RMI: Remote Method Invocation

Integrantes:

- Gómez Bobadilla Julio
- León Mamani Rolando
- Tupayachi Moina Miguel
- Vega Colque Milagros

Índice

1. Introducción

1. Introducción

Hemos visto que el uso de sockets permite la elaboración de aplicaciones distribuidas.

Sin embargo, son la abstracción de bajo nivel (**nivel de transporte**) y en ocasiones necesitamos un grado mayor de abstracción (**nivel de aplicación**).

2. Definición

2. Definición de RMI

RMI (Remote Method Invocation) es un mecanismo ofrecido por Java para invocar un método de manera remota.

Proporciona un mecanismo simple para la comunicación en aplicaciones distribuidas basadas exclusivamente en Java.

Para comunicación entre otras tecnologías: CORBA o SOAP

Idea básica


Cliente
Servidor
Señala método y
Envía Parámetro
Invoca método
Retorna objeto respuesta

¿Porque se caracteriza RMI?

RMI se caracteriza por la **facilidad de su uso** en la programación.

Solo debemos crear un objeto con sus métodos y usarlos en alguna máquina remota

3. Ventajas y Desventajas

Ventaja

Simplicidad:

Permite distribuir una aplicación de forma muy transparente, es decir, sin que el programador tenga que modificar apenas el código.

Desventaja

Solo Java:

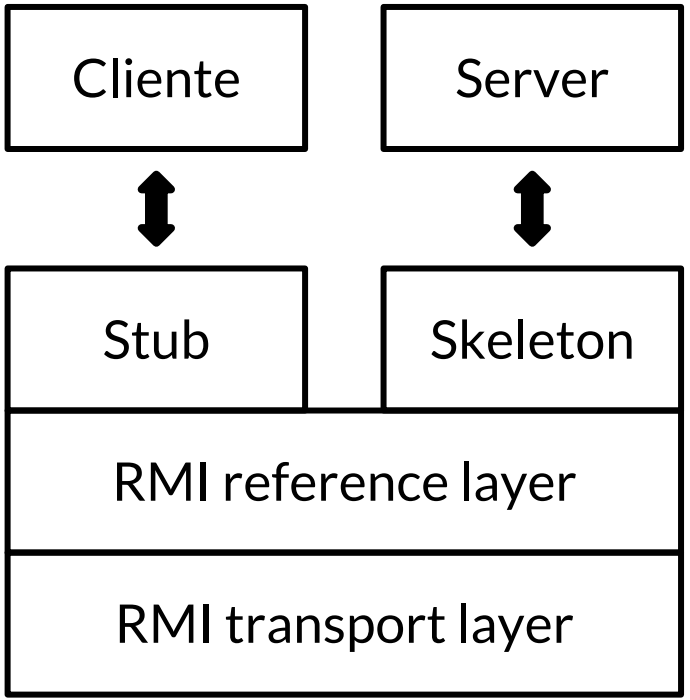
El cliente tiene que ser Java.

Si se desarrolla el front-end en otro lenguaje de programación, habrá inconvenientes.

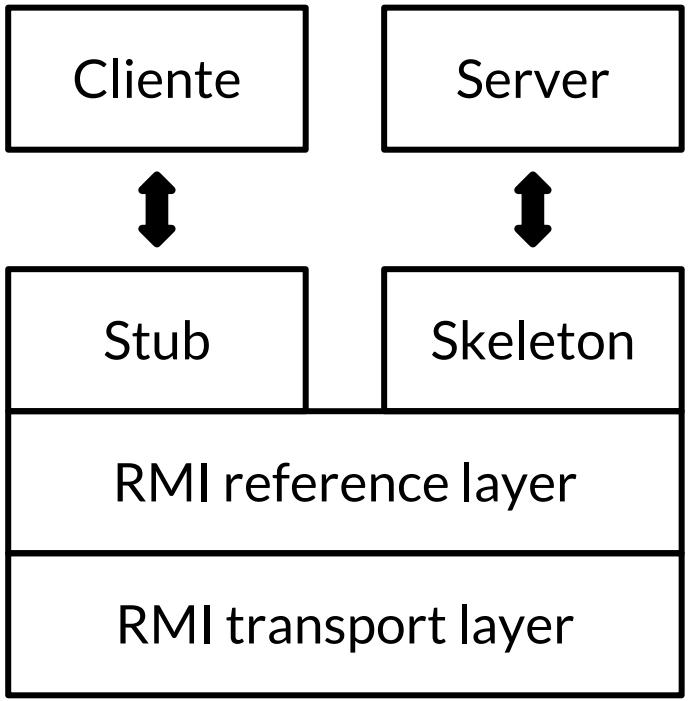
4. Arquitectura

4. Arquitectura

La arquitectura RMI puede verse como un modelo de cuatro capas.



