Fundação CECIERJ - **Vice Presidência de Educação Superior a Distância**

Curso de Tecnologia em Sistemas de Computação
Disciplina de Arquitetura e Projeto de Sistemas II
Gabarito – AD2 2° semestre de 2011.

Nome –

Observações:

1. Prova com consulta.

Atenção: Como a avaliação à distância é individual, caso sejam constatadas semelhanças entre provas de alunos distintos, **será atribuída a nota ZERO** a TODAS as provas envolvidas. As soluções para as questões podem ser buscadas por grupos de alunos, mas a redação final de cada prova tem que ser distinta. ALÉM DISSO, às questões desta AD respondidas de maneira muito semelhantes às respostas oriundas dos gabaritos já publicados de ADs e APs de períodos anteriores, **será atribuída a nota ZERO**, incluindo também cópias diretas e sem sentido de tópicos dos slides das aulas.

Questão 1 [1 ponto]

Cite algum padrão GRASP que contribui para fazer com que as classes pertençam a somente um domínio (recordar as aulas de Princípios de Projeto OO). Justifique.

Resposta:

Padrão *Pure Fabrication* (Invenção Pura). Este padrão indica a criação de classes artificiais sempre que todas as opções ferem o princípio de alta coesão. Ferir o princípio de alta coesão pode significar que a classe esta pertencendo a mais de um domínio ao mesmo tempo.

Questão 2 [3 pontos]

O padrão *Polymorphism* visa tratar alternativas em função do tipo da classe e também criar componentes de software substituíveis.

a) [1 ponto] Descreva um exemplo, **diferente daquele visto em aula**, de uma situação onde esse padrão é útil.
b) [1 ponto] Quais são os benefícios alcançados com o uso desse padrão?

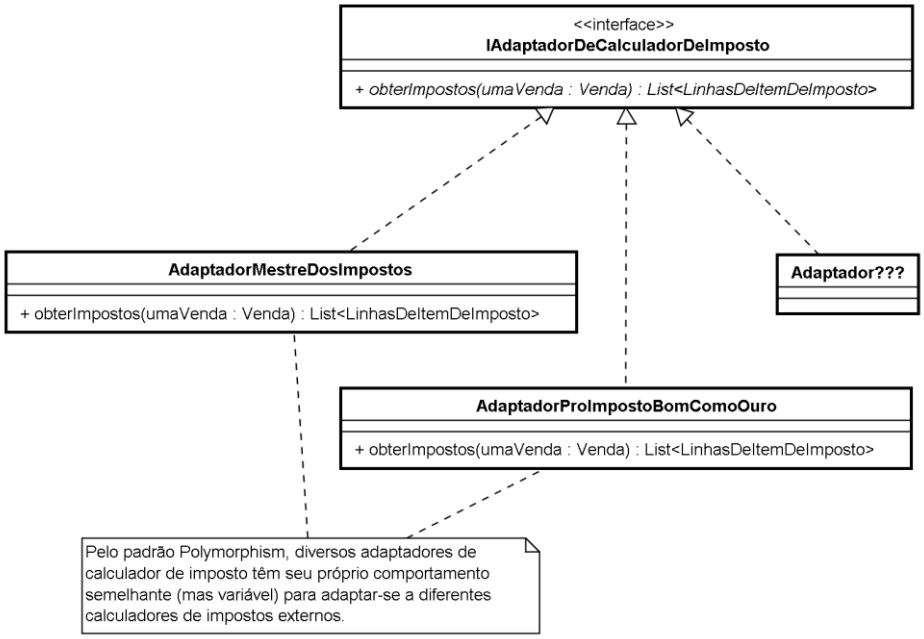
c) [1 ponto] Desenhe um modelo de classes exibindo como o padrão pode ser utilizado na situação descrita.

Resposta:

a) No aplicativo de PDV, podem existir vários calculadores de impostos terceirizados, que precisam ser apoiados (e.g., *Tax-Master* ou Mestre-dos-Impostos, e *Good-As-Gold-Tax-Pro* ou Pro-Imposto-Bom-Como-Ouro), e o sistema precisa se integrar com diversos deles. Cada calculador de imposto tem uma interface diferente e, portanto, há um comportamento similar, mas variado, a ser adaptado para cada uma dessas interfaces fixas ou APIs. Um produto pode apoiar um protocolo de soquete TCP puro, outro pode oferecer uma interface SOAP e um terceiro pode apresentar uma interface RMI Java.

b) Facilidade de manutenção e facilidade de inserção de um novo tipo de autorização.

