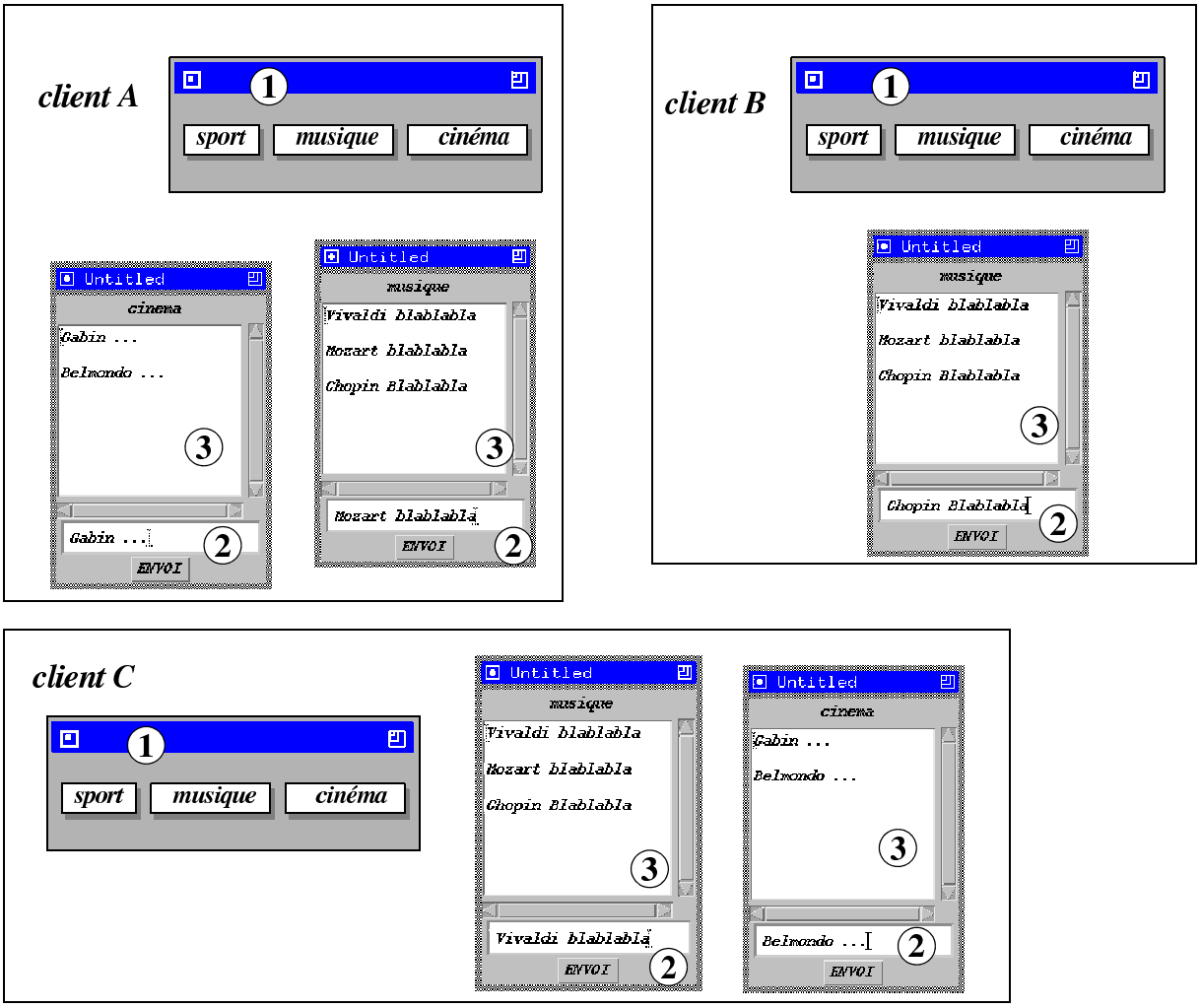
Exercice 3 - TP2 - Application répartie (RMI) forum de discussion

On veut réaliser un forum permettant à des clients situés sur des machines diverses de participer à des discussions sur divers sujets, trois dans l'exemple : ***sport***, ***musique*** et ***cinéma***. Le client s'inscrit à un sujet en cliquant sur un des boutons de la fenêtre **(1)**, ce qui crée une fenêtre pour dialoguer sur ce sujet, composée de deux zone de textes, la zone **(2)** pour composer les messages et la zone **(3)** pour afficher tous les messages des divers protagonistes de la discussion. Un bouton ***ENVOI*** permet d'émettre le message nouvellement composé.
Le client peut se désinscrire d'un sujet en cliquant à nouveau sur le bouton du sujet.
Chaque client peut participer en même temps à autant de sujets de discussion qu'il veut. Dans l'exemple illustré sur la figure, les clients ***A*** et ***C*** sont inscrits aux sujets ***musique*** et ***cinéma*** et le client ***B*** est inscrit au sujet ***musique***. Un client reçoit uniquement les messages émis pendant qu'il est inscrit au sujet de discussion concerné (le serveur ne mémorise pas les messages, il ne fait que les diffuser).



