Haute École Léonard de Vinci

Analyse et modélisation
BINV2160

UML exercices

Cambron I., Damas C., Lehmann B.

Bachelier en Informatique de gestion

Bloc2

2022-2023

Exercices

Diagramme des cas d'utilisation

1. Gestion des options

Objectif de l'énoncé : revoir les relations entre cas d'utilisation :

- Cas d'utilisation interne.
- Relation d'extension.
- Généralisation / spécialisation.

Enoncé.

A l'IPL, un responsable d'année doit pouvoir encoder les options ; il faut pouvoir modifier les ECTS, les professeurs responsables, l'intitulé et le descriptif. Si un professeur qui donne une option n'existe pas encore dans le système, le responsable d'année doit l'encoder.

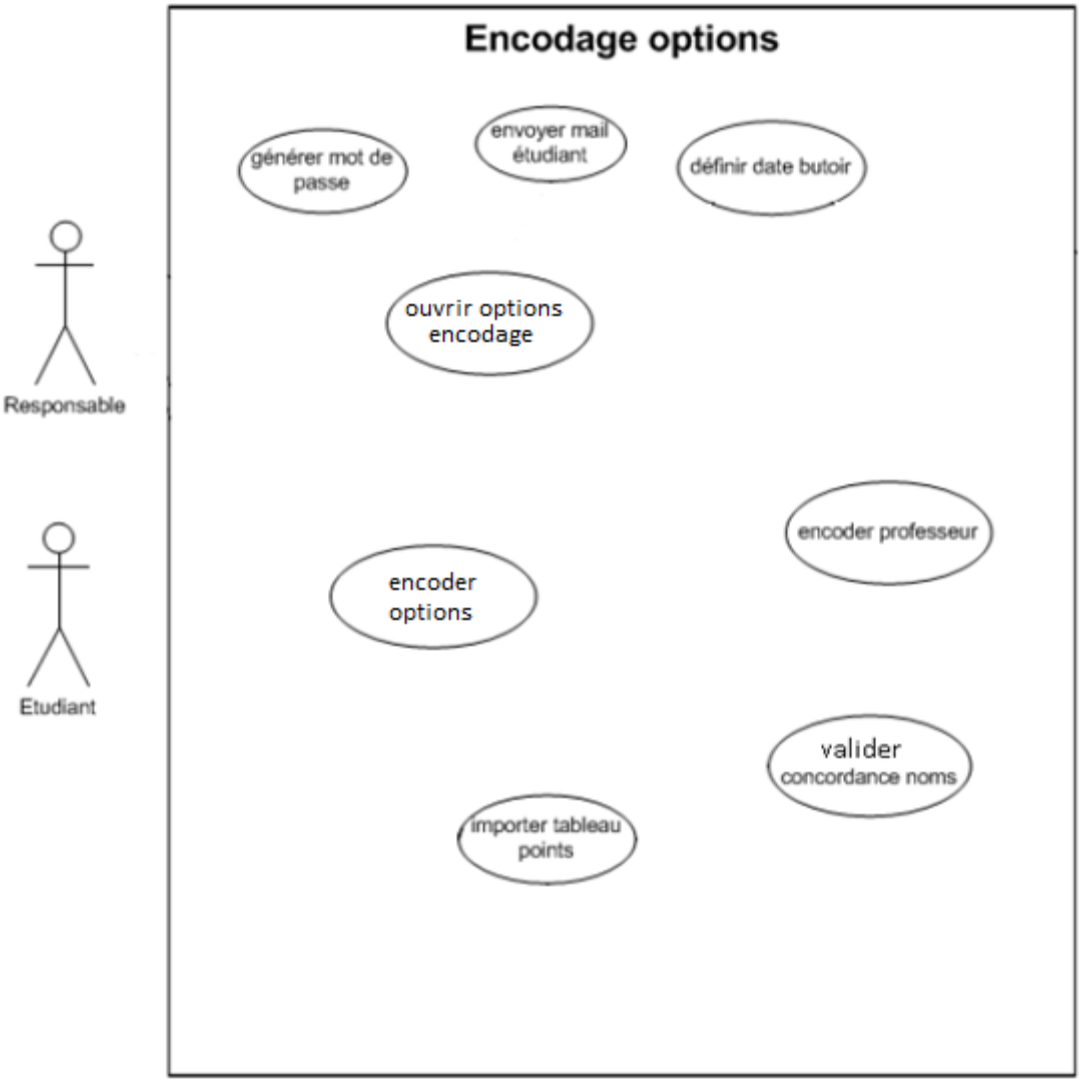
Un responsable d'année doit également importer les tableaux des points des étudiants. Ce tableau des points liste les étudiants en donnant pour chacun sa note aux cours de la session.

Lors de cet import, on retient uniquement le code des cours (par exemple : BINV2160 – Analyse et modélisation). On doit aussi récupérer les informations suivantes : le nom, le prénom, et l'email de tous les étudiants. Il faut donc faire le lien entre les noms/prénoms des listes importées et les noms/prénoms du système. Certains liens seront faits automatiquement ; par ex., pour Jean Dubois. Cependant, pour les noms plus sophistiqués comme Yaëlle du Pont, par ex., il se peut que des problèmes surviennent à cause des caractères spéciaux (accents et espaces). L'application mettra donc en évidence les noms ressemblants et demandera la validation de la concordance des noms au responsable.

L'encodage des options n'est accessible qu'entre deux dates uniquement. Ces dates sont définies par le responsable d'année. Il est important de signaler que le responsable d'année doit ouvrir l'encodage des d'options ; c'est à ce moment-là qu'il doit définir également une date butoir (de fin d'encodage options).

Lorsque l'encodage des choix d'options s'ouvre, un mail est envoyé aux étudiants avec un mot de passe personnel généré automatiquement. Leur login sera le mail.

Compléter le diagramme des cas d'utilisation suivant en ajoutant les relations et corrigeant les erreurs éventuelles.


Encodage options
générer mot de passe
envoyer mail étudiant
définir date butoir
ouvrir options encodage
encoder professeur
encoder options
valider concordance noms
importer tableau points
Responsable
Etudiant

2. Gestion des réunions

On désire concevoir un système en mesure d'organiser les réunions au sein d'une petite PME. Le système tournera sur un réseau local de PC de bureau.

Le but du système est de permettre aux administrateurs d'annoncer des réunions. Une réunion est définie par sa durée, une liste de participants et une période.

Les participants reçoivent alors un mail qui les informe de la réunion. Ils doivent se connecter au système et proposer trois dates où ils sont disponibles. Lorsque tous les utilisateurs ont donné leurs disponibilités, l'administrateur est informé par mail. Il peut alors consulter la date et la plage horaire la plus populaire et fixer la date définitive de la réunion, le système informe les participants de la date définitive. Il est à noter que l'administrateur n'est pas forcé de choisir la date la plus populaire comme date définitive.

Les principales fonctionnalités du système concernent :

1. L'annonce, la modification ou la suppression d'une réunion.
2. Le suivi d'une réunion. Chaque administrateur ne peut suivre que les réunions qu'il a lui-même défini.
3. La proposition des disponibilités d'un participant.
4. La fixation d'une date définitive d'une réunion.

Donnez le diagramme des cas d'utilisation du système.

Diagrammes de classes et d'objets

3. Petites relations structurelles entre classes.

Ces exercices sont extraits du livre « UML par la pratique ».

Déterminez la relation statique appropriée (généralisation, composition, agrégation ou association) dans chaque phrase de l'énoncé. Dessinez le diagramme de classes correspondant à chaque phrase ci-dessous. (Proposez éventuellement différentes solutions pour une phrase).

Considérons les phrases suivantes :

a) Un répertoire contient des fichiers (sous Windows).

b) Une pièce contient des murs.

c) Les modems et claviers sont des périphériques d'entrée/sortie.

d) Les interventions de l'installateur d'alarme chez les clients sont des installations, des maintenances ou des dépannages.

e) Un compte bancaire appartient à une personne physique ou une personne morale (société, asbl…).

f) On veut définir un schéma décrivant les liens familiaux (conjoint, parent/enfant) d'une population de personnes identifiables par un numéro.

4. Gestion de fichiers

Proposez une modélisation du système de gestion de fichiers suivant (Windows-like) :

a) Les fichiers, les raccourcis et les répertoires sont contenus dans des répertoires et possèdent un nom.

b) Un raccourci peut concerner un fichier, un répertoire ou un autre raccourci.

5. Vocabulaire

Classez les phrases suivantes selon le type de relation qu'elles représentent. Chacune représente un type différent de relation : généralisation, spécialisation, instanciation, classification, agrégation, composition, lien et association.

a) Une personne possède un cœur, deux poumons, un foie, …

b) Des personnes conduisent des voitures.

c) "Inspecteur Gadget" est un dessin animé.

d) Ulysse programme son simulateur de vol en Java sur son smartphone.

e) Les personnes sont des étudiants, des professeurs ou des secrétaires.

f) Une équipe de rugby est composée de 8 avants, 2 demis et 5 arrières.

g) Les poules, les dindes et les pintades sont des volailles.

h) Citons parmi les acteurs : Jérémie Renier, Emilie Dequenne et Benoît Poelvoorde.

Au restaurant

Sur le diagramme de classes de la page suivante,

a. Placez les associations se trouvant dans le texte. Lorsque les associations sont nommées, placez la sémantique de l'association. Placez également les multiplicités si elles sont précisées.

Un ***restaurant*** {est composé de} plusieurs ***tables***. Il est midi. Des ***clients*** {sont à} une table. Des ***plats*** et des ***boissons*** {sont posés sur} cette table (la table des clients).

Une boisson peut être une *bouteille de **vin***, une *carafe d'**eau*** ou une *tasse de **café***.

Un ***client*** {mange} un ***plat*** et {boit} une ***boisson***. Plusieurs clients peuvent boire la même boisson (exemple : une bouteille de vin est bue par plusieurs clients). Un client mange un seul plat mais peut {*consommer*} plusieurs boissons (mais une seule à la fois). Un plat est mangé par un seul client.

Un client peut être un ***adulte*** ou un ***enfant***. Un enfant ne boit ni vin ni café. Un enfant a moins de 12 ans.

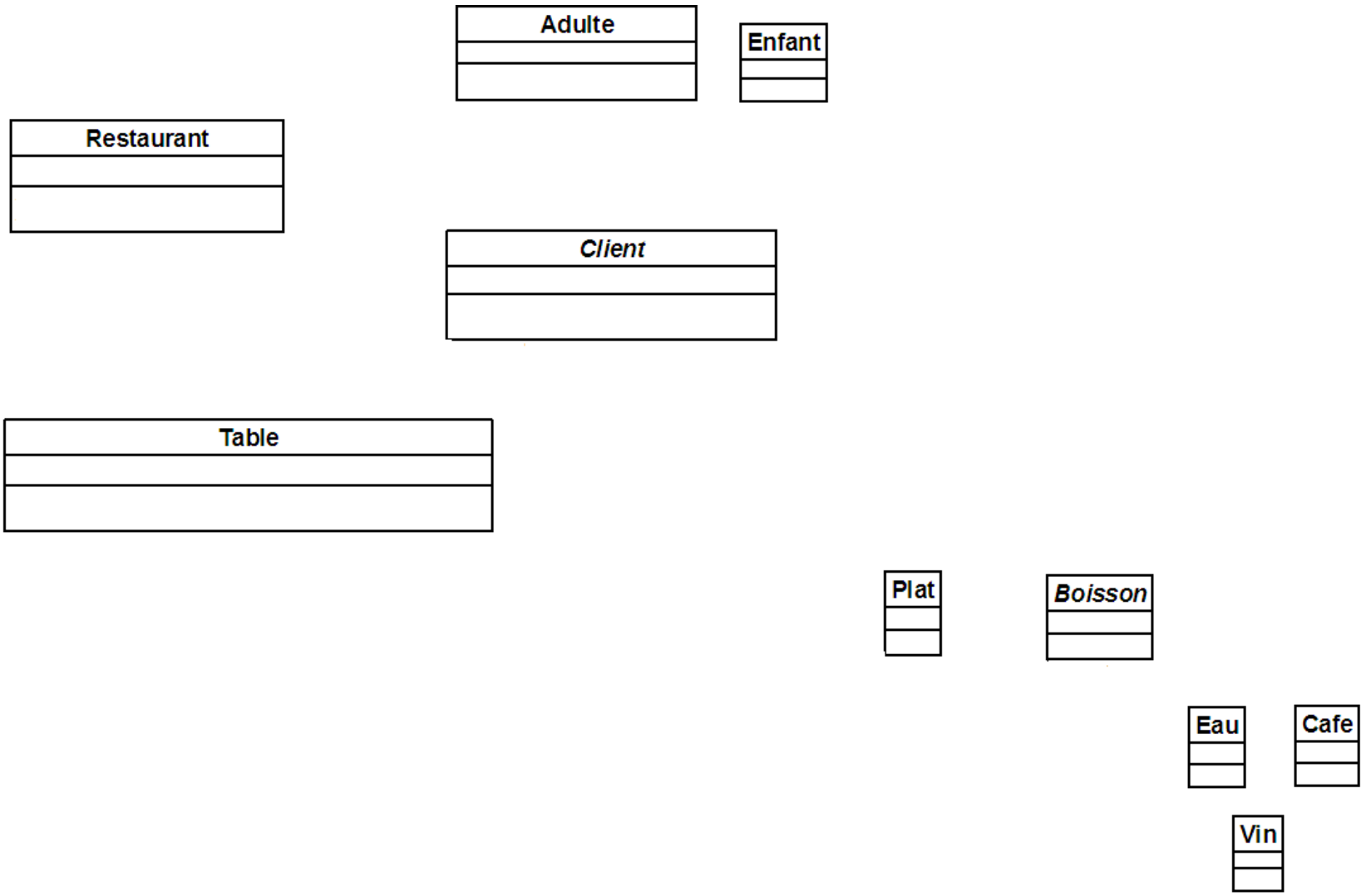
b. Placez ensuite les attributs ou rôles dans le diagramme de telle sorte que les informations suivantes puissent ressortir. Prenez en compte le fait que les bouteilles de vin ont un prix variable, qu'un café coûte 2 euros et qu'une carafe d'eau est gratuite.

