Laboratoire 3
Programmation Socket et B8ZS

CEG 3585 - Communication de données et réseautage

Université d'Ottawa

Professeur : Mohamed Ali Ibrahim

Noms et numéros des étudiants :
Gbegbe Decaho Jacques 300094197
Sissoko Ramatoullaye Bahio 300144949

Date de soumission:12-03-2023

Table des matières:

1- Description du problème

Le but de ce laboratoire est d'implémenter deux programmes en utilisant les sockets pour transmettre B8ZS à partir du programme d'encodage (CLIENT) vers le programme de décodage (SERVEUR). Le programme de décodage décode le flux B8ZS reçu au format original.

2- Explication du programme de solution

Dans ce lab, nous avons été amené à nous familiariser un peu plus avec les socket et les threads. Ce programme traite de l'encodage en B8ZS d'une chaîne binaire d'entrée. Pour pouvoir résoudre ce problème, nous avons configuré une classe server.java et slient.java dans laquelle nous avons utilisé des sockets.
Afin de configurer notre thread, dans les deux fonctions nous avons utilisé certaines méthodes de la bibliothèque java tel que (PrintStream, BufferedReader et DataOutputStream) qui chacun:

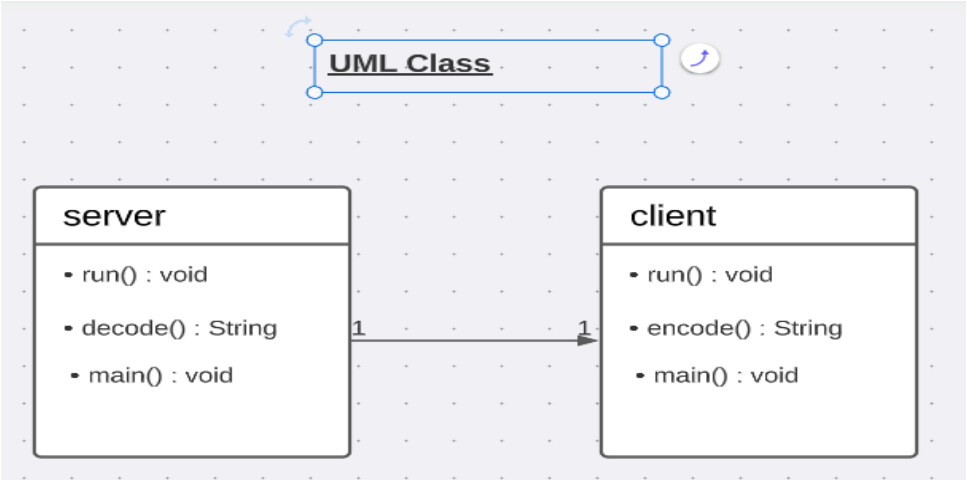
- PrintStream: afin d'envoyer des données
- BufferedReader: afin de faire la lecture des données reçues
- DataOutputStream: afin d'envoyer des données entre les 2 classes

Afin de faciliter notre travail, nous avons implémenté à la suite de recherche un outil JPanel dans notre code pour déjà l'aspect visuel futuriste et aussi le modèle simpliste que cela nous apporte, cet outil nous a permis de mettre en lien les 2 classes et leurs méthodes decode pour le serveur et encode pour le client qui respectivement:

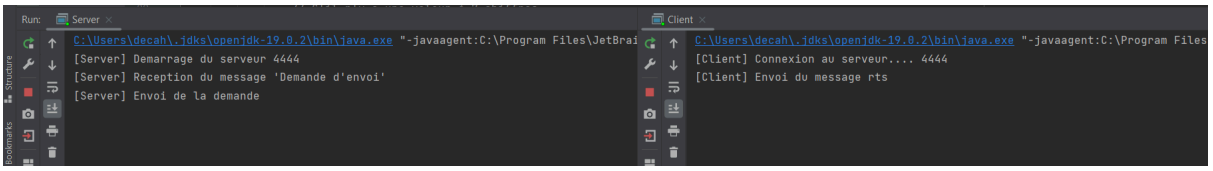
- Decode: reçoit une chaîne de caractères suivant le modèle de flux B8ZS et le change en chaîne binaire
- Encode : reçoit une chaîne de caractères binaire en entrée et la transforme en expression suivant le modèle de transformation B8ZS.

Cela a grandement facilité notre travail. Puis à la fin de nos 2 classes nous avons fermé les sockets qui servaient à maintenir les thread et la connexion entre eux.

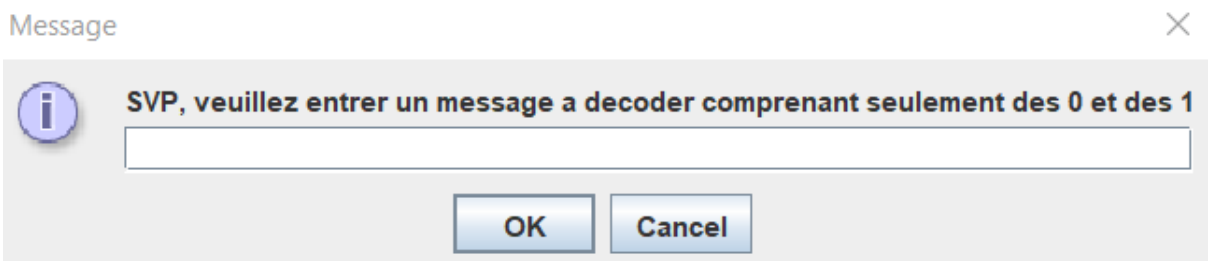
3- Document Représentatif (UML)



