- [B] condicionar a execução de um comando de atribuição a um operador lógico.
- [C] segmentar a execução de um loop em duas ou mais partes aninhadas.
- [D] estabelecer um intervalo de depuração durante a execução de um loop.
- [E] impossibilitar o aninhamento de loops não lógicos.

Instruções de salto

Instrução	Exemplo
break	Termina as instruções dos ciclos for, while, do-while ou a instrução switch onde se encontra Utilizada quando é necessário terminar um ciclo sem verificar a condição de paragem
continue	Passa à frente o restante pedaço do corpo do ciclo for, while ou do-while onde esta instrução se encontra Utiliza-se quando é necessário ignorar o restante código do ciclo que precede esta instrução

Desenvolvimento de sistemas

[Questão 16] Os serviços de gerenciamento, oferecidos pelo contêiner EJB (Enterprise JavaBeans), são de:

- [A] Transações. Persistência. Ciclo de Vida. Segurança.
- [B] Transições. Pertinência. Ciclo de Vida. Risco.
- [C] Transformações. Persistência. Ciclo de Projeto. Segurança.
- [D] Transações. Comunicação. Ciclo de Vida. Mercado.
- [E] Transações. Consistência. Fases. Segurança.

Desenvolvimento de sistemas

[Questão 16] Os serviços de gerenciamento, oferecidos pelo contêiner EJB (Enterprise JavaBeans), são de:

- **[A] Transações. Persistência. Ciclo de Vida. Segurança.**
- [B] Transições. Pertinência. Ciclo de Vida. Risco.
- [C] Transformações. Persistência. Ciclo de Projeto. Segurança.
- [D] Transações. Comunicação. Ciclo de Vida. Mercado.
- [E] Transações. Consistência. Fases. Segurança.

