TECNOLOGIA AJAX – INICIANDO NA TECNOLOGIA COM JAVA

AJAX Technology – Starting with the Java technology

Erik Leite MOTA
Faculdade Politécnica de Campinas

Peter JANDL JR
Faculdade de Jaguariúna
Faculdade Politécnica de Campinas

Resumo: Este artigo trata de um novo modelo alternativo que quebra o paradigma de desenvolvimento de aplicações *web* atual, o AJAX. Grandes corporações vêm adotando esta metodologia em seus sistemas e obtendo um grande sucesso. A sua utilização com a linguagem Java é simples e proporciona muito ganho de performance e produtividade.
Palavras-chave: AJAX, Java, *framework*, aplicações *web*.

Abstract: This article deals with a new alternative model that breaks the paradigm of development of web applications today, the AJAX. Large corporations are adopting this approach in their systems and getting a great success. Its use with the Java language is simple and provides much gain in performance and productivity.
Keywords: AJAX, Java, framework, web applications.

INTRODUÇÃO

Segundo Jesse James Garret, sócio fundador da *Adaptive Path*, a grande maioria dos novos sistemas de software inovadores são *on-line*. Essa tendência vem se deparando com uma dificuldade presente no modelo atual de desenvolvimento de sistemas *on-line* na *web*, pois esse modelo desfavorece qualquer tipo de aplicação *on-line* que exija um alto grau de interação com o usuário, por se basear em uma arquitetura de requisição e resposta.

Os sistemas clássicos desenvolvidos para *web* trabalham com o paradigma de requisição e resposta, ou seja, cada ação do usuário o navegador de Internet faz uma requisição HTTP (*Hyper Text Transfer Protocol*) ao servidor que faz os processamentos necessários e devolve uma resposta em um formato HTML (*Hyper Text Markup Language*) para o usuário.

Esse modelo gera o descontentamento dos usuários uma vez que, para cada solicitação feita, é necessário realizar todo o processo de envio, processamento no servidor e a devolução da página de resposta, ocorrendo na maioria dos casos o envio desnecessário de dados que não foram alterados. Isto se agrava em sistemas que exigem alta interatividade ou em páginas com um grande volume de recursos e dados, que gera um alto tráfego de rede. Nestes sistemas o cliente fica a maior parte do tempo esperando o navegador carregar a próxima página. Um exemplo clássico desse modelo é a página do FIES (Programa de Financiamento Estudantil do governo federal), aplicação repleta de caixas de seleção, onde para cada escolha efetuada pelo o usuário o navegador realiza todo o processo de carregamento da página novamente gerando uma lentidão na utilização do sistema.

É dentro deste cenário que o AJAX, acrônimo de *Asynchronous Javascript And XML* (eXtensible Markup Language), se aplica. A proposta do AJAX é justamente reduzir o tráfego de dados e aumentar a interatividade com o usuário. Apenas os dados alterados ou novos trafegam pela rede, evitando o envio de toda página (Cerqueira, 2008). AJAX é uma técnica de utilização sistemática da linguagem Javascript e da estruturação de dados XML para tornar o navegador mais interativo com o usuário, utilizando solicitações assíncronas de informações (Direct Web Remoting, 2008). Então esse termo não se refere a uma única tecnologia, mas sim a um modelo que propõem a utilização conjunta de diversas tecnologias conhecidas e aceitas na comunidade de desenvolvedores *web*.

DEFININDO AJAX

A proposta essencial do AJAX é diminuir a lentidão no processo de *request* e *response* das ações do usuário e apresentação de páginas quando feita uma requisição ao servidor. A solução para isso seria o envio de parâmetros e dados por um canal exclusivo (por meio uma *thread* própria dentro do navegador) utilizando Javascript (W3Schools, 2008). Em seguida, o servidor devolve a resposta através do mesmo canal, evitando carregar uma nova página. Dessa forma, a resposta não terá o conteúdo de uma página inteira, mas apenas os dados que necessitam modificação. Um trecho de

código Javascript se encarrega de manipular esses dados e atualizar a página, passando ao usuário a impressão de que a aplicação é mais interativa. A Figura 1 mostra a arquitetura utilizada na metodologia AJAX.