'prix', 'mesClients', 'maTable', 'mesTables', 'maBoisson', 'mesBoissons', 'mesBoissons', 'monPlat' et 'mesPlats'.

c. Placez ensuite les navigations.

d. Placez enfin les méthodes.

'void seMettreATable(Table)', 'void poserBoissonSur (Boisson)', 'void poserPlatSur (Plat)', 'void debarrasserLesTables()', 'void boireDuVin(Vin)', 'void boireDeLEau(Eau)'.


Adulte
Enfant
Restaurant
Client
Table
Plat
Boisson
Eau
Cafe
Vin

Dessiner un diagramme d'objets correspondant au texte suivant :

> Le restaurant « Bacchus » comprend trois tables. Laurence et Paul ont une fille Léa, 5 ans. Ils sont à la table 1 avec Valérie. Laurence et Valérie boivent une bouteille de bourgogne. Laurence mange un riz cantonnais. Valérie mange une salade de tomates. Paul boit une tasse de café. Léa mange son dessert et boit de l'eau. A la table 2, Benoit et Anne-Laure boivent du vin mais n'ont pas faim. Leur fille Zoé, 6 ans, mange une glace à la vanille. La table 3 est vide.

Diagramme d'états

6. Tamagotchi

On souhaite modéliser des tamagotchis ; on vous demande de réaliser le diagramme d'états de celui-ci. Souvenez-vous le tamagotchi … Petit animal virtuel japonais…

Le Tamagotchi en état normal n'a pas faim pendant un certain temps (appelé temps d'autonomie). Au bout de ce temps, le Tamagotchi a faim et il pleure. Pour lui donner à manger, l'utilisateur du Tamagotchi le met à table et le Tamagotchi s'arrête de pleurer.

Un Tamagotchi mange pendant un certain temps (appelé temps de restauration). Au bout de ce temps, il se remet à pleurer. Il pleure jusqu'à ce que l'utilisateur le sorte de table. Quand il sort de table, le Tamagotchi revient dans l'état normal... et ainsi de suite tant que le Tamagotchi ne meurt pas. Si le Tamagotchi pleure plus de 5 minutes d'affilée, il meurt.

On suppose que les actions réalisées par le Tamagotchi vers l'utilisateur sont : « avoir faim », « ne plus avoir faim », « mourir » et que les actions effectuées par l'utilisateur et reçues par le Tamagotchi sont « mettre à table » et « sortir de table ».

1. Dessiner un automate à 5 états modélisant le comportement du Tamagotchi. On utilisera les noms « pas faim pleure pas », « faim pleure », « à table ne pleure pas », « à table pleure » et « mort » pour ces 5 états. On fera figurer les actions nommées ci-dessus sur les transitions de l'automate.

2. L'horloge se présente sous forme d'une instance de la classe `Horloge` avec une méthode `lancerTempo(int durée)`. Si le Tamagotchi appelle la méthode `lancerTempo(durée)` à un instant t, il recevra un événement « tempo écoulé » à l'instant t+durée. Ajouter les appels à la méthode `lancerTempo(int t)` et les événements « tempo écoulé » reçus par le Tamagotchi dans l'automate détaillé.

3. Le bip se présente sous forme d'une instance de la classe `Bip`. Si le Tamagotchi appelle la `déclencherPleurs()`, le bip commence à émettre des pleurs sans interruption. Si le Tamagotchi appelle la `arreterPleurs()`, le bip s'arrête. Les méthodes `lancerTempo(durée)`, `déclencherPleurs()` et `arreterPleurs()` s'exécutent instantanément. Au sens donné par UML, sont-elles des activités ou des actions ?

Rappel : une activité est quelque chose qui se déroule tant qu'un objet se trouve dans un certain état. (do) et une action est quelque chose déclenché par un évènement (entry, exit, on « event » …)

4. Préciser à nouveau l'automate en ajoutant les appels aux méthodes `déclencherPleurs()` et `arreterPleurs()` sous forme d'actions d'entrée ou de sortie.

7. Photocopieuse

Représentez le diagramme d'états du système décrit ci-dessous.

Un Système de Contrôle de machine photocopieuse.

On se propose de mettre au point un logiciel de contrôle destiné à être incorporé dans une machine photocopieuse pour donner le fonctionnement suivant :

La machine est allumée en appuyant sur le bouton ON/OFF, se met à préchauffer pendant 30 secondes (*en préchauffage*) avant d'être *prête* à lire les commandes entrées par l'usager (nombre de copies, autres options d'impressions).

La machine est complètement *éteinte* si le bouton ON/OFF est relâché.

Lorsqu'on appuie sur le bouton START, la machine se met à produire des copies (*en production*). Si le bac à papier est vide, une charge de papier est réclamée en affichant un message « charger papier », la machine est *en attente de papier*.

Dès que le papier est mis, la production de copies continue son cours. Parfois, il peut arriver un blocage (*photocopieuse bloquée*), le processus de production de copies s'arrête et le problème est signalé en affichant le message « problème à diagnostiquer ».

Dès que le problème est corrigé (manuellement) et que la machine ne détecte plus aucun blocage, elle reprend automatiquement son fonctionnement normal en s'apprêtant à recevoir de nouvelles commandes. À tout moment, le processus de production des copies peut être arrêté à l'aide du bouton STOP.

8. Système MonAuto

Voici un code de la classe `Réparation` provenant de la modélisation d'un système de gestion des réparations automobiles. Dans cette classe, un diagramme d'états a été implémenté.[1]

- Pouvez-vous fournir ce diagramme d'états de la réparation correspondant à cette implémentation. Afin de simplifier le diagramme, on vous demande de ne pas tenir compte des cas d'échec (pas de fin avec échec).
- Comment vous y prendriez-vous pour modéliser toutes les transitions menant à l'échec ? Répondez en français. NE MODIFIER PAS VOTRE DIAGRAMME !

```
package monAuto;
import java.util.GregorianCalendar;
import java.util.HashMap;
```

[1] Dans le cadre du cours de Patterns en 3ème année, vous verrez comment implémenter proprement un diagramme d'états.

```
import java.util.Map;
import util.Util;

public class Réparation {
private static final int CREEE=0, EN_COURS=1, TERMINEE=2, EN_ORDRE=3, EXPORTEE=4, ARCHIVEE=5;
          private final Voiture voiture;
          private GregorianCalendar demande;
          private GregorianCalendar réception;
          private GregorianCalendar restitution;
          private Map<Pièce, Integer> pièces;
          private Map<Mécanicien, Integer> heuresPrestées;
          private int état;
          public Réparation(Voiture voiture) {
                    Util.checkObjet(voiture);
                   this.voiture = voiture;
                   pièces = new HashMap<Pièce, Integer>();
                   heuresPrestées = new HashMap<Mécanicien, Integer>();
                   état = CREEE;
                   demande = new GregorianCalendar();
          }
          public Voiture getVoiture() {
                   return voiture;
          }
          public GregorianCalendar getDemande() {
                   return demande;
          }
          public GregorianCalendar getRéception() {
                   return réception;
          }
         public GregorianCalendar getRestitution() {
                   return restitution;
          }
          public void ajouterPièce(Pièce pièce, int quantité) {
                   Util.checkObjet(pièce);
                   Util.checkPositive(quantité);
                   if (état == CREEE) {
                             état = EN_COURS;
                             réception = new GregorianCalendar();
                   }
                   if (état != EN_COURS)
                             throw new IllegalStateException();
                   if (pièces.containsKey(pièce))
                             pièces.put(pièce, quantité + pièces.get(pièce));
                   else
                             pièces.put(pièce, quantité);
          }
          public void supprimerPièce(Pièce pièce, int quantité) {
                   Util.checkObjet(pièce);
                   Util.checkPositive(quantité);
                   if (état != EN_COURS)
                             throw new IllegalStateException();
                   if (!pièces.containsKey(pièce))
                             throw new IllegalArgumentException(
                                       "pièce non enregsitrée pour cette réparation");
```

```
        if (pièces.get(pièce) > quantité)
                pièces.put(pièce, pièces.get(pièce) - quantité);
        else if (pièces.get(pièce) == quantité) {
                pièces.remove(pièce);
        } else
                throw new IllegalArgumentException(
                        "impossible de supprimer une quantité plus " +
                    "grande que celle enregistrée pour la réparation");
    }
    public int prix(Pièce pièce){
        Integer nombre = pièces.get(pièce);
        if (nombre == null)
                return 0;
        return nombre*pièce.getPrix();
    }
    public int prixTotalPièces(){
        int somme = 0;
        for (Pièce pièce : pièces.keySet()){
                somme += prix(pièce);
        }
        return somme;
    }
   public void saisirHeuresTravail(Mécanicien mécanicien, int quantité) {
        Util.checkObjet(mécanicien);
        Util.checkPositive(quantité);
        if (état == CREEE) {
                état = EN_COURS;
                réception = new GregorianCalendar();
        }
        if (état != EN_COURS)
                throw new IllegalStateException();
        if (heuresPrestées.containsKey(mécanicien))
                heuresPrestées.put(mécanicien, quantité +
                                                heuresPrestées.get(mécanicien));
        else  heuresPrestées.put(mécanicien, quantité);
    }
    public void terminerRéparation() {
        if (état != EN_COURS)
                throw new IllegalStateException();
        état = TERMINEE;
    }
    public void enregistrerEssai(boolean OK) {
        if (état != TERMINEE)
                throw new IllegalStateException();
        if (OK) {
                état = EN_ORDRE;
                restitution = new GregorianCalendar();
        } else
                état = EN_COURS;
    }
    public boolean résultatEssai() {
        return état >= EN_ORDRE;
    }
```

```
	public void exporter() {
		if (état != EN_ORDRE)
			throw new IllegalStateException();
	// envoyer la réparation à la comptabilité pour facturation
		état = EXPORTEE;
	}
	public void archiver() {
		if (état != EXPORTEE)
			throw new IllegalStateException();
		état = ARCHIVEE;
	}
}
```

Diagramme d'activités

9. Système de commande

Construire un diagramme d'activité avec les couloirs pour modéliser le processus de commande d'un produit quelconque. Le processus concerne les acteurs suivants:

- **Client**
  - Commande un produit.
  - Paie la facture (après réception de celle-ci).
- **Caissier**
  - Encaisse l'argent du client.
- **Vendeur**
  - Traite la commande du client et la transmet à l'entrepôt.
  - Facture la commande au client (après la sortie des produits et leur expédition).
- **Responsable entrepôt**
  - Sort les produits.
  - Expédie la commande.

10. Envoyer un sms

Représentez le diagramme d'activités en rapport avec l'envoi d'un sms.