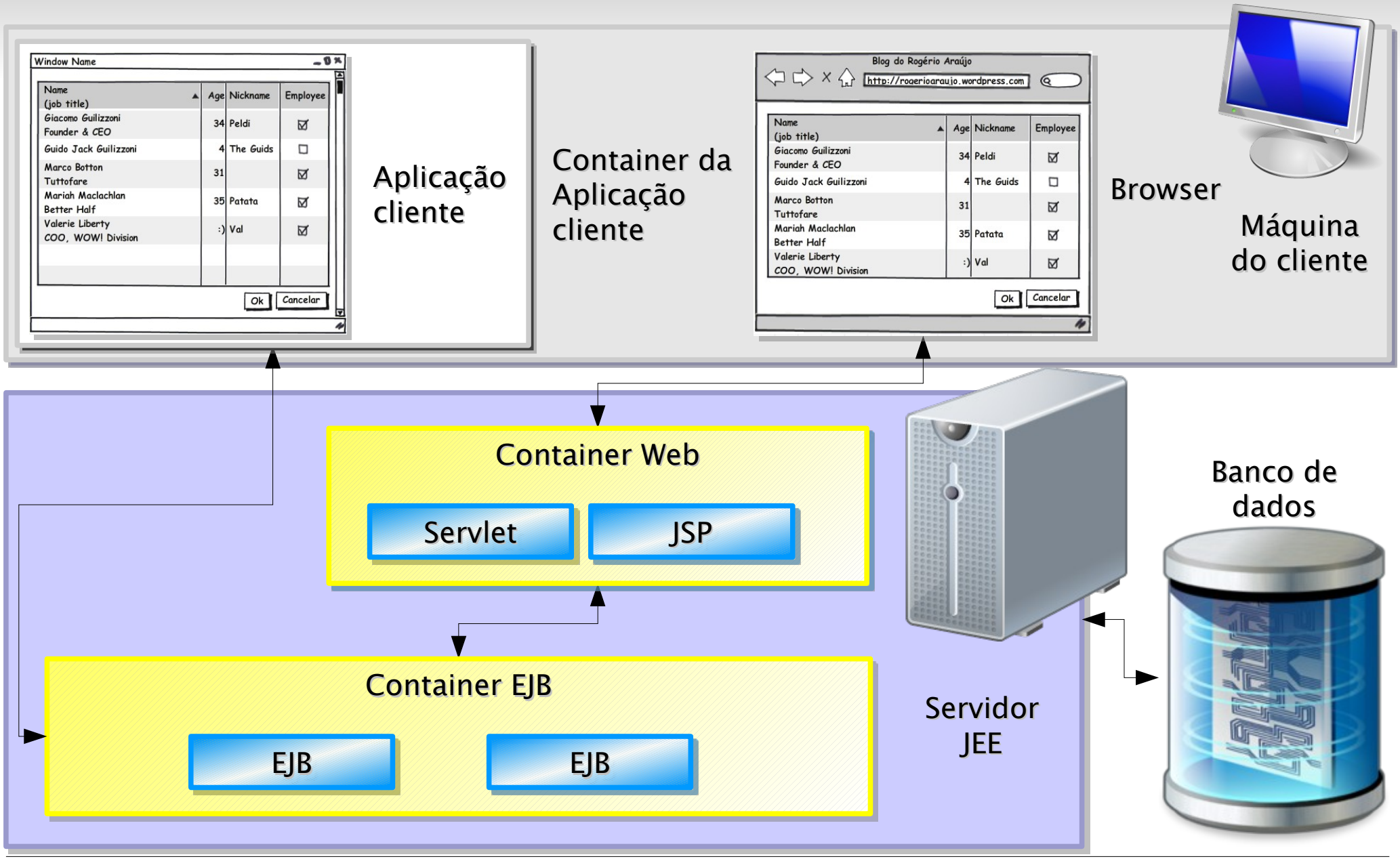
Containers JEE

- O container também gerencia os **serviços não configuráveis**:
  - Ciclo de vida de:
    - EJBs
    - Servlets
  - Pooling de conexões de banco de dados
  - Persistência de dados
  - Acesso às APIs da plataforma

Containers JEE

- **Tipos**
  - **Container Enterprise JavaBeans (EJB)**
    - Gerencia a execução de **EJBs**
  - **Container Web**
    - Gerencia a execução de:
      - **Páginas JSP**
      - **Servlets**
  - **Container da Aplicação cliente**
    - Gerencia a execução de componentes da **aplicação cliente**
  - **Containter de Applet**
    - Gerencia a execução de **applets**
    - Consiste em um navegador da Web e Java Plug-in em execução juntos no cliente

Containers JEE



Containers JEE x colmeias de abelhas

Componente

Container

Servidor

Desenvolvimento de sistemas

[Questão 17] Os níveis da plataforma J2EE são:

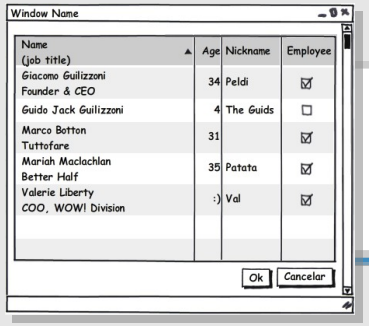
- [A] Patrocinador. Web. Negócios. Sistemas de Computação Corporativos.
- [B] Cliente. Web. Negócios. Sistemas de Informação Corporativos.
- [C] Cliente. Interno. Externo. Negócios.
- [D] Fornecedor. Web. Político. Sistemas de Informação Camada.
- [E] Cliente. Stakeholders. Negócios. Background corporativo.

Desenvolvimento de sistemas

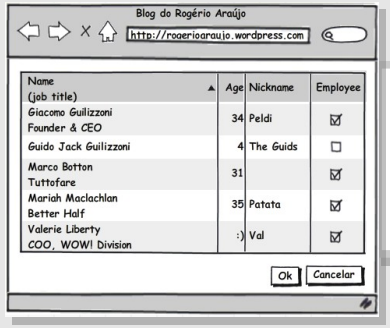
[Questão 17] Os níveis da plataforma J2EE são:

- [A] Patrocinador. Web. Negócios. Sistemas de Computação Corporativos.
- **[B] Cliente. Web. Negócios. Sistemas de Informação Corporativos.**
- [C] Cliente. Interno. Externo. Negócios.
- [D] Fornecedor. Web. Político. Sistemas de Informação Camada.
- [E] Cliente. Stakeholders. Negócios. Background corporativo.

Modelo de aplicação JEE



Aplicação JEE 1
Aplicação cliente



Aplicação JEE 2
Páginas web dinâmicas

Camada cliente

Máquina do cliente

Camada web

Servlet | JSP | JSF

Camada de negócios

EJB | EJB | EJB

Servidor JEE

Camada EIS

Servidor de BD

Desenvolvimento de sistemas

[Questão 18] Assinale a opção correta.

- [A] A API de solicitação de MDD é usada para consultar um registrador MDD por condições de acesso.
- [B] A APL de atualização de UDDI é usada para consultar um usuário UDDI por informações sobre localização de uma empresa.
- [C] A UDDI é usada para manter a consistência de registradores API de propriedade de uma empresa.
- [D] A API de solicitação de UDDI é usada para consultar um registrador UDDI por informações sobre uma empresa.
- [E] A API de solicitação de UDDL é usada para consultar um usuário de UDDL por informações sobre interesses de negócio de uma empresa.

Desenvolvimento de sistemas

[Questão 18] Assinale a opção correta.

- [A] A API de solicitação de MDD é usada para consultar um registrador MDD por condições de acesso.
- [B] A APL de atualização de UDDI é usada para consultar um usuário UDDI por informações sobre localização de uma empresa.
- [C] A UDDI é usada para manter a consistência de registradores API de propriedade de uma empresa.
- **[D] A API de solicitação de UDDI é usada para consultar um registrador UDDI por informações sobre uma empresa.**
- [E] A API de solicitação de UDDL é usada para consultar um usuário de UDDL por informações sobre interesses de negócio de uma empresa.

