Invocación a Métodos Remotos

Virginia Padilla

Universidad Nacional Experimental de Guayana

virginiapadillas@gmail.com

28 de junio de 2021

Contenido

1. RMI
   - Invocación a Métodos Remotos - RMI
     - Invocación de Objetos
     - Instanciación de Objetos
     - Arquitectura RMI
     - Colector de Basura Distribuida

RMI

RMI

Los puntos en común entre RMI y RPC son los siguientes:

- Ambos admiten programación con interfaces.
- Están construidos sobre protocolos de solicitud-respuesta y pueden ofrecer un rango de semántica de llamadas como al menos una vez y como máximo una vez.
- Ambos ofrecen un nivel similar de transparencia, es decir, las llamadas locales y remotas emplean la misma sintaxis, pero las interfaces remotas suelen exponer la distribución de la llamada subyacente, por ejemplo, admitiendo excepciones remotas.

RMI

RMI

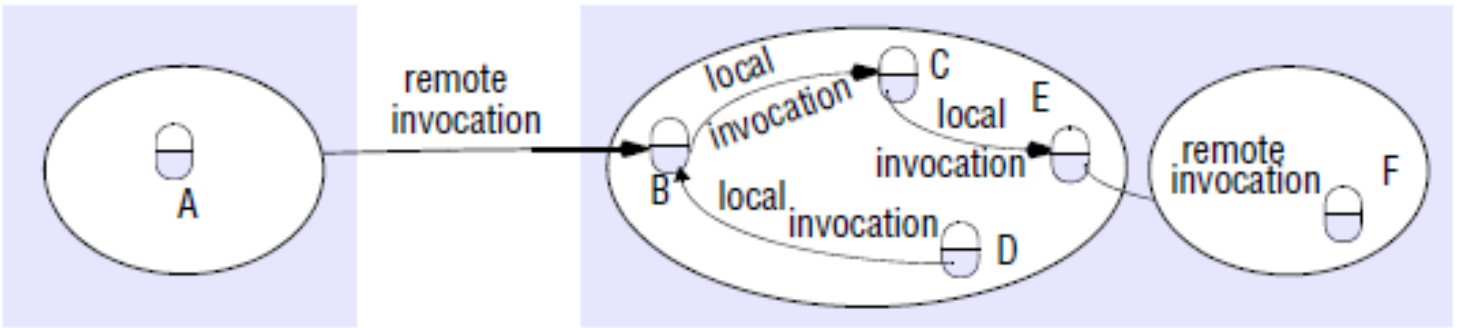
Las siguientes diferencias conducen a una mayor expresividad cuando se trata de programación de aplicaciones y servicios distribuidos complejos:

- El programador puede utilizar todo el poder expresivo de la programación orientada a objetos para el desarrollo de software de sistemas distribuidos, incluido el uso de objetos, clases y herencia,metodologías de diseño y herramientas asociadas.
- Partiendo del concepto de identidad de objeto en sistemas orientados a objetos, todos los objetos en un sistema basado en RMI tienen referencias de objeto únicas.
- Estas referencias de objeto también se pueden pasar como parámetros, ofreciendo así semántica de paso de parámetros significativamente más rica en RPC.

Invocación de Objetos

Invocación de Objetos

- En la figura se hace referencia a los objetos que pueden recibir invocaciones remotas como objetos remotos.
- Los objetos **B** y **F** son objetos remotos. Todos los objetos pueden recibir invocaciones locales, aunque solo pueden recibirlas de otros objetos que contienen referencias a ellos.
- Por ejemplo, el objeto **C** debe tener una referencia al objeto **E** para que pueda invocar uno de sus métodos.



Instanciación de Objetos

Instanciación de Objetos

Los siguientes dos conceptos fundamentales están en el corazón del modelo de objeto distribuido:

- Referencias a objetos remotos: otros objetos pueden invocar los métodos de un objeto remoto si tienen acceso a su referencia de objeto remoto. Por ejemplo, un objeto remoto la referencia para B en la Figura debe estar disponible para A.
- Interfaces remotas: cada objeto remoto tiene una interfaz remota que especifica qué de sus métodos se pueden invocar de forma remota. Por ejemplo, los objetos B y F en la Figura deben tener interfaces remotas.