Il s'agit d'un ancien modèle de gsm, il faut le déverrouiller si nécessaire et il est obligatoire d'écrire le message avant de choisir les destinataires ou d'encoder les numéros encore inconnus.

Supposez qu'il n'existe pas de possibilité d'annuler.

11. Garage MonAuto

Le garage MonAuto répare des véhicules. Il utilise un logiciel à cet effet. Celui-ci est destiné en priorité au chef d'atelier.

Lorsqu'un véhicule se présente au garage, le conducteur se présente au chef d'atelier qui crée alors une fiche de réparation.

Pour créer une fiche de réparation :

- Il faut saisir les critères de recherche de voitures dans le système. Le logiciel de gestion des réparations fournit alors une liste des voitures correspondant aux critères entrés.

- Si la voiture existe déjà dans le système, le chef d'atelier va sélectionner la voiture. Le logiciel fournit, ensuite, les informations sur le véhicule.
- Si la voiture n'existe pas, le chef introduit les informations concernant ce nouveau véhicule.
- Si la voiture est sous garantie, le chef devra saisir la date de demande de réparation.
- Dans tous les cas, le chef d'atelier saisit la date de réception et de restitution.
- Si le dommage de la voiture est payé par l'assurance, le logiciel va fournir une liste d'assurances au chef d'atelier. Ce dernier sélectionnera l'assurance adéquate.
- Finalement, le chef d'atelier enregistre la fiche de réparation.

Pour effectuer la réparation, les mécaniciens et autres employés de l'atelier vont chercher des pièces de rechange au magasin. Le magasinier doit alors sélectionner la fiche de réparation et introduire les pièces utilisées.

Les magasiniers ne fournissent des pièces que pour les véhicules pour lesquels une fiche de réparation est ouverte ; ils saisissent directement les pièces fournies depuis un terminal installé au magasin.

Lorsqu'une réparation est terminée, les mécaniciens fournissent au chef d'atelier les heures prestées sur le véhicule. C'est le chef qui introduit ces heures. Le chef d'atelier essaye alors la voiture.

Si tout est en ordre, il met la voiture sur le parc clientèle et boucle la fiche de réparation.

Les fiches de réparations bouclées par le chef d'atelier seront importées par le comptable dans le logiciel comptable.

On vous demande de

a) Créer un diagramme d'activités avec couloirs pour tout le traitement d'une réparation.
b) Créer un diagramme d'activités pour le cas d'utilisation "Créer une fiche de réparation".

Diagramme d'interactions

12.Self Scanning

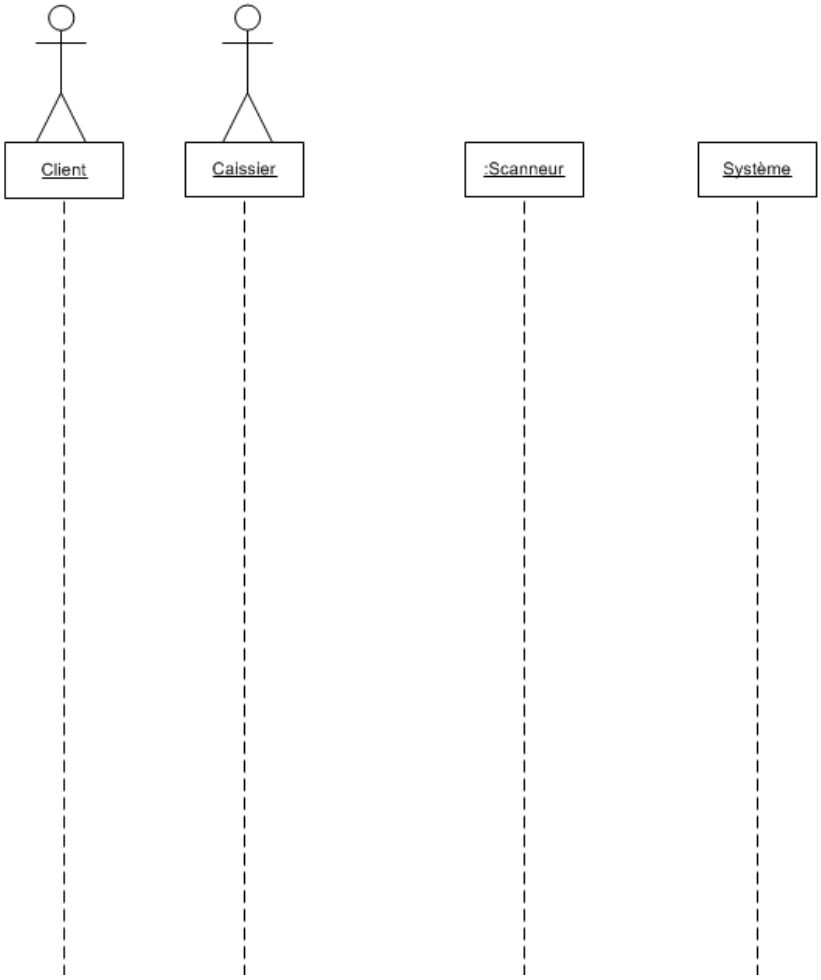
Un magasin offre la possibilité à ses clients de faire du self scanning.

Comment fonctionne ce système ? Le texte ci-dessous présente le scénario nominal d'utilisation par Mr Tchang de ce système. On vous demande de modéliser ce scénario de niveau conceptuel avec un diagramme de séquence.

Mr Tchang entre dans le magasin. Il scanne alors sa carte de client. Le système lui sélectionne un scanneur (petit scanneur à main). Celui-ci affiche alors un message « Bienvenue Mr Tchang Pabong ». Mr Tchang commence alors ses achats en prenant bien soin de scanner chaque article qu'il dépose dans son panier. Lorsqu'un article est scanné, le scanneur récupère le prix et le nom de l'article en s'adressant au serveur, cumule cet article à la liste et affiche un message avec son prix.

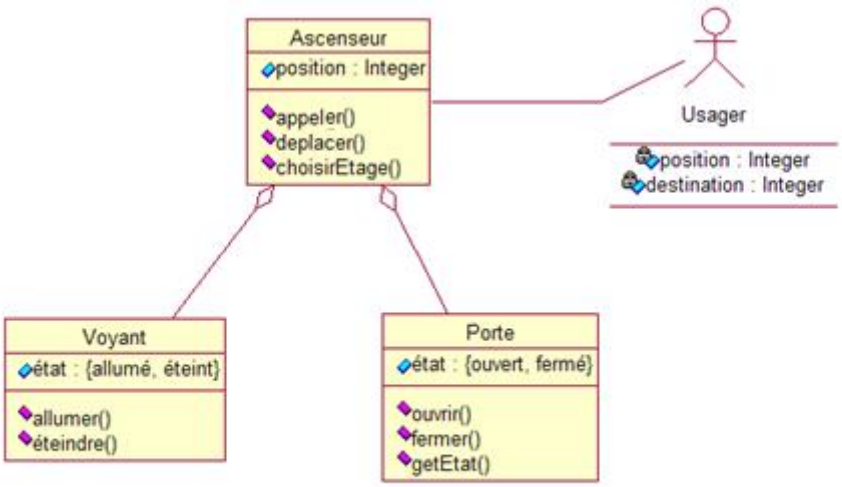
Quand Mr Tchang a terminé ses courses, il se rend vers les caisses et donne son scanneur à un caissier. Le caissier exporte alors les données du scanneur dans le système. Il demande à Mr Tchang de payer (uniquement par carte bancaire). Mr Tchang introduit alors sa carte bancaire dans le système. Il introduit ensuite son code. Le système valide la transaction et imprime le ticket.

Le caissier remet ce ticket à Mr Tchang et lui souhaite une bonne journée.



13.Ascenseur

A partir du diagramme de classes ci-dessous



Rédigez un diagramme de séquence pour modéliser un scénario où un usager voudrait monter en utilisant un ascenseur. Utilisez les attributs des classes pour détailler votre scénario.

Le voyant est à l'extérieur de l'ascenseur. Il s'allume dès que vous appelez l'ascenseur et s'éteint lorsque vous êtes arrivés à destination.

Lorsque l'usager arrive, on considère que le voyant est éteint et que les portes sont fermées.

Voici le scénario :

1. L'utilisateur appelle l'ascenseur.
2. Si l'ascenseur est présent à l'étage de l'utilisateur, les portes s'ouvrent quand on l'appelle sinon l'ascenseur se déplace avant que les portes ne s'ouvrent.
3. L'utilisateur monte ensuite dans l'ascenseur et choisit l'étage. Les portes se ferment.
4. L'ascenseur se déplace jusqu'à l'étage choisi, les portes s'ouvrent et l'utilisateur sort de l'ascenseur.

Cas complets

Tous les cas complets proposés dans le syllabus sont des cas d'examens d'années antérieures.

14. Gestion d'une pizzeria en ligne

PizzaEnLigne fait appel à vos services afin d'élargir ses activités de livraison de pizzas à domicile. L'idée est de permettre aux clients de commander des pizzas en ligne durant les heures d'ouverture du restaurant (de 11h à 15h et de 17h à 24h tous les jours).

Les fonctionnalités attendues :

- Les clients commandent une ou plusieurs pizzas via l'application. Pour commander, un client doit être identifié.
- Le client est identifié via son adresse e-mail et mot de passe. Pour s'inscrire, le client remplit un formulaire (nom, prénom, adresse (rue, numéro, code postal et ville, tel et email). Le système renverra automatiquement un mail accusant bonne réception de son inscription. Ce mail contient le mot de passe du client ainsi qu'une invitation à activer le compte par redirection (click) sur un lien mentionné dans le mail.

Monsieur Dubois,
Vous venez de vous inscrire sur le site de PizzaEnLigne. Voici les informations que vous nous avez transmises :
Nom : Dubois
Prénom : Jean
Adresse : Clos Chapelle aux champs 43 à 1200 Bruxelles
Tel : 02/764.46.57
Mail : JeanDubois@gmail.com
Voici votre mot de passe : 5Fr8PJK8
Pour activer votre compte, vous devez vous rendre sur le site en cliquant sur le lien suivant :
http://www.pizzaEnLigne.be/~jdubois_activate
Merci et à bientôt,
PizzaEnLigne

- Lorsqu'un client veut commander une ou plusieurs pizzas, il mentionne les pizzas qu'il désire et les quantités pour chacune d'entre elles. Quand il a fini, il précise qu'il désire passer sa commande.
- L'application affiche alors l'adresse de livraison (obtenue à l'inscription) et le client doit confirmer ses données. Si cette adresse n'est plus valide, le client introduit une nouvelle adresse de livraison qui sera alors enregistrée dans son profil en remplacement de la précédente. Le système vérifie le format de l'adresse. Le système affiche le récapitulatif et le montant total de la commande :

Commande n° 509 date : 19/11/2020

Pizza	Description	Quantité	Prix unitaire
4 saisons	Artichauts, jambon, champignons et mozzarella	2	14,50 €
Napolitaine	Tomates fraiches, origan et mozzarella	1	11,00 €
	Total à payer		40,00 €

- Le système attend confirmation du client et enfin affiche l'état de la commande.
- Sur le site, le client peut voir où en est sa commande (une seule à la fois en cours). Il peut voir si elle est en train d'être cuisinée, en livraison ou encore en attente (voir point sur la réception des commandes).
- Sur le site, il est possible de visualiser toutes les pizzas de la carte. Un client peut visualiser toutes les pizzas qu'il a déjà commandées et dégustées. Il peut aussi ajouter un commentaire sur une pizza qu'il a déjà commandée (et payée/livrée). Ces commentaires sont modérés par le gérant. Les clients connectés peuvent visualiser les commentaires des pizzas lorsqu'ils consultent la carte. Les clients non connectés peuvent visualiser la carte sans les commentaires.
- La pizzeria est une société composée de 5 personnes à temps plein : 1 gérant, 1 réceptionniste, 2 pizzaïolos et plusieurs livreurs (selon l'intensité de l'activité).
- Le gérant et le réceptionniste sont les utilisateurs de l'application dans la société.
- Lorsque le gérant se connecte, il accède à un récapitulatif des commandes journalières/hebdomadaires/mensuelles ou annuelles avec les données suivantes : le nombre de pizzas commandées, le chiffre d'affaires, la pizza qui rapporte le plus, et autres statistiques intéressantes pour la gestion administrative. Il n'est pas nécessaire de définir des mots de passe pour le gérant et le réceptionniste : le gérant se connecte sur un autre poste que celui du réceptionniste auquel ce dernier n'a pas accès. Le gérant peut effectuer toutes les actions du réceptionniste.
- Le réceptionniste peut consulter les informations des clients : les pizzas commandées et le profil (adresse etc.). Il est également chargé d'organiser les tournées de livraison mais ceci n'entre pas dans le cadre de la nouvelle application. Le réceptionniste peut introduire une nouvelle pizza à la carte, supprimer une pizza ou modifier une pizza. Il adapte aussi les prix selon les demandes du gérant.
- Le réceptionniste réceptionne les commandes de pizzas : il enclenche manuellement la réception des commandes et le système fournit alors une liste des commandes passées de la plus ancienne à la plus récente.
- Le réceptionniste se charge alors de transmettre les nouvelles commandes, une à une, au(x) pizzaïolo(s) afin de préparer les pizzas. Le réceptionniste doit transmettre les commandes en tenant compte de l'activité des pizzaïolos : si les cuisiniers sont débordés, il veillera à patienter avant de transmettre une nouvelle commande.
- Une fois la commande entièrement réalisée et prête à être livrée, le réceptionniste donne les pizzas et l'adresse de livraison au livreur. Il indique à l'application que la commande est « en livraison ».
- C'est le livreur qui reçoit le paiement après avoir livré le client (càd le livreur donne les pizzas et reçoit en échange le paiement). A chaque étape (réception, préparation et livraison) de la commande, le réceptionniste doit indiquer au système où en est la commande. Quand le livreur revient d'une course, le réceptionniste indique au système que la commande est payée (et donc finie).
- Donc, le réceptionniste doit vérifier à intervalles réguliers les commandes passées. Il procèdera selon l'affluence des commandes et la disponibilité des pizzaïolos et des livreurs.