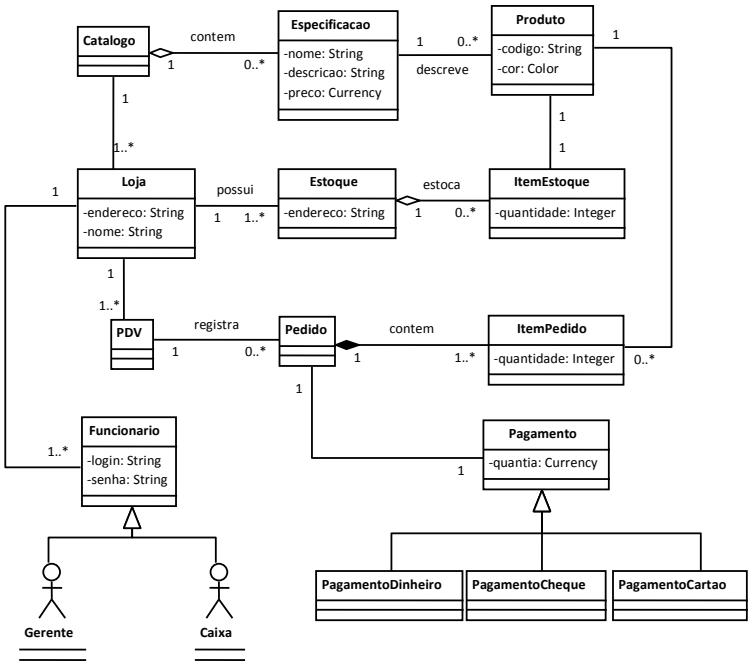
c) Diagrama de classes requerido:



Como o comportamento das adaptações do calculador varia de acordo com o tipo de calculador, por meio do padrão *Polymorphism*, deve-se atribuir a responsabilidade da adaptação aos próprios objetos calculadores externos ou o adaptador para o calculador. Ao se enviar uma mensagem para o objeto local, será feita uma chamada ao calculador externo em sua API nativa. Cada método *obterImpostos* recebe o objeto *Venda* como parâmetro para que o calculador possa analisar a venda. A implementação de cada método *obterImpostos* será diferente: *AdaptadorMestreDosImpostos* vai adaptar a solicitação à API do *Tax-Master*, e assim por diante.

Questão 3 [3 pontos]

Considere os padrões GRASP e o modelo a seguir:



a) Responda com suas palavras qual classe deve ser a responsável pela criação de objetos da classe `ItemPedido`? Justifique a sua resposta citando os padrões utilizados nessa tomada de decisão.

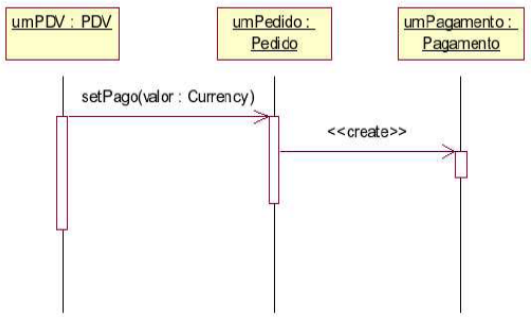
b) Forneça um diagrama de sequência que exiba a atribuição de responsabilidades para o pagamento de um pedido, representada pelo método `paga(valor : Currency, umPedido : Pedido)` a ser criado na classe PDV.

c) Responda com suas palavras qual padrão GoF poderia ser adotado na implementação do sistema para que exista somente uma instância de `Catalogo` para toda a rede de lojas?

Resposta:

a) De acordo com os padrões *Creator* e *Low Coupling*, a classe `Pedido` deve ser a responsável por criar objetos da classe `ItemPedido`. Isso ocorre porque a responsabilidade de criação de objetos da classe B deve ser atribuída à classe A se A contém objetos de B, favorecendo assim a manutenção do baixo acoplamento do sistema.

b) Diagrama de sequência requerido:



c) Padrão *Singleton*. O propósito do padrão *Singleton* é assegurar que uma classe tenha somente uma instância, e fornecer um acesso global a essa instância.

Questão 4 [3 pontos]

Em relação às arquiteturas web, responda:

a) Em uma situação onde se deseja carregar uma página uma única vez e evitar interações adicionais com o servidor, qual estilo é mais indicado: *Scripted Client* ou AJAX? Justifique a sua resposta.
b) Discuta com as suas palavras a diferença entre arquitetura em 1, 2 e 3 camadas.
c) Quais são as camadas de uma arquitetura em 3 camadas e como essas camadas devem interagir para atender às requisições do cliente?

Resposta:

a) O estilo mais indicado é o *Scripted Client*. O estilo *Scripted Client* irá carregar todas as informações necessárias de uma só vez, fazendo com que a carga inicial da página seja mais demorada, mas que não sejam necessárias novas interações com o servidor. Por outro lado, de acordo com o estilo AJAX, a carga inicial seria mais rápida, mas seriam necessárias interações adicionais com o servidor para obter informações complementares.

b) Arquiteturas em uma camada, também conhecidas como arquiteturas cliente-servidor, assumem que a única responsabilidade do servidor é o armazenamento de dados. Por outro lado, arquiteturas em duas camadas também transferem para o servidor a responsabilidade de apresentação das informações. Nessas duas arquiteturas, não fica definido com precisão qual camada é responsável por processar as regras de negócio. Finalmente, arquiteturas em três camadas definem claramente, no lado servidor, os elementos responsáveis por armazenamento, processamento das regras de negócio e apresentação das informações.

c) As camadas de uma arquitetura em três camadas são: *apresentação*, *aplicação* e *armazenamento*. Usualmente, o servidor recebe solicitações do cliente via camada de aplicação, que consulta a camada de armazenamento para processar os dados de acordo com as regras de negócio. Finalmente, a camada de apresentação é acionada para construir a página com a resposta referente à solicitação.