MONDRAGON UNIBERTSITATEA

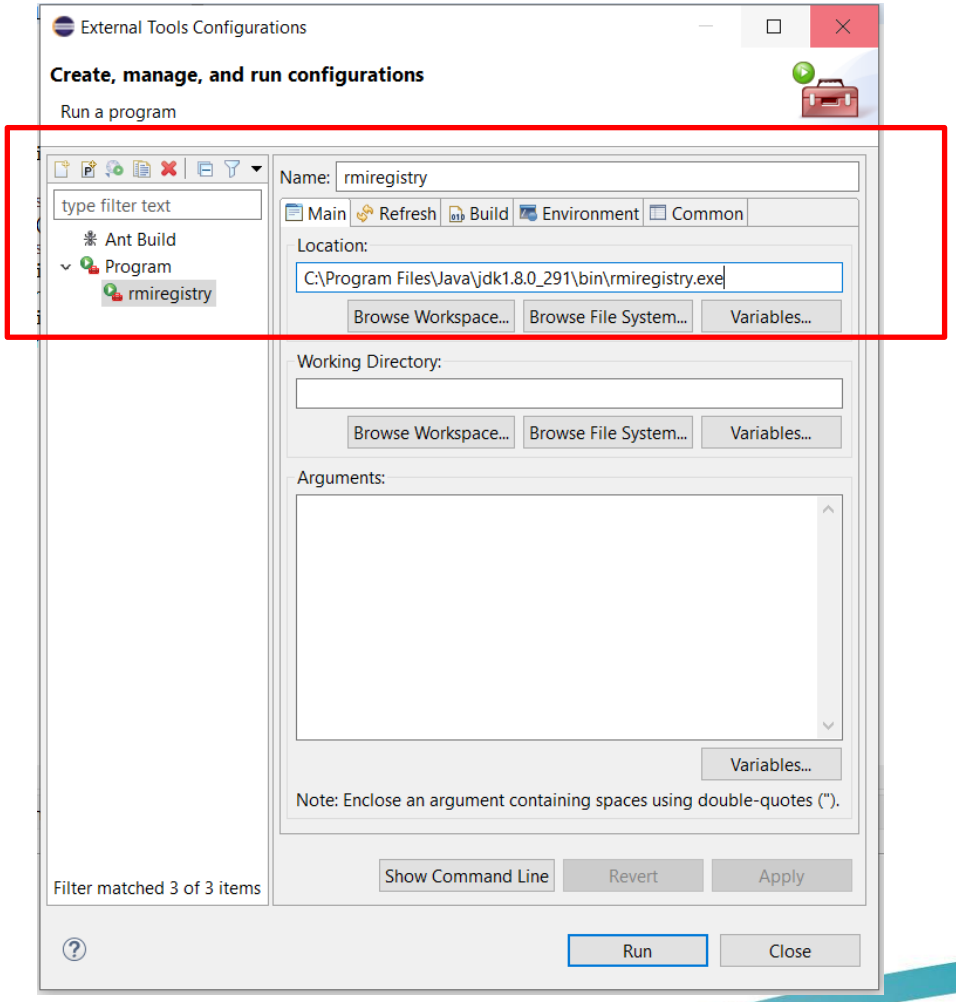
GOI ESKOLA POLITEKNIKOA

ESCUELA POLITÉCNICA SUPERIOR

Sistemas concurrentes y distribuidos
Java Remote Method Invocation (Java RMI)