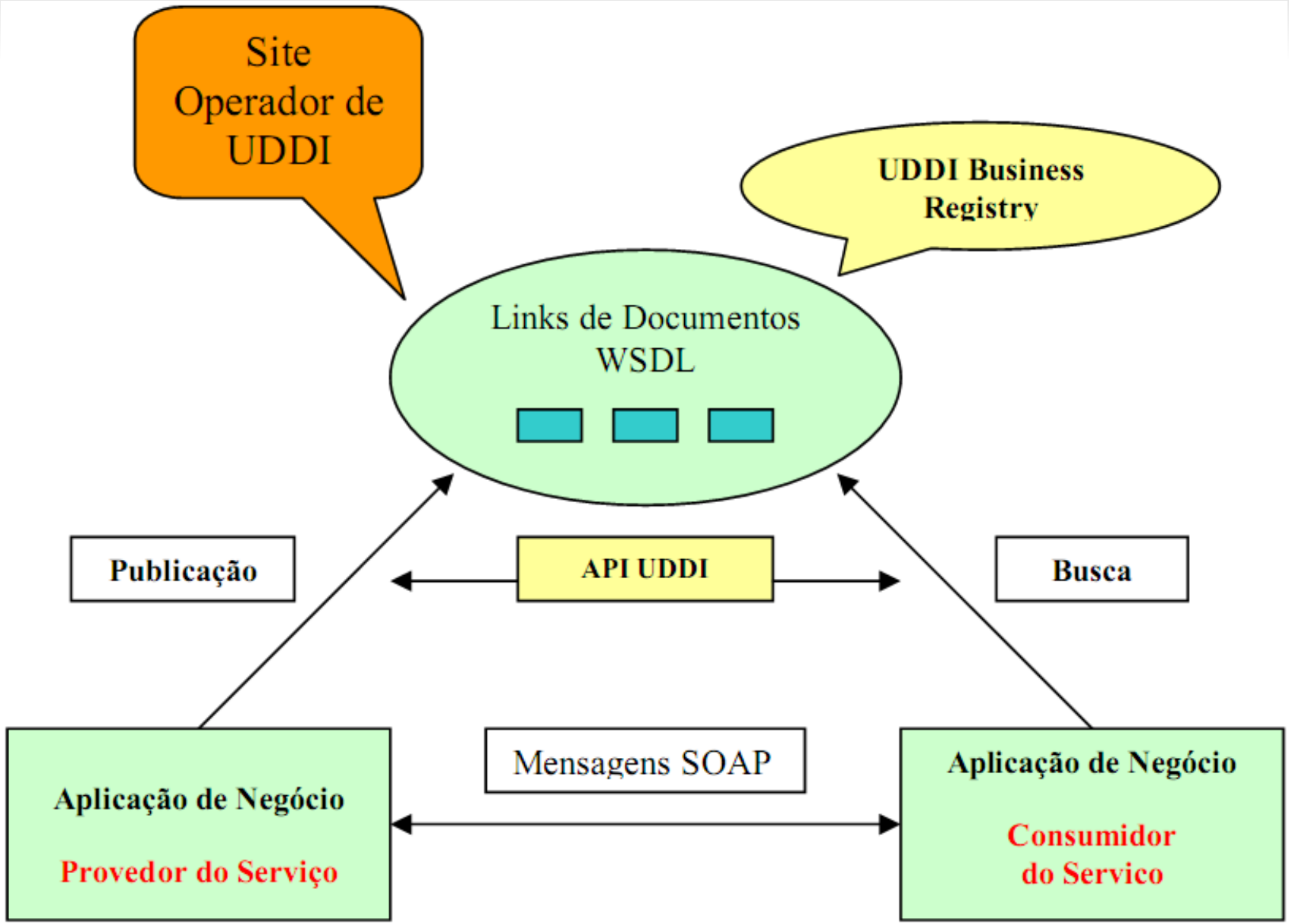
UDDI

- Significa **Universal Description, Discovery and Integration**
- É um serviço de diretório onde as empresas podem se registrar e procurar por Web Services
- É um diretório:
  - Para armazenar informações sobre os serviços web
  - De interfaces de serviços web descritas por WSDL
- Comunica via UDDI SOAP
- Está incorporada no Microsoft.NET
- Usa padrões da W3C e da ITEF (Internet Engineering Task Force), como XML, HTTP e DNS
- Usa o WSDL para descrever interfaces de web services

UDDI

- A **arquitetura técnica de UDDI** consiste de **três partes**:
  - **Modelo de Informação UDDI**
    - Um esquema XML para descrever:
      - Negócios
      - Serviços Web
  - **API UDDI**
    - Uma API baseada em SOAP para publicação e busca de informação UDDI
  - **UDDI Business Registry (UDDI cloud services)**
    - Sites-operadores que:
      - Provêem implementações da especificação UDDI
      - Sincronizam todos os dados sobre uma "scheduled basis"

