COBENGE 2014
ENGENHARIA: Múltiplos saberes e atuações
16 a 19 de setembro | Juiz de Fora - MG

APLICAÇÃO BASEADA NA TECNOLOGIA CLIENTE E SERVIDOR UTILIZANDO INVOCAÇÃO DE MÉTODO REMOTO

Rafael Sanzio Macedo Porto – rafael.sanzio@outlook.com
Rodrigo Rafael Rodrigues – rodrigorafael1994@gmail.com
Ruy de Oliveira – ruy@cba.ifmt.edu.br
Valtemir Emerencio do Nascimento – valtemir.emerencio@cba.ifmt.edu.br
Ed' Wilson Tavares Ferreira – edwilson.ferreira@ifmt.edu.br
IFMT/ Departamento de Informática
Rua Professora Zulmira Canavarros, 95, Centro
78005-200 – Cuiabá – Mato Grosso

Resumo: *O presente artigo resulta da experiência de alunos de graduação em atividades de tarefas de programação. O objetivo proposto é de desenvolver, em equipe, uma aplicação cliente-servidor, com recursos geralmente encontrados em softwares de comunicação. Para isso, foram desenvolvidas duas aplicações em Java. Os procedimentos metodológicos empregados envolveram o aprendizado baseado em problemas, com a implementação dos programas baseados em Invocação Remota de Métodos – RMI e comunicação Paralela – Threads. Os resultados alcançados demonstram que houve interesse e motivação dos estudantes, o que contribuiu para melhorar a fixação do conhecimento pelos discentes.*

Palavras-chave: *Sistemas distribuídos, Cliente Servidor, Chamada de Procedimento Remoto, Invocação de Método Remoto.*

1. INTRODUÇÃO

Há muito tempo que o profissional de engenharia deixou de ter um conhecimento estático, puramente relacionado à sua área principal de formação, tornando-se um profissional com conhecimento mais abrangente. Neste cenário, a computação tornou-se essencial para o bom desenvolvimento das atividades de engenharia.

Com o novo contexto, o engenheiro deve deter conhecimentos necessários, principalmente ao desenvolvimento de algumas aplicações, que poderão auxiliá-los em suas atividades cotidianas. Portanto, torna-se natural, durante o período de sua formação, o contato e o incentivo pelo professor, com linguagens de programação.

A escolha entre uma ou várias linguagens de programação envolve muitas questões e fogem do escopo da pesquisa, porém, geralmente consideram-se aspectos regionais, tempo necessário ao aprendizado, formação do corpo docente, disponibilidade de equipamentos, entre outros. No campus Cuiabá do Instituto Federal do Mato Grosso, optou-se em trabalhar este projeto com a linguagem de programação Java, principalmente pelas características que podem auxiliar o processo de aprendizagem, em especial, a possibilidade de utilização da tecnologia cliente-servidor.

O principal objetivo da aplicação baseada na tecnologia cliente-servidor é proporcionar soluções que serão uteis em ambientes coorporativos, acadêmicos e outros com a utilização da comunicação entre equipamentos, com aplicação distribuída que compartilha a carga de trabalho entre os fornecedores de um recurso ao serviço (servidor) e os requerentes (clientes). A comunicação através da rede (que pode ser local ou não) é transparente ao desenvolvedor, que pode concentrar seus esforços seus esforços na aplicação e solução do problema.

Para incentivar os discentes, optou-se em aplicar a Aprendizagem Baseada em Problemas – ABP. Esta técnica estimula a discussão entre pequenos grupos de alunos acerca de um problema real, visando encontrar uma maneira de solucioná-lo (CERA, DAL FORNO, GINDRI VIEIRA, 2013). Além disso, o estudante também precisa recordar conhecimentos previamente adquiridos em outras matérias e em períodos anteriores. A metodologia também visa ao aprendizado por meio da investigação científica. Descreve-se a seguir o trabalho desenvolvido no escopo do projeto proposto.

