Conception Orientée Objet Examen Année 2024-2025

Durée : 3 heures - Documents autorisés sauf livres et appareils électroniques - Utilisation du crayon autorisée

Exercice 1. Diagramme d'activité : Maison intelligente automatisée (3,5 points)

Considérons une maison « intelligente ». Quand le maître rentre chez lui le soir, un « robot serveur » lui demande ce qu'il veut manger au dîner. Le maître choisit un menu que le robot va préparer dans la cuisine automatisée. Le repas est préparé en deux étapes : la cuisine prépare les ingrédients et puis le robot fait la cuisson. Ensuite, le repas est servi par le robot dans la salle à manger. Pendant que le maître déguste le repas, la salle à manger analyse l'humeur de son maître et mettra une sorte de musique d'accompagnement appropriée. Après le dîner, le robot allume une bougie de massage et puis prépare le lit, en même temps, la salle de bain prépare tout ce qu'il faut pour un bain. En attendant, le maître regarde une émission à la télé. Après, il prend son bain et va au lit. Le maître peut aussi choisir un autre mode : après le dîner, il éteint tout ce qui est automatisé. Dans ce cas-là, il prend juste une douche et puis un roman pour lire quelques pages avant de dormir. Proposez un diagramme d'activité UML pour modéliser toutes les activités décrites ci-dessus du maître, du robot, et des différentes pièces (lieux) automatisées de l'appartement : cuisine, salle à manger et salle de bain (**ne pas** mentionner, comme couloir d'activité dans votre diagramme, ni la chambre ni le salon qui ne sont pas automatisés).

Exercice 2. Diagramme d'état-transition : guichet automatique bancaire (4 points)

Considérons une machine de guichet automatique bancaire (ATM en anglais). La machine peut être éteinte. A chaque fois que l'on rallume la machine, elle effectue une auto-vérification. Si cette vérification aboutit avec succès, la machine se met en état « disponible » (inoccupée), sinon, elle se met en état « hors-service » et attend la réparation faite par un technicien. Pendant la réparation, la machine est en état « maintenance » et elle doit être éteinte et rallumée pour repasser l'auto-vérification. Le service d'un utilisateur débute quand on insère une carte dans la machine. On vérifie d'abord le code saisi par l'utilisateur et puis on affiche sur l'écran le choix d'opération si le code saisi est correct. La carte sera avalée si on a 3 saisies erronées, et dans ce cas-là, la machine revient en état « disponible ». Une fois que l'utilisateur choisit l'opération, la machine la traite et à la fin du traitement, la machine redevient « disponible » en rendant la carte à l'utilisateur. On peut mettre en pause la machine pendant le service et après la pause, la machine poursuit l'étape qui n'a pas été terminée avant la pause. Il est possible que la machine tombe en panne (« hors-service ») à tout moment quand elle est allumée, même pendant le service. Modélisez cette machine par un diagramme d'état-transition UML.

Exercice 3. Domain Diven Design : réservation des vols (3,5 points)

On souhaite modéliser un mini système pour gérer les réservations des vols d'une compagnie aérienne. Dans le système, il y a des aéroports enregistrés et les vols prévus. Le système permet d'effectuer des recherches de vols disponibles : ex. en indiquant la date / heure, les villes de départ et de destination, etc. Pour la réservation des vols, en plus des aspects de base (disponibilités, création du profil du voyageur, paiement, etc.), on s'intéresse également au prix du vol réservé qui est calculé en fonction de plusieurs aspects : l'historique des réservations du même voyageur, l'horaire du vol (ex. prix plus élevé si la date / heure est très recherchée), nombre de voyageurs de la même réservation (voyage en groupe), etc.

Langage ubiquitaire :

Vol : un trajet qui relie deux aéroports (départ et destination), avec un horaire précis prévu.
Aéroport : un aéroport se trouve dans une ville et est identifié par un code unique dans le système.
Ville : une ville peut avoir plusieurs aéroports enregistrés qui sont utilisés dans différents vols.
Réservation : une réservation est associée à un seul vol, mais peut concerner plusieurs voyageurs.
Voyageur : une personne enregistrée dans le système avec les informations personnelles.

Modélisez ce problème avec un diagramme de classe UML en suivant l'approche DDD. Dans le diagramme, vous indiquerez les noms des classes*, ainsi que les liens entre les classes*. Vous légenderez / indiquerez aussi les agrégats avec la classe* racine, les types de classes* « Entité » et « Objet de valeur », ainsi que les classes* de type service : services fonctionnels, factory et repository. **Vous n'avez pas besoin** d'indiquer les attributs ni les méthodes dans les classes.