UDDI


Site
Operador de
UDDI
UDDI Business
Registry
Links de Documentos
WSDL
Publicação
API UDDI
Busca
Aplicação de Negócio
Provedor do Serviço
Mensagens SOAP
Aplicação de Negócio
Consumidor
do Servico

UDDI

- A **UDDI API Specification** descreve as **APIs** de:
  - **Publicação (publishing)**
    - Suporta a operação publish que habilita empresas a colocarem e atualizarem a informação em um registro UDDI
  - **Busca (inquiry)**
    - Suporta a operação **find**, que habilita consumidores de serviços a navegarem num sistema UDDI para pesquisar registros de provedores de serviços que oferecem um determinado serviço ou tipo de serviço
    - Qualquer pessoa pode usar a API de busca para realizar consultas sobre o **UDDI Business Registry (UBR)**

Desenvolvimento de sistemas

[Questão 19] O padrão de projeto singleton é usado para restringir

- [A] a instanciação de uma classe para objetos simples.
- [B] a instanciação de uma classe para apenas um objeto.
- [C] a quantidade de classes.
- [D] as relações entre classes e objetos.
- [E] classes de atributos complexos.

Desenvolvimento de sistemas

[Questão 19] O padrão de projeto singleton é usado para restringir

- [A] a instanciação de uma classe para objetos simples.
- **[B] a instanciação de uma classe para apenas um objeto.**
- [C] a quantidade de classes.
- [D] as relações entre classes e objetos.
- [E] classes de atributos complexos.

Desenvolvimento de sistemas

[Questão 20] A definição de que um sistema deve ser desenvolvido em três níveis é feita pelo padrão de projeto

- [A] MVC (Model View Controller).
- [B] MVC-Dev (Model Value Constructive Development).
- [C] TMS (Time Milestones Setting).
- [D] PMC (Project Main Controller).
- [E] MCA (Model Classes Assignment).

Desenvolvimento de sistemas

[Questão 20] A definição de que um sistema deve ser desenvolvido em três níveis é feita pelo padrão de projeto

- **[A] MVC (Model View Controller).**
- [B] MVC-Dev (Model Value Constructive Development).
- [C] TMS (Time Milestones Setting).
- [D] PMC (Project Main Controller).
- [E] MCA (Model Classes Assignment).

