DESENVOLVIMENTO DE UM SOFTWARE DE ACESSO REMOTO E CONTROLE DE ÁREA DE TRABALHO DE UM COMPUTADOR ATRAVÉS DA WEB

Gabriel Sanches, Francisco Assis da Silva

Faculdade de Informática – Universidade do Oeste Paulista (UNOESTE) 19.050-920 – Presidente Prudente – SP – Brasil.
gabriel.sanches@gmail.com, chico@unoeste.br

RESUMO

Nos dias de hoje, a prática de acessar computadores remotamente usando uma conexão com a Internet tem se tornado cada vez mais constante, seja para transferência de arquivos ou para suporte técnico realizado por empresas. Devido à necessidade de acessar arquivos remotamente ou realizar suportes técnicos de forma rápida e eficaz, surgiu à idéia de desenvolver uma aplicação que necessitasse estar presente somente no computador servidor, ou seja, a máquina que disponibilizaria seus recursos. O computador cliente, aquele responsável pelo acesso, se enquadraria em qualquer computador independente de Sistema Operacional, que possuísse uma conexão com a Internet e um navegador Web com recursos para JavaScript. O navegador Web no lado do cliente é o responsável pela interface de comunicação e utilização dos recursos do servidor. A aplicação utiliza o protocolo de comunicação HTTP, tecnologias JSP e Servlets para apresentação e execução dos serviços respectivamente e JavaScript para permitir a interação do usuário com o servidor com o uso do navegador Web. A forma como foi projetada a aplicação, permite uma maior flexibilidade para o usuário, possibilitando a este acessar os recursos ou realizar suporte técnico em qualquer local onde haja as condições previstas para o funcionamento da aplicação.
Palavras-chaves: Aplicações Web; Sevlets; JSP; Acesso Remoto.

DEVELOPING A SOFTWARE FOR REMOTE ACCESS AND CONTROL OF A DESKTOP COMPUTER VIA THE WEB

ABSTRACT

Nowadays, the practice of accessing computers remotely using an Internet connection has become increasingly more frequent, or for transferring files or technical support performed by companies. Due to the need to remotely access files or perform technical support quickly and effectively, emerged the idea to develop an application that needed to be present only on the server computer, ie the machine would make available its resources. The client computer, the one responsible for access, it is any computer regardless of operating system, which possessed an Internet connection and a web browser with JavaScript resources. The Web browser on the client side is responsible for the communication interface and use of server resources. The application uses the HTTP communication protocol, JSP and Servlets technologies for presentation and delivery of services respectively, and JavaScript to allow user interaction with the server using the Web browser. The way it was designed the application, allows greater flexibility to the user, allowing it to access resources or perform technical support at any location where the requirements for running the application.
Keywords: WEB applications; Servlets;JSP; Remote Access

1. INTRODUÇÃO

Atualmente a prática de acessar computadores remotamente tem se tornado cada vez mais constante, seja para transferência de arquivos, ou para suporte técnico realizado por empresas.

Com o aumento crescente do uso da Internet tanto por usuários domésticos quanto por empresas e o contínuo acesso a dados e arquivos, foi constatada a necessidade de usar uma aplicação que tornasse possível a transferências de dados entre computadores e que também possibilitasse o controle de computadores localizados remotamente.

O acesso a computadores remotos e a transferência de arquivos, na grande maioria é feita com o uso de *sockets* [Nunes 2009]. Logo, uma aplicação construída com esse recurso exige que para obter a comunicação entre duas máquinas, ambas precisam necessariamente conter a aplicação instalada, uma cliente e outra servidora. No entanto, algumas aplicações desenvolvidas dessa forma limitam o acesso remoto somente aos computadores com mesmo sistema operacional, dependendo dos recursos utilizados para o desenvolvimento.

