Soluzione Esercizio 2

```java
// Parametro.java
import com.thoughtworks.xstream.*;

public class Parametro {
  private boolean stabilitaCampioni;
  private double sommaCampioni;
  private int numeroCampioni;

  public Parametro(double c) {
    sommaCampioni = c; numeroCampioni = 1; stabilitaCampioni = false;
  }

  public Parametro(String xml) {
    Parametro p = (Parametro)(new XStream()).fromXML(xml);
    numeroCampioni = p.numeroCampioni;
    stabilitaCampioni = p.stabilitaCampioni;
    sommaCampioni = p.sommaCampioni;
  }

  public void aggiungiCampione (double c) {
    if ((Math.round(sommaCampioni/numeroCampioni) !=
         Math.round((sommaCampioni+c)/(numeroCampioni+1)))
    ) { sommaCampioni+=c; numeroCampioni++; }
    else
      stabilitaCampioni = true;
  }

  public boolean stabilitaCampioni() {
    return stabilitaCampioni;
  }

  public double calcolaParametro() {
    return Math.round(sommaCampioni/numeroCampioni);
  }

  public String toString() {
    return (new XStream()).toXML(this);
  }
}
```

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

public class Terminale {

  public static void inviaComeStringa(int porta, Parametro p) {
    try (Socket sock = new Socket("localhost", porta);
```

```
 9       DataOutputStream dos =
10         new DataOutputStream(sock.getOutputStream());
11     ) {  dos.writeUTF(p.toString());
12     } catch (IOException e) { e.printStackTrace();}
13     System.out.println("- invio a " + porta); //2
14   }
15
16   public static Parametro riceviComeStringa(int porta) {
17     Parametro p = null;
18     try (
19       ServerSocket servsock = new ServerSocket(porta);
20       Socket sock = servsock.accept();
21       DataInputStream dis =
22          new DataInputStream(sock.getInputStream());
23     ) {
24       p = new Parametro(dis.readUTF());
25     } catch (IOException e) { e.printStackTrace();}
26     System.out.println("- ricevo\n" + p); //2
27     return p;
28   }
29
30   public static void main(String[] args) {
31     System.out.println("- sono " + args[1]); //2
32     Parametro p;
33     if (args[1].equals("8080")) { // è la radice
34       p = new Parametro(Double.parseDouble(args[0])); //1
35       System.out.println(p); //2
36       inviaComeStringa(Integer.parseInt(args[2]), p); //3
37       p = riceviComeStringa(Integer.parseInt(args[1])); //6
38       System.out.println(p.calcolaParametro()); //7
39     }
40     else {  //non è la radice
41       p = riceviComeStringa(Integer.parseInt(args[1])); //4
42       if (p.stabilitaCampioni()) {  //5.a
43         inviaComeStringa(Integer.parseInt(args[2]), p); //5.a.1
44       } else { //5.b
45         p.aggiungiCampione(Double.parseDouble(args[0])); //5.b.1
46         System.out.println("- aggiunto campione\n" + p); //5.b.2
47         if (args.length < 4) { //5.b.3.c foglia
48           inviaComeStringa(Integer.parseInt(args[2]), p); //5.b.3.c.1
49         } else { // 5.b.3.d nodo
50           if (!p.stabilitaCampioni()) { //5.b.3.d.e nodo e campioni instabili
51             inviaComeStringa(Integer.parseInt(args[3]), p); //5.b.3.d.e.1
52             p = riceviComeStringa(Integer.parseInt(args[1])); //5.b.3.d.e.2
53           }
54           inviaComeStringa(Integer.parseInt(args[2]), p); //  //5.b.3.d.1
55         }
56       }
57     }
58   }
59 }
60
```