Université du Québec
École de technologie supérieure

Département de génie logiciel et des TI

Rapport de laboratoire

N° de laboratoire	1
Étudiant(s)	Steven Pelletier Max Moreau
Code(s) permanent(s)	PELS11068809 MORM30038905
Cours	LOG720
Session	Automne 2014
Groupe	01
Professeur(e)	Carle Côté
Chargé de laboratoire	Carle Côté
Date	2014-10-13

Table des matières

Introduction

Ce laboratoire consiste à construire une application distribuée en utilisant l'architecture CORBA. Ce dernier permet de lier divers objets et d'utiliser des ressources provenant de divers hôtes, tout en donnant l'impression que l'application roule en local sur le poste de l'usager. CORBA offre des techniques pour masquer l'hétérogénéité des systèmes qui ont été utilisés dans le cadre de ce laboratoire.

Analyse

Partie 1

Niveau de portée et concept JAVA

grid.idl
-Scope demo => package
+-Scope grid => package
+-Scope MyGrid => class
+-Scope GridException => function

Fichiers IDL générés

demo => Dossier / representation du package demo (scope demo)
demo/grid => Dossier / representation du package grid (scope grid)

demo/grid/MyGridHelper.java	=> Méthode pour Caster vers le type Grid
demo/grid/MyGrid.java	=> Interface définissant les service de l'objet MyGrid
demo/grid/MyGridPOATie.java	=> Utilisé par le POA
demo/grid/MyGridHolder.java	=>Utilisé lors du passage d'instances de l'objet MyGrid
demo/grid/MyGridOperations.java	=> Liste les opérations MyGrid
demo/grid/MyGridPOA.java	=> Portable Object Adaptator MyGrid
demo/grid/_MyGridStub.java	=>Stub de MyGrid (Client)

demo/grid/MyGridPackage => Dossier / representation du package MyGridPackage (scope MyGrid)
demo/grid/MyGridPackage/GridExceptionHelper.java => Support au error reporting
demo/grid/MyGridPackage/GridExceptionHolder.java => Utilisé pour le rapport d'erreur lors du passage d'instances
demo/grid/MyGridPackage/GridException.java => Utilisé pour le rapport d'erreur lors de la définition de services

Différence entre serveur et servant

Server = Programme offrant des services, contient un ensemble d'objet disponible pour accès distant.

Servant = Object permettant d'accéder a un objet distant, (ressemble a l'invocator).

Références:

http://en.wikipedia.org/wiki/Servant_%28CORBA%29
http://www.omg.org/gettingstarted/orb_basics.htm

Étapes pour créer les requis du côté serveur

```
org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args, null);                    => 1
try {
        org.omg.PortableServer.POA poa = org.omg.PortableServer.POAHelper      => 2
                        .narrow(orb.resolve_initial_references("RootPOA"));
// 3
        poa.the_POAManager().activate();

        org.omg.CORBA.Object o = poa.servant_to_reference(new GridImpl());     => 4

        if (args.length == 1) {
                                                                               => 5
                // write the object reference to args[0]

                PrintWriter ps = new PrintWriter(new FileOutputStream(new File(
                                args[0])));
                ps.println(orb.object_to_string(o));
                ps.close();
        } else {
                // use the naming service

                NamingContextExt nc = NamingContextExtHelper.narrow(orb        => 6
                                .resolve_initial_references("NameService"));
                nc.rebind(nc.to_name("grid.example"), o);                      => 7
        }

        orb.run();
                                                                               => 6
} catch (Exception e) {
        e.printStackTrace();
}
```

1) Création de l'orb (intergiciel)
2) Récupération du POA
3)Définition des politiques du POA (optionel)
4) Création du servant "GridImpl()", et mise en référence
5) Non essentiel, mais serialisation de l'objet dans fichier
6) Recuperation du lookup
7) Assigner le nom "grid.example" au servant et disponible pour ecoute sous ce nom.
6) Démarrage et mise en écoute

Références:
http://docs.oracle.com/javase/7/docs/technotes/guides/idl/POA.html

Transparence de localisation de la solution

banque_abc = BanqueABCHelper.narrow(nc.resolve("ressource.java"))

Selon la commande utilisée pour accéder à une ressource, il n'est pas nécessaire de spécifier la localisation de cette ressource, le naming context est responsable de la trouver et de la transmettre. On peut donc affirmer qu'il y a une transparence de localisation dans la solution.

Partie 2

Activation et désactivation

L'activation sert a réserver de l'espace mémoire et du CPU pour le servant.

La désactivation sert a relâcher les ressources (release) associés à ce servant.

