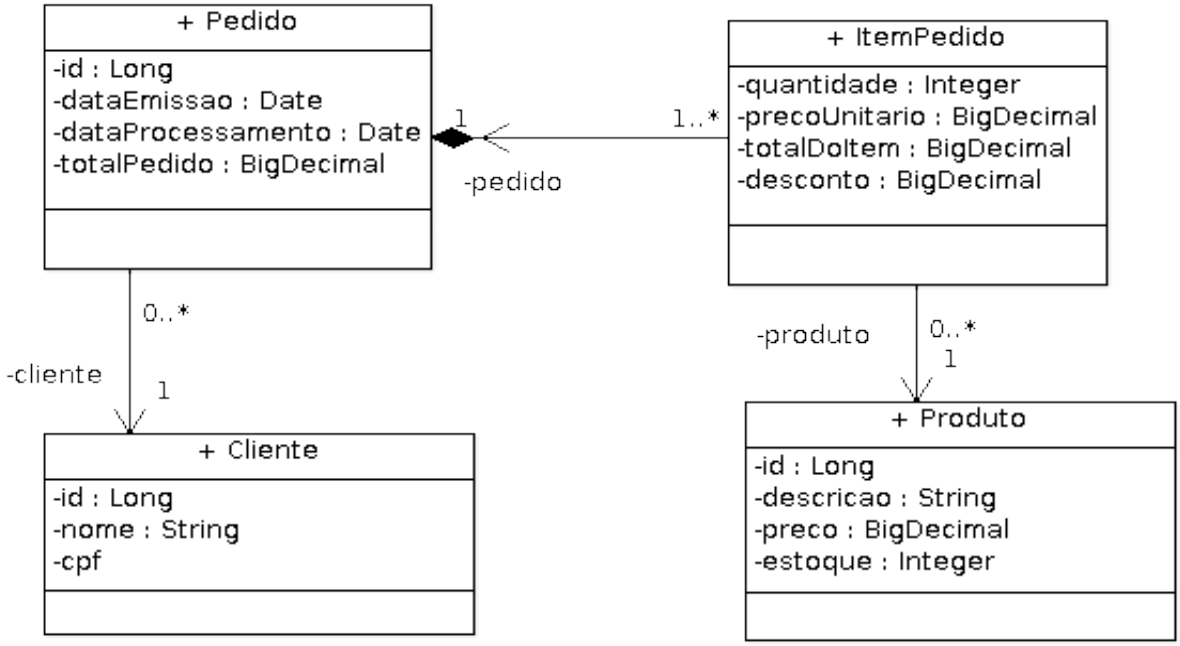
FIAP
Faculdade de Informática e Administração Paulista
Sistemas de Informações

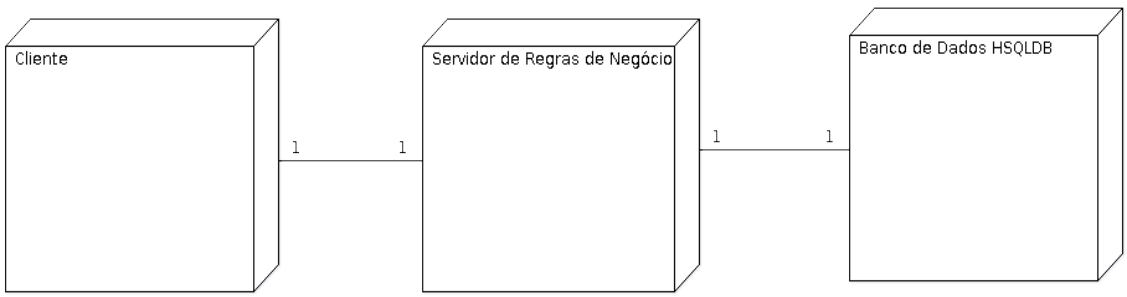
NAC20 CONJUGADA
Sistemas Distribuídos / Técnicas de Programação 3
Entrega até: 12/06/2012
Grupos de 5 a 10 pessoas

Objetivo do trabalho
Implementar um sistema que corresponda à arquitetura descrita abaixo:

Modelo de domínio



Arquitetura de distribuição



- Cliente – Aplicativo modo CONSOLE que se comunica com o servidor de regras de negócio.
- Servidor de Regras de Negócio – Servidor Socket TCP que recebe comandos do cliente e consulta o banco de dados. Implementar o acesso usando Hibernate e, no caso de quem não faz TP3, implementar usando JDBC.
- Banco de Dados HSQLDb – Máquina onde está o HSQLDB.
- O sistema deverá obrigatoriamente funcionar distribuido em 3 máquinas (3 IPs diferentes).

O sistema deverá implementar uma interface com o usuário em modo console que atenda aos seguintes requisitos:

Regras gerais

- Deve ser feito em Java.
- Considere que são obrigatórios todos os campos para os quais não existam regras especiais.
- Campos Date são “data e hora”.
- Chave-primária é ID, caso não haja regra específica.
- Implementem corretamente as associações.
- Respeitar as cardinalidades.
- Todos os campos devem ser persistidos.
- É responsabilidade do grupo definir os modelos de interação com o usuário.

