Projeto Orientado em Computação

Implementação de um Web Service para Busca de Cotações na Bolsa de Valores de São Paulo

Aluno: Henrique Velloso Ferreira Melo
Orientadora: Mariza Bigonha

Introdução

As antigas fronteiras de mercado estão desaparecendo e o mundo atual se encontra cada vez mais globalizado. Não existem mais fatos isolados. Um pequeno acontecimento em um lado do globo pode afetar negociações no lado oposto. Os investidores de hoje não podem mais ter uma visão local do mercado. Para se ter lucro real, é necessário aplicar em ações no mundo inteiro.

Juntamente com a globalização, avança também a robotização e a informática. A Internet se espalha com incrível velocidade pelo mundo, aproximando os povos, culturas e idéias. O mundo dos negócios nunca mais foi o mesmo depois da grande rede.

Os investidores ganharam uma poderosa arma com a Internet. Agora é possível acompanhar em tempo real o sobe e desce das ações em todo o mundo, efetuando compras e vendas a milhares de quilômetros. O investidor moderno não pode ficar longe desta tecnologia e a informática produz ferramentas ideais para isto.

Atualmente existem inúmeros sites contendo informações sobre as ações nas centenas de bolsas espalhadas pelo mundo. Vários destes sites são gratuitos e apresentam diversas informações sobre os ativos, contendo inclusive gráficos e históricos.

O grande problema é que a busca por estas informações é manual. Para se manter atualizado, o investidor precisa a todo tempo atualizar a página de seu navegador contendo as cotações atualizadas. Existem programas no mercado que fazem a atualização em tempo real, mas a maioria destes programas é cara e depende da conexão com um servidor que fornece as informações constantemente.

A idéia inicial deste projeto era a criação de um sistema de busca de cotações da Bolsa de Valores de São Paulo, e criação de gráficos sobre estes valores. A objetivo do projeto continua o mesmo, entretanto, a forma de implementação será alterada. Projetaremos agora um Web Service para disponibilizar os dados. Web Services são sistemas instalados em servidores que recebem e retornam dados para os clientes, de acordo com parâmetros enviados. Toda a troca de dados é feita utilizando a porta 80, normalmente com protocolo HTTP. Implementaremos o sistema de busca de cotações fazendo uso de Web Services, e para isso teremos que fazer modificações adicionais no projeto, como por exemplo, a alteração da linguagem de programação utilizada. A alteração foi feita pois durante os estudos sobre o tema, tomamos um maior conhecimento da tecnologia por trás dos Web Services e vimos que era totalmente adequada à implementação de nosso projeto. Além disso, descobrimos que esta é uma nova

tecnologia que vem sendo largamente empregada na atualidade, devido a sua interoperabilidade e acessibilidade. Dessa forma, surgiu grande interesse por nossa parte em aprofundar nossos conhecimentos nessa tecnologia, que certamente é um novo paradigma da computação e será muito utilizada nos próximos anos.

Na primeira parte deste documento, falaremos sobre a nova proposta numa visão geral. Veremos que este trabalho é apenas uma implementação parcial de um grande projeto, que certamente exigirá anos de estudos e desenvolvimento.

Na segunda parte, explicaremos detalhadamente os itens que serão implementados. Destaca-se a explicação do funcionamento de Web Services e a motivação para seu uso. Também será descrito o porquê de modificar a linguagem de implementação para C# e uma descrição o uso que será feito dos bancos de dados.

As dificuldades encontradas e soluções propostas serão descritas na terceira parte. Na quarta parte falamos sobre os projetos para o futuro.

Na quinta e última parte do documento estão as conclusões do projeto, detalhando o que foi aprendido e as perspectivas de implementação para o próximo semestre.

PARTE 1

A Nova Proposta

Na introdução deste documento, explicamos de forma resumida porque houve a modificação da proposta. Ao descrever os conceitos de Web Service, deixaremos mais claro o motivo da opção por esta alternativa de implementação. Uma explicação detalhada sobre Web Services será dada na próxima parte do documento. Nesta parte, nos limitaremos a explicar o que será implementado na nova proposta, que como veremos, é apenas uma parcial de um projeto maior e mais ambicioso.

1.1 Descrição da Nova Proposta

Na figura abaixo, podemos ter uma visão geral do sistema a ser implementado.

Podemos observar a existência de dois aplicativos. Um deles é executado no cliente e o outro no servidor. Estes dois aplicativos se comunicam via Web Service.

