II Semana de Ciência e Tecnologia do IFMG campus Bambuí
II Jornada Científica
19 a 23 de Outubro de 2009

Jogo da Forca em rede: uma aplicação distribuída utilizando *sockets* e Java

Pollyanna Carolina BARBOSA[1]; Bruno FERREIRA [2] ; Magna Caetano da SILVA[1] ; Taísa Silva de SOUSA [1]

[1]Alunas do Curso Superior Tecnólogo em Análise e Desenvolvimento de Sistemas do IFMG-Campus Bambuí.
[2]Professor Ms. do Curso Superior Tecnólogo em Análise e Desenvolvimento de Sistemas do IFMG-Campus Bambuí

RESUMO

Com o crescente aumento da tecnologia, empresas estão buscando alternativas para melhorarem seus sistemas, utilizando redes que possibilitem trocar informações com o objetivo de aperfeiçoar a comunicação dentro de seu ambiente. Uma rede de computadores é uma ligação entre dois ou mais computadores que tem como objetivo compartilhar recursos. Uma das formas para que a troca de informações ocorra, é o uso do mecanismo *socket*. Sua função principal é a implementação dos recursos de rede. A linguagem de programação Java oferece os recursos necessários para que seja possível sua implementação. Dois fatores essenciais precisam ser definidos para que esta comunicação entre socket ocorra: o protocolo de transporte e a porta de comunicação. Essa comunicação facilita a troca de mensagens usadas no sistema cliente/servidor. Onde, o cliente solicita uma conexão ao servidor e se tudo ocorrer bem, o servidor aceita, estabelecendo a conexão e gerando um socket em uma de suas portas. Portanto, tendo em vista que o desenvolvimento em rede é uma busca constante de pessoas e empresas, foi desenvolvida uma aplicação do Jogo da Forca com o objetivo de trocar informações entre os computadores em rede, utilizando o mecanismo *socket*. A aplicação foi desenvolvida na plataforma Java, sabendo que a linguagem possui uma interface fácil para a programação com s*ocket* e uma rica bibliografia.

Palavras-chave: Sistemas distribuídos.

INTRODUÇÃO

O desenvolvimento de redes de computadores e a busca incessante das pessoas e principalmente das empresas por uma comunicação mais rápida e eficaz (CANDIDO; FERREIRA, 2002), levou-se a pesquisa e desenvolvimento de primitivas *sockets* de transporte, tal estudo exige uma compreensão não só do que vem a ser um socket, mas de toda forma como é efetuada a interligação entre dois ou mais computadores.

Com o acelerado crescimento da tecnologia, as pessoas e principalmente as empresas, vêm buscando o desenvolvimento de redes de computadores com comunicação cada vez mais rápida e eficaz. De acordo com Stallings (2005), houve uma queda no custo de equipamentos de processamento de dados e aumento de sua capacidade. Por isso hoje é muito comum o uso de computadores de pequeno porte ligados em redes e não a utilização de computadores de grande porte como mainframes.

Segundo Correira (2008), o mecanismo que oferece capacidade para troca de informações em rede mais utilizado atualmente é chamado s*ocket*. Um *socket* define um mecanismo de troca de dados entre dois ou mais processos distintos, processos estes que podem estar em execução na mesma máquina ou em máquinas diferentes, porém ligadas em rede. Uma vez estabelecida à ligação entre dois processos através dos *socket*, podem ser enviados dados em ambos os sentidos até que um dos pontos termine a ligação.

Segundo Deitel (2003), a linguagem Java foi lançada em 1995 pela Sun MicroSystems, revolucionando a criação de softwares com códigos orientados a objetos independentes de plataforma. Além de ser uma linguagem orientada a objetos, Java possui facilidade de uso, oferece

portabilidade do código, segurança, mecanismo de comunicação *socket* e fácil integração a outros ambientes, destacando-se a internet.

MATERIAL E MÉTODOS

O desenvolvimento deste trabalho compreendeu em estudos da tecnologia de *socket* e da linguagem Java, uma linguagem orientada a objetos, independente de plataforma e segura.

A aplicação desenvolvida, além de utilizar o mecanismo *socket*, utiliza também os componentes GUI (*graphical user interfaces* – interfaces gráficas para os usuários), que fornece os recursos necessários para criação de interface. Deitel (2003) diz que interfaces são muito importantes na criação de qualquer software, já que é através delas que o usuário pode aprendê-lo mais rapidamente e utilizá-lo melhor.

Para Correira (2008), o mecanismo de comunicação s*ocket*, é o mais utilizado entre aplicações e permite seu uso através do Modo Orientado à Conexão, que funciona com o uso do protocolo TCP/IP, fornecendo serviços confiáveis, sem perda de dados na rede e ordem dos pacotes e ainda possibilita o uso de *DataStreams*.