Setup de Java RMI en eclipse

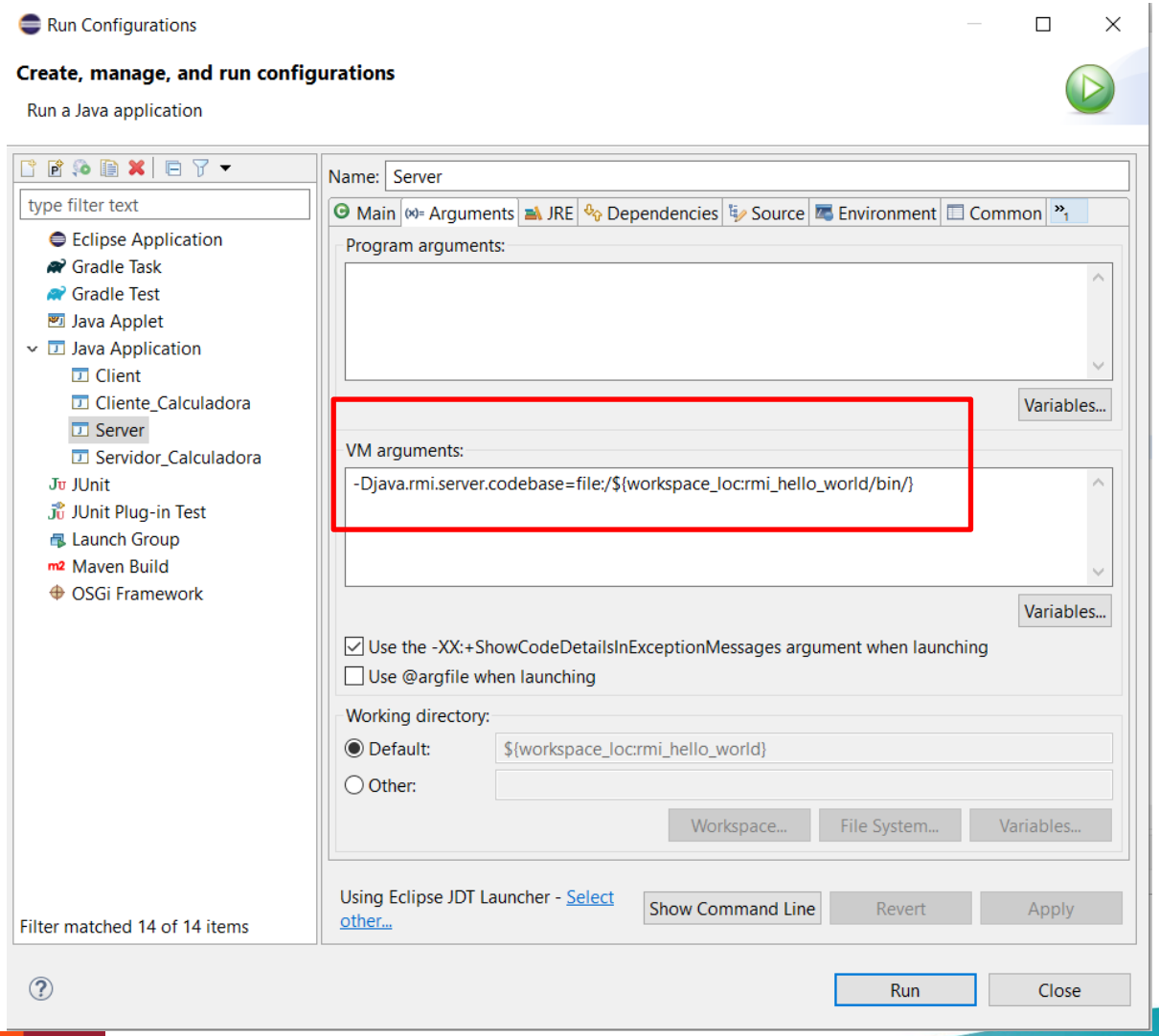
- Link de ayuda para el setup.
- Configurar el **rmiregistry** como external tool en eclipse.



Setup de Java RMI en eclipse

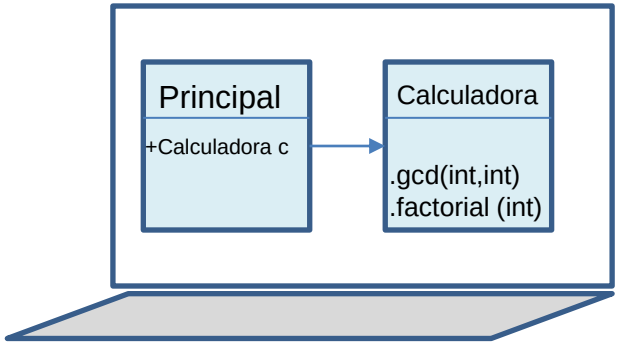
En caso de que se necesite hacer la configuracion en remoto.