Références:

http://www.omg.org/gettingstarted/orb_basics.htm

POA: Retrouver l'objet servant à partir d'une référence CORBA

Dépend du 'vendor', mais le POA a un 'hash' contenant les 'objetID' et l'objet (objhash<objID,obj>

POA: Déterminer si un objet est activé ou non

Cela dépend probablement du 'vendor' de POA, mais si l'on se fie a la documentation d'IBM, le POA aurait un 'hash' contenant l'ensemble des objet présent et leur états.

quote: *"A map of object IDs and active servants is stored inside the POA"*

Références :

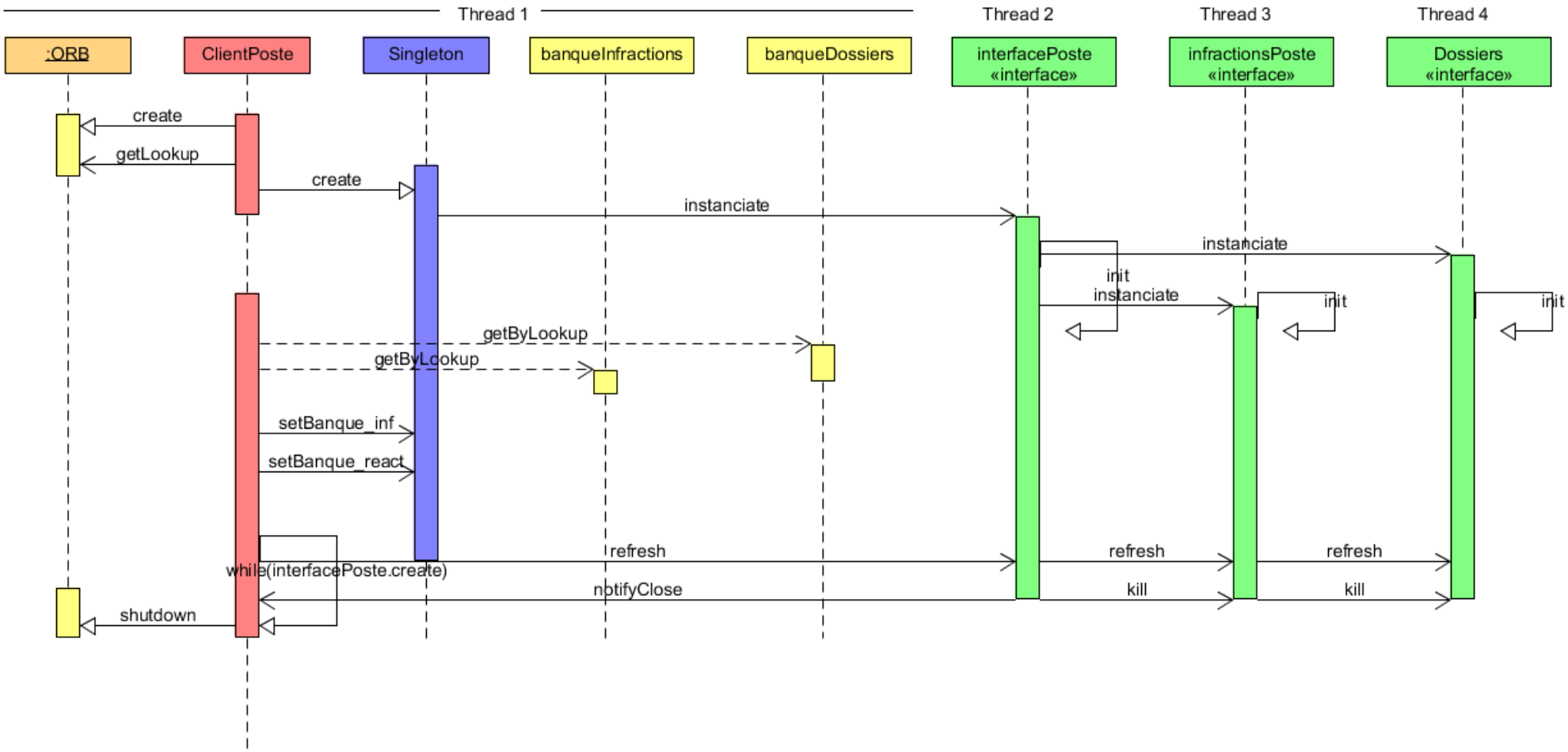
http://www01.ibm.com/support/knowledgecenter/SSYKE2_5.0.0/com.ibm.java.doc.diagnostics.50/diag/understanding/orb_extras_poa.html

Objet tentant d'accéder aux services d'un autre objet désactivé

Le résultat dépends des 'policies' définies dans le POA principalement. Suite à des tests d'accès sur des objets qui sont relâchée, l'objet semble être réactivé sans action supplémentaire.

Diagrammes UML pertinents

Ce diagramme décrit le cycle de vie de notre solutions. On aperçoit les grandes étapes de la création d'un client. La boucle while est l'endroit ou s'effectuent les appels divers des clients, ainsi que sont rafraichissement par rapport aux objets distants. La dernière étape montre les étapes nécessaires à la fermeture de l'application.



Conclusion

Au final, malgré les configurations nécessaires avant de pouvoir utiliser CORBA sur nos postes, sa contribution au système développé est assez importante pour affirmer qu'il est un outil de choix dans la construction d'une application. Le potentiel d'un système distribué est énorme, tant au niveau de la force de calcul, qu'à valider l'exactitude de ces dernières, mais de nombreux défis doivent êtres surmontés, ce qui est exactement la raison d'être de CORBA, de créer une architecture d'objets communs et de simplifier le développement de systèmes distribués.