ENSEEIHT 2ième année Département Sciences du Numérique. Parcours ABLMR. Année 2021-2022

Intergiciels

Examen session 1

Daniel Hagimont

NOM : …….

Durée: 1h30, documents autorisés

*Lire l'ensemble des énoncés avant de commencer à répondre. La **clarté**, la **précision** et la **concision** des réponses, ainsi que leur **présentation matérielle**, seront des éléments importants d'appréciation. Donnez essentiellement les programmes Java demandés. Dans vos programmes, vous n'avez pas à programmer les imports et le traitement des exceptions.*

PROBLÈME (Sockets et RMI) (16 points)

Nous voulons implanter en Java un service de diffusion de message à un groupe d'applications, les applications pouvant s'insérer dynamiquement dans le groupe. Ce service permet à des applications réparties de s'insérer dans le groupe (un seul groupe est géré), les messages envoyés au groupe étant reçus par toutes les applications insérées dans le groupe. La procédure d'insertion dans le groupe est implantée en utilisant Java-RMI et la diffusion de message dans le groupe est implantée avec des Sockets.

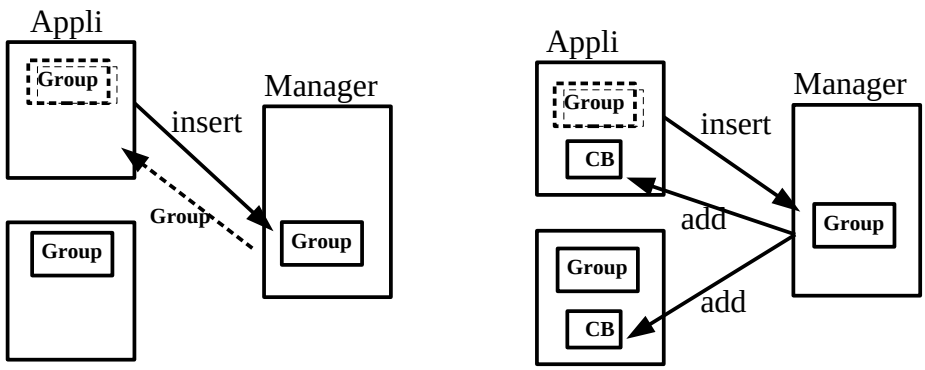

Figure 1

La procédure d'insertion dans le groupe est structurée comme suit :

1) Un objet réparti Manager d'interface/classe Manager/ManagerImpl permet à une application de s'insérer dans le groupe. La méthode d'insertion prend en paramètre (par copie) un objet de la classe Host (ci-dessous) contenant les coordonnées (host/port) de l'application permettant la communication par messages. Le Manager inclut un objet Group d'interface/classe Group/GroupImpl qui inclut la liste des Host présents dans le groupe. L'objet Group est retournée (par copie) en résultat de la méthode d'insertion pour que l'application s'insérant récupère l'état courant du groupe (Figure 1 – gauche).

2) La méthode d'insertion prend un paramètre de type CallBack (par référence) lui permettant d'être avertie de l'arrivée d'une nouvelle application dans le groupe. Le Manager gère donc une liste des Callbacks permettant d'avertir les membres du groupe de l'arrivée d'un nouveau membre (Figure 1 – droite).

La classe Host est donnée :

```
public class Host implements Serializable {
      public Host(String h, int p) { host=h; port=p; }
      public String host;
      public int port;
}
```

Les interfaces ci-dessus définissent donc les méthodes suivantes :

Manager

```
public Group insert (Host h, CallBack cb);
```

CallBack

```
public void addNewMember (Host h);
```

La diffusion des messages fonctionne comme suit :

1) Au début, chaque application démarre un thread chargé de recevoir les messages (en acceptant des connexion TCP sur hostname + port qui sont des paramètres (args) de l'application). Ce thread se contente d'afficher les messages reçus sur le terminal.
2) Ensuite, comme décrit ci-dessus, l'application s'insère dans le groupe en appelant le Manager. Elle récupère ainsi l'état courant du groupe. Lors de l'insertion, elle passe en paramètre un objet CallBack pour être avertie de l'arrivée d'un nouveau membre.
3) Enfin les applications peuvent communiquer en utilisant le groupe (Group) contenant les coordonnées de toutes les applications. Pour ce faire, Group fournit une méthode send() permettant d'envoyer un message (une String). Chaque envoi de message est implanté avec une nouvelle connexion TCP et l'envoi du message (Figure 2). Vous pouvez utiliser la sérialisation pour envoyer/recevoir un message.

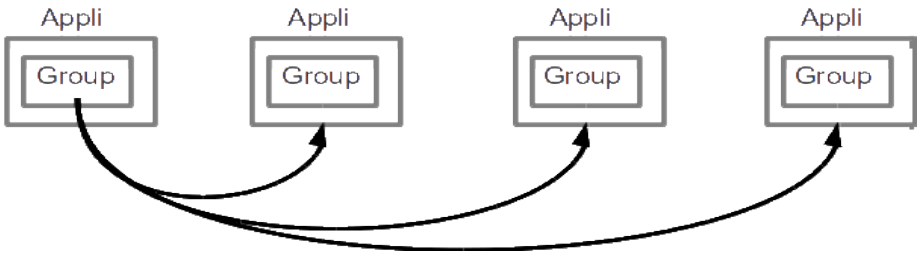

Figure 2

L'interface Group définit donc les méthodes suivantes :

Group

```
public void add(Host h);
public void send(String message);
```

Notez que la méthode add() est appelée dans le Manager et dans le CallBack, alors que la méthode send() est appelée par l'application uniquement.
On ne traitera pas les problèmes liés à la synchronisation.

Vous devez donner l'implantation
- des interfaces : Group, Manager, CallBack
- des classes : GroupImpl, ManagerImpl, CallBackImpl
- une classe Appli qui montre comment on utilise le service.

QUESTIONS DE COURS (4 points)

1) Expliquez la différence entre Web Services SOAP et Web Services REST. Vous devez expliquer dans chaque cas ce qui est **fourni par le serveur exportant** un WS et ce qui est **utilisé par le client utilisant** un WS (2 lignes)

WS SOAP :

WS REST :

2) Supposons que nous disposons de 3 applications : une application A permettant l'exportation de données dans un fichier, une application B permettant l'importation de données dans une base de données, et une application C permettant l'importation de données avec un Web Service. Nous voulons que les données exportées par A soient importées par B et C. Dessinez l'architecture de principe d'interconnexion avec l'ESB Mule en ne détaillant que les endpoints utilisés. (juste un dessin)