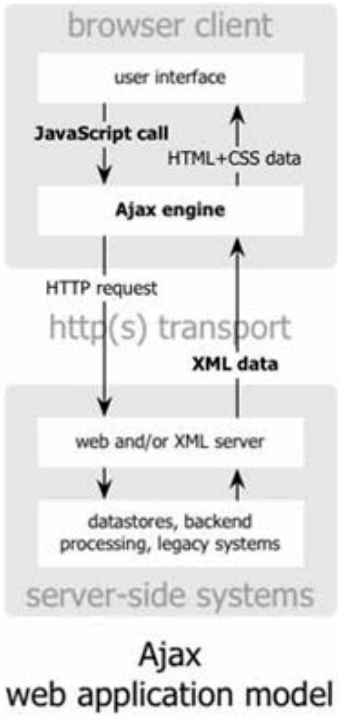

Figura 1. Ciclo de Vida de uma aplicação utilizando AJAX.

Esse modelo surgiu quando se começou usar scripts para manipulação dos elementos das páginas HTML. A idéia inicial do canal exclusivo era implementada por meio de um frame HTML de tamanho mínimo, que realizava a requisição e devolvia uma página contendo além dos dados, um *script* que atualizava o *frame* principal. O problema era que esse tipo de abordagem aumentava demais a complexidade para o gerenciamento. Neste caso era necessário uma página contendo o *frameset* (que define o *layout* dos *frames* na janela), um *frame* principal para mostrar os dados, e uma ou mais páginas para implementar a atualização do *frame* principal (cada atualização normalmente precisa de uma página de resposta específica). A Figura 2 exibe o ciclo de vida de uma aplicação *web* tradicional.

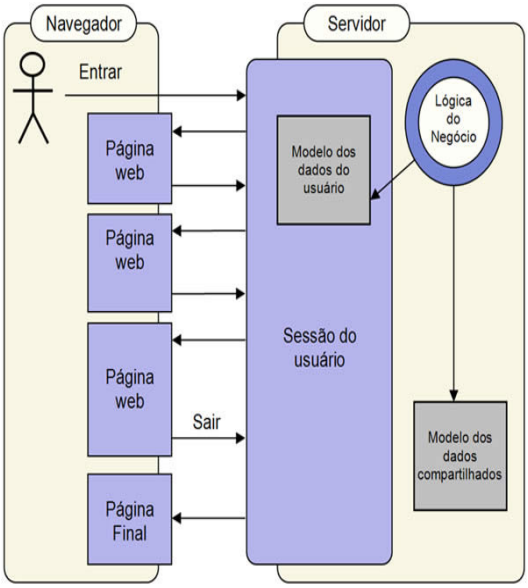

Figura 2. Ciclo de Vida de uma aplicação *web* clássica .

A situação melhorou com a evolução dos navegadores *web* através da classe XMLHttpRequest, que permite a requisição assíncrona de páginas *web*, a manipulação do conteúdo dessas páginas no formato de dados (desde que a página esteja num formato apropriado). Isso simplificou muita a implementação de requisições através de *threads* próprias dentro do navegador, sem a necessidade de utilização de artifícios como *frames* escondidos. A técnica de criar uma página atualizada dinâmica e parcialmente por meio de requisições ao servidor que utilizam o objeto XMLHttpRequest ficou conhecida como AJAX.

TIPOS DE APLICAÇÃO

Existem alguns elementos das aplicações *web* cujas características permitem explorar os benefícios obtidos com o emprego da tecnologia AJAX. Tais elementos são:

- Caixas de seleção;
- Auto-preenchimento, que completa palavras digitadas em caixas de texto, tal como no Google Sugest;
- Árvores representando dados com organização hierárquica;
- Envio de mensagens que geram um diálogo imediato entre usuários e a aplicação; e

- Filtro para campos em formulários que não necessitam o recarregamento da página.

