Sistemas Distribuidos

Práctica JavaDSM

Coherencia de entrada (EC)

```
int main() {
    shared tipo1 v1; shared tipo2 v2; shared tipo3 v3;…
    cerrojo c1, c2;
    ……..
    asociar(c1, v1);
    asociar(c1, v2);
    asociar(c2, v4);
    ……..
    adquirir(c1);
    ……..
    v1.xx...
    v2.yy...
    ……..
    liberar(c1);
    ……..}
```

Objetivo

- Diseño simplificado de sistema RT-DSM con EC para Java
  - Tecnología Java RMI
- En principio aplicable a todo tipo de objetos
  - Finalmente sólo para objetos tipo array de “cualquier” tipo (tipo [] v)
    - Ese nivel de indirección simplifica desarrollo
- Gestor centralizado G para control sec. crítica: ServidorDSM
  - Cerrojos con modo exclusivo y compartido
- Problema al implementar EC:
  - ¿Y si proceso libera SC y termina antes de que otro lo adquiera?
- Gestor G también almacén de objetos
  - Caché que guarda número de versión de cada objeto
- Tanto cerrojos como objetos compartidos tienen nombre

Ejemplo de JavaDSM

```
import dsm.*;
public class ClienteDSM {
    static public void main (String args[]) {
        StringBuffer [] v1 = new StringBuffer[1]; v1[0] = new StringBuffer();
        ObjetoCompartido o1 = new ObjetoCompartido("objecto1", v1);
        DSMCerrojo cerrojo =  new DSMCerrojo("cerrojo");
        cerrojo.asociar(o1);
        cerrojo.adquirir(true);
        System.out.println("Valor de objeto al entrar: " + v1[0]);
        v1[0].append("|hola");
        cerrojo.liberar();
        cerrojo.desasociar(o1);
    }
}
```

Esquema EC atípico

- Cliente C entra en SC (compartida|exclusiva) de cerrojo Ci
  - Le pide a G cada objeto asociado a Ci enviando su nº versión local
  - G retorna contenido de objetos con mayor número de versión
  - C actualiza dichos objetos
- Cliente C sale de SC exclusiva de cerrojo Ci
  - Incrementa nº versión de objetos asociados a Ci
  - Envía esos objetos a G
  - NOTA: por simplicidad no control de modificaciones
    - se supone todos objetos asociados a Ci se han modificado

Desarrollo incremental

- Servicio remoto de cerrojos
  - No DSMCerrojos: sólo proporcionan acceso exclusivo y compartido
    - Pero sin objetos asociados
- Servicio remoto de almacén de objetos
- Funcionalidad final: clase DSMCerrojo
  - Basada en ambos servicios remotos
    - Secciones críticas gracias a servicio remoto de cerrojos
    - Gestión de objetos asociados a DSMCerrojos mediante almacén

Componentes de JavaDSM

- FabricaCerrojos: interfaz remoto e implementación
  - Creación de cerrojos remotos
- Cerrojo: interfaz remoto e implementación
  - Operaciones sobre un cerrojo remoto
- ObjetoCompartido: clase que asocia nombre y versión a objeto
- Almacén: interfaz remoto e implementación
  - Repositorio de objetos
- ServidorDSM:
  - Registra servicios de FabricaCerrojos y Almacén
    - Puerto del rmiregistry como primer argumento
- DSMCerrojo: clase con funcionalidad final
- Clientes...

Servicio remoto de cerrojos

- FabricaCerrojos almacena cerrojos existentes:
  - Contenedor con entradas: (nombre, Cerrojo)
  - Ofrece método iniciar que especifica nombre de cerrojo
    - Retorna objeto de tipo Cerrojo
    - Si no existe: lo incluye en el contenedor y lo retorna
    - Si existe: retorna el almacenado en el contenedor
- Cerrojo
  - Implementa operaciones adquirir (exc.|comp.) y liberar
    - Típica sincronización lectores/escritores
- Por simplicidad, control de errores mínimo
  - En liberar: sólo que el cerrojo está previamente cerrado
- Sin operaciones de destrucción explícitas o implícitas

Clase ObjetoCompartido

- Envoltorio de un objeto que se pretende compartir
- Constructor recibe nombre de objeto y el propio objeto
  - Objeto debe ser un array (por simplicidad en la implementación):
    - ObjetoCompartido(String nombre, Object [] objeto);
  - Asocia un número de versión
- Se apoya en objeto CabeceraObjetoCompartido
  - Incluye nombre y versión, pero no objeto que se pretende compartir
- Proporciona métodos para leer/escribir sus campos
  - Método para asignar nuevo contenido a objeto envuelto + complejo
  - Comprueba que objeto es de la clase adecuada
- Ya está programada

Servicio remoto de almacén

- Gracias a *serialización* automática de Java RMI
  - Parámetros y resultado de métodos RMI son *serializados*
- Guarda objetos existentes (colección ObjetoCompartido)
- Ofrece métodos para leer/escribir múltiples objetos
- leerObjetos: recibe lista de cabeceras de objetos en cliente:
  - Retorna lista de objetos obsoletos en el cliente
  - Por cada objeto
    - Si versión cliente < actual: incluye ObjetoCompartido en lista retornada
    - Si versión cliente = actual || objeto no existe en gestor: nada
- escribirObjetos: recibe lista de objetos compartidos del cliente:
  - Los incluye directamente en el almacén
- Descarga dinámica de clases si objeto de clase de usuario
- Sin operaciones de destrucción explícitas o implícitas

DSMCerrojo

- Usa dos servicios remotos previos
  - Obtiene del registro servicios remotos de cerrojos y de almacén
    - Host y puerto del rmiregistry en variables de entorno
- Contenedor para objetos asociados al cerrojo
- (des)asociar: elimina/agrega ObjetoCompartido de contenedor
- Método adquirir (exc.|comp.) de DSMCerrojo:
  - Usa mismo método de cerrojo remoto vinculado
  - Llama a leerObjetos para todos los ObjetosCompartidos asociados
  - Actualiza localmente objetos obsoletos
- Método liberar (exc.|comp.) de DSMCerrojo:
  - Si exc.: escribirObjetos para ObjetosCompartidos asociados (++vers.)
  - Usa mismo método de cerrojo remoto vinculado