Pensando nessas problemáticas, a necessidade de se ter duas aplicações uma no cliente e outra no servidor, e limitação ao mesmo sistema operacional em alguns casos, foi desenvolvida uma aplicação que oferece suporte para o gerenciamento de computadores remotos utilizando a Internet. Esta aplicação permite que usuários possam acessar recursos de computadores através de um navegador Web. Dessa forma, há somente uma aplicação, e esta deve estar localizada no computador alvo (servidor). Assim, a aplicação cliente é substituída por um navegador Web e a aplicação servidora fornece todos os recursos necessários que possibilitam as transferências de arquivos e o seu controle.

2. A APLICAÇÃO

A aplicação desenvolvida é um software que deve ser instalado no computador que se deseja acessar e que responde aos comandos remotos enviados por outra máquina que controla suas ações a partir de um navegador Web. Para a construção dessa aplicação, foram utilizadas as tecnologias Servlets e JSPs, e Ajax.

Entre as aplicações exploradas que possuem uma abordagem semelhante à desta aplicação, foram encontradas: o VNC (*Virtual Network Computing*) [VNC 2009] e o LogMeIn [LogMeIn 2009].

VNC é um software de controle remoto que permite visualizar e interagir um computador cliente usando o "VNC Server" com um computador servidor usando o "VNC Viewer" em qualquer lugar na Internet. Os dois computadores não têm de ser do mesmo tipo, assim como, por exemplo, pode-se usar o VNC para visualizar um *desktop* do Windows Vista, em um computador Mac ou Linux. Por último, existe ainda um visualizador Java, de modo que qualquer computador pode ser controlado remotamente a partir de dentro de um navegador, sem ter que instalar o software. Existem várias versões para se escolher, incluindo uma versão gratuita e algumas versões comerciais substancialmente reforçadas.

LogMeIn é um produto que permite que o acesso e o controle do computador sejam feitos de forma rápida e fácil à um outro computador em uma localização remota. É necessário instalar o LogMeIn em um computador que se deseja controlar remotamente, não precisando instalá-lo na máquina responsável pelo acesso, pois ele utiliza o navegador Web como interface para o acesso remoto. Existem algumas versões, incluindo uma versão gratuita com algumas limitações (não possibilita a transferência de arquivos entre outras coisas) e algumas versões comerciais.

A idéia da aplicação desenvolvida neste trabalho consiste em permitir que qualquer computador conectado a outro em uma rede local ou na Internet possam trocar arquivos e permitir o controle por meio do mouse e teclado, seguindo a idéia de cliente e servidor. O cliente é o computador utilizado pelo usuário que fará as navegações, controle, trocas de arquivos pelo navegador Web; e o servidor a máquina remota que oferece seus recursos e permite ao cliente à visualização da sua tela e que possibilita a este requisitar arquivos para gravação ou recuperação.

O usuário para fazer o acesso abre em qualquer computador, independente de sistema operacional e de navegador Web, uma janela do *browser* e na barra de endereço fornece o IP do computador servidor, informa o *login* e a senha para realizar o acesso. A partir daí o usuário tem duas opções, abrir a uma janela que lhe permite visualizar a tela do computador remoto e provocar eventos de mouse e teclado, ou poderá abrir outra janela que lhe possibilita navegar na estrutura de arquivos e diretórios do computador servidor e assim então inserir arquivos ou recuperá-los, criar ou excluir diretórios e arquivos.

Para ser possível a manipulação do usuário, pelo navegador, o computador servidor oferece páginas JSPs construídas com JavaScript e AJAX utilizando o *framework* JQuery, que permite a interação com as páginas.

2.1. Atividades do cliente Web

Um cliente permite ao usuário fazer solicitações ao servidor, exibindo no navegador o resultado do pedido.

As solicitações que requerem retorno são de tela do servidor e estrutura de arquivos e diretórios, e as requisições que não trazem retorno são aquelas referentes aos eventos de mouse e teclado gerados pelo usuário na tela exibida em seu navegador.

A Figura 1 representa o funcionamento básico de comunicação entre cliente e servidor [Basham et. al. 2008].