PRINCÍPIOS DO AJAX

Para se adequar à nova idéia de programação otimizada e ágil para *web* os desenvolvedores precisam se acostumar com algumas idéias descritas a seguir.

Hospedagem no Cliente

Quando uma aplicação *web* clássica faz a requisição de algum dado ao servidor, o navegador não sabe o que está acontecendo, pois ele é um terminal "burro" que não tem idéia das ações que o usuário está realizando. Todas as informações estão gravadas no servidor em um objeto de sessão distinto para cada usuário. A manutenção de sessões de usuário no servidor muito comum atualmente. Quando o usuário faz seu primeiro acesso ao *site* é criado uma sessão no servidor na qual vários outros objetos são inseridos representando as atividades do usuário neste *site*. Nesse tempo, é enviada a página principal para o navegador com todo o seu conteúdo juntamente com dados que podem ter sido requisitados pelo usuário como, por exemplo, uma consulta a uma tabela de preços.

O usuário efetuando a saída ou fechando o navegador, implica no fechamento da aplicação e a sessão do usuário no servidor é destruída. Se o usuário necessitar de alguma informação depois, terá que acessar novamente o servidor e fazer novamente todas as requisições.

Em AJAX, parte da lógica da aplicação é deslocada para o navegador. Nesse contexto, quando o usuário entra é carregado um documento mais complexo no navegador, onde grande parte é o código Javascript. Esse documento permanecerá com o usuário até que ele feche a sessão, possibilitando, por exemplo, que o *webmail* do usuário continue logado mesmo depois de fechar o navegador.

Regra de codificação

Nas aplicações web clássicas, o Javascript (W3Schools, 2008) é utilizado para tentar aproximar as aplicações *web* das aplicações *desktop* em termos de interatividade e validação. Mas isso tem um preço, o uso excessivo dessa técnica faz com que essas aplicações fiquem lentas causando a irritação do usuário.

Escrever o código em AJAX envolve disciplina, pois a programação pesada não ficará somente no servidor, mas também no cliente e neste caso, preocupações concernentes ao lado cliente são: criação de código de alto desempenho e de manutenção fácil.

Interação dinâmica com o usuário

O navegador oferece duas formas de enviar entradas de dados para um outro computador: *hyperlinks* e formulários HTML. Os *hyperlinks* apontam para páginas dinâmicas ou *servlets* e podem estar associados a imagens ou folhas de estilo (*CSS – Cascading StyleSheets*) para melhorar a interface com o usuário.

Os formulários HTML oferecem componentes padrões para interface com o usuário como caixas de texto, caixas de checagem, botões de submissão. Em contrapartida, não contém outros componentes como seleção em árvore, grades para edição, ou caixas de combinação. Os formulários, assim como os *hyperlinks,* apontam para URLs armazenadas no servidor.

Outra alternativa possível é apontar os *hyperlinks* e formulários para funções Javascript. Atualmente essas funções têm como objetivo validar formulários verificando campos vazios, valores de intervalo entre outras até ser submetido ao servidor. A vantagem do AJAX para esse tipo de problema é que não é necessário o usuário clicar em *hyperlinks* ou fazer uma submissão de um frame para que a comunicação seja efetivada com o servidor. Essa comunicação pode ir acontecendo em paralelo com as atividades do usuário na pagina. Por exemplo, enquanto o usuário vai digitando algo em uma caixa de

texto, automaticamente um processo em paralelo carrega as opções de auto preenchimento.

Dados são fornecidos

Na aplicação clássica, quando o usuário faz uma requisição ao servidor, são retornados todos os conteúdos e dados misturados modificando toda a página. A Figura 3 mostra a comparação (dados x tempo) de uma aplicação clássica *web* com uma aplicação que utiliza AJAX.