On vous demande de procéder à l'analyse de ce projet en précisant explicitement les points suivants :

1. Énumérez les différents acteurs du système et leurs responsabilités
2. Produisez le diagramme des **cas d'utilisation** (de niveau tâche utilisateur).
3. Détaillez le **modèle complet** du cas d'utilisation relatif à la commande de pizzas.
4. Représentez sous forme de diagramme d'**activités** les cas d'utilisation relatifs à la commande de pizzas jusqu'au paiement et à l'ajout de commentaires. Présentez des couloirs d'activités par acteur.
5. Dessinez le diagramme de **classes** conceptuel. Vous ne devez pas y indiquer de méthode. Par contre, n'oubliez pas de placer les attributs, les associations ainsi que les multiplicités. S'ils sont mentionnés, indiquez les rôles ou la sémantique des associations.
6. Représentez le diagramme d'**objets** suivant :
    - La carte comprend les pizzas :
        - 4 saisons (14.5€ ; artichauts, jambon, champignons et mozzarella)
        - napolitaine (11€ ; tomates fraiches, origan et mozzarella)
        - roma (13€ ; jambon, champignons et mozzarella)
        - 4 fromages (13,5€ ; gorgonzola, parmesan, mozzarella et gruyère)
        - sicilienne (12 € ; anchois, olives et oignons)
    - Jean Dubois commande 2 pizzas 4 saisons et 1 pizza napolitaine. Il s'agit de sa commande en cours.
    - Jean Dubois avait déjà commandé la pizza napolitaine dans une commande du 12 janvier 2016 contenant cette seule pizza au prix de 10,5€. Il avait alors commenté cette pizza « di prima qualità ».
7. Concevez le diagramme d'**états** d'une commande.
8. Réalisez le diagramme de **séquence** du scénario nominal du cas d'utilisation « passer commande » en détaillant toutes les interactions entre les classes du Système. Le diagramme de classes sera complété avec les méthodes nécessaires.

15.Agence Immobilière

Une agence immobilière vous contacte afin de compléter l'analyse d'un nouveau système informatique qu'elle désire mettre en place. Ce système permettra la gestion des ventes de biens.

Les biens sont soit des appartements soit des maisons.

Lorsqu'une personne désire vendre un bien, elle se présente à l'agence. Le vendeur du bien doit alors fournir à l'agent toutes les informations nécessaires à la mise en vente du bien (adresse, le nombre de chambres, prix, surface habitable, année de construction, le revenu cadastral ainsi qu'une description textuelle). Pour les appartements, il faut en plus retenir l'étage du bien ainsi que le nombre d'étages total du bâtiment. Pour une maison, il faut retenir la superficie du terrain et l'éventuelle présence d'un jardin. Un agent prend en charge la mise en vente du bien (*état* « pris en charge »). Il attribue au bien une référence qu'il communique au vendeur. Cet agent devient le responsable de la vente du bien.

L'agent encode également les informations du vendeur (nom, prénom, tel, adresse et email). L'application envoie au vendeur un mot de passe associé à son adresse email (login dans l'application) s'il s'agit d'une personne inconnue du système. Si ce vendeur est déjà encodé dans l'application, il faut éventuellement mettre à jour les données. Le vendeur et l'agent doivent convenir d'une visite du bien qui enverra une demande de rendez-vous au vendeur (sur l'agenda associé à son email). Après visite du bien, l'agent complète la mise en vente et place les photos qu'il a prises. Il doit désigner une photo, parmi celles introduites, qui sera celle mise en évidence dans le listing des biens. Ceci complète la mise en vente. Lorsqu'il le décide, l'agent met le bien en ligne. Le bien est « mis en vente » (*état*).

Cette mise en ligne a deux effets :

- tout internaute peut visualiser le bien sur l'application ;
- le vendeur peut voir ce bien mis en vente parmi tous ses autres biens en vente avec, pour chaque bien, son détail et le nombre de visites.

Les internautes peuvent effectuer des recherches en introduisant zéro, un ou plusieurs critères :

- le prix maximum de vente
- le code postal de la localité où se trouve le bien
- le type de bien : appartement ou maison
- la référence du bien

La recherche aboutit normalement à la présentation d'une liste de biens qui répondent aux critères sauf si aucun bien ne les satisfait.

Si un bien intéresse l'internaute, il peut visualiser toutes les informations de ce bien. Seront alors affichées : toutes les photos et détails du bien. L'internaute peut éventuellement obtenir encore plus de détails en demandant de visiter ce bien, soit par téléphone soit via l'application qui propose alors certaines plages horaires de visite possibles. Cette option envoie une demande de rendez-vous au vendeur (sur l'agenda associé à son email)

Après la visite, si le client veut acheter le bien, il peut effectuer une offre ; il s'agit d'un papier standard à compléter à l'agence ou à lui renvoyer. Cette offre renferme les informations suivantes : la référence du bien concerné, le montant de l'offre, les nom, prénom, adresse email et adresse de l'acheteur (un seul acheteur mentionné), sa date ultime de validité et enfin la signature de l'acheteur. Un agent encode alors l'offre dans le système. Il doit tout d'abord donner la référence du bien. Évidemment, le système vérifie l'état dans lequel se trouve ce bien. Il encode ensuite le nom, le prénom ainsi que l'adresse de l'acheteur s'il n'existe pas déjà dans le système ; si l'acheteur est déjà connu dans le système alors l'agent vérifie ses données, sinon il encode un nouvel acheteur. Un mail lui est alors envoyé avec son mot de passe. L'acheteur a alors accès à ses offres d'achat. Ensuite, l'agent doit indiquer le montant de l'offre et sa date ultime de validité. L'offre est alors soumise (état offre). Cette offre sera désormais visualisable par le vendeur et l'agent.

L'agent responsable du bien doit alors contacter le vendeur et lui signaler l'offre. Le vendeur a la possibilité soit de l'accepter soit de la refuser. Cette décision peut être prise dans le système par le vendeur ou l'agent responsable. Si l'offre est acceptée, le bien apparaîtra désormais avec la mention « option ». Dès qu'une offre est acceptée (état), toutes les autres offres soumises deviennent refusées (état) et aucune nouvelle offre ne peut être soumise. Les parties doivent alors se rencontrer pour fixer les formalités de la vente. La rencontre n'est pas mentionnée dans le système.

Après la signature du compromis de vente entre l'acheteur et le vendeur, l'agent signale au système que le compromis a été signé et le bien apparaît désormais avec la mention « compromis signé » (*état*).

Enfin, quand les notaires des parties s'accordent, en général après 3 ou 4 mois, l'acte de vente est signé. L'agent, averti par courrier notarial, signale alors au système que l'acte a été signé et le bien apparaîtra avec la mention « vendu » (*état*).

Moyennant des frais, toute mise en vente d'un bien peut être annulée par le vendeur pour autant que le bien ne soit pas vendu. Le bien est alors retiré avec la mention « retiré » (*état*).

L'acheteur peut également annuler une offre (état annulée) après une option et même après signature du compromis (moyennant dédommagement). C'est l'agent qui encode les annulations dans le système.

Chaque agent possède un nom, un prénom, un email, un tel et un mot de passe de connexion.

Exercices UML

On vous demande de procéder à l'analyse de ce projet en précisant explicitement les points suivants :

1. De réaliser le diagramme des **cas d'utilisation**. Présentez les tâches-utilisateur et les sous-fonctions pertinentes. [6 pts]
2. De fournir le **diagramme d'états** d'un bien mis en vente, en décrivant les transitions au moyen des cas d'utilisation. [4 pts]
3. De réaliser le **diagramme de classes conceptuel** du système. Vous ne devez pas y indiquer de méthode. Par contre, n'oubliez pas de placer les attributs, les associations ainsi que les multiplicités. S'ils sont mentionnés, indiquez les rôles ou la sémantique des associations [6 pts]
4. De réaliser au choix l'un des deux diagrammes suivants :
    a. le **diagramme d'activités** liées à la vente d'un bien. [4 pts]
    b. le **diagramme de séquences** d'encoder une offre. [4 pts]
5. Détaillez le modèle complet du cas d'utilisation « prendre en charge une mise en vente » (ou appellation utilisée dans la série). Décrivez :
    a. Le modèle,
    b. Le scénario nominal qui concerne un appartement,
    c. Le scénario alternatif qui concerne une maison,
    d. Une extension.

Représentez dans un diagramme de séquence le scénario principal du cas d'utilisation « encoder une offre » en considérant le système en boîte blanche, c.-à-d. détaillé en ses classes. Indiquez les barres d'activation.

L'acteur principal s'adresse uniquement à la couche IHM (en boite noire) qui elle s'adresse au use case contrôleur.

Dans le système, on a établi les classes (Bien, Offre et XX de votre diagramme de classes) et méthodes suivantes qui suffisent pour votre scénario :

UseCaseControleur
- biens : List<Bien> - acheteurs : List<XX> …
+ rechercherBien(ref :String) : Bien + rechercherAcheteurs(nom :String) : List<XX> + selectionnerAcheteur(acheteur :XX) +encoderOffre(montant :double,date :LocalDate) :boolean

Bien
…
… + ajouterOffre(offre :Offre) …

Offre
… - état {acceptée, soumise, refusée, annulée} …
+ Offre(bien :Bien, montant : double, date : LocalDate) + ajouterAcheteur(quelquun :XX) …

16. Boulangerie

Un boulanger désire installer un distributeur automatique de pain dans sa boulangerie afin que les clients qui commandent leur pain puissent venir le chercher quand ils veulent.

L'idée est innovante et surpasse les machines à pain que l'on voit déjà partout. Il s'agit véritablement de satisfaire le client avec les pains de son choix par le biais d'un système de commande et surtout qu'il puisse venir les chercher quand ça l'arrange.

L'application sera disponible en ligne et également directement sur la machine distributrice. Toutefois, tout n'est pas accessible en ligne ni sur le distributeur.

Sur l'application en ligne, le boulanger bénéficiera d'un accès « administrateur » afin de pouvoir notamment ajouter des nouveaux pains, supprimer des pains et modifier des prix. Le pain est identifié par son nom, son poids, son type (blanc, gris, multi, froment) et son prix.

Un client pourra passer une commande via l'application internet. Le site internet présente la boulangerie et la carte des différents pains ainsi que leur prix. Il propose également de se connecter. Effectivement, pour pouvoir passer commande, il faut être authentifié.

Un internaute peut également s'inscrire sur le site en fournissant les renseignements suivants : nom, prénom, date de naissance, adresse, email, gsm et tel. Il crée également son login et son mot de passe.

La commande passe par un traditionnel panier à valider dans lequel on ajoute et supprime des pains. Ce panier peut être rempli de pain blanc, pain gris, … en quantité désirée. Il y a une vérification à faire de la place nécessaire à la commande. Si le panier contient trop de pains alors l'application le signalera. La manière dont cette quantité sera calculée n'a pas encore été établie. Evidemment, il faut aussi que la commande contienne au moins un pain.

