

ESERCITAZIONE 7

Java RMI e Riferimenti Remoti RMI Registry Remoto come pagine gialle

OBIETTIVI

Si progetti un servizio di pagine gialle distribuito:

- I servitori possono registrarsi con nome del servizio e localizzazione di deployment. Ai servizi sono associati dei Tag.
- I clienti interrogano il RegistryRemoto attraverso i tag che descrivono i servizi



SPECIFICHE

Metodi remoti* invocabili da:

CLIENTI:

- cerca (String nomeLogico)
- cercaTutti (String nomeLogico)
- cercaTag (String tag)

SERVER:

- restituisciTutti ()
- eliminaTutti (String nomeLogico)
- eliminaPrimo (String nomeLogico)
- aggiungi (String nomeLogico, Remote riferimento)
- associaTag (String nome_logico_server, String tag)

*i metodi sono tutti implementati dalla classe RegistryRemotoImpl.java

INTERFACCE

```
public interface RegistryRemotoClient extends Remote {  
    public Remote cerca(String nomeLogico) throws RemoteException;  
  
    public Remote[] cercaTutti(String nomeLogico) throws RemoteException;  
}
```

```
import java.rmi.RemoteException;  
  
public interface RegistryRemotoTagClient extends RegistryRemotoClient {  
    public String[] cercaTag(String tag) throws RemoteException;  
}
```

 Interfacce cliente

```
public interface RegistryRemotoServer extends RegistryRemotoClient {  
    // Tabella: la prima colonna i nomi, la seconda i riferimenti remoti  
    public Object[][] restituisciTutti() throws RemoteException;  
  
    public boolean aggiungi(String nomeLogico, Remote riferimento) throws RemoteException;  
  
    public boolean eliminaPrimo(String nomeLogico) throws RemoteException;  
  
    public boolean eliminaTutti(String nomeLogico) throws RemoteException;  
}
```

```
import java.rmi.RemoteException;  
  
public interface RegistryRemotoTagServer extends RegistryRemotoServer, RegistryRemotoTagClient {  
    public boolean associaTag(String nome_logico_server, String tag) throws RemoteException;  
}
```

 Interfacce server



STRUTTURA DATI

```
public class RegistryRemotoTagImpl extends UnicastRemoteObject implements RegistryRemotoTagServer {  
    final int tableSize = 100;  
    final int maxTag = 10;  
    // Tabella: la prima colonna contiene i nomi, la seconda i riferimenti remoti,  
    // la terza i tag  
    Object[][] table = new Object[tableSize][3];  
  
    // Costruttore  
    public RegistryRemotoTagImpl() throws RemoteException {  
        super();  
        for (int i = 0; i < tableSize; i++) {  
            table[i][0] = null;  
            table[i][1] = null;  
            table[i][2] = new String[maxTag];  
        }  
    }  
}
```



REGISTRY REMOTO

```
public synchronized boolean associaTag(String nome_logico_server, String tag) throws RemoteException{  
    int j=0;  
    System.out.println(tag);  
    if (nome_logico_server == null)  
        return false;  
    for (int i = 0; i < tableSize; i++)  
        if (nome_logico_server.equals((String) table[i][0])) {  
            while(j<(maxTag)){  
                if(((String[]) table[i][2])[j] == null){  
                    ((String[])table[i][2])[j]=tag;  
                    return true;  
                }  
                j++;  
            }  
        }  
    return false;  
}
```

```
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++)  
        for (int j = 0; j < maxTag && ((String[]) table[i][2])[j]!=null; j++)  
            if ( tag.equals(((String[]) table[i][2])[j])) {  
                cont++;  
            }  
    String[] risultato = new String[cont];  
    // usato come indice per il riempimento  
    cont = 0;  
    for (int i = 0; i < tableSize; i++)  
        for (int j = 0; j < maxTag && ((String[]) table[i][2])[j]!=null; j++)  
            if (tag.equals(((String[]) table[i][2])[j])) {  
                risultato[cont++] = (String) table[i][0];  
            }  
    return risultato;  
}
```

CLIENTE

Differenza rispetto a Cliente esercitazione 6 /7 svolta:

- La ricerca avviene tramite un Registry Remoto su una macchina diversa dal Server
- Si consente all'utente di scegliere a quale servitore richiedere il servizio

```
try {
    String completeRemoteRegistryName = "/" + registryHost + ":" + REGISTRYPORT + "/" + registryRemotoName;
    RegistryRemotoTagClient registryRemoto = (RegistryRemotoTagClient) Naming.lookup(completeRemoteRegistryName);
    serviceList = registryRemoto.cercaTag(serviceTag);

    // Scelta ServiceTag
    do {
        System.out.println("Seleziona il numero del servitore:");
        for (int i = 1; i <= serviceList.length; i++) {
            System.out.println(i + "\t" + serviceList[i - 1]);
        }

        serviceNum = Integer.parseInt(stdin.readLine());
        // Controllo servitore
        if (serviceNum > 0 && serviceNum <= serviceList.length)
            serverRMI = (ServerCongresso) registryRemoto.cerca(serviceList[serviceNum]);
        else
            System.out.println("Numero servitore errato!");
    } while (serviceNum <= 0 || serviceNum > serviceList.length);
}
```





SERVER

- I server registrano il proprio nome logico sul Registry Remoto
- I Server associano uno o più Tag al proprio nome logico

```
try {  
    RegistryRemotoTagServer registryRemoto = (RegistryRemotoTagServer) Naming.lookup(completeRemoteRegistryName);  
    ServerCongressoImpl serverRMI = new ServerCongressoImpl();  
    registryRemoto.aggiungi(serviceName, serverRMI);  
  
    registryRemoto.associaTag(serviceName, "Congresso");  
}
```



CONCLUSIONI

- Ogni servitore ha la possibilità di associare un array di tag ad un suo servizio
- Abbiamo deciso di usare un array di Tag per registrare Tag multipli senza dover registrare il servizio più volte