C) Aspect fonctionnel de la partie C++

1) Présentation de l’application C++

L’application C++ a pour but d’envoyer et de recevoir des données avec le serveur ; cette communication permet le déroulement d’une partie de poker.

La communication Client C++/ Serveur s’effectue par l’envoi et la réception de socket.

Cette application est constituée de différentes fonctions qui interagissent entre elles grâce à des *Threads*.

L’ensemble et le fonctionnement de ces fonctions ne sont pas bien compliqués, elles établissent le lien au sens imagé entre le serveur et l’homme.

Le serveur a besoin de mot clé très précis, l’application demande à l’homme sous forme de questions et d’affirmations, le pseudo qu’il veut et son mot de passe.

Une fois que l’application a les informations qu’elle désire, il ne lui reste plus qu’à les envoyer au serveur qui traite les informations reçues pour ensuite dire si tout est bon ou sinon quel est le problème.

On pourrait donc comparer l’application C++ à un intermédiaire, un traducteur qui parlerait le langage de l’homme et celui de la machine.

L’application c++ est développée sous Visual Studio 2008. Disons plutôt qu’une partie est développée sous Visual 2008 et une autre le sera sous Visual 2010.

Dans un premier temps nous avons codé sous QT car c’est ce qui était demandé, puis nous nous sommes rapidement tournés vers un logiciel que nous connaissions bien (Visual 2008)

Une fois une bonne partie du projet faite, nous avons réalisé qu’il nous serait impossible de continuer sans l’utilisation de *Threads. N*ous avons donc dû passer sous Visual Studio 2010

2) Programmation de l’application C++

L’application C++ devait être traitée sous QT lire « cute » mais pour des raisons techniques (pour que le serveur puisse avancer en même temps que les différents clients), j’ai préféré faire un client dit «en mode console» (donc sans interface graphique). Ceci me permettais d’avancer en même temps que le serveur et donc de tester les fonctions que je créais au fur et à mesure avec celui-ci.

Mon binôme devait en parallèle apprendre le QT et utiliser mes fonctions pour faire une application avec une interface graphique.

Le Client C++ compte deux fonctions principales. Une qui écoute et une qui lit. En fonction de ce qui a été lu, donc envoyé par le serveur, le client va lancer telle ou telle fonction.

La fonction « ecoute » est créée à l’aide de Thread, ce qui lui permet d’écouter à n’importe quel moment ce que lui dit le serveur et d’agir en conséquence.

Les problèmes rencontrés :

- QT :

Ce Client n’aura pas d’interface graphique parce que l’apprentissage de la programmation sous QT s’est avéré plus longue et plus fastidieuse que prévu.

Un autre problème de QT c’est qu’il faut joindre les fichiers .dll

- Visual Studio :

Le projet s’est déroulé comme prévu dans le planning prévisionnel jusqu'à ce que l’on bloque sur les Threads. On ne peut pas réaliser ce projet entièrement si on ne maîtrise pas complètement le multi-tâches.

1>.\main.cpp(214) : error C2673: 'main' : les fonctions globales n'ont pas de pointeurs 'this'

1>.\main.cpp(214) : error C2227: la partie gauche de '->ThreadStart' doit pointer vers un type class/struct/union/générique

1>.\main.cpp(214) : error C2065: 'myThreadDelegate' : identificateur non déclaré

1>.\main.cpp(214) : error C2673: 'main' : les fonctions globales n'ont pas de pointeurs 'this'

1>.\main.cpp(214) : error C3350: 'System::Threading::ThreadStart' : un constructeur délégué attend 2 argument(s)

1>.\main.cpp(215) : error C2673: 'main' : les fonctions globales n'ont pas de pointeurs 'this'

1>.\main.cpp(215) : error C2227: la partie gauche de '->trd' doit pointer vers un type class/struct/union/générique

1>.\main.cpp(215) : error C2065: 'myThreadDelegate' : identificateur non déclaré

3) Le dialogue avec le serveur

Le serveur envoie des messages grâce a des sokets et le Client C++ dispose d‘une fonction nommé « ecoute » qui a pour but de récupérer ces messages, de les traiter et de réagir en conséquence.

A son tour, le Client C++ envoie des messages qu’il place dans des sokets. Ces messages sont traités par le serveur qui retourne un message. Ppar exemple :

Si on envoie : CREERCOMPTE@NOMDUCOMPTE@PASSWORD

Le serveur traite l’information

Et renvoie : OK

Il y a un protocole très précis qui a été mis en place pour une bonne communication entre le serveur et les clients.

