

Bachelier en Informatique

De Gestion

Bloc 2 Q 2

Analyse et conduite de projets

Année 2019-2020

**Introduction**

****

**Définir les mots suivants :**

Besoins :

Le Client veut connaître une disponibilité.

Sécurisation des données.

Connexion Rapide.

Spécifié congé

Fonctions :

Consultation de la liste des Rendez Vous.

Connexion sécurisé.

Cahier des charges

Modèle

UML

Le recours à la modélisation est essentiel pour l’étude d’un système d’information.  
Ainsi, pour concevoir un nouveau système, on a besoin de représenter les différentes facettes de ce qui existe déjà (« as is ») et de ce vers quoi l’on tend (« to be ») : c’est la raison pour laquelle il est utile d’utiliser des modèles.  
  
De plus, les exigences que devra satisfaire la nouvelle application sont souvent ambiguës, incomplètes ou mal exprimées. Les gestionnaires et les futurs utilisateurs sont donc invités, avec l’assistance d’experts, à travailler rigoureusement sur la définition des services attendus pour élaborer un cahier des charges du système d’information, sur lesquels les informaticiens s’appuieront pour construire le système informatique. La modélisation, correctement utilisée, améliore la qualité du cahier des charges.

**Chapitre 1 : L’aspect fonctionnel**

Le diagramme des cas d’utilisation est une technique de description du système étudié privilégiant ***le point de vue de l’utilisateur***. Un cas d’utilisation est une façon spécifique d’utiliser le système ; il représente la ***relation entre un acteur et une fonctionnalité du système***

Ils défi­nissent les limites du système et ses relations avec son environnement.

**Acteur**

Un utilisateur externe du système peut jouer différents rôles vis-à-vis du système. Un couple (utilisateur, rôle) constitue un acteur spécifique désigné en UML uniquement par le nom du rôle. Cette définition est étendue aux autres systèmes qui interagissent avec le système. Ils forment autant d’acteurs qu’ils jouent de rôle.

**Scénario**

Un scénario est une instance d’un cas d’utilisation dans laquelle toutes les conditions relatives aux différents événements ont été fixées. Il n’y a donc pas d’alternatives lors du déroulement.

À un cas d’utilisation donné correspondent plusieurs scénarios.

Comme une classe qui détient les aspects communs de ses instances, un cas d’utilisation décrit de façon commune l’ensemble de ses scénarios en utilisant des branchements conditionnels pour représenter les différentes alternatives.

**Relation de communication**

La relation qui lie un acteur à un cas d'utilisation s'appelle la relation de communication.

Cette relation supporte différents modèles de communication, par exemple :

- les services que le système doit fournir à chacun des acteurs du cas d'utilisation ;

- les informations du système qu'un acteur peut introduire, consulter ou modifier ;

**Diagramme Des Cas D’utilisations**

Ce dernier montre les cas d’utilisation représentés sous la forme d’ovales et les acteurs sous la forme de personnages. Il indique également les relations de communication qui les relient.

Le système est représenté sous la forme d’un rectangle englobant le cas

**Etude de cas : le GAB**

**Introduction**

La description des cas d'utilisation va nous permettre de préciser les frontières du système et de détailler les différentes façons dont les acteurs vont pouvoir utiliser ce système.

**Enoncé**

Cette étude concerne un système simplifié de guichet automatique de banque (GAB). Le GAB offre les services suivants :

1. Distribution d'argent à tout porteur de carte de crédit (carte Visa ou carte de la banque), via un lecteur de cartes et un distributeur de billets.
2. Consultation de solde de compte, dépôt en numéraire et dépôt de chèques pour les clients de la banque porteurs d'une carte de crédit de la banque.

3. Toutes les transactions sont sécurisées.

4. Il est parfois nécessaire de recharger le distributeur

**Etapes**

1. Identification des acteurs du GAB

1. Les 4 phrases nous permettent de retrouver les entités externes qui interagissent directement avec le GAB.

La représentation des acteurs en interaction avec le système peut se faire à l'aide du *diagramme de contexte statique :* chaque acteur est relié à une classe centrale représentant le système par une association, ce qui permet de spécifier le nombre d'instances d'acteurs connectés au système à un moment donné.

1. Dans la deuxième solution, nous pourrons considérer que le client de la banque est une *spécialisation* de porteur de carte bancaire.

2. Identification des cas d'utilisation

Un *use case* représente un ensemble de séquences d'actions réalisées par le système, qui produisent un résultat observable intéressant pour un acteur particulier. Il modélise un service rendu par le système. Il exprime les interactions acteurs/système et apporte une valeur ajoutée à l'acteur concerné.