Regras específicas

1. **Cadastro de cliente**
   (a) CRUD
      i. Create
         A. Escolha um tipo de dados adequado para CPF. CPF não pode se repetir entre dois clientes.
      ii. Retrieve
         A. Implementar pesquisa por ID, NOME (trecho inicial) e CPF. Retornar todos os dados, sempre em ordem alfabética.
      iii. Update
         A. Não permitir alterar o CPF de clientes que possuam pedidos.
      iv. Delete
         A. Não permitir excluir clientes que possuam pedidos.

2. **Cadastro de Produto (CRUD)**
   (a) Gerais
      i. Estoque nunca pode ser negativo.
      ii. Preço deve ser maior que zero.
   (b) CRUD
      i. Create
         A. Estoque inicial tem que ser pelo menos 10.
      ii. Retrieve
         A. Pesquisar produto por ID e trecho inicial da DESCRIÇÃO. Retornar todos os dados, sempre em ordem alfabética.
      iii. Update
         A. Sem regras especiais.
      iv. Delete
         A. Não permitir excluir produtos para os quais existam pedidos (de qualquer tipo).

3. Cadastro de Pedido e ItemPedido

(a) Gerais
   i. Um pedido pode ser criado, salvo, editado, salvo, editado... etc... até que seja feito o processamento (ver “Processamento do Pedido”)
   ii. ItemPedido
      A. Quantidade sempre maior que zero.
      B. Valor deve ser o valor do produto naquele momento.

(b) CRUD
   i. Create
      A. A data do pedido é sempre a data atual.
      B. A data de processamento inicialmente é NULL.
      C. A chave-primária de ItemPedido deve ser {Pedido,Produto}.
   ii. Update
      A. Permitir incluir, editar e excluir APENAS itens de pedidos não processados (Data de processamento = null).
      B. Cliente não pode ser alterado após o pedido ser salvo.
   iii. Retrieve
      A. Pesquisar todos os dados de um pedido pelo seu número (retornar ID, data de emissão (dd/mm/aaaa), data de processamento (dd/mm/aaaa), total do pedido e quantidade de produtos)
      B. Pesquisar todos os pedidos de um certo cliente (retornar ID, data de emissão (dd/mm/aaaa), data de processamento (dd/mm/aaaa), total do pedido e quantidade de produtos)
   iv. Delete
      A. Não podem ser excluidos pedidos processados.

(c) REGRAS ADICIONAIS
   i. Processamento do Pedido
      A. Deve haver um comando para processar o pedido. Processar o pedido define a data e hora de processamento e impede que sejam feitas alterações no pedido.
      B. Se a pessoa comprar mais que 10 unidades de um produto, ganha 2% de desconto naquele item.
      C. Ao processar um pedido, o estoque dos produtos deve ser atualizado.
      D. O processamento de um pedido não pode deixar o Produto com estoque negativo. Se isto for ocorrer, colocar NULL na quantidade do ItemPedido correspondente. Isto indica que aquele item não pode ser atendido.

<u>Deverão ser entregues</u>

1. Arquivo ZIP contendo:
   (a) Os códigos-fonte do projeto (incluindo arquivos de mapeamento e de configuração do Hibernate);
   (b) JARs executáveis.
      i. A única dependência dos aplicativos deverá ser da JVM e a biblioteca de classes do Hibernate. Não deverá ser exigido nem NetBeans nem Eclipse;
   (c) Arquivo alunos.txt contendo:
      i. Identificação da TURMA;
      ii. Identificação de cada aluno integrante do grupo (RM, Nome)
      iii. Descrição de ajustes no modelo, se foi feita alguma.

2. Como entregar:
   (a) Encaminhar, até a data determinada, e-mail para fabian.martins@gmail.com contendo no assunto 2012.NAC20.<turma>.<RM do integrante do grupo>.<nome do integrante do grupo>. O integrante do grupo constante no nome do arquivo será o daquele responsável por enviar o trabalho.
   (b) O arquivo zip deverá ser nomeado da mesma forma que o assunto da mensagem.
      i. Exemplo: 2012.NAC20.4SIS.99199.Francisco Bento.zip
         A. veja que é apenas um (1) RM e Nome que será utilizado para nomear o arquivo ZIP.
   (c) Não será permitida a adição de integrantes dos grupos posteriormente à entrega.

Serão considerados no processo de avaliação:

- Existência de cópia de trabalho entre grupos;
- Qualidade do código (identação, documentação, qualidade na definição das responsabilidades das classes);
- Execução.
- Funcionalidades entregues (entrega mais funcionalidades corretas, ganha mais pontos);
- Funcionalidades não solicitadas contam como ponto negativo (over scoping);