O aplicativo cliente é executado na máquina do usuário e é responsável pela construção de gráficos de acompanhamento de informações de ativos.

O aplicativo será configurável, de forma que o usuário terá a opção de escolher os gráficos que deseja visualizar, além da periodicidade destes. Mais detalhes sobre a implementação deste aplicativo serão dados na próxima parte neste documento.

A comunicação entre o aplicativo cliente e o aplicativo servidor se dá através de um Web Service instalado no servidor, que recebe um pedido do cliente na porta 80, na forma de uma mensagem no formato do protocolo HTTP. Esta mensagem se assemelha a uma página ASP. Além do cabeçalho HTTP, vários parâmetros podem ser passados no formato GET ou POST. Pode também ser passado no corpo da página ASP, um documento no formato XML, que contém os métodos que queremos acessar no servidor.

Uma descrição detalhada da tecnologia de Web Services será dada na próxima parte deste documento. Dentro do escopo do nosso projeto, podemos dizer que o aplicativo cliente passa os parâmetros configurados pelo usuário para o servidor. Estes parâmetros são definidos de acordo

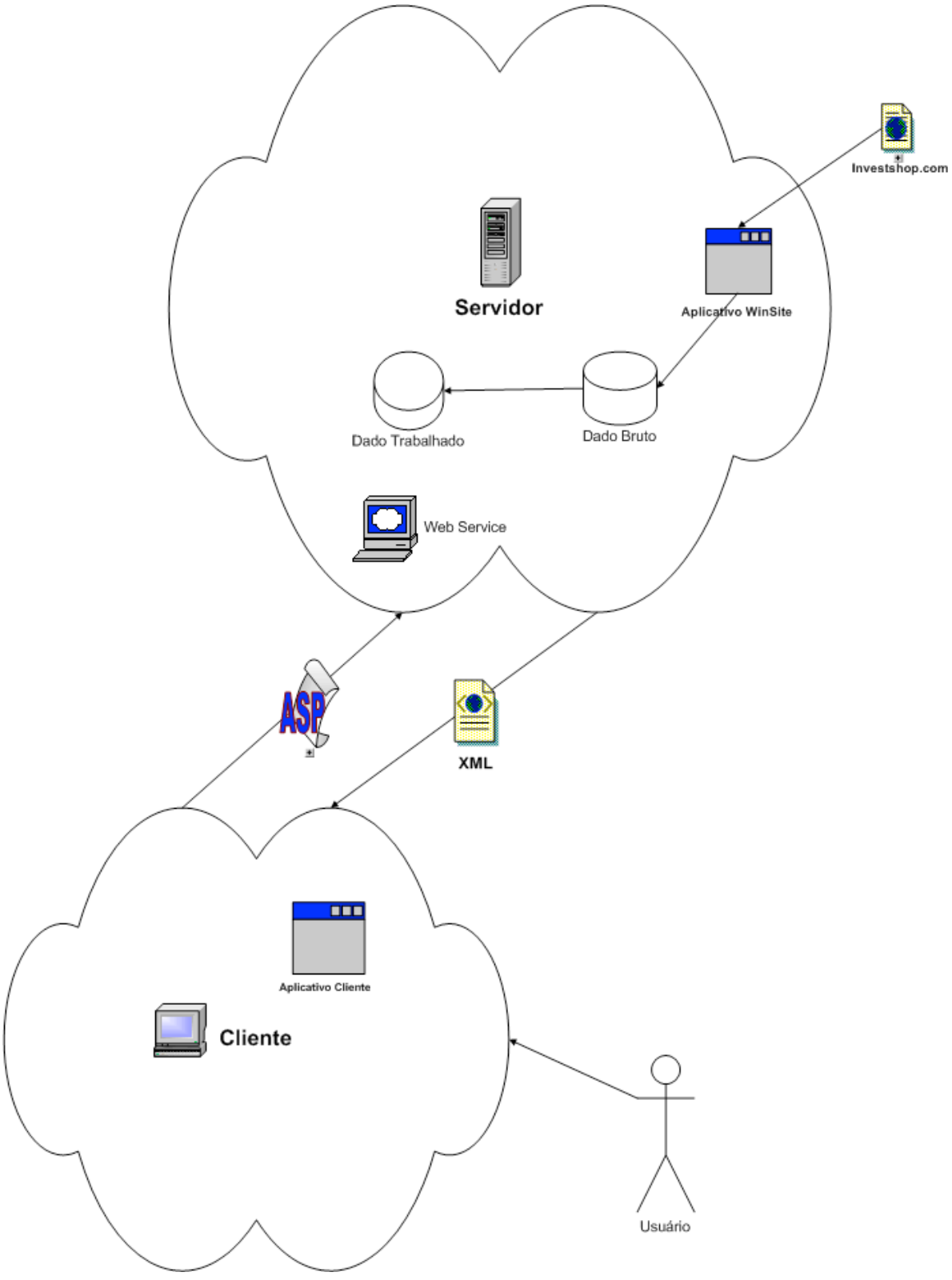

Figura 1 – Estrutura do Projeto

com as opções escolhidas pelo usuário no aplicativo cliente. O usuário poderá escolher entre as seguintes opções: o ativo que deseja acompanhar, um conjunto de gráficos para este ativo e a periodicidade dos gráficos. Este conjunto de informações compõe um *pedido*, que é passado para o Web Service em um formato pré-definido.

O Web Service trata os pedidos do cliente chamando rotinas no aplicativo servidor. A partir dos parâmetros contidos, o aplicativo servidor busca as informações requisitadas no Banco de Dados Trabalhado. Um usuário poderia pedir, por exemplo, o acompanhamento do ativo da Petrobrás, através dos gráficos MACD e IFR, com a periodicidade de 15 minutos. O aplicativo servidor faria uma busca no Banco de Dados Trabalhado por estes ativos. Quando encontra o ativo desejado, o aplicativo executa *stored procedures* para gerar infomações relevantes à criação dos gráficos requisitados. Além disso, baseando em informações históricas presentes no banco de dados, é possível recuperar dados dentro da periodicidade indicada.

Os dados recuperados são retornados via Web Service para o aplicativo cliente. Agora este estará pronto para executar sua segunda função: exibir os dados para o usuário na forma de gráficos.

