

Cours	Méthodologies d'analyse et de conception logicielles
Auditoire	3 ^{ème} année Licence Réseau et Télécommunication (LA3RT)
Etablissement	Institut National des Sciences Appliquées et de Technologies
Responsable du cours	Aymen SELLAOUTI
Années Universitaires	2013-2014

Objectifs :

Ce cours est destiné aux étudiants de licence en réseau et télécommunication. Il a pour objectif d'introduire les concepts d'analyse et de conception logicielles objets. Il devra aussi initier les étudiants au langage de modélisation UML. Ce cours devra permettre, à l'étudiant, d'avoir les outils nécessaires pour entamer un Projet de fin d'étude et de lui permettre de modéliser un système à travers des diagrammes UML.

Pré requis :

Programmation orientée objet

Plan du Cours :

- 1. Introduction**
- 2. Diagramme de classe**
- 3. Diagramme de cas d'utilisation**
- 4. Diagramme d'interaction**
- 5. Diagramme d'état transition**
- 6. Diagramme d'activité**
- 7. Diagramme de composant**
- 8. Diagramme de déploiement**

Bibliographie :

[1] F. Barbier, UML 2 et MDE : Ingénierie des modèles avec études de cas, Ed. DUNOD, 2005

[2] P. Roques et F. Vallee, UML 2 en action 4^{ème} Edition, Ed. EYROLLES, 2007

[3] J. Gabay et D. Gabay, UML 2 Analyse et conception, Ed. DUNOD, 2008

[4] B. Charroux, A. Osmani et Y. Thierry-Mieg, UML2 : Pratique de la modélisation 2^{ème} Edition, Ed. Pearson, 2009

[5] P. Roques, UML 2 par la pratique : Etudes de cas et exercices corrigés, Ed. EYROLLES, 2009

Méthodologies d'analyse et de conception logicielles

Introduction

UML
Unified Modeling Language
aymen.sellaouti@gmail.com

1 Aymen Sellaouti



Introduction

- Après la programmation par carte perforée, la programmation structurée, ...
- Avec la croissance et la complexité des applications
 - La programmation orientée objet
 - La technologie objet : conséquence ultime de la modularisation dictée par la maîtrise de la conception et de la maintenance d'applications toujours plus complexes
- Nécessité de concevoir de nouvelles méthodes de modélisation

3 Aymen Sellaouti

Qu'est ce qu'un système

- Un système est un **tout** constitué **d'éléments** unis par des **relations**.
- Les éléments et les relations sont munis de **propriétés**.

4 Aymen Sellaouti

Comment décrire un système

- Aspect statique**
 - Déterminer les éléments et les relations
 - Déterminer les propriétés des éléments et des relations
 - Déterminer les valeurs que peuvent prendre les propriétés
- Aspect dynamique**
 - Décrire l'activité du système (réaction par rapport à son environnement)
- Aspect organisationnel**
 - Décrire les acteurs

5 Aymen Sellaouti

Notion de système d'information (SI)

- Le bon fonctionnement d'une organisation est conditionné par la mise en place d'une communication cohérente et fluide :
- L'information, mais elle ne sera utile que si elle est bien exploitée,
- Le SI est une représentation possible de n'importe quel système, notamment tout système humain organisé.
- Le système d'information est le **véhicule** de la **communication** dans l'entreprise. Cette communication possède un langage dont les mots sont les données.

6 Aymen Sellaouti

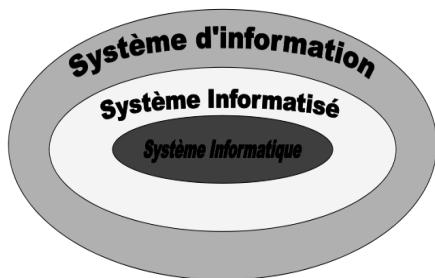
Fonctions d'un SI

- Un système d'information doit assurer les fonctions suivantes :
 - Saisie : Saisie des données faisant partie du SI pour qu'elles aient une existence réelle.
 - Mémorisation : Permet de retrouver la donnée ultérieurement (persistance),
 - Traitement : Permet d'accéder aux données, les mettre à jour et les mettre en forme,
 - Communication : Permet la communication entre le SI et son environnement

7

Aymen Sellaouti

Ne confondant pas



8

Aymen Sellaouti

Méthodes de conception de SI (MCSI)

Définition :

- Une MCSI permet la description des SI :
 - à l'aide d'un **formalisme**,
 - selon une **démarche**,
 - et des moyens de **contrôle qualité**.

9

Aymen Sellaouti

Objectifs des MCSI

- Aider à **réaliser le systèmes informatisé** correspondant au SI.
- Diminuer les **coûts** et les risques des projets d'informatisation.
- Permettre à l'équipe de conception et de développement de disposer d'un **vocabulaire standard**.

10

Aymen Sellaouti

Notion d'objets et Classes

• Concepts d'objets

- Un objet représente une entité du monde réel (ou du monde virtuel pour les objets immatériels) qui se caractérise par un ensemble de propriétés (attributs), des états significatifs et un comportement.
- L'état d'un objet correspond aux valeurs de tous ses attributs à un instant donné. Les propriétés sont définies dans la classe d'appartenance de l'objet.
- Le comportement d'un objet est caractérisé par l'ensemble des opérations qu'il peut exécuter en réaction aux messages provenant des autres objets. Les opérations sont définies dans la classe d'appartenance de l'objet.

11

Aymen Sellaouti

Notion d'objets et Classes

• Concepts d'objets : Exemple

- Considérons l'employé Ahmed, n° 74, embauché en tant qu'ingénieur travaillant sur le site de Nafta.
- Cet objet est caractérisé par la liste de ses attributs et son état est représenté par les valeurs de ses attributs :
 - n° employé : 74,
 - nom : Ahmed,
 - qualification : ingénieur,
 - lieu de travail : site de Nafta.
- Son comportement est caractérisé par les opérations qu'il peut exécuter. Dans notre cas nous pouvons avoir les opérations suivantes :
 - entrer dans l'organisme,
 - changer de qualification,
 - changer de lieu de travail,
 - sortir de l'organisme.

12

Aymen Sellaouti

Notion d'objets et Classes

- **Concept de classe**

- Une classe est l'abstraction d'un ensemble d'objets qui possèdent une structure identique (liste des attributs) et un même comportement (liste des opérations).



Un objet est une instance d'une et une seule classe.

13 Aymen Sellaouti

Notions Fondamentales de l'orienté objet

- **Encapsulation :**

- Par rapport à l'approche classique, l'approche objet se caractérise par le **regroupement dans une même classe** de la description de la structure des **attributs** et de la description des **opérations**. Ce regroupement des deux descriptions porte le nom **d'encapsulation** données-traitements.

14 Aymen Sellaouti

Notions Fondamentales de l'orienté objet

- **Association :**

- Représente une **relation entre plusieurs classes**. Elle correspond à l'**abstraction des liens** qui existent entre les **objets** dans le monde réel.

- **Agrégation:**

- C'est une **forme particulière d'association** entre plusieurs classes. Elle exprime le fait qu'**une classe est composée d'une ou plusieurs autres classes**.

15 Aymen Sellaouti

Notions Fondamentales de l'orienté objet

- **Généralisation:**

- Consiste à **factoriser** dans une classe, appelée super-classe, les attributs et/ou opérations des classes considérées.

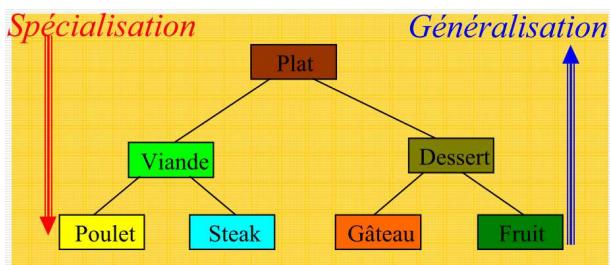
- **Spécialisation:**

- Consiste en la **démarche inverse de la généralisation** puisqu'il s'agit de créer à partir d'une classe, plusieurs **classes spécialisées**.

→ La **généralisation-spécialisation** est un des mécanismes les plus importants de l'approche objet qui **facilite la réutilisation** des classes.

16 Aymen Sellaouti

Notions Fondamentales de l'orienté objet



Spécialisation et généralisation par l'exemple

17 Aymen Sellaouti

Notions Fondamentales de l'orienté objet

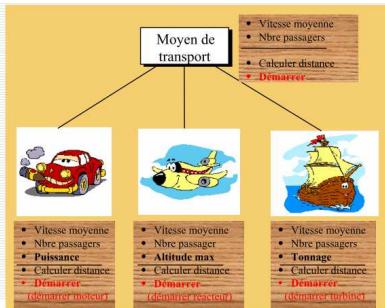
- **Polymorphisme:**

- C'est la capacité donnée à une même opération de s'exécuter différemment suivant le contexte de la classe où elle se trouve.
- Ainsi une opération définie dans une super-classe peut s'exécuter de manière différente selon la sous-classe où elle est héritée.

18 Aymen Sellaouti

Notions Fondamentales de l'orienté objet

C'est la capacité des objets d'une même hiérarchie de classes de répondre différemment à la même opération.



Polymorphisme par l'exemple

19 Aymen Sellaouti

Méthode d'analyse et de conception

En ingénierie :

- Procédé qui a pour objectif de permettre de formaliser les étapes préliminaires du développement d'un système
 - Objectif : rendre le développement plus fidèle aux besoins du client.
- Principe,
- On part d'un énoncé informel
 - Le besoin tel qu'il est exprimé par le client,
 - Complété par
 - des recherches d'informations auprès des experts du domaine fonctionnel, comme les futurs utilisateurs d'un logiciel,
 - de l'analyse de l'existant éventuel (c'est-à-dire la manière dont les processus à traiter par le système se déroulent actuellement chez le client).

20 Aymen Sellaouti

Méthode d'analyse et de conception

- Phase d'analyse : permet de lister les résultats attendus
 - Fonctionnalités,
 - Performance,
 - Robustesse,
 - Maintenance,
 - Sécurité,
 - Extensibilité,
 - Etc.
- Phase de conception :
« permet de décrire de manière non ambiguë, le plus souvent en utilisant un langage de modélisation, le fonctionnement futur du système, afin d'en faciliter la réalisation »

21 Aymen Sellaouti

Méthodes d'analyse et de conception

- RACINES
- Merise
- MMTS
- MASE
- CISAD
- MKSH
- OMT (Object Modeling Technique)
- Booch
- OOSE
- SADT
- SART
- SA/SD
- MACAO
- FAST
- APTE
- Unified Process utilisant la méthode de notation UML
- Analysé décisionnelle des systèmes complexes
- OOD

22 Aymen Sellaouti

Méthodes avec notation formelle

- Méthode B
- Méthode Z
- Méthode VDM
- Lotos
- Lds

23 Aymen Sellaouti

Modélisation

- Modèle (Déf1):** simplification de la réalité, une représentation abstraite permettant de mieux comprendre le système que l'on développe. Un Modèle est synonyme de théorie, mais avec une connotation pratique : un modèle, c'est une théorie orientée vers l'action qu'elle doit servir (Laurent Audibert).
- Modèle (Déf2):** (informatique et construction d'applications) : les modèles d'applications informatiques sont des représentations, à différents niveaux d'abstraction et selon plusieurs vues, de l'information nécessaire à la production et à l'évolution des applications.

24 Aymen Sellaouti

Modélisation

- **Modélisation :**
 - La construction de modèles
 - A quatre objectifs principaux :
 - Les modèles aident à visualiser un système tel qu'il est ou tel que nous voudrions qu'il soit ;
 - Les modèles permettent de préciser la structure ou le comportement d'un système ;
 - Les modèles fournissent un cadre qui guide la construction d'un système ;
 - Les modèles permettent de documenter les décisions prises.
 - La modélisation d'un système s'effectue indépendamment de toute méthode ou de tout langage de programmation.

25

Aymen Sellaouti

Pourquoi modéliser ?

- Pour mieux comprendre les fonctionnalités du système
- Pour pouvoir communiquer (langage commun à tous, une convention)
- Mieux répartir les tâches
- Qualité et efficacité
- Facilite la maintenance

26

Aymen Sellaouti

Principes de la modélisation

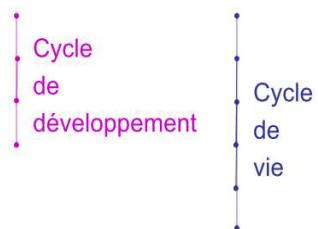
- Quatre principes :
 1. *Premier Principe* : « Le choix des modèles à créer a une très forte influence sur la manière d'aborder un problème et sur la nature de sa solution ».
 2. *Second Principe* : « Tous les modèles peuvent avoir différents niveaux de précision ».
 3. *Troisième Principe* : « Les meilleurs modèles ne perdent pas le sens de la réalité ».
 4. *Quatrième principe* : « Il est préférable de décomposer un système important en un ensemble de petits systèmes presque indépendants »

27

Aymen Sellaouti

Cycles de vie du logiciel

- Analyse
- Conception
- Réalisation
- Tests
- Exploitation
- Maintenance



28

Aymen Sellaouti

Cycles de vie du logiciel

- Analyse de l'existant et définition des besoins, du système d'information et du logiciel
- Conception du système d'information et du logiciel
- Réalisation (ou codage, programmation) : traduction des algorithmes dans un langage compréhensible par un ordinateur

29

Aymen Sellaouti

Cycles de vie du logiciel

- Tests :
 - vérification du logiciel
 - validation du logiciel
 - vérification du système d'information
 - validation du système d'information
- Vérification : le produit en cours d'élaboration répond-il à la définition des besoins ? (est-ce bien le produit ?)
- Validation : le produit en cours d'élaboration remplit-il les fonctionnalités désirées par l'utilisateur ? (est-ce le bon produit ?)

30

Aymen Sellaouti

Cycles de vie du logiciel

- Exploitation : utilisation du logiciel une fois installé
- Maintenance
 - Correction des erreurs
 - Amélioration des fonctions existantes
 - Ajout de nouvelles fonctionnalités

31

Aymen Sellaouti

UML : Unified Modeling Language

- Langage uniifié pour la modélisation
- Développé dans le cadre de la conception orientée objet
 - Langage graphique de modélisation objet
 - Permet de spécifier, de construire, de visualiser et de décrire les détails d'un système logiciel.
- Issu de la fusion de plusieurs méthodes dont « Booch » et « OMT »
- Adapté à la modélisation de tous types de systèmes.

32

Aymen Sellaouti

UML : Langage pour la modélisation

- UML est un langage :
 - Comprend un vocabulaire
 - Comprend un ensemble de règles
 - Centrés sur la représentation conceptuelle et physique d'un système logiciel.
- Ses domaines d'utilisation sont :
 - Visualisation d'un système ;
 - Spécification d'un système ;
 - Construction d'un système ;
 - Documentation d'un système.

33

Aymen Sellaouti

Modèle conceptuel d'UML

- Modèle conceptuel d'UML
- Comprend les notions de base génériques du langage
- Définit trois sortes de « briques » de base :
 - Les éléments : abstractions essentielles à un modèle ;
 - Les relations : liens entre les éléments ;
 - Les diagrammes : regroupent des éléments et des liens au sein de divers ensembles.

34

Aymen Sellaouti

Les éléments

- Il existe quatre types d'éléments dans UML :
 - Les éléments structurels (classe, interface, collaboration,...) ;
 - Les éléments comportementaux (interaction, automate à états finis) ;
 - Les éléments de regroupement (package) ;
 - Les éléments d'annotation (note).

35

Aymen Sellaouti

Les relations

- Il existe quatre types de relations dans UML :
 - La dépendance ;
 - L'association ;
 - La généralisation ;
 - La réalisation.

36

Aymen Sellaouti

Les diagrammes

- Représentation graphique d'un ensemble d'éléments et de relations qui constituent un système.
- UML :
 - 13 types de diagrammes (UML 2)
 - Deux catégories :
 1. **Diagrammes statiques (diagrammes structurels)**
 - Diagrammes de classes,
 - Diagrammes d'objets,
 - Diagrammes de composants,
 - Diagrammes de déploiements
 - Diagramme de paquetage
 - Diagramme de structure composite
 2. **Diagrammes dynamiques (diagrammes comportementaux) :**
 - Diagrammes de cas d'utilisation.
 - Diagrammes d'activités,
 - Diagrammes de séquences,
 - Diagrammes d'états-transitions,
 - Diagramme de communication (diagrammes de collaborations),
 - Diagramme global d'interaction,
 - Diagramme de temps.

37

Aymen Sellaouti

Les diagrammes

2. **Diagrammes dynamiques (diagrammes comportementaux) :**
 - Diagrammes de cas d'utilisation.
 - Diagrammes d'activités,
 - Diagrammes de séquences,
 - Diagrammes d'états-transitions,
 - Diagramme de communication (diagrammes de collaborations),
 - Diagramme global d'interaction,
 - Diagramme de temps.

38

Aymen Sellaouti

Bon travail

aymen.sellaouti@gmail.com

39

Aymen Sellaouti

Méthodologies d'analyse et de conception logicielles

Diagramme de classes

UML
Unified Modeling Language
aymen.sellaouti@gmail.com

1 Aymen Sellaouti



Diagramme de classes

- Structure statique d'un système, en termes de classes et de relations entre ces classes.
- Collection d'éléments de modélisation statiques (classes, paquetages...), qui montre la structure d'un modèle
- Statique* : fait abstraction des aspects dynamiques et temporels.
- Pour un modèle complexe, plusieurs diagrammes de classes complémentaires doivent être construits.

3 Aymen Sellaouti

Diagramme de Classes

- Structure Statique du Système
- Classe :

4 Aymen Sellaouti

Les classes

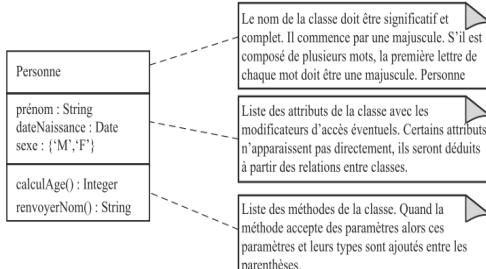
- Classe:**
- Représentation d'un ensemble d'éléments partageant les mêmes attributs, les mêmes opérations, les mêmes relations et les mêmes sémantiques.
- En programmation orientée objet, une classe définit une abstraction, un type abstrait qui permettra d'instancier des objets.
- Graphiquement, une classe décrite en UML peut être plus ou moins précise

5 Aymen Sellaouti

Les classes : Exemple

6 Aymen Sellaouti

Attributs et Opérations



7 Aymen Sellaouti

Attributs et Opérations

- Syntaxe pour la description des attributs

Nom_Attribut : Type_Attribut = Valeur_Initiale

- Syntaxe pour la description des opérations

Nom_Operation(Nom_Argument : Type_Argument = Valeur_Par_Défaut, ...) : Type_Retourné

8 Aymen Sellaouti

Attributs et Opérations

- Attribut de classe

Par défaut, les valeurs des attributs définis dans une classe diffèrent d'un objet à un autre.

Parfois, il est nécessaire de définir un attribut qui garde une valeur partagée par toutes les instances de la classe.

En Java, comme en C++, par exemple, un attribut de classe s'accompagne du mot-clé static.

Graphiquement un attribut de classe est souligné

Exemple **AttributDeClasse : Integer**

9 Aymen Sellaouti

Attributs et Opérations

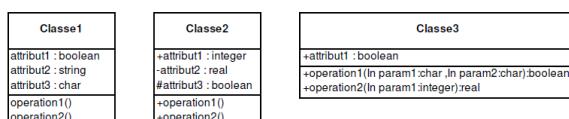
- Les attributs dérivés

Les attributs dérivés, symbolisés par l'ajout d'un slash (/) devant leur nom, peuvent être calculés à partir d'autres attributs et des formules de calcul. Lors de la conception, un attribut dérivé peut être utilisé comme marqueur.

Exemple /AttributDérivé : Integer

10 Aymen Sellaouti

Visibilité d'une classe



Représentation graphique d'une classe

- La première classe (*classe1*) est dite à visibilité réduite, tandis que les deux autres (*classe2* et *classe3*) sont dites à visibilités détaillées.

11 Aymen Sellaouti

Visibilité d'une caractéristique

- Détermine si d'autres éléments peuvent l'utiliser.