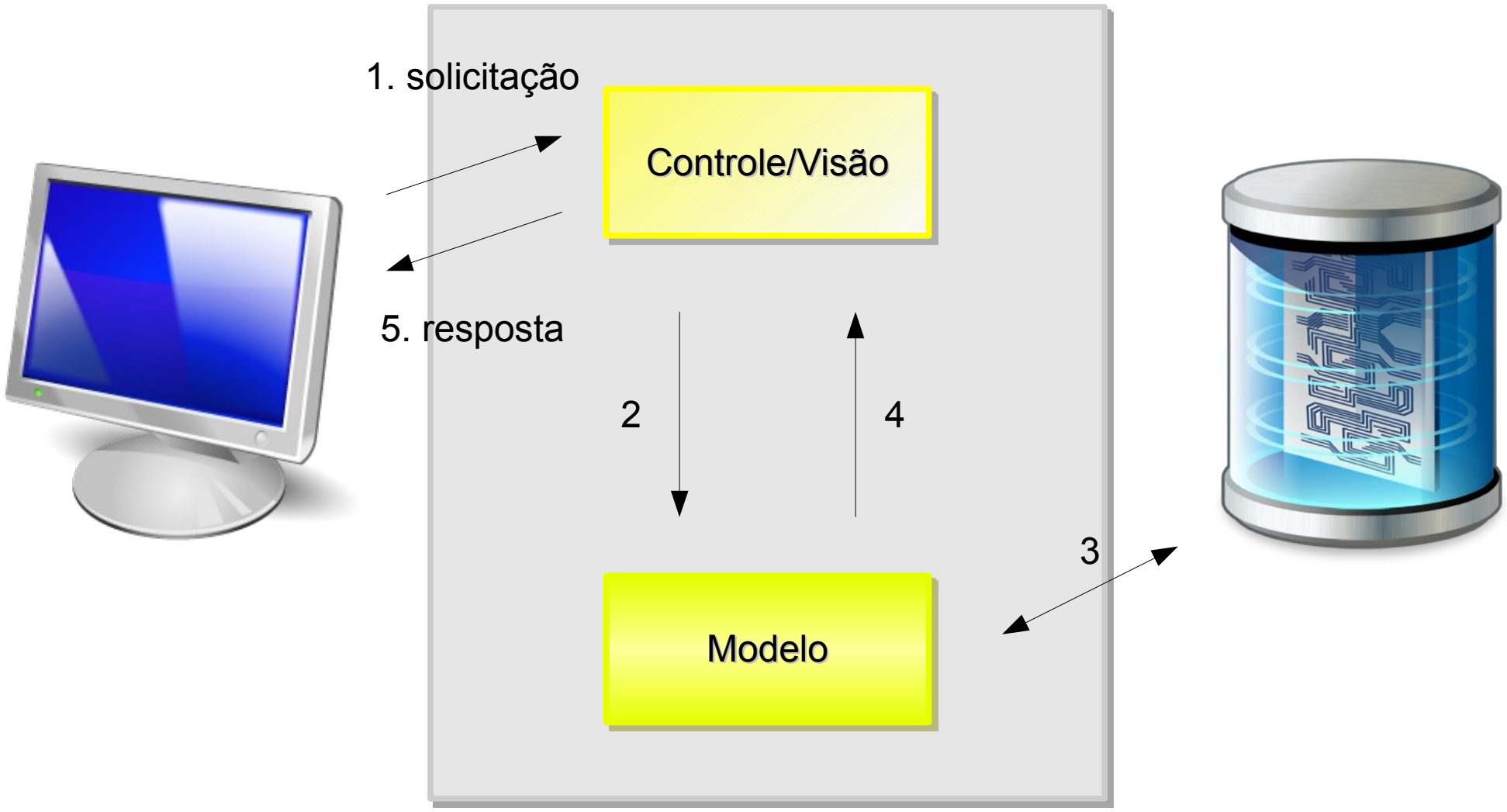
Padrão MVC

- Significa Modelo-Visão-Controle
- Originalmente veio do projeto Smalltalk-80
- É muito similar ao modelo PAC
  - Presention-Abstraction-Control
- O propósito do MVC é decompor o sistema em 3 subsistemas
- O MVC também pode ser chamado de estilo de arquitetura baseada em componentes
- Cada módulo nessa arquitetura tem sua própria responsabilidade
- Permite que membros da equipe de diferentes habilidades possam trabalhar no seu módulo específico

