

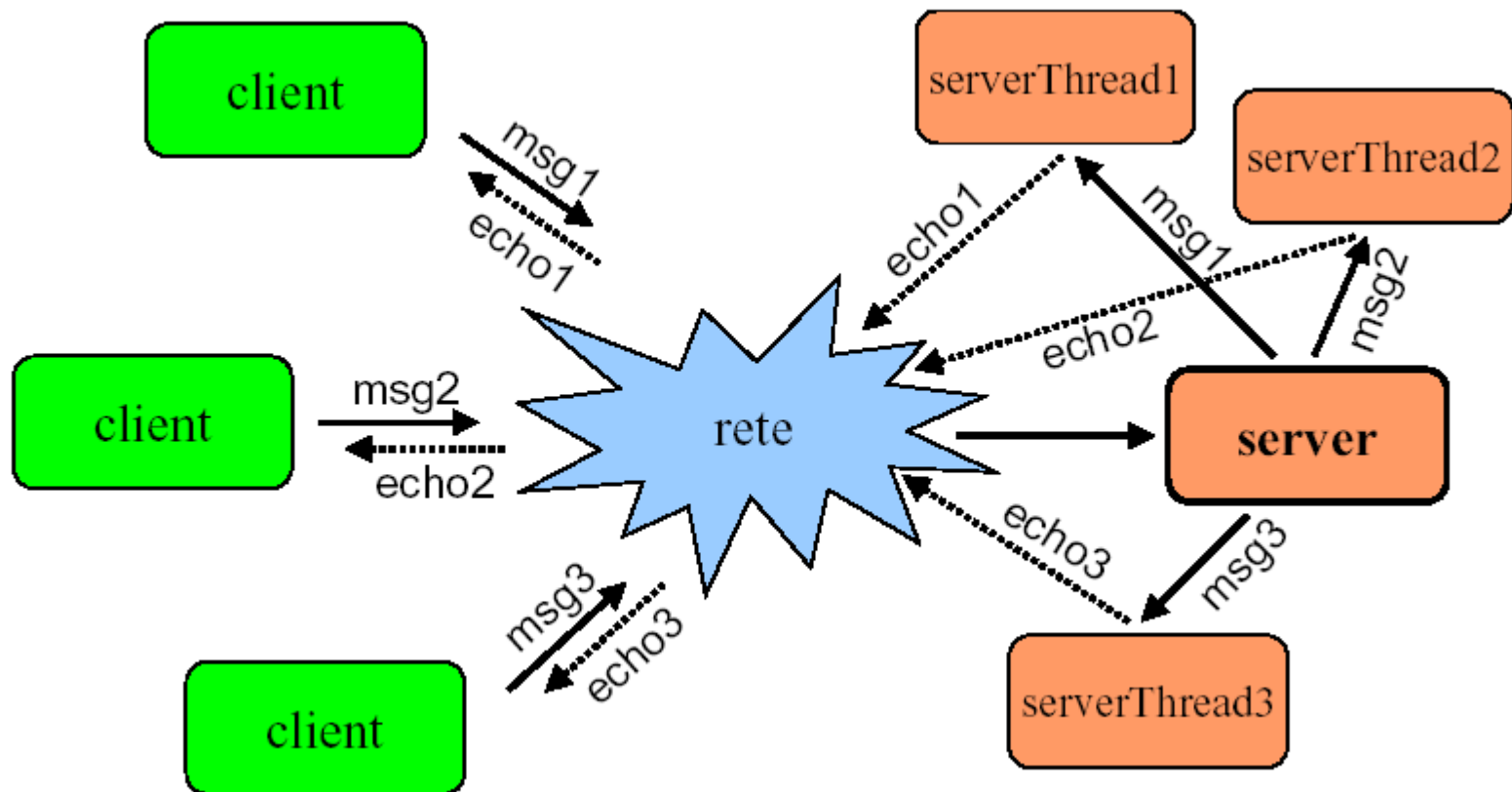
Università degli Studi della Calabria
Corso di Laurea in Ingegneria Gestionale
A.A. 2007/2008

Corso di
Sistemi di Elaborazione in Rete

Lucidi delle Esercitazioni

Agostino Forestiero

Multithreaded Echo Application



EchoMultiServer (3)

```
public class EchoMultiServer {
    public static void main(String[] args) throws IOException {
        ServerSocket serverSocket = new ServerSocket(1050);
        System.out.println("EchoMultiServer: started");
        try {
            while(true) {
                // bloccante finchè non avviene una connessione:
                Socket clientSocket = serverSocket.accept();
                System.out.println("Connection accepted: "+ clientSocket);
                try {
                    new ServerThread(clientSocket);
                } catch(IOException e) { clientSocket.close(); }
            }
        }
        catch (IOException e) {
            System.err.println("Accept failed");
            System.exit(1);
        }
        System.out.println("EchoMultiServer: closing...");
        serverSocket.close();
    }
} // EchoMultiServer
```