- Quatre niveaux de visibilité possibles (attributs ou opérations) :

- Public** « + »
 - Visible par tous
 - Peut être utilisée par n'importe quelle instance ayant une visibilité sur les instances de la classe complète.
- Private** « - »
 - Visible à la classe seule
 - Ne peut être utilisée que par des instances de la classe elle-même.
- Protected** « # »
 - Visible par les sous-classes
 - Ne peut être utilisée que par des instances de la classe elle-même ou bien par les descendants directs de cette classe.
- Paquetage** « ~ »
 - Attribut ou opération ou classe seulement visible à l'intérieur du paquetage où se trouve la classe..

12 Aymen Sellaouti

Exemple d'une classe

```

Personne
-chargeSalaire : float = 0.15
-autreCharge : float = 0.2
- nom : String
- prénom : String
- sexe : {'F', 'M'}
- dateNaissance : Date
- salaire : Integer
- autreRevenu : Integer

« constructor »
Personne (String nom, Date dNaissance)
    « update »
    +setPrenom (String prenom)
    +setAge(Date dNaissance)

    +getNom : String
    +getPrénom : String
    +calculAge() : Date
    +calculCharges () : float

Exceptions
Calcul charge si décès
  
```

La méthode correspondante est :
Renvoie (salaire * chargeSalaire + autreRevenu * autreCharge)

13 Aymen Sellaouti

Les relations entre classes

- L'association
- La dépendance
- La généralisation
- L'implémentation
- L'agrégation
- La composition

14 Aymen Sellaouti

L'association

- Relation d'association:
- Relation structurelle entre classes d'objets
- Précise que les objets d'un élément sont reliés aux objets d'un autre élément
- Une association exprime une connexion sémantique bidirectionnelle entre deux classes

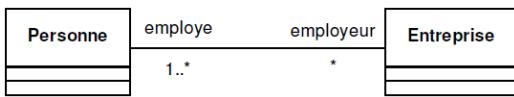


Figure 5. Représentation graphique d'une association

15 Aymen Sellaouti

L'association

- L'association est instanciable dans un diagramme d'objets ou de collaboration, sous forme de liens entre objets issus de classes associées

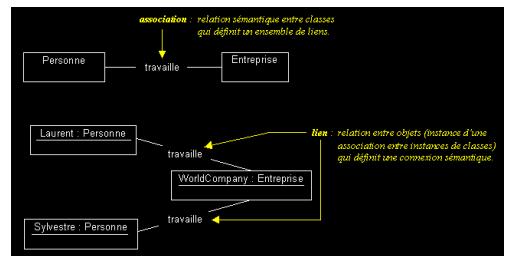


Figure 9. Association et Lien

Représentation d'une association



- Avec possibilité de préciser :
- Un nom d'association
 - Une décoration
 - Le rôle de chaque classe
 - Des cardinalités

17 Aymen Sellaouti

Représentation d'une association



18 Aymen Sellaouti

L'association

- L'extrémité d'une association est appelée rôle



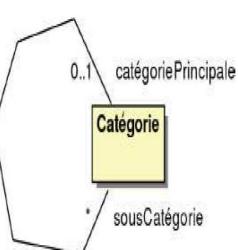
- Décrit comment une classe voit une autre classe au travers de l'association



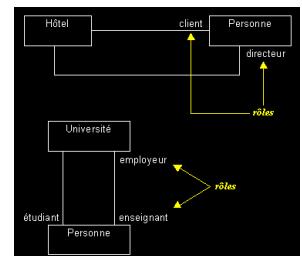
19 Aymen Sellaouti

L'association

Rôles : indispensable pour les associations réflexives



20 Aymen Sellaouti



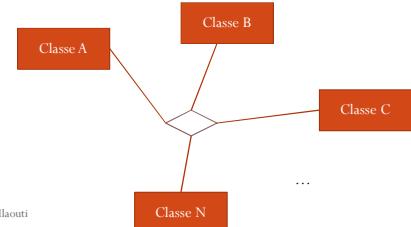
/ref {Cours Mme Bouzidi}

L'association

- Relation Binaire



- Arité des associations



21 Aymen Sellaouti

Cardinalité (Multiplicité)

- *Multiplicité* des associations

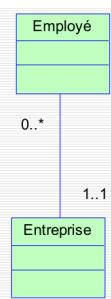
- Information portée par le rôle
- Expression entière bornée
- Précise le nombre d'instances qui participent à une relation
- Les valeurs de multiplicité expriment les contraintes liées au domaine de l'application

22 Aymen Sellaouti

Cardinalité (Multiplicité)

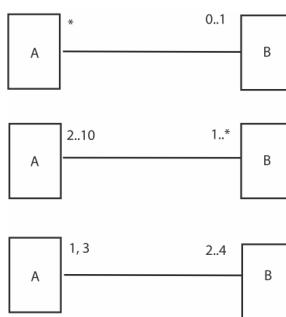
Cardinalités possibles :

- **1** Un et un seul
- **1..1** Un et un seul
- **0..1** Zéro ou un (optionnel)
- **m..n** De m à n
- **0..*** Zéro ou plusieurs
- ***** Zéro à plusieurs
- **1..*** Un à plusieurs



23 Aymen Sellaouti

Cardinalité (Multiplicité)



24 Aymen Sellaouti

- À une instance de A correspond 0 ou 1 instance de B.
- À une instance de B correspond 0 à nombre non déterminé d'instances de A.
- À une instance de A correspond 1 à un nombre non déterminé d'instances de B.
- À une instance de B correspond 2 à 10 instances de A.
- À une instance de A correspond 2 à 4 instances de B.
- À une instance de B correspond 1 ou 3 instances de A.

Cardinalité (Multiplicité)

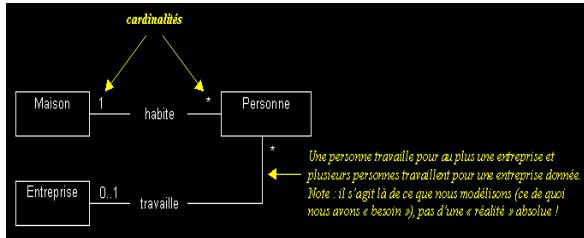


Figure 12. Cardinalités

25 Aymen Sellaouti

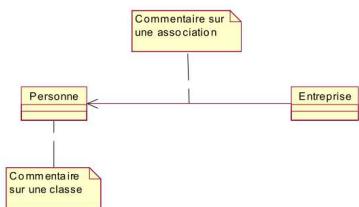
Les notes

- Lors des phases de modélisation et de spécification, il est nécessaire de documenter ses modèles :
 - Contraintes matérielles
 - Contraintes de performance
 - Choix techniques réalisés
 - Références à d'autres docs
 - Explications techniques, etc.

26 Aymen Sellaouti

Les notes

La note représente un commentaire placé sur un des composants du diagramme de classe



27 Aymen Sellaouti

La Navigabilité

- La navigabilité indique si l'association fonctionne de manière unidirectionnelle ou bidirectionnelle, elle est matérialisée par une ou deux extrémités fléchées. La non-navigabilité se représente par un « X »
 - Par défaut, une association est navigable dans les deux sens.
 - Souvent réalisée en phase d'implémentation.



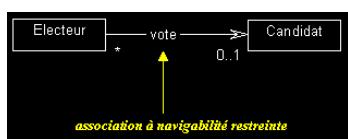
- Signification :
 - Une occurrence de Commande stocke une liste des Articles.
 - Une occurrence d'Article ne stocke pas une liste des commandes.

28 Aymen Sellaouti

La Navigabilité

• Réduction de la portée de l'association

- Peut aussi être exprimée dans un modèle pour indiquer que les instances d'une classe ne "connaissent" pas les instances d'une autre



29 Aymen Sellaouti

La Navigabilité

- Navigabilité unidirectionnelle de A vers B.
- Pas de navigabilité de B vers A (non définie explicitement).
- Navigabilité unidirectionnelle de B vers A.
- Navigabilité de A vers B (non définie explicitement).
- Navigabilité bidirectionnelle entre A et B. Habituellement représentée sans flèche.

30 Aymen Sellaouti

Les contraintes

- Des propriétés particulières proposées dans UML pour préciser la sémantique d'une association.
- Ils sont utilisées lorsque les cardinalités ne sont pas suffisantes pour traduire les règles de gestion, un diagramme de classe peut être complété par des contraintes.
 - Syntaxe : {contrainte}
- Une contrainte peut concerner une ou plusieurs associations.

31

Aymen Sellaouti

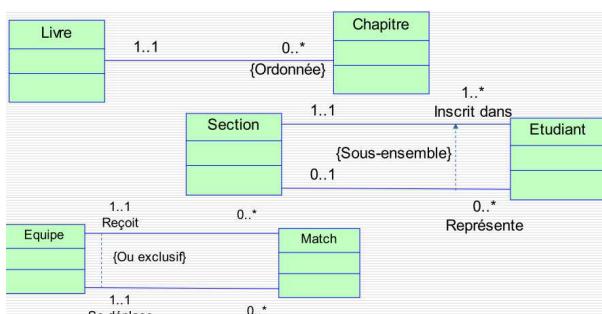
Les contraintes

- Toute règle peut être représentée sous forme de contraintes.
- Quelques contraintes couramment utilisées :
 - {ordonnée} : Signifie l'existence d'une relation d'ordre entre les objets de la classe indiquée par la contrainte.
 - {Sous-ensemble} : Indique qu'une collection est incluse dans une autre.
 - {Ou exclusif} : Signifie la participation exclusive d'un objet à l'une ou l'autre des associations.

32

Aymen Sellaouti

Les contraintes



33

Aymen Sellaouti

Association d'agrégation

- Une agrégation est une association qui représente une relation d'inclusion structurelle ou comportementale d'un élément dans un ensemble. Une relation de subordination,
- L'agrégation est une association qui permet de représenter un lien de type « ensemble » comprenant des « éléments ».
- Il s'agit d'une relation entre une classe représentant le niveau « ensemble » et 1 à n classes de niveau « éléments ». L'agrégation représente un lien structurel entre une classe et une ou plusieurs autres classes.
- La signification de cette forme simple d'agrégation est uniquement conceptuelle. Elle ne contraint pas la navigabilité ou les multiplicités de l'association. Elle n'entraîne pas non plus de contrainte sur la durée de vie des parties par rapport au tout.

34

Aymen Sellaouti

Association d'agrégation



35

Aymen Sellaouti

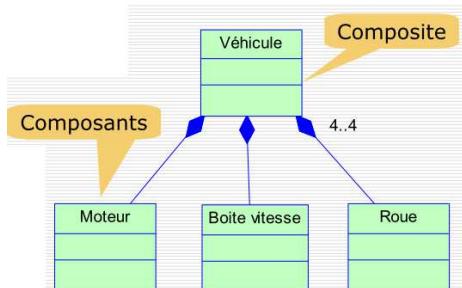
Association de composition

- La composition est une relation d'agrégation dans laquelle il existe une contrainte de durée de vie entre la classe « composant » et la ou les classes « composé ».
- Autrement dit la suppression de la classe « composé » implique la suppression de la ou des classes « composant ».
- Une composition est une association qui décrit une contenance structurelle entre instances.

36

Aymen Sellaouti

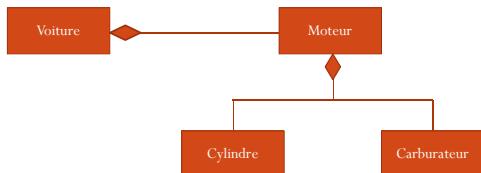
Association de composition



37 Aymen Sellaouti

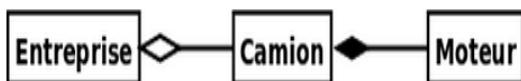
La composition

- Exemple



38 Aymen Sellaouti

Agrégation & composition

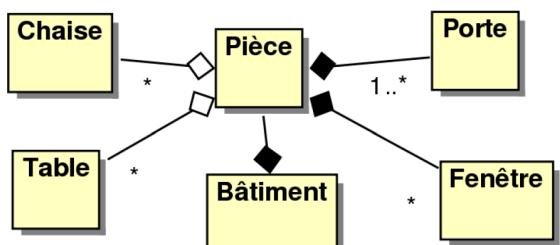


Remarque

Les notions d'agrégation et surtout de composition posent de nombreux problèmes en modélisation et sont souvent le sujet de querelles d'experts et sources de confusions.

39 Aymen Sellaouti

Exemple d'agrégation/composition



/ref{Cours Mme Bouzidi}

40 Aymen Sellaouti

Agrégation & composition

Ex. d'objets graphiques :

- Une forme graphique comporte un polygone, une couleur de remplissage et une couleur de trait.
- Un polygone comporte plusieurs segments.
- Une palette graphique comporte un ensemble de couleurs

Quelles questions faut-il se poser pour différencier agrégation et composition ?

41 Aymen Sellaouti

Agrégation & composition

Ex. d'objets graphiques :

- Une forme graphique comporte un polygone, une couleur de remplissage et une couleur de trait.
- Un polygone comporte plusieurs segments.
- Une palette graphique comporte un ensemble de couleurs

Les éléments sont-ils détruits lorsqu'on détruit l'ensemble ?

Le polygone est-il détruit en même temps que la forme graphique ?
Existe-t-il indépendamment de la forme graphique ?

Un élément peut-il faire parti de plusieurs ensembles ?

Un polygone peut-il faire parti de plusieurs formes graphiques ?

42 Aymen Sellaouti

Agrégation & composition

Ex. d'éléments graphiques :

- Une forme graphique comporte un polygone, une couleur de remplissage et une couleur de trait.
- Si on détruit la forme graphique, le polygone est détruit aussi. Un polygone ne peut faire partie que d'une seule forme graphique.
- Un polygone comporte plusieurs segments.
- Si on détruit le polygone, les segments sont détruits aussi. Un segment ne peut faire partie que d'un seul polygone.
- Une palette graphique comporte un ensemble de couleurs
- Une couleur peut appartenir à plusieurs palettes différentes

43

Aymen Sellaouti

Agrégation & composition

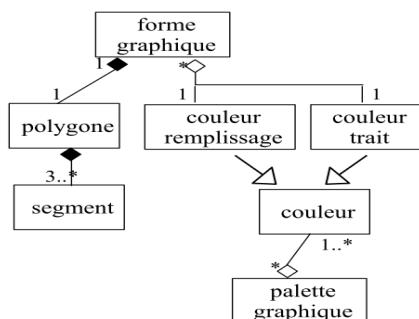
Ex. d'éléments graphiques :

- Une forme graphique comporte un polygone, une couleur de remplissage et une couleur de trait.
- Si on détruit la forme graphique, le polygone est détruit aussi. Un polygone ne peut faire partie que d'une seule forme graphique.
- Un polygone comporte plusieurs segments.
- Si on détruit le polygone, les segments sont détruits aussi. Un segment ne peut faire partie que d'un seul polygone.
- Une palette graphique comporte un ensemble de couleurs
- Une couleur peut appartenir à plusieurs palettes différentes

44

Aymen Sellaouti

Agrégation & composition

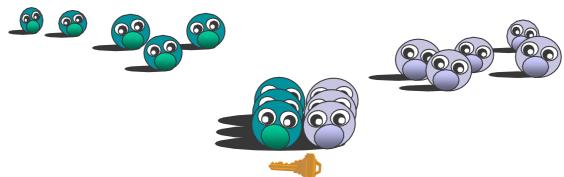


45

Aymen Sellaouti

L'association : Restriction des opérations

- Consiste à sélectionner un sous-ensemble d'objets parmi l'ensemble des objets qui participent à une association



46

Aymen Sellaouti

L'association : Restriction des opérations

- La restriction est
 - Réalisée au moyen d'un tuple particulier appelé **clé**
 - Utilisée conjointement avec un objet de la classe de départ
- Chaque instance de la classe A accompagnée de la valeur de la clé identifie un sous-ensemble des instances de B qui participent à l'association

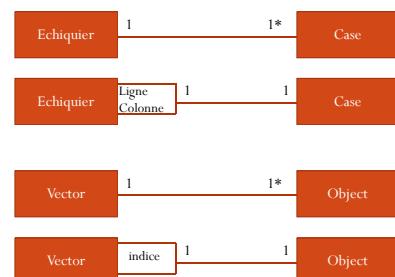


47

Aymen Sellaouti

L'association : Restriction des opérations

Exemples



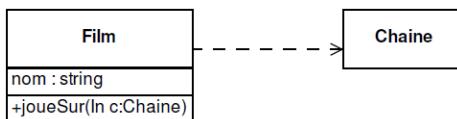
48

Aymen Sellaouti

La dépendance

- **Relation de dépendance:**

- Relation sémantique entre deux éléments selon laquelle un changement apporté à l'un peut affecter la sémantique de l'autre.



49

Aymen Sellaouti

L'association n-aire

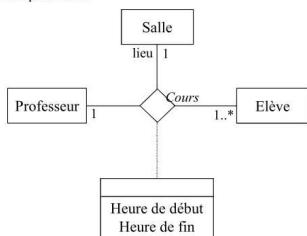
- Une **association de dimension supérieure à 2** se représente en utilisant un losange permettant de relier toutes les classes concernées.
- De telles associations sont difficiles à déchiffrer et peuvent induire en erreur
- Il vaut mieux limiter leur utilisation, en définissant de nouvelles catégories d'associations.
- Une **classe-association** permet de décrire soit des attributs soit des opérations propres à l'association. Cette classe-association est elle-même reliée par un trait en pointillé au losange de connexion.
- Une classe-association peut être reliée à d'autres classes d'un diagramme de classes.

50

Aymen Sellaouti

L'association n-aire

- **Association n-aire** = Une association parmi 3 classes ou plus. Chaque instance de l'association est un n-tuple de valeurs des classes respectives.



51

Aymen Sellaouti

L'association n-aire

Exemple

Une commande est liée à plusieurs produits. À chaque produit de la commande sont associées des dates.

Plus précisément, deux dates sont affectées à chaque produit d'une commande, une date et un produit sont associés à une seule commande et, à une commande et une date donnée sont liés 0 à plusieurs produits.

Donnez une modélisation UML de l'association entre ces trois classes.

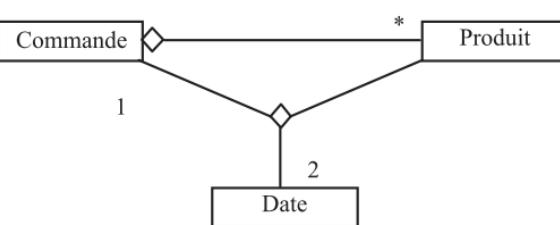
Remarque:

Pour définir la multiplicité à mettre à côté d'une classe dans une relation n-aire, calculez le nombre d'objets de la dite classe consistant avec chaque ensemble d'objets des autres classes. Pour cette application, calculez le nombre d'objets de la classe Date consistant avec chaque couple d'objets des classes Commande et Produit, le nombre d'objets de la classe Commande consistant avec chaque couple des classes Produit et Date et le nombre d'objets de la classe Produit consistant avec chaque couple d'objets des classes Commande et Date.

52

Aymen Sellaouti

L'association n-aire

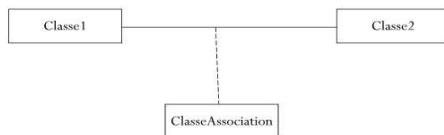


53

Aymen Sellaouti

Classe-association

- Les liens entre les instances de classes peuvent porter des informations. Dans ce cas l'association qui décrit un tel lien reçoit le statut d'une classe (peut être dotée d'attributs et d'opérations)
- Une classe-association est une association porteuse d'attributs.

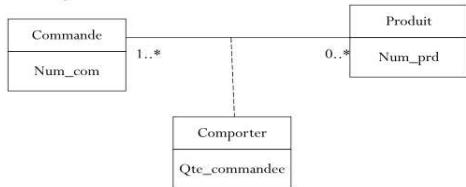


54

Aymen Sellaouti

Classe-association

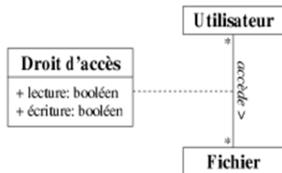
- Une commande comporte plusieurs articles. Pour chaque article elle précise la quantité commandée.
- l'attribut « quantité commandée » dépend fonctionnellement du numéro de commande et du code produit. On va donc placer l'attribut « quantité commandée » dans l'association « comporter ».



