UNIVERSITE PIERRE & MARIE CURIE
LA SCIENCE A PARIS

Maîtrise d'informatique 2003-2004
Algorithmique et Programmation
Tous documents autorisés
Durée 2h30

Module Programmation Concurrente, Réactive, Répartie
Devoir sur table décembre 2003

Les exercices sont à rédiger sur trois copies séparées

Exercice 1 *Objective Caml[6 pts]*

Nous voulons simuler la prise en charge pour l'atterrissage des avions d'un aéroport :

- Le centre de contrôle avec un canal universel de communication `cuc` pour le premier contact avec un avion en approche,
- Une seule piste d'atterrissage,
- Les avions qui atterrissent.

Il est possible que plusieurs avions, pour des raisons diverses, arrivent en même temps à l'aéroport.
Le travail du centre de contrôle est de permettre l'atterrissage d'un avion à la fois en donnant l'ordre aux autres avions de faire des boucles dans le ciel à des altitudes différentes de sécurité présentées sous forme de couches. Un avion ne quitte la couche qui lui est attribuée pour aller vers une autre couche ou atterrir que s'il a reçu l'ordre du centre de contrôle. On suppose qu'il y a 5 couches.
Nous proposons ici un scénario simple (voir simpliste).

Scénario côté avion :

a) L'avion est en approche de l'aéroport.
b) En premier, il prend contact en envoyant au centre un message de valeur `cuc` sur le canal `cuc` pour lui demander un canal privé de communication différent de tous les autres canaux en cours de fonctionnement.
c) Une fois le canal privé `Y` reçu sur `cuc`, le pilote communique sur `Y` avec le centre jusqu'à l'arrêt au sol de l'avion.
d) Sur `Y`, le pilote envoie au centre un message `"PRET"` pour dire qu'il est prêt.
e) Il attend un ordre du centre.
f) Il reçoit un ordre, envoie au centre un message `"REÇU"` de confirmation de la réception et il l'exécute.
g) Ca prend un certain temps.
h) Une fois l'ordre totalement exécuté, il envoie au centre un message `"PRET"` pour dire qu'il est prêt de nouveau.
i) Il répète l'étape e) jusqu'à l'atterrissage et l'arrêt complet de l'avion.
j) Et il envoie un dernier message `"FIN"` au centre (à la place du `"PRET"`).

Scénario côté centre :

a) Il se met en attente d'une demande de création d'un canal privé sur le `cuc`. Et en même temps, il surveille les autres avions et leur donne des ordres de trafics.

b) Si sur le `cuc` une demande de création d'un canal privé arrive, il créée un canal privé `cp` et l'envoie par le `cuc`.
c) Il attend un message `"PRET"` sur le `cp`.
d) Une fois le `"PRET"` reçu, il peut commencer à envoyer des ordres.
e) A chaque ordre donné, il attend le message "`REÇU`" pour confirmer la réception et après, le message `PRET` pour confirmer de la fin de l'exécution de l'ordre.
f) Les ordres sont "`COUCHE1`", "`COUCHE2`",…, "`COUCHE5`" et "`ATTERRISSAGE`".
g) Pour simplifier, on suppose qu'un avion passe toujours d'une couche supérieure $COUCHE_s$ vers une couche inférieure $COUCHE_i$ (avec i < s).
h) Pour un avion donné, le dernier ordre est "`ATTERRISSAGE`". Le centre attend alors un message `RECU`, puis un dernier message `FIN`. Il se libèrera alors de l'avion.