O aplicativo servidor, além de fazer buscas no banco de dados, tem duas outras importantes funcionalidades: requisição da página HTML contendo os dados brutos, e filtragem dos dados brutos do banco. Em períodos razoavelmente pequenos, o aplicativo busca na Web uma página HTML no site www.investshop.com, que contém dados brutos sobre as cotações. Esses valores são armazenados no Banco de Dados Brutos, acompanhados da hora e data de aquisição.

1.2 O Projeto Final

Como foi dito na introdução deste documento, o projeto que será implementado é apenas uma parte de um projeto maior e mais ambicioso. Como planos futuros ficará a conclusão deste projeto, que poderá levar anos. Aqui falaremos brevemente sobre o projeto final.

A estrutura do projeto final é muito parecida com a estrutura apresentada para este projeto parcial. As principais diferenças estão na forma como os dados são obtidos e no retorno que é dado para o cliente.

No projeto parcial, os dados e informações sobre os ativos são obtidos no site InvestShop.com fazendo-se o *parse* do HTML que contém os dados relevantes. Evidentemente, para um projeto que se torne um produto comercial, isto não pode ser feito. Os dados devem ser obtidos de um

servidor de dados próprio, que pode inclusive ser um Web Service. A Bovespa, por exemplo, disponibiliza servidores como este. Fazemos o acesso e requisitamos atualizações de dados constantes, que podem ser feitas em intervalos inferiores a 5 segundos.

Surge então a pergunta: por que este tipo de aquisição de dados não é implementado na versão parcial do projeto? Simplesmente pelo fato de que uma assinatura para obtenção em tempo real de dados, como esta, é extremamente cara. Por isso, fez-se a opção pelo *parsing* do HTML de um sistema que disponibiliza estes dados gratuitamente.

A maior diferença entre o projeto implementado e o final, e que causa o grande custo de implementação, é na verdade a informação que é retornada ao usuário. De acordo com a especificação deste projeto, o sistema retorna dados para o cliente em um formato conhecido pelo aplicativo em sua máquina. Estes dados são exibidos na forma de gráficos. O usuário então, analisa estes dados e decide se compra, vende ou mantém ativos.

No projeto final as decisões de compra e venda de ativos deixa de ser do usuário. O sistema passa a analisar os gráficos e dá ao usuário os sinais adequados. Por exemplo, se o sistema detecta no gráfico que os preços dos ativos vão crescer, ele emite ao cliente um sinal de compra. Fica então a critério do usuário se o aplicativo em sua máquina deverá fazer a compra automática dos papéis, conectando-se à Bolsa, ou se deverá apenas informar o sinal.