Figura 1. Funcionamento do cliente Web

O lado cliente da aplicação é dividido em duas partes: controle de tela e transferência de arquivos. O controle de tela é responsável pela exibição da tela do servidor e pela detecção dos eventos do mouse e teclado no navegador do cliente. A cada 0,3 segundos a página de controle de tela requisita automaticamente para o Servlet uma imagem da tela do servidor. Esta chamada em intervalos de tempos constantes são feitas usando uma função em JavaScript “window.setInterval(função, tempo)”, onde no parâmetro “função” são colocados outros códigos JavaScript que capturam a largura e altura do *browser*. Essas medidas servem para o Servlet ser capaz de retornar ao JSP uma imagem com o tamanho equivalente a área utilizável do navegador, como ilustrado na Figura 2.

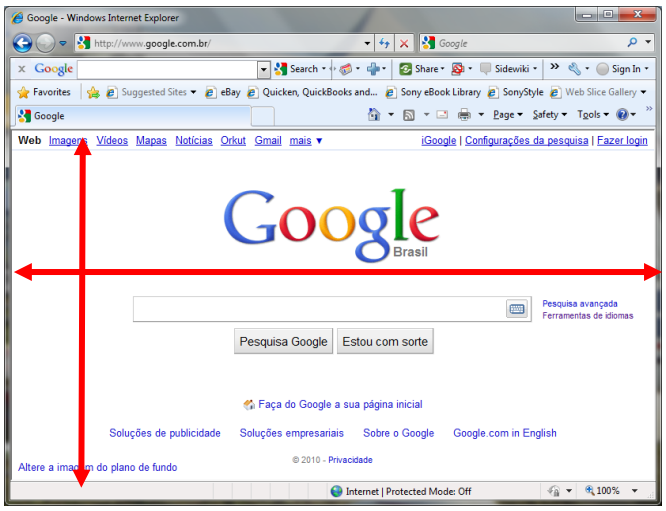

Figura 2. Área utilizável do navegador

Com a área utilizável do *browser* e com a posição *X*, *Y* do mouse, é possível calcular a posição de forma aproximada do mouse do computador remoto e colocá-lo nessa posição. Então, quando há o pressionamento de um botão do mouse é acionado uma função que obtêm as coordenadas do mouse na área utilizável do navegador, qual o botão pressionado e também as dimensões da imagem. Com esses parâmetros fica possível reproduzir o evento causado pelo usuário no navegador para o computador remoto. Para os eventos de teclado, somente é utilizada uma função para obter a tecla pressionada e enviar ao servidor para execução. Esta função retorna o *keycode* da tecla, cujo número é referente à tabela ASCII.

A transferência de arquivos é responsável pela exibição de uma estrutura montada em JavaScript que permite a navegação na estrutura de arquivos e diretórios do computador remoto. Nas Figuras 3 e 4 é mostrada a forma como é exibida essa estrutura no computador cliente.