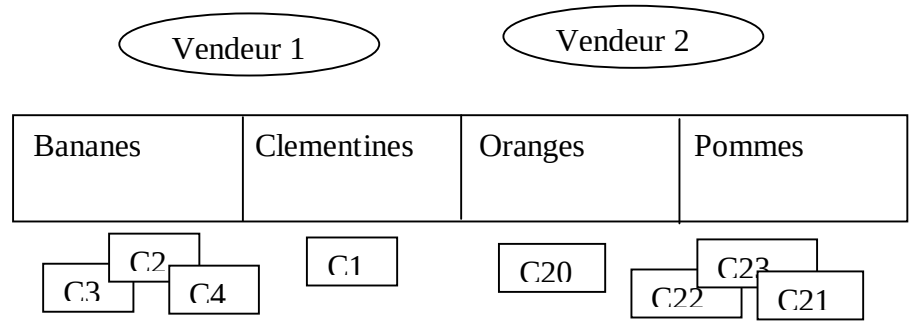
On donne les définitions de variable suivantes :
```
let cuc = Event.new_channel();;
let canaux_prives = Array.create 10 (Event.new_channel());;
```

1) Ecrire la fonction `controler` pour contrôler les avions en approche.

2) Ecrire la fonction `piloter` pour piloter l'avion.

Exercice 2 *Threads concurrents et synchronisation, à traiter en Java **ou** en ADA [9 pts]*
Cet exercice est à traiter une seule fois ***en Ada OU en Java***. Pour vous aider, les énoncés sont spécialisés pour ces deux langages, il est conseillé de n'en lire qu'un suivant vos préférences.

Enoncé ADA
On veut simuler le marchand de fruits du marché, avec son étalage et deux vendeurs.



Chaque vendeur peut servir tous les fruits; cependant, pour ne pas se bousculer, un seul à la fois peut se servir dans un bac donné : si le vendeur 1 prend des clementines, le second attend qu'il ait fini pour prendre des clémentines (mais il peut servir des bananes). Chaque vendeur dispose de sa caisse propre.
Les clients s'adressent à un vendeur particulier dès qu'ils arrivent; le dessin suggère que, pendant qu'un client fait ses achats, les suivants attendent ensemble, il n'y a pas de file ordonnée, le prochain servi sera fixé au hasard (cette hypothèse est destinée à faciliter votre programmation, elle évite de gérer des files d'attente explicites : ne pas chercher à compliquer !).

L'étalage contient initialement 20 kilos de chaque fruit et est mis à jour lors de chaque prélèvement dans un bac. Lorsque le vendeur reçoit moins que ce qu'il a demandé (0 kilo pour 1 demandé, 1 pour 2 demandés..), il va chercher un cageot du fruit manquant dans le camion - action simulée par un affichage suivi d'un délai - et l'ajoute; le camion est supposé suffisamment chargé pour la durée du marché.
Les fruits sont tous au même prix : 2 euros le kilo.

1) L'étalage
a)Définir la spécification nécessaire pour permettre le partage de l'étalage et sa mise à jour fruit par fruit, aussi bien pour retirer la commande d'un client que pour ajouter un cageot de10 kilos.
b)Ecrire le corps : vous pouvez écrire complètement la mise à jour d'un seul fruit, représentant les autres.

Le tableau suivant présente côte à côte les séquences respectives du processus (tâche) vendeur et du processus Client. Ceci n'est qu'indicatif, votre solution peut s'en écarter, en restant une cohérente.

Le vendeur répète la séquence suivante :	Le client déroule la séquence suivante :
attendre l'arrivée d'un client	signaler son arrivée à un vendeur
dire bonjour à ce client	attendre son bonjour
initialiser le compte	répondre
boucle	tirer variable aléatoire pour banane
attendre commande du client	suivant valeur commander 1, 2 kilos
ou demande de compte	de bananes ou pas de commande
s'il s'agit d'une commande	si commande
prendre le fruit correspondant	attendre le sachet de fruits
si quantité insuffisante	
aller chercher cageot	… idem pour les autres fruits
prendre complement	
donner sachet au client	
ajouter le prix au compte	
sinon quitter boucle	
fin boucle	demander compte
donner compte	attendre compte
attendre paiement	payer
rendre monnaie et /ou	attendre monnaie et/ou
dire au revoir au client	dire au revoir au vendeur

2) En déduire la spécification des processus Vendeur et Client.

Remarque : La question 5 peut-être traitée dès maintenant, en utilisant Vendeur et Client.

3) Ecrire le corps du processus Vendeur.

