Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

Curso de Tecnologia em Sistemas de Computação
Disciplina: Engenharia de Software
AD2 2° semestre de 2010.

Atenção: Como a AD é individual, caso seja constatado que provas de alunos distintos são cópias uma das outras, independentemente de qualquer motivo, a todas será atribuída nota ZERO. As soluções para as questões podem sim, ser buscadas por grupos de alunos, mas a redação final das respostas para as questões da prova tem que ser individual!

ADS enviadas pelo correio, devem ser postadas cinco dias antes da data final de entrega estabelecida no calendário de entrega de ADs.

1. Modelagem em UML: Considerando o problema da loja de venda de móveis abaixo, apresente (7,0):

O gabarito para esta questão é dependente do modelo produzido pelos alunos. Portanto, as respostas abaixo representam apenas uma das possíveis soluções. O mais importante deste tipo de questão é permitir ao aluno aplicar os conceitos de modelagem ao mesmo tempo que pode **discutir com o tutor sobre sua solução**!

a. a descrição de 1 caso de uso (1,0 ponto)

Os casos de uso representam casos de interação entre atores e o sistema, descrevendo os passos a serem executados para se atingir um determinado objetivo. Procure identificar o que o sistema deve fazer.
Na descrição de casos de uso, devem-se descrever todos os passos da interação, sempre descrevendo o que será feito e não como. Por exemplo: "Sistema verifica saldo do cliente" e não "Sistema verifica saldo do cliente na tabela conta no bando de dados de produção".
Na descrição, é importante relacionar os atores participantes, as pré e pós condições, Requisitos atendidos, regras de negócio atendidas, além dos passos do caso de uso de sucesso e alternativos (exceções e caminhos alternativos).

b. o diagrama de classes para o sistema (1,5 pontos)

O diagrama de classes depende da perspectiva (ponto de vista) utilizada pelo engenheiro. Entretanto, os seguintes conceitos principais deveriam estar representados de acordo com alguma abstração utilizada:

- Cliente
- Vendedor
- Móvel

- Nota Fiscal
- Compra

Uma boa maneira de identificar classes em um documento é ler atentamente e identificar substantivos no documento. Em geral, estes quando relacionados ao domínio do problema podem ser classes do seu sistema. **Atenção** não são todos os substantivos classes conceituais do sistema. Outra dica é representar os atores como classe, porém isto nem sempre é verdadeiro. Em geral, tente verificar se a classe criada, que correspondente a um ator, tem algum relacionamento com outra classe do sistema, senão pode haver um equívoco. Pois este ator pode ser um elemento externo ao sistema e não deveria estar representado no diagrama de classes conceitual. **Atenção**: Não é que todo ator não possa ser uma classe conceitual, mas existem casos onde algum ator não é uma classe, pois é um elemento externo.

Prestem atenção para as classes de associação que podem existir. Em geral, elas surgem em relacionamento entre dois tipos de classes que requerem não só informações sobre as classes participantes, mas também sobre a própria relação. Um exemplo é a relação entre empresa e funcionário, o seu relacionamento contrato tem informações próprias como data de início, data fim, etc.

c. o diagrama de seqüência para o caso de uso descrito em a) (1,0 ponto)

Novamente, a estrutura final do diagrama de seqüência depende de como o engenheiro definiu os conceitos (classes). Dicas: as classes existentes nos diagramas de sequência TÊM que corresponder às classes do diagrama de classes. Se um objeto envia mensagem para outro objeto, tem que existir uma relação entre eles nos diagramas de classe. Ainda, a mensagem enviada tem que compor a relação de serviços do objeto (classe) que receber a mensagem.

d. Indique que técnicas de inspeção podem ser usadas para revisar estes diagramas, descrevendo os critérios que você usou para escolher e os tipos de defeito que podem ser encontrados. Dê um exemplo aplicando a técnica nos diagramas que você criou (1,0 ponto)

Três tipos de técnica de inspeção poderiam ser usados para revisar estes modelos. A mais simples, *ad-hoc*, deve ser aplicada seguindo a percepção e experiência do inspetor. Uma intermediária, baseada em *checklist*, pode também ser aplicada. Entretanto, neste caso, um *checklist* deve ser configurado e calibrado antes da aplicação. Em complemento, e de forma mais elaborada, podem ser aplicadas técnicas de leitura. Neste caso, temos disponível as OORTs, composta por 7 diferentes técnicas de leitura construídas especificamente para tratar modelos descritos em UML. Estas técnicas podem ser usadas para identificar defeitos de omissão, fato incorreto, inconsistência, ambigüidade e informação estranha.

