Contrôle continu

Ce travail peut être réalisé en binômes.

Contexte

Une enseigne vend des objets, personnalisables de différentes façons : impression de texte, impression de photos, couleur de fond, etc. Les objets sont eux mêmes de différentes natures (bol, tasse, etc).

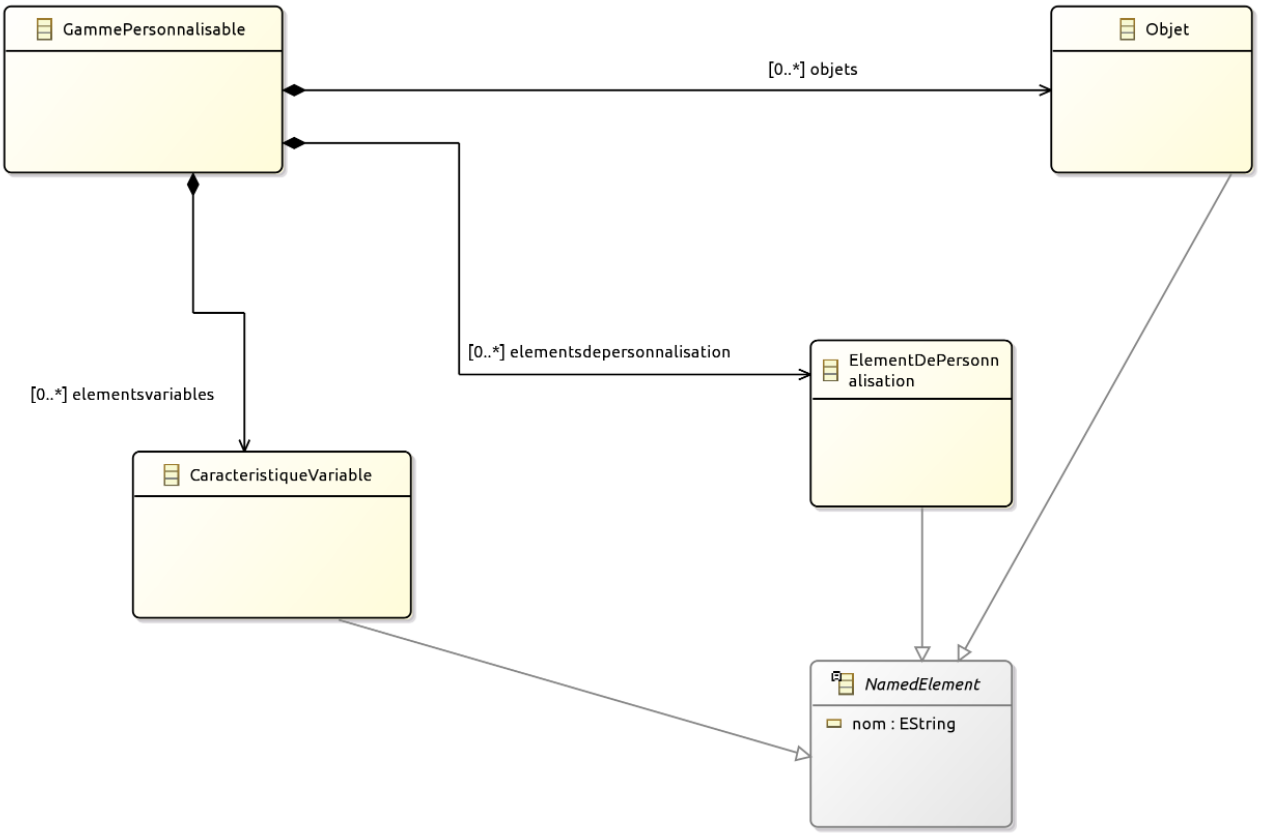
On propose dans un premier temps le méta-modèle suivant pour permettre à l'enseigne de modéliser des gammes d'objets personnalisables.

Une gamme comprend des objets à personnaliser (composition **objets** vers la classe **Objet**). Des exemples d'instances d'objets seraient ici les bols et les tasses par exemple.

Une gamme contient également des éléments de personnalisation (composition **elementsdepersonnalisation** vers la classe ElementDePersonnalisation). Des exemples d'éléments de personnalisation seraient par exemple ici la couleur de fond, l'impression de texte, l'impression de photo, etc.

Une gamme contient enfin des caractéristiques variables, c'est-à-dire les caractéristiques de l'objet personnalisé qui vont varier en fonction des éléments de personnalisation choisis. Ici, on peut par exemple imaginer que ces éléments seraient le prix, et l'image à imprimer. On ne donne pour l'instant qu'un nom aux caractéristiques variables, et pas un type, ce serait une évolution souhaitable du métamodèle.