4) Ecrire le corps du processus Client. On suppose qu'on dispose d'une procédure alea qui renvoie une variable flottante entre 0 et 1 pour décider si on commande tel fruit et combien on en commande (seuil 0.7 par exemple pour commander 1 kilo, 0.9 pour commander 2 kilos).

5) Ecrire le programme principal qui

crée et initialise l'étalage avec 20 kilos de chaque fruit

crée et initialise les deux vendeurs

crée et initialise une vingtaine de clients (n'arrivant pas tous en même temps !) pour chaque vendeur

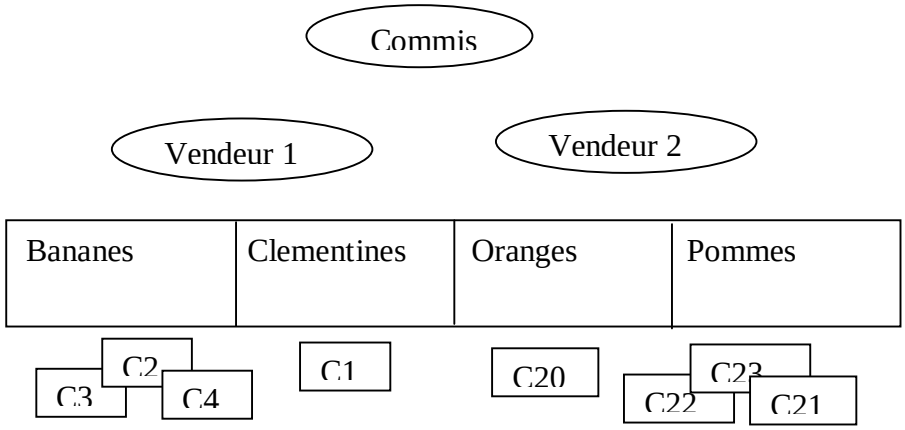
6) Compléments

a) Que se passe-t-il lorsqu'il reste 1 kilo d'oranges et que les deux vendeurs en demandent chacun 2?

b)Que suggérez vous pour terminer les processus vendeurs à la fin du marché ?

Enoncé Java

On veut simuler le marchand de fruits du marché, avec son étalage, deux vendeurs et un commis.



Chaque vendeur peut servir tous les fruits; cependant, pour ne pas se bousculer, un seul à la fois peut se servir dans un bac donné : si le vendeur 1 prend des clémentines, le second attend qu'il ait fini pour prendre des clémentines (mais il peut servir des bananes). Chaque vendeur dispose de sa propre caisse.

Les clients s'adresse à un vendeur particulier dès qu'ils arrivent; le dessin suggère que, pendant qu'un client fait ses achats, les suivants attendent ensemble, il n'y a pas de file ordonnée, le prochain servi sera fixé au hasard (cette hypothèse est destinée à faciliter votre programmation, elle évite de gérer des files d'attente explicites : ne pas chercher à compliquer !).

L'étalage est constitué de quatre bacs à fruits. Chaque bac est initialement pourvu de 20 kilos, et est mis à jour lors de chaque prélèvement. Lorsque le vendeur prélève moins que ce que lui a demandé le client, il se plaint du fait que le commis ne fait pas son travail assez vite, et attend (ainsi que son client) que le bac soit à nouveau rempli.

Le commis surveille en permanence l'état de l'étalage, et veille à ce que chaque bac soit rempli. Quand il détecte qu'un bac est vide, il l'approvisionne avec un cageot de 10 kilos.

Les fruits sont tous au même prix : 2 euros le kilo.

1) Définir la classe Bac et la classe Etalage (les bananes sont dans le bac 1, les clémentines dans le bac 2, etc...).

Le tableau suivant présente la forme de la transaction entre le client et le vendeur. Les clients guident les transactions : ce sont des Threads. Les vendeurs, eux, ne font que réagir aux demandes des clients : ce sont des objets classiques.