O alto custo de implementação do projeto final reside justamente na análise dos gráficos. Isto exigiria um algoritmo muito eficiente e com poucas falhas para determinar os sinais de compra. Tal algoritmo deve ser implementado através de redes neurais, de tal forma que “aprenda” com erros e acertos. Portanto, mesmo depois de pronto, seria necessário grande tempo de execução para que este algoritmo se torne realmente eficiente.

Um sistema como esse não pode falhar com freqüência no momento de dar sinais de compra ou venda, pois isto poderia em um sinal, pois isto poderia ocasionar a perda de muito dinheiro por parte dos clientes. Portanto, sua implementação requer anos de estudos e testes, o que a torna inviável para o presente projeto.

PARTE 2

Detalhes de Implementação

Na parte anterior, explicamos o fluxo de funcionamento do projeto como um todo, sem maiores detalhes de implementação. Esse detalhamento acontecerá nesta parte. Falaremos primeiro sobre a tecnologia de Web Services, que é a base para entendimento do restante do projeto. Discutiremos a importância dessa nova tecnologia, a forma como ela é utilizada e a perspectiva futura.

Em seguida, detalharemos a implementação do aplicativo cliente. Evidentemente, como este documento se trata apenas de uma especificação dos requisitos, e não de uma descrição de desenho, o detalhamento será dado de forma geral, no âmbito das funcionalidades que o aplicativo deverá ter.

O mesmo vale para o aplicativo servidor, que será detalhado em seguida. A descrição deste aplicativo será decomposta nas suas duas funcionalidades principais: busca de dados na Web e filtragem destes dados.

A estrutura dos bancos de dados utilizados será o próximo e último tópico tratado nesta parte do documento.

2.1 Web Services

Sem exageros, hoje podemos dizer que todos os desenvolvedores deveriam pensar em fazer suas aplicações como Web Services. Vamos esclarecer a motivação para isso.

É evidente que nenhuma empresa existe isolada. Todas têm obrigatoriamente relações comerciais e este relacionamento é o motivo da existência das mesmas. Então, devemos considerar que é mais do que desejável que se criem mecanismos para que isto aconteça da melhor forma possível, com velocidade e segurança. Boa parte desta tarefa pode ser feita através de Web Services.

Vamos pensar no caso mais óbvio: gestão de estoques. Se utilizarmos Web Services para este trabalho, podemos obter desde um simples "controle de estoque" da geladeira doméstica, até de grandes varejistas, por exemplo. Pode-se controlar o estoque dos alimentos de uma geladeira, gerenciados automaticamente através do código de barras (no caso de alimentos sem código de barras, é necessário que se digite o código), e o Web Service instalado na geladeira, equipada com Windows CE, leitora de códigos de barras e conexão com Internet, solicita automaticamente ao supermercado de preferência, os alimentos que estão com o "estoque" baixo na data em que a pessoa tiver escolhido para regularização do abastecimento da casa. Nada disso faz parte de um "admirável mundo novo". A geladeira já existe na Europa e logo a teremos no Brasil. Podemos ampliar este exemplo para bares, restaurantes ou hotéis onde o sistema ao invés de simplesmente comprar, vai poder fazer cotações, e efetuar a compra no melhor fornecedor, utilizando Web Services e regras de negócio muito simples.

Outro exemplo para o uso de Web Services seria a cobrança bancaria. Hoje um título de cobrança só pode ser pago em qualquer banco até a sua data de vencimento, após isso somente no banco que deu origem ao titulo junto à empresa. Este problema existe em virtude de não termos até o momento uma forma dos bancos se comunicarem de maneira rápida e segura, com regras comuns. Se estabelecermos Web Services de cobrança entre bancos com um padrão para que as instruções de cobrança possam ser informadas com facilidade, todo o sistema se tornará muito mais eficiente.

As questões deixam de ser tecnológicas e passam a ser políticas. O problema passa a ser somente de entendimento entre as partes envolvidas. Os preços, tanto dos equipamentos, quanto dos softwares envolvidos, são acessíveis. A relação custo-benefício passa a ser totalmente favorável.

