Università degli Studi di Bologna
Scuola di Ingegneria

Corso di
Reti di Calcolatori T

Esercitazione 6 (proposta)
Java RMI e Riferimenti Remoti
RMI Registry Remoto come pagine gialle

Antonio Corradi, Luca Foschini
Lorenzo Rosa, Giuseppe Martuscelli, Marco Torello
Anno accademico 2022/2023

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 registrarsi con **nome del servizio** e **localizzazione di deployment (come già visto)**.
  Ai servizi può essere associato uno dei **Tag disponibili e consentiti sul RegistryRemoto** che meglio descrivono il servizio stesso.
  In questo modo viene abilitata la possibilità di **ricercare una funzionalità in base alla descrizione contenuta nei Tag (in base ai Tag)**
- ai **clienti** di poter interrogare **il RegistryRemoto attraverso i tag** che descrivono i servizi, ottenendo il/i nome logici relativi ai servizi di cui hanno bisogno

SERVIZIO DI PAGINE GIALLE DISTRIBUITO

Si progetti un **servizio di naming remoto** **con funzionalità di pagine gialle** (**RegistryRemoto**) che **consenta ai Client di recuperare il nome logico relativo a oggetti remoti Server che si siano registrati** **a partire dal codice dell'esercitazione svolta**

- Il **RegistryRemoto** è realizzato come **server RMI** e deve poi consentire un'invocazione da parte **dei clienti con un tag** per ottenere dei **nomi logici di server** (di cui singolarmente ottenere riferimenti remoti dal RegistryRemoto)
- Il **RegistryRemoto** conserva, oltre che ai riferimenti remoti, tutti i possibili **Tag** che possono essere oggetto delle nuove operazoni di **pagine gialle**: **permette ai server di registrarsi con questi tag e ai clienti di ricercare con questi**
  Il RegistryRemoto deve controllare le richieste di registrazione e di ricerca per verificare che vengano usati Tag consentiti, e in caso contrario sollevare un'eccezione

SPECIFICA: IL REGISTRYREMOTO

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

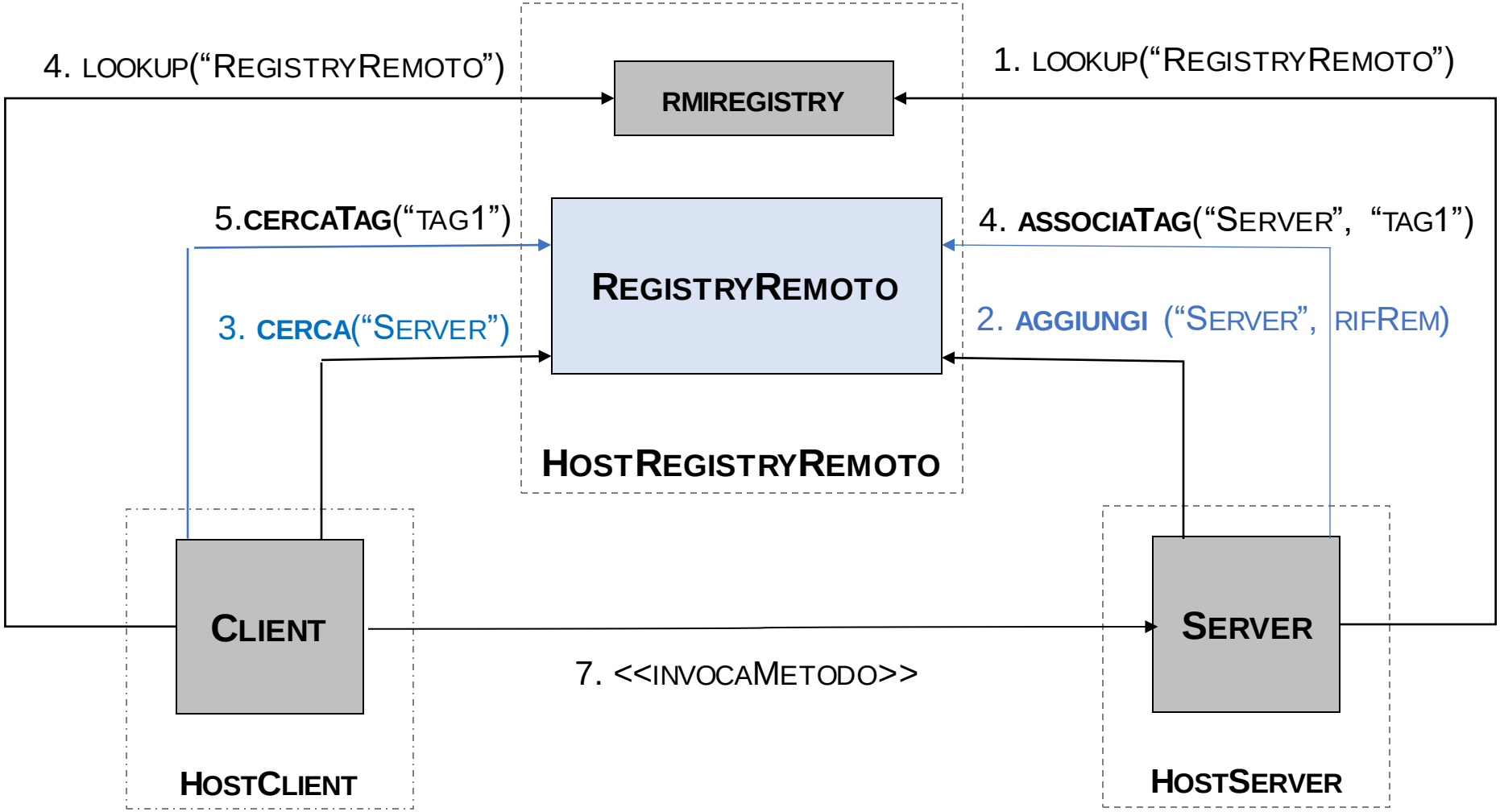
Per i clienti:

- **ricerca del primo riferimento** al server remoto registrato con il **nome logico** dato
- **ricerca di tutti i riferimenti** ai server remoti registrati con lo stesso **nome logico**
- **ricerca per tag** 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 un 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 5 e 6, 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(tag)`***) che viene usato dai clienti per cercare i servizi che fanno match con un tag
- **Un'interfaccia remota** ***RegistryRemotoTagServer*** (contenuta nel file *RegistryRemotoTagServer.java*) che estende ***RegistryRemotoServer*** **e** ***RegistryRemotoTagClient*** aggiungendo il metodo invocabile dai servitori (**`associaTag(nome_logico_server, 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 del 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**:

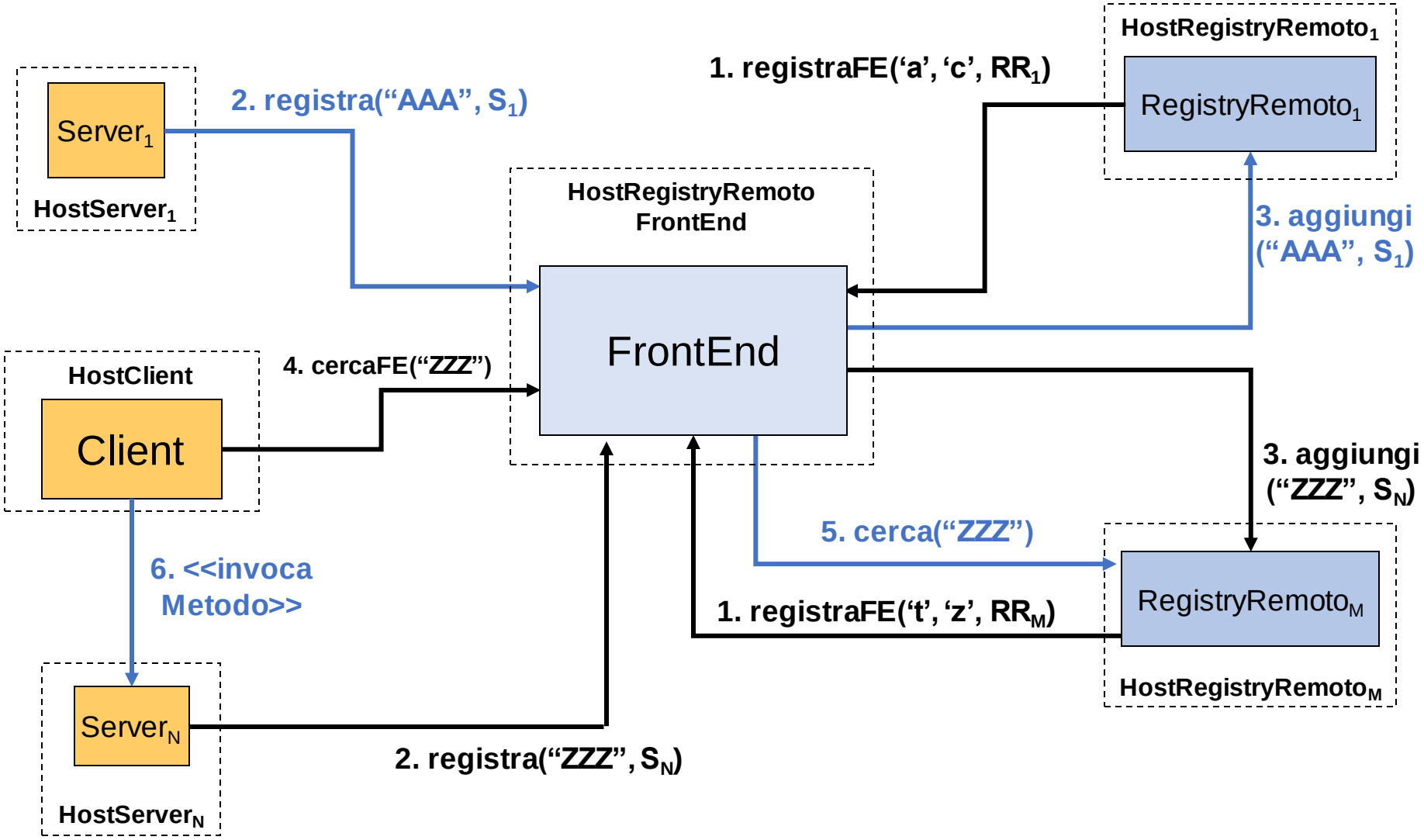
- una con **i riferimenti ai RegistryRemoti**
- una con **le corrispondenze lettera iniziale/finale di competenza del nome logico per il RegistryRemoto** e **riferimento remoto**