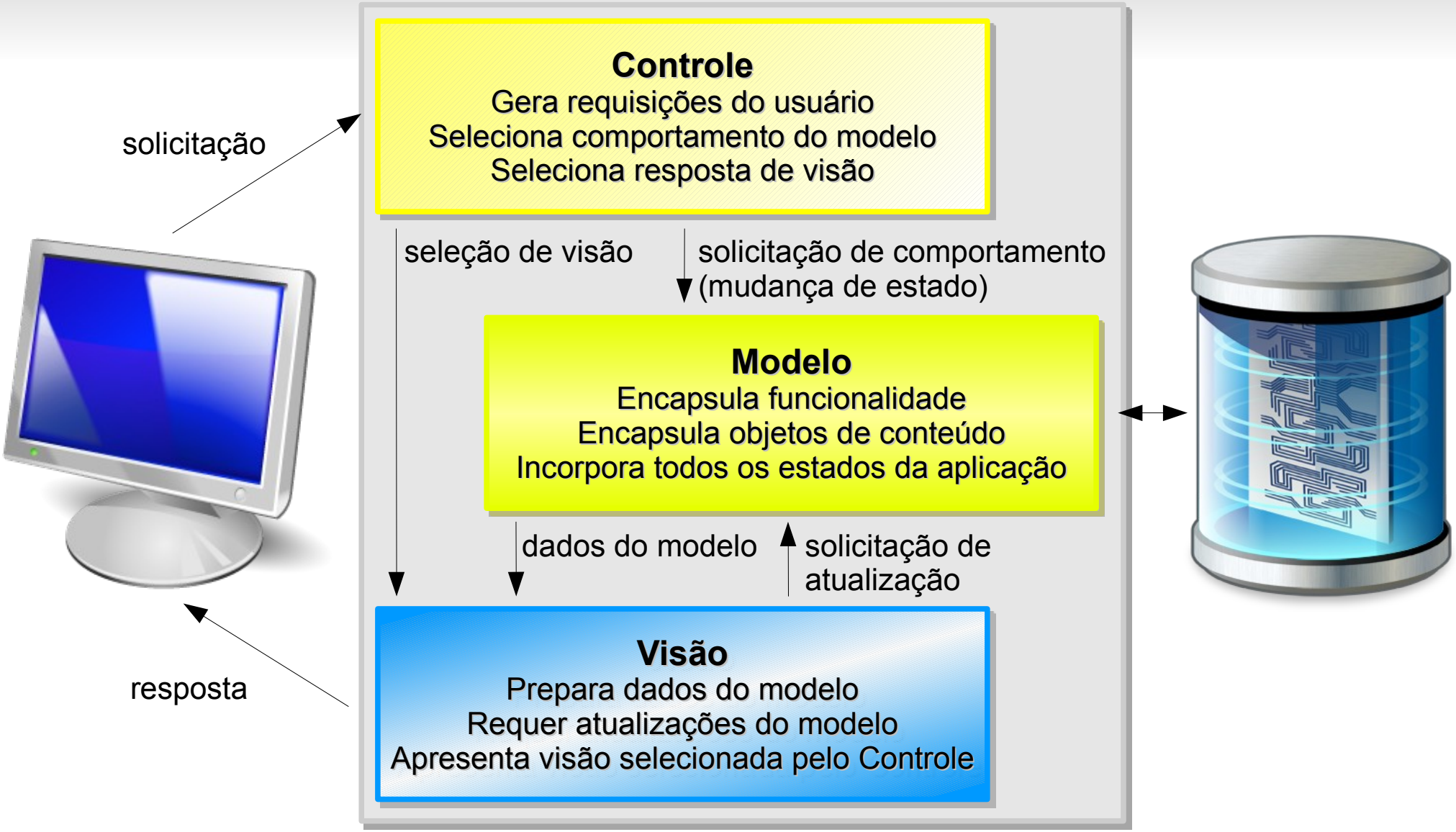
Padrão MVC

- **Tipos ou Modelos**:
  - **MVC Tipo 1**
    - Usado em **projetos simples**
    - Combina os módulos Controle e Visão em um só módulo para o processamento de entrada e saída
    - O módulo Modelo cuida da questão de dados
  - **MVC Tipo 2**
    - Indicado para **projetos mais complexos**
    - O módulo Controle é dedicado para processamento de solicitações de usuários
    - A separação clara entre apresentação e processamento de dados e de solicitação resulta em uma divisão bem-definida dos papéis e das responsabilidades da equipe que desenvolverá o projeto usando esse tipo do MVC

Padrão MVC



Padrão MVC



Engenharia de Software – 6ª Edição – Roger Pressman

Desenvolvimento de sistemas

[Questão 21] O padrão de projeto Factory provê uma classe de decisão que retorna

- [A] um objeto de uma de suas subclasses, sem fixação de parâmetros.
- [B] um atributo de uma de suas classes conexas, com base em um parâmetro reservado.
- [C] um objeto de uma de suas subclasses, com base em um parâmetro recebido.
- [D] um atributo de uma de suas classes conexas, sem fixação de parâmetros.
- [E] um objeto de uma de suas subclasses, com parâmetros fatorados.

Desenvolvimento de sistemas

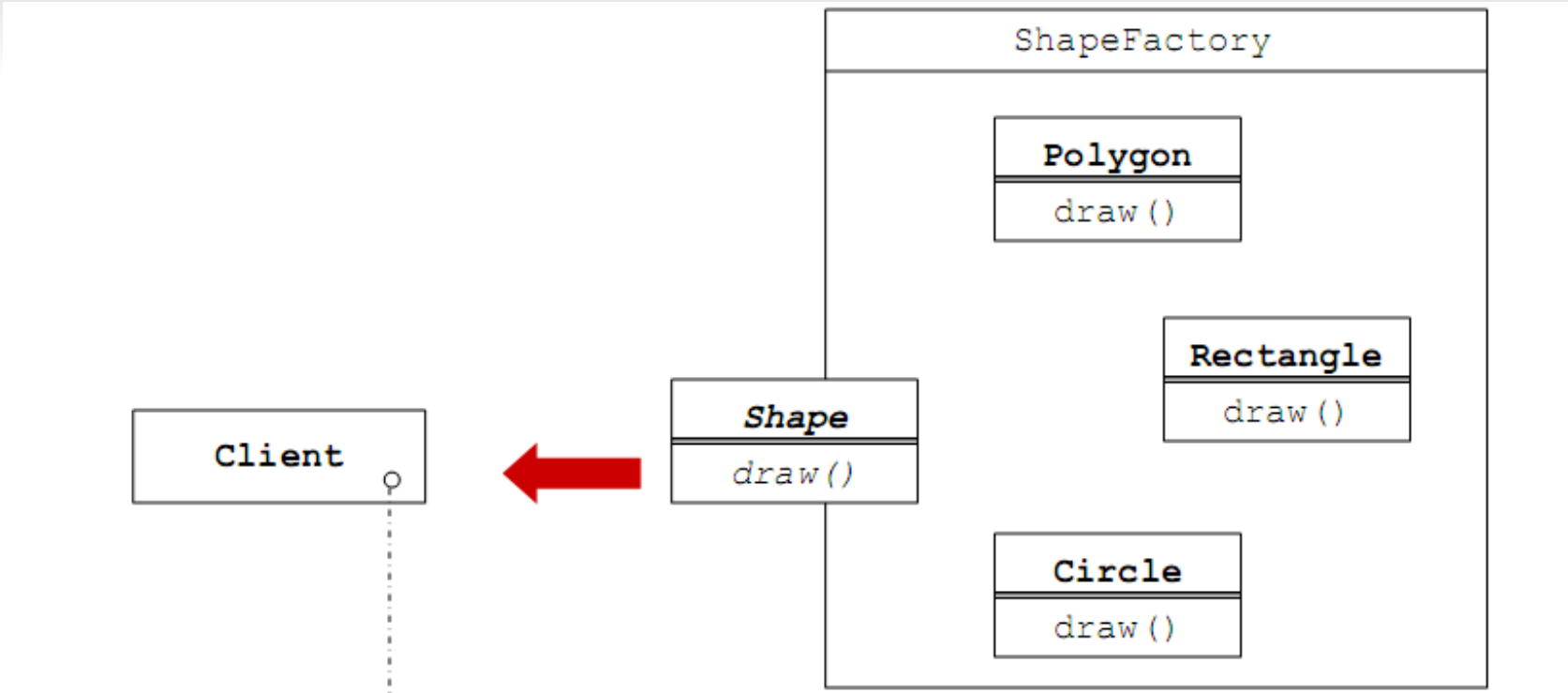
[Questão 21] O padrão de projeto Factory provê uma classe de decisão que retorna