*Toutes ces classes peuvent être des classes concrètes, classes abstraites ou interfaces.

(Tournez la page pour les exercices suivants)

Exercice 4. Principes d'objet et design patterns (5,5 points)

5.1 Expliquez **pourquoi** le pattern « *Visitor* » respecte **partiellement** l'OCP (*Open Close Principle*). (1 point)

5.2 Comparez avec un petit tableau les deux modes du pattern « *Composite* » : mode transparent et mode sécurisé. (1 point)

5.3 Considérons un programme de ventes de livres. Chaque livre a un éditeur. On distingue deux types d'éditeurs : privé et public, car ils n'ont pas la même façon pour calculer les frais de droit d'auteur pour les livres vendus. On a deux catégories de livres : livres populaires et livres académiques. On enregistre les ventes des livres avec les informations des consommateurs. On souhaite profiter du principe d'objet **ISP** (*Interface Segregation Principle*) pour fournir deux interfaces (classes abstraites) du programme. La première interface permet de manipuler (construction, ajout, suppression, mise à jour, etc.) les objets du programme. La deuxième interface permet de calculer les statistiques intéressantes sur les données de vente. Modéliser ce problème par un diagramme de classe UML. Vous y indiquerez **uniquement** les noms des classes ainsi que les relations entre les classes, **pas besoin de mentionner les attributs, ni les méthodes**. **Sur** votre diagramme proposé, indiquez les **design patterns** utilisés. (3,5 points)

!!! Attention : Vous choisirez UN SEUL exercice à faire parmi 5a et 5b

Exercice 5a. Diagramme de cas d'utilisation : tournoi de jeux en ligne (3,5 points)

On a besoin d'un système informatique pour organiser un tournoi de jeux en ligne. L'organisation d'un tournoi est chargée par le dirigeant de la ligue des jeux en ligne. Il s'agit d'abord de trouver les sponsors (donateurs) pour financer le tournoi. Ces sponsors feront de la publicité lors du « *kickoff* » (démarrage en ligne) du tournoi, sous-forme de vidéo. Les joueurs sont professionnels et ils sont inscrits au tournoi par leurs clubs. Les clubs financent aussi le tournoi. Il y a deux types de match : solo et en équipe. Chaque joueur peut participer aux deux types de matchs. Tous les matchs se déroulent en ligne. Pendant les matchs, les sponsors feront de la publicité (vidéo) entre les parties des jeux. Tous les joueurs peuvent enregistrer la revue (*replay*) de matchs s'ils le souhaitent. Quant aux spectateurs, ils regardent en ligne les matchs du tournoi. Les spectateurs abonnés ont le privilège de regarder les matchs en haute définition. En fin du tournoi, parmi les spectateurs abonnés, on fera un tirage au sort pour avoir un spectateur chanceux qui jouera une partie « show match » contre le joueur champion en solo. Modéliser ce système avec un diagramme de cas d'utilisation UML.

Exercice 5b. Diagramme de séquence : site web d'achat en ligne (3,5 points)

Considérons les interactions entre un client internaute et les pages d'un site web d'achat en ligne. Le site est composé de plusieurs pages : la page login, la page d'accueil pour le client connecté, la page panier et la page validation de l'achat. La page login s'affiche en premier quand le client visite le site. Une fois que les login/mot de passe saisis, la page login vérifie ces informations afin de voir si le client peut continuer vers sa page d'accueil. Si le client se trompe sur ses login/mot de passe, la page login redemande les saisies (sans limite de nombre de fois). Une fois connecté, le client peut effectuer la recherche d'articles sur sa page d'accueil. Pour chaque article trouvé, il peut l'ajouter (toujours sur la page d'accueil) dans le panier s'il le souhaite. Le client finit sa recherche et son ajout des articles pour passer à l'achat. Avant de valider son achat, il visualise sur la page panier tous les articles qu'il a pris. Sur la page validation de l'achat, le client doit saisir l'adresse de livraison et les informations du payement. Le payement s'effectue soit par PayPal soit par carte bancaire. Enfin, la page validation de l'achat envoie un email au client pour confirmer de sa part que l'achat est bien enregistré**.** Modéliser les interactions entre le client et différentes pages (login, accueil, panier et validation) du site web avec un diagramme de séquence UML.