Cette modélisation est partielle, elle ne permet par exemple pas encore de gérer les contraintes entre les différents éléments de personnalisation.



Travail à réaliser

Question 1. Saisissez le métamodèle en eCore.

Question 2. Générez le code Java/EMF correspondant à ce métamodèle.

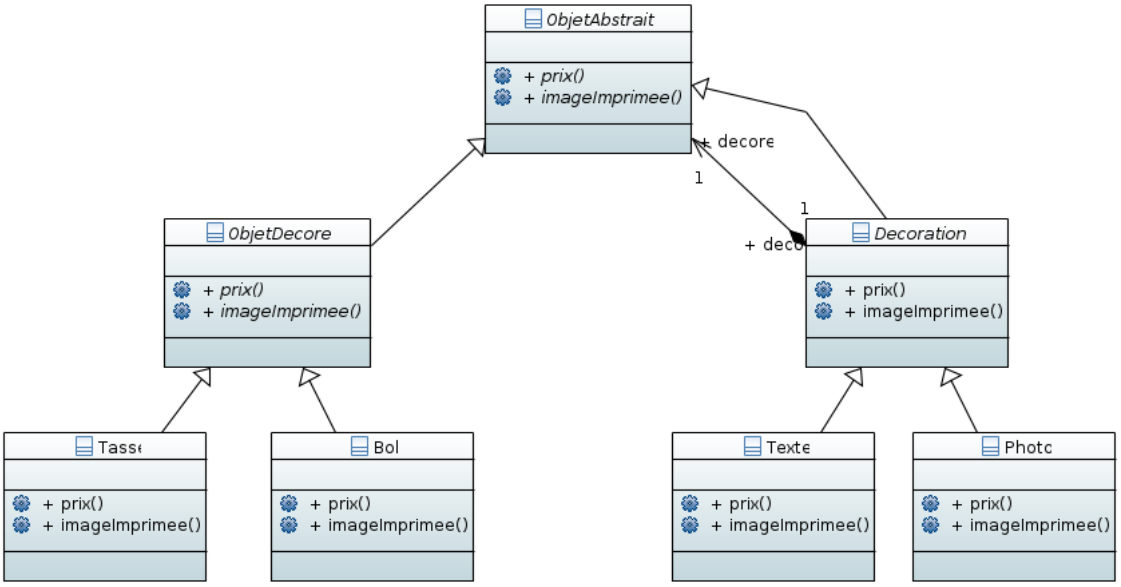
Question 3. Saisissez un petit modèle représentant une gamme permettant de personnaliser des bols et des tasses, avec comme éléments de personnalisation l'impression de texte et l'impression d'image, et comme caractéristique variable le prix et l'image à imprimer.

Question 4. Pour implémenter le logiciel permettant à un utilisateur de personnaliser un objet, et en calculer les caractéristiques, on décide de mettre en place le patron de conception Decorator. On crée donc un modèle

de classes UML à partir du modèle de gamme personnalisable. Le modèle UML est construit de la manière suivante :

- on crée une classe abstraite **ObjetAbstrait**
- on crée une classe abstraite mère des objets : **ObjetDecore**
- on crée autant de classes filles d'**ObjetDecore** que d'objets de la gamme personnalisée (en gardant les noms)
- on crée une classe abstraite pour les décorations : **Decoration**
- on crée autant de classes filles à **Decoration** que d'éléments personnalisables de la gamme personnalisée (on garde les mêmes noms)
- on ajoute une composition entre **Decoration** et **Objet**, de cardinalité 1, et de nom **decore**
- on ajoute à chaque classe de la hiérarchie autant de méthodes que de caractéristiques variables de la gamme personnalisable (en en gardant les noms). Ces méthodes n'ont pas de type de retour car ils ne sont pas spécifiés dans **CaractéristiqueVariable**. Ces méthodes sont abstraites dans **ObjetAbstrait** et **ObjetDecore**.

Pour le petit modèle de gamme personnalisable déjà saisi, on doit donc obtenir le modèle suivant.



Ecrivez la transformation de modèle permettant d'obtenir ce modèle automatiquement.

Question 5. Donnez une syntaxe graphique aux modèles de gammes personnalisables. Vous ne réaliserez pas la palette permettant la saisie des modèles, seulement leur affichage en syntaxe graphique.

Remise de votre travail

Vous remettrez sur moodle une archive de votre projet EMF contenant la réponse aux premières questions, ainsi que pour la dernière question, votre projet définissant la syntaxe graphique. Si vous avez travaillé à 2, un seul étudiant fait la remise, et précisez bien en commentaire les membres du binôme.