Le vendeur répète la séquence suivante :	Le client déroule la séquence suivante :
attendre l'arrivée d'un client repondre bonjour à ce client initialiser le compte servir chaque commande donner compte attendre paiement rendre monnaie et /ou dire au revoir au client	signaler son arrivée à un vendeur dire bonjour tirer variable aléatoire pour banane suivant valeur commander 1, 2 kilos de bananes ou pas de commande si commande attendre le sachet de fruits … idem pour les autres fruits demander compte attendre compte payer attendre monnaie et/ou dire au revoir au vendeur

2) En analysant les interactions entre le client et le vendeur pour chaque transaction, donnez l'ensemble des méthodes de la classe Vendeur.

On rappelle que pour obtenir un entier aléatoire entre 0 et n-1, on utilise r.nextInt(n), où r est de instance de java.util.Random.

3) Ecrire la classe Client, de sorte que les clients choisissent au hasard leur vendeur et déroulent le scénario de la transaction.

4) Ecrire les classes Vendeur et Commis, et expliquer votre synchronisation concernant l'accès aux bacs.

5) Ecrire un programme principal dans une classe MarcheAuxFruits, avec quelques clients.

6) En l'état, le commis ne termine jamais. Proposer une solution pour la terminaison du programme.

Exercice 3 *Threads concurrents et synchronisation en Esterel [5 pts]*
On veut simuler le rayon 'fruits' d'une supérette: on suppose qu'il y a trois étalages de fruits (Bananes, Pommes, Oranges), initialement fournis chacun avec 10 kilos de fruits et au plus deux clients qui se servent simultanément dans le même bac. Un gérant automatique de chaque étalage connaît à chaque instant la quantité disponible de chaque fruit, il doit prévenir un employé lorsqu'il reste moins de 2 kilos d'un fruit pour qu'un cageot soit ajouté et éviter ainsi la rupture de stock.

Le signal d'entrée B1, O1, ou P1 (resp. B2, O2, ou P2) indique que le client 1 (resp client 2) souhaite se servir un kilo du fruit B, O ou P. Bien sur, un client ne peut se servir que d'une sorte de fruit à la fois.
Un instant plus tard, le client dispose de son kilo de fruit il émet l'un des signaux valués B(1), O(1), P(1) (destiné au gérant automatique); bien sur, les clients peuvent émettre ce signal au même instant.
Le gérant des bananes émet le signal MB pour indiquer qu'il risque de manquer de bananes lorsqu'il en reste 2 kilos ou moins, celui des oranges émet MO, celui des pommes MP dans les mêmes circonstances. Plusieurs de ces signaux peuvent être émis au même instant. Il reçoit un signal interne (ou déclaré output pour être observable en phase de test) AB (resp. AO ou AP) lorsque 10 kilos de bananes (resp. oranges, pommes) ont été ajoutés à l'étalage. Ces signaux sont émis par l'employé lorsqu'il a rechargé l'étalage.

1) Déclarations et structure générale
a) Déclarer les signaux nécessaires.
b) Indiquer l'organisation générale du programme : montrer tous les blocs parallèles (en indiquant leur rôle par une ligne de commentaire).

2) Les clients (en supposant qu'un étalage n'est jamais vide)
a) Ecrire les deux clients
b) Quelles sorties obtient-on avec la séquence d'ensembles d'entrées O1;B1 O2; P1; ?
c) Proposer une séquence d'ensembles d'entrées en incluant des instants où aucun client n'est présent au rayon fruit.

3) Les gérants
a) Déclarer les variables nécessaires.
b) Ecrire le gérant des oranges.
c) Proposer une séquence d'ensembles d'entrées menant à l'émission des signaux MO puis AO

4) L'employé, récepteur des signaux MB, MO, MP et émetteur de AB, AO, AP
a) Ecrire l'employé, en supposant qu'il peut recharger plusieurs étalages dans le même instant.
b) Si on lève cette hypothèse, un étalage peut se trouver vide ; quelles modifications proposez-vous ?

5) Les gérants sont complètement superposables sauf par le nom des signaux qu'ils échangent avec l'extérieur. Proposer une écriture du module gérant et trois instanciations dans le programme initial.