**Prof.ª Ana Cristina Barreiras Kochem Vendramin**
Universidade Tecnológica Federal do Paraná (UTFPR), Câmpus Curitiba
Engenharia de Computação, Sistemas de Informação
Programa de Pós-Graduação em Computação Aplicada (PPGCA)
Departamento Acadêmico de Informática (DAINF)

**Avaliação** (valor 2,5)
**Arquitetura Cliente-Servidor**
**Modelo *Publisher/Subscriber* (Eventos e Notificações)**

## Mercado da Bolsa de Valores

Desenvolver uma aplicação de compra, venda, cotações e notificações de limite de ganho e limite de perda de ações em uma bolsa de valores.

- Utilizar uma *middleware* para ambientes homogêneo como **Java RMI** (*Remote Invocation Method*) ou **PyRO** (*Python Remote Objects*) para prover a comunicação entre um grupo de **acionistas/clientes** e um ***home broker*/servidor**;

## Servidor/*Home broker*

O servidor provê os seguintes métodos aos seus acionistas:

- Inserir ou remover ações em/de uma lista de interesse de ações (chamada *cotações*). Essa lista servirá para obter cotações de ações quando for solicitado (valor 0,2);
- Compra e venda de ações do mercado da bolsa de valores. Será transacionada a quantidade de ações assim que aparecer um par de ordens onde o preço de venda seja igual ao preço de compra. Quando uma ordem de compra ou venda for concluída, o servidor deverá informar ao cliente, de forma assíncrona (servidor chamará o método notificar do cliente), as seguintes informações: código da ação, quantidade negociada e o valor transacionado. Após efetuar a compra de uma ação em nome de um cliente, o servidor irá inserir essa ação na *lista cotações* do cliente (valor 0,5);
- Obter as cotações correntes das ações que estão na lista *cotações* (valor 0,2);
- Registrar interesse em receber notificações de eventos (alerta) quando a cotação de ações de seu interesse atingir um limite de ganho ou um limite de perda previamente cadastrados pelo acionista. O servidor manterá uma lista de *limites* de ações para cada cliente interessado. Para cada ação experimentando um evento, o servidor atua como *publisher* e envia notificações assíncronas a todos os acionistas que

subscreveram para este tipo de evento. Essa notificação se dará na forma de uma chamada de método do servidor no cliente passando nessa chamada uma mensagem de notificação do evento (valor 0,5).

**Cliente/Acionista**

Um processo cliente pode invocar as seguintes operações no servidor:

- Inserir e remover ações da lista *cotações*. Nessas chamadas, o cliente deve informar apenas o código do ativo de interesse. Essa lista conterá um conjunto de ações cujo preço o cliente deseja monitorar (não somente as ações que possui em sua carteira, mas qualquer outra ação que seja do seu interesse) (valor 0,2);
- Comprar e vender ações. Na ordem de compra ou venda, o cliente deve informar ao servidor o código do ativo, a quantidade que deseja negociar, o preço máximo que deseja pagar (método compra) ou o preço mínimo que deseja receber (método venda) pelo ativo e o prazo máximo em que a ordem (compra/venda) permanecerá no estado condicional (valor 0,3);
- Obter cotações atualizadas sobre as ações do seu interesse. O cliente apenas chamará o método obter cotações (valor 0,2);
- Registrar interesse em receber **notificações assíncronas** de eventos de uma ou mais ações de seu interesse. Nesse caso, o cliente atua como *subscriber*, devendo informar ao servidor, o código da ação, o limite de ganho e o limite de perda (valor 0,2).

O processo cliente fornece o seguinte método ao servidor:

- Notificar eventos (valor 0,2). Esse método serve para o cliente receber mensagens de notificações de eventos do servidor. Ao receber uma notificação, o cliente irá mostrar a mensagem na tela. O servidor chamará esse método do cliente nas seguintes situações:
  - Quando uma ação de interesse do cliente atingir seu limite de ganho ou seu limite de perda;
  - Quando uma ordem de compra ou venda disparada pelo cliente for concluída com sucesso.

Como a aplicação em questão será *multithread*, é possível que algum recurso possa ser acessado por duas ou mais *threads* ao mesmo tempo. Por esse motivo, se faz necessário a implementação de um mecanismo que controle o acesso concorrente aos recursos (ações) compartilhados.

Observações:

- Desenvolva uma interface gráfica com recursos de interação apropriados;
- Adicionar comentários no próprio código sobre as variáveis e métodos de sua aplicação;
- Desenvolvimento da aplicação pode ser individual ou em dupla.