Modélisation en UML d'un programme Java

Il s'agit de proposer un modèle UML d'un programme java pour gérer les salaires des employés d'une entreprise.
Un employé est caractérisé par son nom, son prénom, son âge, il est modélisé par une classe `Employe` dotée des attributs nécessaires, d'une méthode abstraite `calculerSalaire` (voir le mode de calcul ci-dessous) et d'une méthode `toString` retournant une chaîne de caractère obtenue en concaténant la chaîne de caractères "`L'employé `" avec le prénom, le nom et son age.

Mode de calcul du salaire
Le calcul du salaire mensuel dépend du type de l'employé. On distingue les types d'employés suivants :
• Ceux affectés à la *Vente*. Leur salaire mensuel est le 20 % du *chiffre d'affaire* qu'ils réalisent mensuellement.
• Ceux affectés à la *Représentation*. Leur salaire mensuel est également le 25 % du *chiffre d'affaire* qu'ils réalisent mensuellement.
• Ceux affectés à la *Production*. Leur salaire vaut le *nombre d'unités* produites mensuellement multipliées par 5.
• Ceux affectés à la *Manutention*. Leur salaire vaut leur *nombre d'heures* de travail mensuel multipliées par 20 euros.

Modéliser une hiérarchie de classes pour les employés en respectant les conditions suivantes :
• La super-classe de la hiérarchie doit être la classe `Employe`.
• Les nouvelles classes doivent contenir les attributs qui leur sont spécifiques ainsi que le codage approprié des méthodes `calculerSalaire` et `toString`, en changeant le mot "employé" par la catégorie correspondante.
• N'hésitez pas à introduire des classes intermédiaires pour éviter au maximum les redondances d'attributs et de méthodes dans les sous-classes

Employés à risques
Certains employés des secteurs *production* et *manutention* sont appelés à fabriquer et manipuler des produits dangereux. Ces derniers obtiennent une prime de risque mensuelle.
Complétez votre modèle en introduisant deux nouvelles sous-classes d'employés. Ces sous-classes désigneront les employés des secteurs *production* et *manutention* travaillant avec des produits dangereux.
Ajouter également à votre modèle une interface pour les *employés à risque* permettant de leur associer une *prime mensuelle* fixe de 100 Euros.

Collection d'employés
On souhaite maintenant pouvoir afficher le salaire de tous les employés de l'entreprise ainsi que le salaire moyen. Ajoutez une classe `Personnel` contenant une "collection" d'employés. Il s'agira d'une collection polymorphique.
Définissez ensuite les méthodes suivantes à la classe `Personnel` :
• `void ajouterEmploye(Employe)` qui ajoute un employé à la collection.
• `void calculerSalaires()` qui affiche le salaire de chacun des employés de la collection.
• `double salaireMoyen()` qui affiche le salaire moyen des employés de la collection.

Q1 : Construire le diagramme de classe de conception dans StarUML

Q2 : Générer avec StarUML le squelette de code

Q3 : Comparer le squelette généré avec les règles du cours. ***Noter et expliquer les écarts éventuels***

Q4 : Coder les méthodes dans les classes correspondantes

Q5 : Réaliser avec StarUML l'ingénierie inverse du code généré et comparer le résultat avec le modèle initial. ***Noter et expliquer les écarts éventuels***.