Prova finale di Ingegneria del Software  
Documento di progetto

# TEMA COMUNE

Realizzare una libreria in Java per la comunicazione remota tra sistemi.

La libreria deve implementare **in modo preciso** la specifica del protocollo JSON-RPC.

I dettagli del protocollo sono descritti qui: <http://www.jsonrpc.org/specification>

La libreria deve consentire di implementare client e server JSON-RPC:

* Un server deve poter ricevere richieste o notifiche e inviare risposte (ove queste siano previste dalla specifica).
* Un client deve poter inviare richieste o notifiche e ricevere risposte (ove queste siano previste dalla specifica).

Allo scopo di garantire la comunicazione remota, è necessario selezionare un “canale di comunicazione” su cui il protocollo è implementato.

Si suggerisce di rendere la libreria agnostica rispetto al canale di comunicazione, tramite un’apposita interfaccia e/o factory.

Come canale si suggerisce fortemente l’uso di un approccio a code, come ad esempio ZeroMQ.

Le risorse di ZeroMQ sono accessibili qui: <http://zeromq.org/> (e verranno introdotte a lezione).

Una volta realizzato il tema comune, è possibile utilizzare la libreria ottenuta per implementare una specifica applicazione. In particolare:

* Ogni gruppo può scegliere se realizzare l’applicazione A) o B)
* Inoltre, è possibile ottenere 3 punti di bonus aggiuntivo se due gruppi si coordinano per usare le due applicazioni insieme: vedere descrizione dell’applicazione I (Integration).

# Tema B) Broker

Si vuole realizzare un sistema che fa da broker per servizi online, in modo trasparente.

Il sistema detiene una lista di servizi online disponibili e consente di registrare nuovi servizi o cancellarne di esistenti.

Ogni servizio è descritto da un identificativo di servizio e un formato di input e output. Inoltre, sono disponibili una serie di metadati: proprietario, settore, parole chiave, titolo, descrizione, data di attivazione.

Il broker associa ad ogni servizio un metodo JSON-RPC che lo identifica in modo univoco e che ne permette l’invocazione trasparente (più client possono esporre servizi con lo stesso identificativo di servizio, ad esempio “Somma”).

E’ possibile richiedere al broker (anch’esso implementato tramite metodi JSON-RPC):

* Di restituire un elenco di servizi in base a una ricerca (una o più parola, che viene cercata in tutti i metadati). Il risultato è una lista di servizi con tutta la loro descrizione e il metodo JSON-RPC associato. Ad esempio se cerco “Somma” e ci sono più servizi con questo identificativo di servizio, il broker ritorna tutti questi servizi e il loro identificativo JSON-RPC (“SommaXX”)
* Di invocare, in modo trasparente, un servizio, tramite il metodo JSON-RPC associato (“SommaXX”).

Quando un client si disconnette dal broker i suoi servizi vengono automaticamente cancellati ed eventuali richieste pendenti completate con il codice di errore più opportuno.

Si può supporre che il broker e i servizi siano invocabili tramite quanto specificato nel tema comune.

# Sintesi dei Goal

## Goal tema comune:

* Obiettivi Client
  + Generate Request:

Generare un Request Object (ovvero una call rpc).

* + Handle Response
* Obiettivi Server
  + Generate Response
  + Handle Request

## Goal tema B) Broker

* Obiettivi Client
  + Invoke a Service:

Invocare un servizio fra quelli registrati nel sistema. (Se si conosce il servizio non è necessario richiedere informazioni riguardo i servizi disponibili.). Il Client deve poter dare la possibilità all’utente di usufruire dei servizi disponibili nel Broker.

* + Request a list of desired services:

Richiedere una lista, basata su una ricerca, di servizi registrati nel sistema. Al fine di comunicare all’utente come utilizzare ciascun servizio è necessario che il Client sia in possesso della lista.

* Obiettivi Server
  + Publish a Service:

Pubblicare il servizio che il Server vuole fornire al sistema. Il Server deve poter essere raggiunto nel dal Broker qualora un Client volesse usufruire (attraverso il canale di comunicazione trasparente stabilito dal Broker) del suo servizio.

* + Delete the own Service:

Cancellare il servizio che il Server ha pubblicato nel sistema. Dev’essere sempre possibile per il Server di cancellare il proprio servizio dal Broker.

* Obiettivi Broker
  + Make Services available to all connected Clients
  + Create a transparent communication
  + Manage efficiently services
* Obiettivi comuni (Server-Broker)
  + Delete an available Service