55 Aymen Sellaouti

Exercice

- Reproduire la phrase suivante :
- Un fichier est accessible par un utilisateur selon des droits d'accès.



56 Aymen Sellaouti

Généralisation & spécialisation

L'héritage:

- Les hiérarchies de classes permettent de gérer la complexité, en ordonnant les objets au sein d'arborescences de classes, d'abstraction croissante.

Spécialisation

- Démarche descendante, qui consiste à capturer les particularités d'un ensemble d'objets, non discriminés par les classes déjà identifiées
- Consiste à étendre les propriétés d'une classe, sous forme de sous-classes, plus spécifiques

→ permet l'extension du modèle par réutilisation

57 Aymen Sellaouti

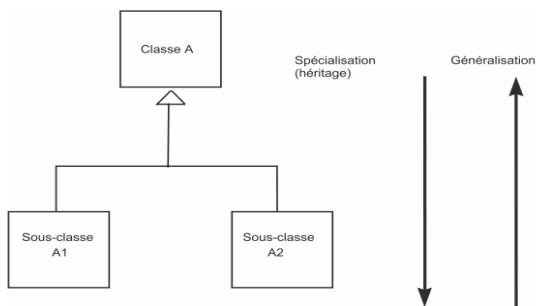
La généralisation

Relation de généralisation:

- Relation entre un élément général et un élément dérivé de celui-ci, mais plus spécifique (*sous-élément* ou *élément fils*)
- Le plus souvent, utilisée pour représenter une relation **d'héritage**.
- Attributs, opérations, relations et contraintes définis dans les super-classes sont hérités intégralement dans les sous-classes
- L'opération qui consiste à créer une super-classe à partir de classes s'appelle la **généralisation**.

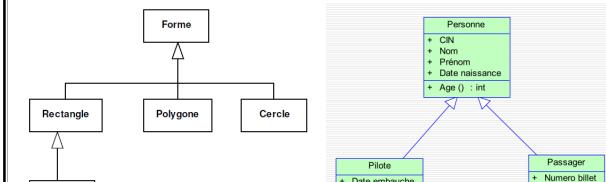
58 Aymen Sellaouti

Généralisation & spécialisation



59 Aymen Sellaouti

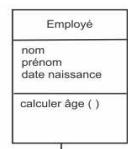
Généralisation & spécialisation



Exemples d'une généralisation

60 Aymen Sellaouti

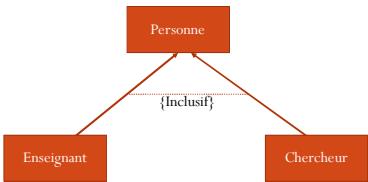
Généralisation & spécialisation



61 Aymen Sellaouti

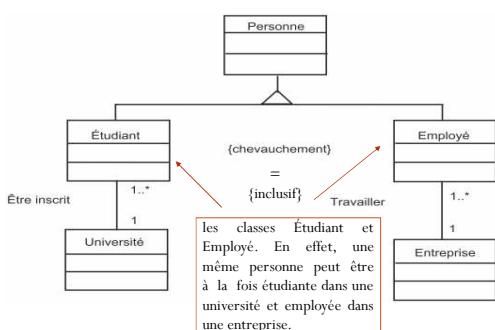
Généralisation & spécialisation

- On peut ajouter des contraintes sur les liens de généralisation
- Par défaut, les liens sont {exclusif} ou {disjoint}
- {Chevauchement} ou {inclusif}
 - Indique qu'une classe descendante d'une classe A appartient au produit cartésien des sous-classes de A



62 Aymen Sellaouti

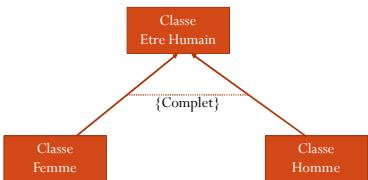
Généralisation & spécialisation



63 Aymen Sellaouti

Généralisation & spécialisation

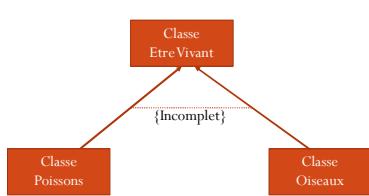
- {Complet}
- Indique que la généralisation est terminée
- Il n'est pas possible de rajouter des sous-classes



64 Aymen Sellaouti

Généralisation & spécialisation

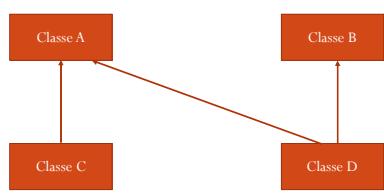
- {Incomplet}
- Désigne une généralisation extensible



65 Aymen Sellaouti

Généralisation & spécialisation

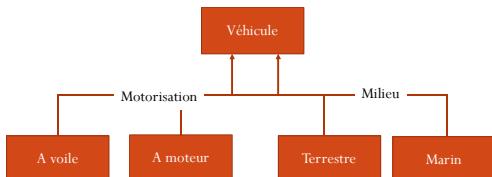
- Généralisation multiple
 - Dans certains cas, il est nécessaire de faire hériter une même classe de deux classes « parents » distinctes. Ce cas correspond à une généralisation (héritage) multiple
 - Les classes peuvent avoir plusieurs super-classes



66 Aymen Sellaouti

Généralisation & spécialisation

- Critère de généralisation
- Indiqué en associant un discriminant à la relation de généralisation

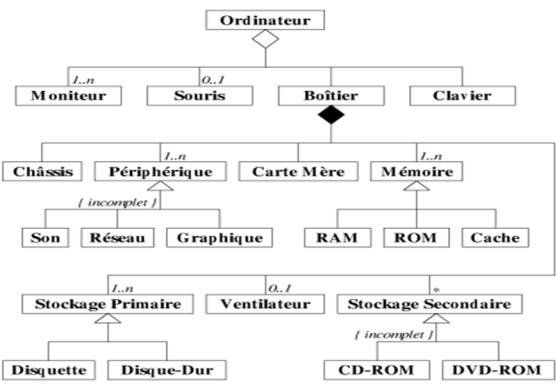


67 Aymen Sellaouti

Exercice

- Un ordinateur est composé d'un ou plusieurs moniteurs, d'un boîtier, d'une souris optionnelle et d'un clavier. Un boîtier a un châssis métallique, une carte mère, plusieurs barrettes de mémoire (RAM, ROM et cache), un ventilateur optionnel, des supports de stockage (disquette, disque-dur, CD-ROM, DVD-ROM...), et des cartes périphériques (son, réseau, graphique...). Un ordinateur possède toujours au moins un lecteur de disquette ou un disque-dur.

68 Aymen Sellaouti

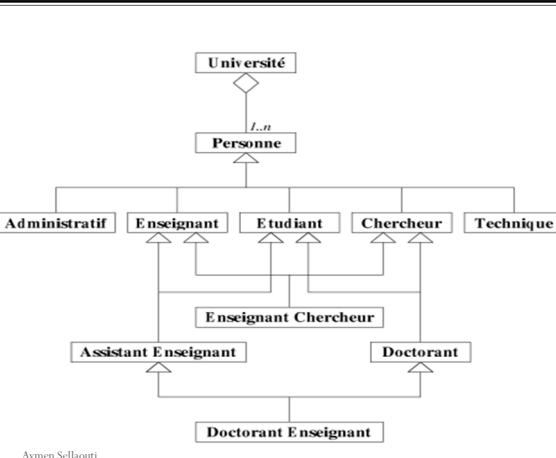


69 Aymen Sellaouti

Exercice

- L'université comporte des personnels administratifs et techniques, des enseignants, des étudiants et des chercheurs (qui sont tous des personnes). Certains étudiants peuvent être des chercheurs (les doctorants) ou des enseignants (les assistants enseignants). Certaines personnes (étudiants ou non) peuvent être à la fois chercheurs et enseignants.

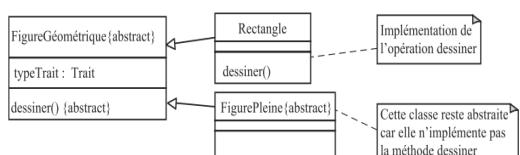
70 Aymen Sellaouti



71 Aymen Sellaouti

Classes abstraite

- Une méthode est dite abstraite lorsqu'on connaît son entête mais pas la manière dont elle peut être réalisée.
- Il appartient aux classes enfant de définir les méthodes abstraites.
- Une classe est dite abstraite lorsqu'elle définit au moins une méthode abstraite ou lorsqu'une classe parent contient une méthode abstraite non encore réalisée.

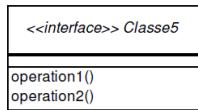


72 Aymen Sellaouti

Les interfaces

- Une interface

- Décrit un contrat d'une classe ou d'un composant sans en imposer l'implémentation
- Ne décrit aucune structure ni aucune implémentation
- Ne peut donc pas contenir d'attributs, ni de méthodes fournies sous la forme d'une implémentation.



Représentation graphique d'une interface

73

Aymen Sellaouti

Les interfaces

- Le rôle d'une interface est de regrouper un ensemble d'opérations assurant un service cohérent offert par un classeur et une classe en particulier.
- Le concept d'interface est essentiellement utilisé pour classer les opérations en catégories sans préciser la manière dont elles sont implémentées.

74

Aymen Sellaouti

Les interfaces

- La réalisation d'une interface par une classe est représentée graphiquement par un trait discontinu qui se termine par une pointe triangulaire. Une manière simplifiée de représenter une interface consiste à placer un petit cercle, souvent appelé «lollipop» (sucette), rattaché à la classe qui réalise l'interface



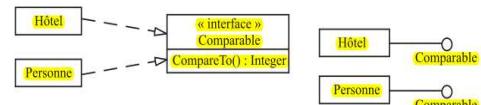
75

Aymen Sellaouti

Les interfaces

- Exemple

Selon des critères à définir a posteriori, les objets de la classe Hôtel et les objets de la classe Personne doivent être comparables. L'opération qui compose l'interface et permet la comparaison entre les instances des classes Personne et Hôtel doit être la même. Cependant, les critères de comparaison sont, évidemment, différents.



76

Aymen Sellaouti

Les packages

- Mécanismes d'ordre général

- Permettent d'organiser les éléments en groupes
- Permettent de définir des sous-systèmes formés d'éléments ayant entre eux une certaine logique.

77

Aymen Sellaouti

Les packages

- Caractéristiques des packages :

- Regroupent des éléments de modélisation selon des critères purement logiques ;
- Permettent d'encapsuler des éléments de modélisation par l'intermédiaire d'interfaces ;
- Permettent de structurer un système en catégories ou sous-systèmes ;
- Servent de « briques » réutilisables dans la conception d'un logiciel.

78

Aymen Sellaouti

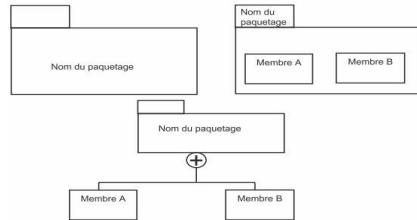
Les packages

- Chaque package
 - doit avoir un nom différent de celui des autres packages
 - peut être composé d'autres éléments,
 - peut être composé d'autres packages.
- Les éléments contenus sont en fait « possédés » (au sens UML du terme),
 - ➡ La destruction d'un package implique la destruction de tous ses éléments.

79 Aymen Sellaouti

Représentation graphique d'un paquetage

- **Représentation globale** – Le nom du paquetage se trouve à l'intérieur du grand rectangle.
- **Représentation détaillée** – Les membres du paquetage sont représentés et le nom du paquetage s'inscrit dans le petit rectangle.
- **Représentation éclatée** – Les membres du paquetage sont reliés par un lien connecté au paquetage par le symbole \oplus .



80 Aymen Sellaouti

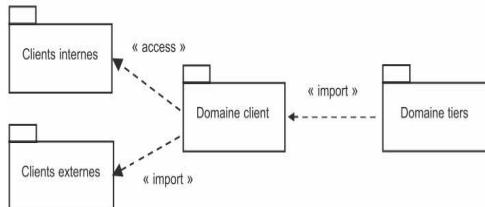
Dépendance entre paquetages

- **L'importation <<import>>**
 - Permet aux éléments d'un package d'importer les éléments d'un autre package.
 - Relation à sens unique
 - Représentée par une relation de dépendance associée à un stéréotype « import ».
- **L'accès <<access>>**
 - Permet aux éléments d'un package d'accéder aux éléments d'un autre package.
 - L'espace de nommage n'est donc pas importé et ne peut être transmis à d'autres paquetages par transitivité.

81 Aymen Sellaouti

Dépendance entre paquetages

- Exemple



82 Aymen Sellaouti

Construction d'un diagramme de classes

- Trouver les classes du domaine étudié ;
 - Souvent, concepts et nom du domaine.
- Préparer le dictionnaire de données : classes retenues.
- Trouver les associations entre classes ;
 - Souvent, verbes mettant en relation plusieurs classes.
 - exemple « l'arbre est composé de feuilles », « un avion est piloté par un pilote »
- Trouver les multiplicités
- Trouver les attributs des classes ;
 - Souvent, noms correspondant à un niveau de granularité plus fin que les classes et composé de noms ou de groupes nominaux.
- Organiser et simplifier le modèle en utilisant l'héritage permettant d'éliminer les classes redondantes;
- Vérifier que le diagramme inclut toutes les demandes du cahier des charges.;
- Itérer et raffiner le modèle. En effet, rare sont les modèles correct dès leur première construction. L'approche est généralement itérative.

83 Aymen Sellaouti

Etude de cas : Gestion de bibliothèque

- Un gérant de bibliothèques désire automatiser la gestion des prêts.
- Il commande un logiciel permettant aux utilisateurs de connaître les livres présents, d'en réserver jusqu'à 2.
- L'adhérent possède un mot de passe qui lui est donné à son inscription.
- L'emprunt est toujours réalisé par les employés qui travaillent à la bibliothèque. Après avoir identifié l'emprunteur, ils savent si le prêt est possible (nombre max de prêts = 5), et s'il a la priorité (il est celui qui a réservé le livre).
- Ce sont les employés qui mettent en bibliothèque les livres rendus et les nouveaux livres. Il leur est possible de connaître l'ensemble des prêts réalisés dans la bibliothèque.

84 Aymen Sellaouti

Etude de cas : Gestion de bibliothèque

Identifier les classes: les classes candidates

- Un gérant de bibliothèque désire automatiser la gestion des prêts.
- Il commande un logiciel permettant aux utilisateurs de connaître les livres présents, d'en résérer jusqu'à 2.
- L'adhérent possède un mot de passe qui lui est donné à son inscription.
- L'emprunt est toujours réalisé par les employés qui travaillent à la bibliothèque. Après avoir identifié l'emprunteur, ils savent si le prêt est possible (nombre max de prêts = 5), et s'il a la priorité (il est celui qui a réservé le livre).
- Ce sont les employés qui mettent en bibliothèque les livres rendus et les nouveaux livres. Il leur est possible de connaître l'ensemble des prêts réalisés dans la bibliothèque.

```

Gérant bibliothèque gestion prêts logiciel utilisateurs
            livres
            adhérent liste mot de passe inscription emprunt
            employés emprunteur ensemble
  
```

Aymen Sellaouti

85

Etude de cas : Gestion de bibliothèque

Les classes retenues

■ Gérant	non pertinente, n'intervient pas
■ <u>bibliothèque</u>	<u>oui</u> responsabilité : gérer les livres, adhérents, prêts
■ gestion	non vague
■ prêts	<u>oui</u> responsabilité : contenir les infos et actions sur les prêts
■ logiciel	non vague
■ utilisateurs	(choix entre utilisateur, adhérent, emprunteur)
■ livres	<u>oui</u> responsabilité : permettre de connaître son état
■ adhérent identifiée	<u>oui</u> responsabilité : permettre à la personne d'être identifiée
■ liste	non implémentation ou conception
■ mot de passe	non attribut
■ Inscription	non action
■ emprunt	non action
■ employés	<u>oui</u> responsabilité : reconnaître qui a fait un prêt, etc..
■ emprunteur	(choix entre utilisateur, adhérent, emprunteur)
■ Ensemble	non implémentation ou conception

Aymen Sellaouti

86

Etude de cas : Gestion de bibliothèque

dictionnaire des données

- bibliothèque : organisme gérant une collection de livres qui peuvent être empruntés par ses adhérents. Une bibliothèque est gérée par ses employés.
- prêt : un prêt est caractérisé par le numéro du livre, la date, la durée. Il ne peut être fait que par un adhérent.
- livre: ouvrage pouvant être emprunté.
- adhérent: personne inscrite à la bibliothèque.
- employé: personne travaillant à la bibliothèque.

Aymen Sellaouti

87

Etude de cas : Gestion de bibliothèque

Chercher les associations

- Un gérant de bibliothèque désire automatiser la gestion des prêts.
- Il commande un logiciel permettant aux utilisateurs de connaître les livres présents, d'en résérer jusqu'à 2.
- L'adhérent possède un mot de passe qui lui est donné à son inscription.
- L'emprunt est toujours réalisé par les employés qui travaillent à la bibliothèque. Après avoir identifié l'emprunteur, ils savent si le prêt est possible (nombre max de prêts = 5), et s'il a la priorité (il est celui qui a réservé le livre).
- Ce sont les employés qui mettent en bibliothèque les livres rendus et les nouveaux livres. Il leur est possible de connaître l'ensemble des prêts réalisés dans la bibliothèque.

Associations sous entendues

```

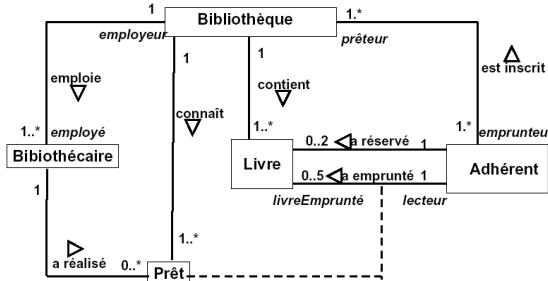
Une adhérent est inscrit à la bibliothèque.
La bibliothèque contient des livres
  
```

Aymen Sellaouti

88

Etude de cas : Gestion de bibliothèque

Les associations

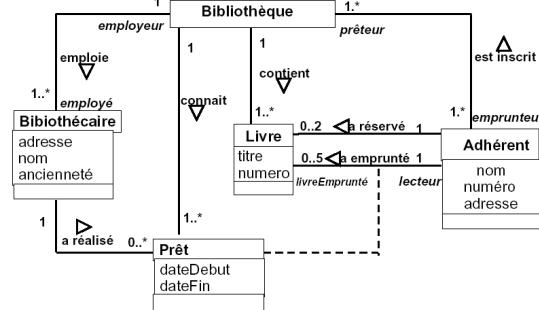


Aymen Sellaouti

89

Etude de cas : Gestion de bibliothèque

Chercher les attributs



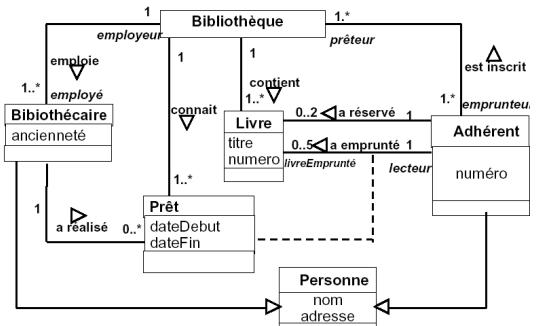
Aymen Sellaouti

90

www.enit-breizh.fr

Etude de cas : Gestion de bibliothèque

Généraliser par héritage



Aymen Sellaouti

Etude de cas (SYSTÈME DE RÉSERVATION DE VOL) [UML2 par la pratique]

Interview des experts

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 d'arrivée et une heure de départ.
 10. Chaque aéroport dessert une ou plusieurs villes.

Aymen Sellaouti

Etude de cas (SYSTÈME DE RÉSERVATION DE VOL) [UML2 par la pratique]

Interview des experts

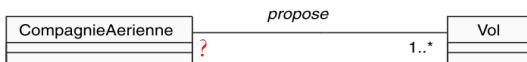
1. Des compagnies aériennes proposent différents vols.
 2. Un vol est ouvert à la réservation et refermé sur ordre de la compagnie.
 3. Un client peut réserver un ou plusieurs vols, pour des passagers différents.
 4. Une réservation concerne un seul vol et un seul passager.
 5. Une réservation peut être 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 d'arrivée et une heure de départ.
 10. Chaque aéroport dessert une ou plusieurs villes.

