GIS-IMA 3ème année
Devoir Surveillé Architectures Logicielles

© Olivier Caron

Durée 2 heures, tous documents autorisés

1 Programmation par composants Java Beans (8 points)

Dans notre monde économique actuel, le prix d'un produit à l'étalage est sensiblement différent du prix originel. Cela est dû souvent à la multitude d'intermédiaires qui appliquent, à chaque étape, un *surcoût* à ce prix (le contraire de la notion de commerce équitable...).

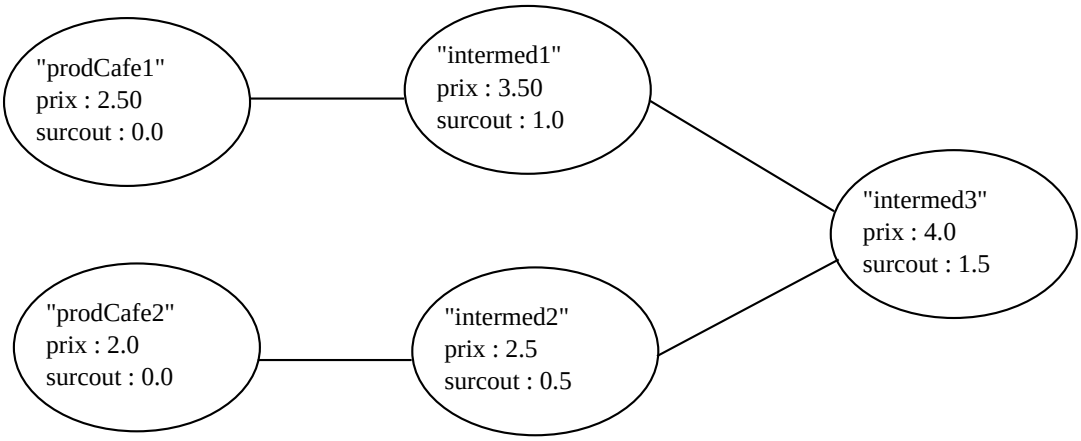

FIG. 1 – Evolution du prix d'un produit

Par exemple, la figure 1 illustre un graphe de fournisseurs partant de deux producteurs (qui sont considérés aussi comme des fournisseurs). Un fournisseur peut fournir plusieurs autres fournisseurs (non montré dans cet exemple) et un fournisseur peut choisir de s'approvisionner parmi plusieurs fournisseurs (ex : le cas de "intermed3"). Dans ce cas, seul le fournisseur proposant le prix le plus bas approvisionne l'intermédiaire (ex : le cas de "intermed2" qui est sélectionné au détriment de "intermed1"). Les prix de chaque intermédiaire peuvent être calculés automatiquement : c'est le prix le plus bas d'un des fournisseurs auquel on ajoute le surcoût.
On désire simuler (de manière simpliste), cette chaîne partant du producteur jusqu'aux derniers intermédiaires et mesurer ainsi automatiquement l'impact de la modification du prix initial sur les prix finaux.
Pour cela, nous développons le composant Java Beans `Fournisseur` qui dispose des caractéristiques suivantes :
– une propriété *nom* de type `String` pour désigner le fournisseur
– une propriété booléenne *intermédiaire*, pour distinguer le producteur d'un intermédiaire
– une propriété *surcout* de type `double` pour connaître le surcoût appliqué au prix de l'intermédiaire précédent (vaut toujours zéro pour un producteur)
– une propriété liée `prix` de type `double`. Cette propriété liée a pour but de signaler la modification du prix.

Question 1 : écrire le composant `Fournisseur`.
Question 2 : écrire l'extrait de code JAVA qui, pour des variables f1, f2, f3, f4, f5 de type `Fournisseur` configure ces objets **déjà instanciés** pour initialiser l'exemple décrit dans la figure 1

2 Architecture logicielle distribuée (12 points)

On désire concevoir une architecture client/serveur multi-sites qui permet de gérer une chaîne d'hôtels. C'est à la charge de chaque hôtel de gérer localement la réservation de ses chambres. Chaque hôtel dispose d'un nom unique et doit permettre la réservation d'une chambre à une date donnée, indiquer le nombre de chambres de l'hôtel, le nombre de chambres occupées à une date donnée, le prix d'une chambre (pour simplifier, on considère qu'il n'y a qu'un seul type de chambre).

Le siège central de la chaîne d'hôtels connaît bien sûr l'ensemble de ses hôtels et effectue des statistiques globales sur son réseau notamment le taux d'occupation global pour une date donnée (rapport nombre de chambres occupées/nombre de chambres pour l'ensemble des hôtels).

Un client peut se déplacer dans un hôtel pour demander une réservation dans cet hôtel. Un serveur web unique existe aussi sur le réseau pour la chaîne d'hôtels et permet la réservation et la consultation d'une chambre d'un des hôtels de la chaîne.

Pour la suite, on utilisera la technologie J2EE-EJB pour mettre en place cette architecture logicielle, on ne se préoccupera pas ni des composants web, ni de la manière dont on gère la persistance des informations mais uniquement des fabriques et interfaces distantes des composants EJB.

Question 3 : Proposez une architecture logicielle qui répond à ce problème. Décrire le rôle de chaque application et composant EJB introduits. Détaillez comment ces composants s'articulent entre eux (vous pouvez donner des exemples). Décrire également par quel moyen, il est possible d'accéder à tel composant. Décrire pour chaque composant introduit, les éléments accessibles (fabrique et interface). Utilisez la norme EJB pour spécifier ces interfaces Java (ne pas mettre les lignes import).

Question 4 : Ecrire le code Java qui permet de connaître le taux d'occupation global pour une date donnée de l'ensemble des hôtels.