Educar pela pesquisa tem como condição essencial primeira que o profissional da educação seja pesquisador, ou seja, maneje a pesquisa como princípio científico e educativo e a tenha como atitude cotidiana (DEMO, 1998). Então com isto o aluno deixa de ser um objeto de estudo e se torna parte do trabalho junto ao pesquisador, fazendo com que o aluno tenha participação, junto ao pesquisador de maneira em que haja um feedback entre os dois.

Introduzindo-se a Internet, em meados de 1993 esta rede global começa a ser olhada de uma maneira diferente, pois a sua popularidade começou a aumentar. Em 2003, havia uma mudança visível no modo como as pessoas e os negócios estavam usando a Web (DEITEL, 2009). O principal motivo desta mudança se deu pelo fato que o hardware se tornava cada vez mais barato e potente. Consequentemente, os desenvolvedores passaram a dispor de máquinas mais potentes para suportar aplicações mais robustas. Portanto, torna-se evidente que as aplicações desenvolvidas tenham suporte para a comunicação em rede, especialmente com o emprego da Internet.

A Internet se manteve em ascensão, possibilitando criações de novas tecnologias, onde o usuário pudesse interagir mais com a máquina. A contínua evolução das tecnologias web propiciou a criação de novos recursos e paradigmas de desenvolvimento, levando ao usuário a junção de tecnologias web com desktop. E isso foi denominado como Aplicações ricas para Internet a *RIA* (Rich Internet Application). São aplicações Web que oferecem sensibilidade, recursos e funcionalidade ‘ricos’, que se aproximam das aplicações desktop RIAs constituem o resultado de tecnologias atuais mais avançadas, que possibilitam Interfaces Gráficas de Usuários (GUIs) avançadas e com maior sensibilidade (DEITEL, 2009). Um exemplo dessas aplicações é a plataforma JavaFX, que foi lançada em 2008 como base de sustentação para o Java.

O objetivo deste artigo é relatar a experiência dos discentes de uma turma de Engenharia da Computação no desafio de desenvolver uma aplicação cliente-servidor, bem como as vantagens e desvantagens do emprego de ABP neste cenário específico.

As demais seções deste artigo estão organizadas com segue. Na seção 2 são apresentados alguns trabalhos semelhantes, e na seção 3 descreve características importantes da plataforma JavaFX. Na seção 4 descreve aos aspectos importantes da RMI. A metodologia de ensino aplicação é tratada na seção 5. Na seção 6 apresenta-se o desenvolvimento do trabalho e as principais funcionalidades da aplicação Cliente-Servidor. Finalmente, a seção 7 expõe as considerações finais.

2. TRABALHOS RELACIONADOS

O emprego Aprendizagem Baseada em Problemas pode contribuir para construção integrada de conhecimento de forma estruturada ao rede de problemas, em um contexto de determinada área, através da integração entre conhecimentos construídos. Como consequência, têm-se o desenvolvimento de habilidades para a aprendizagem autônoma e para o trabalho em equipe. Diversas experiências bem sucedidas têm sido registradas, a exemplo de (CERA, DAL FORNO, GINDRI VIEIRA, 2013).

Como ferramenta de apoio e complemento às atividades pedagógicas, a tecnologia tem se tornado frequente. Em um trabalho prévio, os autores em (FERREIRA, et al, 2013) apresentaram um conjunto de dados, oriundos de uma rede sem fio, para auxiliar o ensino de comunicação de dados, em cursos de graduação de computação e engenharia. O conjunto de dados pode ser utilizado para apresentar detalhes importantes na comunicação, sem a necessidade de disponibilizar fisicamente toda a rede de computadores.

Percebe-se que a utilização de computadores, softwares de simulação, enfim, os recursos disponibilizados pela Tecnologia da Informação tornou-se importante ferramenta de apoio pedagógico, cujo objetivo é incentivar a busca do conhecimento pelos alunos, onde o professor assume o papel de orientador. Colaborando com este cenário, neste artigo é apresentada uma proposta do uso da tecnologia, para as disciplinas de comunicação de dados e redes de computadores, nos cursos de engenharia e computação.