Como mostra Stallings (2003)**,** cada *socket* tem um número que consiste no IP do *host* mais um numero de 16 *bits* local para este *host*, denominado porta. Para que a comunicação funcione é necessário que uma conexão seja estabelecida entre um *socket* da máquina transmissora e um *socket* da máquina receptora.

A comunicação entre os s*ockets* TCP/IP acontece da seguinte forma: o servidor escolhe uma porta e aguarda conexões a essa porta, o cliente deve informar o endereço do Servidor e a porta usada pelo Servidor. Com essa informação o cliente solicita uma conexão ao Servidor (Figura 1).

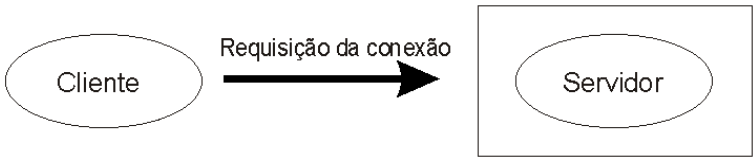

Figura 1: Conexão ao servidor

Se após o pedido de conexão não ocorrer nenhum problema, o servidor aceita a conexão gerando um socket numa porta do servidor, o que vai criar um canal de comunicação entre o cliente e o servidor (Figura 2).

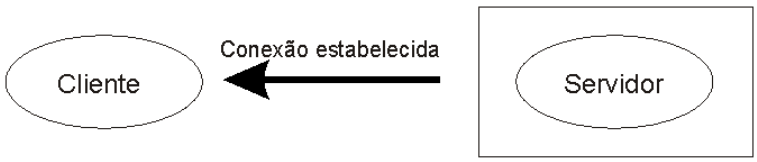

Figura 2: Conexão estabelecida

No Quadro I é mostrado o código fonte comentado da classe Cliente, que solicita a comunicação entre cliente e servidor.

```
//Socket Cliente
//1 - Abrir Conexão.
import java.io.*;
import java.net.*;
Socket client = new Socket("127.0.0.1",8080); //Conectar ao servidor localhost na porta 8080.

//2 - Obter Streams de entrada e saída para comunicação com o servidor.
DataInputStream in = new DataInputStream(client.getInputStream());//Cria um canal para receber dados.
DataOutputStream out = new DataOutputStream(client.getOutputStream());//Cria um canal para enviar dados.

//3 - Realizar a comunicação com o servidor.
out.writeInt(3000); //Envia o inteiro 3000.
out.writeUTF("Olá - Socket Cliente."); //Envia a String "Olá - Socket Cliente.".
int valor = in.readInt();//Espera pelo recebimento de um inteiro.
String texto = in.readUTF();//Espera pelo recebimento de uma String.

//4 - Fechar as Streams e a conexão.
in.close();//Fecha o canal de entrada.
out.close();//Fecha o canal de saída.
client.close();//Fecha o Socket.
```

Quadro I: Códigos *Socket Cliente*

No Quadro II é mostrado o código fonte comentado da classe Servidora, que estabelece a comunicação entre cliente e servidor. Os quadros apresentados não são aplicados ao Jogo da Forca, apenas um exemplo de comunicação de *sockets*.

```
//Socket Servidor
//1 - Criar Socket Server.
import java.io.*;
import java.net.*;
ServerSocket serveer = new ServerSocket(8080); //Cria um socket servidor na porta 8080.

//2 - Aguardar Conexões.
//O método accept retorna um socket para comunicação com o proximo cliente.
Socket sock = server.accept();

//3 - Obter Streams de entrada e saída para comunicação com o cliente.
DataInputStream in = new DataInputStream(sock.getInputStream());//Cria um canal para receber dados.
DataOutputStream out = new DataOutputStream(sock.getOutputStream());//Cria um canal para enviar dados.

//4 - Realizar a comunicação com o cliente.
int valor = in.readInt();//Espera pelo recebimento de um inteiro.
String texto = in.readUTF();//Espera pelo recebimento de uma String.
out.writeInt(6000); //Envia o inteiro 6000.
out.writeUTF("Olá - Socket Servidor."); //Envia a String "Olá - Socket Servidor.".

//5 - Fechar Streams e socket cliente.
in.close();//Fecha o canal de entrada.
out.close();//Fecha o canal de saída.
sock.close();//Fecha o Socket que está a atender o cliente.
```

Quadro II: Códigos *Socket Servidor*

RESULTADOS E DISCUSSÃO

Com base na linguagem Java e no mecanismo de comunicação *socket* foi desenvolvido o jogo da forca. Como é de conhecimento de todos, uma pessoa seleciona aleatoriamente uma palavra e descreve a quantidade de letras desta palavra em traços. As demais pessoas que participam do jogo tentam descobrir a palavra, sugerindo o preenchimento dos traços letra a letra. Cada letra sugerida é uma jogada. Cada jogador tem direito a uma jogada por vez. Caso a letra sugerida não exista na palavra, uma parte do corpo é desenhada na forca. O jogo termina quando a palavra é descoberta ou quando o desenho da forca for completado, enforcando a figura desenhada.

