Programmation Réseau - Chat RMI

Le chat fonctionne avec polling (TPRMI2) et callback (TPRMI3).
La version la plus avancée est la version callback (TPRMI3), que je vais décrire.

TPRMI3 : callback

Fonctionnement général :

Au démarrage, le serveur indique quand il est prêt automatiquement.
Les clients peuvent ensuite se connecter, sans limite de nombre.
Pour ce faire, ils doivent indiquer un nom d'utilisateur dans la console, qui est stocké côté client, après quoi s'affiche le message de bienvenue (du côté du nouvel utilisateur uniquement). Puis un callback est généré et communiqué au serveur qui le stocke.
Le serveur envoie ensuite tous les messages de l'historique dans l'ordre pour recréer le fil de conversation.

Une fois connecté, un message s'affiche notifiant tous les utilisateurs.
Après ce message le client peut envoyer des messages indépendamment des autres utilisateurs : pour ce faire, ils entrent simplement leur message dans la console. Le serveur reçoit cette chaîne de caractère ainsi que l'username de l'expéditeur, crée un Message réunissant ces informations, l'enregistre dans son historique et le diffuse sur la liste des callbacks. On a donc un affichage en temps réel des messages envoyés par les différents utilisateurs.

Un utilisateur peut quitter le chat à tout moment en envoyant "quit" dans la console. Ce faisant, il se déconnecte proprement : il est retiré de la liste de diffusion et les autres utilisateurs reçoivent une notification.

Package Serveur :

Serveur :	
msgMemory :	Liste de Message, sert d'historique.
CBs :	Liste des callbacks, à notifier lors de la diffusion d'un nouveau message.

Les fonctions du serveur sont décrites dans IServ.

Message :	Sert à associer nom d'expéditeur et message reçu, pour facilement les restituer dans le chat. N'est utilisé que du côté serveur, déconstruit avant d'être diffusé.
user :	Nom de l'expéditeur.
msgStr :	Contenu du message

Package Client :

Client :	
Serveur :	Stub du serveur avec lequel le Client peut communiquer
Username :	Nom de l'utilisateur, utilisé lors des communications, à défaut de les garder en mémoire du côté serveur.

CBClient :	Callback d'un Client, met en forme les Message et les affiche dans la console.

Améliorations :

- Je ne gère pas toutes les déconnexions sauvages. La perte du serveur ou la déconnexion d'un utilisateur au mauvais moment génère toujours des erreurs. Erreurs qu'il faudrait gérer.

- Il pourrait être judicieux de garder l'username du côté serveur. Quitte à créer une classe User qui associerait username et CB. Cela ouvrirait la voie à plus de fonctionnalités (messages privés, historique personnel, …) et serait une meilleure séparation des données.