3. JAVAFX

O JavaFX fornece um modelo unificado de desenvolvimento e implantação para a construção de aplicações cliente ricas que integram mídia imersiva rica, como áudio e vídeo, gráficos, texto rico e serviços Web. JavaFX permite aos desenvolvedores de criação programar em um contexto visual, o que os ajuda a trazer suas ideias para a vida real de forma mais rápida e melhor (AGOSTI & RODRIGUES, 2010).

O JavaFX trabalha com o conceito de linguagem de marcação para a interface com o usuário e essa linguagem é o FXML, possibilitando o desenvolvimento com aspecto de apresentação da interface do usuário. A marcação FXML é desenvolvida no *Scene Builder*, a qual pode ser portada para um Ambiente de Desenvolvimento Integrado (IDE) para que desenvolvedores possam adicionar a lógica de negócios na aplicação. Com isso, torna-se possível manter independente diversas atividades do desenvolvimento: codificação, banco de dados e designer.

4. MÉTODO DE INVOCAÇÃO REMOTA - RMI

A RMI é baseado em uma tecnologia anterior chamada de *Remote Procedure Call* - RPC, criada nos anos 80, que permite que programas procedurais, escritos em C, Pascal, Fortran, COBOL, DBASIC, entre outros chamem procedimentos ou funções em máquinas distintas e remotas, como se tudo estivesse presente na máquina que o requisitou.

A RPC facilita o desenvolvimento de aplicações que realizam comunicação entre máquinas remotas, de modo que o programador ou desenvolvedor possa se concentrar nas atividades de desenvolvimento. Porém, como a RPC possui um número limitado de tipos de dados simples, optou-se pela implementação utilizando a tecnologia RMI.

A tecnologia RMI é a implementação da RPC em Java para comunicação distribuída de um objeto Java. Uma vez que o método de um objeto Java é registrado como sendo remotamente acessível, um cliente pode "pesquisar" esse serviço e receber uma referência que permita ao cliente utilizar esse serviço (DEITEL & DEITEL, 1999).

5. MÉTODOLOGIA DE ENSINO APLICADA

O interesse da metodologia usada aqui está voltado a fundamentar a importância da pesquisa para a educação, até a educação, até o ponto de tornar a pesquisa a maneira escolar e acadêmica própria de educar (DEMO, 1998). Assim os discentes são incentivados a realizar pesquisa e com esse intuito cria-se uma rotina de ensino na qual os próprios alunos aprendem muito.

Não é o caso fazer dele um pesquisador "profissional", sobretudo na educação básica, já que não a cultiva em si, mas como instrumento principal do processo educativo. A relação precisa ser de sujeitos participativos, tomando-se o questionamento reconstrutivo como desafio comum (DEMO, 1998).

Sem a intenção de distribuir receitas prontas, que desde logo destruíram a qualidade propedêutica desta proposta, busca-se orientar estratégias que facilitem a capacidade de educar pela pesquisa (DEMO, 1998). Portanto o professor apresenta uma proposta de problemas na qual os indivíduos presentes discutiram sobre este problema, porém o professor os orientará, mas não dará as respostas, fazendo com que os alunos se esforcem para resolver o problema e sempre buscando mais conhecimentos.

O desafio proposto aos estudantes era o desenvolvimento de uma aplicação cliente-servidor, com comunicação através da Internet, capaz de prover o envio de arquivo de vídeo, realizar a compactação de arquivo, executar a operação de cálculo da inversa de uma matriz e também disponibilizar um bate-papo (chat) através dos programas clientes e servidor.

6. DESENVOLVIMENTO

Foi utilizada a API (Interface de Programação de Aplicativos) gráfica JavaFX, a qual suporta o FXML (que é baseado no XML, porém é utilizada para construir GUI no JavaFX); a CSS (Linguagem de Folhas de Estilo), para oferecer interfaces mais agradáveis; e tecnologias web para possibilitar possível migração para ambiente web, caso desejável ou necessário. A ferramenta Scene Builder cria o layout e o grava em arquivos a serem interpretados pelo JavaFX. Essa ferramenta possui uma técnica simples e rápida para desenvolver aplicativos. Os seus componentes possuem uma biblioteca que facilita a criação e modularização do design da interface gráfica. O JavaFX Runtime, utilizado para executar programas em JavaFX, já é parte nativa do Java Runtime Enviroment desde a versão 7 lançada em 2011.