Aymen Sellaouti

Etude de cas (SYSTÈME DE RÉSERVATION DE VOL) [UML2 par la pratique]

1. Des compagnies aériennes proposent différents vols.

2 Concepts + 1 association



Aymen Sellaouti

Etude de cas (SYSTÈME DE RÉSERVATION DE VOL) [UML2 par la pratique]

1. Des compagnies aériennes proposent différents vols.

2 Concepts + 1 association



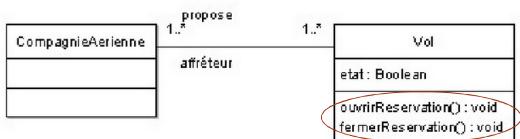
Hypothèse : un vol est proposé par une seule compagnie mais affréter par plusieurs compagnies

Remarque : Un **affréteur** est une société qui loue un navire, un avion, un camion, etc. pour un temps déterminé.

Aymen Sellaouti

Etude de cas (SYSTÈME DE RÉSERVATION DE VOL) [UML2 par la pratique]

2. Un vol est **ouvert** à la réservation et **refermé** sur ordre de la compagnie.



C'est la compagnie qui ouvre et qui referme les vols, alors pourquoi les opérations sont placées dans la classe Vol !!!!?

Aymen Sellaouti

Etude de cas (SYSTÈME DE RÉSERVATION DE VOL) [UML2 par la pratique]



En orienté objet, on considère que l'objet sur lequel on pourra réaliser un traitement doit le déclarer en tant qu'opération. Les autres objets qui posséderont une référence dessus pourront alors lui envoyer un message qui invoque cette opération.

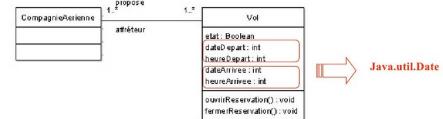
Il faut donc placer les opérations dans la classe Vol, et s'assurer que la classe CompagnieAérienne a bien une association avec elle.

97

Aymen Sellaouti

Etude de cas (SYSTÈME DE RÉSERVATION DE VOL) [UML2 par la pratique]

7. Un vol a un jour et une heure de départ, et un jour et une heure d'arrivée.



Objet ou attribut?

Un objet est un élément plus « important » qu'un attribut.

- si l'on ne peut demander à un élément que sa valeur - **attribut**
- si plusieurs questions s'y appliquent - **objet** (qui possède lui-même plusieurs attributs, ainsi que des liens avec d'autres objets.)

98

Aymen Sellaouti

Etude de cas (SYSTÈME DE RÉSERVATION DE VOL) [UML2 par la pratique]

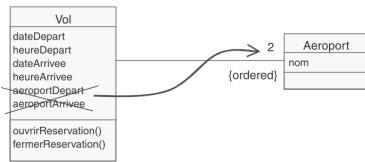
6. Un vol a un aéroport de départ et un aéroport d'arrivée

Pourquoi l'aéroport n'est pas modélisé en tant qu'attribut comme la date ?

La notion d'aéroport est complexe ; elle fait partie du « métier ». Un aéroport ne possède pas seulement un nom, il a aussi une capacité, dessert des villes, etc.

Solution 1 :

Modélisation « tordue » et pas très descriptive pour l'expert métier



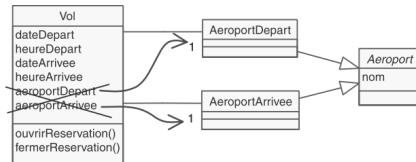
99

Aymen Sellaouti

Etude de cas (SYSTÈME DE RÉSERVATION DE VOL) [UML2 par la pratique]

6. Un vol a un aéroport de départ et un aéroport d'arrivée

Solution 2 :



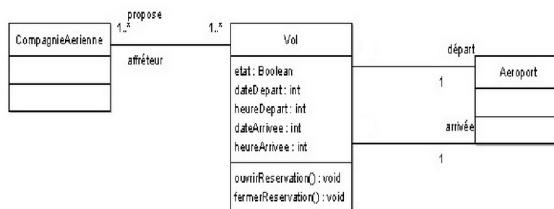
Erronée : Les classes AeroportDepart et AeroportArrivée ont exactement les mêmes instances redondantes

100

Aymen Sellaouti

Etude de cas (SYSTÈME DE RÉSERVATION DE VOL) [UML2 par la pratique]

6. Un vol a un aéroport de départ et un aéroport d'arrivée



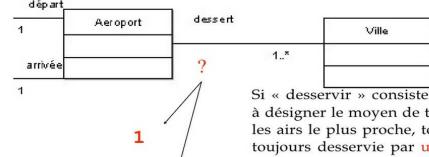
Solution 3

101

Aymen Sellaouti

Etude de cas (SYSTÈME DE RÉSERVATION DE VOL) [UML2 par la pratique]

10. Chaque aéroport dessert une ou plusieurs villes.



Si « desservir » consiste simplement à désigner le moyen de transport par les airs le plus proche, toute ville est toujours desservie par **un** et **un seul** aéroport.

Si « desservir » vaut par exemple pour tout moyen de transport aérien se trouvant à moins de trente kilomètres, alors une ville peut être desservie par **0 ou plusieurs** aéroports.

102

Aymen Sellaouti

Etude de cas (SYSTÈME DE RÉSERVATION DE VOL) [UML2 par la pratique]

- 8. Un **vol** peut comporter **des escales** dans **des aéroports**.
- 9. Une **escale** a une **heure d'arrivée** et une **heure de départ**.

Solution 1 :

Chaque escale a deux propriétés : l'heure de départ et l'heure d'arrivée
L'escale est en relation avec des vols et des aéroports,

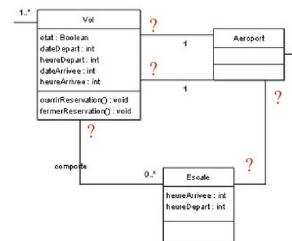


Création d'une classe Escale

103

Aymen Sellaouti

Etude de cas (SYSTÈME DE RÉSERVATION DE VOL) [UML2 par la pratique]



La phrase 8 est imprécise : une escale peut-elle appartenir à plusieurs vols, et quelles sont les multiplicités entre **Escale** et **Aéroport** ?

De plus, le schéma n'indique toujours pas les multiplicités du côté **Vol** avec **Aéroport**

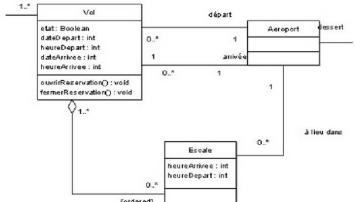
104

Aymen Sellaouti

Etude de cas (SYSTÈME DE RÉSERVATION DE VOL) [UML2 par la pratique]

Pour finaliser le diagramme des phrases 8 et 9, il nous suffit d'ajouter deux précisions :

- l'association entre **Vol** et **Escale** est une **agrégation** (pas une composition, puisqu'elle est partageable) ;
- les escales sont **ordonnées** par rapport au vol.



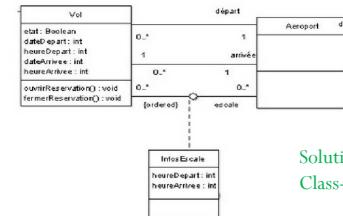
105

Aymen Sellaouti

Etude de cas (SYSTÈME DE RÉSERVATION DE VOL) [UML2 par la pratique]

On peut considérer plutôt cette notion **d'escale** comme un troisième rôle joué par un **aéroport** par rapport à un **vol** ? Les attributs **heureArrivee** et **heureDepart** deviennent alors des **attributs d'association**

. La classe **Escale** disparaît alors en tant que telle, et se trouve remplacée par une **classe d'association** **InfosEscale**.



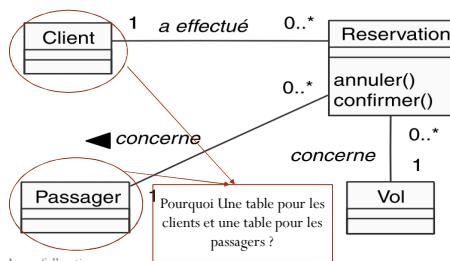
Solution 2 :
Class-association

106

Aymen Sellaouti

Etude de cas (SYSTÈME DE RÉSERVATION DE VOL) [UML2 par la pratique]

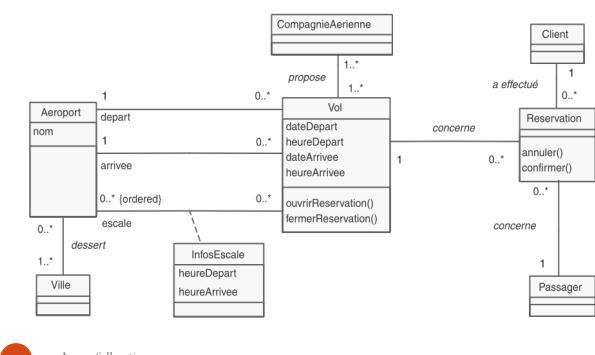
3. Un **client** peut réserver **un ou plusieurs vols**, pour des **passagers différents**.
4. Une **réservation** concerne **un seul vol** et **un seul passager**.
5. Une **réservation** peut être **annulée** ou **confirmée**.



107

Aymen Sellaouti

Etude de cas (SYSTÈME DE RÉSERVATION DE VOL) [UML2 par la pratique]



108

Aymen Sellaouti

Exemple (location de matériel)

L'objectif est de concevoir un système de gestion pour un magasin de location de matériels audio et vidéo.

Le réceptionniste reçoit les demandes de location. Il vérifie la présence du matériel en stock par l'intermédiaire du système. Si l'objet n'est pas en stock, la demande est rejetée. Dans le cas où le matériel est disponible, il demande une pièce d'identité et un chèque de caution au client, le montant de la location et celui de la caution étant fournis par le système.

Le réceptionniste saisit les coordonnées du client, la référence de l'objet ayant été trouvée précédemment par le système lors de sa recherche dans le stock. Le contrat est référencé automatiquement et un exemplaire est imprimé

109

Aymen Sellaouti

Exemple (location de matériel)

Recherche des classes

L'objectif est de concevoir un système de gestion pour un magasin de location de matériels audio et vidéo.

Le réceptionniste reçoit les **demandes de location**. Il vérifie la **présence du matériel en stock** par l'intermédiaire du système. Si l'objet n'est pas en stock, la demande est rejetée. Dans le cas où le matériel est disponible, il demande une **pièce d'identité** et un **chèque de caution** au client, le **montant de la location** et **celui de la caution** étant fournis par le système.

Le réceptionniste saisit les **coordonnées du client**, la **référence de l'objet** ayant été trouvée précédemment par le système lors de sa recherche dans le stock. Le **contrat** est référencé automatiquement et un **exemplaire** est imprimé

110

Aymen Sellaouti

Exemple (location de matériel)

Recherche des associations

L'objectif est de concevoir un système de gestion pour un magasin de location de matériels audio et vidéo.

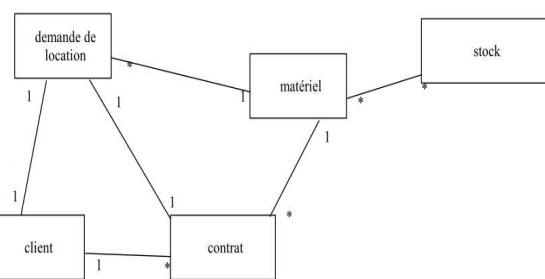
Le réceptionniste **reçoit** les **demandes de location**. Il **vérifie la présence du matériel en stock** par l'intermédiaire du système. Si l'objet n'est pas en stock, la demande est **rejetée**. Dans le cas où le matériel est disponible, il demande une **pièce d'identité** et un **chèque de caution** au client, le **montant de la location** et **celui de la caution** étant fournis par le système.

Le réceptionniste **saisit** les **coordonnées du client**, la **référence de l'objet** ayant été **trouvée** précédemment par le système lors de sa recherche dans le stock. Le **contrat** est **référencé** automatiquement et un **exemplaire** est **imprimé**

111

Aymen Sellaouti

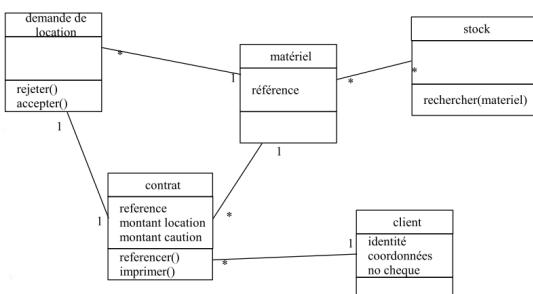
Exemple (location de matériel)



112

Aymen Sellaouti

Exemple (location de matériel)



113

Aymen Sellaouti

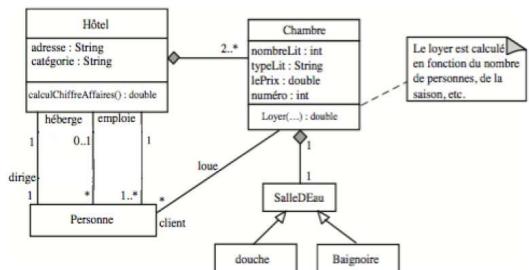
Exercice (Gestion des hôtels)

- 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.

114

Aymen Sellaouti

Exercice (Gestion des hôtels) : Solution



Proposez une amélioration de cette solution

115

Aymen Sellaouti

Diagramme de Objets

116 Aymen Sellaouti

Diagramme d'objets : présentation

- Diagramme de classe appliqué à quelques objets
- Il est aussi appelé **diagramme d'instances**
- Permet de mieux modéliser un diagramme de classe en l'illustrant par des exemples
- Permet de **préciser certains aspects du système** (en mettant en évidence des détails imperceptibles dans le **diagramme de classes**)
- Permet de vérifier et de valider l'adéquation du diagramme de classe à certains cas particulier

117

Aymen Sellaouti

Représentation graphique des objets

- Similaire à une classe mais ne contient que deux compartiments :

- Nom
- Attributs

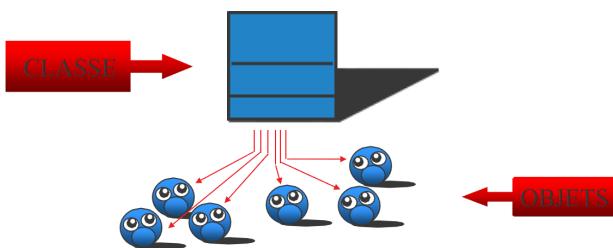
Pourquoi on ne mentionne pas les opérations ?

→ l'interprétation des opérations est la même pour tous les objets

118 Aymen Sellaouti

Représentation graphique des objets

- Les objets sont des instances de classes



119

Aymen Sellaouti

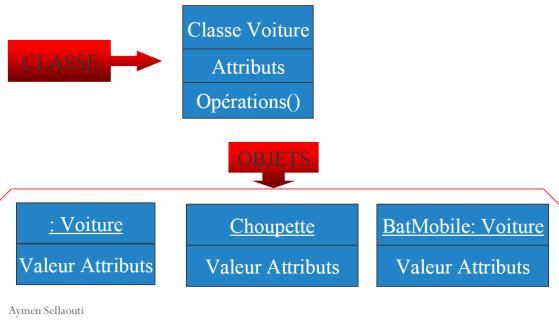
Représentation graphique des objets

- Contrairement au nom d'une classe, le nom d'un objet est **souligné** et son identifiant peut être ajouté devant le nom de la classe.
- Les attributs reçoivent des valeurs. **Quand certaines valeurs d'attribut d'un objet ne sont pas renseignées**, on dit que l'objet est **partiellement défini**

120 Aymen Sellaouti

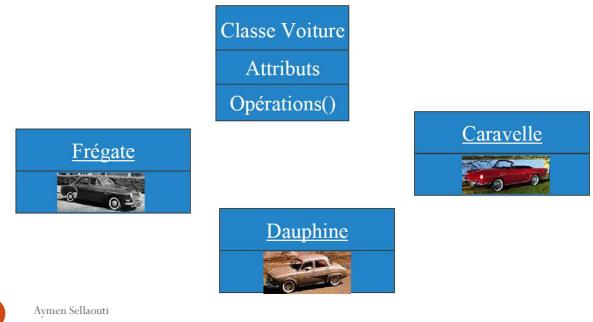
Représentation graphique des objets

- Les objets représentent la structure statique



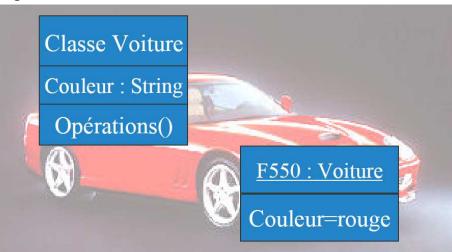
Représentation graphique des objets

- Les objets sont utilisés pour décrire un contexte

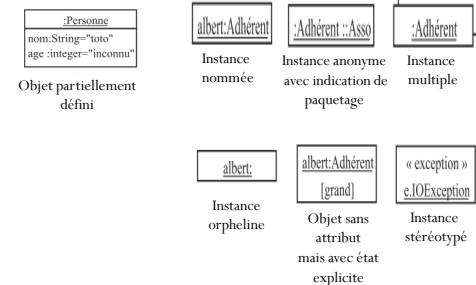


Représentation graphique des objets

- Les représentations des objets peuvent aussi contenir un deuxième comportement qui contient les valeurs d'attributs
- Exemple :



Représentation graphique des objets



Représentation des liens

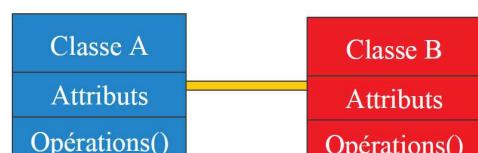
- Les objets sont reliés par des liens



125 Aymen Sellaouti

Représentation des liens

- Ces liens sont des instances de relations entre les classes

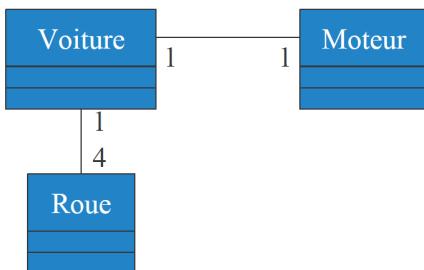


126 Aymen Sellaouti



Représentation des liens

- Exemple : Diagramme de classes

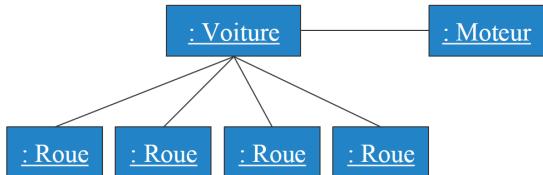


127

Aymen Sellaouti

Représentation des liens

- Exemple : Diagramme d'objets équivalents



- Ce diagramme est une instance du diagramme de classes précédent

128

Aymen Sellaouti

Représentation des liens

- Les liens instances d'associations réflexives peuvent relier deux objets différents
- Exemple :



129

Aymen Sellaouti

Représentation des liens

- Les liens instances d'associations réflexives peuvent relier deux objets différents *ou un objet à lui-même*
- Exemple

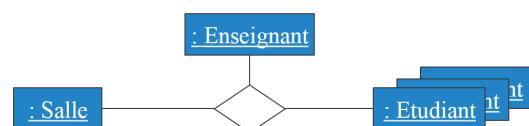


130

Aymen Sellaouti

Représentation des liens

- Les liens d'arité supérieure utilisent le losange
- Exemple :



131

Aymen Sellaouti

Les objets composites

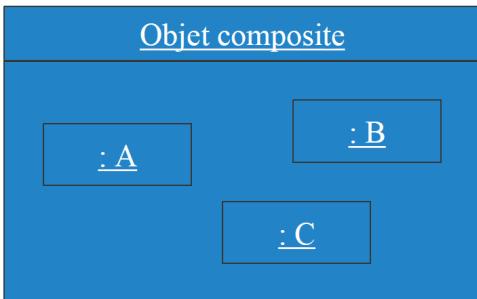
- Représentent les objets composés de sous-objets
- Les attributs sont alors remplacés par des objets
- Les objets composites sont des instances de classes composites
 - Elles sont construites à partir d'autres classes par la plus forte forme d'agrégation