Tendo isso em mente, é fundamental que os desenvolvedores se adaptem a essa nova tecnologia. O conceito de Web Services é um pouco diferente daquele a qual estamos habituados. Ele se baseia num novo paradigma da computação denominado “rede seamless”. Trata-se de uma rede integrada de serviços que vai além das barreiras de software e hardware.

Através de padrões pré-definidos de comunicação, é possível que plataformas diferentes se comuniquem sem a necessidade do uso de protocolos específicos para cada máquina. Esta visão se baseia no conceito de operabilidade conjunta, ou seja, a capacidade de sistemas diferentes se comunicarem e compartilhar dados "seamlessly", sem estarem ligados entre si. Este é o objetivo dos Web Services. Um Web Service é uma aplicação lógica, programável,

acessível, que usa os protocolos padrão da internet, para que se torne possível a comunicação transparente de máquina-para-máquina e aplicação-para-aplicação.
Para desenvolvermos um Web Service utilizamos tecnologias pensadas sob esta visão, como o SOAP (Simple Object Access Protocol), WSDL (Web Service Description Language), e HTTP (HyperText Transfer Protocol), usadas para passar mensagens através das máquinas. Estas mensagens podem variar muito na complexidade, desde a chamada de um método, às submissões da ordem de compra.

O SOAP é o protocolo simples do acesso ao objeto. SOAP é baseado em XML e descreve um formato de mensagens para comunicação de máquina-à-máquina. Contém também diversas seções opcionais que descrevem os atendimentos do método (RPC) e que detalham a emissão de mensagens do SOAP sobre o HTTP.

Este é um pedido típico do SOAP (incluindo o cabeçalho HTTP) para um método a*nalisaString*, que faça o exame de uma string como um parâmetro:

```
POST /test/simple.asmx HTTP/1.1
Host: 131.107.72.13
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://soapinterop.org/echoString"
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns="http://soapinterop.org/"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
<soap:Body soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<tns:analisaString>
<inputString>string para analise</inputString>
</tns:analisaString>
</soap:Body>
</soap:Envelope>
```

Como podemos ver, o nome do método é codificado como o XML: <*tns:echoString* >, e o parâmetro é codificado como <*inputString* >.

O método C # que corresponde a isto é:

public String echoString(String inputString);

Esta é a resposta do servidor:

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
```

```
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns="http://soapinterop.org/"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
<soap:Body soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<tns:analisaStringResponse>
<Return>string de retorno</Return>
</tns:analisaStringResponse>
</soap:Body>
</soap:Envelope>
```

O XML *<tns:analisaStringResponse>* define que a resposta virá do método *analisaString,* e a resposta em si vem no XML *<Return>.*

Dessa forma é feita a troca de informações entre clientes e servidor. O cliente faz uma requisição de um método do servidor com os parâmetros específicos. O servidor retorna a resposta do método chamado. Tudo isso é feito utilizando a padronização XML.

2.2 O Aplicativo Cliente

O aplicativo cliente é a parte do sistema que é visível para o usuário. Sua função é obter os dados requisitados e exibi-los na forma de gráficos. O aplicativo deverá ser totalmente configurável, de forma que o usuário poderá escolher desde a freqüência de atualização dos gráficos até o ativo que será exibido.

Requisitos de Funcionalidade

Para que os objetivos do projeto sejam atingidos o aplicativo cliente deverá conter uma série de requisitos de funcionalidade. Abaixo listaremos as funcionalidades que deverão estar disponíveis para o usuário, seguidas de explicação.

- *Escolha de um ativo a ser analisado.*

O aplicativo deverá conectar ao servidor para obter a lista de ativos disponíveis para acompanhamento. Deverá então exibir uma listagem para que o usuário escolha o ativo o qual deseja acompanhar as informações.

- *Até três tipos de gráfico para o ativo escolhido poderão ser exibidos simultaneamente.*

Depois de escolhido o ativo a ser analisado, o programa deverá permitir que até três tipos de gráficos relacionados a este ativo sejam exibidos simultaneamente. A limitação máxima de três

gráficos é a opção escolhida por dois motivos. O primeiro é o mais óbvio: por questões de desempenho. Quanto maior o número de gráficos, maior seria o processamento exigido. O outro motivo é o fato de que uma limitação como essa não chega a ser uma desvantagem. Dificilmente os analistas de bolsas de valores trabalham com mais que três tipos de gráficos simultaneamente para um mesmo ativo. Se houver a necessidade de visualizar outro tipo de gráfico, basta substituir algum dos que estão em exibição.

- *Escolha do tipo de gráfico a ser exibido partir de uma listagem.*

Para definir o tipo de gráfico que deverá ser exibido, o aplicativo deverá disponibilizar ao usuário uma lista com os tipos de gráfico existentes para o ativo escolhido anteriormente. Para obter esa lista, o aplicativo deverá se conectar ao servidor.

- *Escolha da periodicidade do gráfico a ser exibido.*

Além do tipo de gráfico, é necessário definir a periodicidade dos dados exibidos. É bom lembrar que o número de valores no eixo horizontal dos gráficos é fixo. Dessa forma, periodicidades maiores perdem em precisão, mas em compensação, abrangem o período total de análise que é maior.

- *Opção de mostrar ou ocultar gráficos.*

Como já foi dito, no máximo três gráficos poderão ser exibidos simultaneamente. Entretanto, um usuário poderá escolher visualizar apenas um, dois, ou até mesmo nenhum. Dessa forma, deve haver opção de habilitação para cada um dos três gráficos.