Après avoir validé son panier, le client doit préciser une tranche horaire durant laquelle il viendra retirer sa commande à la machine. Finalement, il doit payer sa commande. Il existe un système de paiement en ligne (de type Ogone) qui prend en charge le paiement et signale au système le bon versement de la somme due.

Entre le moment de la commande et la mise à disposition de celle-ci, il s'écoule toujours au minimum 12h et au maximum 168 heures (càd une semaine). Une commande reste dans la machine 12h maximum ; au-delà de ces 12 heures, la commande est retirée de la machine et est donnée au resto du cœur.

La machine qui distribue le pain est un système assez ingénieux qui dispose de 100 emplacements (de petite ou grande capacité) pouvant accueillir les commandes (petites et grandes) des clients. Elle dispose d'un terminal permettant au client de retirer ses commandes et d'acheter du pain. Elle comprend donc un système bancaire qui permet le paiement lors de l'achat de pain. Lorsqu'on achète du pain directement à la machine à pain, il n'y a pas de commande ni connexion. Il s'agit alors d'une machine à pain traditionnelle qui distribue un seul pain à la fois et enregistre une vente.

Lorsque le client retire sa commande à la machine, il doit se connecter au système. Le système lui propose alors de retirer sa commande en lui indiquant le numéro de l'emplacement de celle-ci.

Le boulanger récolte les commandes passées au minimum 2 fois par jour. Quand il se connecte à l'application internet, via son login et son mot de passe, celle-ci doit lui permettre de lister toutes les commandes (par exemple, pour connaître son chiffre d'affaires) ainsi que de lister tous les pains à préparer (pour que ceux-ci soient disponibles dans la bonne tranche horaire). Quand les pains sont terminés, le boulanger doit se connecter à l'application mais cette fois-ci directement sur le distributeur pour disposer les commandes dans les emplacements.

Lorsqu'il se connecte pour disposer les commandes dans le distributeur, si des commandes n'ont pas été retirées dans les 12h alors le boulanger en est immédiatement averti. Le boulanger devra donc d'abord retirer les commandes abandonnées afin de libérer les emplacements. Le système lui propose ensuite de disposer les commandes. Il liste toutes les commandes une à une en indiquant l'emplacement prévu. Le boulanger rassemble les composantes de la commande et les dispose dans cet emplacement. Ensuite, le boulanger passe à la commande suivante. Ainsi de suite, la machine se remplit donc des commandes. Le boulanger peut aussi placer des pains en vente libre dans le distributeur.

Le système est donc disponible en ligne mais également directement sur le distributeur ; les besoins fonctionnels sont différents selon l'une ou l'autre application. On vous demande de faire en sorte que votre analyse présente bien cela.

Actuellement, le boulanger est seul utilisateur mais il sera bientôt accompagné de son fils et plus tard peut-être de sa belle-fille ou de son beau-fils. Votre système doit permettre de savoir qui a récolté la liste des commandes et qui a disposé la commande dans son emplacement.

Attention le panier est lié à la session du client, plus précisément à sa session. Il est maintenu en JavaScript côté client et non côté serveur.

On vous demande de :

a. Lister les acteurs du système et de les définir si besoin.
b. Réaliser les diagrammes des cas d'utilisation.
c. Réaliser le diagramme d'activités qui décrit ce que doit faire le boulanger quand il se connecte au système qui se trouve sur le distributeur.
d. Réaliser le diagramme d'états de la commande.
e. Définir le diagramme des classes. Vous ne devez pas y indiquer de méthode. Par contre, n'oubliez pas de placer les attributs, les associations ainsi que les multiplicités. S'ils sont mentionnés, indiquez les rôles ou la sémantique des associations.
f. Dessiner le diagramme d'objets qui correspond au scénario suivant « Eliott passe commande de 2 pains : un pain soleil de 400 gr à 2,10€ et un pain ardennais de 800gr à 2,80€. Sa commande a été prise en compte par le boulanger et il l'a également disposé dans l'emplacement n°13. »
g. Tracer le diagramme de séquence du scénario nominal du cas d'utilisation « passer commande », sur base d'un diagramme de classes fourni par les professeurs.

17.Ne brassons pas que de l'air !

Beaucoup de brasseries artisanales wallonnes existent actuellement. Ces brasseries issues de la tradition ou de nouvelles idées prometteuses véhiculent des valeurs importantes du terroir belge. Le secteur brassicole wallon est un milieu où il y a beaucoup d'échanges de savoirs, conseils, savoir-faire, … bref la collaboration est indispensable pour survivre !

L'application à développer est une plateforme d'échanges à destination des brasseurs pour le matériel brassicole d'occasion.

L'application doit être accessible à un certain nombre d'administrateurs, aux brasseurs et à des internautes curieux. Elle doit également être accessible au comptable qui pour le moment est l'un des administrateurs (mais peut-être que cela changera à l'avenir).

Quand un brasseur s'inscrit, il fournit les informations suivantes : son nom, son prénom, son email, son login, son mot de passe et une adresse postale. Il précise également la brasserie pour laquelle il travaille. Plusieurs brasseurs peuvent travailler dans une même brasserie. Donc lorsqu'un brasseur s'inscrit, soit il crée sa brasserie soit il la référence car elle a déjà été créée par un collègue.

Pour chaque brasserie, on doit indiquer son nom et son adresse postale, son email et une image.

Un brasseur authentifié peut soumettre une annonce de vente d'occasion (« soumise »). Une annonce qui contient un titre, un descriptif, un tarif éventuel et une photo. L'annonce est liée à la brasserie.

L'annonce n'est publiée que lorsqu'un administrateur l'a validée et qu'elle a été payée (5€ pour une annonce). Donc, lorsqu'une annonce est soumise et lorsque l'administrateur la valide, un virement est envoyé par mail au brasseur avec le montant, la communication et le numéro de versement.

Il n'y a pas de paiement en ligne. Quand le comptable constate le paiement de l'annonce, il le signale dans l'application.

Une annonce est publiée pour une durée par défaut de 6 mois. On doit retenir quel administrateur a validé l'annonce. On retient également quel comptable a enregistré le paiement.

Les annonces sont triées par rubrique et éventuellement des sous-rubriques. Par exemple, on a les rubriques :

- *Brassage*
- *Pressoirs*
- *Bouchons*
- *Nettoyage*
- …

Dans la rubrique *Brassage*, on peut avoir les sous-rubriques suivantes par exemple :

- Levures à bières
    - Levures liquides WYEAST
        - Levures type Ale

        - Levures type Lager
        - Levures bière blanche
        - …
    - Levures à bière BREWFERM
- Malts et flocons
- Houblons
- …

Chaque annonce est donc liée à une seule rubrique ou sous-rubrique ou sous-sous-rubrique ou etc.

C'est un administrateur qui doit pouvoir créer des rubriques, les modifier et les supprimer.

Les internautes, les brasseurs et les administrateurs doivent pouvoir consulter toutes les annonces.

Un administrateur peut supprimer une annonce à tout moment tant que celle-ci n'est pas périmée. L'annonce reste dans le système mais elle n'est plus visible.

Un administrateur peut également modifier une annonce, après le paiement de celle-ci.

On connaît pour chaque administrateur et le comptable son nom, son prénom, son email, son téléphone, son login, son mot de passe. Chaque administrateur doit également mettre une photo (pas le comptable).

Un administrateur doit pouvoir faire tout ce que les brasseurs peuvent faire.

Questions

1. Fournissez le diagramme des cas d'utilisation du système à analyser.
2. Fournissez le diagramme d'états d'une annonce.
3. Fournissez le diagramme de classes de niveau conceptuel. Vous ne devez pas y indiquer de méthode. Par contre, n'oubliez pas de placer les attributs, les associations ainsi que les multiplicités. S'ils sont mentionnés, indiquez les rôles ou la sémantique des associations.
4. Fournissez le diagramme des activités liées aux annonces d'occasion.

18.Marathons

Une agence souhaite organiser des marathons un peu partout dans le monde. Elle compte sur votre aide pour un premier dossier d'analyse.

Un marathon est une course mythique de 42,195 km ; il s'agit de la distance parcourue par Phidippidès en 490 avant JC pour annoncer la victoire des grecs contre l'envahisseur perse.

L'agence souhaite s'implanter comme une référence dans l'organisation de marathons. Elle prévoit également un système de fidélisation.

Tout d'abord l'application doit permettre à n'importe qui (connecté ou pas) de voir les marathons organisés : les marathons passés seront visualisables avec les temps des coureurs tandis que les futurs marathons proposent un récapitulatif des informations de celui-ci ainsi qu'un formulaire d'inscription au marathon.

L'application offre la possibilité à tout moment de s'inscrire sur le site ou de se connecter sauf si l'utilisateur est déjà connecté. Lors de l'inscription, il est demandé d'indiquer les informations suivantes : nom, prénom, mail, mot de passe, gsm, meilleur temps sur marathon, poids et taille. L'utilisateur devient dès lors membre de l'agence.

L'inscription à un marathon diffère légèrement si l'utilisateur est membre authentifié ou pas. Authentifié, il lui suffit de confirmer son inscription au marathon et de payer. S'il n'est pas authentifié, il lui est proposé soit de s'inscrire sur le site soit de s'authentifier avant de pouvoir confirmer son inscription au marathon et payer.

En tant que membre, un utilisateur peut voir toutes les courses auxquelles il a déjà participé avec le détail de sa course. Il peut aussi apercevoir celles auxquelles il est inscrit.

Un membre qui s'inscrit à un marathon reçoit un dossard pour ce marathon-là dès que le paiement a été effectué. L'agence sous-traite les paiements au système PayPal.

Les membres sont évalués par une échelle de niveau qui leur octroie des avantages croissants. Cette échelle s'évalue en nombre de courses déjà terminées. Un membre débute au niveau caillou. Dès qu'il achève 5 marathons, il devient bronze. Au bout de 10, il devient argent, après 15 il passe or et enfin après 21 marathons il termine diamant. Ce niveau impacte le prix à payer lors de l'inscription à un marathon.

Lorsqu'un marathon a lieu, différents check points sont dispersés sur le parcours (tous les 5 km, au début et à la fin) ; ce système est existant. Chaque marathonien dispose en fait d'une puce sur son dossard qui permet d'enregistrer son temps lors du passage au checkpoint. Quand un marathon se termine, un organisateur exportera les données de temps du système de checkpoints dans notre système à développer. Cet import dans le système permet d'enregistrer les différents temps aux checkpoints de chaque coureur et d'évaluer les niveaux. Si un membre change de niveau, il en est averti par un sms envoyé directement sur son gsm personnel.

Les organisateurs peuvent effectuer différentes actions sur le site. Ils doivent pouvoir ajouter un marathon en précisant la ville concernée, la date, une description et un prix de base. Il doit être

possible de modifier ces informations ultérieurement. Pour chaque marathon, il doit être possible de lister les inscrits ou, s'il a déjà eu lieu, le classement.

Les organisateurs doivent pouvoir modifier les échelles de niveaux. Ceci doit être faisable pour autant que les avantages acquis ne soient jamais diminués. Si des membres sont impactés par ce changement, un mail leur est envoyé avec l'explicatif par le système de mail existant.

Les organisateurs sont des membres qui peuvent effectuer tout ce qu'un membre peut faire. Ils bénéficient toutefois directement d'un niveau supplémentaire au niveau auquel ils ont normalement droit. Il arrive souvent que des membres deviennent organisateurs. Ce changement peut être effectué par un autre organisateur. Il conserve alors toutes ses données. Evidemment, l'inverse se produit aussi, un organisateur peut redevenir un « simple » membre tout en conservant ses données mais il perd alors son privilège de niveau supplémentaire.

L'administrateur de la plateforme, par contre, bénéficie d'un compte particulier qui lui permet de supprimer des courses, des membres ou des organisateurs. Il peut effectuer ce que l'organisateur peut faire.

On vous demande de réaliser :

1. Le diagramme des **cas d'utilisation**. Présentez les tâches-utilisateur et les sous-fonctions pertinentes. [6 pts]
2. Le **diagramme de classes conceptuel**. N'y indiquez aucune méthode. Par contre, n'oubliez pas de placer les attributs, les associations, les rôles ou la sémantique ainsi que les multiplicités. [6 pts]
3. Le **diagramme d'activités** de toutes les activités que peuvent faire un internaute et un membre [4 pts]
4. Le **diagramme de séquence** du cas d'utilisation « exporter les données de temps » [4 pts].

