UNIVERSIDADE FEDERAL DO RIO DE JANEIRO – UFRJ

INSTITUTO DE MATEMÁTICA – IM
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO – DCC

SISTEMA CLIENTE-SERVIDOR
TEMA 1

Disciplina: Sistemas Operacionais
Professores: Thomé e Valéria

Júlio César Machado Bueno	106033507
Luiza Diniz e Castro	107362705
Roberta Santos Lopes	107362886

Sumário

Apresentação

O documento presente tem como principal função estabelecer a documentação requisitada no Trabalho 3 da disciplina. O tema sorteado é conhecido como Sistema Cliente-Servidor. Neste programa são utilizadas técnicas de *Threads e* comunicação por TCP/IP e *sockets.*

Especificações gerais

O Sistema foi concebido de forma a conter dois Programas distintos que possuem o seu funcionamento simultâneo em máquinas distintas. Desta forma, tem-se um Programa Servidor e um Programa Cliente. Cada um possui codificação, uso e propriedades diferentes. Dada a fácil implementação de Interface Gráfica e compatibilidade entre os diversos Sistemas Operacionais, ambos os programas são escritos utilizando a linguagem ***Java.***

Para o funcionamento correto do Sistema deve-se ter apenas uma instância do Programa Servidor sendo executado e até 5 instâncias do Programa Cliente em máquinas distintas da máquina em que o Programa Servidor está sendo executado. O limite de instâncias a serem executadas pelo Programa Cliente foi fixado para possibilitar um melhor controle das operações com os arquivos durante a apresentação simulada e também para respeitar os requisitos técnicos do trabalho. Entretanto, esse limite de instâncias não influencia em nenhum momento a lógica de operação do Sistema e, portanto, pode ser estendido facilmente.

É necessário também que as máquinas onde as instâncias do Programa Cliente estão sendo executadas, tenham conexão direta a máquina onde está sendo executado o Programa Servidor, através do protocolo TCP/IP. Ou seja, o número IP da máquina do Programa Servidor deve ser acessível externamente.

Para fins de simplificação, o Programa Servidor oferece aos Clientes somente os arquivos encontrados na raiz do diretório onde o mesmo se encontra em execução. Isto é, o Servidor oferece apenas os arquivos do nível em que ele está na sua árvore de diretórios. Desta forma, os Clientes fazem requisições ao Servidor destes arquivos. Da mesma forma, o Programa Cliente só possui acesso localmente aos arquivos que se encontram em seu diretório raiz de execução.

Regras de Negócio

Dado as especificações técnicas encontradas na descrição do Trabalho e também na lógica a ser implementada no Sistema, temos as Regras de Negócio para o Programa Servidor e Programa Cliente abaixo.

- Servidor
  - O Servidor deve ser a única instância do Programa Servidor em execução em uma máquina. Caso contrário, retornará um erro de recursos indisponíveis.
  - O Servidor sempre será inicializado com a porta 2222 (definida no código do Programa Servidor) e endereço local (*localhost*, 127.0.0.1).
  - O Servidor não possui interface gráfica.
  - O Servidor só aceita até 5 Clientes conectados a ele. Caso um novo Cliente tente a conexão, ele retornará um erro.
  - O Servidor terá como saída somente os comandos enviados pelos Clientes e o registro de novas conexões de Clientes ou Clientes desconectados.
  - Ao receber o comando CTRL+C o Servidor é encerrado.
  - Um Recurso (arquivo) só pode sofrer um comando destrutivo caso nenhum outro Cliente esteja executando um comando sobre o mesmo.