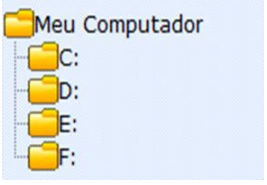

Figura 3. Diretório raiz

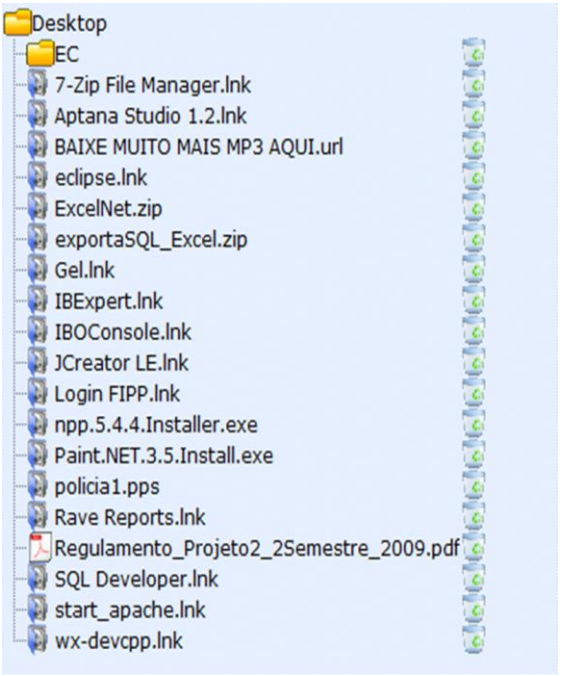

Figura 4. Arquivos dentro de um diretório

A tela da Figura 4 ilustra o conteúdo do diretório corrente “Desktop”, nela podem ser vistos os subdiretórios e arquivos. Tanto os nomes de diretórios e arquivos são links, que no caso de diretórios os abrem e em arquivos são feitos os *downloads*. A cada diretório adentrado uma requisição é feita ao servidor, e o servidor responde retornando uma lista com os diretórios e arquivos dele. Na página de transferência de arquivos também é possível criar novos diretórios e também excluir diretórios já existentes, assim como também é possível a inserção de mais arquivos no servidor e também a remoção de arquivos já existentes.

2.2. Atividades do servidor Web

Um servidor Web recebe uma solicitação e devolve uma resposta para o cliente. Na Figura 5 é ilustrada uma relação de comunicação entre cliente e servidor [Basham et. al. 2008].

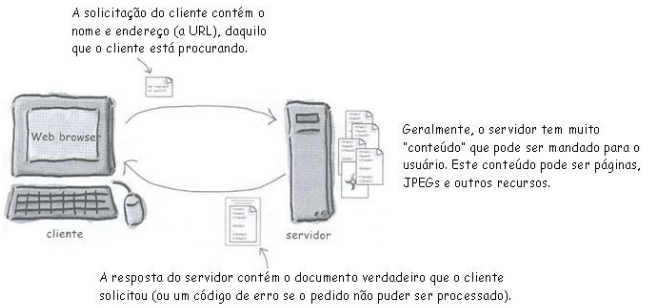

Figura 5. Comunicação entre cliente e servidor

As solicitações recebidas pelo servidor são de imagens, arquivos para transferências, lista de diretórios e arquivos, e execução de eventos de entrada (mouse e teclado). Quando o servidor recebe uma requisição de imagem ele também recebe a largura e altura da área do *browser* utilizável. A partir deste momento, fazendo uso da classe Java *Robot* é feita a captura da tela do servidor e depois é feito o redimensionamento desta tela para o tamanho requisitado. A Figura 6 faz uma comparação da dimensão da área utilizável do *browser* (lado cliente) com a do *desktop* (lado servidor).

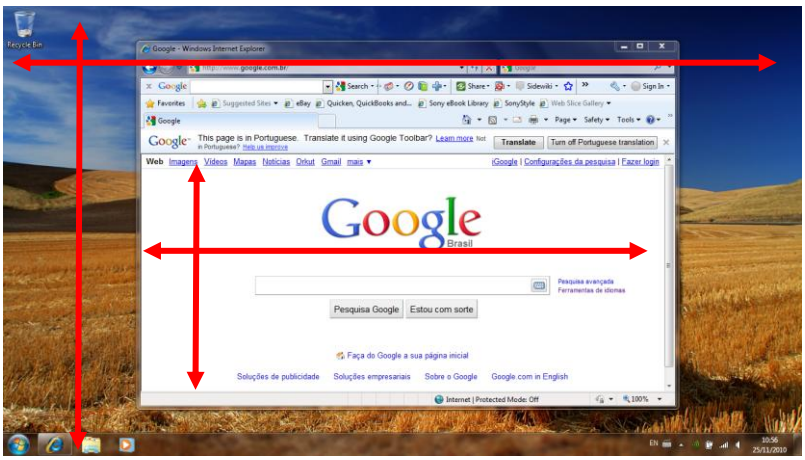

Figura 6. Comparação da área do *browser* do cliente com a tela do servidor

Nas solicitações de lista de diretórios e arquivos o Servlet recebe o caminho corrente, monta uma lista generalizada contendo na cabeça o diretório corrente e na cauda os seus subdiretórios e arquivos.