- Añadir argumentos al “run configuration” del servidor:

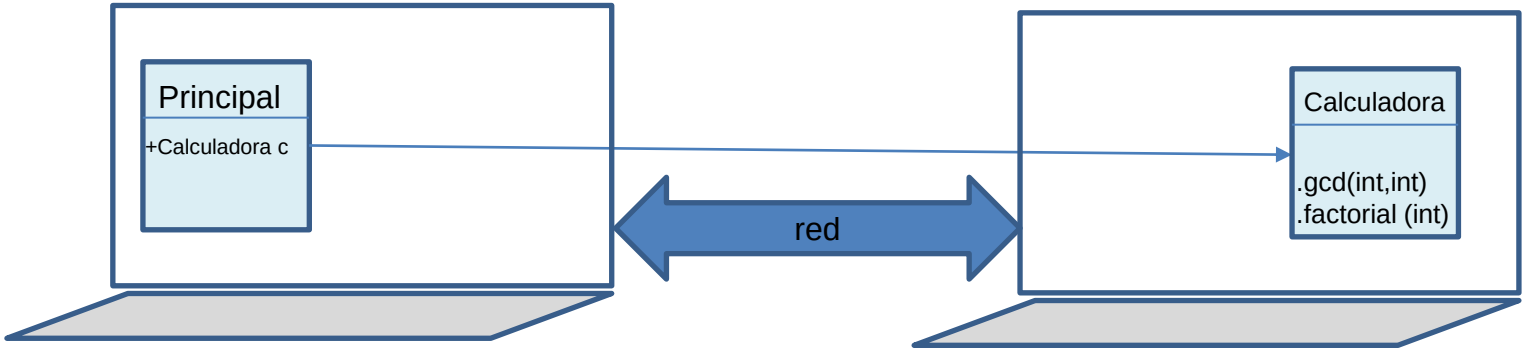


Introducción

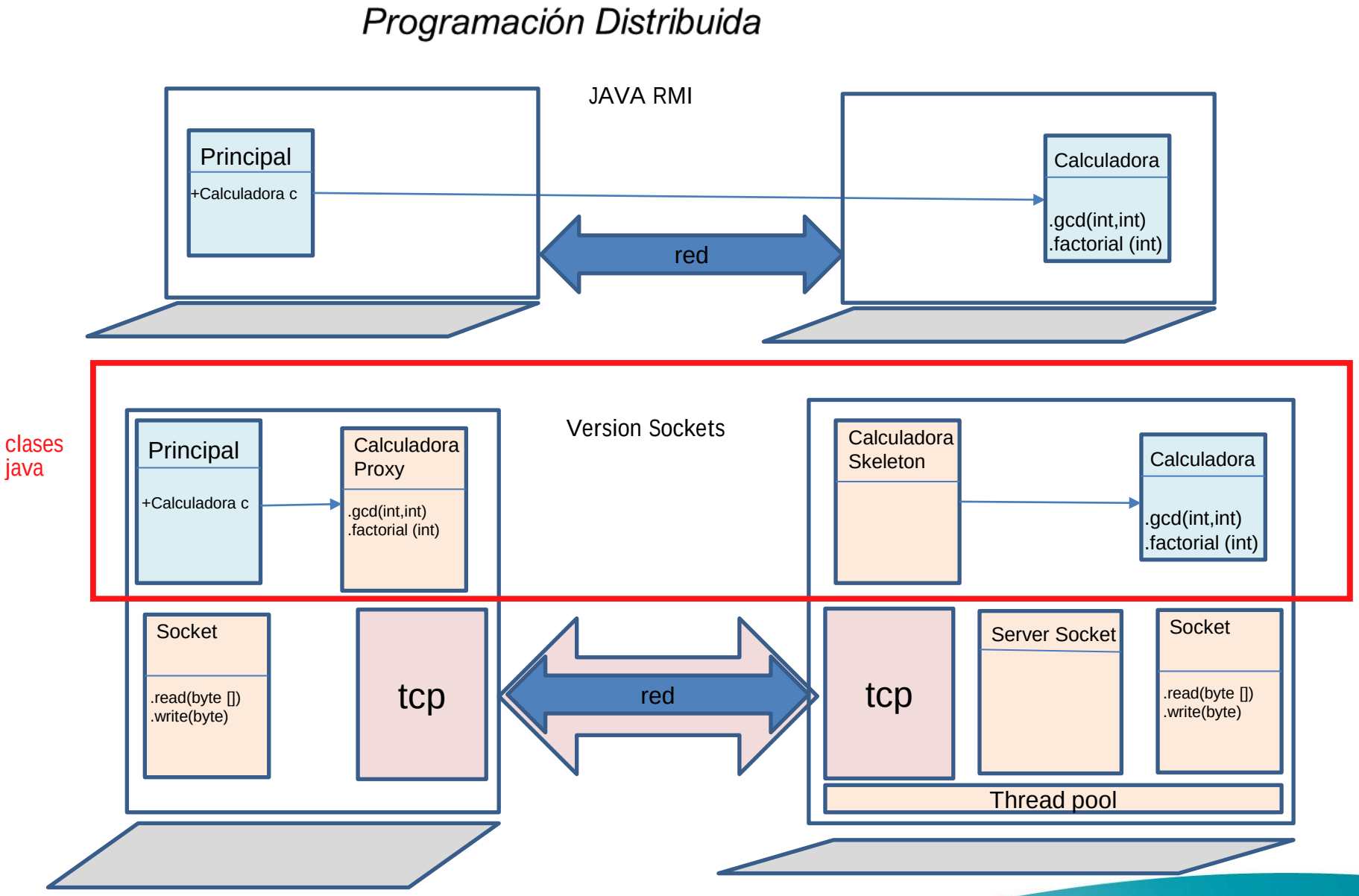
Programación Distribuida



Lo que queremos es invocar el metodo que esta en otro ordenador.