- Cliente
  - O Cliente deve ser a única instância do Programa Cliente em execução em uma máquina.
  - Ao iniciar o Cliente, somente ao campo *host* e o botão “Conectar” estarão habilitados ao Usuário.
  - O Usuário deve preencher o campo *host* com um endereço de IP para um Servidor válido. Caso contrário é retornado um alerta de falha na conexão.
  - Ao ser preenchido o campo *host* com um endereço de IP válido e clicado o botão “Conectar”, uma conexão remota deve ser criada.
  - Ao ser criada uma conexão remota, o campo *host* e o botão “Conectar” devem estar desabilitados e os outros campos estarão habilitados.
  - O Usuário só executa uma funcionalidade por vez, representada pelo seu respectivo botão.
  - O Usuário só executa uma funcionalidade em um único arquivo selecionado.
  - Ao terminar a execução do Programa Cliente sem efetuar previamente o comando representado pelo botão “Desconectar”, é enviada uma mensagem de erro ao Servidor.
  - Se um Recurso estiver bloqueado por uso pelo Servidor, qualquer comando sobre este não terá efeito.

Durante a codificação, utilizou-se estas regras como práticas de Engenharia de Software. Estas práticas incluem: a notificação de ações do Sistema para o Usuário de forma clara e a segurança da manipulação de dados (baseada nas ferramentas disponibilizadas na Linguagem Java). Desta forma, garante-se a integridade do Sistema com o mínimo de intrusão, utilizando-se a linguagem para garantir o funcionamento correto do Sistema. Isto será de grande importância para garantir que os Clientes possam executar de forma concorrente ao Servidor.

Modelagem

A Modelagem do Sistema foi concebida baseando-se nos conceitos adquiridos em sala de aula, nas Regras de Negócio apresentadas, na realização do Trabalho 2 e no material bibliográfico.

Podemos caracterizar as Entidades do Sistema da seguinte forma:

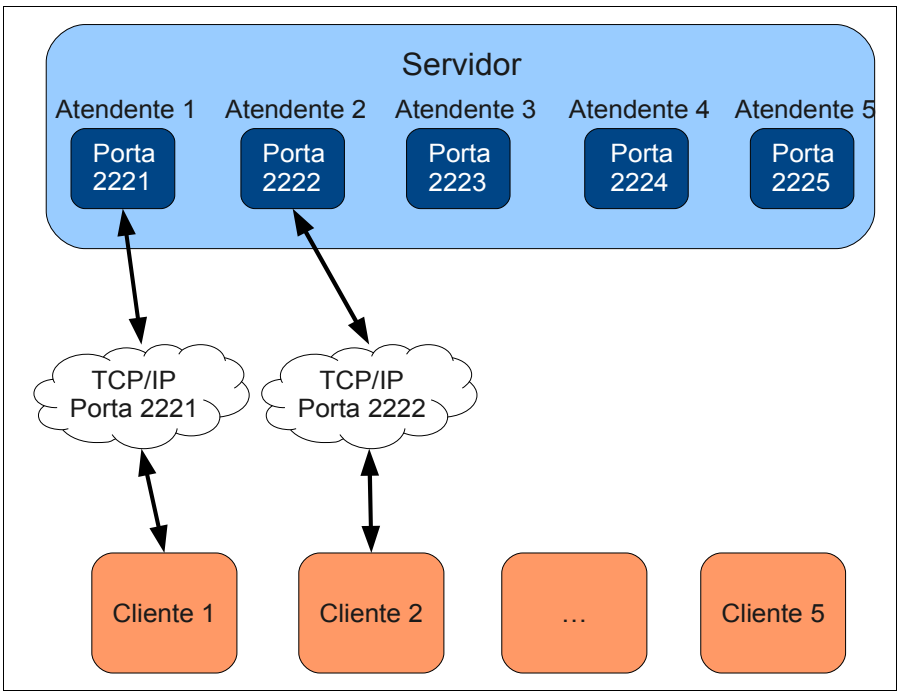
- **Servidor**
  É a Entidade responsável pela criação de *Threads* Atendentes e controle de acesso a recursos. Desta forma, a função básica do Servidor é gerenciar o número e execução das *Threads* Atendentes.

