Simulation du fonctionnement d'un bureau d'accueil

1- Cahier des charges

Nous voulons simuler le déroulement d'une journée du **bureau d'accueil** d'une entreprise.

Ce bureau d'accueil reçoit tous les jours la visite d'un certain nombre de **clients** à son guichet. Ces clients viennent tous **déposer un paquet** et **prendre possession d'un colis**.

Le personnel du bureau est composé de 2 personnes que nous nommerons A et B. Le bureau ne sert un nouveau client que lorsque le client actuellement présent au guichet a été servi par les deux personnes de l'accueil.

La personne A est chargée de récupérer le paquet apporté par chaque client et de le déposer dans une **aire de stockage** dont le volume est prévu pour accueillir au maximum **P** paquets. De plus, les paquets étant très lourds, il est nécessaire pour les acheminer vers l'aire de stockage d'utiliser un **chariot**.

La personne B est, quant à elle, chargée de donner au client un colis, chaque client ayant droit au même type de colis. Les colis (tous identiques) sont déposés au fur et à mesure de leur livraison par un **transporteur** (qui les livre un par un) dans une **aire de stockage** prévue pour contenir un maximum de **C** colis à un instant donné. La personne B va donc retirer un colis de la zone de stockage pour le remettre au client présent au guichet du bureau d'accueil.

Mais les colis étant très lourds il est nécessaire pour les acheminer vers le guichet d'utiliser un **chariot**.
Or, le bureau d'accueil ne dispose que d'un seul et unique chariot que ses personnels doivent se partager pour exécuter leur mission.

Lorsque l'aire de stockage des paquets est pleine, le bureau d'accueil est obligé de fermer et il n'ouvre à nouveau ses portes que le lendemain matin.

Les valeurs **P** et **C** seront respectivement fixées à 5 et 3.

2- Implémentation

Environnement de développement

L'application sera réalisée en **C** sous **UNIX**. Elle devra tourner sous Solaris 9. Chaque tâche sera réalisée par un **processus** différent, la synchronisation et la communication entre ceux-ci s'effectueront à l'aide des mécanismes UNIX, en particulier les **IPC System V**.

Tâches de l'application

L'arrivée au guichet d'un nouveau client sera simulé par une tâche.

Chaque personne du bureau sera mise en œuvre par une tâche distincte.

Une autre tâche sera chargée de simuler le transporteur faisant la livraison des colis.

L'application sera donc composée des 5 tâches suivantes:

- **tâche principale:** simule l'ouverture et la fermeture du bureau
- **client**: simule l'arrivée d'un nouveau client
- **personneA**: simule le travail de la personne A
- **personneB**: simule le travail de la personne B
- **livreur**: simule le travail du livreur de colis

Supervision du déroulement

Le déroulement de l'application sera supervisé par des affichages effectués par chaque tâche:

client: - affiche le numéro du client arrivant au guichet

personneA: - affiche le numéro du client qu'elle sert
- affiche le moment où elle prend possession du chariot
- affiche le moment où elle libère le chariot
- affiche si l'aire de stockage des paquets est arrivée à saturation

personneB: - affiche le numéro du client qu'elle sert
- affiche le moment où elle prend possession du chariot
- affiche le moment où elle libère le chariot
- affiche le numéro du colis qu'elle remet au client

livreur: - affiche le numéro du colis déposé dans la zone de stockage des colis
- affiche si la zone de stockage des colis est temporairement pleine
- affiche qu'il peut à nouveau déposer un colis après que la zone ait été temporairement saturée

tâche principale: - affiche lors de la terminaison de l'application la liste des paquets et des colis présents dans les zones de stockage

Synchronisation et communication

L'arrivée d'un nouveau client est signalé par la tâche **client** à la tâche **personneA** et à la tâche **personneB**. La tâche **client** ne fait arriver un nouveau client que lorsque les tâches **personneA** et **personneB** ont fini de servir le client précédent. Il faut bien sûr attendre que les 2 tâches aient toutes deux fini de servir un client avant de commencer à s'occuper du suivant.

