Correction du devoir de Cours sur UML

1) Le concept

a- Qu'est ce que c'est qu'un modèle ? Appuyez vous sur un exemple d'un autre domaine que la modélisation UML, en mettant en avant l'avantage de modéliser un problème.

Un **modèle** est une **représentation abstraite** d'un système, qui facilite l'étude et la communication entre intervenants au sein d'un projet.
Abstraction : ensemble des caractéristiques essentielles d'une entité, retenues par un observateur

Autre modèle : Modèle économique. A partir d'hypothèses macro-économiques (évolution du chômage, taux de croissance...), on créé un modèle qui permet de simuler l'évolution de cours boursiers

b- Quel est la différence entre une vue statique et une vue dynamique ?

Une vue statique permet de représenter la **structure** du modèle sans tenir compte de l'évolution au cours du temps. Une vue dynamique représente au contraire les **changements** qui interviennent au cours du temps.

2) Les cas d'utilisation

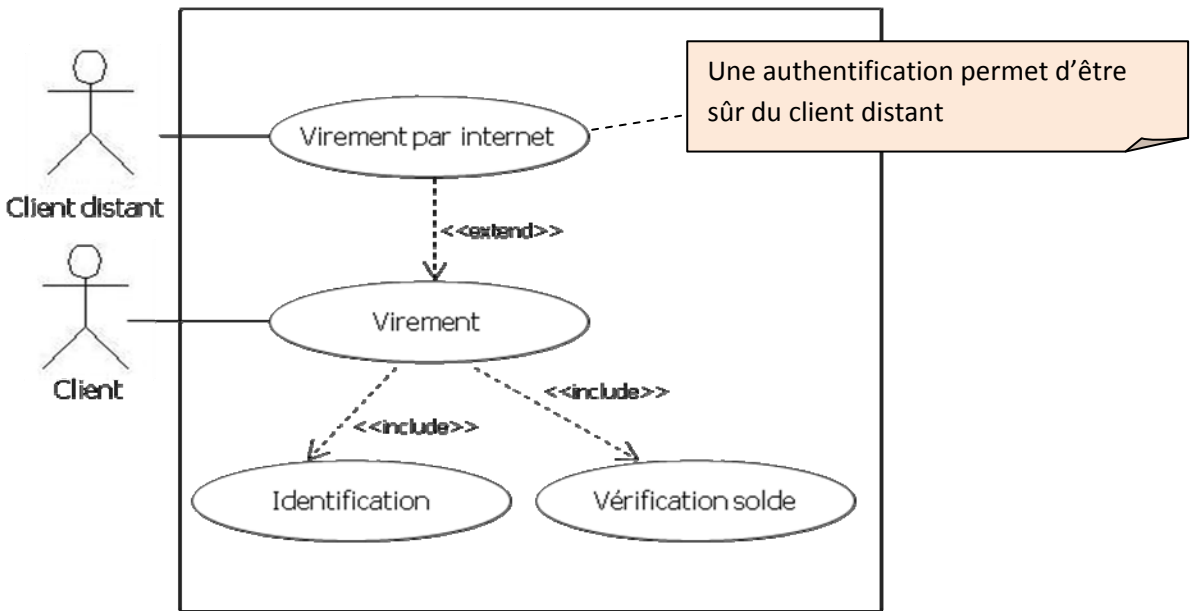
a- Que cherche-t-on à modéliser avec un diagramme de cas d'utilisation ?

Expression du **comportement du système** (actions et réactions), selon le **point de vue de l'utilisateur.**

b- Quel est l'intérêt de ce diagramme ?

Permet de délimiter les frontières du système
Constitue un moyen d'exprimer les besoins d'un système
Utilisé par les utilisateurs finaux pour exprimer leurs attentes et leurs besoins
Permet d'impliquer les utilisateurs dès les premiers stades du développement
Constitue une base pour les tests fonctionnels

c- Donner un exemple de ce diagramme avec (sur le même schéma) un include, un extend, et un commentaire. Expliquez en 2-3 phrases le schéma que vous avez proposé.



3) Donner le nom de ce diagramme et expliquez brièvement de qu'il représente :

Le piège du devoir !
En effet, ce n'est pas un diagramme de collaboration, mais la modélisation d'une collaboration dans les diagrammes de classe. Cela permet de représenter quelles classes travaillent à un cas d'utilisation. Il représente les 3 classes qui participent au cas d'utilisation « vente de véhicule ».