e. Apresente o conjunto de casos de testes para realizar o teste do caso de uso que você descreveu em a) e modelou em c), indicando qual abordagem foi utilizada para gerar esta informação (1,5 pontos)

> Você deve ter aplicado uma das abordagens, ou combinação delas, para gerar os casos de teste para o caso de uso que você descreveu e modelou. As abordagens podem ser particionamento por equivalência, analise valor-limite ou grafo de causa-efeito.

f. Utilize as métricas NOC, DIT e CBO para identificar as classes que poderiam ser mais propensas à falha no seu projeto. Indique as classes e o motivo desta indicação (1,0 ponto)

> Você tem que extrair as medidas de NOC, DIT e CBO. As classes mais propensas a falha serão aquelas que tiverem, no projeto, os maiores valores para estas métricas, ou seja, quanto Mario o valor destas métricas no projeto, maior a chance daquelas classes apresentarem defeitos.

1) O cliente telefona para a loja e informa que deseja comprar um móvel. O vendedor pergunta sobre a quantidade desejada, informando o preço unitário do móvel desejado ao cliente;

2) Se o cliente confirmar a compra, o funcionário verifica se ele já tem cadastro na loja. Caso não tenha cadastro, o funcionário pergunta o nome completo do cliente, seu endereço completo (rua, complemento, CEP, bairro, cidade, estado e país), telefone fixo e telefone de contato;

3) Cadastrado o cliente e confirmada a compra, o vendedor emite a nota de serviço, que indica os dados do cliente, o nome do vendedor, a data da compra, o mobiliário desejado (com suas respectivas quantidades e preços). A nota é encaminhada ao almoxarifado da loja;

4) Ao receber uma nota de serviço, o almoxarife verifica a existência de estoque para os produtos desejados. Se não existir estoque para algum móvel, o almoxarife emite um pedido de compra para o fornecedor do móvel, comprando sempre duas unidades além do que seria necessário para atender ao pedido do cliente;

5) O cliente pode cancelar a compra até dois dias depois de realizada. Se uma compra for cancelada, os produtos voltam para o estoque e os pedidos de compra que tiverem sido emitidos para estes produtos são cancelados junto ao fornecedor;

6) Quando os produtos estiverem disponíveis, o almoxarife emite a nota fiscal e encaminha os produtos para entrega ao cliente. A nota fiscal possui os mesmos dados da nota de serviço, além de indicar seu número e a data em que foi emitida.

2. Explique a diferença entre as linguagens de programação interpretadas e as linguagens de programação compiladas. (valor 1,0 ponto)

   Programas interpretados são transformados em uma linguagem intermediária – em geral mais simples que a linguagem de programação em si – e executados em um interpretador ao invés de diretamente sobre uma plataforma computacional. Eles tendem a ter menor desempenho, dado que disputam processador com o próprio interpretador que lhes executa, mas podem ser executados em uma diversidade de plataformas computacionais, desde que estas tenham versões do interpretador.

   Programas construídos em uma linguagem compilada são transformados em código nativo de uma determinada plataforma computacional e podem ser executados diretamente nesta plataforma. Eles tendem a apresentar maior desempenho (pois são executados diretamente), mas também uma limitação de ser executados somente nesta plataforma.

3. Explique o que é uma inversão de comando na codificação de um projeto de software (valor: 1,0 ponto; máximo: 5 linhas).

   A inversão de comando ocorre quando uma rotina recebe um parâmetro de controle, como um flag, por exemplo, que determina como a rotina deve agir. Dizemos que é uma inversão de comando porque um agente externo, que chama a rotina, determina como a rotina deve atuar, quebrando sua independência. A inversão de comando geralmente dificulta o entendimento da rotina e do sistema.

4. Ao discutirmos os princípios de projeto de software, vimos um ciclo onde requisitos geram a demanda por mais software, que entra em operação, provoca mudanças no ambiente de trabalho, mudanças estas que acabam gerando novos requisitos. Explique porque este ciclo ocorre e como normalmente é encerrado. (Valor: 1,0 ponto; máximo: 10 linhas)

   Um software normalmente é construído de acordo com os requisitos de seus futuros usuários. No entanto, quando o software entra em operação, ele altera o ambiente de trabalho, mudando a forma com que seus usuários trabalham. Desta forma, o software geralmente precisa ser adaptado para a nova realidade, da qual ele participou na formação. Esta é uma das razões pela qual um software usado está em constante alteração, sendo ajustado a medida que os processos que ele suporte evoluem. O ciclo é encerrado quando o sistema não atende mais as necessidades dos seus usuários e a manutenção e/ou evolução demanda muito esforço e custo, tornando-se, neste caso, mas viável a construção de um novo produto.