



Licenciatura em Engenharia Informática
1º TESTE– Redes de Computadores – Versão 2
1º Semestre, 2008/2009 (1/Abril/2009)

NOTAS: Leia com atenção cada questão antes de responder. A interpretação do enunciado de cada pergunta é um factor de avaliação do teste. As respostas erradas às alíneas da Questão 1 **descontam** 25% da respectiva cotação.

A duração do teste é 1 hora e 30 minutos sem tolerância.

O enunciado contém **6 páginas** que devem ser entregues com a resposta ao teste.

NOME: _____ N° Aluno: _____

Questão 1) Considere um canal bidireccional com a velocidade de transmissão de 1 Mbps e 2000 Km de comprimento. Assuma que a velocidade de propagação é de $2 * 10^5$ Km por segundo. Sobre este canal estão-se a usar pacotes com 10^4 bits. Assinale com um círculo a sua resposta às seguintes questões:

a) Quanto tempo leva um pacote a ser transferido através deste canal, isto é, quanto tempo tem de passar desde o início da transmissão até o último bit do pacote chegar à outra extremidade do canal ?

1, 2, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 60, 75, 80, 90, 100, 110, 120 mili segundos ou nenhum destes valores ?

b) A que distância do receptor se encontra no canal o 1º bit do pacote, quando o último bit é transmitido ?

0, 100, 200, 500, 750, 1000, 1250, 1500, 1750, 2000, 2250, 2500, 2750, 3000, 3500, 3750, 4000, 4200, 4500, 5000 Km, nenhum destes valores

c) Admita que o emissor só pode passar a transmitir o pacote seguinte quando recebe a confirmação vinda do receptor da recepção do último pacote transmitido. Essa confirmação vem através de um pacote especial, com tempo de transmissão desprezável, transmitida logo que o receptor recebe o último bit de cada pacote. Quantos pacotes por segundo conseguem ser transferidos em média pelo canal ?

1, 2, 5, 7, 10, 12, 15, 18, 19, 20, 22, 23, 25, 27, 30, 32, 33, 40, 45, 51, 52, 55, 60 pacotes / s, nenhum destes valores

d) Que fracção aproximada da velocidade de transmissão do canal está a ser usada na alínea c) ?

1, 2, 5, 7, 10, 12, 15, 18, 19, 20, 22, 25, 30, 32, 33, 35, 40, 45, 51, 52, 55, 60, 70, 80, 90, 100%, nenhum destes valores

e) Considere que um ficheiro F cabe em 100 pacotes transferidos todos usando o protocolo da alínea c). Quanto tempo leva o ficheiro a ser transmitido com aproximação aos segundos ?

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21 segundos, nenhum destes valores

f) Numa extremidade do canal está um browser WEB e na outra um servidor. O browser sabe o endereço IP do servidor e leva um RTT a abrir um canal TCP para o servidor. Quanto tempo leva o browser a mostrar uma página ao utilizador admitindo que a mensagem com o pedido HTTP tem um tempo de transmissão desprezável, que o servidor responde assim que a recebe, e que a mensagem de resposta ocupa exactamente 1 pacote:

1, 2, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 60, 65, 70, 75, 80, 85, 90, 95, 100, 110, 120 mili segundos, nenhum destes valores ?

Questão 2) Imagine uma unidade de produção industrial onde, por questões de segurança, existe a necessidade de monitorizar a temperatura dos equipamentos em funcionamento. Para tal, existem espalhados pela fábrica pequenos computadores, ligados em rede e munidos de sensores de temperatura, que designaremos genericamente por *sensores*. Cada sensor pode trocar mensagens com outros computadores. No sistema existem computadores que funcionam como monitores (operados automaticamente ou por pessoas) e cujo papel é colectar periodicamente as medidas de temperatura de um ou mais sensores. Um sensor pode receber pedidos de monitorização provenientes de vários computadores diferentes, ao mesmo tempo, e por isso deve estar preparado para servir simultaneamente a vários monitores. Para facilitar a configuração do sistema, cada sensor quando arranca regista-se num computador central indicando o seu identificador, endereço IP e porto. Um computador C, que deseje monitorizar um sensor S, pode consultar a lista de sensores activos registados na central, e descobrir, assim, o respectivo endereço IP e porta. Posteriormente, pode manifestar directamente a S o desejo de receber periodicamente o resultado das leituras de temperatura feitas por S. Isto é, o sensor S passará a enviar periodicamente as leituras da temperatura para C. Assim, S irá tomar a iniciativa de enviar periodicamente para cada cliente monitor que se registou nele os valores que vai lendo.

O código em anexo representa uma hipotética implementação do sensor (listagem **TemperatureSensor.java**) e da central de registo de sensores (listagem **SensorCentral.java**). Os monitores serão clientes da central para descoberta do IP e porto de cada sensor e são clientes dos sensores para passar a receber os valores de temperatura (listagem **Client.java**).

Neste cenário, a comunicação é toda suportada por troca de linhas de texto (strings), de acordo com o seguinte protocolo aplicacional:

- Registo do sensor na central: o sensor, usando uma ligação TCP, envia à central uma string na forma “REGISTER *sensor_id sensor_ip sensor_port*”;
- Consulta da central pelo cliente monitor: o monitor usa uma ligação TCP à central para enviar um pedido na forma “SENSOR *sensor_id*”. A central responde ao monitor, pela ligação, com uma string contendo “SENSOR *sensor_id sensor_ip sensor_port*” se este estiver registado ou “ERROR: unknown sensor...” se ‘*sensor_id*’ não estiver registado;
- Registo do monitor no sensor: o cliente monitor envia uma mensagem UDP ao sensor contendo uma string “MONITOR”. A identificação da origem desta mensagem é usada pelo sensor para conhecer os monitores;
- Envio das leituras das temperaturas do sensor para os seus clientes: cada valor de temperatura é enviado a todos os clientes conhecidos usando uma mensagem UDP contendo uma string na forma de “TEMPERATURE *sensor_id currTemperature*”.