- **Atendente**
  É uma *Thread* inicializada pelo Servidor que é responsável pela criação do *socket* de comunicação através do uso da porta do Sistema. Formando então, uma conexão direta a um Cliente específico e por todo o processamento requerido pelo mesmo. É ela que, de fato, realiza as operações no servidor. Também é responsável pelo registro de ações realizadas por cada Cliente. Cada *socket* constitui um mecanismo para a transferência de pacotes de dados, com base em uma combinação de endereços IP e números de porta.

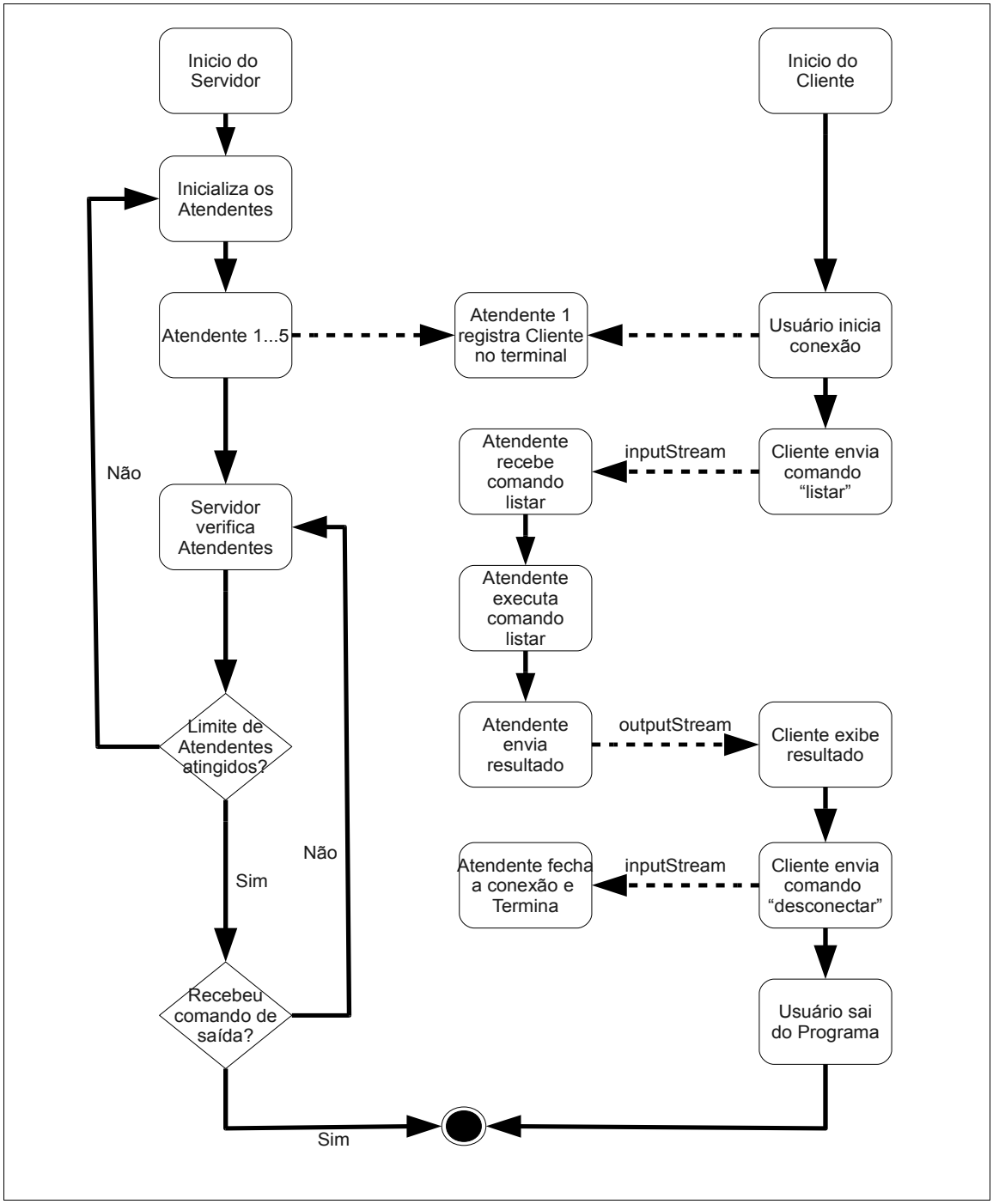
- **Cliente**
  É uma aplicação dissociada do Servidor que possui sua execução em uma máquina distinta e é responsável tanto pela conexão e envio de informações ao Servidor, quanto pelo processamento de informações locais.