Foram desenvolvidas duas aplicações nesse projeto, uma aplicação cliente e uma servidor. O servidor oferece serviços ao cliente, os quais serão citados posteriormente, e para

que o cliente se beneficie de um ou mais serviços do servidor é necessário uma conexão entre cliente e servidor. Nessa conexão se estabelece a porta a ser usada na comunicação, pela qual será realizada a comunicação assíncrona entre cliente-servidor no escopo da RMI.

6.1. Componentes das Aplicações

Conforme mencionado, para a confecção da interface gráfica foi utilizado o programa Scene Builder, utilizado para aprimorar e otimizar o desenvolvimento de interfaces em JavaFX baseado no conceito de drag-and-drop. Já na codificação foi utilizado a IDE (*Integrated Development Enviroment*), ambiente de desenvolvimento integrado, Eclipse 4.3 (Kepler), que é utilizado para agilizar a programação, mostrando erros, e tendo opção de depuração.

A linguagem utilizada foi Java na JDK (Java Development Kit) da versão 7.0 para desktop. O principal motivo da escolha do Java é a diversidade de plataformas (Windows, distribuições UNIX e LINUX e APPLE – OS) na qual ele funciona, pelo fato de possuir a JVM (Maquina Virtual Java). A familiaridade dos estudantes com essa linguagem também foi decisivo nessa escolha.

As características gerais, do cliente e do servidor, foram escolhidas pelo professor, com o objetivo de abranger diversos cenários de implementação. Com isso, o estudante deverá familiarizar-se com necessidades distintas, que possivelmente no futuro, poderá auxiliá-lo.

6.2. Interface Gráfica Servidor e Cliente

As funcionalidades da interface do cliente e do servidor estão ilustradas nas Figuras 1 e 2, respectivamente. Nota-se que as interfaces gráficas detêm uma aparência agradável e otimizada devido ao uso do CSS (Folhas de Estilo em Cascata).

Na interface do servidor, a aba conexão possibilita a configuração inicial do número da porta a ser usada na comunicação cliente-servidor, a pasta de armazenamento dos dados no servidor. Apenas depois dessas configurações o servidor pode ser iniciado. Obviamente é necessário que o host esteja conectado a alguma rede ou Internet, para permitir acesso remoto. Na guia de vídeo, há uma tela que possibilita a execução dos vídeos (arquivos de vídeos com extensão FLV ou MP4). O envio do arquivo de vídeo ao servidor deve ser realizado por um cliente.

Existe ainda uma thread (subprograma) que é executada no servidor (em background), a fim de atender possíveis requisições de serviço de bate-papo (chat) do cliente. Quando isso acontece, uma nova janela aparece, uma em cada lado, ou seja, no servidor e no cliente.

Figura 1 – Interface Gráfica Servidor

No cliente existem 7 guias ou abas. A aba conexão permite ao cliente escolher o servidor, com a indicação do endereço IP (Protocolo de Internet) e número da porta de conexão.

Figura 2 – Interface Gráfica Cliente

Método Conexão

A atribuição do número de porta de conexão, utilizada pelo servidor e cliente, é manual, e pode ser alterada antes da execução do servidor. Depois que esta dado é configurado, pode-se inicializar o servidor, que ficará aguardando pelas conexões dos clientes, para realizar o armazenamento de imagens, documentos e outros tipos de arquivos.

Com o servidor em execução, no software cliente é necessário apenas informar os mesmo endereço IP e número de porta de conexão. Com a realização da conexão, o servidor informa ao cliente alguns dados (data, hora) e o seus status de conectado. A figura 3 mostra o procedimento de conexão entre o cliente e o servidor.