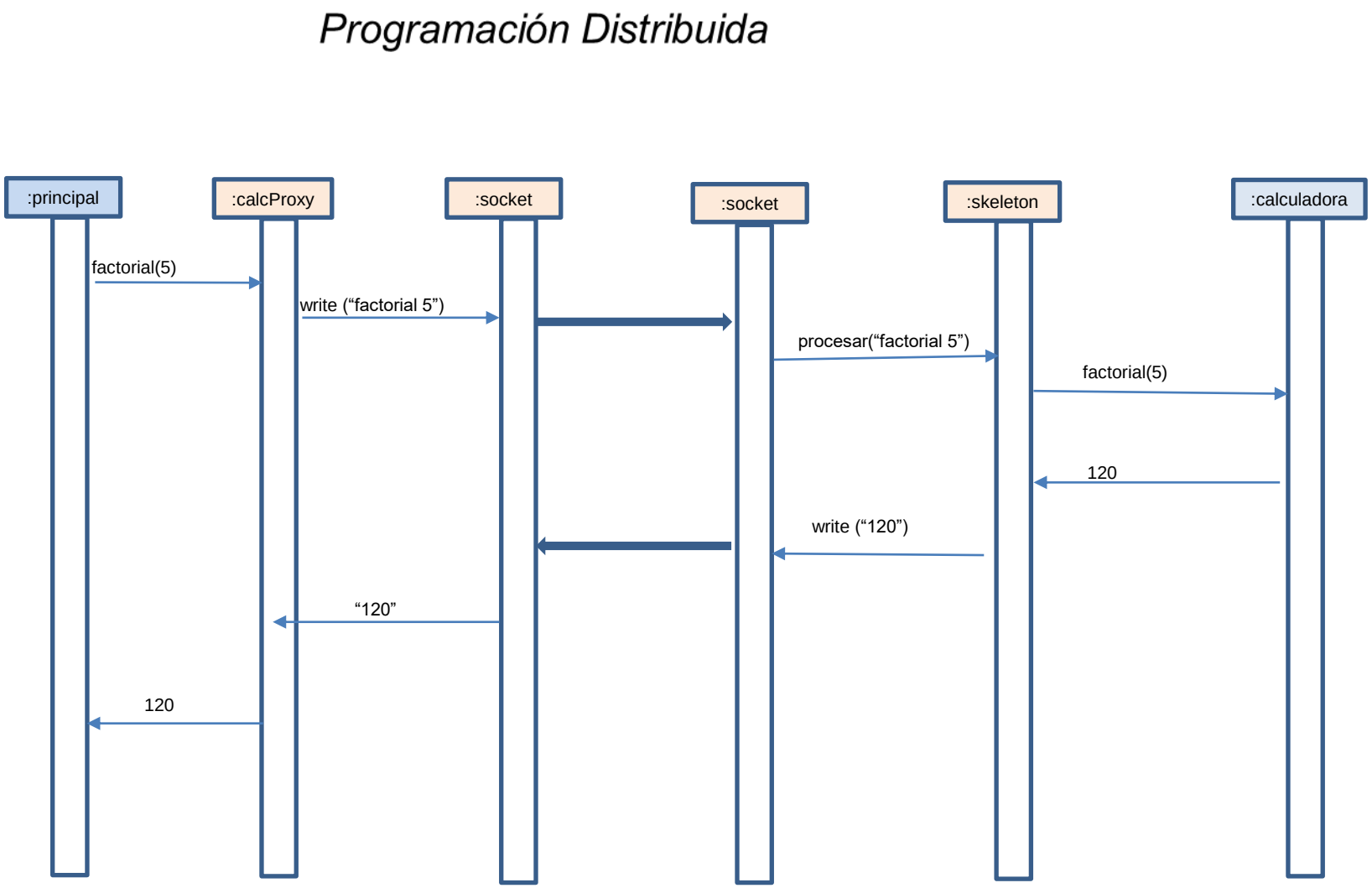


Introducción



-> JavaRMI simpliifica enormemente las comunicaciones.

Introducción



¿Qué es Java RMI?

- Middleware de comunicación orientado a objetos que proporciona una solución para el lenguaje JAVA.
  - No es multilenguaje, pero es multiplataforma

  Es una solucion exclusiva de java

- Es una API que proporciona un mecanismo para crear aplicaciones distribuidas en JAVA.

- Permite a un objeto invocar métodos en un objeto que se está ejecutando en otro JVM (Java Virtual Machine), y pasar objetos Java como argumentos cuando se invocan dichos métodos.

- Proporciona una comunicación remota entre dos aplicaciones utilizando los objetos **stub** y **skeleton**.

  -> Stub: Es el equivalente del proxy, es la comunicacion de parte del cliente

  -> Skeleton: Es lo mismo que en los sockets, es la comunicacion de parte del cliente.

El servidor de nombres de Java RMI

-> La transferencia se hace a traves del servidor de nombres.

- Almacena para objeto: nombre simbólico + referencia
- Puede residir en cualquier nodo y es accedido desde el cliente o el servidor usando la interfaz local ***Registry***
- En las distribuciones Oracle de Java, el servidor de nombre se lanza usando el comando ***rmiregistry***.

Remote lookup (String name)
->

void bind (String name, Remote obj)

Void rebind (String name, Remote obj)

Void unbind (String name)

Registry -> es la interfaz que encapsula los objetos para la transferencia. Y permite transmitirlo

El servidor de nombres de Java RMI

- RMI Registry es un servicio proporcionado por Java RMI (Remote Method Invocation) que actúa como un directorio de objetos remotos.

- Los objetos remotos se registran en el RMI Registry, que actúa como un punto de entrada para que los clientes puedan localizar y acceder a ellos.

- El RMI Registry es un proceso que se ejecuta en la máquina virtual de Java y escucha en un número de puerto específico para las solicitudes de los clientes. -> es 1099 pero se puede cambiar

- Cuando un objeto remoto se registra en el RMI Registry, se le asigna un identificador único de objeto remoto que se puede utilizar para acceder a él desde cualquier máquina virtual de Java en la red.