4) Diagramme de classe

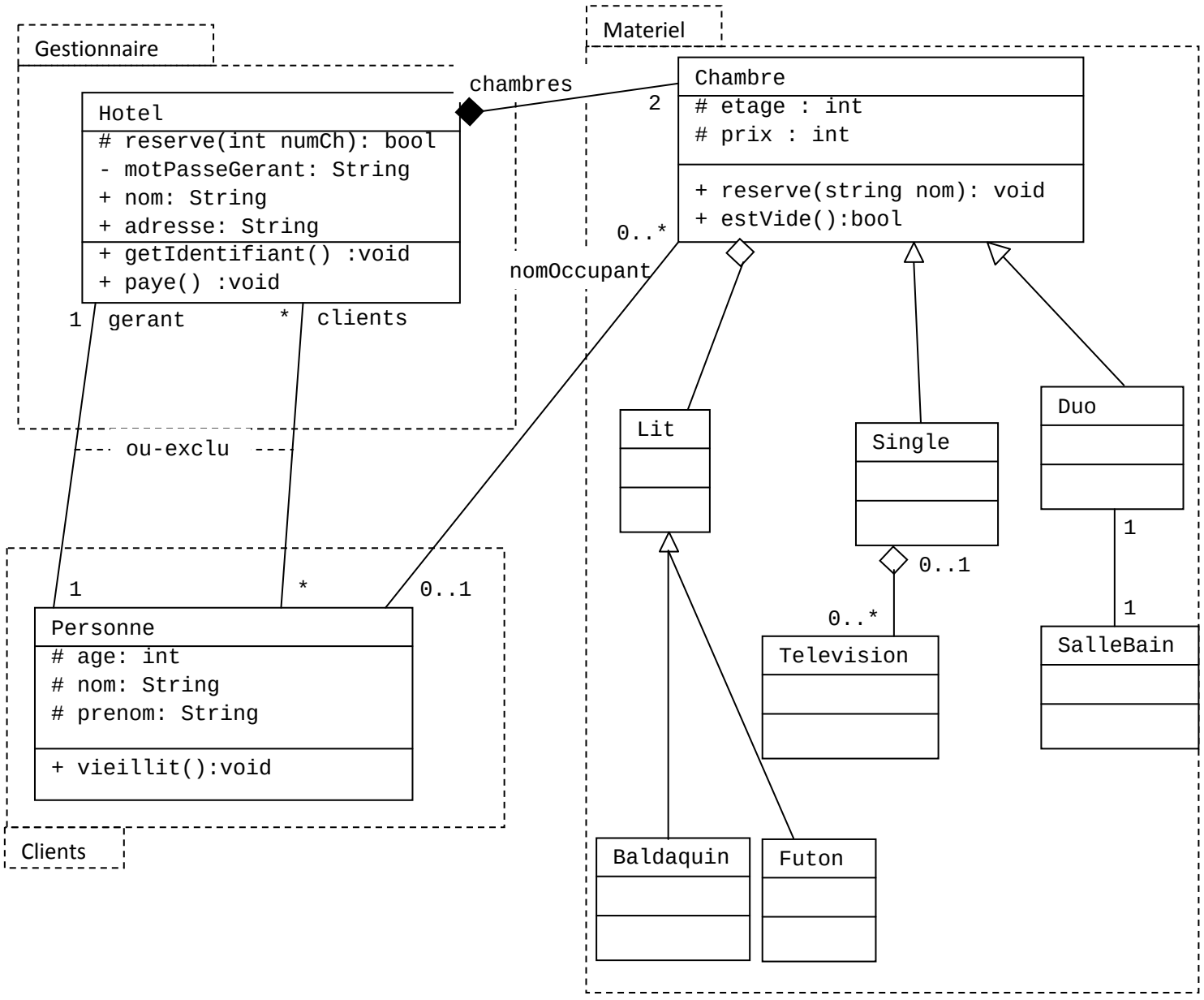
a- En utilisant le maximum de détail, représenter ces classes en utilisant les diagrammes de classe (prévoyez une page entière, vous devrez ajouter des trucs)

b- Ajoutez les classes suivantes (les noms en gras) : Les chambres Single ont une ou plusieurs Télévision et les chambres Duo ont une salle de bain. Par contre toutes les chambres ont un Lit, et il existe deux type de lit : des lits à Baldaquin, et des lits Futon.

c- Essayez de faire figurer une agrégation et une composition, en expliquant votre choix.

d- Représentez la phrase suivante sur le diagramme : une personne est soit un client de l'hôtel, soit le gérant.