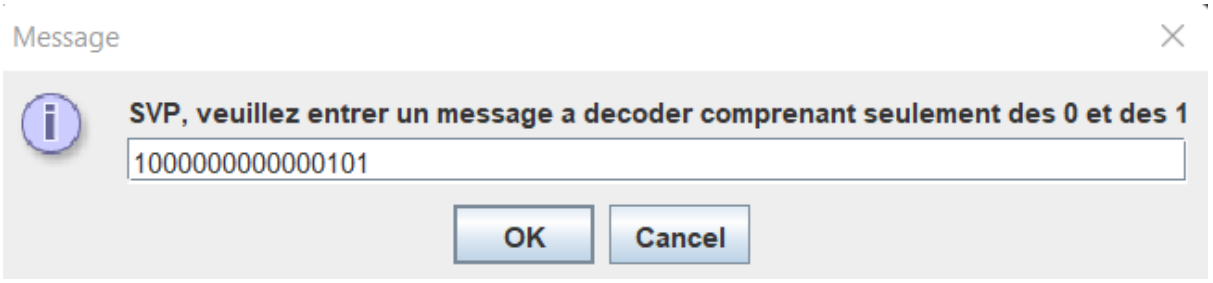
4- Capture d'écran



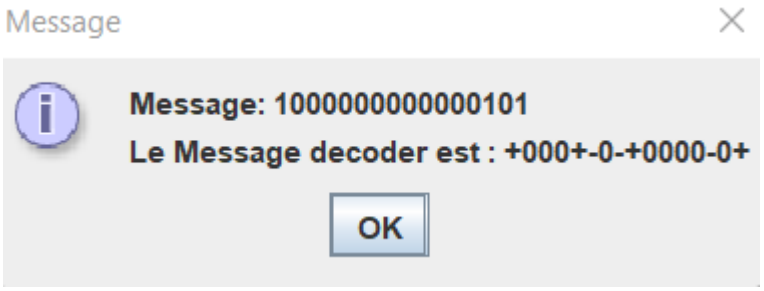
Lancement du serveur et du client



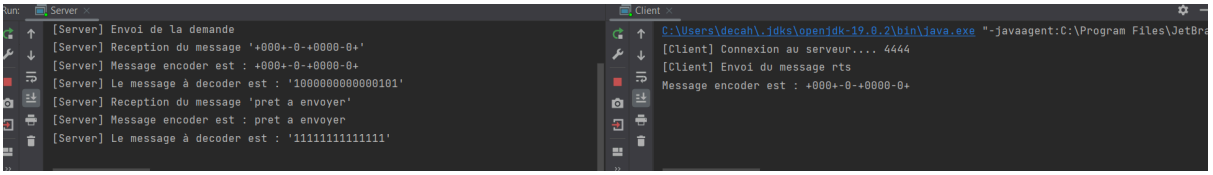
La fenêtre de dialogue qui s'affiche au run de client



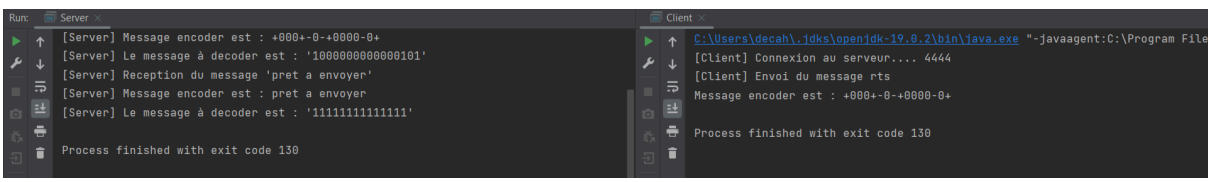
entrée de la chaîne de nombre binaire à décoder



fenêtre de dialogue affichant le résultat de l'opération



Affichage de la réponse dans l'invite de commande



Fermeture des thread

5- Discussion

Dans ce laboratoire on a utilisé toutes les données misent à notre disposition pour mener à bien le travail demandé. Néanmoins on a eu quelques difficultés notamment le transfert et la transformation du flux de la méthode decode à la méthode encode. Ceci a été la partie qui nous a donné le plus de mal à être réalisé mais toutefois ce fut une belle expérience de nous familiariser un peu plus avec les sockets et les thread. Pour la résolution de ce problème nous avons simplement utilisé l'outil de la bibliothèque java (Jpanel) afin de simplifier notre travail et pouvoir concatener facilement certaines expressions. Finalement, ce fut une expérience différente vu que comparé à un serveur sur lequel 2 clients discutent, celui-ci est une interface qui interagit avec l'utilisateur en lui apportant uniquement ce qu'il demande et rien d'autre ce qui s'ajoute à la continuité du sujet du laboratoire 2

6- Conclusion

L'objectif du laboratoire 3 était, dans un premier temps, de trouver et de programmer le flux B8ZS d'une chaîne binaire de données. Dans un deuxième temps, de transcrire le changement de ce flux d'un état de chaîne binaire de donnée en flux B8ZS et vice-versa.
En somme, ce laboratoire a été gratifiant pour nous car il nous a permis d'approfondir nos connaissances en matière de sockets, de clavardage entre socket, la connexion client-serveur et flux de données B8ZS. On a pu affronter et régler les différents problèmes rencontrés lors de l'implémentation des différentes classes et nous nous en servirons pour les prochaines expériences.