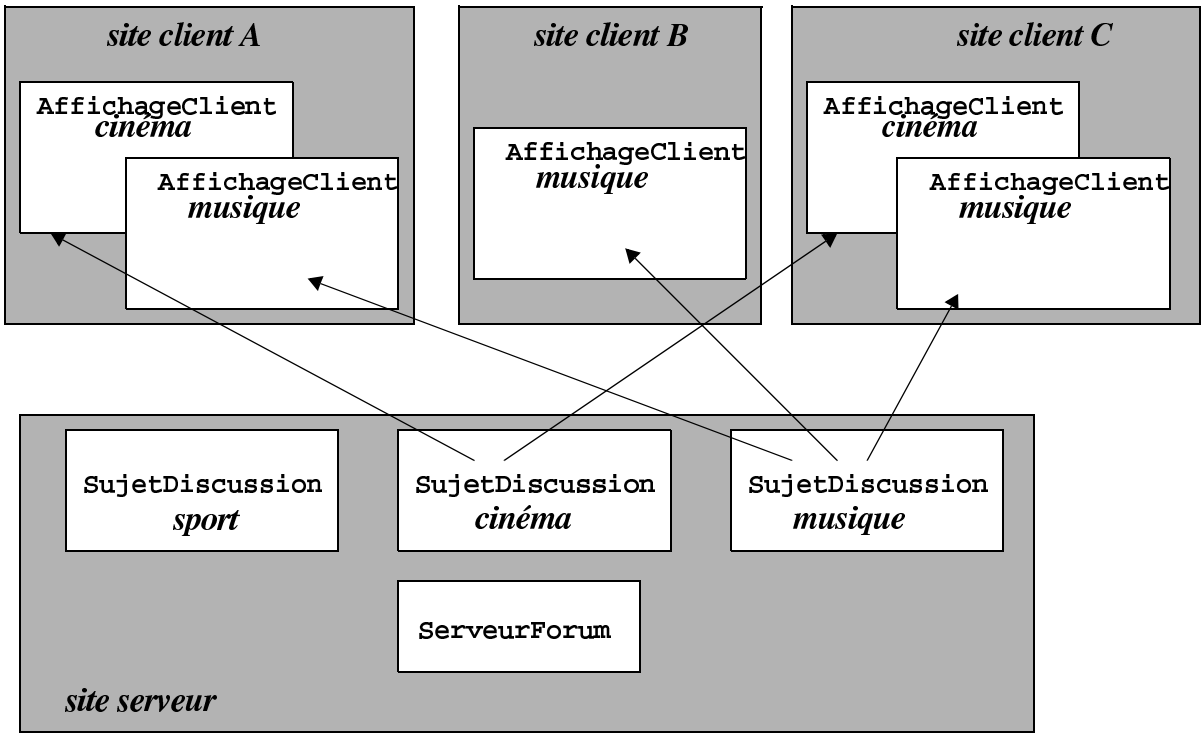
L'architecture de cette application est illustrée sur le schéma suivant.

Sur le site serveur résident 2 types d'objets distants :

- Le serveur lui-même, défini par la classe **`ServeurForum`**, objet unique connu par les clients grâce à un nom externe de la forme "**`//machine:port/leServeur`**"
- Les sujets de discussion, définis par la classe **`SujetDiscussion`**.

Sur un site client réside un type d'objet distant, défini par la classe **`AffichageClient`**. Il y a un objet de cette classe par sujet auquel est inscrit le client. Un tel objet est matérialisé par la fenêtre permettant la composition de message et l'affichage de la discussion.

Dans cette application simple, un sujet de discussion n'est rien de plus que la liste des affichages clients qui participent à la discussion. Il permet de diffuser le message émis par un client vers tous les autres participants.