Nas solicitações de eventos de mouse, o Servlet no servidor recebe qual o botão do mouse, quantas vezes houve o clique dele, as coordenas *X* e *Y* do mouse no navegador e as dimensões da área utilizável do *browser*. De posse destes parâmetros o Servlet então encarrega a classe Java "EventosClientes" de executar o evento. Com as coordenadas do mouse e com as dimensões da imagem no navegador e da tela do servidor são calculados os novos valores das coordenadas do mouse no servidor. Sendo *X* e *Y* as coordenadas do mouse no cliente, *X*' e *Y*' as coordenadas do mouse no servidor, *L* e *H* a largura e altura da imagem no cliente e *L*' e *H*' a largura e altura da resolução da tela do servidor, tem-se que:

$$X' = (X*L')/L$$

$$Y' = (Y*H')/H$$

Com essas novas coordenadas e usando a classe Java *Robot* é realizado o posicionamento do mouse e efetuado o clique do botão desejado. Para os eventos do teclado deve-se informar ao objeto da classe *Robot* usando a função *keypressed* e ou *keyrealese*, o parâmetro *keycode* capturado no navegador do cliente.

Quando o usuário efetua um clique em um *link* de um arquivo, o Servlet no servidor entende como sendo uma requisição de *download*. O Servlet então obtém o caminho do

arquivo, carrega o arquivo no servidor e o transforma em um *array* de *bytes* e os envia de volta ao cliente.

Também há a possibilidade de fazer o *upload* de um arquivo no servidor. Foi utilizado um *listener* para armazenar o caminho atual do diretório corrente. Com o *listener* é possível guardar uma informação durante todo ciclo de vida da aplicação. Assim, enquanto o usuário percorre os diretórios, o caminho é atualizado pelo diretório corrente. Ao transferir um arquivo, este é gravado primeiramente em um diretório temporário em uma pasta do servidor e depois movido para a pasta especificada no *listener* (caminho corrente), pois não é possível gravar o arquivo diretamente no caminho escolhido.

3. TECNOLOGIAS UTILIZADAS

3.1. JSP (*Java Server Pages*) e Servlet

JSP é uma tecnologia utilizada no desenvolvimento de aplicações Web fundamentada na arquitetura SSI (*Server Side Includes*), que são comandos extensivos à linguagem HTML, os quais podem conter conteúdo estático (HTML) e dinâmico (JSP entre outros). Uma página JSP é bem parecida com uma página HTML, com a diferença de que se pode inserir Java (e itens relacionados ao Java) dentro da página [Basham, et. al. 2008].

A função do Servlet é receber uma solicitação do cliente e devolver uma resposta. A resposta leva a informação que o *browser* precisa para montar uma página como resultado à solicitação do usuário.

3.2. JavaScript

JavaScript é uma linguagem de programação criada pela Netscape em 1995, que a princípio se chamava LiveScript. No lado cliente, efetua comandos sem a necessidade de se processar no lado servidor. Sua sintaxe é semelhante à do Java, mas é totalmente diferente no que diz respeito ao seu conceito e sua finalidade. Pode-se dizer que o JavaScript é uma linguagem compatível com a linguagem Java, por esta razão, a semelhança dos nomes.

A linguagem JavaScript foi criada para atender, principalmente, as seguintes necessidades: validação de formulários no lado cliente (programa navegador) e interação com a página.

3.3. Ajax (*Asynchronous* JavaScript *and* XML)

Algumas informações descritas nesta subseção foram obtidas em Mclaughlin [Mclaughlin 2008].