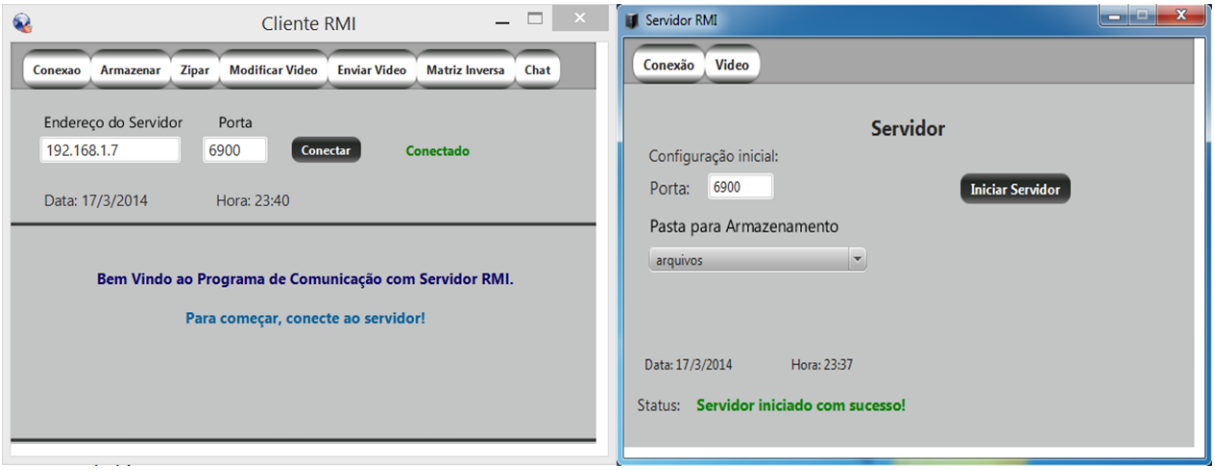

Figura 3 – Conexão entre o cliente e o servidor

Método Armazena Arquivo

Com o objetivo de realizar a transferência de arquivos (texto, imagem, vídeo, entre outros) entre o cliente e servidor, foi implementado um método no servidor para que seja invocado pelo cliente.

O método no servidor recebe os bytes do arquivo e seu respectivo nome (nome e extensão deste dado. Exemplo: "file.jpeg") e o usuário do cliente poderá escolher o arquivo que deseja enviar ao servidor. Quando o usuário do cliente o envia, o servidor o armazena em uma pasta que foi definida durante a configuração na interface gráfico do Servidor. Na Figura 4 apresenta-se como o ***Método Armazena Arquivo*** funciona no lado do cliente, pois no servidor ele apenas armazena os arquivos nos locais definidos no servidor.

Figura 4 – Interface do cliente realizando o método armazenar

Método Compactar

O método compactar realiza a compactação de dois arquivos, pelo servidor. O cliente deve enviar os arquivos ao servidor, que após realizar a compactação, devolve um único

arquivo compactado, para o cliente. É importante perceber que, além das transferências dos arquivos, também é executado a compactação pelo servidor.

Figura 5 – Interface do cliente com o método de compactação de dois arquivos

Método Executar Vídeo

Por este método o servidor recebe o vídeo enviado pelo cliente e o executa na sua própria interface gráfica. O servidor utiliza a mesma técnica do ***Método Armazena Arquivo*** para armazenar o arquivo do vídeo. Apenas vídeos no formatos MP4 e FLV podem ser executados pois o JavaFX não oferece suporte para outros tipos de mídias.

