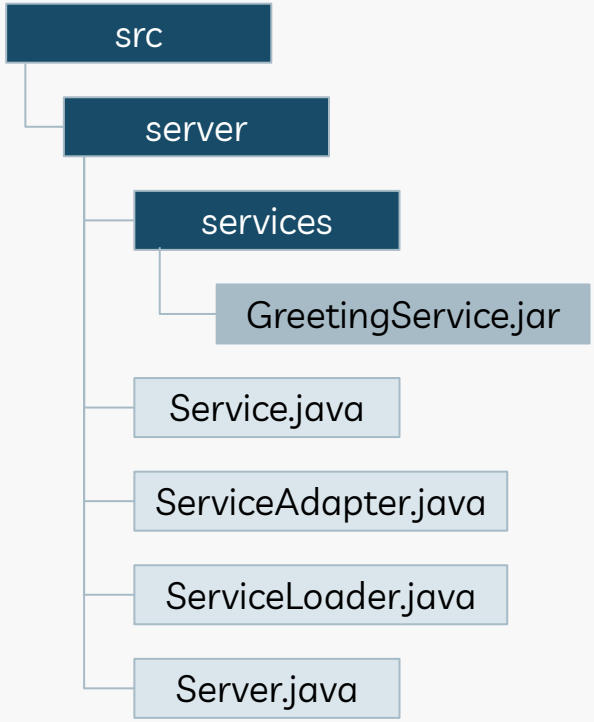
Servidor



Directorio services
Contiene los servicios que el servidor puede ejecutar, empaquetados en archivos .jar.

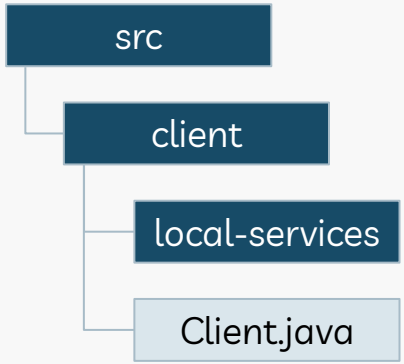
Service.java
Es la interfaz común que el servidor usa para ejecutar servicios.

ServiceAdapter.java
Adapta nuevos servicios a la interfaz Service.java.

Server.java
Maneja la comunicación y ejecución de los servicios cargados dinámicamente.

ServiceLoader.java
Encuentra y carga los servicios disponibles para el servidor.

Cliente

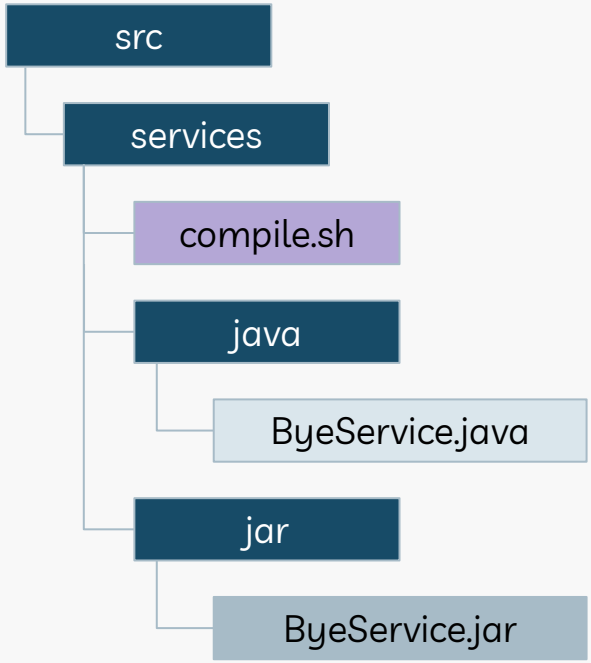


Directorio local-services
Almacena los archivos .jar que el cliente puede enviar al servidor.

Client.java
Presenta un menú en consola para que el usuario:

- Ejecute servicios en el servidor.
- Envíe un nuevo servicio al servidor.

Servicicios



Script compile.sh

1. Compila los archivos .java del directorio java.
2. Añade metadatos con el nombre completo de las clases (incluyendo el package).
3. Los empaqueta en un .jar en el directorio jar.

Directorio java

Contiene la interfaz Service.java.
Puede incluir nuevos servicios que se quieran implementar.

Directorio jar

Almacena los servicios ya empaquetados y listos para ser ejecutados por el servidor.

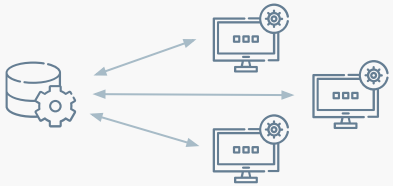
Comunicación Cliente-Servidor

Actual:

- Cliente y servidor se comunican mediante sockets en un modelo 1 a 1.
- El servidor puede atender solo un cliente a la vez.

Mejora Propuesta - Hilos (Threads):

- Permitir que el servidor maneje múltiples clientes simultáneamente.
- Es el comportamiento típico en sistemas cliente-servidor modernos.



Aviso Importante

El cliente tiene la capacidad de enviar nuevos servicios al servidor para su ejecución. Sin embargo:

- **No se realiza ninguna comprobación de seguridad** durante la carga o ejecución de estos servicios.
- Este enfoque **no es seguro ni adecuado para un entorno real**.
- Este ejemplo es únicamente **educativo** y busca ilustrar la arquitectura cliente-servidor.