Le diagramme de séquence est préparé sur les feuilles de réponses.

L'organisateur demande l'export pour un marathon.

Notre système, dont les classes ne sont pas détaillées, dispose de la liste des coureurs inscrits à ce marathon.

Il importe les différents temps aux checkpoints de chaque coureur.

Il évalue le niveau de chaque coureur.

Vous devez représenter tous les retours des appels.

Voici les méthodes dont vous disposez :

UCC

- La méthode **exportMarathon(marathon):boolean** demande l'export pour un marathon.

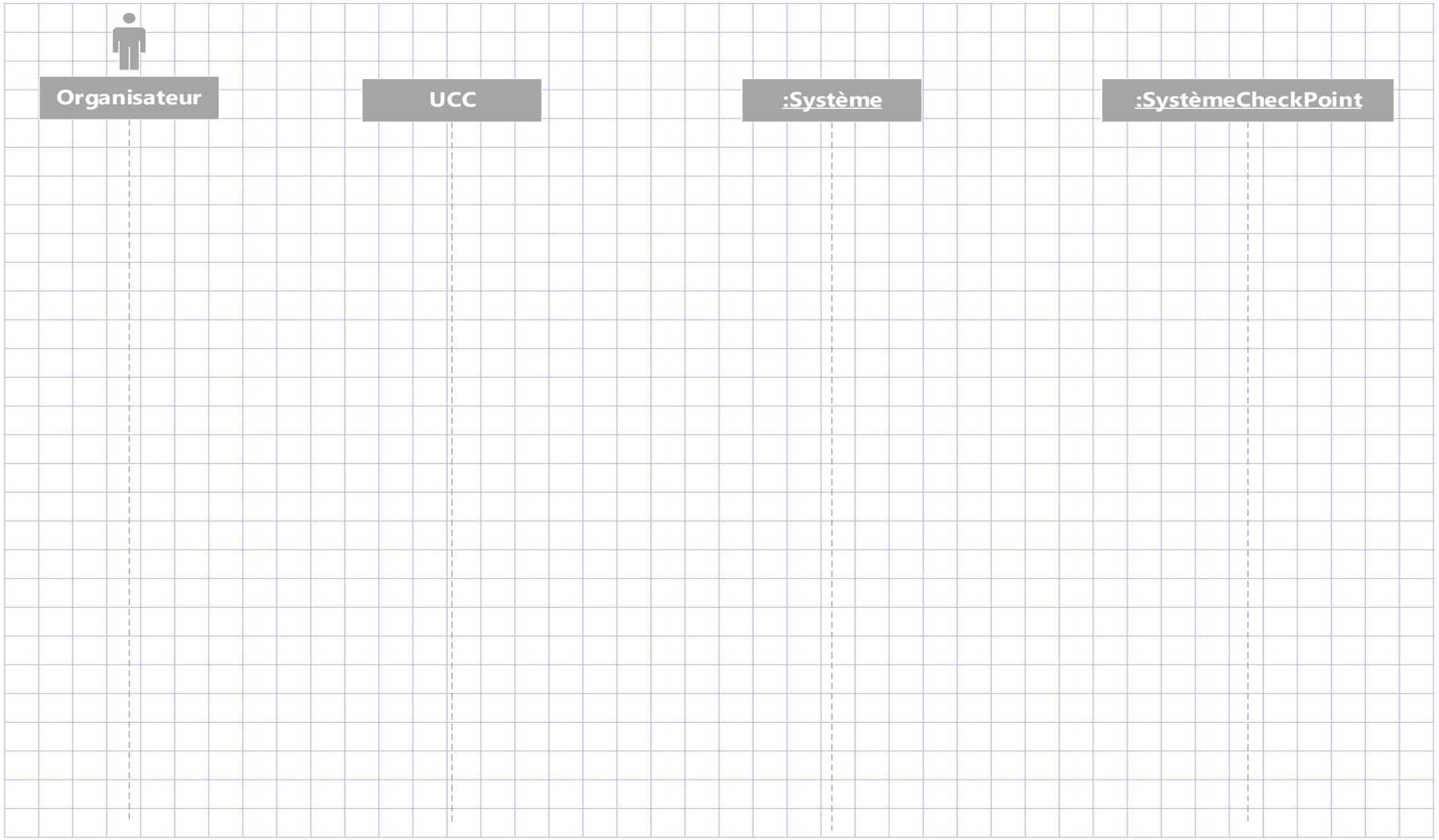
Système

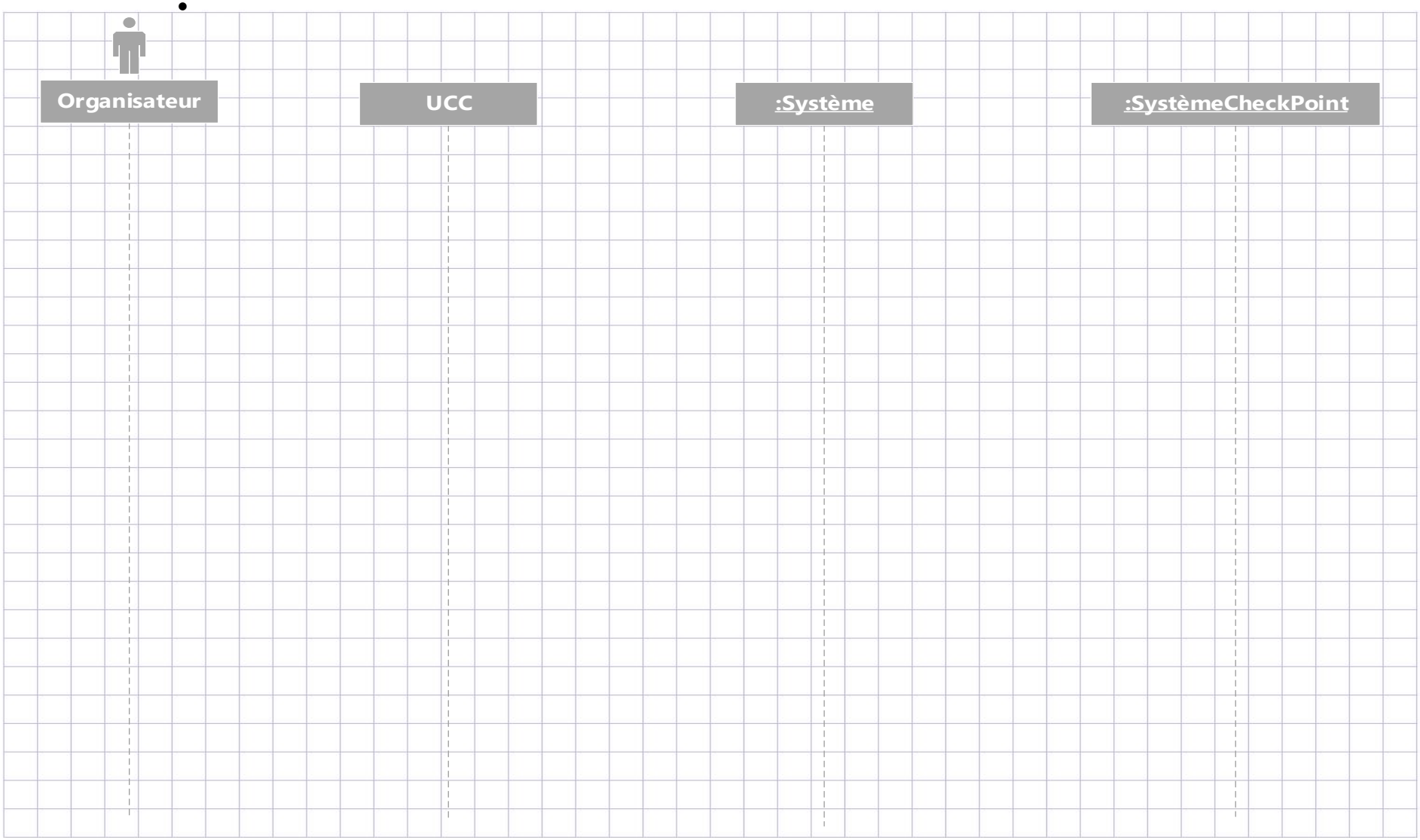
- La méthode **fournirCode(marathon):codeMarathon** fournit le code du marathon.
- La méthode **fournirCoureursInscrits(marathon):coureurs** fournit la liste des inscrits à un marathon fourni.
- La méthode **fournirDossard(coureur):dossard** fournit le dossard du coureur.
- La méthode **enregistrerTemps(coureur,checkPoint,temps):boolean** permet l'enregistrement du temps du coureur au check point fourni. Si le temps est null, il ne faut pas l'enregistrer.
- La méthode **évaluerNiveau(coureur):niveau** renvoie le niveau du coureur en fonction des courses qu'il a effectuées.

SystèmeCheckPoint

- La méthode **exporter(codeMarathon,dossard):temps** fournit la liste des temps du coureur dont le dossard est donné en paramètre, pour le marathon, dans l'ordre strictement croissant des checkpoints. Certains des temps peuvent être nulls.

- Réponse question 4:




Organisateur
UCC
:Système
:SystèmeCheckPoint

19. Père Noël

Monsieur Noël possède un célèbre magasin de jouets. Soucieux des traditions de fin d'année, il s'efforce chaque année d'offrir à ses clients un service spécial « Noël » rappelant la tradition du père Noël. En effet, les parents prennent contact avec lui et communiquent les nom et prénom de chacun de leurs enfants ainsi qu'un montant maximum de dépense par enfant.

Monsieur Noël, se faisant passer pour le père Noël, envoie par courrier postal aux enfants le catalogue des jouets qu'il possède ainsi qu'une invitation à retourner une liste des jouets désirés (cette liste est ordonnée, le premier jouet étant celui le plus désiré). Les enfants envoient ainsi leur courrier au « père Noël » qui se charge de vérifier le coût des cadeaux des enfants par rapport au plafond des parents. Après accord des parents sur la commande des jouets, monsieur Noël envoie la facture aux parents. Dès que son compte est crédité, les cadeaux de la famille sont rassemblés dans un colis. Le soir de Noël, monsieur Noël se charge de distribuer les colis.

Monsieur Noël désire améliorer son service. Il s'adresse à vous afin de mettre en place un site internet. Votre futur système devrait **permettre aux enfants d'effectuer leur « commande » de Noël en ligne sous l'autorisation parentale**.

Comment monsieur Noël envisage-t-il son futur système ?

Les parents s'inscrivent sur le portail en ligne du magasin soit à partir de leur pc personnel soit dans le magasin (plusieurs terminaux seraient à disposition des clients).

L'inscription requière un système d'authentification sécurisé ; c'est pourquoi on songe à faire appel à un autre système informatique. Ce système « Au temps TIC » permet de lire les données d'une carte d'identité électronique et de vérifier sa validité par rapport au registre national. Soit, grâce à « Au temps TIC », les parents seront identifiés de manière sécurisée pendant leur inscription. Le système retiendra leur numéro de registre national, leurs nom et prénom ainsi que leur adresse. Le système leur attribuera un code unique sur base duquel ils pourront ensuite se connecter au système.

Lorsqu'ils sont connectés, les parents ont accès aux fonctionnalités décrites ci-dessous.

Les parents peuvent préciser la composition de leur famille, càd inscrire leurs enfants (en mentionnant leur nom, prénom, date de naissance) et mentionner leur conjoint si celui-ci est inscrit sur l'application. Pour chaque enfant, les parents recevront un code unique sur base duquel l'enfant pourra se connecter à l'application.

Les parents peuvent modifier la composition de leur famille au fur et à mesure des années (ajouter les nouveau-nés, modifier le prénom d'un enfant ou indiquer que des enfants ne sont plus à la maison).

Chaque année, les parents pourront décider de demander un colis de Noël pour leur famille. A ce moment, ils seront invités à :

- mentionner le montant maximum par enfant qui recevra un cadeau cette année. Le montant est identique pour chacun des enfants de la famille.
- éventuellement demander qu'un seul cadeau soit effectué par enfant. Si cette dernière mention n'est pas effectuée, monsieur Noël effectuera un maximum de dépenses en deçà du montant plafond.

Les enfants accèdent au système soit en magasin soit à partir de leur pc personnel (via le portail en ligne du site). Un enfant se connecte au système en indiquant son code unique.