Interface d'objet distant pour le serveur de forum,
objet unique, localisé sur le site serveur,
connu par un nom externe "//machine:port/leServeur"

```
public interface InterfaceServeurForum extends Remote {

        public InterfaceSujetDiscussion obtientSujet(String titre)
                                                    throws RemoteException;
```
rend en résultat le sujet de discussion identifié par la chaine de caractère titre : "sport", "musique", "cinema", ...
rend **`null`** si le titre ne correspond à aucun sujet
```
}
```

Interface d'objet distant pour les sujets de discussion : sport, musique, cinema, ...
objets localisés sur le site serveur
captés depuis les sites clients par références rendues en résultat par le site serveur

```
public interface InterfaceSujetDiscussion extends Remote {

public void inscription(InterfaceAffichageClient c) throws RemoteException;
```
inscrit l'afficheur client **`c`** à ce sujet de discussion

```
public void desInscription(InterfaceAffichageClient c)
                                                    throws RemoteException;
```
désinscrit **`c`**

```
public void diffuse(String Message) throws RemoteException;
```
diffuse le message à tous les afficheurs clients couramment inscrits à ce sujet
```
}
```

Interface d'objet distant pour les afficheurs clients d'un sujet de discussion
objets localisés sur les sites clients du forum, captés sur le site serveur par références passées en paramètre depuis les sites clients

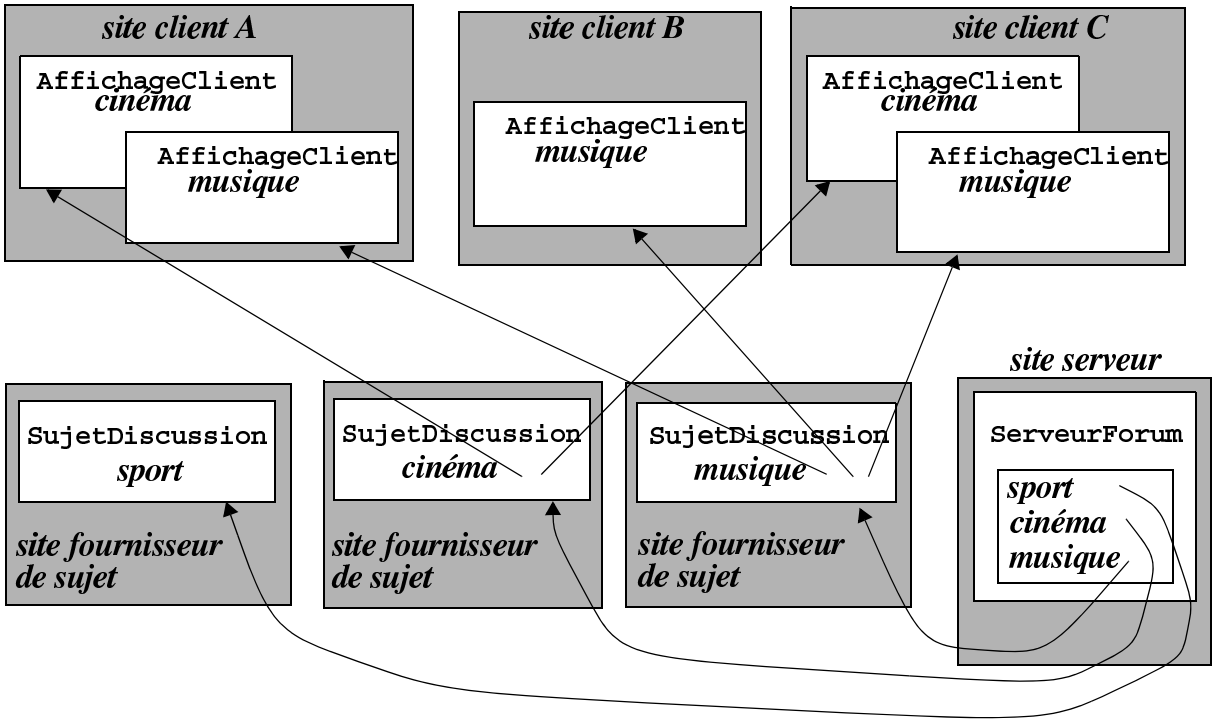
```
public interface InterfaceAffichageClient extends Remote {

public void affiche(String Message) throws RemoteException;
```
affiche le message sur cet afficheur client
```
}
```

Exercice 4 - TP3 - Forum amélioré

On désire améliorer l'application répartie ***forum de discussion***, programmée en travaux pratiques à l'aide des appels distants du système RMI de Java. Nous rappelons en annexe l'architecture de l'application telle qu'elle a été programmée en travaux pratiques.
Nous voulons apporter les améliorations suivantes :