1. Préparez une liste préliminaire des cas d'utilisation du GAB par acteur.
2. Distinguez les acteurs primaires des acteurs secondaires.

3. Les diagrammes des cas d'utilisation

1. La réponse obtenue à l'étape 2 est transposée sur un schéma qui montre les use case (ovales) reliés par des associations (lignes) à leurs *acteurs principaux*

(<< stick man »).

1. Modifions le diagramme des use case en tenant compte de la relation de *spécialisation* entre client de la banque et porteur de carte visa (cette solution ne se sera pas retenue par la suite).
2. Complétez le diagramme des cas d'utilisation en ajoutant les *acteurs secondaires.*

Le cas d'utilisation partagé *Retirer de l'argent pose problème :* si l'acteur principal est porteur de carte visa, il faudra faire appel à la société SA Visa ; s’il s'agit d'un client de la banque, le GAB contactera directement le SI banque.

4. Description textuelle des cas d'utilisation

Une fois les Use cases déterminés, il faut les décrire !

La structuration suivante est préconisée pour la fiche de description textuelle :

1. Le sommaire d'identification (obligatoire)
2. La description des enchaînements (obligatoire) : un *enchaînement* est une unité de description de séquence d'actions. Un *scénario* représente une succession particulière d'enchaînements, s'exécutant du début à la fin du cas d'utilisation (on distingue les scénarios nominatifs, alternatifs et d'exception).
3. Les besoins d'IHM (optionnel) : ajoute les contraintes d'interface homme-/machine, c'est-à-dire ce qu'il faut montrer à l'utilisateur (écrans, …)
4. Les contraintes non fonctionnelles (optionnelles) : fréquence, disponibilité, fiabilité, confidentialité, ...

5. Description graphique des Use cases

La description textuelle des use cases présentent des désavantages puisqu'il est difficile de montrer comment les enchaînements se succèdent, ou à quel moment interviennent les acteurs secondaires. La description textuelle sera complétée par un ou plusieurs diagrammes dynamiques UML (diagramme d'activités, diagramme de séquence système).

1. Réalisez un *diagramme de séquence* qui décrit le scénario nominal du cas d'utilisation retirer de l'argent avec une carte visa.
2. Contrairement au diagramme de séquence qui ne décrit que le scénario nominal, *le diagramme d'activités* doit représenter l'ensemble des activités représentées par le système.

Réalisez un diagramme d'activités qui décrit la dynamique du cas d'utilisation retirer de l'argent avec une carte visa.

6. Organisation des cas d'utilisation

Dans cette dernière étape, nous allons affiner les diagrammes et les descriptions en utilisant : les *relations d'inclusion,* les *relations d'extension et de généralisation,* le *regroupement en packages.*

1. Les relations d'inclusion ou relation d'utilisation (uses) : le cas de d'utilisation de base en incorpore *obligatoirement* un autre à un endroit spécifié dans ses enchaînements ; cette tec11.L'lÎque évite de décrire plusieurs fois le même enchaînement en factorisant le comportement commun dans un use case à part.

Identifiez une partie commune aux différents cas d'utilisation et factorisez-la dans un nouveau cas inclus dans ces derniers.

1. La relation d'extension ( extends) : cette relation sert à séparer un comportement *optionnel* ou *rare* du comportement obligatoire.

En extrapolant sur les besoins initiaux, identifiez une relation d'extension entre deux cas d'utilisation du client de la banque,

1. La relation de généralisation : les cas d'utilisation descendants *héritent* la description de leur parent commun. On utilise cette relation pour formaliser des variations importantes sur le même cas d'utilisation.

Identifiez une relation de généralisation qui implique deux cas d'utilisation du client de la banque.

d. Le package: est un mécanisme général de *regroupement* en UML.

Définissez le diagramme des cas d'utilisation complet. Quels défauts présentent ce diagramme ?

Proposez une structuration des cas d'utilisation du GAB en packages, élaborez ensuite un diagramme de cas d'utilisation par package.