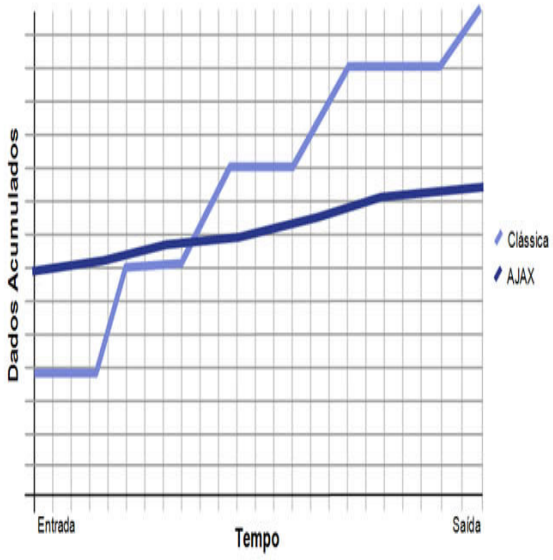

Figura 3. Tráfego de dados entre as duas arquiteturas.

FRAMEWORK DWR

O DWR (*Direct Web Remoting, 2008)* é uma biblioteca RPC (*Remote Procedure Call*) que facilita a chamada de métodos Java a partir do Javascript e chamadas de Javascript para métodos Java (Jandl Jr., 2007). Grandes corporações estão utilizando devido ao seu desenvolvimento ser prático, ágil e dinâmico.

O *framework* possui um número de *jobs* similares a chamadas em *batch*, atualizando e controlando os dados entre o Java e o Javascript. Ele realiza a criação dos métodos Java com a API do Javascript, o que faz com que se otimize ao máximo o tempo de desenvolvimento da sua aplicação. Esta forma

de trabalho não exige que os usuários necessitem de qualquer *plugin* instalado no *browser*.

A Figura 4 mostra a integração funcional do *framework* com as classes Java e os navegadores *web*.

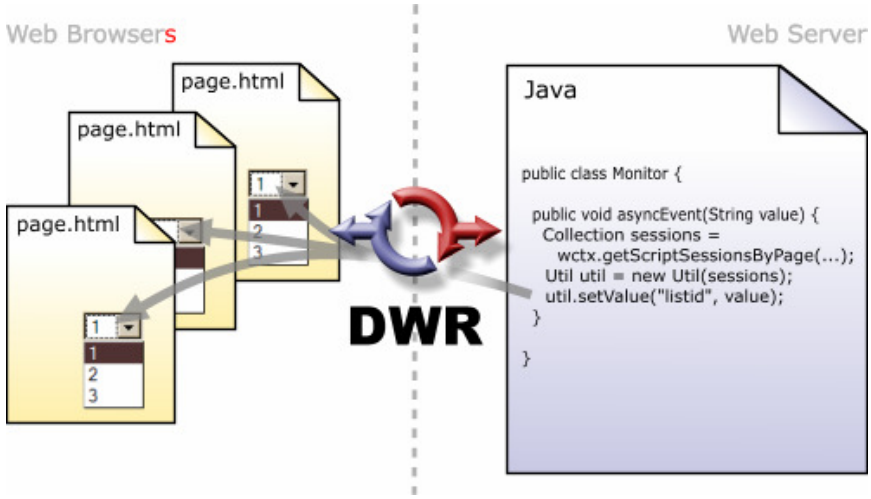

Figura 4. Exemplo funcional do *framework* DWR.

É importante ressaltar que para sua utilização deve-se conhecer toda a arquitetura envolvida no desenvolvimento de aplicações *web*.

Seguem as orientações para seu uso numa implementação simples, a qual tem por objetivo atualizar dinamicamente um dado sem ter que realizar o carregamento de dados que não foram alterados.