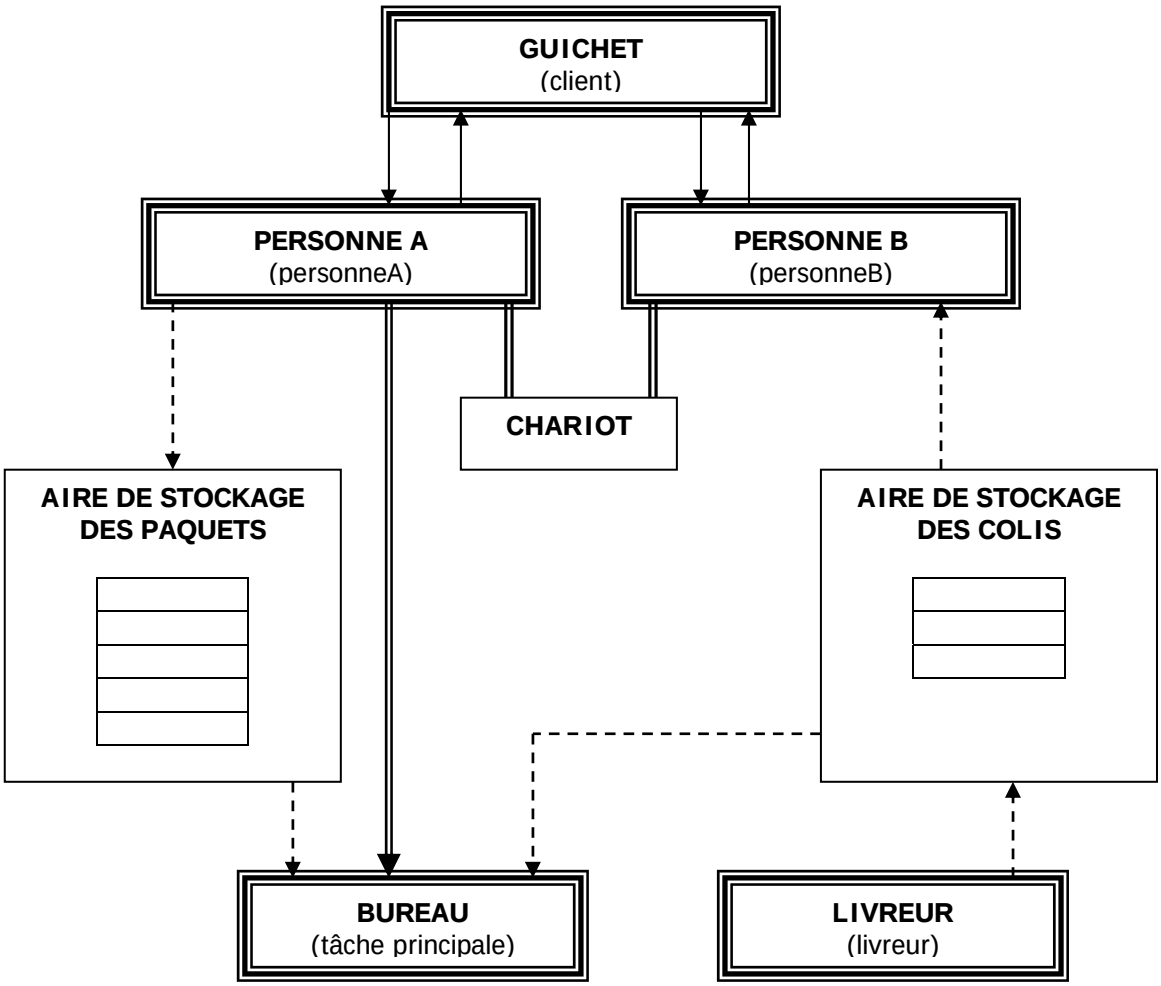
Lorsque la tâche **personneA** est avertie de la présence d'un nouveau client, elle cherche le chariot nécessaire pour transporter le paquet vers la zone de stockage. S'il est disponible, elle transporte le paquet vers la zone de stockage où elle le dépose s'il reste de la place. Une fois le paquet déposé, elle rend le chariot. Si la zone de stockage est saturée, elle ne peut donc pas poser son paquet et elle signale la nécessité de fermer le bureau à la **tâche principale**. Celle-ci effectue alors la fermeture du bureau qui consiste en l'affichage du contenu des zones de stockage puis en la terminaison de l'application, c'est à dire la fin de toutes les tâches avec libération de toutes les ressources systèmes utilisées.

Lorsque la tâche **personneB** est avertie de la présence d'un nouveau client, elle doit aller chercher un colis dans la zone de stockage pour le remettre au client. Pour ce faire, elle doit prendre le chariot qu'elle rendra après avoir transporté le colis.

La tâche **livreur** dépose en continu des colis dans la zone de stockage. Il ne peut bien sûr en déposer quand la zone est pleine. Dans ce cas, il attendra qu'une place s'y libère pour le déposer.

Les **zones de stockage** seront représentées par des structures de file d'attente où seront écrits les numéros des paquets (=numéro du client) et des colis.

Schéma général



3- Travail demandé

Ce travail doit s'inscrire dans une démarche projet. Comme tout projet, il doit comporter une phase d'**analyse** (qu'est ce que fait l'application?), de **conception** (comment va on traiter techniquement le problème?), de **réalisation** (codage) et de **tests** (permettant de vérifier que l'application fonctionne dans le cas nominal, les cas aux limites et les cas d'erreur).

Document à produire

Un rapport écrit (forme électronique ou manuscrite acceptées) doit rendre compte des phases du projet précédemment citées.

Son plan idéal est donc le suivant:

- **analyse du problème**: pas trop compliqué ici, tout est dans le sujet…
- **étude technique** (= document de conception): doit détailler l'architecture de l'application, décrire les choix techniques effectués (en particulier les moyens de synchronisation et de communication) en les justifiant et en discutant leurs limites, et comporter l'algorithme de chaque tâche. N'oubliez pas de traiter en particulier de la gestion des erreurs et des phases de lancement et de terminaison de l'application. Cette étude technique doit pouvoir être fournie à un programmeur et lui permettre de coder l'application sans information complémentaire.
- **dossier de tests**: celui-ci doit être écrit avant de commencer le code. Il est aussi fourni au programmeur.
- **résultats des tests**: votre code fonctionne-t-il? Y a t-il des problèmes non résolus? D'où proviennent t-ils?

Les trois premiers items sont à rédiger avant de coder…

Code de l'application

Le code C de l'application doit être remis. Il doit être lisible et commenté.