ESERCITAZIONE 7

Java RMI e riferimenti remoti

- Bernardi Daniel
- Chichifoi Karina
- Gjura Endri
- Ivan Andrei Daniel
- Pizzini Cavagna Hiari

Introduzione

To be defined

Client

```
// Ricerca per nome del servizio o per tag
while (!check) {
      System.out.println("Scegliere il tipo di ricerca (T = Tag, S = service name):");
      foo = stdIn.readLine();
      if (foo == "T") {
            System.out.println("Richiesta ricerca per tag\n" + "Inserire Tag richiesto");
            serviceTag = stdIn.readLine();
            String [] serversWithTag = (String []) registryRemoto.cercaTag(serviceTag);
            if (serversWithTag.length > 0) {
                  for (String nameS : serversWithTag) {
                        System.out.println(nameS);
                  }
                  check = true;
            } else {
                  System.out.println("Nessun elemento trovato");
            }
      }
      else if (foo == "S" || check == true) {
            System.out.println("Richiesta ricerca per service name\n" + "Inserire service richiesto");
            serviceName = stdIn.readLine();
            check = true;
      } else System.out.println("Opzione non disponibile.");
}
```

RegistryRemotoTagImpl (1)

```
public synchronized boolean associaTag(String nome_logico_server, String tag) {
      boolean risultato = false;
      if ((nome_logico_server == null) || !itag.check(tag)) {
            return risultato;
      }

      for (int i = 0; i < tableSize; i++) {
            if (nome_logico_server.equals((String) table[i][0])) {
                  risultato = true;
                  table[i][2] = tag;
                  break;
            }
      }

      return risultato;
}
```

RegistryRemotoTagImpl (2)

```
public synchronized String[] cercaTag(String tag) throws RemoteException {
	int cont = 0;
	if (tag == null) {
		return new String[0];
	}
	for (int i = 0; i < tableSize; i++) {
		if (tag.equals((String) table[i][2])) {
			cont++;
		}
	}
	String[] risultato = new String[cont];

	cont = 0;
	for (int i = 0; i < tableSize; i++) {
		if (tag.equals((String) table[i][2])) {
			risultato[cont++] = (String) table[i][0];
		}
	}

	return risultato;
}
```

ServerImpl

```
try {
	RegistryRemotoTagServer registryRemoto = (RegistryRemotoTagImpl) Naming.lookup(completeRemoteRegistryName);
	ServerImpl ServerImplRMI = new ServerImpl();
	registryRemoto.aggiungi(serviceName, ServerImplRMI);
	System.out.println("ServerImpl RMI: Servizio \"" + serviceName + "\" registrato");

	if (args.length == 4) {
		boolean association = registryRemoto.associaTag(serviceName, args[3]);
		if (association) {
			System.out.println("ServerImpl RMI: Tag associato correttamente");
		} else {
			System.out.println("ServerImpl RMI: Tag scelto non disponibile");
		}
	}
}
catch (Exception e) {…}
```

Tag

```java
public class Tag implements Serializable {

    private static final long serialVersionUID = 1L;
    public String [] tags = new String [10];

    public Tag() {
        tags[0] = "Bologna";
        tags[1] = "Milano";
        tags[2] = "Roma";
        tags[3] = "Torino";
        tags[4] = "Palermo";
        tags[5] = "Padova";
        tags[6] = "Firenze";
        tags[7] = "Genova";
        tags[8] = "Reggio Calabria";
        tags[9] = "Modena";
    }

    public boolean check(String str) {
        for (int i = 0; i < 10; i++) {
            if (str.equals(tags[i])) {
                return true;
            }
        }
        return false;
    }
}
```

Conclusione

To be defined