Si les parents demandent un colis pour Noël de l'année en cours, chacun des enfants possède un panier de jouets qu'il peut remplir selon ses desideratas. Les jouets du magasin sont présentés dans un catalogue en ligne (diverses recherches possibles : par mot-clé, par âge, …). En consultant donc le catalogue en ligne, l'enfant peut ajouter dans son panier les jouets qui lui plaisent. L'enfant peut modifier son panier en supprimant également ce qu'il y avait placé précédemment. L'enfant peut éventuellement classer les jouets selon sa préférence.

Les parents peuvent consulter à tout moment le panier de chacun de leur enfant. Ils peuvent aussi ajuster la somme des dépenses maximum qu'ils désirent effectuer.

Les parents peuvent effectuer toutes les tâches pour un ou plusieurs de leurs enfants.

Le 15 décembre, les enfants ne peuvent plus modifier leur panier.

A partir de ce moment, Mr Noël compose le colis de chaque famille en suivant l'ordre de préférence éventuellement indiqué par les enfants et en respectant la volonté des parents (montant maximum par enfant et mention d'un seul cadeau par enfant).

Il doit ensuite facturer le colis, ce qui rend la facture disponible aux parents.

Monsieur Noel doit constater le paiement du colis avant le 25 décembre sinon il annule la distribution des cadeaux de la famille. Il accepte le paiement en liquide au magasin jusqu'au 24 ou encore une preuve de paiement bancaire.

Monsieur Noel pourra lister tous les colis demandés et tous les cadeaux à faire (par famille ou par type de cadeaux).

Le jour de Noël, Monsieur Noël fait sa tournée de distribution des colis. Quand il rentre chez lui, il indique, pour chaque colis distribué, que la distribution a été faite.

Monsieur Noël pourra lister toutes les familles inscrites par nom et par date d'inscription.

Actuellement, Monsieur Noel est le seul gestionnaire sur le site mais à terme, d'autres membres du magasin pourront l'être également.

Le catalogue en ligne des jouets est disponible à tout internaute mais seuls les parents voient le prix des jouets.

On vous demande de:

1. Compléter le diagramme des **cas d'utilisation**. Présentez les tâches-utilisateur et les sous-fonctions importantes. [6 pts]
2. Tracer le **diagramme d'activités** des UCs qui amènent à la livraison d'un colis à une famille le jour de Noël (d'une année). Vous représenterez les couloirs d'activités des acteurs. Les activités d'inscription, de connexion et de composition de la famille doivent être ignorées. [4 pts]
3. Réaliser le **diagramme de classes conceptuel** du système. Vous ne devez pas y indiquer de méthode. Par contre, n'oubliez pas de placer les attributs, les associations ainsi que les multiplicités. [6 pts]
4. Tracer le **diagramme de séquence** décrit ci-dessous [4 pts].

   Représentez, dans un diagramme de séquence, le scénario principal du cas d'utilisation « composer un colis ». Indiquez les barres d'activation.

   L'acteur principal s'adresse uniquement à la couche UCC qui elle s'adresse au système (en boite noire).

   Le diagramme de séquence est préparé sur les feuilles de réponses.

   Pour composer le colis d'une famille, le gestionnaire va d'abord récupérer la liste des enfants ainsi que le montant maximum par enfant. Ensuite pour chaque enfant, il va récupérer son panier. Le gestionnaire va ensuite parcourir les cadeaux du panier dans l'ordre de préférence. Pour chaque cadeau, il va récupérer son prix et l'ajouter au colis si le montant maximum n'est pas encore atteint pour cet enfant.

   Vous devez indiquer le retour des méthodes appelées.

   Voici les méthodes dont vous disposez :

   UCC

   - la méthode **composerColis(Famille) : Colis** compose le colis pour tous les enfants d'un parent.

   Système

   - la méthode **recupererMontantMaximum(Famille) : double** récupère le montant maximum par enfant
   - La méthode **recupererEnfants(Famille) : List<Enfant>** fournit la liste des enfants du parent
   - La méthode **recupererPanier(Enfant) : List<Cadeau>** renvoie la liste ordonnée par préférence des cadeaux du panier de l'enfant

- La méthode **recupererPrix(Cadeau) : double** renvoie le prix d'un cadeau

Colis

- Le constructeur **Colis(Famille)** permet de créer un colis relatif à la famille d'un parent
- La méthode **ajouterCadeau(Cadeau, Enfant) : double** rajoute un cadeau au colis pour cet enfant. La méthode renvoie le montant des cadeaux du colis pour l'enfant en paramètre.

5. Représenter le **diagramme d'états** d'un colis décrit ci-dessous. N'oubliez pas d'utiliser les cas d'utilisation pour indiquer les changements d'états.

   Représentez, dans un diagramme d'états, les différents états d'un colis.

   - Demandé
   - En préparation (le colis contient au minimum un jouet dans un des paniers)
   - Arrivé à échéance (à partir du 15.12).
   - Composé
   - Facturé
   - Payé
   - Annulé
   - Distribué.

Diagramme des UC de l'exercice Monsieur Noël

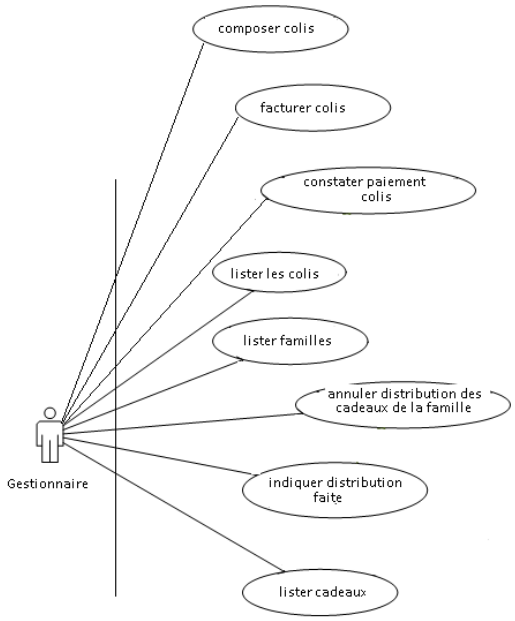
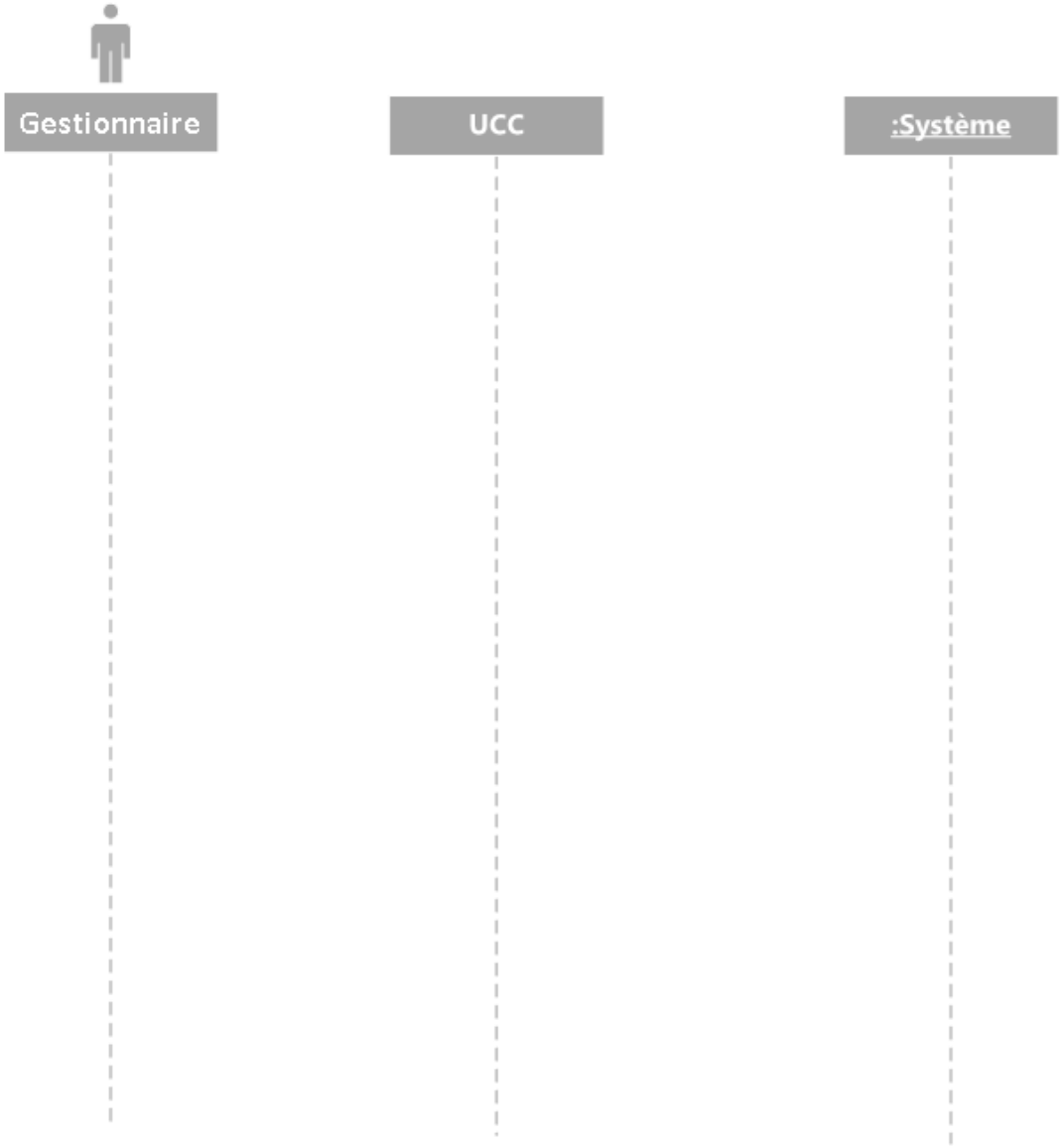
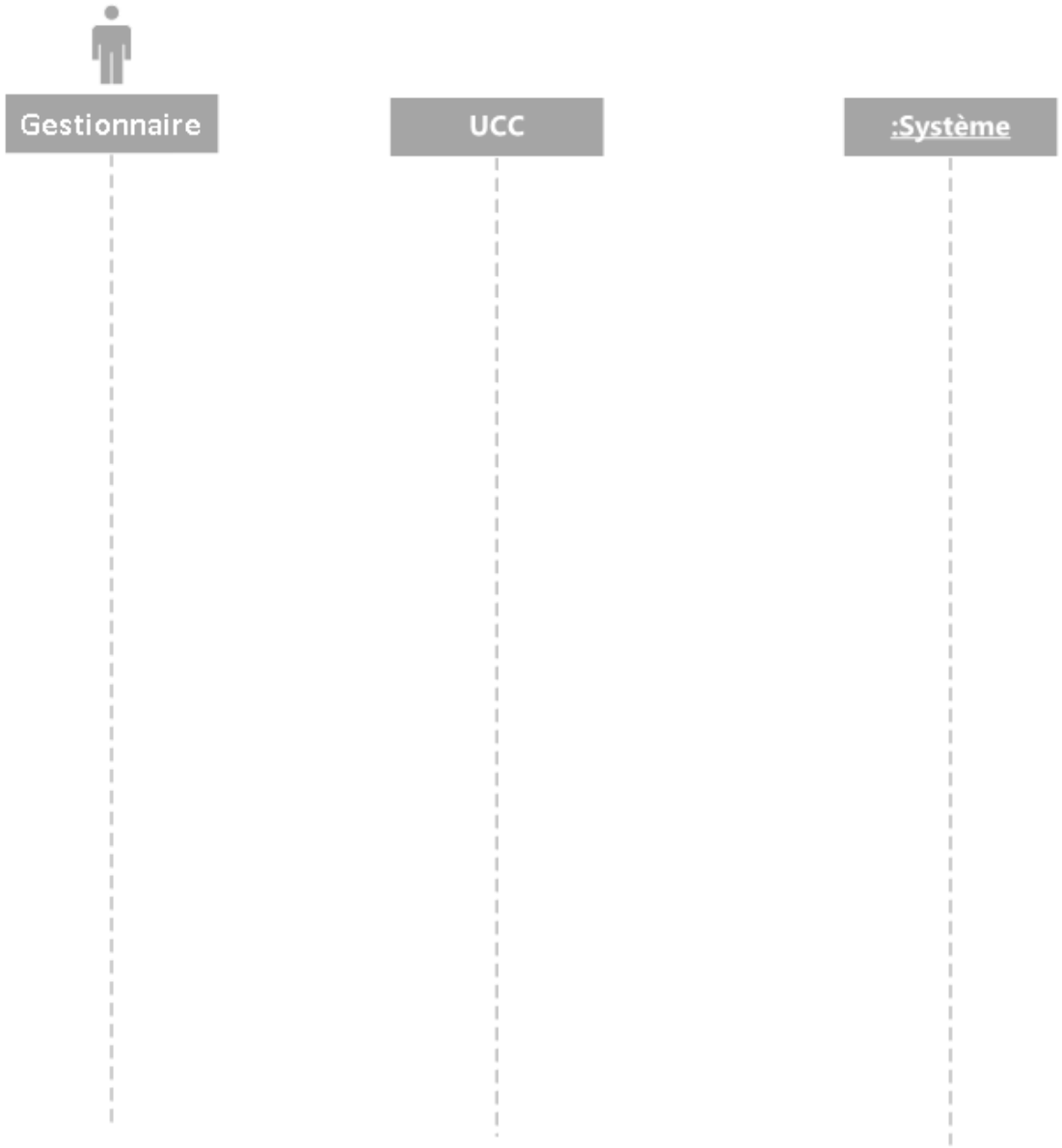


Diagramme de séquences de l'exercice Monsieur Noël




Gestionnaire
UCC
:Système

20.Kinésithérapeutes

Des kinésithérapeutes souhaitent s'associer et unir leurs moyens dans l'investissement d'un bâtiment adéquat pour recevoir au mieux leurs patients. Ils envisagent également la mise en place d'une application qui permettra de gérer au mieux les rendez-vous, les patients et les paiements.