- [A] um objeto de uma de suas subclasses, sem fixação de parâmetros.
- [B] um atributo de uma de suas classes conexas, com base em um parâmetro reservado.
- **[C] um objeto de uma de suas subclasses, com base em um parâmetro recebido.**
- [D] um atributo de uma de suas classes conexas, sem fixação de parâmetros.
- [E] um objeto de uma de suas subclasses, com parâmetros fatorados.

Factory Method

- GoF
  - Define uma interface para criar um objeto
    - Mas deixar que subclasses decidam que classe instanciar
  - Permite que uma classe delegue a responsabilidade de instanciamento às subclasses

Factory Method




```
Shape shape = new Rectangle();
Shape shape = ShapeFactory.getShape("rect");
shape.draw();
```

return new Rectangle()
neste contexto

```
public static Shape getShape(String type) {
   ShapeFactory factory = (ShapeFactory)typeMap.get(type);
   return factory.getShape(); // non-static Factory Method
}
```

Desenvolvimento de sistemas

[Questão 22] Para indicar a visibilidade da propriedade, a UML

- [A] incorpora um prefixo a um nome de atributo ou nome de operação.
- [B] incorpora um sufixo a um nome de atributo ou origem de operação.
- [C] gera um nome de atributo e nome de transação totalmente distinto do anterior.
- [D] duplica nome de atributo ou nome de operação.
- [E] sublinha o nome de atributo ou nome de operação.

Desenvolvimento de sistemas

[Questão 22] Para indicar a visibilidade da propriedade, a UML

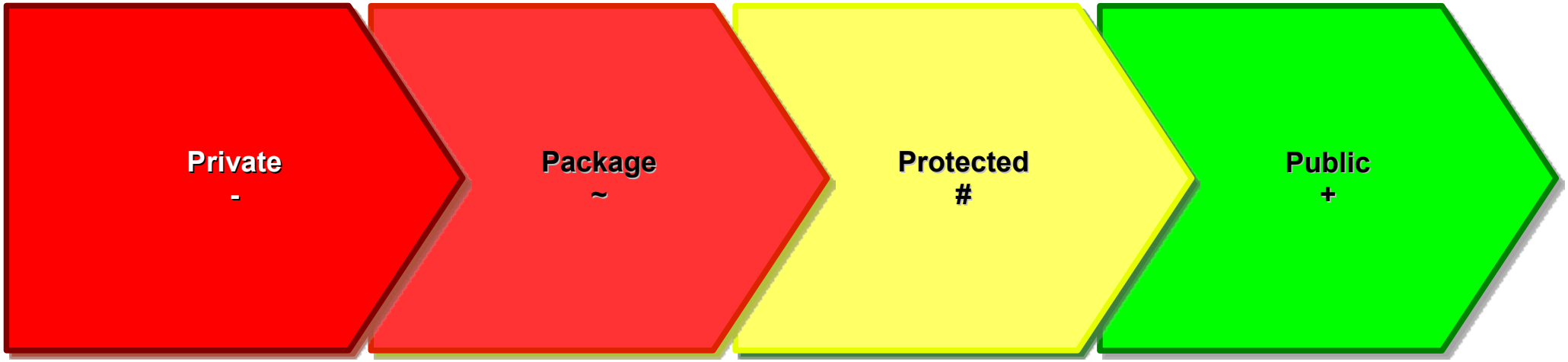
- **[A] incorpora um prefixo a um nome de atributo ou nome de operação.**
- [B] incorpora um sufixo a um nome de atributo ou origem de operação.
- [C] gera um nome de atributo e nome de transação totalmente distinto do anterior.
- [D] duplica nome de atributo ou nome de operação.
- [E] sublinha o nome de atributo ou nome de operação.