Passo 1: Inicialmente você deve realizar a instalação do arquivo JAR DWR e colocá-lo no WEB-INF/lib de sua aplicação;

Passo 2: Criar um arquivo chamado dwr.xml (conforme Listagem 1) no mesmo diretório de seu web.xml . Este arquivo será o responsável por dizer ao DWR quais os métodos poderão ser acessados:

```
<!DOCTYPE dwr PUBLIC
        "-//GetAhead Limited//DTD Direct Web Remoting 1.0//EN"
        "http://www.getahead.ltd.uk/dwr/dwr10.dtd">
<dwr>
      <allow>
           <create creator="new" javascript="Demo">
                 <param name="class"
                 value="com.exemplo.Demo"/>
```

```
            </create>
      </allow>
</dwr>
```

Listagem 1. Arquivo dwr.xml.

Passo 3: Criar uma página HTML e adicionar conforme a Listagem 2.

```
<p>
     Name:<input type="text" id="nome"/>
    <input value="Send" type="button" onclick="atualizar()"/>
     <br/>
     Reply: <span id="resposta"></span>
</p>
```

Listagem 2. Arquivo HTML.

Passo 4: Criar um arquivo *Javascript* conforme a Listagem 3.

```
function atualizar() {
     var name = dwr.util.getValue("nome");
     Demo.Ola(name, function(data) {
           dwr.util.setValue("resposta", data);
      });
}
```

Listagem 3. Arquivo *Javascript.*

Passo 5: Criar uma classe Java conforme a Listagem 4.

```
package com.exemplo;

public class Demo {
    public String Ola(String nome) {
        return "Hello, " + nome;
    }
}
```

Listagem 4. Classe Java.

As Figuras 5 e 6 que seguem, mostram o resultado do exemplo descrito anteriormente.

Figura 5. Página antes da execução.

Figura 6. Página após a execução.

Observa-se no momento de execução que a página não é carregada totalmente, ou seja, os dados que não foram alterados permanecem intactos, e apenas o conteúdo do campo *text* é atualizado no *label* de resposta.

CONSIDERAÇÕES FINAIS

Este artigo procurou descrever o AJAX, mostrando que se trata de uma técnica de programação que combina tecnologias conhecidas como Javascript, XML, CSS, dentre outras tecnologias que são utilizadas para se obter uma navegabilidade similar as aplicações para *desktops*, com um grande ganho de performance e transparência para o usuário. Pode-se afirmar que o AJAX não se trata de um componente ou simplesmente uma API, trata-se na realidade de várias técnicas abrangentes reunidas para formar uma arquitetura que permita agilizar a interação entre o usuário e o navegador. Com o AJAX os sistemas *web* poderão atender a certos critérios de usabilidade não atendidos nos sistemas tradicionais: navegação e tempo de resposta das ações.

Além do AJAX, podem ser empregados outros *frameworks* auxiliares, tal como o DWR, que permite utilizar chamadas de métodos Java a partir de Javascripts e vice versa, com essa facilidade se obtém um ganho na codificação e sua complexidade diminui. O protótipo apresentado, embora simples, possibilita verificar a utilização da tecnologia AJAX em aplicações *web* desenvolvidas juntamente com Java, bem como sua funcionalidade e simplicidade na codificação.

Desta forma, percebe-se que a contribuição do AJAX é bastante significativa e que seu emprego no desenvolvimento de aplicações *web* é positivo e conveniente.

REFERÊNCIAS BIBLIOGRÁFICAS

CERQUEIRA, F. **AJAX – O que você gostaria de saber**. Disponível em: http://www.linhadecodigo.com.br/artigos.asp?id_ac=952. Acessado em 23/06/2008.

DIRECT WEB REMOTING**. Easy AJAX for Java**. Disponível em: http://directwebremoting.org**.** Acessado em 22/09/2008.

JANDL JR, Peter. **Java: Guia do Programador**. São Paulo: Novatec, 2007.

OLSEN, Steven Douglas. **Ajax on Java.** New York: O'Reilly, 2007.

W3SCHOOLS. **Java Script and DOM Reference**. Disponível em: http://www.w3schools.com/jsref/default.asp. Acessado em 22/09/2008.