PROGRAMMATION SYSTEME ET RESEAUX

MINI-PROJET

Résumé

Ce projet de développement a pour objectif la mise en place d'un système informatique pour la gestion d'un stock de marchandises d'un vendeur et de la préparation des factures des ventes effectuées pour chaque client. Le système doit fournir un ensemble de fonctions (Partie 1) et doit être architecturé selon le modèle client-serveur (Partie 2). Il doit gérer les informations de stock et de facturation de manière persistante (Partie 3) et doit utiliser un protocole de communication bien défini entre les clients et le serveur (Partie 4).

Table des matières

1. Fonctionnalités du système

Le système informatique doit gérer un stock de marchandises d'un vendeur et savoir combien doit payer chaque client.

1.1. Données manipulées

a) Données du stock

Le système informatique dispose d'une base de données où sont décris tous les produits commercialisés par le vendeur. Pour chaque produit, on dispose des informations suivantes :

- la référence précise du produit;
- le prix unitaire de ce produit ;
- le nombre d'exemplaires de ce produit disponible en stock

Toute cette base de données est enregistrée sous la forme d'un fichier texte **"stock.txt"** et ce selon le format suivant:

Référence Produit	Prix Unitaire	Stock

Un exemple de contenu du fichier **"stock.txt"** est :

Référence Produit	Prix Unitaire	Stock
1	150	50
2	20	15
3	45	0
4	50	200

Ce fichier doit être mis à jour à chaque fois où une commande est honorée.

b) Données des commandes des clients

Les commandes des clients sont gérées par le vendeur et sont sauvegardées dans un fichier "**histo.txt**". Ce fichier permet d'assurer la traçabilité des différentes commandes reçues. Il est structuré selon le format suivant:

Identificateur Client	Référence Produit	Valeur Commande	Résultat

Un e**xemple** du fichier "**histo.txt**" est :

Identificateur Client	Référence Produit	Valeur Commande	Résultat
1	2	10	succes
2	2	7	echec
1	1	40	succes
2	3	1	echec
3	1	10	succes
2	4	20	succes

Le stock en produit 2 contient 15 exemplaires (d'après le fichier **"stock.txt"**). Le client 1 en demande 10 exemplaires. Chose qui va être honorée puisque la valeur du stock le permet ($15 \geq 10$). Le niveau du stock en produit 2 sera donc mise à jour et aura la valeur 15-10 =5. La nouvelle commande en ce produit 2 de la part du client 2, en demande 7 exemplaires, chose qui ne peut pas être satisfaire puisque le stock ne permet pas de couvrir ce besoin ($5 < 7$).

Ce fichier "**histo.txt**" doit être mis à jour à chaque fois où une requête client est reçue.

c) Données de facturation des clients

Le système informatique dispose aussi d'un fichier qui décrit l'ensemble des factures que les clients devront payer. Ce fichier contient pour chaque client :

- L'identificateur du client ;
- La somme totale à payer par ce client

Ces factures sont sauvegardées dans un fichier texte **"facture.txt"** et ce selon le format suivant:

Identificateur Client	Somme à payer

Un exemple de contenu du fichier **"facture.txt"** est :

Identificateur Client	Somme à payer
1	6200
2	1000
3	150
4	0

Le client 1 a acheté 10 exemplaires du produit 2 (avec un P.U. de 20) et 40 exemplaires du produit 1 (avec un P.U. de 150). Ces achats lui encaisse une somme à payer égale à 10*20+ 40*150 = 6200. Le client 2 n'a acheté que 20 exemplaires du produit 4 avec un prix unitaire de 50, chose qui lui oblige de payer 20*50 =1000. Dans ce sens aussi, le client 3 va payer 10*150 = 1500. Alors que le client 4 ne va rien payer puisqu'il n'a rien acheté.

Dans le cadre de ce projet, on ne demande pas le détail des différentes factures. On ne s'occupe que par la somme totale que le client doit payer au vendeur.