Paso de objetos como argumentos

- Cuando se invoca un método remoto, se pueden pasar objetos como argumentos
  - Si el objeto que se pasa como argumento implementa la interfaz Remote
    - Se pasa por referencia, se comparte
  - Si el objeto que se pasa como argumento **NO** implementa la interfaz Remote
    - Se serializa y se pasa por valor
    - Se crea un objeto en la máquina virtual destino totalmente independiente al original

Características de la comunicación

Utilización

- Las llamadas a métodos remotos abstraen al programador de la capa de comunicación.

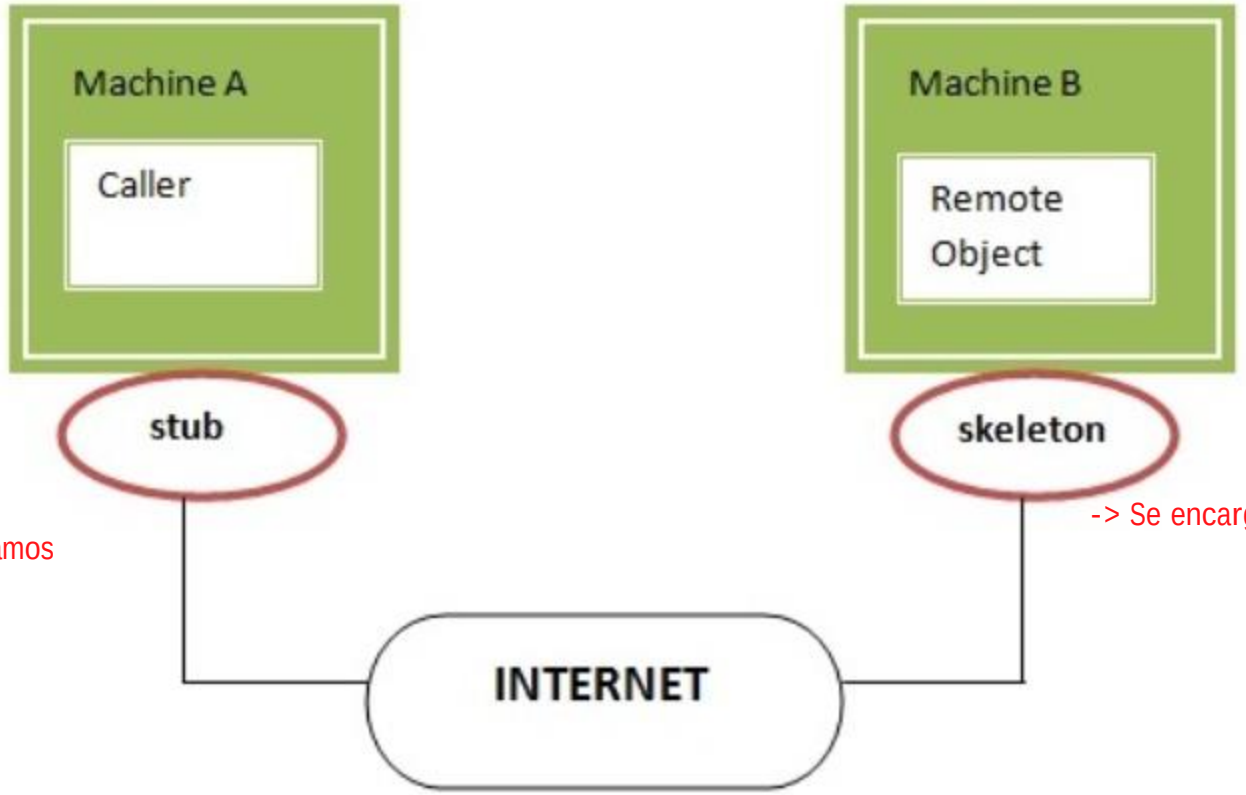
Estructura y contenido de los mensajes

- Determinados por el compilador, transparentes al programador.

Sincronización

- Comunicación síncrona entre cliente-servidor.

Stub and Skeleton



-> Stub gestiona los objetos que tratamos

-> Se encarga de las comunicaciones.