On vous demande de préparer l'analyse de cette application.

Cette application va être utilisée par les kinésithérapeutes et par le secrétariat du cabinet[2]. Pour chacun de ces utilisateurs, il faut connaître son nom, son prénom, son email, son numéro de téléphone professionnel et privé. Pour les kinésithérapeutes, il faut également retenir le numéro INAMI[3].

Les kinésithérapeutes, dans un premier temps, doivent fournir leurs disponibilités (une plage horaire dans la journée) : par exemple de 8h à 13h et de 13h30 à 18h le lundi, de 13h à 17h30 et de 18h à 21h le mardi, etc. Une disponibilité a une période de validité.

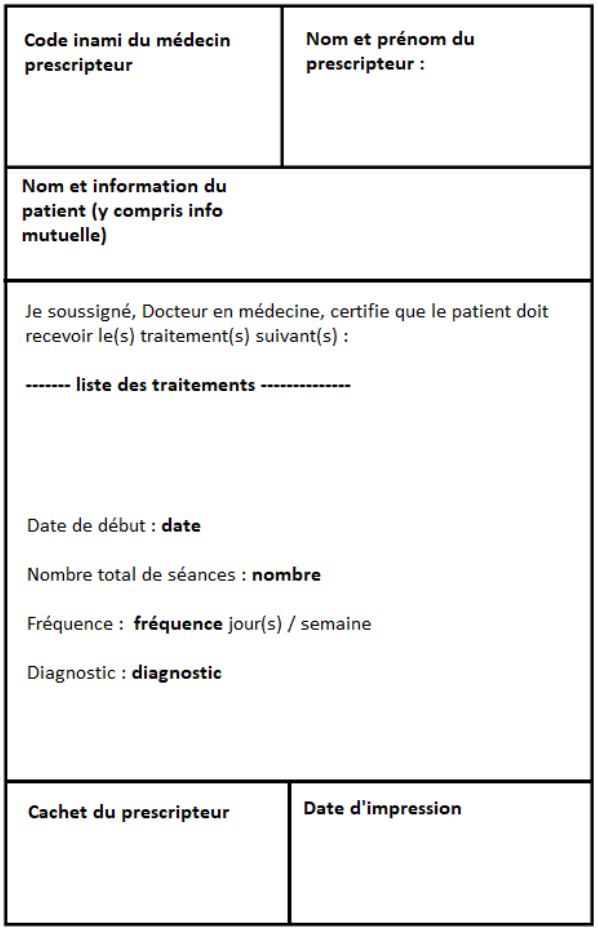
Un rendez-vous a toujours une durée de 30 minutes. Pendant ce temps, le kiné effectue ce que l'on appelle une prestation.

Quand un patient souhaite un rendez-vous, il téléphone au cabinet. Le secrétariat enregistre alors le rendez-vous (rendez-vous pris) auprès d'un kiné et au besoin crée un nouveau patient (nom, prénom et téléphone et/ou email). Le kiné peut également enregistrer un rendez-vous.

Lors du premier rendez-vous avec un patient, le kiné vérifie ses informations : nom, prénom, téléphone, email, date de naissance et son numéro d'identification à la sécurité sociale (NISS). Le système envoie alors au patient un email avec un code d'accès qui lui permet d'accéder à l'application.

C'est un médecin qui prescrit au patient des séances[4] de kinésithérapie. Le kiné doit enregistrer cette prescription dont la structure vous est donnée ci-dessous.

[2] Un cabinet médical est un local (ou un ensemble de locaux) où sont dispensés des soins de santé. Ici, il s'agit du bâtiment acheté par les kinésithérapeutes.
[3] Institut national d'assurance maladie-invalidité
[4] Une séance = un rendez-vous.



Code inami du médecin prescripteur	Nom et prénom du prescripteur :
Nom et information du patient (y compris info mutuelle)	
Je soussigné, Docteur en médecine, certifie que le patient doit recevoir le(s) traitement(s) suivant(s) : **------- liste des traitements ----------** Date de début : **date** Nombre total de séances : **nombre** Fréquence : **fréquence** jour(s) / semaine Diagnostic : **diagnostic**	
Cachet du prescripteur	**Date d'impression**

PRESCRIPTION DE KINESITHERAPIE

Il faut enregistrer les informations importantes que la prescription contient :

- le prescripteur
- les traitements
- la date de début
- le nombre de séances,
- la fréquence des séances,
- le diagnostic
- et enfin la date d'impression.

Pour renseigner un médecin prescripteur, qui n'est pas encore présent dans le système, l'application peut communiquer avec un système intitulé « L'annuaire des médecins de Belgique » qui permet de récupérer les nom, prénom, numéro inami et téléphone de tous les médecins de Belgique.

Par exemple, le Docteur Gravière prescrit à Jean Dupont de suivre 18 séances au rythme de 2 par semaine pour le diagnostic « fracture du métatarse[5] ». Le traitement souhaité est « massage et rééducation fonctionnelle ». La date de début est le 10 décembre 2020, la date d'impression est le 9 décembre.

A la fin de chaque rendez-vous (une séance), le kiné indique que celui-ci a été effectué. Si le patient ne s'est pas présenté, le rendez-vous sera annulé (mais sera quand même facturé, voir ci-dessous).

Dans les deux cas, le kiné doit :

- encoder sa prestation. Une prestation a un numéro de nomenclature[6] et un montant.
- mentionner la prescription pour laquelle le rendez-vous a eu lieu.

Il peut encoder sa prestation ou mentionner la prescription dans n'importe quel ordre mais les deux doivent être réalisés.

[5] Cinq os longs du pied

[6] Le numéro de nomenclature correspond à la prestation effectuée.

A la fin du traitement, (lorsque toutes les séances d'une prescription sont effectuées/annulées) ou plus régulièrement, le kiné délivre une attestation de soins donnés qui regroupe les numéros de nomenclature des prestations par date (document ci-joint).

Le kiné peut donc délivrer une seule attestation de soins pour une prescription, en fin de traitement, ou en délivrer plusieurs tout au long du traitement. Cela est fréquent lorsque le nombre de séances prescrites est élevé.

Sur l'attestation de soins figurent effectivement les dates des différentes prestations ainsi que leur numéro de nomenclature. Les informations du dispensateur (kiné) de soins apparaissent également sur l'attestation. Cette attestation de soins doit être fournie par l'application à la demande du kiné.

Une attestation de soins se rapporte à une seule prescription. Elle a une date et un montant total.

Lorsque l'attestation de soins est fournie, le(s) rendez-vous est/sont indiqués comme facturé(s).

Quand le paiement est effectué, le kiné doit le signaler dans l'application (rendez-vous payé). Le secrétariat peut également signaler quand le paiement est effectué.

L'application doit permettre de gérer l'agenda des kinés : ils peuvent enregistrer des rendez-vous mais ils peuvent également modifier ou supprimer des rendez-vous pris. Le secrétariat également.

Les kinés et le secrétariat doivent pouvoir visualiser tous les patients du cabinet et, pour chaque patient, la liste des rendez-vous passés et futurs. Il doit également être possible de voir si les rendez-vous ont été payés ou pas.

COMPLETER CI-DESSOUS OU APPOSER UNE VIGNETTE DE L'OA

Nom et prénom du patient : .....

Organisme assureur : .....

NISS : .....

Adresse du patient : .....

ATTESTATION DE SOINS DONNES

A COMPLETER PAR LE DISPENSATEUR

Nom et prénom du patient : .....

Date de la prestation	N° de nomenclature		Date de la prestation	N° de nomenclature	
(1)	(1)		(1)	(1)	

Prescrit par : .....
Nom et prénom
en date du : ...../...../.....
Numéro d'identification I.N.A.M.I.
du/des prescripteur(s) : .....
Prescription(s) annexée(s) :
- à la présente (2)
- à l'attestation du ...../...../.....(2)

Le patient est hospitalisé / ambulant (2) :
N° de l'établissement : .....
Service : .....

(1) Barrer les cases non utilisées
(2) Biffer les mentions inutiles

A.R. 15.07.2002

Identification du dispensateur : ..... EUR

G

Date : .....
Signature du dispensateur

REÇU

Perçu pour le compte du N° BCE :

Reçu la somme de : .....EUR Date : .....
Signature

Les patients qui se connectent à l'application peuvent visualiser leurs prescriptions ainsi que les séances déjà effectuées et les séances futures prévues.

Le secrétariat ainsi que les kinés peuvent modifier les informations des patients à tout moment.

Questions

On vous demande de réaliser :

1. Le diagramme des **cas d'utilisation**. Présentez les tâches-utilisateur et les sous-fonctions pertinentes. [6 pts]
2. Le **diagramme de classes conceptuel**. N'y indiquez aucune méthode. Par contre, n'oubliez pas de placer les attributs, les associations ainsi que les multiplicités. Placez également les rôles suivants : maPrestation, monPatient, monPatient, mesPrescriptions. Indiquez la navigation en fonction des rôles donnés (uniquement). [6 pts]
3. Le **diagramme d'activités** de l'enchainement des cas d'utilisation et sous-fonctions pertinentes de ces cas qui permettent de traiter tous les rendez-vous (séances) d'une prescription, de l'enregistrement du premier rendez-vous à la dernière attestation de soins délivrée. Chaque activité est un cas d'utilisation de votre système, tâche-utilisateur ou sous-fonction [4 pts].
4. Au choix [4 pts]:
    a. Le **diagramme d'états** d'un rendez-vous, en décrivant les transitions au moyen des cas d'utilisation. [4 pts]
    b. Le **diagramme de séquences** décrit ci-dessus [4 pts].

Représentez, dans un diagramme de séquence, le scénario principal du cas d'utilisation « délivrer une attestation de soins donnés». Indiquez les barres d'activation.

Le kinésithérapeute s'adresse uniquement à la couche UCC qui elle s'adresse au système (en boite noire).

Pour délivrer une attestation à partir d'une prescription, le système va d'abord créer une nouvelle attestation vide et récupérer toutes les séances liées à la prescription. Ensuite, il va parcourir chacune des séances. Pour toutes les séances qui sont effectuées ou annulées, il va rendre la séance facturée et il va mettre à jour le montant total de l'attestation en ajoutant le montant de la prestation liée à la séance.

Vous devez indiquer le retour des méthodes appelées.

Voici les méthodes dont vous disposez :

UCC

- La méthode **delivrerAttestation(Prescription p) : Attestation** délivre une attestation à partir d'une prescription

Systeme

- La méthode **recupererSeances(Prescription p) : List<Seance>** récupère toutes les séances liées à une prescription
- La méthode **changerEtatSeanceAFacture(Seance s) : void** change l'état d'une séance à « facturé »

Attestation

- Le constructeur **Attestation(Prescription p)** permet de créer une attestation vide pour une prescription. La date de l'Attestation créée sera mise à la date du jour et le montant total à 0.
- La méthode **ajouterMontantPrestation(Seance s) : void** modifie le montant total de l'attestation en ajoutant le montant de la prestation de la séance.