**Conclusion**

Les cas d’utilisation sont intéressants quand les utilisateurs sont amenés à intervenir : ***expression des besoins, modélisation du futur système***, test ***et validation et mise en œuvre.***

**Les scénarios**

***Identification : retirer de l’argent avec une carte visa.***

Scénario nominal :

1. le porteur de carte visa introduit sa carte dans le lecteur de cartes GAB.
2. le GAB vérifie que la carte introduite est bien une carte visa
3. le GAB demande au porteur de CB visa de saisir son code d’identification.
4. le porteur de carte visa saisit son code d’identification.
5. le GAB compare le code d’identification avec celui qui est codé sur la puce de la carte.
6. le GAB demande une autorisation au système visa
7. le système visa donne son accord et indique le solde hebdomadaire.
8. le GAB demande au porteur de CB d’introduire le montant désiré du retrait
9. le porteur de CB saisit le montant désiré du retrait
10. le GAB contrôle le montant demandé par rapport au solde hebdomadaire
11. le GAB rend sa carte au porteur de CB visa
12. le porteur de CB visa reprend sa carte visa
13. le GAB délivre les billets et un ticket
14. le porteur de CB visa prend les billets et le ticket

Enchaînements alternatifs

* Le code d’identification est provisoirement erroné
* Le GAB indique au client que le code est erroné pour la deuxième et troisième fois. Le porteur a droit à trois essais.
* Le montant demandé est supérieur au solde
* le GAB indique au client que le montant demandé est supérieur au solde hebdomadaire (retour au point 3)

**Enchaînements d’exception**

* La carte d’identification n’est pas valide.
* le GAB indique que la carte n’est pas valide et la confisque
* Le code d’identification définitivement erroné
* Le GAB indique au client que le code est erroné pour la troisième fois
* Le GAB confisque la carte
* Le GAB informe le système d’autorisation visa

**Chapitre 2 : l’aspect statique**

**La forme simplifiée de la représentation des classes**  
Les objets du système sont décrits par des classes dont une forme simplifiée de la représentation UML est donnée à la figure suivante :

|  |
| --- |
| NomClasse |
| nomAttribut1 |
| nomAttribut2 |
| nomAttribut3 |
| nomMéthode() |
| nomMéthode() |

La première partie contient le nom de la classe, la seconde représente les attributs qui contiennent l’information portée par l’objet et la troisième partie comporte les méthodes qui correspondent aux services offerts par l’objet. Elles peuvent modifier la valeur des attributs. L’ensemble des méthodes constitue le comportement de l’objet.

**Les associations entre objets**

**les liens entre les objets** : dans le monde réel, de nombreux objets sont liés entre eux. Ce lien correspond à une association entre les objets  
  
exemples :  
  
  
  
  
**La représentation des associations entre les classes.**

* La représentation d’une association de type binaire :  
    
  Le nom d’une association est indiqué au-dessus du trait.  
  Chaque extrémité d’une association peut être nommée. Ce nom est significatif du rôle que jouent les instances de la classe correspondante dans l’association.

* La représentation graphique d’une association ternaire :
* La cardinalité d’une association : indique à combien d’instances de classe située à cette extrémité, une instance située à l’autre extrémité est liée  
    
    
  Exemples :

|  |  |
| --- | --- |
| **Spécification** | **Cardinalités** |
| 0 .. 1 | zéro ou une fois |
| 1 | une et une seule fois |
| \* | de zéro à plusieurs fois |
| 1..\* | de un à plusieurs fois |
| M .. N | entre M et N fois |
| N | N fois |

* La Navigation:  
    
  Les associations ont par défaut une navigation bidirectionnelle, c'est-à-dire qu’il est possible de déterminer les liens de l’association depuis une instance de chaque classe d’origine.

**Les classes associations**  
Les liens entre les instances des classes peuvent porter des informations. Dans ce cas, l’association qui décrit de tels liens reçoit le statut de classe et peut être dotée d’attributs et d’opérations

**Les qualificateurs :**

En cas de cardinalité maximale non finie à une extrémité d’une association, si les instances situées à cette extrémité sont qualifiables, il est possible d’utiliser cette qualification pour passer de la cardinalité maximale non finie à une cardinalité maximale finie.

**Exercice 1**

Modélisez les phrases suivantes :

* un répertoire contient des fichiers
* les modems et les claviers sont des périphériques d’entrée/sortie
* un compte bancaire peut appartenir à une personne physique ou morale
* un pays comprend des villes dont l’une est la capitale
* Un message électronique est composé d’un entête, d’une adresse de destinataire, d’un corps et de fichiers attachés.
* le but est de décrire la gestion d’une connexion utilisateur à un serveur de bases de données. L’utilisateur se connecte avec un nom de compte et un mot de passe. La date et l’heure de la connexion doivent être stockées. Une connexion dispose d’un espace de travail.