METODI REMOTI

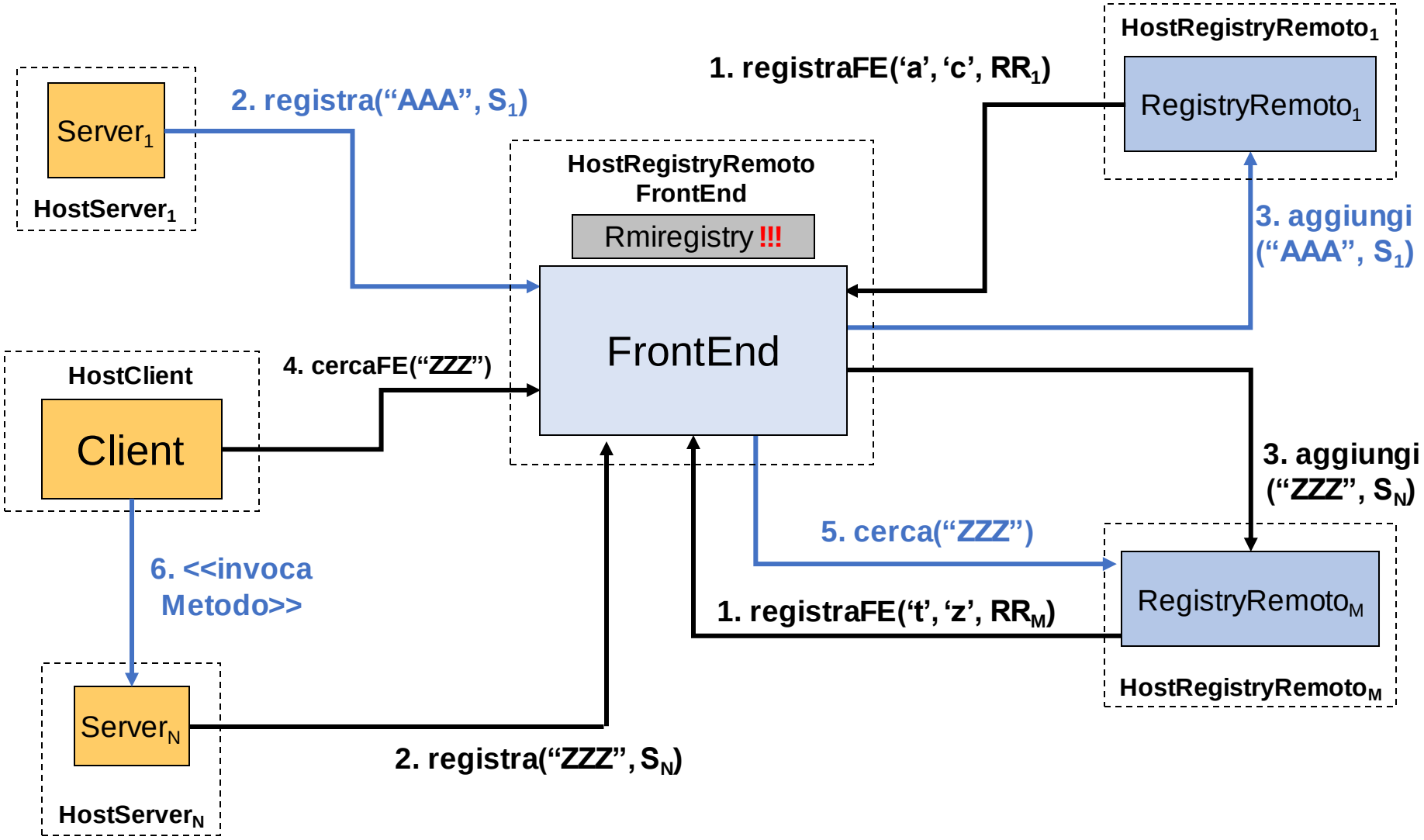
Il progetto RMI si basa su **due interfacce remotizzabili** (una per client/server e una per i RegistryRemoti) in cui vengono definiti i **metodi invocabili da client e RegistryRemoti**:

- 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



CLASSI IN GIOCO

In aggiunta alle classi dell'esercitazione 6 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