EchoMultiServer (1)

```
import java.io.*;
import java.net.*;

class ServerThread extends Thread {
    private static int counter = 0;
    private int id = ++counter;
    private Socket socket;
    private BufferedReader in;
    private PrintWriter out;

    public ServerThread(Socket s) throws IOException {
        socket = s;
        in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
        OutputStreamWriter osw = new OutputStreamWriter(socket.getOutputStream());
        out = new PrintWriter(new BufferedWriter(osw), true);
        start();
        System.out.println("ServerThread "+id+": started");
        System.out.println("Socket: " + s);
    }
}
```

EchoMultiServer (2)

```
public void run() {
    try {
        while (true) {
            String str = in.readLine();
            if (str.equals("END")) break;
            System.out.println("ServerThread "+id+": echoing -> " + str);
            out.println(str);
        }
        System.out.println("ServerThread "+id+": closing...");
    } catch (IOException e) {}
    try {
        socket.close();
    } catch (IOException e) {}
}
} // ServerThread
```



```
import java.net.*;  
import java.io.*;
```

EchoMultiClient (1)

```
class ClientThread extends Thread {  
    private Socket socket;  
    private BufferedReader in;  
    private PrintWriter out;  
    private static int counter = 0;  
    private int id = counter++;  
    private static int threadcount = 0;  
  
    public static int threadCount() {  
        return threadcount;  
    }  
    public ClientThread(InetAddress addr) {  
        threadcount++;  
        try {  
            socket = new Socket(addr, 1050);  
            System.out.println("EchoClient n° "+id+": started");  
            System.out.println("Client Socket: "+ socket);  
        } catch(IOException e) {}  
    }  
}
```

// Se la creazione della socket fallisce non è necessario fare nulla

EchoMultiClient (2)

```
try {
    InputStreamReader isr = new InputStreamReader(socket.getInputStream());
    in = new BufferedReader(isr);
    OutputStreamWriter osw = new OutputStreamWriter(socket.getOutputStream());
    out = new PrintWriter(new BufferedWriter(osw), true);
    start();
} catch(IOException e1) {
    // in seguito ad ogni fallimento la socket deve essere chiusa, altrimenti
    // verrà chiusa dal metodo run() del thread
    try {
        socket.close();
    } catch(IOException e2) {}
}
```

EchoMultiClient (3)

```
public void run() {  
    try {  
        for(int i =0;i <10; i++) {  
            out.println("client "+id +" msg "+i);  
            System.out.println("Msg sent: client "+id+" msg"+i);  
            String str = in.readLine();  
            System.out.println("Echo: "+str);  
        }  
        out.println("END");  
    } catch(IOException e) {}  
    try {  
        System.out.println("Client "+id+" closing...");  
        socket.close();  
    } catch(IOException e) {}  
    threadcount--;  
}  
  
} // ClientThread
```


EchoMultiClient (4)

```
public class EchoMultiClient {
    static final int MAX_THREADS = 10;

    public static void main(String[] args) throws IOException, InterruptedException {
        InetAddress addr;
        if (args.length == 0) addr = InetAddress.getByName(null);
        else addr = InetAddress.getByName(args[0]);

        while(true) {
            if (ClientThread.threadCount() < MAX_THREADS)
                new ClientThread(addr);
            Thread.currentThread().sleep(1000);
        }
    }
} // EchoMultiClient
```

Trasmissione di oggetti serializzati (1)

```
import java.io.*;

public class Studente implements Serializable {
    private int matricola;
    private String nome, cognome, corsoDiLaurea;
    public Studente (int matricola, String nome, String cognome,
                     String corsoDiLaurea) {
        this.matricola = matricola; this.nome = nome;
        this.cognome = cognome; this.corsoDiLaurea = corsoDiLaurea;
    }
    public int getMatricola () { return matricola; }
    public String getNome () { return nome; }
    public String getCognome () { return cognome; }
    public String getCorsoDiLaurea () { return corsoDiLaurea; }
}
```

Trasmissione di oggetti serializzati (2)

```
import java.io.*;
import java.net.*;
public class SendObject {
    public static void main (String args[]) {
        try {
            ServerSocket server = new ServerSocket (3575);
            Socket client = server.accept();
            ObjectOutputStream output =
                new ObjectOutputStream (client.getOutputStream ());
            output.writeObject("<Welcome>");
            Studente studente = new Studente (14520,"Leonardo","da Vinci","Ing. Informatica");
            output.writeObject(studente);
            output.writeObject("<Goodbye>");
            in.close();out.close();
            client.close();
            server.close();
        } catch (Exception e) { System.err.println (e); }
    }
}
```

Trasmissione di oggetti serializzati (3)

```
import java.io.*;
import java.net.*;
public class ReceiveObject {
    public static void main (String args[]) {
        try {
            Socket socket = new Socket ("localhost",3575);
            ObjectInputStream input =
                new ObjectInputStream (socket.getInputStream ());
            String beginMessage = (String)input.readObject();
            System.out.println (beginMessage);
            Studente studente = (Studente)input.readObject();
            System.out.print (studente.getMatricola()+" - ");
            System.out.print (studente.getNome()+studente.getCognome());
            System.out.print (studente.getCorsoDiLaurea()+"\n");
            String endMessage = (String)input.readObject();
            System.out.println (endMessage);
            socket.close();
        } catch (Exception e) { System.err.println (e); }
    }
}
```