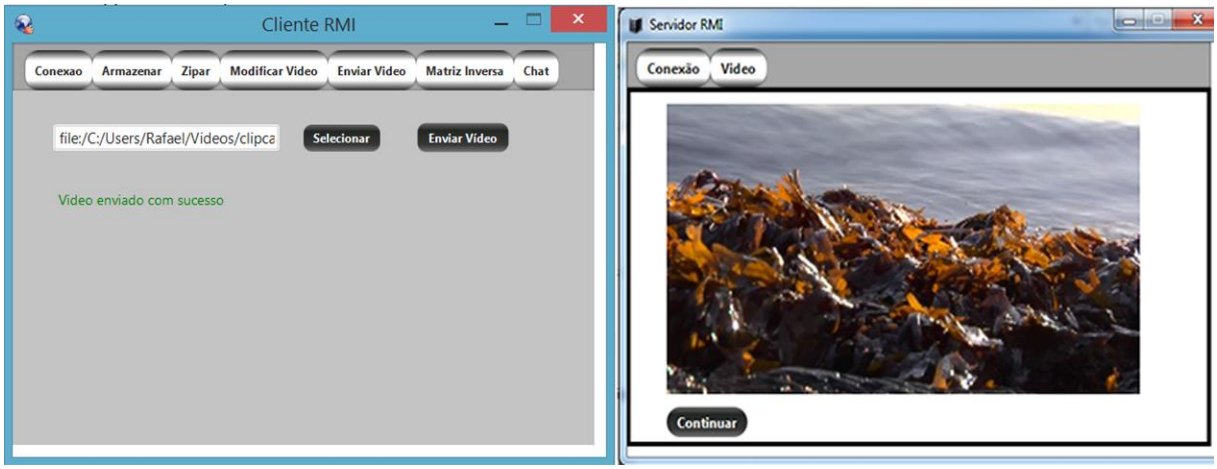

Figura 6 – Execução de vídeo enviado do cliente ao servidor

Método Matriz Inversa

Nesse método, o cliente envia ao servidor uma matriz de tamanho máximo 4x4, e o servidor devolve a matriz inversa para o cliente. Para a resolução deste problema foi utilizada a API (Interface de Programação de Aplicativos) JAMA (Java Matrix) (HICKLIN, et al, 2000)

De acordo com Joe Hicklin, Cleve Moler e Peter Webb (2014) o JAMA é um pacote de álgebra linear básica para Java, que fornece classes de nível de usuário para construir e manipular matrizes densas reais. Foi criada para fornecer funcionalidade suficiente para problemas rotineiros de maneira simples para pessoas que não sejam especialistas.

Portanto, o uso dessa ferramenta torna a execução deste método mais simples e eficiente, e com isso o usuário do cliente só precisa definir qual será a ordem da matriz (exemplo da ordem: 1x1, 2x2 até 4x4). Conforme mostrado na Figura 7, o usuário do cliente apenas precisa definir os valores dos elementos da matriz e clicar calcular. Em seguida, os dados são enviados ao servidor que realiza o cálculo da matriz inversa e a retorna ao cliente.

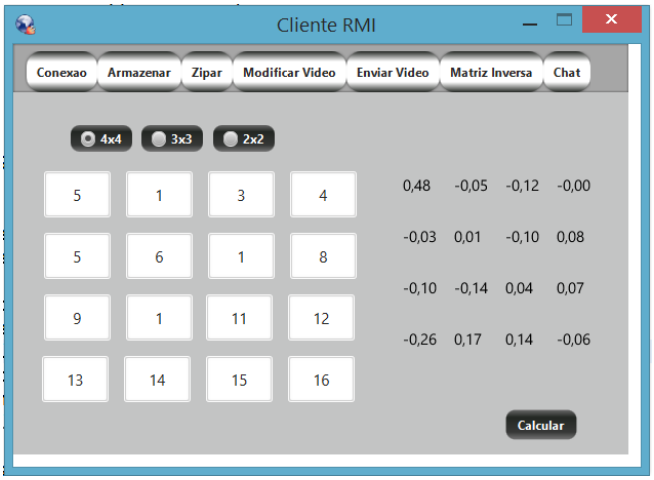

Figura 7 – Interface gráfica do cliente executando o método da matriz inversa

Método Bate-papo (Chat)

O servidor, como exibido na Figura 8, cria uma thread para cada solicitação do cliente a fim de estabelecer uma sessão de bate papo individual com cada sessão do cliente. Para realizar essa solicitação, o usuário do cliente, conforme Figura 9, ao requisitar o serviço de bate-papo, dispara a criação de uma thread específica.

