```java
import java.io.BufferedWriter;
import java.io.ByteArrayInputStream;
import java.io.ByteArrayOutputStream;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.FileWriter;
import java.net.DatagramPacket;
import java.net.Socket;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.net.*;

public class BootstrapServer_KeepAlive extends Thread {
		int k, network;
		ArrayList<String> storageNodeList;
		List<String>messList;
		BootstrapServer_Implementation BS;
		BootstrapServer_Monitor BSM;
		DatagramSocket ds;
		FileWriter fw;
		BufferedWriter bw;
		DateFormat dateFormat;
		Date date;

		public BootstrapServer_KeepAlive(BootstrapServer_Implementation BS, BootstrapServer_Monitor BSM,
int network) {
			this.setDaemon(true);
			storageNodeList = new ArrayList();
			messList = new ArrayList();
			this.BSM = BSM;
			this.BS = BS;
			this.network = network;
			dateFormat = new SimpleDateFormat("yyyy/MM/dd HH:mm:ss");
			date = new Date();

			try {
					fw = new FileWriter("KeepAlive.txt");
					bw = new BufferedWriter(fw);
					bw.write("KeepAlive");
					bw.newLine();
					bw.flush();
					ds = new DatagramSocket(10000);
				} //apro una socket UDP in ingresso
			catch (Exception e) {
					e.printStackTrace();
				}
		}

		public void run() {
			try {
					ds.setSoTimeout(5000); //timeout di ascolto di 5 secondi

					while (true) {
							bw.write
("_____________________________________________");
							bw.newLine();
							bw.write(new Date() + " Inizio ascolto:");
							bw.newLine();

							try {
									while (true) {
```

```
                                                                byte []data = new byte
[200];

DatagramPacket (data, data.length);

                                                                DatagramPacket dp = new

                                                                ds.receive(dp);
                                                                ByteArrayInputStream bin =
new ByteArrayInputStream(dp.getData(), 0, dp.getLength());

DataInputStream(bin);

                                                                DataInputStream din = new

in ciascun pacchetto viene aggiunta alla lista
                                                                //ogni stringa contenuta

());
                                                                messList.add(din.readUTF

Ricevuto messaggio " + messList.get(messList.size() - 1));
                                                                bw.write(new Date() + "

                                                                bw.newLine();
                                                        }

                                                } catch (SocketTimeoutException e) //Timeout!
                                                {bw.write(new Date() + " Timeout");
                                                        bw.newLine();
                                                        bw.newLine();
                                                        bw.flush();} catch (Exception e) {
                                                        e.printStackTrace();
                                                }

                                        boolean trovato, daAggiornare = false;
                                        ByteArrayOutputStream bout = new ByteArrayOutputStream( );
                                        DataOutputStream dos = new DataOutputStream(bout);
                                        storageNodeList = BS.getStorageNodeList(); //prendo dal
BootStrapServer la lista aggiornata
                                        bw.write(new Date() + " Tabella BootstrapNode:");
                                        bw.newLine();

                                        for (int i = 0;i < storageNodeList.size();i++) {
                                                        bw.write(new Date() + " " +
storageNodeList.get(i));
                                                        bw.newLine();
                                                }

                                        bw.newLine();
                                        bw.flush();

                                        for (int i = storageNodeList.size() - 1;i >= 0;i--) {
                                                        trovato = false;

                                                        for (int j = messList.size() - 1;j >=
0;j--) {
                                                                String senderMessage =
messList.get(j);
                                                                //ricavo il numero della
porta della socket UDP del StorageeNode_KeepAlive in questione
                                                                int senderPort =
                                                                    Integer.parseInt
(senderMessage.substring(0, senderMessage.indexOf(" ip=")));
                                                                senderMessage =
senderMessage.substring(senderMessage.indexOf(" ip=") + 4);

                                                                if (senderMessage.equals
(storageNodeList.get(i))) {
                                                                        trovato =
true;
messList.remove(j);
```

```
                                                                                InetAddress
ia = InetAddress.getLocalHost();

                                                                                if (network
== 1) //versione di rete
                                                                                        ia
= InetAddress.getByName(storageNodeList.get(i).substring(0, storageNodeList.get(i).indexOf(" UDPport=")));


dos.writeUTF("ok"); //avviso il nodo che ho ricevuto il suo pacchetto

                                                                                byte [ ]
data = bout.toByteArray();


DatagramPacket dp = new DatagramPacket(data, data.length, ia, senderPort);

                                                                                ds.send
(dp);

                                                                                bout.reset
();

                                                                                break;
                                                                        }
                                                                }

                                                        if (!trovato) { //il nodo in posizione i
non ha risposto
                                                                storageNodeList.remove(i);
                                                                daAggiornare = true; //ho
modificato la lista dei nodi di boostrap: devo aggiornare quella del Bootstrap Server
                                                                }
                                                        }

                                                bw.write(new Date() + " Tabella BootstrapNode
Aggiornata:");
                                                bw.newLine();

                                                for (int i = 0;i < storageNodeList.size();i++) {
                                                        bw.write(new Date() + " " +
storageNodeList.get(i));

                                                        bw.newLine();
                                                }

                                                bw.flush();
                                                //se ho modificato la lista, cerco di accedere alla lista
(ho la priorità sia sui nuovi nodi che sui client)

                                                if (daAggiornare) {
                                                        BSM.start_keep_alive();
                                                        BS.setStorageNodeList(storageNodeList);
                                                        BSM.end_keep_alive();
                                                }

                                                //elimino tutti gli elementi rimanenti in messList
(possibili risposte perse dei vecchi bootstrap node)
                                                for (int i = messList.size() - 1;i >= 0;i--)
                                                        messList.remove(i);
                                        }

                        } catch (Exception e) {
                                e.printStackTrace();
```

```
                    }
            }
}
```