O nome AJAX é uma sigla para "*Asynchronous* JavaScript *and* XML". Trata-se de uma maneira de fazer com que o navegador, com JavaScript, carregue o conteúdo do servidor sem recarregar a página atual. Pode-se, por exemplo, verificar os dados digitados em um formulário, preencher o endereço baseado em algum campo (código postal, por exemplo), exibir uma mensagem de ajuda ou mesmo recarregar uma área da página fazendo requisições em segundo plano ao servidor, sem a necessidade de recarregar a página atual.

Além de eliminar as recargas de página, o JavaScript de um aplicativo Ajax se comunica com o servidor Web assincronamente, ou seja, o código JavaScript fará uma solicitação ao servidor, mas enquanto o servidor estiver trabalhando em segundo plano, pode-se inserir dados em formulário Web e até mesmo acionar botões. Quando o servidor terminar de executar a requisição, o código poderá atualizar apenas parte da página, porém não é necessário aguardar, ressaltando-se o poder das solicitações assíncronas. Ao se combinar solicitações assíncronas com atualização de páginas sem recarga ("*refresh*" ou "*round-trip*") tem-se aplicativos Ajax.

3.4. Conteiner Web

O TomCat [TomCat 2009] é um exemplo de *Container*. Quando sua aplicação Web recebe uma solicitação para um Servlet, o servidor entrega a solicitação não ao Servlet em si, mas para o *Container* no qual o Servlet é distribuído. O *Container* entrega a solicitação, e chama os[68] métodos do Servlet (como o doPost() ou o doGet()). As Figuras 7 e 8 exemplificam o funcionamento de um *Container* em uma solicitação e resposta.

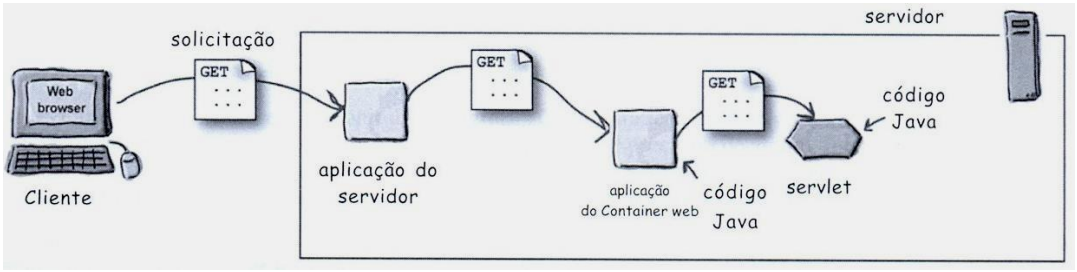

Figura 7. Funcionamento do *Container* na solicitação

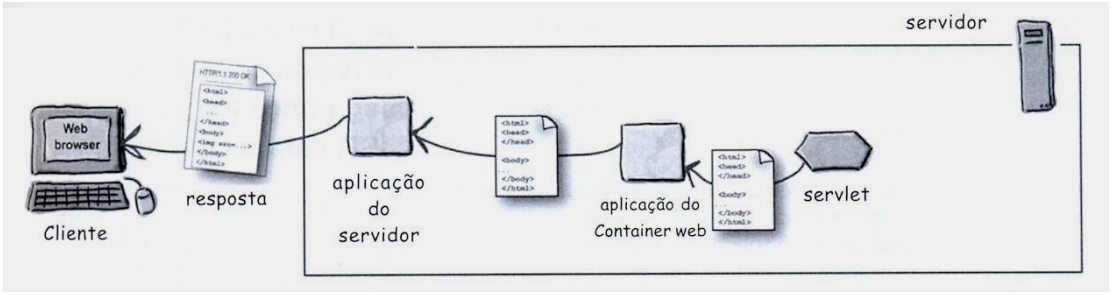

Figura 8. Funcionamento do *Container* na resposta

Com o *Container* pode-se concentrar mais na lógica do negócio, em vez de se preocupar em escrever códigos para *threads*, segurança e rede. O *Container* fornece suporte para comunicações, não é necessário construir *ServerSocket*, escutar uma porta, criar um tráfego, etc. O *Container* cria automaticamente uma nova *thread* em Java para cada solicitação do Servlet recebida. Quando o Servlet conclui a execução do método do serviço HTTP para a solicitação daquele cliente, a *thread* termina.