- **Recurso**
  É o item a ser disponibilizado pelo Servidor para os Clientes ou pelos Clientes para o Servidor. Neste Sistema, todo Recurso é um arquivo encontrado na raiz do diretório de execução de cada Programa. No caso do Recurso do Servidor, este possui a característica de ser bloqueante durante ações destrutivas ou de transferência.

A seguir temos a **Figura 1** que ilustra o funcionamento do Sistema e mostra suas principais entidades:



Pode-se apresentar um fluxograma (**Figura 2**) que exemplifica o fluxo básico da execução do Programa Servidor e a conexão de um Programa Cliente através de um Atendente.



Lógica de Semáforos

Como os Programas Clientes atuam de forma concorrente no Servidor através dos Atendentes, é necessária a implementação de uma Lógica que permita que os Recursos sejam utilizados da melhor forma possível. Neste caso, Recursos são arquivos localizados no mesmo diretório do Servidor.

Como um exemplo de problema de concorrência, pode-se ter um Recurso sendo transferido ao Cliente conectado ao Atendente 1, enquanto o Cliente conectado ao Atendente 2 pode requerer que este mesmo Recurso (arquivo em transferência) seja removido do Servidor. Neste caso, dadas as Regras de Negócio implementadas, é necessário que a transferência seja concluída para que remoção ocorra.

Pode-se identificar um evento de acesso a uma área de Exclusão Mútua. Desta forma o Sistema conta com a implementação de uma Lógica de Semáforos incluída no Servidor a fim de garantir a integridade do Sistema. Esta lógica é implementada da seguinte forma:

Etapa	Pseudo-Código	Função

1	`static List<String> localFiles = new LinkedList<String>();`
	
	`static List<Integer> localFilesStatus = new LinkedList<Integer>();`
		O Servidor inicializa duas lista acessíveis a todos os atendentes.

2	`public static void acquire(String fileName){`
	`  Servidor.localFilesStatus.set(Servidor.localFiles.`
	`  indexOf(fileName), 0);`
	`}`
	
	`public static void release(String fileName){`
	`  Servidor.localFilesStatus.set(Servidor.localFiles.`
	`  indexOf(fileName), 1);`
	`}`
		A 1 lista contem os nomes dos arquivos no diretório local.
		
		A 2 lista contém o valor do semáforo para o determinado arquivo. Este valor pode alternar entre 0 ou 1. Em 0 o arquivo está em uso, e 1 está disponível.

3	`// Verifica o estado atual do arquivo na posição []`
	`if(Servidor.getSemaphoreStatus(fileName[1]) == 1 ){`
	`  // Entrou na região crítica`
	`  Servidor.acquire(fileName[1]);`
	
	`  executa_comando_bloqueante();`
	
	`  // Saiu da região crítica`
	`  Servidor.release(fileName[1]);`
	`}`
		O Atendente ao executar uma função em um arquivo que seja bloqueante, altera o valor da 2 lista de forma que outras *Threads* Atendentes não possam executar os comandos.

Interface e Operação

Programa Servidor

A interface de cada Programa é diferenciada. Para o Programa Servidor, foi especificado que este operasse somente em modo texto. Para o Programa Cliente existe uma interface gráfica GUI multi-plataforma. Desta forma, a operação do Servidor se dá somente através do console. Cada ação do Servidor representativa para o Sistema é expressa como saída do Terminal.