**Exercice 2 définir le diagramme de classes correspondant à l’énoncé**

Un hôtel est composé d’au moins deux chambres.

Chaque chambre dispose d’une salle d’eau qui peut être une douche ou une salle de bain. L’hôtel héberge des personnes. Il peut employer du personnel et est dirigé par un des employés. L’hôtel a les caractéristiques suivantes :

* Une adresse, le nombre de pièces, la catégorie.
* Une chambre est caractérisée par le nombre et le type de lits, le prix et le numéro. On peut calculer le chiffre d’affaires, le loyer en fonction des occupants….

**Etude de cas : le système de réservation des vols (tiré du livre UML par la pratique de Pascal Roques)**

Introduction

Dans une première étape, nous avons étudié la modélisation d'un point de vue fonctionnel (U se Cases).

A travers une deuxième étape, nous avons commencé à étudier la modélisation d'un point de vue statique (diagrammes de classes ou diagrammes d'objets)

Une troisième étape nous permettra de découvrir la modélisation d'un point de vue dynamique (diagrammes d'états, diagrammes d'activités.

L'exemple « Le système de réservation des vols » illustre le modèle statique

Enoncé

Cette étude de cas concerne un système simplifié de réservation des vols pour les agences de voyages.

Les interviews des experts métier auxquelles on a procédé ont permis de résumer leur connaissance du domaine sous la forme des phrases suivantes :

1. **Des compagnies aériennes proposent différents vols**
2. **Un vol est ouvert à la réservation et refermé sur ordre de la compagnie**
3. **Un client peut réserver un ou plusieurs vols pour des passagers différents**
4. **Une réservation concerne un seul vol et un seul passager**
5. **Une réservation peut être annulée ou confirmée**
6. **Un vol a un aéroport de départ et un aéroport d'arrivée**
7. **Un vol a un jour et une heure de départ, et un jour et une heure d'arrivée**
8. **Un vol peut comporter des escales dans des aéroports**
9. **Une escale a une heure de départ et une heure d'arrivée**
10. **Chaque aéroport dessert une ou plusieurs villes.**

Etapes

1. Identification des classes
2. Modélisation des phrases 1 et 2
   1. Sur quelle classe s’effectuent les traitements d’ouverture et de fermeture ?
   2. Les traitements d’ouverture et de fermeture sont réalisés sur les vols : ce sont des opérations de la classe vol. Ces opérations seront déclenchées par des messages envoyés par la compagnie aérienne.
3. Modélisation des phrases 6, 7 et 10   
   1. Quand faut-il utiliser un attribut ? Quand une notion doit-elle être représentée par un objet ? Si l’on ne peut demander à un élément que sa valeur, il s’agit d’un simple attribut ; si plusieurs questions s’y appliquent, il s’agit plutôt d’un objet.
   2. Comment pourrait-on représenter le fait qu’un aéroport joue deux rôles: point de départ et point d’arrivée?
4. Modélisation des phrases 8 et 9
   1. Lever l’ambiguïté contenue dans la phrase 8 : une escale a lieu dans un seul aéroport et un aéroport peut servir à plusieurs escales ; une escale peut appartenir à des vols différents
   2. 1 ère solution : la modélisation des deux phrases comporte une agrégation entre vol et escale
   3. 2ème solution : la classe escale comporte peu d’attributs propres et peut être considérée comme une spécialisation de la classe aéroport. Peut-on dire qu’une escale est un aéroport à part entière ?
   4. 3ème solution : on considère l’escale comme une classe association entre vol et aéroport.
5. Modélisation des phrases 3, 4 et 5   
   1. La distinction entre « passager » et « client » est-elle justifiée ?
   2. Compléter le schéma avec les attributs indispensables et quelques attributs dérivés. Ajouter des contraintes et une association qualifiée.
6. Amélioration du modèle   
     
   La classe vol possède trop de responsabilités: d’une part, elle gère tout ce qui se retrouve dans les catalogues des compagnies aériennes et d’autre part tout ce qui touche aux réservations.   
   Il faut donc répartir les attributs, les opérations et les associations entre les deux classes « v*ol »* et « *vol générique »* et ajouter une association entre ces deux classes.
7. Les packages   
     