4. Arquitectura



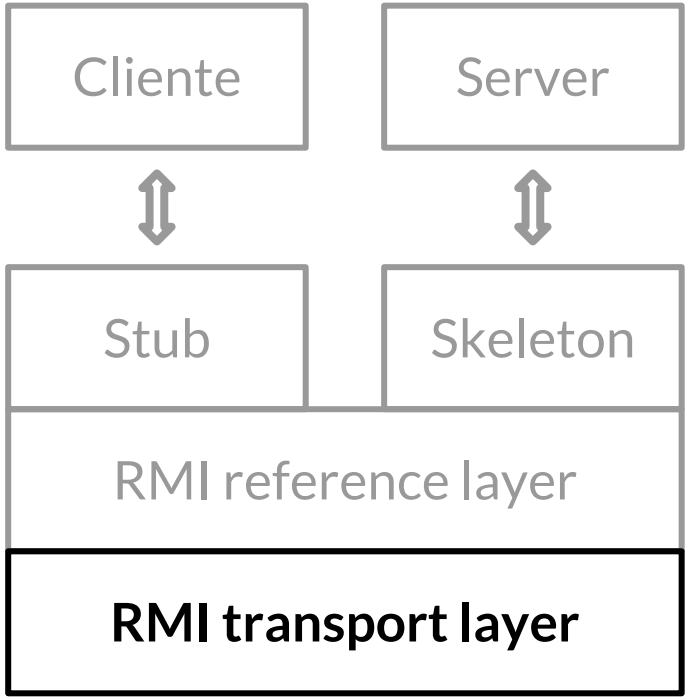
Capa es la de aplicación

Capa proxy, o capa stub-skeleton

Capa de referencia

Capa de transporte

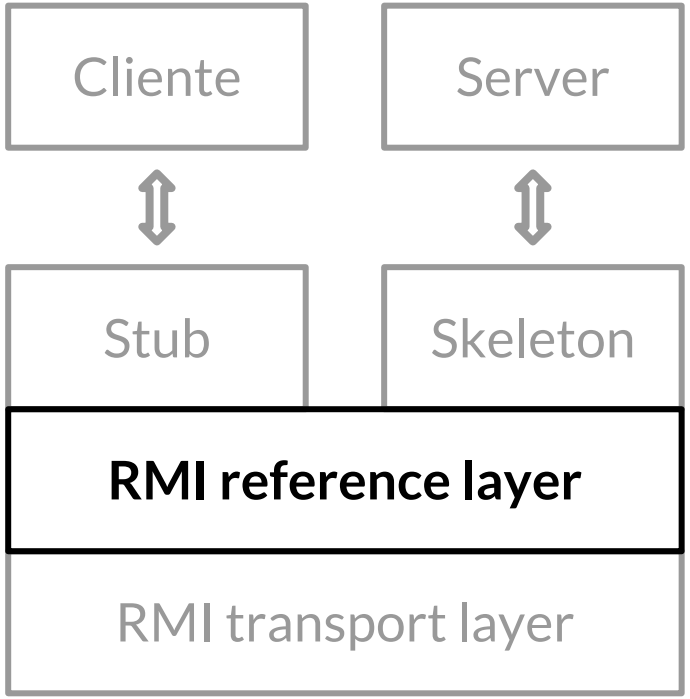
Capa de transporte



Realiza las **conexiones necesarias y manejo del transporte** de los datos de una máquina a otra.

El protocolo de transporte para RMI es JRMP (Java Remote Method Protocol), **solo para programas Java**.

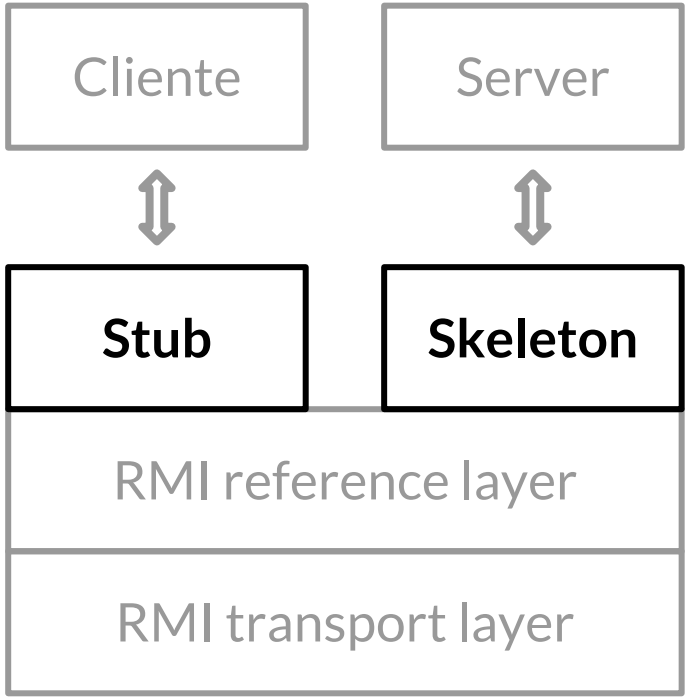
Capa de referencia



Se encarga de la **creación y gestión** de las referencias a objetos remotos,

Convierte las llamadas remotas en **peticiones** hacia la capa de transporte.