A seguir temos o exemplo de execução apresentado no Fluxograma da Modelagem do Módulo “Modelagem”.

```
Servidor criado em localhost
Foram criadas 5 Threads Atendentes para 5 Clientes
Um novo Cliente se conectou a Thread Atendente 1 pela porta 2221
Foi executado o comando <ls -p>
Cliente conectado ao Atendente 1 desconectou-se do Servidor
```

Neste exemplo podemos observar que:

- O Servidor indica a sua criação. (Linha 1)
- O número de *Threads* Atendentes em execução. (Linha 2)
- A conexão de um Cliente a *Thread* Atendente na porta determinada. (Linha 3)
- A requisição e execução de uma funcionalidade. (Linha 4)
- A desconexão de um Cliente. (Linha 5)

Para terminar a execução do Programa Servidor é necessário executar o comando CTRL+C através do console.

Programa Cliente

O Programa Cliente segue o modelo de divisão da tela onde no painel superior são exibidos as informações de conexão de desconexão ao Servidor. Na área a esquerda estão os comandos e a exibição da listagem de arquivos do Servidor. A direta está a listagem de arquivos locais e os comandos disponíveis.

A seguir temos telas que demonstram a operação do Programa Cliente.
Tela de inicio.

Tela após a conexão.

Tela após a execução do comando de Listagem de arquivos do Servidor.

Tela após a execução do comando de Listagem de arquivos Locais.

Tela após o Download de um arquivo do Servidor.

Tela após o Upload de um arquivo para o Servidor.

Dificuldades Encontradas

Durante o desenlvolvimento do Sistema encontramos as seguintes dificuldades:

- Implementação da comunicação entre os programas através da rede TCP/IP. Dado que esta são podem gerar diversas exceções que devem ser tratadas.
- Problemas relacionados a execução remota de comandos. Alguns comandos são

complicados para serem enviados e tratados pelo Atendente.
- Leitura e transferência de arquivos por *Socket* requer uma sequencia de métodos que podem retornar erros e devem ser tratados.
- Necessidade de múltiplas máquinas para testes eficazes e validação do Sistema.

Resultados Alcançados

Com o uso do Programa Servidor e Programa Cliente obtemos então um Sistema que atua de acordo com a maioria das especificações requeridas satisfatoriamente. As funcionalidads que não constam no Sistema foram excluidas da solução, devido a sua complexidade de implementação e possíveis problemas de confiabilidade dado o prazo de entrega do Trabalho. Por exemplo, a transferência de múltiplos arquivos por extenção enquanto há outras requisições pendentes de outros Clientes. Em caso de falha na transferência, o gerenciamento desta falha se torna bastante difícil.

No entanto, a solução apresenta uma boa qualidade aliada a confiabilidade e facilidade de uso. As funcionalidades do Sistema são adequadas e robustas para o uso destinado.

Bibliografia

Referências bibliográficas utilizadas no estudo:

- Operating Systems: Internals and Design Principles (5th Edition)
  ISBN-13: 978-0131479548

Apêndice

Método de execução

Para compilar e executar os programas em um ambiente UNIX é necessário executar os seguintes comandos:

No caso do Programa Servidor:
1- Rodar o arquivo Servidor.jar
java -jar ./Servidor.jar

2- O log da execução será mostrado no próprio terminal.

No caso do Programa Cliente:
1- Rodar o arquivo Cliente.jar
java -jar ./Cliente.jar

2- Será exibida uma interface.
Obs.: O Programa Cliente pode ser compilado e executado em qualquer ambiente que possua uma máquina virtual Java 1.6 ou superior.

Obs.: Para verificação do IP do Servidor pode-se executar o seguinte comando */sbin/ifconfig.*
Após isso, procurar pelo valor do campo *inet end*.