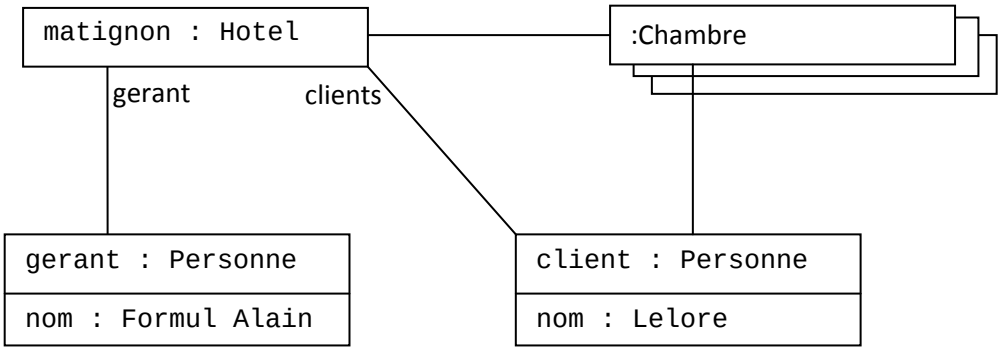
e- En utilisant la notion de package, séparez en trois groupe distinct les classes.



5) Diagramme d'objet

a- En vous basant sur le précédant diagramme de classe faite un diagramme d'objet qui représente cette situation :

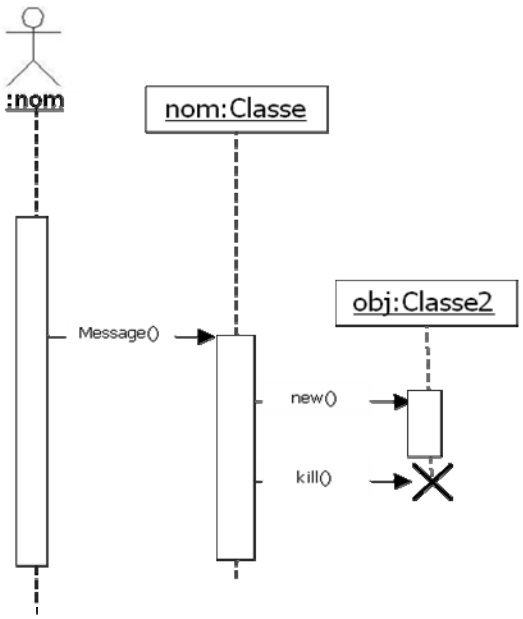
L'hotel « matignon » dont le gérant Mr « Formul Alain » s'occupe, possède 50 chambres. L'une des chambres est louée à Mr « Lelore ».