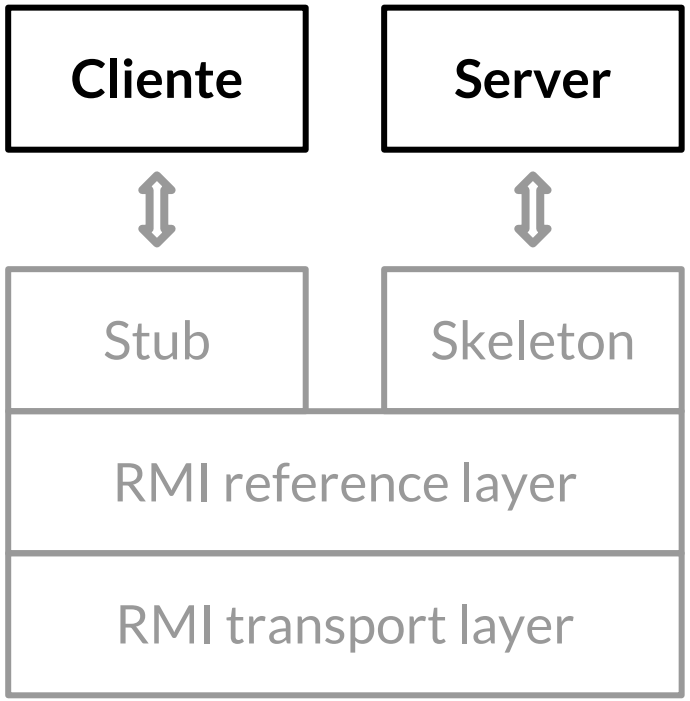
Capa proxy, o capa stub-skeleton



Permite localizar objetos remotos para invocar sus métodos como si fueran locales.

El Stub y el Skeleton permiten la comunicación con el objeto remoto

Capa de aplicación



Corresponde con la implementación real de las aplicaciones cliente y servidor.

HelloClient.java

HelloImpl.java

5. Funcionamiento

La clase java.rmi.Naming

El URL se presenta en la forma rmi://host:port/objectName, donde:

Componente	Valor por defecto	Especificación
`rmi`	`rmi`	El método de acceso (debe ser `rmi`).
`host`	`localhost`	La máquina servidor (`host`).
`port`	1099	El puerto utilizado.
`objectName`	-	El nombre del objeto remoto.

5. Funcionamiento

Cuando la aplicación cliente envía un mensaje al stub local del objeto remoto, la petición se transmite a la máquina que contiene al objeto real, donde el método es invocado y cualquier resultado retornado al stub local, de modo que la aplicación cliente puede obtener la respuesta apropiada.

