



IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

TD8-TP7 – Le repas des philosophes

Fondements théoriques du développement des logiciels concurrents

Objectifs

Au cours de ces deux séances, nous allons découvrir par la pratique la notion d'interblocage (ou « étreinte fatale »), en particulier :

- modéliser un problème célèbre en FSP,
- tenter de trouver des solutions aussi élégantes que possible.

Puis, nous allons sur cet exemple découvrir le passage d'un modèle au code Java correspondant.

Le repas des philosophes

Un groupe de philosophes se retrouvent pour un repas. Ils sont assis autour d'une table ronde, devant un bon plat de *spaghetti*. Une fourchette est placée entre chaque paire d'assiettes, ainsi, un philosophe a toujours une fourchette à sa gauche et une fourchette à sa droite. La configuration de la table est représentée en figure 1.

Une fois assis, un philosophe a besoin de deux fourchettes pour manger, puis il relâche ces deux fourchettes dès que lui prend une irrésistible envie de réfléchir.

Exercice 1 (*Un modèle pour les philosophes*)

▷ Question 1.1 :

Proposer dans un premier temps une modélisation « intuitive » de ce repas, puis déterminer une exécution conduisant à une situation de *deadlock*.

▷ Question 1.2 :

Dans un second temps, modifier votre modèle afin qu'aucune situation d'interblocage ne puisse survenir.

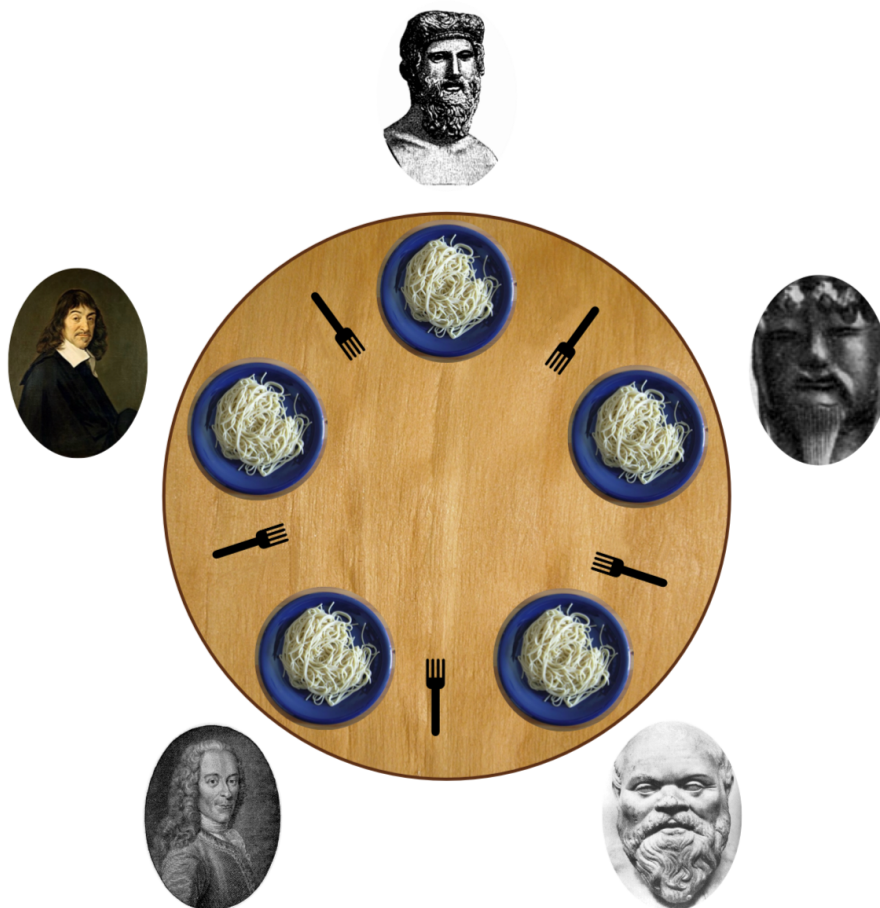


FIGURE 1 – repas des philosophes

Les philosophes en Java

Avant de réaliser cet exercice, il sera utile de relire les notes de cours sur le passage d'un modèle FSP au code Java qui lui correspond.

Pour faciliter la visualisation de la table de philosophes, vous trouverez sur moodle, un projet eclipse contenant deux classes pour construire une interface graphique à notre dîner. Le diagramme de classe est présenté figure 2. Une classe `Diner` permet d'initialiser l'application et son interface graphique. Elle est incomplète car il manque l'initialisation des fourchettes et des philosophes. Une classe `PhiloView` est en charge de l'affichage des philosophes et des fourchettes suivant leur état. Pour cela, elle définit les états des philosophes (définis par des constantes `THINKING`, ...). Une image spécifique est associée à chaque état. Pour gérer la mise à jour de l'interface graphique, vous devrez utiliser les méthodes `void setFork(int id, boolean taken)` et `void setPhil(int id, int s)` qui prennent toutes deux en paramètre le numéro de l'élément à mettre à jour (fourchette ou philosophe) ainsi que son nouvel état.

Remarque :

Attention, la classe `PhiloView` impose une numérotation spécifique des fourchettes. Un philosophe de numéro i a la fourchette de numéro i à sa droite et la fourchette de numéro $i-1$ à sa gauche.

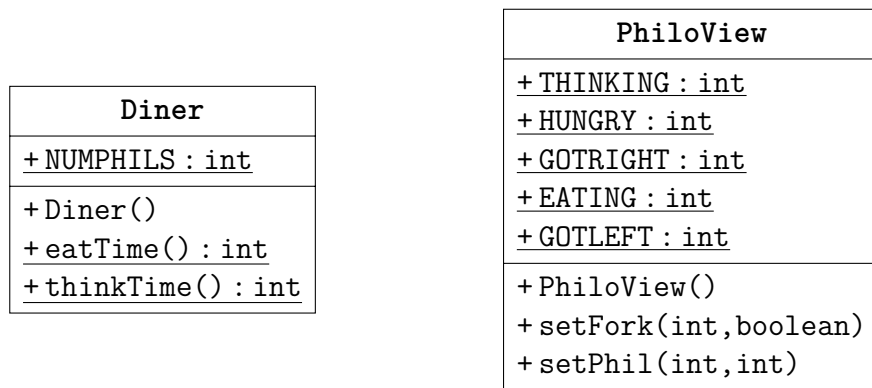


FIGURE 2 – Diagramme de classes fournies.

Exercice 2 (*Les philosophes en Java*)

- ▷ Reprendre l'exercice précédent et proposer une réalisation en Java.