Università degli Studi di Bologna
Scuola di Ingegneria

Corso di

Reti di Calcolatori T

Esercitazione 7 (proposta)

Java RMI e Riferimenti Remoti

RMI Registry Remoto come pagine gialle

Luca Foschini

Michele Solimando

Anno accademico 2016/2017

Servizio di pagine gialle distribuito

Si progetti **un servizio di nomi RegistryRemoto**, che **fornisce un servizio di pagine gialle ad utilizzatori su macchine diverse che intendano usarlo come Clienti o Servitori RMI,** superando il problema della loro collocazione rispetto ad un registry di RMI**.**

Il RegistryRemoto deve permettere:

- ai servitori di registrare la propria **disponibilità di servizio**, tenendo traccia del **nome del servizio** e della **localizzazione di deployment,** e deve associare al servizio una **lista di tag che descrivono il servizio stesso** in modo da permettere la **ricerca di una funzionalità in base alla descrizione** oltre che al nome;
- ai clienti di ottenere i **nomi logici** relativi ai servizi di cui hanno bisogno; i nomi logici **sono ottenuti interrogando il RegistryRemoto attraverso i tag** che descrivono il servizio.

Servizio di pagine gialle distribuito

Il **RegistryRemoto** è realizzato come **server RMI** e deve poi consentire un'invocazione dei servizi da parte dei clienti attraverso riferimenti remoti.

Si progetti un **servizio di naming remoto** **con funzionalità di pagine gialle** (**RegistryRemoto**) che **consenta ai Client di recuperare i nomi logici relativi a oggetti remoti Server che si siano registrati** **a partire dal codice dell'esercitazione svolta**.

Specifica: il RegistryRemoto

In particolare **RegistryRemoto** è realizzato come server RMI e implementa le seguenti operazioni per due tipologie di client.

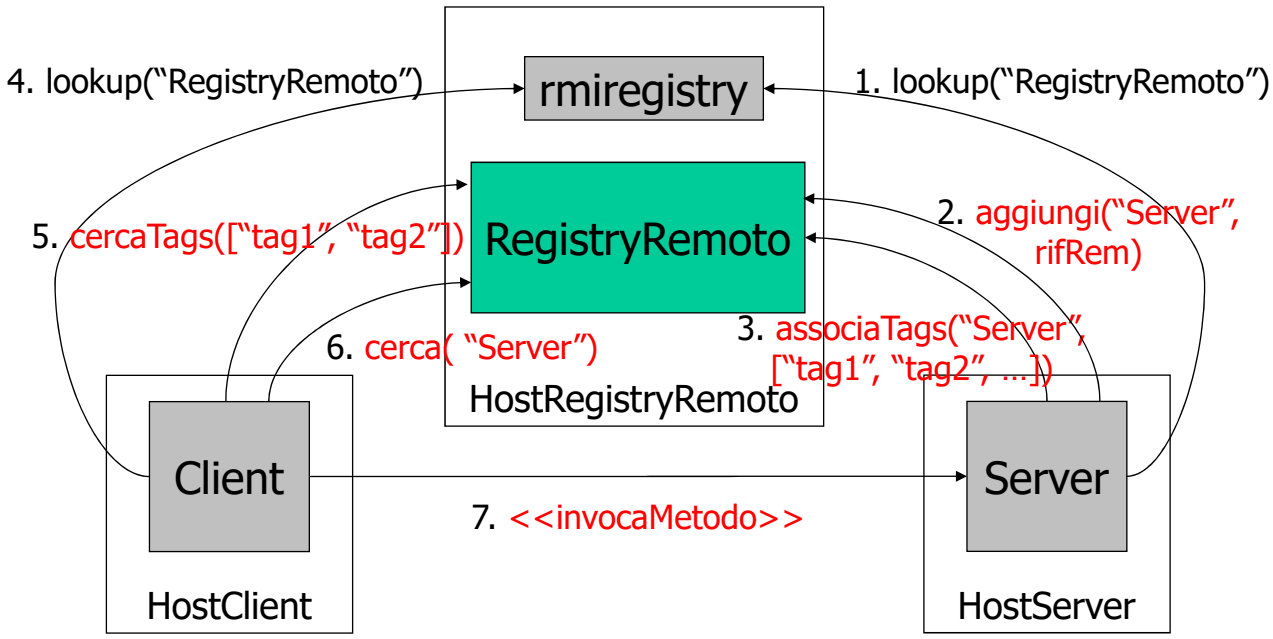
Per i clienti:

- **ricerca del primo riferimento** al server remoto registrato con il **nome logico** dato;
- **ricerca di tutti i riferimenti** ai server remoto registrati con lo stesso **nome logico**;
- **ricerca per tag (uno o più)** dei **nomi logici** dei servizi cercati.

Per i fornitori di servizio, oltre alle funzioni offerte ai client:

- **ottenimento lista di tutte le coppie nome logico/riferimento** mantenute dal RegistryRemoto (servizio **senza parametri di ingresso**);
- **aggiunta** di un server remoto, dato il **nome logico** e il **riferimento remoto**;
- **eliminazione della prima** entry corrispondente al **nome logico** dato;
- **eliminazione di tutte** le entry registrate con il **nome logico**.
- **associazione** di una lista di tag ad un **nome logico già registrato**

Architettura di riferimento



Progetto e sue parti

Il progetto RMI si compone, oltre alle classi già viste nell'esercitazione 6 e 7, delle ulteriori classi :

- **Un'interfaccia remota *RegistryRemotoTagClient*** (contenuta nel file *RegistryRemotoTagClient.java*) che estende ***RegistryRemotoClient*** in cui viene definito il nuovo metodo invocabile dai clienti (cercaTag);
- **Un'interfaccia remota *RegistryRemotoTagServer*** (contenuta nel file *RegistryRemotoTagServer.java*) che estende ***RegistryRemotoServer* e *RegistryRemotoTagClient*** aggiungendo il metodo invocabile dai servitori (associaTags(nome logico server, lista di tag));
- Una **classe per la realizzazione del RegistryRemoto** (***RegistryRemotoTagImpl*** contenuta nel file *RegistryRemotoTagImpl.java*), che implementa i metodi di RegistryRemotoTagServer invocabili in remoto.

Sarà inoltre necessario **modificare opportunamente Server e Client dell'esercitazione svolta** in modo che supportino la registrazione e la ricerca del riferimento all'oggetto Server, presso il RegistryRemoto, utilizzando il meccanismo della lista dei tag.

Proposta di estensione: servizio di nomi distribuito e coordinato

Sviluppare **un servizio di nomi distribuito e coordinato** partendo dal RegistryRemoto sviluppato nell'esercitazione svolta.

Si ipotizzi che sia disponibile una molteplicità di RegistryRemoti che siano, da un punto di vista logico, in esecuzione su host diversi. In tale scenario si vuole realizzare un **FrontEnd per RegistryRemoti** che permetta i seguenti comportamenti:

- sia i *client* sia i *servitori* fanno riferimento per *lookup* e *registrazione* **allo stesso unico FrontEnd**;
- ciascun **RegistryRemoto** gestisce un sottoinsieme dello spazio dei nomi logici (**si dividano i nomi logici seguendo un criterio alfabetico**);
- il **FrontEnd** deve mantenere una **vista di tutti i RegistryRemoti disponibili**.

Specifica: il FrontEnd

Il **FrontEnd** è realizzato come server RMI e implementa le seguenti operazioni (si realizzino interfacce con scope diversi per clienti e RegistryRemoti):

- **ricerca del primo riferimento** registrato con un **nome logico** dato;
- **ricerca di tutti i riferimenti** registrati con uno stesso **nome logico**;
- **registrazione di un RegistryRemoto**, dati due caratteri (inizio e fine dell'intervallo di competenza nell'alfabeto) e il proprio **riferimento remoto**.

e mantiene una struttura dati:

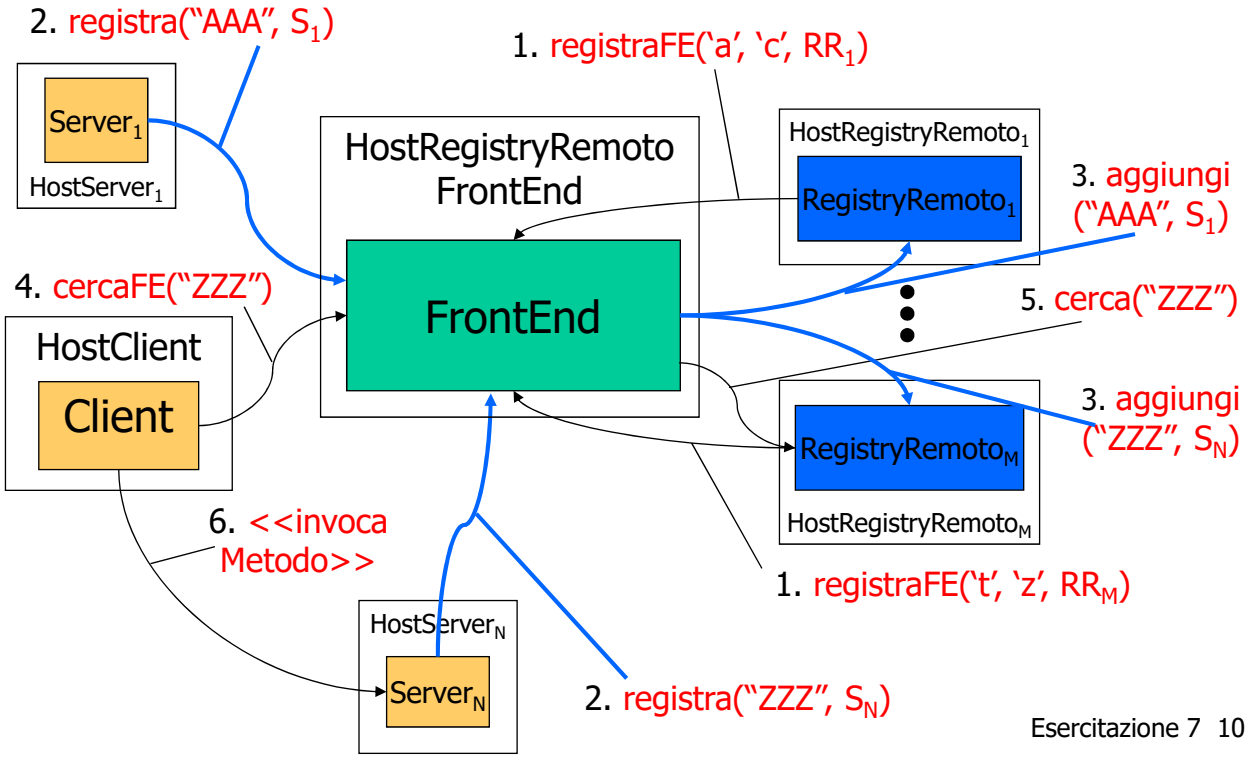
- una con **i riferimenti ai RegistryRemoti**;
- una con **le corrispondenze lettera iniziale/finale di competenza del nome logico per il RegistryRemoto** e **riferimento remoto**.