Enviando mensaje a un objeto remoto

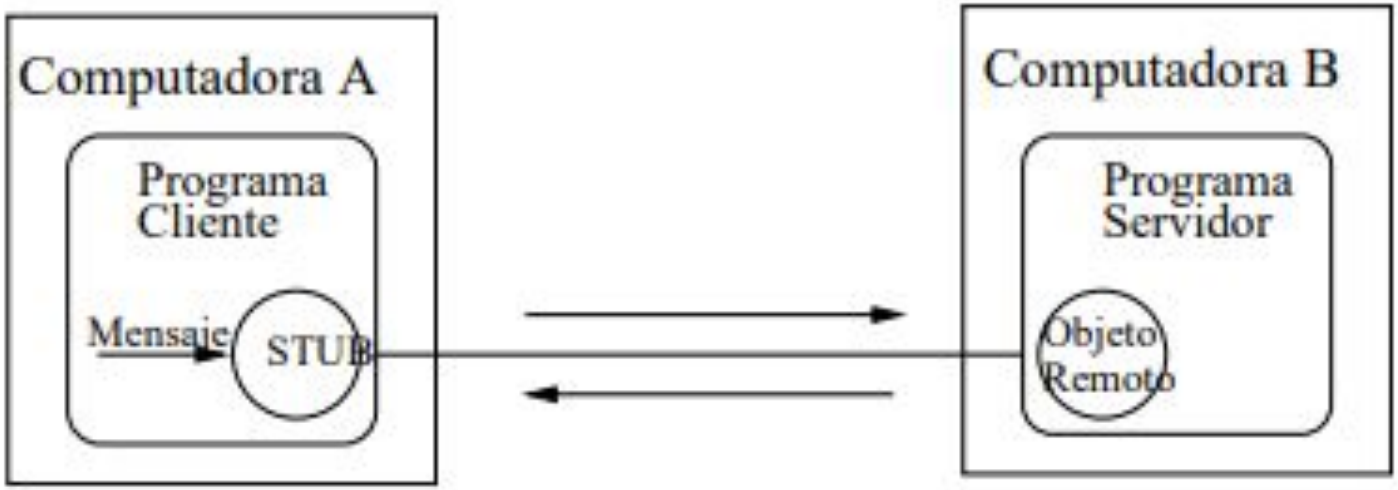


Figura 1: Enviando un mensaje a un objeto remoto.

La clase java.rmi.Naming

Método	Responsabilidad
`bind(url,object)`	Liga un nombre a un objeto remoto. El nombre se especifica como un URL.
`list(url)`	Retorna un arreglo de cadenas representando los URLs en el registro.
`lookup(url)`	Retorna un objeto remoto (un *stub*) asociado con el URL.
`rebind(url, object)`	Similar al `bind()`, pero reemplaza la asociación hecha.
`unbind(url)`	Remueve la asociación entre el objeto remoto y el URL.

La aplicación rmiregistry

La aplicación servidor rmiregistry se utiliza para:

- Registrar un nombre y un lugar de un objeto remoto. Esto se realiza a partir de un servidor que contiene un objeto remoto.

El código en el servidor es de la forma:

```
Naming.rebind("rmi://host/name", object);
```

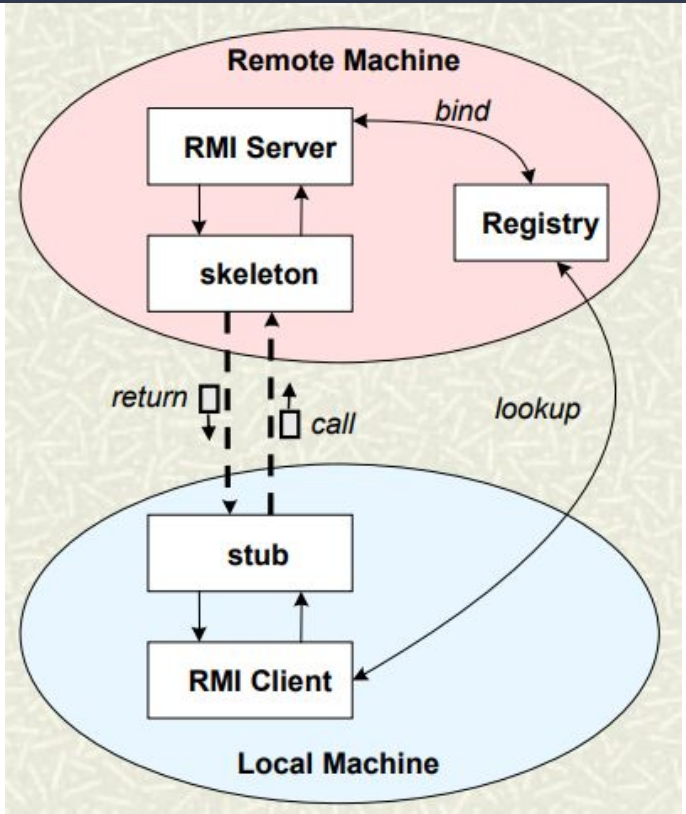
Método LOOKUP

Permitir a un cliente ligar un stub local que le de acceso al objeto remoto contenido en la aplicación servidor. La aplicación cliente se liga al objeto remoto mediante el método lookup(), que retorna un objeto que permite el acceso vía un stub al objeto remoto. El código en la aplicación cliente debe ser de la forma:

```
Naming.lookup("rmi://host/name");
```

En esencia, la aplicación rmiregistry actua como un registro de objetos que pueden ser accedidos remotamente. Los objetos se registran y ligan a un stub local usando los métodos de la clase Naming, del paquete java.rmi.

En Conclusión Tenemos lo Siguiente



6. Ejemplos

Referencias

- https://www.ctr.unican.es/asignaturas/procodis_3_II/Doc/Procodis_7_01.pdf
- http://lya.fciencias.unam.mx/jloa/rmi.pdf
- https://es.wikipedia.org/wiki/Java_Remote_Method_Invocation