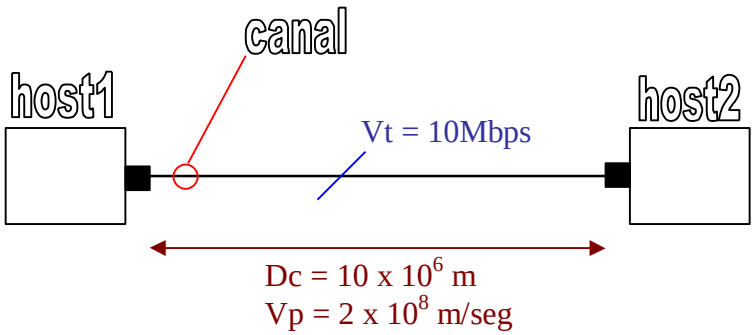
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA
Departamento de Informática

Introdução aos Sistemas e Redes de Computadores 2010/11

11ª Folha de Exercícios

Assunto: Ciclo de desenvolvimento de um programa simples em *linguagem Java*; variáveis do ambiente; desenvolvimento de programas que exercitam a API do Java relacionada com ficheiros e directorias.

A. Canais de comunicação e tempos de transferência de informação



1. Um ficheiro MP3 de 30 M bits de tamanho vai ser transmitido do computador origem para o computador destino. Todos os canais entre os dois computadores têm uma velocidade de transmissão de 10 M bps. Assuma que a velocidade de propagação é de 2 * 10^8 metros por segundo e que a distância entre os dois computadores é de 10.000 Km. A representação esquemática dos dados do problema é dada na figura acima.

1.1. Suponha que só existe um canal entre os dois computadores. Suponha também que todo o ficheiro é transmitido num único pacote. O tempo de transmissão é de:

a) 50 mili segundos
b) 3,05 segundos
c) 3 segundos
d) nenhum desses valores

1.2. Suponha que só existe um canal entre os dois computadores. Suponha também que todo o ficheiro é transmitido num único pacote. O tempo de transmissão de extremo a extremo é de:

a) 3 segundos
b) 6 segundos
c) 3,05 segundos
d) nenhum desses valores

2. No contexto do exercício 1, suponha agora que existem dois canais de 5.000 Km com um router no meio entre os dois computadores, tal como previamente ilustrado na Figura 2.

   Suponha também que todo o ficheiro é transmitido num único pacote. A transmissão do pacote pelo router só começa depois de este o ter recebido integralmente (*store & forward*). Suponha também que não há nenhuma saturação do router.

   O tempo de transmissão de extremo a extremo é agora de:

   a) 3,05 segundos
   b) 6,1 segundos
   c) 6,05 segundos
   d) nenhum desses valores

B. Utilização de *sockets* para programar aplicações que envolvem mais do que um computador

Nesta parte da aula proceder-se-á à programação de algumas aplicações em Java que utilizam a arquitectura cliente-servidor. A comunicação entre o cliente e o servidor utiliza *sockets*.

Pretende-se que faça o desenvolvimento do código Java do cliente e do servidor usando o ambiente especificada na ficha da aula prática anterior: interpretador de linha de comandos do Windows, compilador *javac*, editor de texto *Notepad++*, etc. Recorde que

- Tem de actualizar a variável de ambiente PATH para incluir a directoria onde está o compilador de Java para *bytecode*;

```
PATH=%PATH%;C:\Program Files\Java\jdk1.6.0_07\bin
```

- Quando invoca o interpretador de bytescodes Java tem de garantir que a variável de ambiente CLASSPATH inclui a directoria corrente

```
set CLASSPATH=.
```

Para obter informação sobre os métodos e as classes relacionadas com *sockets* pode consultar os seguintes documentos "on line"

http://download.oracle.com/javase/tutorial/networking/sockets/
http://www.ibm.com/developerworks/java/tutorials/j-sockets/

Veja também no CLIP os slides apresentados na aula teórico-prática desta semana.

1. Escreva o código de um servidor com funcionamento semelhante ao servidor de eco descrito na aula teórico-prática. Este servidor deve receber uma linha de texto terminada por CR e ecoar essa linha. O servidor deve oferecer os seus serviços na porta TCP 5000.

Uma sugestão para auxiliar o seu trabalho é envolver o *inputStream* associado ao *socket* com um objecto do tipo *Scanner*

```
Scanner input = new Scanner( sock.getInputStream( ) );
```

Assim é possível usar coisas como

```
String s = input.nextLine();
```

Pode fazer a mesma coisa para o *outputStream*:

```
PrintWriter output = new PrintWriter( sock.getOutputStream( ), true );
...
output.println( “HTTP/1.1”);
```

Para testar o seu programa, lance uma janela com o interpretador de comandos e faça

```
java  servidor-eco
```

2. Escreva agora um cliente para o servidor desenvolvido no ponto 1. Teste o seu servidor lançando o cliente numa outra consola

```
java  cliente-eco  localhost
```

3. Desenvolva o código de um servidor de ficheiros. Esse servidor está atento à porta 5100 e espera receber o nome de um ficheiro. Se o ficheiro existir, o servidor envia o seu conteúdo pela conexão aberta pelo cliente.

4. Desenvolva um cliente para o seu servidor de ficheiros. O cliente deve receber o nome do ficheiro do terminal e enviá-lo ao servidor. Se o ficheiro existir o servidor envia o seu conteúdo; à medida que recebe os bytes do ficheiro o cliente escreve o seu conteúdo no terminal,