   La structuration d’un modèle conceptuel s’appuie sur deux principes fondamentaux: **cohérence et indépendance.**Le second principe consiste à renforcer le découpage initial en s’efforçant de **minimiser les dépendances** entre packages (suppression des dépendances mutuelles).   
   Le premier consiste à **regrouper les classes proches d’un point de vue sémantique** à partir des critères suivants:   
   1. Finalité : les classes doivent rendre des services de même nature aux utilisateurs.
   2. Evolution: on isole les classes réellement stables de celles qui vont vraisemblablement évoluer au cours du projet ou par la suite
   3. Cycle de vie des objets : ce critère permet de distinguer les classes dont les objets ont des durées de vie très différentes.
8. La structuration de l’exemple en packages   
   1. Découper le modèle conceptuel en deux packages (les classes de chaque package sont fortement liées et les classes des deux packages sont presque indépendantes)
   2. Pourrait-on envisager un autre partage dans notre modèle ?
   3. Trouver une solution qui permette de minimiser le couplage entre les deux packages pour les modèles développés aux points a et b.
9. Réalisation du schéma complet
10. Généralisation et réutilisation   
    Nous souhaitons élargir le champ du modèle en proposant également des voyages en bus, que des transporteurs assurent.   
    Un voyage en bus a une ville de départ et une ville d’arrivée, avec des dates et des heures associées. Le trajet peut comporter des arrêts dans des villes intermédiaires. Un client peut réserver un ou plusieurs voyages, pour un ou plusieurs passagers.
    1. Proposer, par analogie avec le cas précédent, un modèle de la réservation des voyages en bus.   
       Les deux modèles présentent de nombreuses similitudes:   
       certaines classes sont communes aux deux modèles (ex: la classe client) et d’autres ont des points communs (ex: RéservationAvion et RéservationBus)
    2. Proposer une architecture logique fusionnée qui soit la plus évolutive possible.   
       Deux tâches principales doivent être réalisées:   
       Isoler les classes communes dans de nouveaux packages afin de pouvoir les réutiliser et factoriser les propriétés communes dans les classes abstraites.

**Exercice récapitulatif en UML : la demande de formations**

**Le modèle fonctionnel**

Dans le cadre de l'amélioration qu'elle veut apporter à son système d'information, une entreprise souhaite modéliser dans un premier temps, le processus de formation de ses employés afin que quelques-unes de leurs tâches soient informatisées.

1. Le processus de formation est informatisé lorsque le responsable de formation reçoit une demande de la part de l'employé. Cette demande est instruite par le responsable qui la qualifie et donne son accord ou son désaccord à l'intéressé.
2. En cas d'accord, le responsable recherche dans le catalogue des formations agréées un stage qui correspond à la demande. Il informe l'employé du contenu de la formation et lui propose une liste des prochaines sessions. Lorsque l'employé a fait son choix, le responsable formation inscrit le participant à la session auprès de l'organisme concerné.
3. En cas d'empêchement, l'employé doit informer le responsable de formation -au plus tôt pour annuler l'inscription ou la demande.
4. A la fin de sa formation, l'employé doit remettre au responsable de sa formation une appréciation sur le stage qu'il a effectué, ainsi qu'un document justifiant de sa présence.
5. Le responsable formation contrôle par la suite la facture que l'organisme de formation lui a envoyée avant de la transmettre au comptable achats.

**Etape 1 :** **la démarche générale**

Quelles étaient les grandes étapes suivies dans la démarche d'analyse fonctionnelle du cas « GAB » ? Quel était le but recherché ?

Dans le cas présent, quel est le but recherché par l'étude fonctionnelle ?

En quoi la démarche du cas « demande de formation » est -elle différente de la démarche du cas « GAB » ?

**L’aspect statique**

1. Le processus de formation est initialisé lorsque le responsable de formation reçoit une demande de formation de la part de l'employé.
2. Cette demande est instruite par le responsable qui qualifie la demande et transmet son accord ou son désaccord à l'intéressé.
3. En cas d'accord, le responsable recherche dans le catalogue des formations agréées un stage correspondant à la demande.
4. Il informe l'employé du contenu de la formation et lui propose une liste des sessions.
5. Lorsque l'employé retourne son choix, le responsable formation inscrit le participant à la session auprès de l'organisme de formation concerné.
6. Le responsable formation contrôle par la suite la facture qui lui est adressée par l'organisme de formation avant de la transmettre au comptable des achats.

**Etape 1 : établir le diagramme de classes complet**

**Etape 2 : les packages**

* Rassemblez tous les éléments précédents sur un même diagramme de classes
* Proposez une découpe en packages représentant des unités d'organisation métier.
* Dessinez un diagramme de classes par unité d'organisation en essayant de minimiser les dépendances entre packages. Ajouter quelques attributs métiers pertinents pour compléter le modèle statique.