Projet de Système Distribué : Universal Messenger Point-to-Point

1 Travail demandé

Le travail demandé comporte une modélisation et une implantation en Java, C, C++ ou Lisaac (au choix). Un rapport court (2 pages maximum) devra accompagner ce projet pour décrire les choix de votre implantation en format papier.

Implantation

— Votre programme devra utiliser les ou une technologie de programmation distribuée vue en cours, à savoir : RPC, RMI, Corba ou Soap.
— Votre programme devra fonctionner sur la machine *turing* de l'UDS.
— L'interface utilisateur est textuelle (ligne de commande) ou graphique.
— Une petite explication de l'utilisation de votre programme est aussi nécessaire au démarrage de l'application avec l'option `-help`.

2 Remise du projet

La remise du projet se fera :

— Par le rapport papier à remettre dans le casier de Sonntag Benoit durant la (**semaine du 14 Avril 2014**).
— La programmation est à rendre en même temps, par courrier électronique (mail) à cette adresse : `benoit.sonntag@lisaac.org`. Ce mail aura comme titre **Projet de SD** et comportera un fichier attaché en format **.zip** contenant, les **sources** et la **version compilé** de votre programme préalablement disposés dans un **répertoire** ayant comme nom la concaténation des noms de famille de votre groupe. Le message du mail devra contenir impérativement la **liste des noms** du groupe.

3 Réalisation du projet

— La réalisation de ce projet devra se faire impérativement **par groupe de deux ou trois** pour que vous vous répartissiez le travail au sein du groupe.
— Comme tout cahier des charges, celui-ci ne peut être exhaustif. En cas d'ambiguïté, précisez votre interprétation personnelle, et éventuellement les questions à poser à votre interlocuteur (responsable de projet, futurs utilisateurs, etc.). Toute solution cohérente, justifiée et non contradictoire avec le cahier des charges sera acceptée.

4 Sujet

Nous vous proposons d'implanter un outil fort utilisé par les utilisateurs Windows : *MSN messenger (Microsoft Network Messenger).*

Cet outil est une communication texte quasi-instantanée entre deux ou plusieurs utilisateurs distants reliés par un réseau (Internet par exemple).

4.1 Fichier de configuration

Un fichier texte contenant la configuration du réseau actuel doit être maintenu par votre programme. Nous proposons de fixer la syntaxe à l'aide de la grammaire suivante :

```
LIST        →  ADDRESS '\n' LIST
            |  ADDRESS

ADDRESS     →  IDENTIFIER ':' HOST

IDENTIFIER  →  [a-zA-Z_]+

HOST        →  Host
            |  '?'
            |  LIST_ID

LIST_ID     →  IDENTIFIER ',' LIST_ID
            |  IDENTIFIER
```

Exemple de fichier de configuration :

```
Benoit : 212.85.150.133,0x20000000,1
Jerome : ?
Isaac  : Benoit, Jerome
```

Ici, l'identifiant `Benoit` possède bien une adresse connue. L'identifiant `Jerome` n'a pas d'adresse connue actuellement. L'identifiant `Isaac` permet d'envoyer un message directement à `Benoit` et à `Jerome`.

Le programme maintient ce fichier à jour automatiquement. L'utilisateur doit aussi pouvoir modifier manuellement ce fichier (pour définir des groupes comme `Isaac` par exemple).

4.2 Envoi d'un message

L'envoi d'un message aura la syntaxe suivante :

```
MESSAGE     →  ':' String '\n'
            |  LIST_ID ':' String '\n'
            |  COMMAND

LIST_ID     →  IDENTIFIER ',' LIST_ID
            |  IDENTIFIER

IDENTIFIER  →  [a-zA-Z_]+

COMMAND     →  'bye'
```

Exemple de message :

```
Benoit : Hello
Benoit, Jerome : Quoi de 9 ?
Isaac : Bye !
: A bientot.
bye
```

L'action de la touche *"Enter"* envoie le message en question. Un message peut être précédé par une liste de destinataire, ou d'un identifiant de groupe (ici : `Isaac`). En absence de destinataire (message commençant par ' :'), le programme considère les destinataires précédents (si il n'y en a pas, c'est un cas

d'erreur). L'absence de ' :' donne accès aux commandes de votre application. Ici, seule la commande `bye` (non précédée de ' :') est autorisée et permet de quitter l'application.

Un identifiant de destinataire inconnu fait l'objet d'une recherche sur l'ensemble des adresses connues pour obtenir son adresse. Si aucune machine n'est capable de répondre à cette recherche, un cas d'erreur est signalé.

Lors d'un envoi de message, vous devez vérifier que l'adresse utilisée corresponde bien à la personne désirée.

4.3 Reception d'un message

L'affichage d'un message suit la même syntaxe que l'envoi, précédé du caractère '>' permettant de distinguer rapidement la réception d'un envoi.

Par exemple, vous pouvez avoir la discution suivante :

```
Jerome : Salut Jerome.

> Jerome : Salut Ben. Quoi de 9 ?

: Je rédige le sujet du projet de prog. dist., et toi ?

> Jerome : Rien, mais je passe sur Stras, le 31 Mars, OK?

: Ca marche !
: A+, j'ai encore pas mal de boulot.

> Jerome : Bye! A bientot

bye
```

4.4 Déroulement du programme

Au lancement de votre application, l'utilisateur doit préciser son identifiant en paramètre.

Exemple :

```
my_msn Benoit
```

Lors de l'exécution de votre programme, vous êtes succeptible de recevoir un message de n'importe qui ayant votre adresse. Si vous avez commencé à écrire un message, le programme attendra l'envoi de votre message pour afficher les messages reçus.

Ensuite, la communication doit être point-à-point.

4.5 Evolution possible de votre application

Voici quelques idées d'amélioration de votre application :

- Définir un identifiant particulier permettant l'envoi d'un message à tout le monde, celui-ci doit se propager à l'ensemble des identifiants connus par votre application, mais aussi, à l'ensemble des identifiants connus par les autres.
- Interdire la réception d'un message provenant d'un identifiant en particulier, en ajoutant un mot clé dans votre fichier de configuration.
- Informer l'utilisateur de la déconnexion d'un identifiant qui a été utilisé lors d'une session.
- Ajouter une commande permettant de connaitre l'ensemble des identifiants actuellement disponible.
- Permettre l'envoi d'un fichier en même temps qu'un message (Exemple : `Jerome : Salut je t'envoie ce fichier ''archive/lisaac.zip'', pour que tu puisses le tester.`) Ici, le texte mis entre guillemet permet automatiquement de préciser un chemin de fichier et de l'envoyer. Il sera directement écrit dans le répertoire courant du destinataire.

Good luck !