Metodi remoti

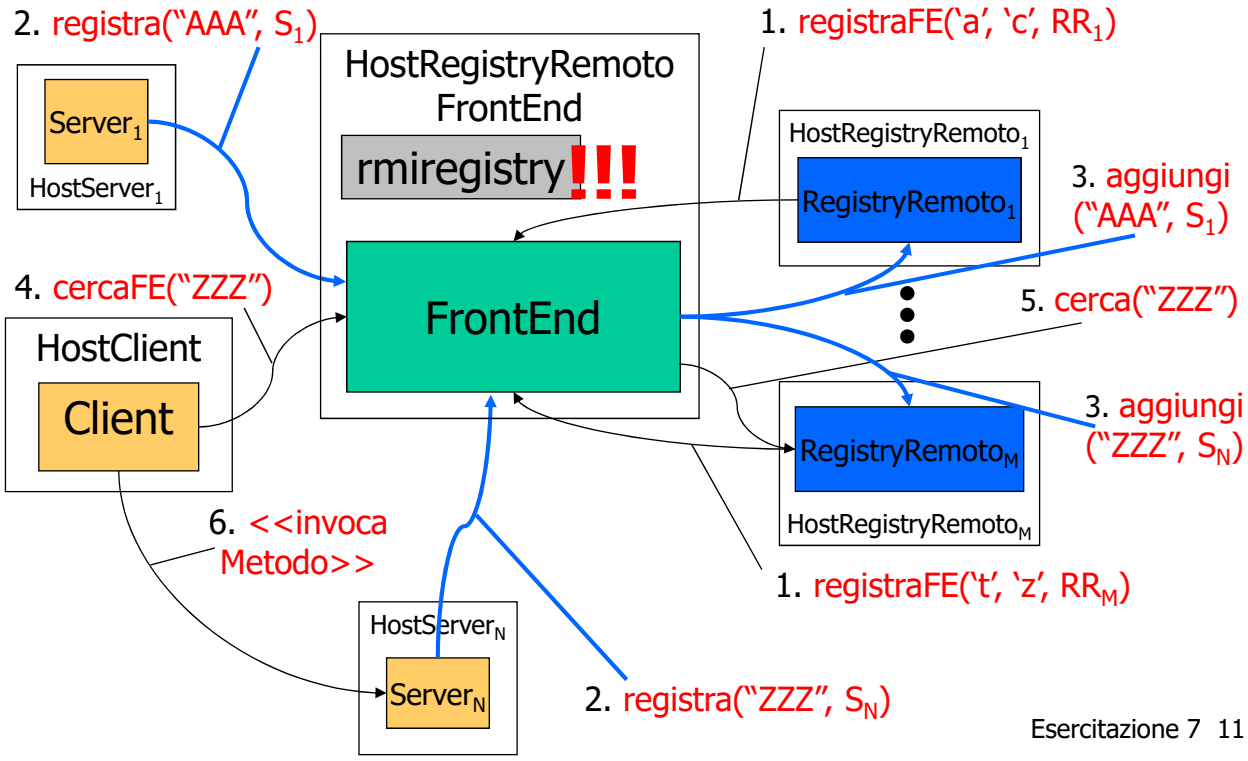
Il progetto RMI si basa su **due interfacce remotizzabili** (una per client/server e una per i RegistryRemoti) in cui vengono definiti i **metodi invocabili da client e RegistryRemoti**:

- Il metodo **`cercaFE`** accetta come parametro d'ingresso **il nome del servizio**, quindi restituisce il primo riferimento al servizio richiesto, oppure *null* se il servizio non è disponibile.
- Il metodo **`registra`** accetta come parametro d'ingresso **un nome logico** e **il riferimento del servizio**, quindi inserisce il riferimento nel RegistryRemoto che gestisce i nomi logici nell'intervallo richiesto.
- Il metodo **`registraFE`** accetta come parametri d'ingresso **due caratteri** e **il riferimento al RegistryRemoto** e lo inserisce nell'opportuna struttura dati.

Architettura di riferimento



Architettura di riferimento: dettagli implementativi



Classi in gioco

In aggiunta alle classi dell'esercitazione 7 svolta, si progettino le classi:

- **FrontEnd** (contenuta nel file FrontEnd.java), che implementa i metodi del invocabili in remoto e presenta l'interfaccia di invocazione:

FrontEnd [registryPort]

Si modifichi inoltre opportunamente:

- Il main del Client in modo da interrogare il FrontEnd al posto del RegistryRemoto;
- Il main del RegistryRemoto in modo da registrare il riferimento del RegistryRemoto presso il FrontEnd.

Consegna

Chi vuole può inviare lo svolgimento ai docenti, con lo strumento specificato sul sito del corso.