Travaux Dirigés CORBA n°1

Université Paris Est, Marne la Vallée

—2007-2008—

Le langage IDL

L'objectif de ce td est de développer une application de forum interrogeable à distance, permettant de poster ou de récupérer des messages.

▸ **Exercice 1.** ***Une application de forum simple***

Pour développer l'application forum, nous vous proposons de respecter une spécification IDL prédéfinie afin d'assurer l'interopérabilité entre vos différentes implantations de fora.

Les messages échangés ne sont pas des objets CORBA. Ils sont représentés par la structure IDL suivante :

```
struct Message {
  string title;
  string author;
  string date;
  string body;
};
```

Le forum est un objet CORBA dédié à la gestion des messages d'un thème particulier (attribut `theme`*) et sous la responsabilité d'un administrateur (attribut* `moderator`*). Il est représenté par l'interface IDL suivante :*

```
interface Forum {
  readonly attribute string theme;
  readonly attribute string moderator;
  boolean postMessage(in Message m);
  Message getMessage(in string title);
  boolean removeMessage(in string title);
};
```

En respectant les étapes vues dans le td précédent, développez l'application forum, c'est-à-dire l'interface IDL, le servant, le serveur et le client. Pour stocker les messages on pourra utiliser une table hachage `ConcurrentHashMap` *qui associe chaque message à son titre.*

▸ **Exercice 2.** ***Les séquences***

Modifier les fichiers de votre application de forum afin d'intégrer un type séquence de messages (`MessageSet`*), puis ajouter dans l'interface* `Forum` *l'opération* `getMessages()` *qui renvoie la liste des messages du forum.*

Rappelons qu'en IDL, une séquence correspond à un tableau de taille variable et qu'un type séquence de nom `TSet` *contenant des objets de type* `T` *est déclaré par :*

```
typedef sequence <T> TSet
```

▸ **Exercice 3.** ***Les exceptions***

On souhaite maintenant améliorer l'interface de notre forum en lui faisant lever des exceptions en cas d'erreur. Pour cela, il faut d'abord définir en IDL les exceptions avec la syntaxe suivante :

```
exception ExceptionName { string message; };
```

Ensuite il faut indiquer dans l'interface du forum les méthodes qui lèvent une exception. Rappelons qu'une méthode suivie du mot clef `raises` *et du nom d'une exception entre parenthèses, indique que la méthode est susceptible de lever ce type d'exception au niveau du client au retour de l'appel.*

Mettre en œuvre l'exception `Reject` *qui sera levée par la méthode* `postMessage()` *lorsque un message de même titre existe déjà et par les méthodes* `getMessage()` *et* `removeMessage()` *s'il n'existe pas de message avec ce nom.*

▸ **Exercice 4.** ***Les paramètres*** `out`

Dans cet exercice nous souhaitons ajouter à l'interface `Forum` *la méthode* `getInfo()` *suivante, qui retourne les différentes informations sur le forum :*

```
interface Forum {
  void getInfo(out string theme, out string moderator,
               out long size);
};
```

Nous rappelons que les classes `IntHolder` *et* `StringHolder`*, entre autres, permettent de transmettre des paramètres par référence ; ce qui permet de récupérer la valeur si elle a été modifiée par le serveur.*

▸ **Exercice 5.** ***Délégation*** *Dans cet exercice vous allez ajouter l'interface* `ForumAdmin` *a votre fichier* `Forum.idl`*. L'interface propose les deux meme attributs que Forum mais avec une autorisation d acces en écriture. Rappelons que pour cela vous ne devez pas étendre ForumPOA et ForumAdminPOA ce qui impossible en Java mais implanter les deux interfaces* `ForumOperations` *et* `ForumAdminOperations`*. Pour cela, vous n'utiliserez qu'une seule classe : $Forum_i$. Vous utiliserez L'option "-fallTIE" du compilateur*

`IDLJ` pour génerer les classe skeleton nécessaires a cette implantation par délégation. Vous developperez une client qui va utiliser l'interface `forum` (postMessage puis getMessage), lire l'attribut `moderateur` du forum puis qui va ensuite le modifier.

```
interface ForumAdmin{
attribute String Theme;
attribute String Moderateur;
}
```