Instanciación de Objetos

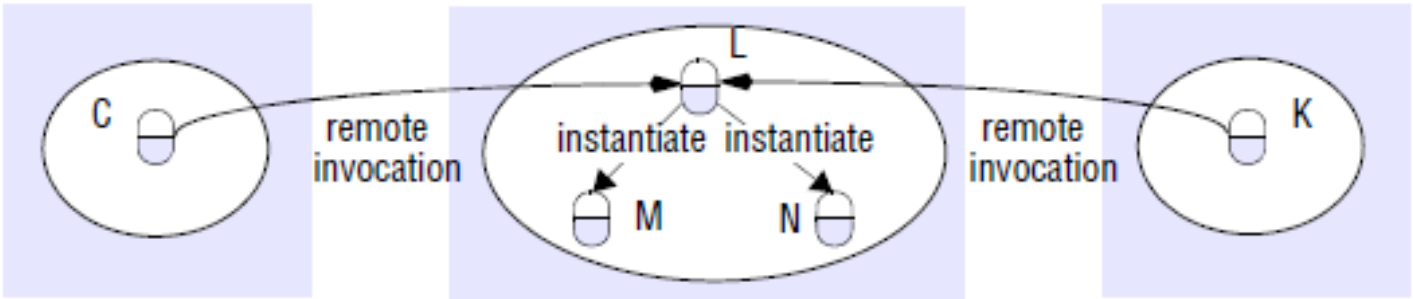
Instanciación de Objetos

- Cuando una acción conduce a la instanciación de un nuevo objeto, ese objeto normalmente debe estar dentro del proceso donde se solicita la instanciación, por ejemplo, donde se utilizó el constructor.
- Si el objeto recién instanciado tiene una interfaz remota, será un objeto remoto con una referencia de objeto remoto.
- Las aplicaciones distribuidas pueden proporcionar objetos remotos con métodos para crear instancias de objetos a los que puede acceder RMI, proporcionando así el efecto de manera efectiva de instanciación remota de objetos.

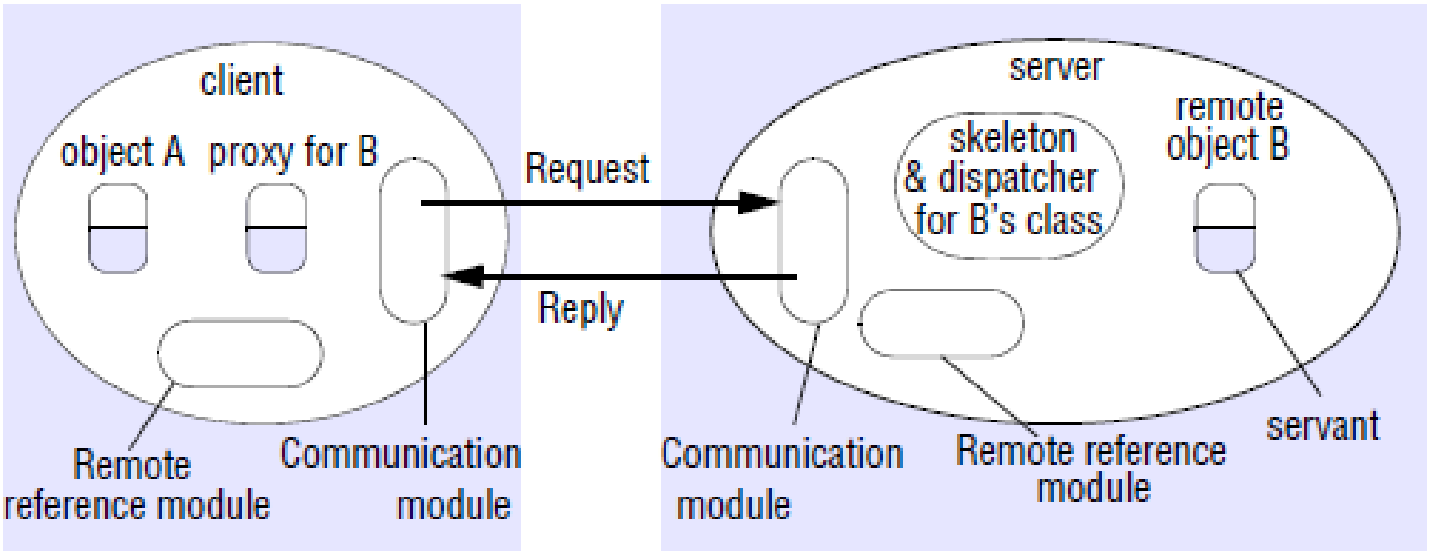
Instanciación de Objetos

Instanciación de Objetos

- Si el objeto L en la figura contiene un método para crear objetos remotos, entonces las invocaciones remotas de C y K podrían conducir a la instanciación de los objetos M y N, respectivamente.