1.2. Opérations possibles sur les données

Le système informatique doit permettre de réaliser les opérations suivantes :

Pour le vendeur :

- **Consulter le stock d'un produit :** en donnant la référence d'un produit, on doit pouvoir récupérer les informations le concernant (quantité disponible en stock et son prix unitaire) ;
- **Consulter les factures d'un client :** il doit être possible de voir la facture correspondant à un client en précisant son identificateur.
- **Consulter l'historique des commandes** : il doit être possible de voir le contenu de l'historique des commandes des clients.

Pour le client :

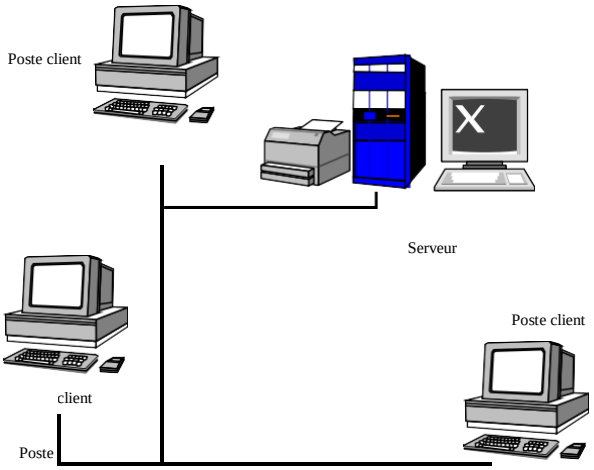
- **Acheter un produit :** un client doit pouvoir acheter un produit disponible en stock. S'il demande plus d'exemplaires alors qu'il n'y en a pas en stock, la vente doit être complètement refusée;
- **Recevoir une facture :** un client doit recevoir une facture à chaque fois qu'il achète un produit.

2. Architecture du système informatisé

Le vendeur possède plusieurs boutiques et souhaite mettre en place un système informatique composé des éléments suivants :

- un serveur central;
- des postes clients répartis dans chacune des boutiques.

Les postes clients doivent permettre, grâce à une interface homme-machine appropriée (qui sera simulée ici sous forme d'une zone de saisie en mode caractères), de réaliser les différentes opérations prévues.



Le fonctionnement sera le suivant :

1. Selon la saisie de l'utilisateur, le client préparera une requête à envoyer au serveur.
2. La requête sera envoyée au serveur et le client se mettra en attente de la réponse.
3. Le serveur réceptionnera la requête et la traitera pour comprendre la demande du client.
4. Il effectuera ensuite le traitement associé,
5. et enverra le résultat de ce traitement au client.
6. Le client réceptionnera le résultat et pourra enchaîner sur une nouvelle requête.

3. Gestion des données

Le serveur central a la charge de l'ensemble des données de la boutique (stock, commande et facturation). Les données manipulées par le serveur sont sauvegardées dans des fichiers. La concurrence d'accès au fichier exige l'emploi d'un mécanisme de synchronisation.

4. Gestion des échanges client/serveur

Les échanges entre les clients et le serveur doivent suivre un protocole bien défini pour que le serveur comprenne les requêtes des clients et pour que les clients comprennent les résultats renvoyés par le serveur. Le protocole à adopter dans ce cas de projet est TCP et ce pour assurer une communication en mode connecté et mettre en *temps réel* le contenu des différents fichiers.

5. Consignes

Le projet est travaillé et étudié en **trinôme** mais la notation est **individuelle**.

5.1. Travail de base demandé

Vous disposez d'un squelette de programme. Après lecture des différents documents, il faudra le compléter pour fournir :

1. **Aspect réseau :** Protocole TCP ; Connexion entre au moins entre deux machines physiques ;
2. **Aspect système :** un serveur parallèle. Il s'agit de modifier le serveur TCP pour qu'il puisse gérer des requêtes en parallèle. Le serveur doit donc créer un nouveau thread chaque fois qu'il reçoit une requête ;
3. **Aspect programmation :** une sauvegarde/rechargement des données dans des fichiers. Cela permet que les informations (stocks, etc.) soient persistantes et non pas limitées à la session en cours.

