ORDRE DES INGÉNIEURS DU QUÉBEC

SESSION de NOVEMBRE 2009

Toute documentation permise.
Calculatrices : modèles autorisés seulement.
Durée de l'examen: 3 heures.

98-INF-B11 CONCEPTION AVANCÉE DE LOGICIEL

QUESTION 1 (15 points) Modélisation d'applications logicielles

Le langage de modélisation UML (Unified Modelling Language) est fréquemment utilisé pour visualiser et communiquer les informations pertinentes à une application logicielle. UML offre plusieurs types de diagrammes permettant de visualiser différentes facettes d'une application logicielle: i) diagramme de cas d'utilisation ("Use-Case diagram"), ii) diagramme logique ("logical view") (incluant les diagrammes de classes, de séquences ou de collaboration), iii) diagramme de composantes ("Component view"), iv) diagramme de répartition ("deployment diagram"), v) diagramme de processus ("Process View").

Décrivez chaque type de visualisation de même que l'utilité de chacun dans un processus de conception orientée objet.

QUESTION 2 (15 points) Principes de conception d'interfaces graphiques d'utilisation

Enumérez et discutez les principaux principes de conception d'interfaces graphiques d'utilisation (GUI) qui doivent être respectés dans une application logicielle afin que celle-ci soit conviviale et facile à maîtriser par les utilisateurs.

QUESTION 3 Conception orientée objet et modélisation UML

Votre compagnie a été mandatée pour concevoir un jeu de quilles Internet. Le client demande que le jeu de quilles puisse être paramétrisé facilement pour offrir soit un environnement de "grosses quilles" soit un environnement de "petites quilles" lorsque l'application est démarrée. De plus, que ce soit pour les grosses quilles ou les petites quilles, on doit pouvoir choisir si l'allée sera en deux dimensions ou en trois dimensions. Il est évidemment obligatoire que le jeu ne contienne qu'une seule allée à la fois. L'allée de quille doit implanter une loi de friction grâce à une méthode publique *DoFriction*, mais vous désirez que l'implantation de cette loi physique puisse être choisie entre plusieurs implantations différentes. Finalement, vous disposez d'une classe *DisplayPanel* qui permet d'afficher facilement l'allée de quille, mais, pour des raisons d'homogénéité de code, l'interface publique de cette classe ne vous convient pas et vous désirez adapter son interface à votre application.

Concevez un diagramme de classes représenté en langage UML (Unified Modelling Language) et respectant les contraintes suivantes:

A) (15 points) L'application de quilles a pour nom *Game* et possède une allée créée selon la contrainte mentionnée en B ci-dessous. En plus de respecter cette contrainte, il faut s'assurer qu'il n'y a qu'une seule allée de quille à la fois dans le jeu grâce au pattern *Singleton*.

B) (20 points) La création d'une allée de petites quilles ou de grosses quilles est paramétrisée grâce au pattern *Abstract Factory* et le type d'allée (2D ou 3D) résulte de l'utilisation d'une *Factory Method* dans le pattern Abstract Factory.

C) (15 points) L'affichage de l'allée grâce à la classe *DisplayPanel* est adaptée via le pattern *Adapter* par une classe de votre conception.

D) (20 points) L'implantation de *DoFriction* utilise le pattern *Bridge* pour implanter les deux lois de friction différentes.