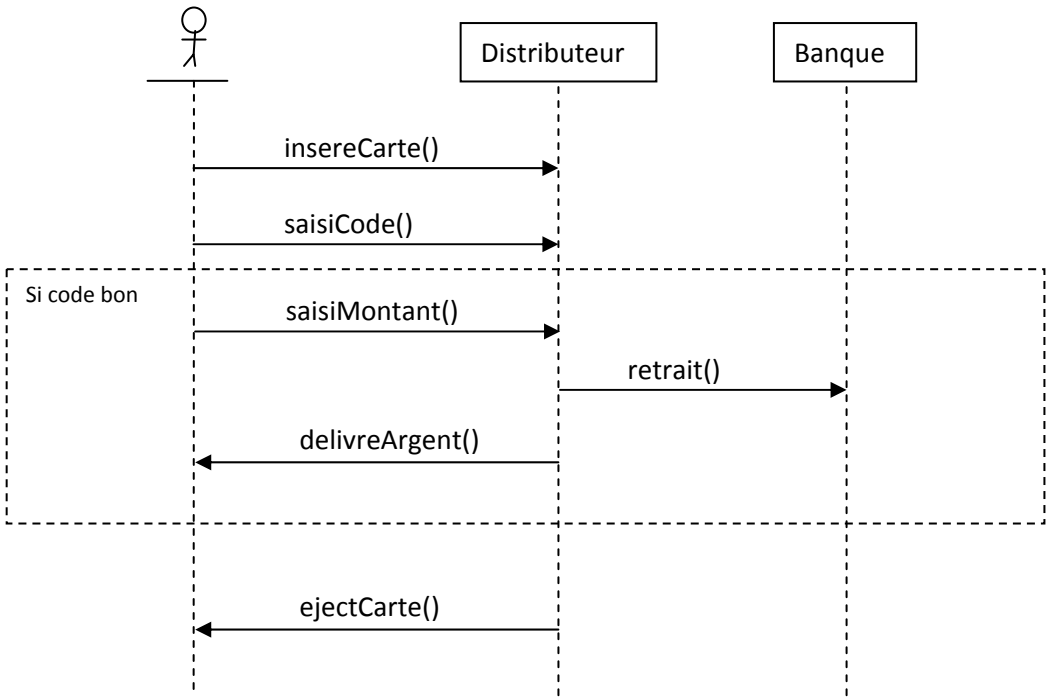
6) Diagramme de séquence

a- Donnez un exemple (pas obligatoirement basé sur un fait réel) présentant les différentes notions : Acteur, objet, ligne de vie, bande d'activation, envoi de message, réponse, création dynamique, suppression d'un objet

- **Acteur :**
- **Objet :**
- **Ligne de vie :**
- **Bande d'activation :**
- **Envoi de message :**
- **Création dynamique :**
- **Supprimer un objet :**



b- Donnez un exemple (obligatoirement basé sur un fait réel) présentant le principe du branchement conditionnel.



7) Diagramme de collaboration

a- Expliquez ce que ces messages font :

`[heure = midi] 1 : manger()`

Ce message « **`manger()`** **»** n'est envoyé que s'il est midi.

`1 / *|| 2.1 : fermer()`

Ce message « fermer() » n'est envoyé qu'une fois le message 1 fini, et il est envoyé en parallèle (en même temps) sur tous les récepteurs (||) un nombre inconnu de fois (*).

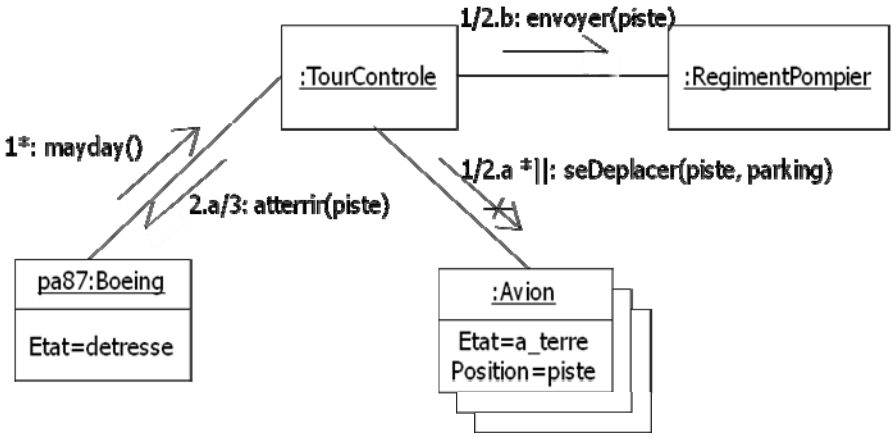
`1.3,2.1 / [t < 10s] 2.5 : age := demanderAge(nom,prenom)`

Ce message n'est envoyé qu'une fois les messages 1.3 et 2.1 fini, et uniquement si **`t`** est inferieur à **`10s`**. De plus, la fonction **`demanderAge`** prend deux paramètres **`(nom,prenom)`** et revoie une valeur dans la variable **`age`**.

```
1.3 / [disk full] 1.7.a * : deleteTempFiles()
1.3 / [disk full] 1.7.b : reduceSwapFile(20%)
```

Les deux messages sont envoyés en même temps (c'est le .a et le .b qui permet de le voir) après que le message 1.3 soit envoyé et que la condition **`disk full`** soit vrai.