O *Container* roda várias *threads* para processar as várias solicitações para um único Servlet. E cada solicitação do cliente gera um novo par de objetos solicitação e resposta. A Figura 9 mostra as *threads* geradas pelo *Container*.

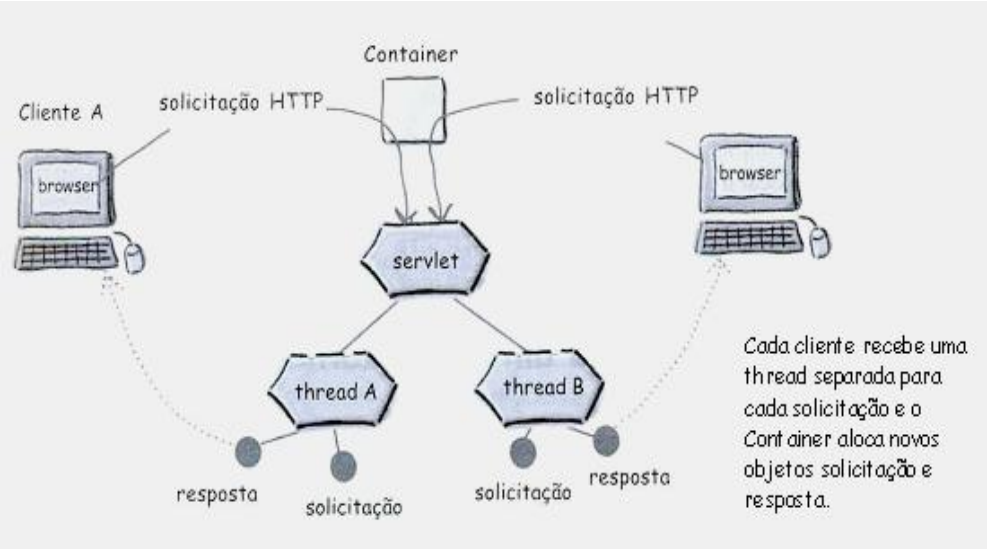

Figura 9. Demonstração de *threads* geradas pelo *Containe*

4. CONSIDERAÇÕES FINAIS

A utilização de uma aplicação que realiza o acesso remoto tem grande auxílio à usuários comuns e à empresas que prestam assistência aos seus clientes. O uso de uma aplicação deste tipo melhora consideravelmente o tempo de atendimento e diminui de forma significativa os gastos com deslocamento de funcionários e uso de equipamentos.

As aplicações desenvolvidas com o intuito de permitir acesso remoto ainda têm algumas limitações como as imagens da tela do computador remoto que em muitas aplicações são de pouca qualidade, ou as aplicações ficam limitadas ao mesmo sistema operacional. Por este motivo é de grande importância manter estudos sobre este tipo de assunto para que cada vez mais surjam idéias que melhorem o desempenho destas aplicações, procurando aumentar a qualidade das imagens e manter ou aumentar a taxa de exibição dessas imagens para tornar mais real o acesso aos computadores remotos.

REFERÊNCIAS

Basham, B., Sierra, K., Bates, B. (2008), Use a Cabeça! Servlets & JSP. AltaBooks. 2ª edição.

LogMeIn. Disponível em: http://logmein.com. Acesso em: 15 de abril de 2009.

Mclaughlin, B. Use a Cabeça!: AJAX (2008), AltaBooks, 2ª edição.

Nunes, L. R. Sockets em Java (2009). Disponível em: http://www.sumersoft.com/publicacoes/SocketsEmJAVA.pdf. Acesso em: 16 de abril de 2009.

TomCat. Disponível em: http://tomcat.apache.org. Acesso em: 21 de maio de 2009.

VNC. Disponível em: http://realvnc.com. Acesso em: 15 de abril de 2009