132

Aymen Sellaouti

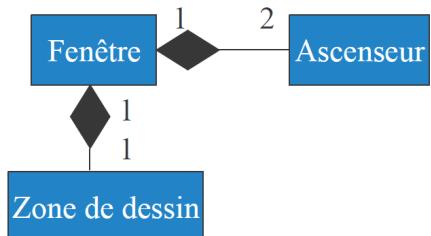
Les objets composites



133 Aymen Sellaouti

Les objets composites

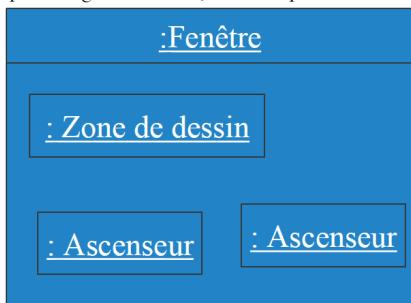
- Exemple : Diagramme de Classes



134 Aymen Sellaouti

Les objets composites

- Exemple : Diagramme d'Objets correspondant



135 Aymen Sellaouti

aymen.sellaouti@gmail.com

136 Aymen Sellaouti

Méthodologies d'analyse et de conception logicielles

Diagramme de cas d'utilisation

UML
Unified Modeling Language
aymen.sellaouti@gmail.com

1 Aymen Sellaouti



Introduction

Modélisation des besoins

- Quels questions faut-il se poser et quoi faire avant de commencer à développer un système?
 - A quoi servira le système ?
 - A quels besoins doit-il répondre ?
 - Lister les besoins du système
Organiser les besoins et faire apparaître les relations qui les relient

Utilisation des diagrammes de cas d'utilisation

3 Aymen Sellaouti

Introduction

Diagramme de Cas d'Utilisation

- Diagramme de cas d'utilisation
- Représentation contextuelle de haut niveau** du système modélisé
- Décrit sous la forme d'actions et de réactions, le comportement d'un système du point de vue d'un utilisateur
 - Montre les **interactions** entre le système et son environnement extérieur

4 Aymen Sellaouti

Introduction

Diagramme de Cas d'Utilisation

- Permet de définir les limites du système et les relations entre le système et l'environnement
 - Permet de définir de manière précise les **frontières** du système à modéliser
- Intervient tout au long du cycle de développement, depuis le cahier des charges jusqu'à la fin de la réalisation
 - Montré les **dépendances** existant entre les cas d'utilisation

5 Aymen Sellaouti

Introduction

Diagramme de Cas d'Utilisation

- C'est une manière spécifique d'utiliser un système
- Permet aux utilisateurs de structurer et d'articuler leurs désirs
- Représente les **cas d'utilisation** du système, les **acteurs** et les **relations** existant entre eux

6 Aymen Sellaouti

Objectifs

- Définir les **besoins** fonctionnels du système
 - ➡ Les cas d'utilisation ont pour principal objectif la capture des **fonctionnalités couvertes par le système**
- Définir le **périmètre** fonctionnel du système
 - ➡ Les cas d'utilisation permettent de définir les frontières du système avec son environnement
- Définir le **dialogue** entre l'utilisateur et le système
 - ➡ Les cas d'utilisation recensent comment **l'utilisateur interagit avec le système**

7

Aymen Sellaouti

Objectifs

- Etablir les **scénarios fonctionnels** qui seront utilisés pour la création du système
 - ➡ Les cas d'utilisation recensent et décrivent les **principales fonctionnalités** attendues du système
- Servir de **support de référence** tout au long des phases de développement du système
 - ➡ Les cas d'utilisation seront consultés et référencés tout au long du processus de développement du système

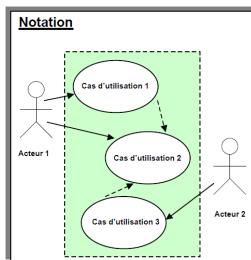
8

Aymen Sellaouti

Modèle des cas d'utilisation

- Le diagramme de cas d'utilisation met en scène :

- Les acteurs
- Les cas d'utilisation
- Les interactions entre acteurs et cas d'utilisation
- Les dépendances entre cas d'utilisation



9

Aymen Sellaouti

Acteur

- L'acteur est à **l'origine** des événements **initiateurs** reçus par le système
- **Dialogue** par la suite avec le cas d'utilisation dont il est l'initiateur
- Possède un **nom** : celui du **rôle** qu'il joue lors de son interaction avec le système
- **N'est pas forcément humain.**
- Peut être :
 - Un autre système
 - Un équipement

10

Aymen Sellaouti

Acteur

- Un acteur
 - Définit un rôle qu'une **entité extérieure** (personne ou chose) assume lors de son interaction avec le système
 - Une même personne physique peut jouer le rôle de plusieurs acteurs
 - Représenté par un **petit personnage** ou par un **rectangle décoré par le stéréotype « acteur »**
 - Le nom de l'acteur apparaît sous le petit personnage ou sous le stéréotype



Un acteur



11

Aymen Sellaouti

Acteur

- « Pierre utilise le système pour gérer son agenda »
- « Philippe utilise aussi le système pour gérer son agenda. Mais Philippe est aussi autorisé à administrer le système »
- **Pierre n'est pas un acteur du système, Philippe n'est pas un acteur du système**

12

Aymen Sellaouti

Acteur

- Le rôle « Utilisateur » est un **acteur** du système
- Le rôle « Administrateur » est un acteur du système

→ **Ne pas confondre personne physique et rôle**
Une personne peut très bien assumer plusieurs rôles et réciproquement

13 Aymen Sellaouti

Acteur

- Les acteurs sont classés en deux catégories selon leur importance :

• **Acteurs principaux** : Ceux qui agissent sur le système.

• **Acteurs secondaires** : Ceux sur lesquels le système agit.

Les acteurs secondaires sont souvent sollicités pour des informations complémentaires ; ils peuvent uniquement consulter ou informer le système lors de l'exécution du cas d'utilisation.

14 Aymen Sellaouti

Acteur

Comment les identifier ?

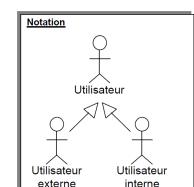
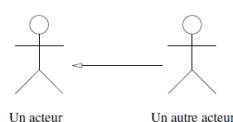
- Les acteurs candidats sont systématiquement :
 - les utilisateurs humains directs : faites donc en sorte d'identifier tous les profils possibles, sans oublier l'administrateur, l'opérateur de maintenance, etc ;
 - les autres systèmes connexes qui interagissent aussi directement avec le système étudié.

15 Aymen Sellaouti

Acteur

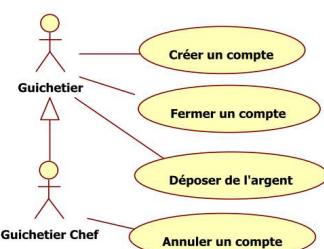
- Un acteur peut également participer à des relations de **généralisation/specialisation**

• On peut définir des **catégories d'acteurs** plus générales ou au contraire **spécialiser** un type d'acteur



16 Aymen Sellaouti

Acteur : Exemple



- Un "Guichetier Chef" est un "Guichetier" spécialisé qui peut faire tout ce que peut faire un "Guichetier" et, en plus, il peut annuler un compte.

17 Aymen Sellaouti

Cas d'utilisation

- Un cas d'utilisation
 - Est une modélisation d'une fonctionnalité ou d'une classe
 - correspond à un certain nombre d'actions que le système devra exécuter en réponse à un besoin d'un acteur.
 - doit produire un résultat observable pour un ou plusieurs acteurs ou parties prenantes du système.

18 Aymen Sellaouti

Cas d'utilisation

- Les cas d'utilisation

- Se déterminent en observant et en précisant, acteur par acteur, les séquences d'interaction du point de vue de l'utilisateur
- Décrivent des interactions potentielles, sans entrer dans les détails de l'implémentation.

19

Aymen Sellaouti

Cas d'utilisation

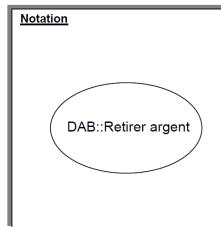
- Pour résumer : un cas d'utilisation
 - **Séquence d'activités** ou d'actions organisées en **étapes distinctes**, et qu'un système effectue en réponse à une **solicitation extérieure**
 - Déclenché par un événement extérieur au système appelé **événement initiateur**
 - Possède un **nom** : celui de la fonctionnalité du système qu'il prend en charge
 - Met en œuvre un **dialogue** entre le système et l'entité à l'origine de l'événement initiateur

20

Aymen Sellaouti

Cas d'utilisation

- Représenté par un ovale
- Son nom apparaît à l'intérieur de l'ovale
- Il est composé :
 - D'un nom optionnel de paquetage
 - Du nom de la fonctionnalité qu'il prend en charge



21

Aymen Sellaouti

Cas d'utilisation

Comment les identifier ?

L'objectif est le suivant : l'ensemble des cas d'utilisation **doit décrire exhaustivement les exigences fonctionnelles du système**. Chaque cas d'utilisation correspond donc à une **fonction métier** du système, selon **le point de vue d'un de ses acteurs**.

Pour chaque acteur, il convient de :

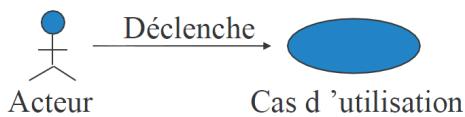
- rechercher les différentes intentions métier avec lesquelles il utilise le système,
- déterminer dans le cahier des charges les services fonctionnels attendus du système.

22

Aymen Sellaouti

Interaction entre acteur et cas d'utilisation

- **Relation de communication**

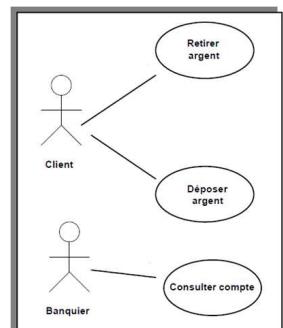


23

Aymen Sellaouti

Interaction entre acteur et cas d'utilisation

- Elle est représentée par une **association** sous la forme d'un lien éventuellement orienté dans le sens de l'interaction
- **Une seule** association est utilisée pour représenter l'ensemble des événements échangés
- L'association peut comporter des **cardinalités**

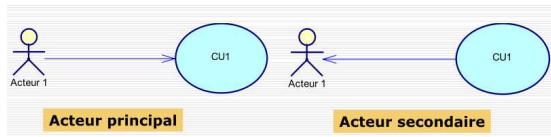


24

Aymen Sellaouti

Interaction entre acteur et cas d'utilisation

- Relation orientée ou non
- Si orientée alors elle indique le type de l'acteur : principal ou secondaire



25 Aymen Sellaouti

Relations entre cas d'utilisation

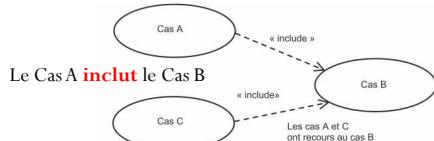
- Il existe trois types de dépendances entre cas d'utilisation :
 - Les **dépendances d'utilisation ou relation d'inclusion**
 - Inclusion d'un cas dans un autre cas
 - Mise en facteur de séquences d'événement communes
 - Les **dépendances d'extension ou relation d'extension**
 - Externalisation de séquences d'événement exceptionnelles
 - Événement optionnels
 - Les **dépendances de généralisation ou relation de généralisation**
 - Généralisation / spécialisation de cas d'utilisation

26 Aymen Sellaouti

Relations entre cas d'utilisation

• Relation d'inclusion

- Une relation d'inclusion d'un cas d'utilisation A par rapport à un cas d'utilisation B signifie qu'une instance de A contient le comportement décrit dans B.
- Les inclusions permettent aussi de décomposer un cas complexe en sous-cas plus simples.



27 Aymen Sellaouti

Relations entre cas d'utilisation

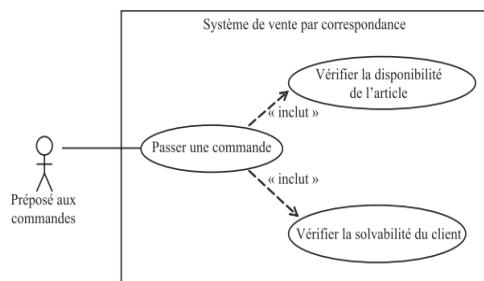
• Relation d'inclusion

- Pour résumer : A <<include>> ou <<includes>> ou <<inclus>> B signifie que :
 - B est une partie de A
 - B est indispensable pour réaliser A
 - A « utilise » B

28 Aymen Sellaouti

Relations entre cas d'utilisation

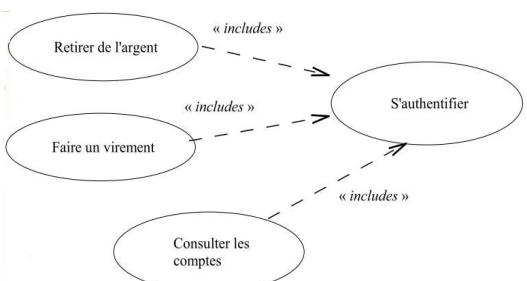
• Relation d'inclusion



29 Aymen Sellaouti

Relations entre cas d'utilisation

• Relation d'inclusion



30 Aymen Sellaouti

Relations entre cas d'utilisation

• Relation d'extension

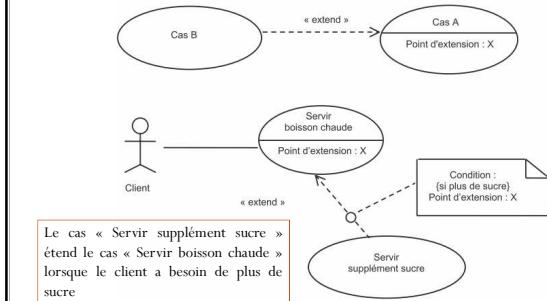
- Une relation d'extension d'un cas d'utilisation A par un cas d'utilisation B signifie qu'une instance de A peut être étendue par le comportement décrit dans B.
- Deux caractéristiques sont à noter :
 - le caractère optionnel de l'extension dans le déroulement du cas d'utilisation standard (A) ;
 - la mention explicite du point d'extension dans le cas d'utilisation standard.

31

Aymen Sellaouti

Relations entre cas d'utilisation

• Relation d'extension

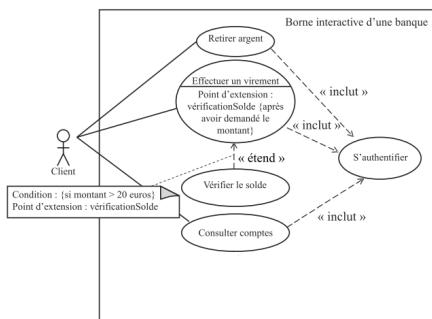


32

Aymen Sellaouti

Relations entre cas d'utilisation

• Relation d'extension

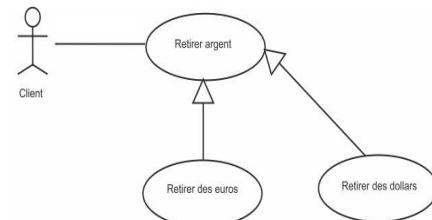


33

Aymen Sellaouti

Relations entre cas d'utilisation

• Relation de généralisation



34

Aymen Sellaouti

Relations entre cas d'utilisation

Cas d'utilisation parent

Cas d'utilisation source

Cas d'utilisation source

Cas d'utilisation enfant

<<inclu>>

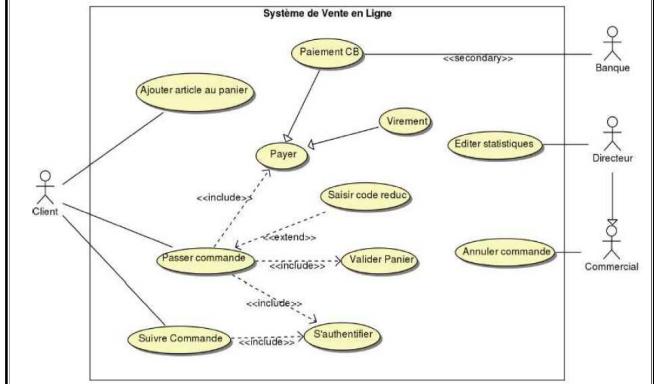
<<étend>>

Les trois relations entre cas d'utilisation

35

Aymen Sellaouti

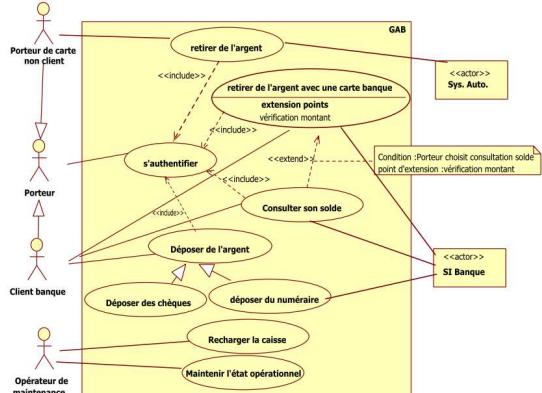
Relations entre cas d'utilisation (Exemple)



36

Aymen Sellaouti

Relations entre cas d'utilisation (Exemple 2)



37

Relations entre cas d'utilisation : Exemple

Un client peut effectuer un retrait bancaire. Le retrait peut être effectué sur place ou par Internet. Le client doit être identifié (en fournissant son code d'accès) pour effectuer un retrait, mais si le montant dépasse 500DH, la vérification du solde de son compte est réalisée.

38

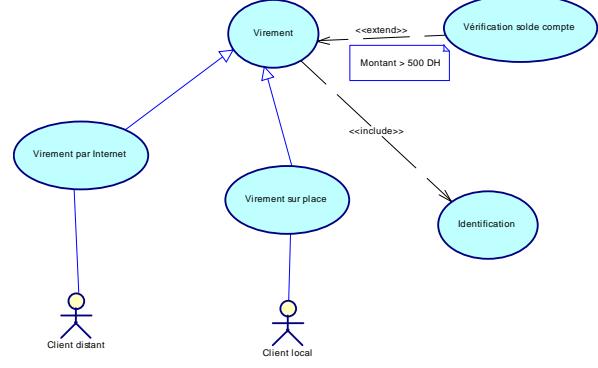
Relations entre cas d'utilisation : Exemple

Un client peut effectuer un **retrait bancaire**. Le retrait peut être **effectué sur place** ou par **Internet**. Le client **doit être identifié** (en fournissant son code d'accès) pour effectuer un retrait, mais si le **montant dépasse 500 DH**, la **vérification du solde de son compte** est réalisée.

39

Aymen Sellaouti

Relations entre cas d'utilisation



40

Aymen Sellaouti

Description des cas d'utilisation

- Le diagramme de cas d'utilisation décrit les grandes fonctions d'un système du point de vue des acteurs.
- Mais il n'expose pas de façon détaillée le dialogue entre les acteurs et les cas d'utilisation.
- nécessité de décrire ce dialogue

41

Aymen Sellaouti

Description des cas d'utilisation

- Deux façons sont couramment utilisées pour décrire les cas d'utilisation :
- Description textuelle
 - Description à l'aide d'un diagramme de séquence
 - À chaque cas d'utilisation doit être associée une description textuelle des interactions entre l'acteur et le système et les actions que le système doit réaliser en vue de produire les résultats attendus par les acteurs.

42

Aymen Sellaouti

Description textuelle d'un UC

- La description textuelle d'un cas d'utilisation est articulée en six points :
 - Objectif – Décrire succinctement le contexte et les résultats attendus du cas d'utilisation.
 - Acteurs concernés – Le ou les acteurs concernés par le cas doivent être identifiés en précisant globalement leur rôle.
 - Pré conditions – Si certaines conditions particulières sont requises avant l'exécution du cas, elles sont à exprimer à ce niveau.