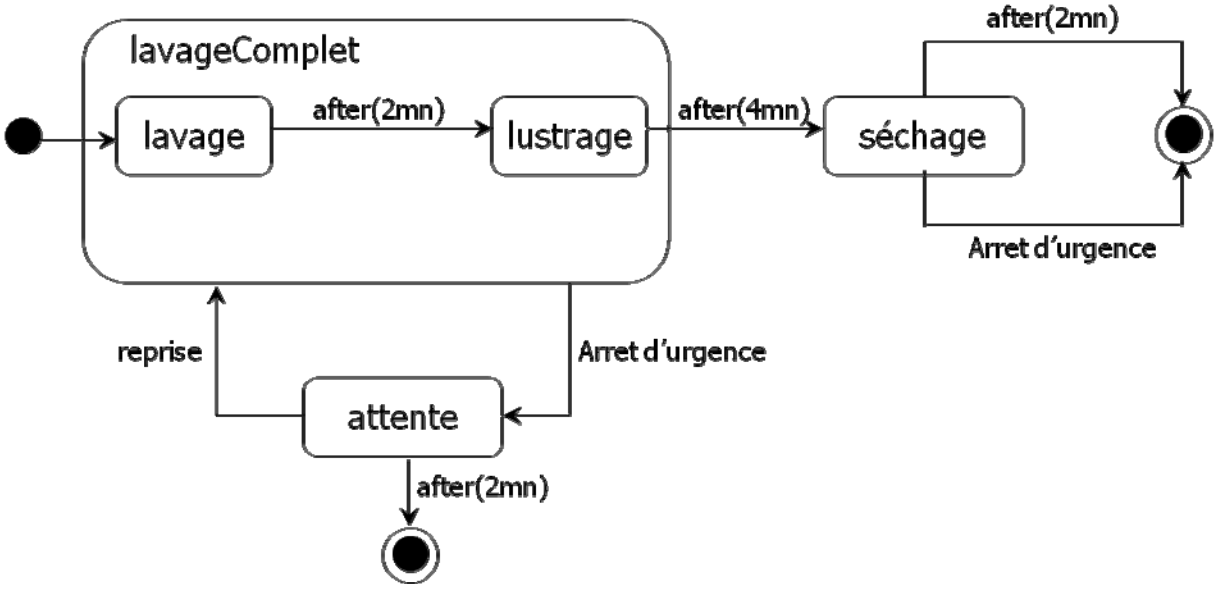
Expliquer ce shéma :



L'objet pa87 dont l' etat=detresse envoie un message à la tour de contrôle de manière répétée un nombre inconnue de fois. Ce message est synchrone, c'est-à-dire que l'émetteur est synchronisé avec la tour (par un dialogue genre « appel tour de contrôle ; tour de contrôle écoute ; mayday ; bien reçu… »).

8) Diagramme d'états transitions

a- Imaginez un système que vous modéliserez avec un diagramme d'état transition. Il faut que votre diagramme contienne au minimum 3 états, et des événements qui déclenchent des actions.

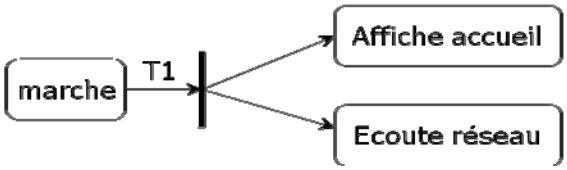


b- Il existe différentes manières de déclencher une action dans un état : quand on arrive, quand on est dedans, quand on en sort et tant qu'on reste dans l'état. Quels sont les différents mots clés associés en UML ?

- **entry / *action*** : action exécutée **à l'entrée** de l'état
- **exit / *action*** : action exécutée **à la sortie** de l'état

- **on *événement* / *action*** : action exécutée **à chaque fois** que l'événement cité survient
- **do / *action*** : action **récurrente** ou significative, exécutée dans l'état

c- Comment représenter qu'un objet se trouve dans deux états en même temps ? En utilisant cette manière, représentez l'état d'un ordinateur quand vous cliquez sur firefox : il est dans l'état marche, et il passe dans l'état « écoute le réseau » et « affiche page accueil ».



9) Diagramme de composant

Que pouvez-vous dire de ce schéma (à quoi ça sert, qu'est ce que ça représente…) :

- ► permet de décrire l'architecture physique et statique d'une application en termes de modules : fichiers sources, librairies, exécutables, etc.
- ► montre la mise en œuvre physique des modèles de la vue logique avec l'environnement de développement.

On voit ici que pour faire l'exécutable « **`bancDeMesuresTask`** », on a besoin de 3 objets. Pour faire l'objet « **`BancDeMesures.obj`** », on a besoin de « **`BancDeMesures.cpp`** » qui a lui-même besoin de 3 fichiers : « **`BancDeMesures.h`** », « **`Bobine.h`** » et « **`Multimetre.h`** »…

10) Diagramme de déploiement

Que pouvez vous dire de ce schéma (à quoi ça sert, qu'est ce que ça représente…) :

- ► Montrent la disposition physique des matériels qui composent le système et la répartition des composants sur ces matériels
- ► Les ressources matérielles sont représentées sous forme de nœuds
- ► Les diagrammes de déploiement peuvent montrer des instances de nœuds (un matériel précis), ou des classes de nœuds