Stub

- El objeto Stub actúa como puerta de enlace (gateway) en el lado del cliente.
- Todas las solicitudes salientes se enrutan a través de él.
- Reside en el lado del cliente y representa el objeto remoto.
- Cuando el cliente invoca un método en el objeto stub, éste realiza las siguientes tareas:
  - Inicia una conexión con la máquina virtual remota (JVM).
  - Escribe y transmite los parámetros a la Máquina Virtual remota (JVM).
  - Espera el resultado.
  - Lee (desmarca) el valor de retorno o la excepción.
  - Finalmente, devuelve el valor al cliente que lo llamó.

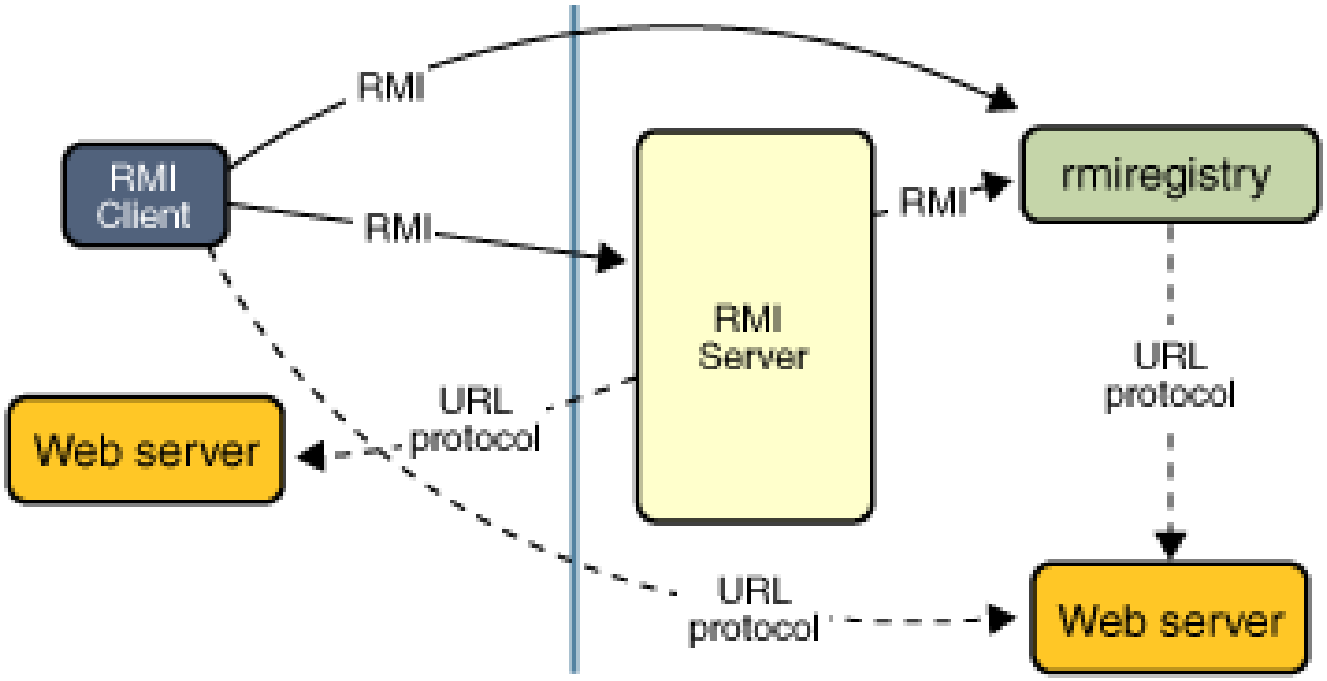
Skeleton

- Skeleton es un objeto que actúa como entrada en el objeto del lado del servidor.

- Todas las peticiones pasan por él.

- Cuando recibe una petición, actúa de la siguiente manera:
  - Lee el parámetro para el método remoto
  - Invoca el método en el objeto remoto actual
  - Escribe y transmite el resultado al cliente que lo invocó.