43 Aymen Sellaouti

Description textuelle d'un UC

- Post conditions – Par symétrie, si certaines conditions particulières doivent être réunies après l'exécution du cas, elles sont à exprimer à ce niveau.
- Scénario nominal – Il s'agit là du scénario principal qui doit se dérouler sans incident et qui permet d'aboutir au résultat souhaité.
- Scénarios alternatifs – Les autres scénarios, secondaires ou correspondant à la résolution d'anomalies, sont à décrire à ce niveau. Le lien avec le scénario principal se fait à l'aide d'une numérotation hiérarchisée (1.1a, 1.1b...) rappelant le numéro de l'action concernée.

44 Aymen Sellaouti

Description des cas d'utilisation (Exemple de description textuelle)

- Identification
 - Nom du cas : retrait d'argent
 - Objectif : détaille les étapes permettant à un guichetier d'effectuer des opérations de retrait par un client
 - Acteurs : Guichetier (Principal), Système central (Secondaire)
- Pré-conditions
 - Le distributeur est alimenté en billets
- Post-conditions
 - La transaction a été enregistré par la banque

45 Aymen Sellaouti

Description des cas d'utilisation (description textuelle)

- Scénarios
 - Scénario nominal
 1. Le Guichetier saisit le numéro de compte client
 2. L'application valide le compte auprès du SC
 3. L'application demande le type d'opération au Guichetier
 4. Le Guichetier sélectionne un retrait de 200 DH
 5. Le système interroge le SC pour s'assurer que le compte est suffisamment approvisionné.
 6. Le SC effectue le débit du compte
 7. Le système notifie au guichetier qu'il peut délivrer le montant demandé

46 Aymen Sellaouti

Description des cas d'utilisation (description textuelle)

- Scénarios
 - Scénario alternatif (exception)
 1. En (5) : si le compte n'est pas suffisamment approvisionné

47 Aymen Sellaouti

Étapes de construction du diagramme des cas d'utilisation

Pour modéliser le diagramme des cas d'utilisation, il faut :

- Identifier les acteurs qui entourent le système. Certains acteurs utilisent le système pour accomplir des tâches (acteurs principaux), d'autres effectuent des tâches de maintenance ou d'administration (acteurs secondaires).
- Organiser les acteurs selon une hiérarchisation de généralisation/specialisation
- Intégrer les acteurs au diagramme en spécifiant les cas d'utilisation auxquels ils se rapportent
- Structurer les cas d'utilisation pour faire apparaître les comportements partagés (relation d'inclusion), les cas particuliers (généralisation/specialisation) ou options (relation d'extension)

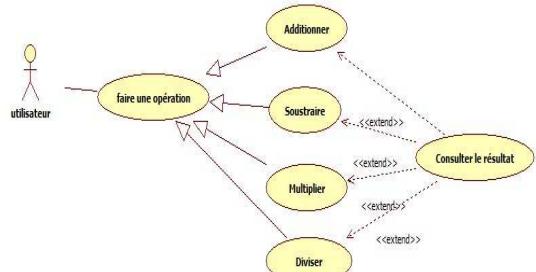
48 Aymen Sellaouti

Exercice

- Proposer le diagramme de Cas d'utilisation d'une calculatrice

49 Aymen Sellaouti

Solution



50 Aymen Sellaouti

Exercice

Une bibliothèque universitaire souhaite automatiser sa gestion. Cette bibliothèque est gérée par un gestionnaire chargé des inscriptions et des relances des lecteurs quand ceux-ci n'ont pas rendu leurs ouvrages au-delà du délai autorisé. Les bibliothécaires sont chargés de gérer les emprunts et la restitution des ouvrages ainsi que l'acquisition de nouveaux ouvrages.

Il existe trois catégories d'abonné. Tout d'abord les étudiants qui doivent seulement s'acquitter d'une somme forfaitaire pour une année afin d'avoir droit à tous les services de la bibliothèque. L'accès à la bibliothèque est libre pour tous les enseignants. Enfin, il est possible d'autoriser des étudiants d'une autre université à s'inscrire exceptionnellement comme abonné moyennant le versement d'une cotisation. Le nombre d'abonné externe est limité chaque année à environ 10 % des inscrits.

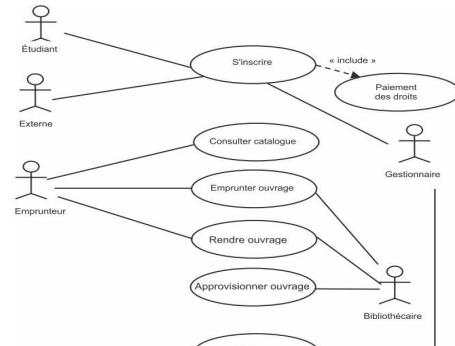
Un nouveau service de consultation du catalogue général des ouvrages doit être mis en place.

Les ouvrages, souvent acquis en plusieurs exemplaires, sont rangés dans des rayons de la bibliothèque. Chaque exemplaire est repéré par une référence gérée dans le catalogue et le code du rayon où il est rangé. Chaque abonné ne peut emprunter plus de trois ouvrages. Le délai d'emprunt d'un ouvrage est de trois semaines, il peut cependant être prolongé exceptionnellement à cinq semaines.

Il est demandé d'élaborer le diagramme des cas d'utilisation.

51 Aymen Sellaouti

Exercice : Corrigé



52 Aymen Sellaouti

aymen.sellaouti@gmail.com

53 Aymen Sellaouti

Méthodologies d'analyse et de conception logicielles

Diagrammes d'interaction

UML
Unified Modeling Language
aymen.sellaouti@gmail.com



Modélisation avec le langage UML

Polytech Paris-Saclay Formation initiale 3^e année Spécialité Informatique

Modéliser les interactions Diagramme de séquence et diagramme de collaboration

Sonia BOUZIDI INSAT

Delphine Longuet delphine.longuet@insat.fr

Génie logiciel et méthodes de conception orientées objet

UML Modélisation dynamique

A. ABDELLATIF abdelaziz.abdellatif@fai.rnu.tn 2007-2008

Aymen Sellaouti

Introduction

Diagrammes d'interaction : modéliser les aspects dynamiques d'un système

Diagramme de cas d'utilisation
Modélise les acteurs interagissant avec les grandes fonctionnalités du système : **vision fonctionnelle et externe d'un système**

PONT entre de ces deux diagramme montrant comment des instances au cœur du système communiquent pour réaliser une certaine fonctionnalité

Diagramme de classes
Décrit le cœur d'un système et montre des classes et la façon dont elles sont associées vision statique et structurelle

Aymen Sellaouti

Diagrammes d'interaction

Objectif : Décrire la **réalisation des cas d'utilisation** sur le système décrit par le diagramme de classes

- Point de vue **interne** sur le fonctionnement du système
- Description au niveau de l'instance (état du système à un instant)
- Description de **séancements** particuliers
- Représentation des **échanges de messages**
 - Entre les acteurs et le système, entre les objets du système
 - De façon chronologique

Aymen Sellaouti

Diagrammes d'interaction

- Un diagramme d'interaction montre **comment** un ensemble d'**objets** et d'**acteurs** **communiquent** ensemble afin:
 - de réaliser les étapes d'un cas d'utilisation
 - de réaliser une certaine fonctionnalité
- L'ensemble des différentes **étapes** à accomplir s'appelle une **interaction**.
- Un diagramme d'interaction montre différents types de communication entre objets, acteurs et sous-systèmes:
 - Appel à des méthodes, information envoyée sur un réseau
 - Ceux-ci sont appelés **messages**.

Aymen Sellaouti

Création d'un diagramme d'interaction

- Un diagramme de classes et des cas d'utilisation sont souvent élaborés en premier lieu.
- Il existe deux sortes de diagramme d'interaction
 - *Diagramme de séquence*
 - *Diagramme de collaboration*

7

Aymen Sellaouti

Diagrammes d'interaction

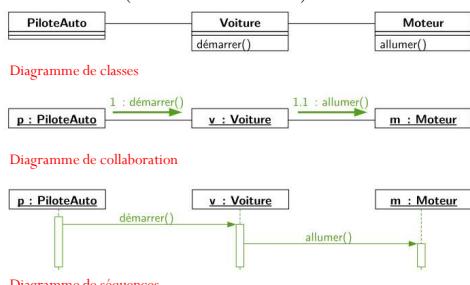
- **Diagramme de séquence**
 - Représentation **temporelle** des interactions entre les objets
 - **Chronologie** des messages échangés entre les objets et avec les acteurs
- **Diagramme de collaboration**
 - Représentation **spatiale** des objets et de leurs **interactions**
 - **Diagramme d'objet** dont les **associations** sont **étiquetées** par les messages envoyés

8

Aymen Sellaouti

Diagrammes d'interaction : Exemple

- Exemple : À partir d'un diagramme de classes et d'un cas d'utilisation (Démarrer la voiture)



9

Aymen Sellaouti

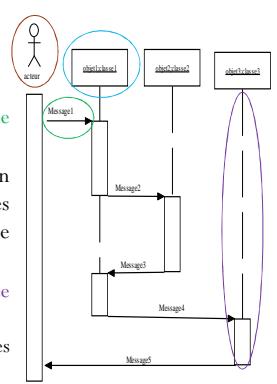
Diagramme de séquence

10

Aymen Sellaouti

Introduction

- Éléments du diagramme de séquence
 - **Acteurs**
 - **Objets (instances)**
 - **Messages**: C'est le véhicule de la communication
- Principes de base : Représentation graphique de la **chronologie** des **échanges de messages** avec le système ou au sein du système
 - « Vie » de chaque entité représentée **verticalement**
 - Échanges de messages représentés **horizontalement**



11

Aymen Sellaouti

Introduction

- Un diagramme de séquence présente une **interaction** en insistant sur le **temps**, en présentant la **séquence** (ordre) des **messages** échangés **entre les objets**
- Un diagramme de séquence est à deux dimensions :
 - **dimension verticale** : le temps;
 - L'ordre d'envoi d'un message est déterminé par sa position sur l'axe vertical du diagramme ; le temps s'écoule de haut en bas de cet axe
 - **dimension horizontale** : les objets (et les acteurs)

12

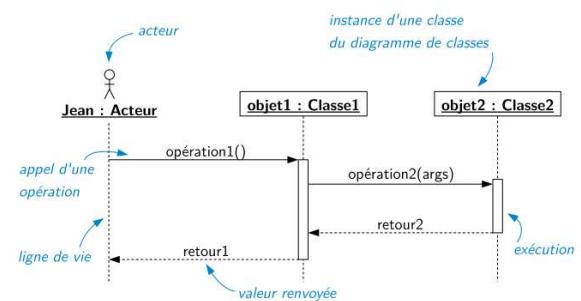
Aymen Sellaouti

Concepts et éléments de base

- **Les participants**
 - Ligne de vie
 - Zones d'activation (exécution)
 - Les acteurs (physique ou objet)
- **Les messages**
- **Les structures de contrôle**
 - Alt: conditionnelle
 - Loop: boucle
 - Réf: référence à un autre diagramme de séquence

13 Aymen Sellaouti

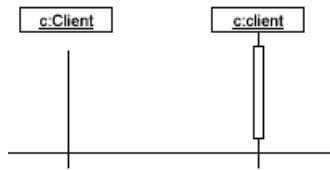
Concepts et éléments de base



14 Aymen Sellaouti

Participants

- **Ligne de vie**
 - Une ligne de vie représente l'existence d'un objet pendant une période de temps
- **Activation**
 - Une activation représente le temps durant lequel un objet est actif, c'est à dire en train d'exécuter une opération



15 Aymen Sellaouti

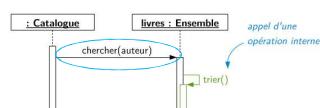
Les Message

- **Définition**
Les objets (instances d'acteurs ou de classes) **communiquent** entre eux via des **messages**
- **Types de messages**
 - Message d'appel
 - Message de retour
 - Message d'envoi
 - Message de création
 - Message de destruction

16 Aymen Sellaouti

Message d'appel et message de retour

- Un **message d'appel** invoque une opération sur un objet
- Un objet peut envoyer un message à lui-même (**message réflexif**) pour invoquer une opération locale



- **Message de retour**
 - Un **message de retour** renvoie une valeur à l'appelant

17 Aymen Sellaouti

Message d'envoi (asynchrone)

- Un message d'envoi envoie un **signal** à un objet
- Un message d'envoi permet d'invoquer une opération d'une manière **asynchrone**

→ On n'attends pas la valeur de retour

18 Aymen Sellaouti

Message synchrone

- La **synchronisation** est le mode par défaut des messages d'appel, de retour, de création et de destruction
 - Un message **synchrone** signifie que l'objet **émetteur** se **bloque** en attendant la **réponse** du récepteur du message
 - Le contrôle est passé de l'appelant à l'appelé;
 - l'opération de l'appelant est **suspendue**
 - Le contrôle est rendu à l'appelant à la fin de l'opération
- Une fois appelée, l'appelant reprend l'exécution de l'opération en cours

19 Aymen Sellaouti

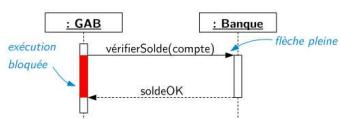
Message asynchrone

- La **concurrence** ou **l'asynchronisme** est spécifiée par l'utilisation de message d'envoi (message asynchrone)
- Un signal est envoyé à un objet; l'objet **envoyeur continue** son **opération**
 - L'objet émetteur n'attend pas la réponse du récepteur et poursuit sa tâche sans se soucier de la réception de son message

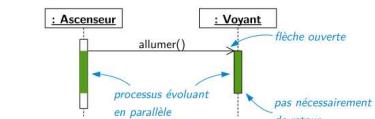
20 Aymen Sellaouti

Message synchrone ou asynchrone

Message synchrone : Émetteur **bloqué** en attente du retour



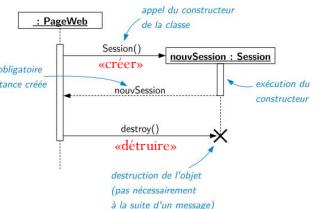
Message asynchrone : Émetteur **non bloqué**, continue son exécution



21 Aymen Sellaouti

Message de création et de destruction

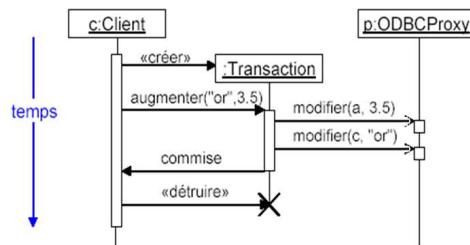
- Message de **création**
 - Un message «**créer**» invoque l'opération de **création d'un objet** Normalement interprété comme **appel de constructeur**
- Un message de **destruction**
 - Un message «**détruire**» invoque l'opération de destruction d'un objet
 - Un objet peut se suicider en s'envoyant un message détruire



22 Aymen Sellaouti

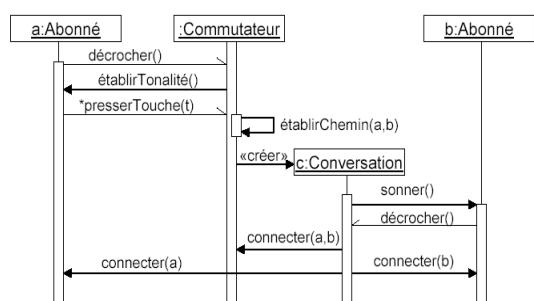
Séquencement des messages

- Le séquencement des messages est spécifié par la position verticale des messages échangés
- Représentation graphique



23 Aymen Sellaouti

Exemple (coup de fil entre deux abonnés)



24 Aymen Sellaouti

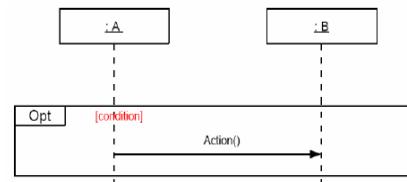
Structure de contrôle

- Les différentes structures de contrôle sont:
 - Opt
 - Loop
 - Alt
 - Ref
 -

25 Aymen Sellaouti

Structure de contrôle (optionnel)

- Principe :** Condition à l'envoi d'un message
- Notation :** Opt
 - Fragment parcouru si la condition est vérifiée



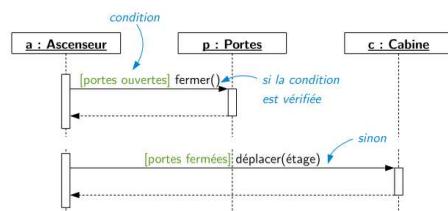
26 Aymen Sellaouti

Structure de contrôle (Alternative)

Principe : Condition à l'envoi d'un message

Notation :

- Deux diagrammes



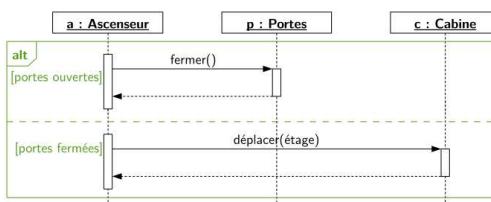
27 Aymen Sellaouti

Structure de contrôle (Alternative)

Principe : Condition à l'envoi d'un message

Notation :

- Deux diagrammes
- Bloc d'alternative alt



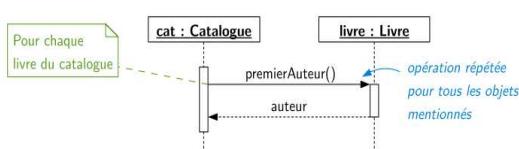
28 Aymen Sellaouti

Structure de contrôle (Boucle)

Principe : Répéter un enchaînement de messages

Notation :

- Note



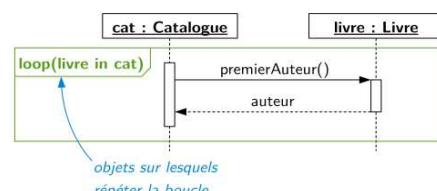
29 Aymen Sellaouti

Structure de contrôle (Boucle)

Principe : Répéter un enchaînement de messages

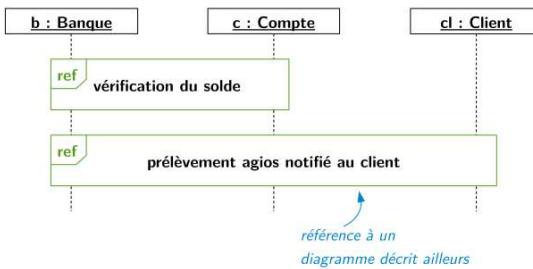
Notation :

- Note
- Bloc de boucle loop



30 Aymen Sellaouti

Référence à un autre diagramme



31 Aymen Sellaouti

Stéréotypes De Jacobson

- Boundary : Couche présentation (vue)



- Control : Couche Application (contrôleur)

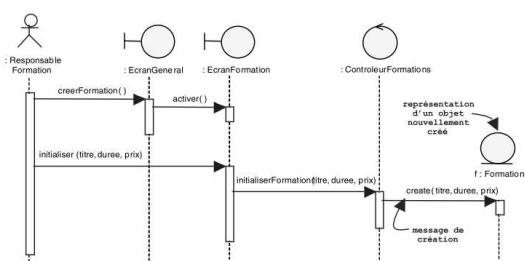


- Entity : Couche métier (Modèle)



32 Aymen Sellaouti

Diagramme de séquence de l'initialisation d'une formation



33 Aymen Sellaouti

Quelques règles

Messages entre acteurs et interface

- Modéliser avec de « Fausses » opérations liées au cas d'utilisation (même nom)

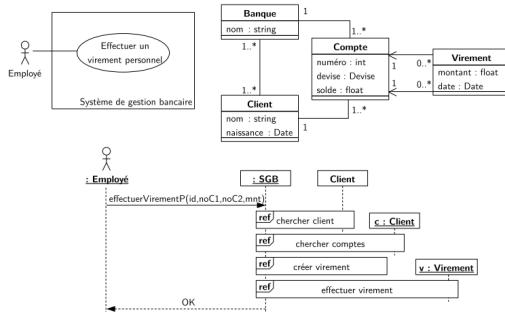
Messages au sein du système

- Opérations du diagramme de classes

- Si message de objA : ClasseA vers objB : ClasseB, alors
 - ClasseA et ClasseB liées par une association
 - Opération du message dans ClasseB

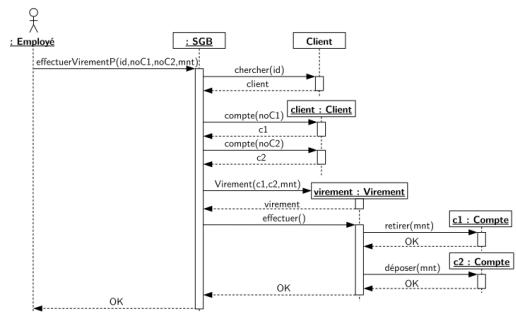
34 Aymen Sellaouti

Exemple



35 Aymen Sellaouti

Exemple



36 Aymen Sellaouti

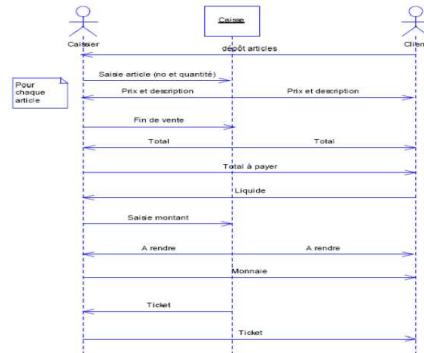
Exercice 1

- Le déroulement normal d'utilisation d'une caisse de supermarché est le suivant
- un client arrive à la caisse avec ses articles à payer
 - le caissier enregistre le numéro d'identification de chaque article, ainsi que la quantité si elle est supérieure à 1
 - la caisse affiche le prix de chaque article et son libellé
 - lorsque tous les achats sont enregistrés, le caissier signale la fin de la vente
 - la caisse affiche le total des achats
 - le caissier annonce au client le montant total à payer
 - le client choisit son mode de paiement
 - liquide : le caissier encaisse l'argent, la caisse indique le montant à rendre au client
 - chèque : le caissier note le numéro de pièce d'identité du client
 - carte de crédit : la demande d'autorisation est envoyée avant la saisie
 - la caisse enregistre la vente et l'imprime
 - le caissier donne le ticket de caisse au client

Modéliser cette situation à l'aide d'un diagramme de séquence en ne prenant en compte que le cas du paiement en liquide.