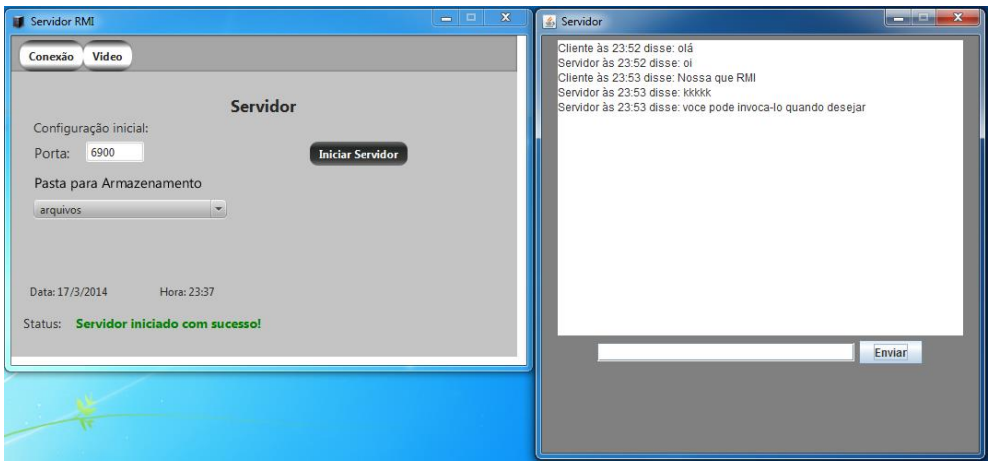

Figura 8 – Servidor conversando com o Cliente

Isto ocorre nesses dois contextos, porque as duas threads monitoram uma a outra. A thread do cliente monitora a do servidor esperando a resposta dele e o servidor faz o mesmo. Porém o servidor pode receber várias requisições, e por isso ele tem de conversar com vários clientes em maquinas distintas. De forma diferente, no lado cliente, o gerenciamento das conexões é mais simples porque o cliente está conectado a apenas um servidor, o que é controlado pelo ***Método Conexão***. Essa é uma característica da arquitetura cliente-servidor, pois de acordo com o Tanenbaum e Van Steen (2007) o servidor gerencia os serviços solicitados pelo cliente, e nesta arquitetura o cliente é dependente do servidor, porque a conexão é fixa ao dispositivo e se algo a derrubar é preciso restabelecer a conexão.

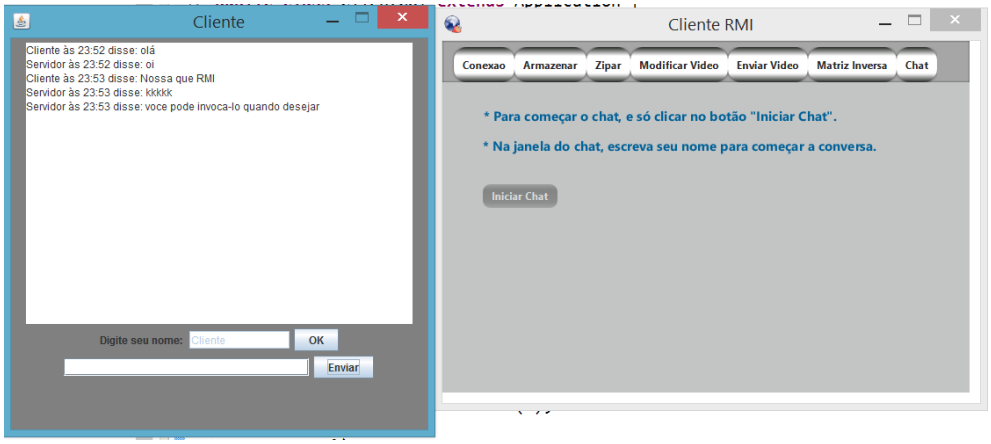

Figura 9 – Cliente conversando com o Servidor

7. CONSIDERAÇÕES FINAIS

O presente artigo apresentou o desenvolvimento de uma aplicação baseada na tecnologia cliente-servidor e o funcionamento de seus respectivos métodos que foi implementada em um ambiente acadêmico, pelos estudantes autores deste trabalho. O principal objetivo era desenvolver aplicações com aplicabilidade no mercado empresarial, educacional e outros, abordando os conceitos de sistemas distribuídos com ênfase para tecnologia cliente-servidor, desde a programação até a arquitetura. Este objetivo foi alcançado e, a experiência contribuiu para os estudantes ampliarem o conhecido a respeito de programação e trabalho em equipe.