- a) Complete os espaços deixados em branco na listagem **SensorCentral.java**.
- b) Complete os espaços deixados em branco na listagem **TemperatureSensor.java**.
- c) Implemente o monitor na listagem **Client.java**.

SensorCentral.java

```
import java.net.*;
import java.util.*;

public class SensorCentral {

    private int port ;
    private Map<String, InetSocketAddress> sensors = new HashMap<String, InetSocketAddress>() ;

    SensorCentral( int port ) {
        this.port = port ;
    }

    public String getSensor( String id ) {
        InetSocketAddress val = sensors.get(id) ;
        if( val != null ) {
            return String.format("SENSOR %s %s %d ", id,
                                val.getAddress().getHostAddress(), val.getPort() ) ;
        } else
            return "ERROR: unknown sensor..." ;
    }

    public void registerSensor( String sensorId, InetSocketAddress sensorAddr) {
        sensors.put( sensorId, sensorAddr) ;
    }

    private void init() {
        try {
            [ ] sock = new [ ] ;
            for(;;) {
                [ ] cs = sock.[ ] ;
                Scanner scanner = new Scanner( cs.[ ] ) ;

                String cmd = scanner.next().trim() ;
                if( cmd.equals("SENSOR") ) {
                    String id = scanner.next() ;
                    String reply = getSensor( id ) ;

                    // send reply to client:
                    [ ]

                } else if( cmd.equals("REGISTER") ) {
                    String sensor_id = scanner.next();
                    String sensor_ip = scanner.next() ;
                    int sensor_port = scanner.nextInt() ;
                    this.registerSensor(sensor_id,
                                        new InetSocketAddress( sensor_ip,sensor_port));
                }
                cs.[ ] ();
            }
        } catch( Exception x ) {
            x.printStackTrace() ;
        }
    }

    public static void main( String[] args ) {
        if( args.length != 1 ) {
            System.err.println("use: java SensorCentral sensor_central_port") ;
            System.exit(0) ;
        }
        new SensorCentral( Integer.parseInt( args[0] ) ).init() ;
    }
}
```

TemperatureSensor.java

```
import java.net.*;
import java.util.*;

public class TemperatureSensor implements Runnable {

    private int sensor_port;
    private String sensor_id;
    private InetAddress sensorCentral;
    private Set<InetAddress> clients = new HashSet<InetAddress>();

    TemperatureSensor(String sensor_id, int sensor_port,
                      InetAddress sensorCentral) {
        this.sensor_id = sensor_id;
        this.sensor_port = sensor_port;
        this.sensorCentral = sensorCentral;
    }

    private double getProbeValue() {
        return // get temperature from the sensor probe
    }

    private void init() {
        try {
            new Thread(this).start();

            String hostAddress = InetAddress.getLocalHost().getHostAddress();
            String request = String.format("REGISTER %s %s %d ", sensor_id,
            hostAddress, sensor_port);

            // Register at SensorCentral:
            regSock = new Socket(
                sensorCentral.getHostAddress(),
                sensorCentral.getPort());

            regSock.connect();

            // Accept clients registrations:
            sock = new ServerSocket(sensor_port);
            for (;;) {
                p = new DatagramPacket(new byte[65536], 65536);
                sock.receive(p);
                String request2 = new String(p.getData(), 0, p.getLength());
                String cmd = new Scanner(request2).next();
                if (cmd.equals("MONITOR")) {
                    clients.add(new InetAddress(p.getAddress(), p.getPort()));
                }
            }
        } catch (Exception x) {
        }
    }
}
```

```

public void run() {
    try {
        [ ] cs = new [ ]();
        for (;;) { // periodically, sends temperature to all the clients
            double currTemperature = getProbeValue(); // Reads the temperature probe...
            String msg = String.format("TEMPERATURE %s %f ", sensor_id,
                                     currTemperature );
            [ ] p = new [ ](msg.getBytes(),
                           msg.getBytes().length);

            for (InetSocketAddress i : clients) {
                p.[ ](i.[ ]());
                p.[ ](i.[ ]());
                cs.[ ](p) ;
            }

            Thread.sleep(50);
        }
    } catch (Exception x) {
    }
}

public static void main(String[] args) {
    if (args.length != 4) {
        System.err.println("use: java TemperatureSensor sensor_id sensor_port
                           sensor_central_ip sensor_central_port");
        System.exit(0);
    }
    int sensor_port = Integer.parseInt(args[1]);
    int central_port = Integer.parseInt(args[3]);
    new TemperatureSensor(args[0], sensor_port,
    new InetSocketAddress( args[2], central_port)).init();
}
}

```

Client.java

```
import java.net.* ;
import java.util.* ;

public class Client {

    public static void main(String[] args ) throws Exception {
        if( args.length < 3 ) {
            System.err.println("usar: java Client serverCentral serverCentralPort sensorId");
            System.exit(0) ;
        }
        String serverCentral = args[0] ;
        int port = Integer.parseInt( args[1]) ;
        String sensor_id = args[2] ;

        // A seguir, contactar a central e pedir a localização do sensorId
        String req = String.format("SENSOR %s ", sensor_id) ;
```

```
// A seguir, registar-se no sensor
```

```
// Receber todas as notificações do sensor e afixar temperatura no ecrã
```

```
    }
}
```