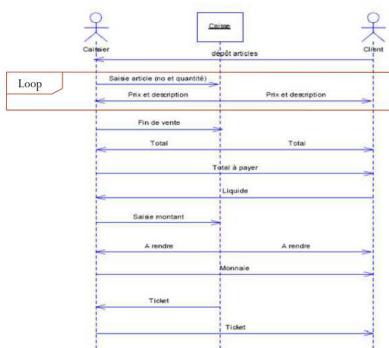
37 Aymen Sellaouti

Correction (v1)



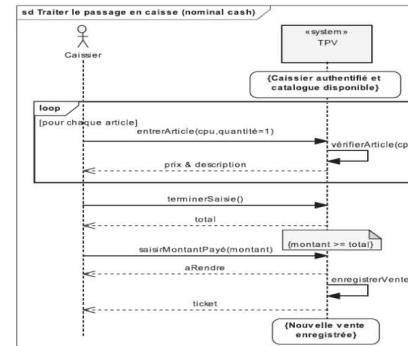
38 Aymen Sellaouti

Correction (v2)



39 Aymen Sellaouti

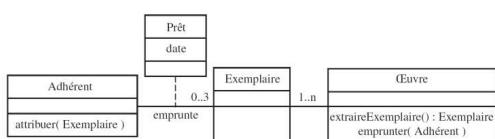
Correction (v3t) : le client n'est plus considéré dans le diagramme



40 Aymen Sellaouti

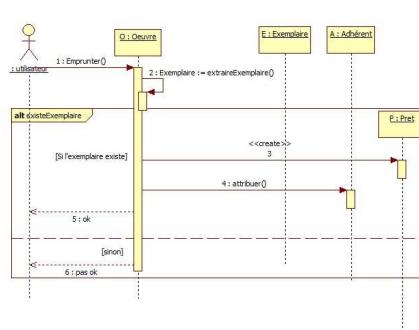
Exercice 2

- Le diagramme de classes présenté ici modélise la structure interne de la bibliothèque. Un acteur adhérent peut emprunter un exemplaire d'une œuvre donnée. L'emprunt se fait de la façon suivante : la méthode emprunter est appellée avec un objet de classe Adhérent donné en argument ; s'il reste des exemplaires dans la bibliothèque, l'un des exemplaires associés à l'œuvre est extrait via la méthode extraireExemplaire, une instance de la classe Prêt est créée, puis l'exemplaire extrait de la bibliothèque est attribué à l'adhérent grâce à l'opération attribuer. S'il restait un exemplaire, l'œuvre retourne « OK » et dans le cas contraire, elle retourne « PasOK » .



41 Aymen Sellaouti

Correction



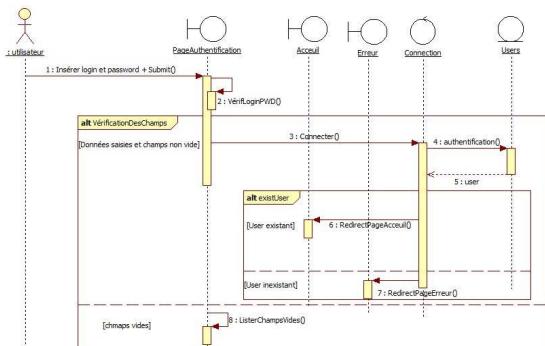
42 Aymen Sellaouti

Exercice 3 (Modèle MVC)

- Une application Web bâtie sur le modèle MVC contient une page web associé qui permet l'authentification des utilisateurs avant l'accès aux différentes fonctionnalités de l'application.
- Pour se connecter, l'utilisateur doit introduire son login et son mot de passe. Avant de déclencher le traitement de la requête d'authentification, les deux champs login et mot de passe doivent être validés au niveau de la page d'authentification (les champs sont vides ou non) afin d'éviter l'envoi de requête inutile. Une fois la requête envoyée au contrôleur, ce dernier consulte le modèle afin de vérifier l'existence de l'utilisateur. Si l'utilisateur existe alors il sera redirigé vers la page d'accueil, sinon il sera redirigé vers une page d'erreur.

43 Aymen Sellaouti

Correction



44 Aymen Sellaouti

Diagramme de collaboration

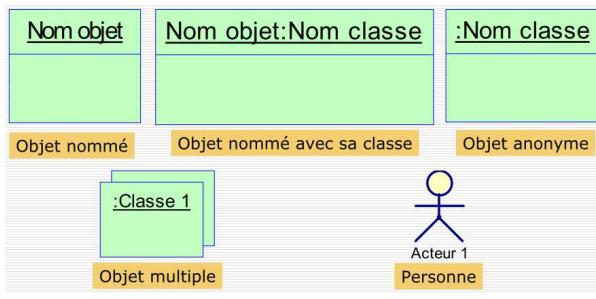
45 Aymen Sellaouti

Introduction

- Un diagramme de collaboration illustre comment les objets coopèrent dans la réalisation d'une interaction
- Un diagramme de collaboration est un graphe dont les objets sont les sommets.
- Les arcs sont des liens de communication entre ces objets
- Les messages sont associés à ces liens.
 - Représentés par des flèches
- L'ordonnancement temporel est indiqué à l'aide d'une numérotation

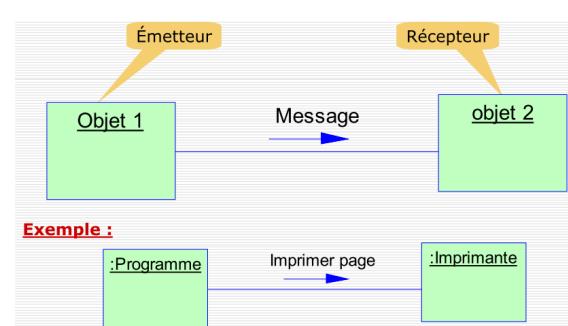
46 Aymen Sellaouti

Représentation d'un objet



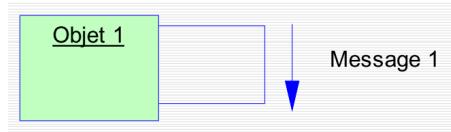
47 Aymen Sellaouti

Représentation des messages



48 Aymen Sellaouti

Messages réflexifs



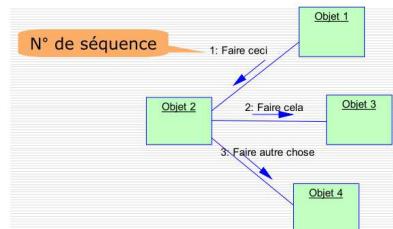
- Remarque :

Il s'agit généralement d'objets composites : un objet composé envoie un message à un de ses objets composants.

49 Aymen Sellaouti

Ordre des messages

Lorsque le diagramme de collaboration comporte **plusieurs messages**, il devient utile de représenter l'**ordre d'envoi** de ces messages.

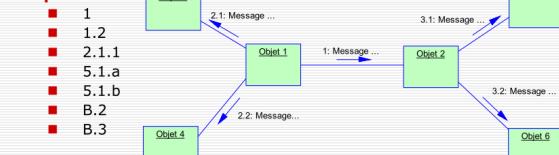


50 Aymen Sellaouti

Hiérarchisation dans l'ordre des messages

- Dans un diagramme de collaboration, tous les messages ne correspondent pas à une seule activité. Il devient utile d'introduire :
 - Un partitionnement des messages (groupes de messages)
 - Une hiérarchie de messages

Exemples :



51 Aymen Sellaouti

Synchronisation des messages

- La synchronisation dans un message est le fait que l'envoi de ce dernier est conditionné par l'envoi préalable d'autres messages.

Exemple :



52 Aymen Sellaouti

Messages conditionnels

- L'envoi d'un message peut être soumis à une condition.



53 Aymen Sellaouti

Messages itératifs

- L'envoi d'un message peut être répétitif.

Exemples :

- *[i := 1..n]
- *[a <= b]

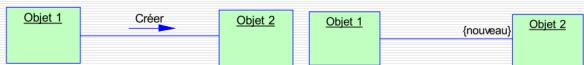


54 Aymen Sellaouti

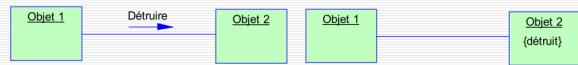
Création et destruction d'objets

- Un message peut créer ou détruire un autre objet.

■ Crédit d'objet :



■ Destruction d'objet :



55 Aymen Sellaouti

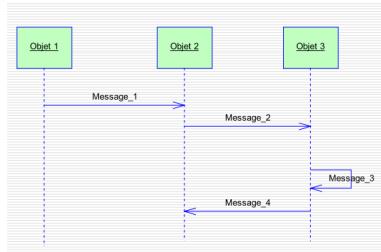
Similitude diagramme séquence/collaboration

- Les diagrammes de collaboration, comme les diagrammes de séquence, permettent de donner une représentation détaillée des diagrammes de cas d'utilisation.
- Inutile de faire les deux diagrammes: un seul suffit.
- Possibilité de déduire l'un des diagrammes à partir de l'autre.

56 Aymen Sellaouti

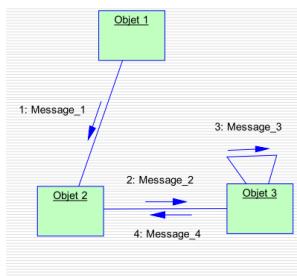
Exercice

Déduire le diagramme de collaboration correspondant à ce diagramme de séquence



57 Aymen Sellaouti

Correction



58 Aymen Sellaouti

Comment choisir entre un diagramme de séquence et un diagramme de collaboration

- Les diagrammes de séquence
- Rendent explicite la séquence temporelle dans l'interaction
 - Les diagrammes de séquence constituent donc le choix naturel lorsque l'interaction est construite à partir d'un cas d'utilisation.
- Facilitent une écriture détaillée des messages
 - Moins d'espace est généralement disponible dans un diagramme de collaboration
 - Permet de décrire de longs scénarios

59 Aymen Sellaouti

Comment choisir entre un diagramme de séquence et un diagramme de collaboration

- Les diagrammes de collaboration
- Puissent être vus comme la prolongation d'un diagramme de classes
 - Préférables lorsque l'interaction est déduite du diagramme de classes
 - Sont aussi très utiles pour valider des diagrammes de classes
 - Reflètent mieux l'architecture du système et ses composantes

60 Aymen Sellaouti

aymen.sellaouti@gmail.com

Méthodologies d'analyse et de conception logicielles

Diagrammes d'état transition

UML
Unified Modeling Language
aymen.sellaouti@gmail.com



Diagrammes dynamiques

Diagramme d'états transitions
Diagramme d'activités

Polytech Paris-Sud Formation initiale 3^e année Spécialité Informatique

Diagrammes états-transitions

Delphine Longuet delphine.longuet@institut-france.fr

Génie logiciel et méthodes de conception orientées objet

UML Modélisation dynamique

A. ABDELLATIF abdelaziz.abdellatif@fai.rnu.tn
2007-2008

Aymen Sellaouti

Introduction

- Objectif : Décrire le **comportement dynamique** d'une entité (logiciel, composant, objet...)
- Comportement décrit par états + transitions entre les états
 - État : abstraction d'un **moment de la vie d'une entité** pendant lequel elle satisfait un ensemble de conditions
 - Transition : **changement** d'état
- Intérêt :
 - Vue synthétique** de la dynamique de l'entité
 - Regroupe un **ensemble de scénarios**

Aymen Sellaouti

Diagramme d'état-transition

- Les **diagrammes d'états-transitions** d'UML décrivent le comportement interne d'un objet à l'aide d'un automate à états finis
 - À tout instant, un système ou un objet se trouve dans un certain état.
 - être dans un état donné signifie que le système se comportera d'une façon spécifique en réponse aux événements se produisant.
 - Certains événements vont provoquer des changements d'états
 - Dans ce nouvel état, le système se comportera de façon différente.
 - Un diagramme d'état est un graphe dans lequel les états sont des nœuds et dont les arcs représentent les transitions.

Aymen Sellaouti

Concepts de base

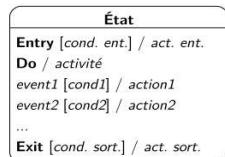
Le diagramme d'états-transitions se base sur les concepts suivants :

- État** : C'est la **situation**, à un instant donné, d'un **objet**. Il est déterminé par les valeurs des propriétés de l'objet à cet instant.
- Transition** : Représente le **passage** d'un objet d'un état à un autre.
- Événement** : C'est la combinaison d'un ensemble d'**informations** pour déclencher une transition.
- Action** : Correspond à une **opération** (ou méthode) de la classe qui sera **exécuté** lorsqu'un événement a lieu.
- Activité** : C'est une **action** particulière dont l'exécution dure dans le temps.

Aymen Sellaouti

Les états

- **État initial** ●
Initialisation du système, exécution du constructeur de l'objet
- **État final** ●
Fin de vie du système, destruction de l'objet
- **États intermédiaires** : étapes de la vie du système, de l'objet



7

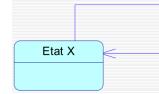
Aymen Sellaouti

Les transitions

- Une transition représente le passage d'un objet d'un état à un autre.
- Une transition est une **connexion unidirectionnelle** entre un **état source** et un **état de destination**.



- La source et la destination peuvent correspondre à un même état (boucle).

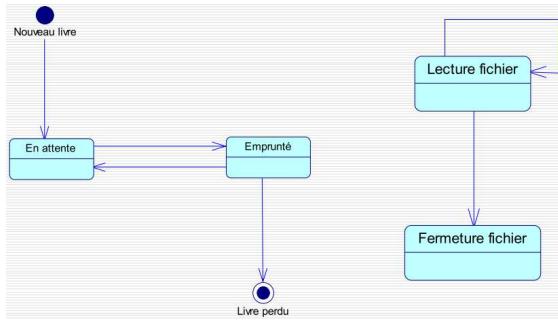


- Une transition est toujours **instantanée**.

8

Aymen Sellaouti

Exemple d'état-transition



9

Aymen Sellaouti

Événement

- **Événement** : Fait **instantané** venant de **l'extérieur** du système et survenant à un **instant donné**
- Types d'événements :
 - **Signal** : réception d'un **message asynchrone**
 - **Appel d'une opération** (synchrone) : liée aux cas d'utilisation, opération du diagramme de classes...
- Satisfaction d'une **condition** booléenne : **when(cond)**, évaluée continuellement jusqu'à ce qu'elle soit vraie
- **Temps**
 - Le paramètre s'évalue comme une durée, par défaut écoulée depuis l'entrée dans l'état courant. Exemple : **after(durée)**

10

Aymen Sellaouti

Action

- **Action** : Réaction du système à un **événement**
- Caractéristiques : atomique, instantanée, non interruptible
- Exemples d'actions :
 - envoi d'un signal
 - appel d'une opération
 - création ou destruction d'un objet

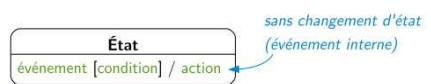
11

Aymen Sellaouti

Action déclenchée par un événement

Représentation : événement [condition] / action

- Lorsque l'**événement** se produit, si la **condition** est vérifiée, alors l'**action** est effectuée

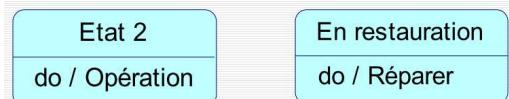


12

Aymen Sellaouti

Les activités

- Une activité est une opération qui s'exécute lorsqu'un objet se trouve dans un état donné.
- Une activité est illustrée avec le mot clef do
- Une activité se distingue par rapport à une action par le fait que :
 - Une action est instantanée
 - Une activité dure dans le temps.

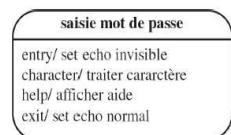


13

Aymen Sellaouti

Transition interne

- Un objet reste dans un état durant une certaine durée et des transitions internes peuvent intervenir.
- Une **transition interne** ne modifie pas l'état courant, mais suit globalement les règles d'une transition simple entre deux états.
- Trois déclencheurs particuliers sont introduits permettant le tir de transitions internes : entry/, do/, et exit/.



14

Aymen Sellaouti

Déclencheurs de transitions internes prédéfinis

Entry: L'action est déclenchée juste à l'**entrée** de l'état

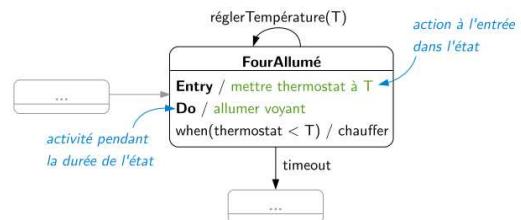
Exit: L'action est déclenchée juste à la **sortie** de l'état.

Do: L'activité se **déroule** dans l'état.

15

Aymen Sellaouti

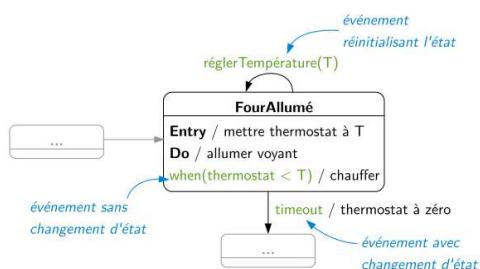
Déclencheurs de transitions :exemples



16

Aymen Sellaouti

Déclencheurs de transitions :exemples



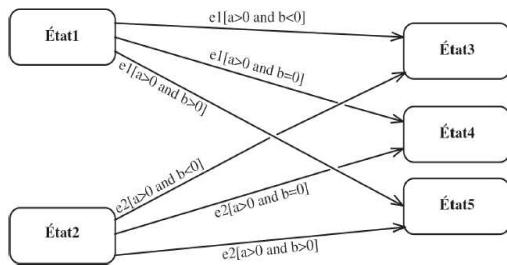
17

Aymen Sellaouti

Point de décision

- On peut représenter des **alternatives** pour le franchissement d'une transition.
- On utilise pour cela des **pseudo-états particuliers** :
 - Les **points de jonction** (petit cercle plein) permettent de partager des segments de transition.
 - Ils ne sont qu'un raccourci d'écriture.
 - Ils permettent des représentations plus compactes.
- Les **points de choix** (losange) sont plus que des raccourcis d'écriture.

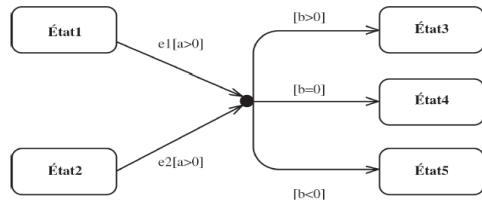
Simplification par point de jonction



- Pour emprunter un chemin, toutes les gardes le long de ce chemin doivent s'évaluer à vrai dès le franchissement du premier segment.