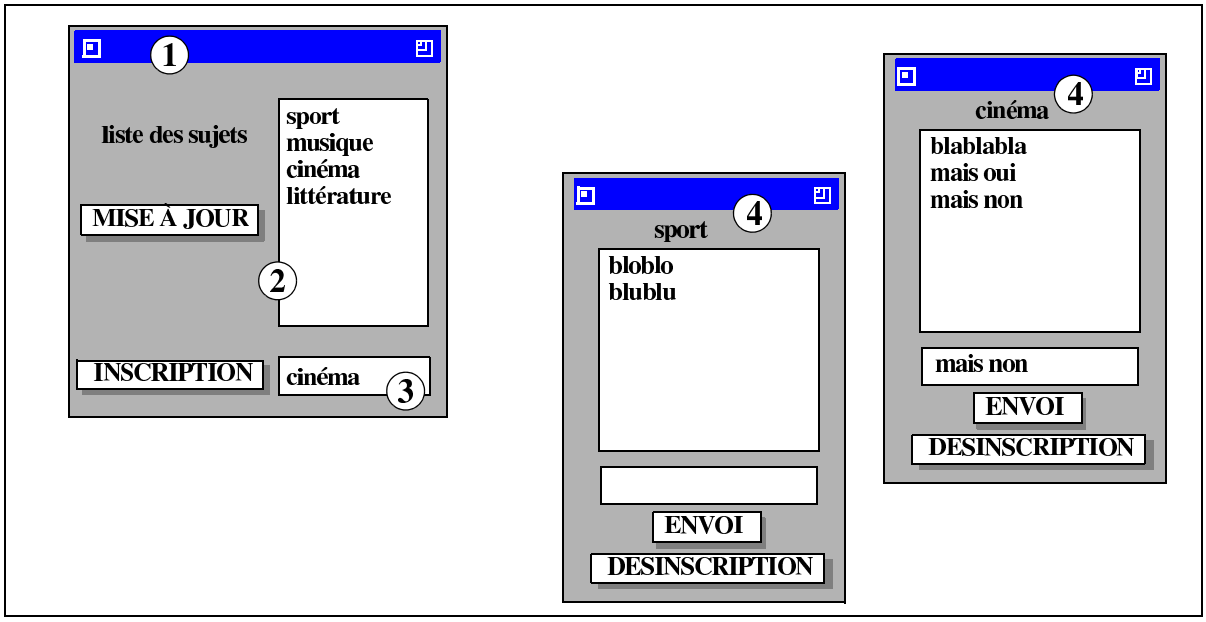
- ***Création dynamique de sujets de discussion*** : les sujets de discussions peuvent être créés à volonté.
- ***Répartition des sujets de discussion*** : pour éviter de surcharger le site du serveur du forum par les activités de diffusion des messages sur tous les sujets, on décide de répartir les sujets de discussion sur des sites distincts appelés ***sites fournisseurs***.

L'architecture de cette nouvelle version peut être illustrée par le schéma suivant :



Dans cet exemple, 3 sites fournisseurs de sujets se sont manifestés, pour les sujets respectifs *sport*, *cinéma* et *musique*. Le site serveur du forum se borne à conserver une table de correspondance entre les titres des sujets et les sujets situés sur les sites fournisseurs.

Un programme **`FournisseurDeSujet`**, exécuté sur un site fournisseur, crée un sujet de discussion et l'enregistre auprès du serveur de forum. Un fournisseur de sujet se lance par la commande : **`java FournisseurDeSujet`** *titre*



La fenêtre **(1)** est la fenêtre initiale du client. La zone de texte **(2)** montre la liste des sujets disponibles. Un bouton **MISE À JOUR** permet de réactualiser cette liste.

Pour s'inscrire à un sujet, le client saisit le titre du sujet dans la zone de texte **(3)** et appuie sur le bouton **INSCRIPTION**. Lors d'une inscription, une nouvelle fenêtre **(4)**, de type **`AffichageClient`** apparaît. Pour des raisons de simplicité, on ne cherche pas à empêcher qu'un client s'inscrive plusieurs fois à un même sujet. D'ailleurs cela peut être intéressant pour lui d'avoir plusieurs fenêtres ouvertes sur le même sujet.

Remarque : les sujets étant ici en nombre dynamique, la désinscription d'un sujet est faite par un bouton associé à l'affichage de ce sujet.