Aplicación distribuida RMI



-> Es un diagrama de carga dinamica en el servidor

- te sirve para cargar la clase que le falta al servidor y esta en el cliente. Con esto se puede descargar las clases faltantes del cliente al servidor usando Servicios web (que van por debajo).

Aplicación distribuida RMI

- La ilustración muestra una aplicación distribuida RMI que utiliza el registro RMI para obtener una referencia a un objeto remoto.
- El servidor llama al registro para asociar (o enlazar) un nombre con un objeto remoto.
- El cliente busca el objeto remoto por su nombre en el registro del servidor y luego invoca un método sobre él.
- El sistema RMI utiliza un servidor web existente para cargar las definiciones de clase, del servidor al cliente y del cliente al servidor, para los objetos cuando sea necesario.

Interfaces, objetos y métodos remotos

- Una aplicación RMI, como cualquier otra aplicación en JAVA, está formada por interfaces y clases.
  - Las interfaces declaran métodos.
  - Las clases implementan los métodos.

- En una aplicación distribuida algunas implementaciones residen en otra JVM.
  - Objeto remoto: Objeto con métodos que pueden ser invocados a través de JVMs.

- Para que un objeto sea remoto es necesario que implemente la interfaz **Remote**
  - La interfaz remote extiende la interfaz java.rmi.Remote
  - Cada método de la interfaz debe lanzar una exception de tipo java.rmi.RemoteException

Diseñar e implementar los componentes de la aplicación

- Definir las interfaz -> Hace falta una interfaz comun para que se puedan comunicar
  - La interfaz define los métodos que se podrán invocar remotamente por el cliente
  - Los clientes hacen referencia a las interfaces, no a las clases que las implementan.

- Implementar los objetos remotos
  - Los objetos remotos deben implementar una o más interfaces
  - La clase del objeto remoto también puede implementar otras interfaces y métodos que solo se usan localmente.

- Implementar los clientes
  - Los clientes que usan los objetos remotos pueden ser implementados en cualquier momento después de definir las interfaces, incluido después de haber lanzado los objetos remotos.

Ejemplo Hello World

Un vistazo al código del hello world…

Ejercicio: Calculadora RMI

- Implementa una calculadora distribuida utilizando RMI.
- El cliente pedirá al usuario introducir una de las siguientes opciones y los dos números a realizar la operación indicada:

  1 – Sumar

  2 – Restar

  3 – Multiplicar

  4 – Dividir
- A continuación ejecutará remotamente en el servidor el método correspondiente y obtendrá el resultado de la operación.

Carga dinámica de código

- Una de las características principales de RMI es su capacidad para descargar la definición de la clase de un objeto si la clase no está definida en la máquina virtual Java del receptor.

- Todos los tipos y comportamientos de un objeto, que antes sólo estaban disponibles en una única máquina virtual Java, pueden transmitirse a otra máquina virtual Java, posiblemente remota.

- Esta capacidad permite introducir nuevos tipos y comportamientos en una máquina virtual Java remota, ampliando así dinámicamente el comportamiento de una aplicación.

Carga dinámica de código: Ejemplo Compute Engine

- Objeto remoto en el servidor que recibe tareas de los clientes, las ejecuta y devuelve los resultados.

- Las tareas se ejecutan en la máquina donde se ejecuta el servidor, en una máquina más potente que los clientes.

- Lo novedoso es que las tareas que ejecuta no necesitan definirse cuando se escribe o se inicia el Compute Engine. Se pueden crear nuevos tipos de tareas en cualquier momento y dárselas al motor de cálculo para que las ejecute.

- El único requisito de una tarea es que su clase implemente una interfaz determinada. El código necesario para realizar la tarea puede ser descargado por el sistema RMI al Compute Engine.

- RMI carga dinámicamente el código de la tarea en la máquina virtual Java del motor de computación y ejecuta la tarea sin conocimiento previo de la clase que implementa la tarea.

- Una aplicación de este tipo, que tiene la capacidad de descargar código dinámicamente, suele denominarse aplicación basada en el comportamiento (**behavior-based application**).

Ejemplo Compute Engine

Un vistazo al código del compute engine…

Eskerrik asko