O aspecto positivo foi o conhecimento adquirido e a motivação dos autores para criação de novos artigos ou trabalhos futuros nessa mesma área dos sistemas distribuídos. Os métodos apresentados podem ser aplicados em softwares coorporativos, a exemplo da comunicação entre usuários de setores distintos dentro de uma corporação.

Conclui-se que o desenvolvimento de sistemas ou aplicações distribuídas embasada na arquitetura cliente-servidor podem melhorar aplicações modernas e antigas, tornando-as mais dinâmicas. E a metodologia aplicada se mostrou viável por despertar o interesse dos estudantes por resolução de problemas de forma autônoma e motivadora.

REFERÊNCIAS BIBLIOGRÁFICAS

AGOSTI, Cristiano; RODRIGUES, Daniel. Construindo aplicações de interface rica com JavaFX. Unoesc & Ciência-ACET, v. 1, n. 2, p. 135-144, 2010.

CERA, Márcia Cristina; DAL FORNO, Mateus Henrique; GINDRI VIEIRA, Vanessa. Uma Proposta para o Ensino de Engenharia de Software a partir da Resolução de Problemas. Revista Brasileira de Informática na Educação, v. 20, n. 03, p. 116, 2013.

DEITEL, Harvey M.; DEITEL, Paul J. Java how to program. 3ª ed. USA: Pearson Prentice Hall, 1999, 1355 p, il.

DEITEL, Paul J. Ajax, Rich Internt Applications e desenvolvimento Web para programadores. 1ª ed. USA: Pearson Prentice Hall, 2009, 747 p, il.

DEMO; Pedro. Educar Pela Pesquisa. 8ª ed. SP, Campinas: Autores Associados, 1998, 130 p, il.

FERREIRA, Ed Wilson Tavares; SHINODA, Ailton Aquira; ARAÚJO, Nelcileno Virgilio de Souza; NASCIMENTO, Valtemir Emerencio; VILELA, Douglas. Construção e uso de base de dados sobre o funcionamento de uma rede sem fio para contribuir no ensino nos cursos de computação e engenharia. **In: XLI CONGRESSO BRASILEIRO DE ENSINO DE ENGENHARIA**, 2013.

JAVAFX OVERVIEW. Sun Microsystems. 2009b. Disponível em: <http://www.javafx.com/about/overview/>. Acesso em: 14 maio 2014.

Hicklin, J., Moler, C., Webb, P., Boisvert, R. F., Miller, B., Pozo, R., & Remington, K. (2000). Jama: A Java matrix package. Disponível em: <http://math. nist.gov/javanumerics/jama>. Acesso em: 22 maio 2014.

NIEMEYER, Patrick; KNUDSEN, Jonathan. Aprendendo Java. Rio de Janeiro: Campus, 2000, 700 p, il.

TANENBAUM, Andrew S.; VAN STEEN, Maarten. DISTRIBUTED SYSTEMS Principals and Paradigms.2.ed. USA:Pearson Pretience Hall, 2007, 686 p, il.

APPLICATION BASED ON CLIENT AND SERVER TECHNOLOGY USING REMOTE METHOD INVOKATION

COBENGE 2014
ENGENHARIA: Múltiplos saberes e atuações
16 a 19 de setembro | Juiz de Fora - MG

Abstract: *This paper presents undergrad students experiences in programming activities. The proposed goal was to develop, in teamwork, a client-server application using resources commonly found in existing communication software. Thus, the students developed two Java applications. The methodological procedures used here are based on the learning by doing paradigm, and the students applications made use of Remote Method Invocation (RMI) and parallel communication (Threads). Overall, the outcome is encouraging as the students got well dedicated and motivated, which contributed to improve their assimilation on the studied matter.*

Key-words: *Client Server Technology, Remote Method Invocation, Distributed Systems.*