Do mesmo modo acontece com o Jogo desenvolvido. A primeira etapa é estabelecer a conexão, onde o usuário entra com o endereço e a porta do servidor. A Figura 3 mostra a tela correspondente à primeira etapa do jogo, onde devem ser informados os dados para estabelecimento da conexão.

Figura 3 – Tela de conexão

Com a conexão estabelecida, a segunda tela do jogo é exibida. Esta tela (Figura 4) é a principal do jogo, é onde se encontram todas as opções necessárias para sua execução. Para iniciá-lo, basta teclar as letras que correspondem a palavra sorteada (as palavras são armazenadas em um arquivo texto localizadas no servidor, que é de onde são sorteadas pelo comando *random*). À medida que os erros vão acontecendo, uma parte do boneco é desenhada. Quando o usuário erra várias vezes o boneco é desenhado e uma mensagem é mostrada na tela. Caso contrário, uma mensagem de acerto é mostrada na tela.

Outras opções como “Novo jogo”, “Ajuda” e “Dica” também estão disponíveis na tela. Apenas uma dica por palavra é mostrada. Ao clicar no botão “Ajuda”, uma nova tela é exibida, informando o funcionamento do jogo.

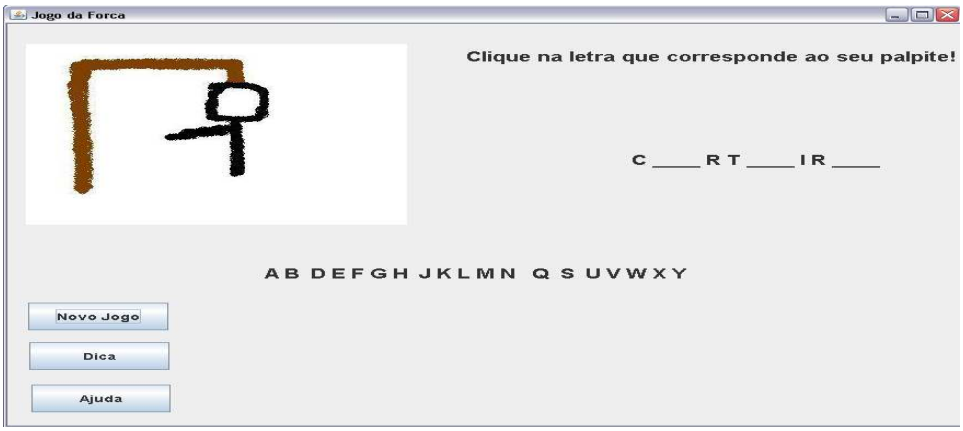

Figura 4 - Jogo da Forca em Execução.

CONSIDERAÇÕES FINAIS

A cada evolução da tecnologia os usuários buscam uma interação melhor e esperam que mais possa ser feito. A rede une o mundo. Um exemplo desta rede é a internet. Empresas e organizações a consideram como crucial para estratégias na área de informação. Java fornece capacidades de rede que tornam fácil o desenvolvimento de sistemas com comunicação. O mecanismo *socket* é uma forma simples e eficaz de estabelecer conexões e enviar informações entre elas.

A tecnologia da comunicação é fundamental para o sucesso de qualquer empresa. A facilidade de comunicação de dados e a interligação de redes eficazes e eficientes são a chave para o sucesso das empresas que necessitam cada vez mais de softwares com conexão, que possibilitem a troca de informações tanto da empresa aos seus funcionários quanto aos seus fornecedores.

O jogo desenvolvido mostra, através de uma aplicação simples, o uso do mecanismo *socket.* Mas percebe-se que por mais complexa possa ser a aplicação, a forma de comunicação se mantêm a mesma; simples. Comprovando sua importância para o desenvolvimento de aplicações que necessitem de comunicações eficazes e eficientes.

REFERÊNCIAS

CÂNDIDO, G.; FERREIRA M.; C. T. Estudo De Protocolos Em Rede - Uma Aplicação Com Socket Universidade Estadual de Mato Grosso do Sul – Dourados 2002.

CORREIRA, F. Java Socket. Disponível em: http://wmagician.wordpress.com/2008/03/05/java-sockets/ Acesso em 18/08/2009

DEITEL, H. M.; **DEITEL**, P. J. **Java:** como programar. 4. ed. Porto Alegre: Bookman, 2003. 1386 p.

STALLINGS, W. **Redes e sistemas de comunicação de dados:** teoria e aplicações corporativas. 5. ed. Rio de Janeiro: Campus, 2005. xvi, 449 p.