Arquitectura RMI



Arquitectura RMI

Arquitectura RMI

Módulo de Comunicación:

- Proporcionan semántica de invocación, como ejemplo, al-menos-uno
- Transmite mensaje de solicitud/respuesta entre cliente y servidor
- Solicitud es (tipo de mensaje, IdSolicitud, ref objeto remoto, IdOperacion, Argumentos)
- En el servidor selecciona el despachador para la clase de objeto que se invoca.
- El despachador ubica la referencia local del objeto en el Módulo de Referencia Remota

Arquitectura RMI

Arquitectura RMI

Módulo de Referencia Remota:

- Responsable de trasladar referencias de objetos locales a remotas y creación de referencias remotas.
- Contiene una tabla de objetos remotos y una tabla para cada proxy local
- Actúa de la manera siguiente:
  - 1era vez cuando se pasa un objeto remoto, el módulo de referencia remota crea una referencia al objeto remoto y se añade a la tabla de objetos remotos
  - Cuando llega referencia a un objeto remoto, el módulo de referencia obtiene referencia al objeto local, la cual es un proxy o un objeto remoto. Si el objeto remoto no está en la tabla se crea el proxy y se añade el módulo de referencia remota

Arquitectura RMI

Arquitectura RMI

Criado (Servant)

- Instancia de una clase que proporciona el cuerpo de un objeto remoto.
- Maneja el requerimiento remoto pasado por el esqueleto correspondiente.
- Se crea cuando se instancia el objeto remoto.

Proxy

- Proporciona que los métodos de invocación remota sean transparente al usuario comportandose como un objeto local.
- Esconde los detalles del empaquetamientos-desempaquetamiento de las referencias a objetos remotos.

Arquitectura RMI

Arquitectura RMI

Despachador

- Un servidor tiene un despachador y un esqueleto.
- Recibe la petición desde el módulo de comunicación.
- Usa el IdOperacion para seleccionar el método adecuado en el esqueleto.

Esqueleto

- Implementa el método en la interface remota.
- Desempaqueta los argumentos e invoca el método correspondiente en el criado.
- Espera que la invocación se complete y empaqueta los resultados

Colector de basura distribuida:

Colector de basura distribuida:

- Algoritmo distribuido cuya función es recolectar la basura distribuida.
- Eestablece una cooperación entre el colector local y un módulo añadido que colecciona la basura distribuida

Colector de basura distribuida:

- Cada proceso servidor mantiene un conjunto de nombres de los procesos que proporcionan referencias a objetos remotos por cada uno de sus objetos remotos. Este conjunto se puede almacenan en una columna adicional de la tabla de objetos remotos.
- Cuando un cliente C recibe una referencia remota a un objeto remoto en particular, B, hace una invocación AddRef (B) al servidor de ese objeto remoto y luego crea un proxy; el servidor agrega un apuntador C a B.
- Cuando recolector de basura de un cliente C nota que un proxy de un objeto remoto B no está accesible, hace una invocación removeRef (B) al servidor correspondiente y luego borra el proxy; el servidor elimina apuntador C de B.
- Cuando apuntador B está vacío, el recolector de basura local del servidor recuperará el espacio ocupado por B a menos que existan apuntadores locales.

References

Van Steen M , Tanenbaum, A (2017)

Distributed Systems

Pearson Education, Inc.

Coulouris G, Dollimore J, Kindberg T, Blair G (2011)

Distributed Systems: Concepts and Design

Addison-Wesley Publishing Company.

Veríssimo, P. and Rodrigues, L. (2012)

Distributed Systems for System Architects

Springer.

Limoncelli T, Strata R, Hogan C. (2014)

The Practice of Cloud System Administration: Designing and Operating Large Distributed Systems

Addison Wesley.

Fin