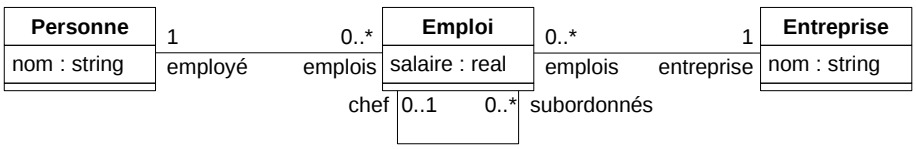
Delphine Longuet, Thibaut Balabonski, Robin Pelle
longuet@lri.fr, blsk@lri.fr, pelle@lri.fr

TD 2 - Spécification des opérations

Semaine du 26 septembre 2016

Exercice 1

On considère le diagramme de classes suivant, qui modélise les différents emplois occupés par des personnes dans des entreprises. Un emploi est caractérisé par son salaire et associe de manière unique une personne et une entreprise. Une personne ne peut pas occuper plusieurs emplois dans la même entreprise mais peut par contre travailler dans plusieurs entreprises. Il peut exister une relation de hiérarchie entre les employés d'une même entreprise, modélisée par l'association réflexive sur la classe Emploi. Une personne reçoit toujours un salaire strictement inférieur à son supérieur. Pour simplifier, les personnes et les entreprises sont caractérisées de manière unique par leur nom.



On associe à ce diagramme de classes les invariants suivants :

(a) $\forall p1, p2 \in \mathsf{Personne}, p1 \neq p2 \Rightarrow p1.\mathsf{nom} \neq p2.\mathsf{nom}$

(b) $\forall p \in \mathsf{Personne}, |p.\mathsf{emplois}| = |p.\mathsf{emplois.entreprise}|$

(c) $\forall e1, e2 \in \mathsf{Entreprise}, e1 \neq e2 \Rightarrow e1.\mathsf{nom} \neq e2.\mathsf{nom}$

(d) $\forall e \in \mathsf{Emploi}, e.\mathsf{salaire} > 0$

(e) $\forall e \in \mathsf{Emploi}, e.\mathsf{chef} \neq \emptyset \Rightarrow e.\mathsf{salaire} < e.\mathsf{chef.salaire}$

(f) $\forall e \in \mathsf{Emploi}, e.\mathsf{chef} \neq \emptyset \Rightarrow e.\mathsf{entreprise} = e.\mathsf{chef.entreprise}$

On veut spécifier les opérations suivantes :

1. Personne :: collegues(ent : Entreprise) : $\mathcal{P}$(Personne) construit l'ensemble des personnes travaillant dans la même entreprise que la personne. Il faut que cette personne ait un emploi dans l'entreprise passée en argument. Une personne ne fait pas partie de l'ensemble de ses collègues.
   ou bien : On peut toujours appeler l'opération, l'ensemble est vide si la personne ne travaille pas dans l'entreprise passée en argument.
2. Emploi :: superieurs() : $\mathcal{P}$(Emploi) construit l'ensemble des emplois placés hiérarchiquement au-dessus de l'emploi concerné.
3. Emploi :: augmenter(montant : int) permet d'augmenter le salaire d'un emploi du montant passé en argument. Il faut que les invariants liés à la hiérarchie dans l'entreprise soient toujours vérifiés après l'augmentation.

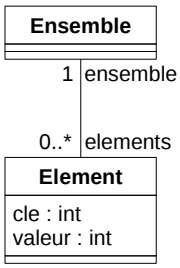
4. Emploi :: diriger(emp : Emploi) permet de devenir le supérieur de l'emploi passé en argument. Il faut que cette personne n'ait pas déjà un supérieur et que la contrainte sur les salaires soit respectée.
5. Entreprise :: embaucher(nom : string, salaire : int) permet d'embaucher une personne avec le salaire donné. Il faut que cette personne existe et n'appartienne pas déjà aux employés de l'entreprise. L'emploi doit être créé lors de l'opération.
6. Entreprise :: licencier(nom : string) de licencier une personne. Il faut que cette personne soit un employé de l'entreprise. Tous ces liens avec l'entreprise et ses collègues doivent être supprimés.

Exercice 2
On considère le diagramme de classes du système de gestion des comptes bancaires du TD1 et la spécification suivante de l'opération retirer :

Compte :: retirer(montant : real) : boolean
pre : $montant > 0 \wedge this.\text{solde} - montant >= 0$
post : $this.\text{solde} = old(this.\text{solde}) - montant$

Spécifier l'opération retirer redéfinie dans la classe Compte_cheque. On rappelle que des frais de découvert sont ajoutés à toute opération dont le résultat laisse le solde du compte en-dessous du découvert autorisé.

Exercice 3
On considère une classe Ensemble dont les éléments sont des couples formés d'une clé et d'une valeur. Les clés et les valeurs sont toutes positives strictement et les clés sont deux à deux distinctes. L'opération chercher prend un entier en argument, renvoie la valeur correspondant à cette clé si elle existe dans l'ensemble et -1 sinon.
On a écrit une première spécification de l'opération chercher :



Ensemble :: chercher(clé : int) : int
pre : $true$
post : $(result >= 0 \wedge \exists e \in \text{Element}, e.\text{clé} = \text{clé}) \vee result = -1$

Cette spécification est-elle correcte ? Est-elle complète ? Quelles implémentations de l'opération chercher sont autorisées par cette spécification ? Corriger la spécification en s'assurant que seules les implémentations voulues sont autorisées.

Exercice 4
L'opération sqrt calcule la racine carrée entière inférieure de son argument entier. On donne la spécification suivante de cette opération :

sqrt(a : int) : int
pre : $a > 0$
post : $result * result = a$

Cette spécification est-elle correcte ? Est-elle complète ? Quelles implémentations de l'opération sqrt sont autorisées par cette spécification ? Corriger la spécification en s'assurant que toutes les implémentations voulues sont autorisées.