Classes e objetos

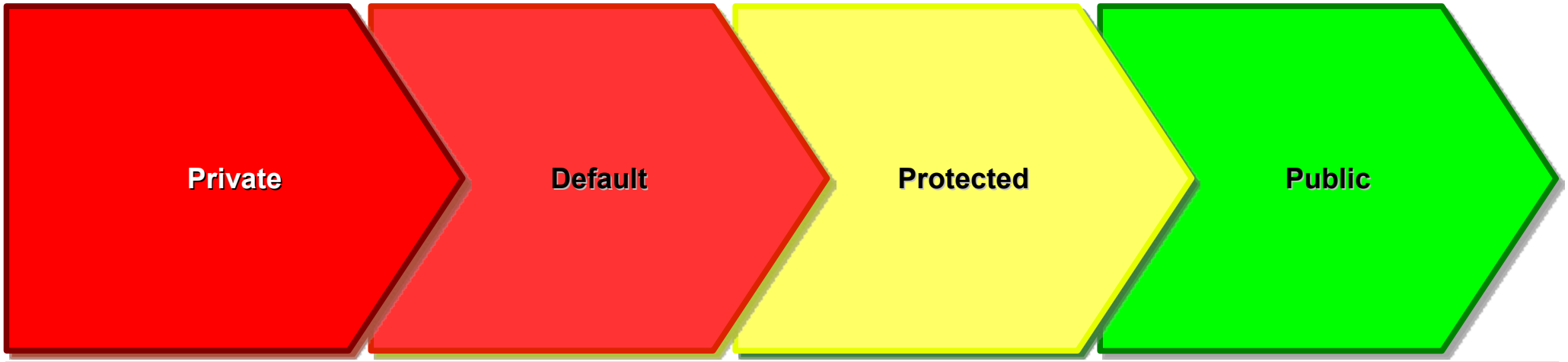
Modificador	Classe	Subclasse	Pacote	Todos
Public UML Símbolo +	☑	☑	☑	☑
Public Java	☑	☑	☑	☑
Protected UML Símbolo #	☑	☑		
Protected Java	☑	☑	☑	
Private UML Símbolo -	☑			
Private Java	☑			
Package UML Símbolo ~	☑		☑	
Default Java	☑		☑	

Classes e objetos

Ordem de visibilidade na UML (do mais restrito para o mais liberal)



Ordem de visibilidade na Java (do mais restrito para o mais liberal)



Desenvolvimento de sistemas

[Questão 23] O Diagrama de Estado mostra

- [A] os estados expressos que os objetos de uma dada classe podem assumir e as transformações entre pares de classes.
- [B] os estados admissíveis que os atributos de uma dada classe podem modificar e os pares de estados mais relevantes.
- [C] os estados de atualização que os objetos de qualquer classe podem assumir e as transições permitidas entre instâncias.
- [D] os estados admissíveis que os objetos de uma dada classe podem assumir e as transições permitidas entre pares de estados.
- [E] os estados coerentes com os objetos priorizados e as restrições de transições entre pares de estados.

Desenvolvimento de sistemas

[Questão 23] O Diagrama de Estado mostra

- [A] os estados expressos que os objetos de uma dada classe podem assumir e as transformações entre pares de classes.
- [B] os estados admissíveis que os atributos de uma dada classe podem modificar e os pares de estados mais relevantes.
- [C] os estados de atualização que os objetos de qualquer classe podem assumir e as transições permitidas entre instâncias.
- **[D] os estados admissíveis que os objetos de uma dada classe podem assumir e as transições permitidas entre pares de estados.**
- [E] os estados coerentes com os objetos priorizados e as restrições de transições entre pares de estados.

Desenvolvimento de sistemas

[Questão 24] Uma associação em UML representa

- [A] uma população variada de relações (engagements) de redundâncias entre instâncias de classe.
- [B] uma população variada de vínculos (links) de relacionamentos entre instâncias de classe.
- [C] uma classificação de vínculos (links) de relacionamentos entre classes de atributos.
- [D] uma população constante de valores (values) de relacionamentos quantitativos entre atributos de instâncias.
- [E] uma estrutura de equivalências (equal features) entre relacionamentos de instâncias de posicionamento de classes.

Desenvolvimento de sistemas

[Questão 24] Uma associação em UML representa

- [A] uma população variada de relações (engagements) de redundâncias entre instâncias de classe.
- **[B] uma população variada de vínculos (links) de relacionamentos entre instâncias de classe.**
- [C] uma classificação de vínculos (links) de relacionamentos entre classes de atributos.
- [D] uma população constante de valores (values) de relacionamentos quantitativos entre atributos de instâncias.
- [E] uma estrutura de equivalências (equal features) entre relacionamentos de instâncias de posicionamento de classes.

Referências

- APIs de Busca e Publicação em UDDI: http://www.inf.ufsc.br/~bosco/ensino/ine5626/uddi/UDDI_5.pdf
- GoF Design Patterns em Java: http://www.argonavis.com.br/cursos/java/j930/
- LARMAN, Craig. Utilizando UML e Padrões. 3ª Edição. Editora Bookman, 2007.
- PAGE-JONES, Meilir. Fundamentos do desenho orientado a objeto com UML. Editora Pearson Education, 2001.
- PRESSMAN, Roger. Engenharia de Software. 6ª Edição. Editora McGraw-Hill, 2006.

Referências

- SANTOS, Rildo F. Desenhado Componente de Software com UML: http://www.slideshare.net/Ridlo/desenhando-componentes-de-software-arquitetura-de-software