- *Escolha da freqüência de atualização dos gráficos.*

As atualizações nos gráficos ocorrem automaticamente, em períodos de tempo. O usuário deverá ter controle sobre estes períodos. Para não sobrecarregar o servidor, é definido que o intervalo mínimo para atualizações é de 5 minutos.

- *Escolha do endereço e porta do servidor*

Para obter os dados, o aplicativo cliente deverá se conectar ao Web Service. O endereço Web para o Web Service deverá ser configurável, assim como a porta para acesso. Um botão para retomar os valores padrão também é desejável.

- *Escolha do número e intervalo de tentativas de conexão*

Eventualmente, por ser um aplicativo que se conecta via rede, pode acontecer do Web Service se tornar inacessível devido a problemas de conexão remota, como uma queda do servidor ou da própria rede. Se isso acontecer, não será mais possível atualizar os dados dos gráficos.

Dessa forma, é desejável que o usuário tenha a opção de escolher o número de tentativas de conexão remota, assim como o intervalo entre essas tentativas.

- *Status do aplicativo*

Também é desejável que o usuário possa acompanhar o status de atuação do aplicativo. Assim, ele poderá saber quando está havendo uma atualização nos dados ou quando não foi possível estabelecer a conexão remota.

- *Atualização manual*

Além da atualização periódica automática, o usuário poderá "forçar" uma atualização para cada gráfico.

Sugestão de Interface

Dentro dos requisitos de funcionalidade descritos acima, sugerimos uma interface para o aplicativo. O protótipo é composto de três tipos de interface. Uma para configurações do aplicativo como um todo, outra para configurações de cada gráfico e a última para exibição dos gráficos. Abaixo demonstramos os protótipos sugeridos para cada tipo de interface, com breve explicação.

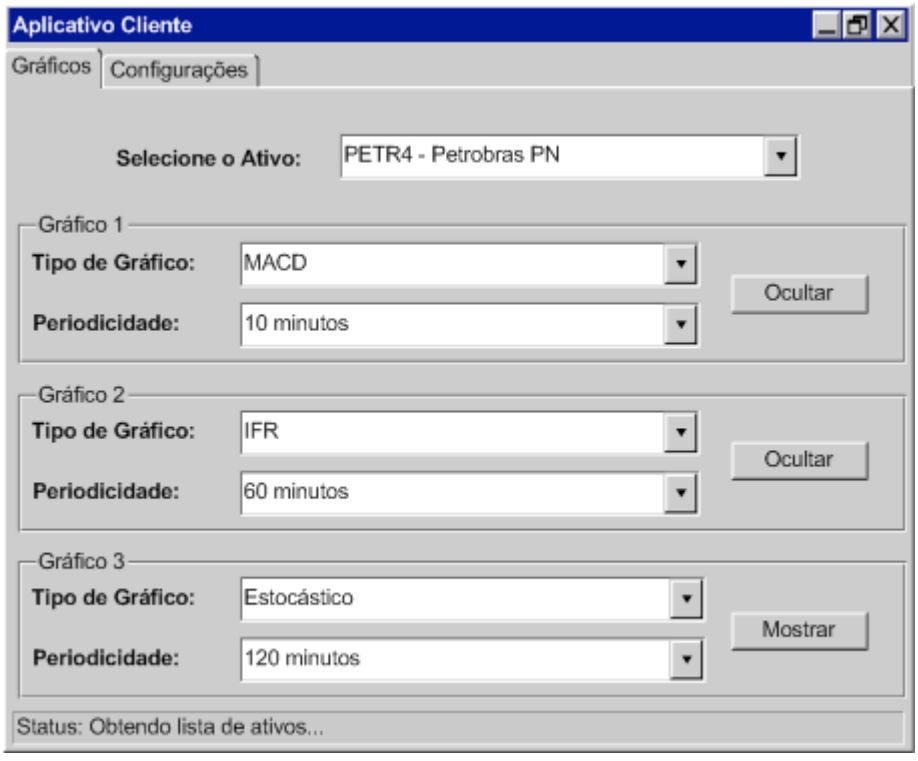

Figura 2 – Interface de Configuração de Gráficos

Na figura 2, temos uma sugestão para a interface de configuração de gráficos. O *combo box* superior define o ativo a ser analisado. Em cada caixa de agrupamento definem-se as configurações para cada um dos três gráficos a serem exibidos. As opções de cada caixa são Tipo de Gráfico e Periodicidade. Além dessas opções, o botão Ocultar/Mostrar, define se o gráfico estará visível ou não.

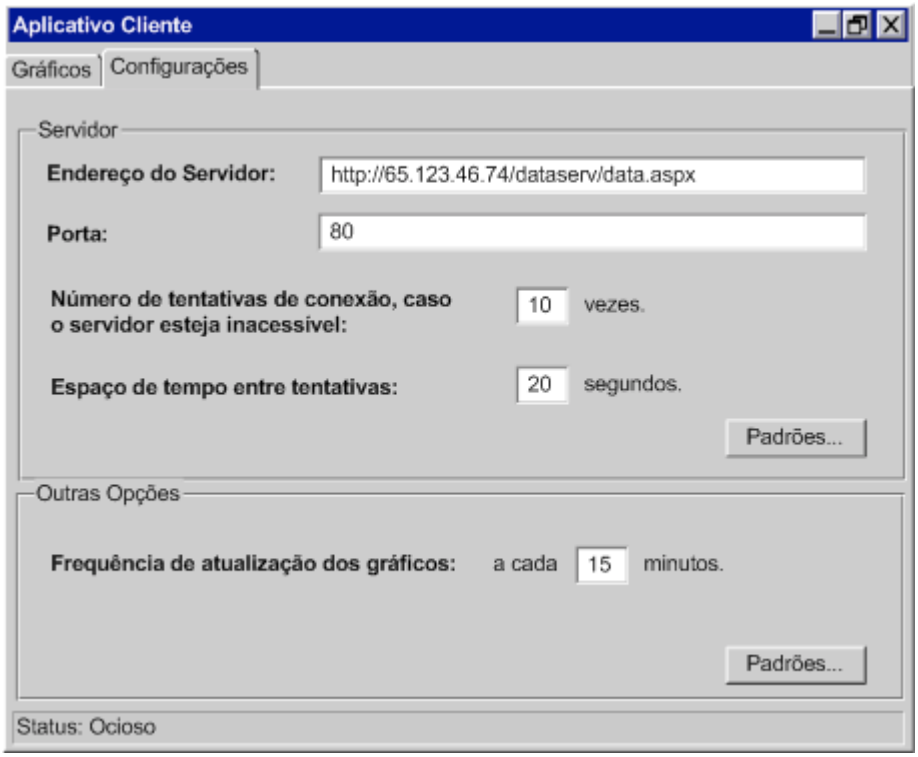

Figura 3 – Interface de Configuração Geral

O protótipo sugerido para a Interface de Configuração Geral é demonstrado na Figura 3. Na caixa de agrupamento “Servidor”, o usuário pode definir o endereço e a porta do servidor onde se localiza o Web Service, o número e o espaço de tempo entre as tentativas de conexão ao servidor, caso ocorram problemas. Na caixa de agrupamento “Outras Opções”, o usuário define a intervalo entre as atualizações de dados.

Figura 4 – Interface de Exibição de Gráficos

A Figura 4 exibe a nossa sugestão de interface para a exibição de gráficos. Sua função é simplesmente exibir na forma gráfica os dados que chegam do servidor.

2.3 O Aplicativo Servidor
2.3.1 O Módulo de Busca
2.3.2 O Módulo de Filtragem
2.4 Os Bancos de Dados
3 Dificuldades Encontradas
4 Projetos Futuros
5 Conclusões