5.2. Travaux complémentaires

Après la réalisation du travail demandé dans la section précédente, vous pourrez choisir un ou plusieurs points présentés ici pour améliorer votre programme.

1. **Aspect réseau :** Faites en sorte que votre serveur et votre client puissent fonctionner indifféremment en TCP ou en UDP. Le protocole TCP ou UDP sera choisi selon la valeur d'un argument de la ligne de commande ;
2. **Evolution fonctionnelle :** Introduisez la notion de profil en distinguant un profil administrateur d'un profil vendeur. Désormais, l'administrateur sera le seul à avoir le droit d'effectuer une requête listant les clients et récupérant les factures.

Dans tous les cas, avant de commencer à coder ces questions, faites une analyse des problèmes que vous voulez résoudre et des solutions que vous allez proposer. Si cette analyse est bien faite, le codage sera facile. Une bonne analyse (même sans implémentation) dans le rapport et la soutenance sera fortement valorisée.

5.3. Organisation du travail

Les groupes travaillent en trinôme. La répartition des tâches entre les trois est libre. Nous vous conseillons de :

- bien respecter le cahier de charges ;
- bien gérer le temps qui vous est imparti ;
- bien discuter dans le groupe ;
- réfléchir avant de programmer.

Pour la réalisation logicielle, nous conseillons vivement de suivre les étapes suivantes :

1. définir les fichiers *.txt;
2. réalisation d'une communication en TCP de chaque côté (client et serveur) ;
3. réalisation des opérations possibles de chaque côté (client et serveur) ;
4. ajout du mode parallèle (thread) ;
5. ajout de la synchronisation ;
6. ajout des fonctionnalités supplémentaires.

Note : certaines étapes peuvent être faites en parallèle. Pensez à vous répartir le travail.

5.4. Évaluation du travail

a) Soutenance

Chaque groupe aura 15 minutes pour présenter son travail. Le groupe expliquera brièvement la répartition du travail entre ses membres, l'avancement du projet et les problèmes rencontrés. Des questions seront posées et la démonstration est évidemment demandée.

b) Rapport

Le groupe doit rendre juste avant la soutenance un rapport de 10 pages au maximum donnant :

- l'organisation du travail dans le groupe ;
- la méthodologie utilisée dans le développement ;
- la pertinence de certains choix ;
- l'état courant du projet ;
- les difficultés rencontrées ;
- un rapide bilan de ce que vous a apporté ce projet.

Il s'agit de mettre en valeur la qualité de votre travail à l'aide d'un rapport. Pour cela le rapport doit explicitement faire le point sur les fonctionnalités du logiciel (lister les objectifs atteints, lister ce qui ne fonctionne pas et expliquer - autant que possible - pourquoi).

Ensuite le rapport doit mettre en valeur le travail réalisé sans paraphraser le code, bien au contraire : il s'agit de rendre explicite ce que ne montre pas le code, de démontrer que le code produit a fait l'objet d'un travail réfléchi et même parfois minutieux. Par exemple, on pourra évoquer comment vous avez su résoudre un bug, comment vous avez su éviter/éliminer des redondances dans votre code, comment vous avez su contourner une difficulté technique ou encore expliquer vos choix, pourquoi certaines pistes examinées voire réalisées ont été abandonnées.

Il s'agira aussi de bien préciser l'origine de tout texte ou toute portion de code empruntée (sur internet, par exemple) ou réalisée en collaboration avec tout autre groupe. Il est évident que tout manque de sincérité sera lourdement sanctionné.

Le rapport ne doit pas redonner des informations présentes dans ce document.

c) Code source

Vous devrez également fournir l'intégralité de votre code source en **Python** sous le format d'un dossier compressé qui doit avoir un nom selon le modèle suivant :

Groupe_PrénomNom1_ PrénomNom2_ PrénomNom3

Le groupe indique votre section : A, B, C, D ou E.

N'oubliez pas de commenter vos programmes. L'archive doit être nettoyée i.e ne doit pas contenir des fichiers objets ou des exécutables.
Le dossier est à charger sur un lien **Filepiper** qui vous sera communiqué. Le mot de passe est **enicarthage**.