19 Aymen Sellaouti

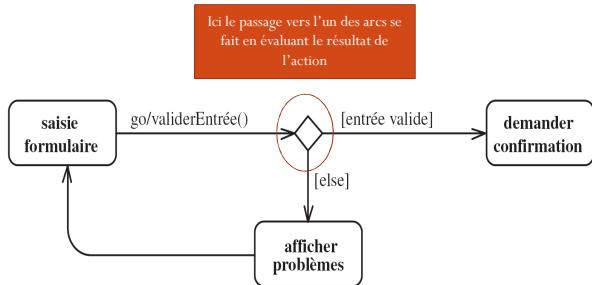
Simplification par point de jonction



Les points de jonction permettent de partager des segments de transition, l'objectif étant d'aboutir à une notation plus compacte ou plus lisible des chemins alternatifs

20 Aymen Sellaouti

Utilisation du point de décision



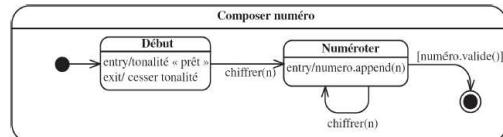
Contrairement aux points de jonction, les points de choix ne sont pas de simples raccourcis d'écriture.

21 Aymen Sellaouti

Etat composite

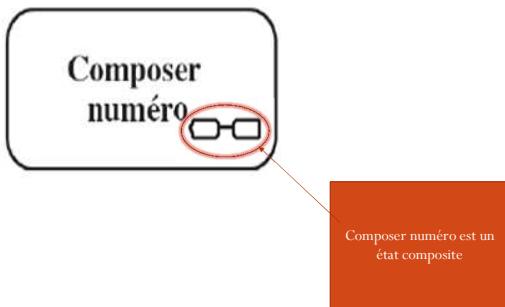
- Un état composite, par opposition à un état dit « simple », est décomposé en deux ou plusieurs sous-états.

- Tout état ou sous-état peut ainsi être décomposé en sous-états imbriqués sans limite a priori de profondeur.
- Un état composite est représenté par les deux compartiments de nom et d'actions internes habituelles, et par un compartiment contenant le sous-diagramme.



22 Aymen Sellaouti

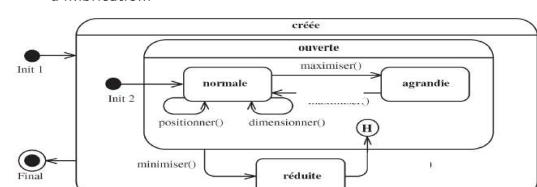
Etat Composite



23 Aymen Sellaouti

Etats composites et états initiaux/finaux

- Les transitions peuvent avoir pour cible la frontière d'un état composite. Elle sont alors équivalentes à une transition ayant pour cible l'état initial de l'état composite.
- Une transition ayant pour source la frontière d'un état composite est équivalente à une transition qui s'applique à tout sous-état de l'état composite source.
- Cette relation est transitive et peut « traverser » plusieurs niveaux d'imbrication.



24 Aymen Sellaouti

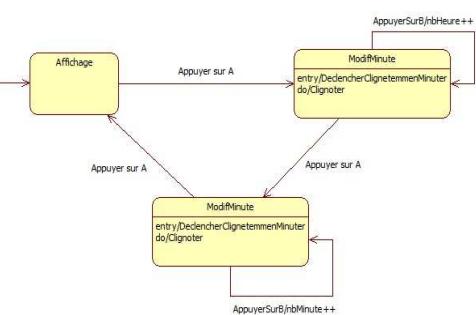
Exercice

- Une montre digitale simple possède un cadran et deux boutons, que l'on nommera A et B, pour la mettre à l'heure. La montre a deux modes d'opérations, affichage de l'heure et mise à l'heure. En mode affichage, les heures et minutes sont affichées, séparées par un signe "deux points" intermittent. Le mode de mise à l'heure a deux sous-modes, heures et minutes. Le bouton A s'utilise pour changer les modes. A chaque fois que l'on appuie dessus le mode change suivant la séquence : affichage, configurer heures, configurer minutes, affichage, etc. Dans un sous-mode, le bouton B s'emploie pour avancer les heures ou les minutes à chaque fois que l'on appuie dessus. Les boutons doivent être relâchés avant de produire un autre événement. Dans les états spécifiques aux changement d'heures et de minutes, les chiffres à changer clignotent jusqu'au changement d'état.
- Donner un **diagramme d'états** de la montre.

25

Aymen Sellaouti

Solution

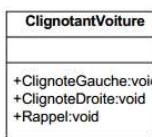


26

Aymen Sellaouti

Exercice

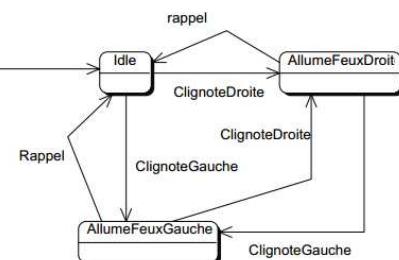
Supposons que la classe « Clignotant voiture » serve à décrire un clignotant de voiture pour signaler à gauche ou à droite. Quand la voiture est démarrée, le clignotant entre dans un état d'attente de commande. Quand la méthode ClignoteGauche est appelée, le clignotant entre dans un état pour lequel le feu arrière gauche clignote. Quand la méthode ClignoteDroite est appelée, le clignotant entre dans un état pour lequel le feu arrière droit clignote. Dans ces deux états de clignotement, le feu droit ou gauche est éteint et le clignotant retourne à son état d'attente de commande lorsque la méthode Rappel est appelée.



27

Aymen Sellaouti

Solution

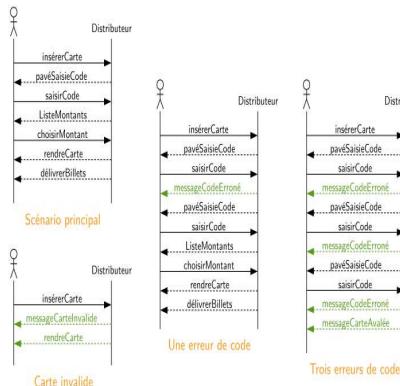


28

Aymen Sellaouti

Exercice

- A partir de ces différents diagrammes de séquence modélisant les différents scénarios d'un distributeur automatique, réalisez un diagramme d'état transition sachant que l'état initial du distributeur est inactif et que l'état final peut être atteint en éteignant le distributeur.

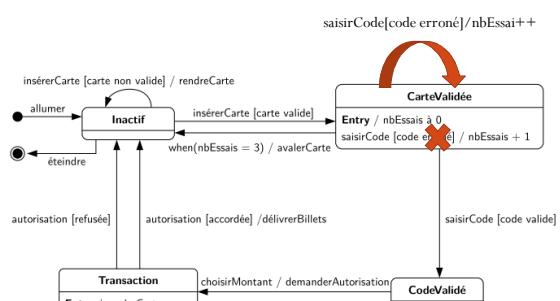


29

Aymen Sellaouti

Solution

Peut-on modifier le diagramme de la sorte ?



30

Aymen Sellaouti

Non parce qu'on changera le contexte de l'état et le nbEssai sera remis à 0 à chaque fois

aymen.sellaouti@gmail.com

Méthodologies d'analyse et de conception logicielles

Diagrammes d'activité

UML
Unified Modeling Language
aymen.sellaouti@gmail.com



Diagrammes dynamiques

Génie logiciel et méthodes de conception orientées objet

UML Modélisation dynamique

A. ABDELLATIF
abdelaziz.abdellatif@fst.rnu.tn
2007-2008

Diagramme d'états transitions
Diagramme d'activités

Sonia Bouzidi
INSAT

Aymen Sellaouti

Introduction

- Permet de modéliser les traitements.
- Permet de représenter graphiquement le comportement d'une méthode ou la réalisation d'un cas d'utilisation.
- Assez proche du diagramme d'états-transitions.
- Se base sur une représentation suffisamment précise pour pouvoir générer du code.

Aymen Sellaouti

Concepts de base

Le diagramme d'activités se base sur les concepts suivants:

- **Activité** : C'est une **étape** dans la **réalisation** d'une **méthode** ou un **cas d'utilisation**.
- **Décision** : C'est une façon de **représenter** les **condition** de déclenchement des **activités**.
- **Synchronisation** : C'est un mécanisme permettant de représenter le **parallelisme d'exécution** des **activités**.
- **Unité organisationnelle** : Permet de répondre à la question « **qui fait quoi ?** ».

Aymen Sellaouti

Structuration d'un diagramme d'activité

Un diagramme d'activités se présente comme un graphe composé de nœuds inter reliés par des arcs orientés :

- **Nœuds** : Chaque nœud peut correspondre à :
 - Une activité (début, exécutables, fin)
 - Une décision
 - Une synchronisation
- **Arcs** : Chaque arc représente une transition entre nœuds.

Aymen Sellaouti

Les activités

- Un diagramme d'activités est un enchaînement d'un ensemble d'activités.
- Une activité correspond à une étape d'une méthode ou d'un cas d'utilisation.
- Une activité est l'équivalent d'une transition dans un diagramme d'états-transitions permettant de faire passer un objet d'un état à un autre.
- Tout diagramme d'activité :
 - commence par une activité de départ (start),
 - comporte un ensemble d'activités exécutables,
 - se termine par une ou plusieurs activités de fin (end).

7 Aymen Sellaouti

Représentation graphique des activités



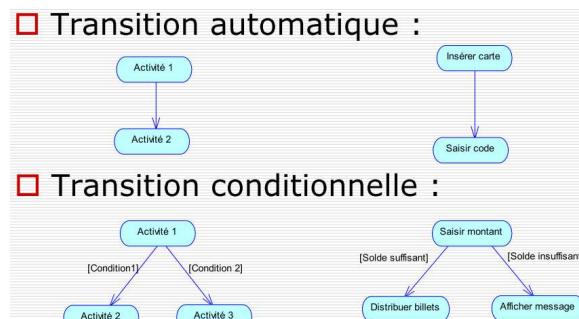
8 Aymen Sellaouti

Les transitions

- Une **transition** représente le passage d'une activité à une autre.
- Ce passage peut être :
 - **Automatique** : la fin d'une activité entraîne le déclenchement automatique d'une autre activité.
 - **Conditionnel** : le déclenchement de la deuxième activité ne peut se faire que :
 - Si une condition est vérifiée alors l'utilisation d'un nœud de type **décision**.
 - Après la fin d'une ou plusieurs autres activités alors l'utilisation d'un nœud de type **synchronisation** ou **fusion**.

9 Aymen Sellaouti

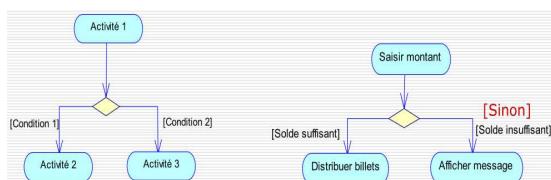
Représentation graphique d'une transition



10 Aymen Sellaouti

Les décisions

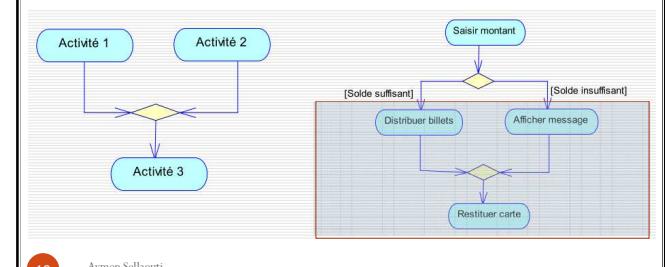
Un nœud de **décision** permet d'assurer la **transition conditionnelle** d'une activité vers **deux ou plusieurs autres activités mutuellement exclusives**.



11 Aymen Sellaouti

Les fusions

- Un nœud de **fusion** (merge) permet d'assurer la transition de plusieurs activités **alternatives** vers une autre activité.



12 Aymen Sellaouti

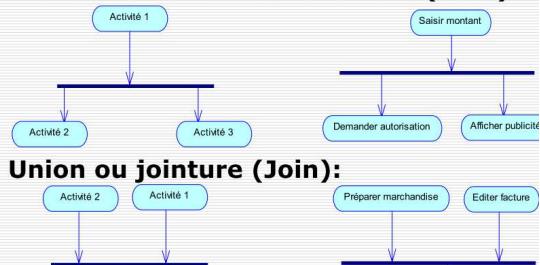
Les synchronisations

- Un nœud de **synchronisation** permet d'ouvrir et de fermer des branches parallèles dans un diagramme d'activités.
- Il peut s'agir d'un nœud de :
 - **Bifurcation ou débranchement (Fork)** : la fin d'une activité entraîne le **déclenchement parallèle** de deux ou plusieurs activités.
 - **Union ou jointure (Join)** : le **déclenchement** d'une activité nécessite la fin d'exécution de deux ou plusieurs activités parallèles.

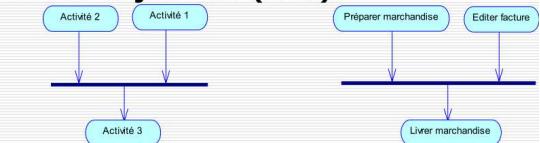
13 Aymen Sellaouti

Représentation graphique des synchronisations

Bifurcation ou débranchement (Fork) :



Union ou jointure (Join) :



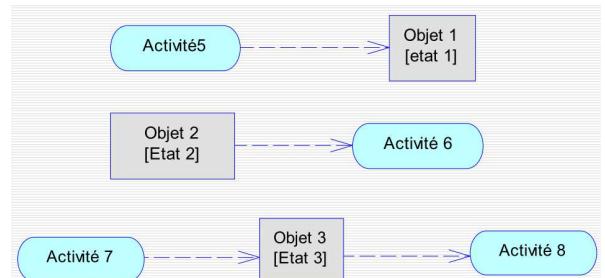
14 Aymen Sellaouti

Objets

- On peut préciser dans un diagramme d'activités **les objets** manipulés.
- Un objet peut être un point d'entrée et/ou un point de sortie d'une activité.
- Lorsque le même objet apparaît plusieurs fois dans un diagramme d'activités, on peut préciser ses différents **états**.

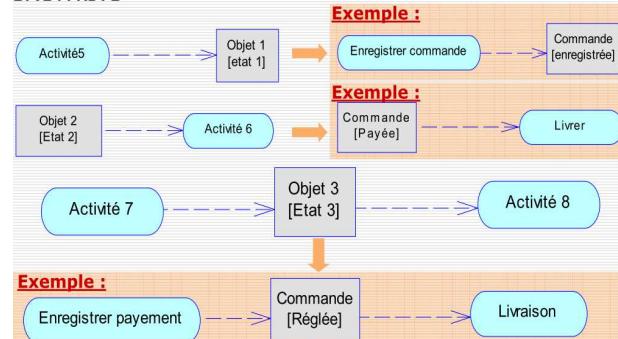
15 Aymen Sellaouti

Représentation graphique d'un objet



16 Aymen Sellaouti

Représentation graphique d'un objet : exemple



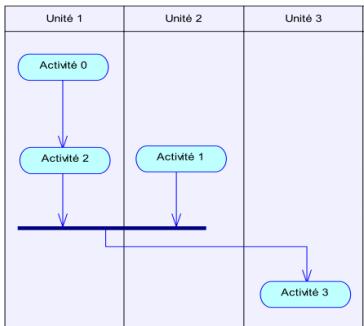
17 Aymen Sellaouti

Les unités organisationnelles

- On peut préciser dans un diagramme d'activités les **acteurs** qui réalisent les activités.
- Un **acteur** peut être une **personne** ou une **structure**. On parle d'**unité organisationnelle**.
- Les unités organisationnelles permettent de répondre à la question: **QUI FAIT QUOI ?**.
- Autres appellations : partitions, couloirs d'activité **swimlanes**, couloirs d'eau.

18 Aymen Sellaouti

Représentation graphique d'une unité organisationnelle



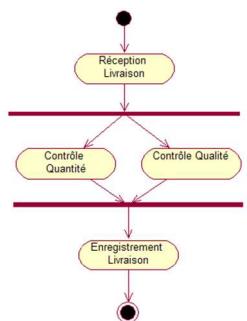
19 Aymen Sellaouti

Exercice

- Modéliser le diagramme d'activité de la réception d'une commande.
- Pour valider la commande et donc valider l'enregistrement de la livraison, le récepteur doit vérifier la quantité et la qualité du produit.

20 Aymen Sellaouti

Solution possible



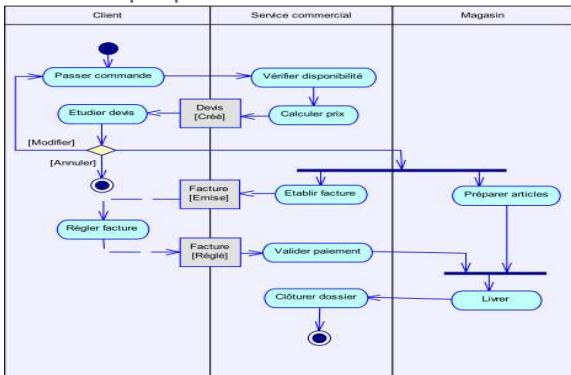
21 Aymen Sellaouti

Exercice

- On veut modéliser la procédure de livraison d'une vente en ligne d'articles de sports. La procédure commence par une commande passée par le client.
- Une fois la commande passée, le service commercial vérifie la disponibilité des produits (dans cet exemple, on ne prends pas en compte le cas de la non disponibilité du produit). Un devis est généré au client en calculant le prix de la commande.
- Une fois le devis à disposition, le client étudie le devis et il a le choix de modifier sa commande, de l'annuler ou de la valider.
- Une fois la commande validé par le client, le service commercial se charge d'établir la facture alors que le magasin prépare les articles.
- La commande sera livré et le dossier ensuite clôturé si la facture a été réglé par le client et le paiement validé par le service commercial.

22 Aymen Sellaouti

Solution proposée



23 Aymen Sellaouti

Synthèse

- Utiliser les diagrammes de cas d'utilisation pour décrire les besoins des utilisateurs, les fonctions du nouveau système, ...
- Utiliser les diagrammes de séquence et les diagrammes de collaboration pour décrire les scénarios des cas d'utilisation.
- Utiliser les diagrammes d'états-transition pour décrire la dynamique des objets.
- Utiliser les diagrammes d'activités pour décrire de façon détaillée les méthodes et les cas d'utilisation.

24 Aymen Sellaouti

Méthodologies d'analyse et de conception logicielles

Diagrammes de composants

UML
Unified Modeling Language
aymen.sellaouti@gmail.com



IFT2255 : Génie logiciel

Chapitre 5. Conception
Section 5. UML et conception architecturale

Julie Vachon

Aymen Sellaouti

3

Diagrammes de composants(Component diagram) & Diagrammes de déploiement(Deployment diagram)

Sous Exercice

Génie logiciel et méthodes de conception orientées objet

UML Modélisation statique

A. ABDELLATIF abdelaziz.abdellatif@fci.mn.tn 2007-2008

Introduction

- Un diagramme de composant décrit la façon avec laquelle les composantes logiciels seront réalisées.
- Il représente les choix de réalisation.
- Il est destiné aux réalisateurs (développeurs).
- Il décrit le système modélisé sous forme de composants réutilisables et met en évidence leurs relations de dépendance.

4 Aymen Sellaouti

Éléments de base

- **Composant :**
 - Élément physique qui représente une partie implémentée d'un système.
- **Interfaces d'un composant :**
 - Élément définissant le comportement offert à d'autres composants.
- **Paquetages :**
 - Les composants peuvent être organisés en paquetages, qui définissent des sous-systèmes.
 - Ils permettent de gérer la complexité, par encapsulation des détails d'implémentation.

5 Aymen Sellaouti

Composant

- Un composant doit fournir un service bien précis.
- Il encapsule des fonctionnalités cohérentes entre elles et génériques

→ doit être réutilisable

- Un composant est une unité autonome comportant une ou plusieurs interfaces requises ou offertes.
- Le comportement interne est réalisé par un ensemble de classes masquées : seules ses interfaces sont visibles

6 Aymen Sellaouti

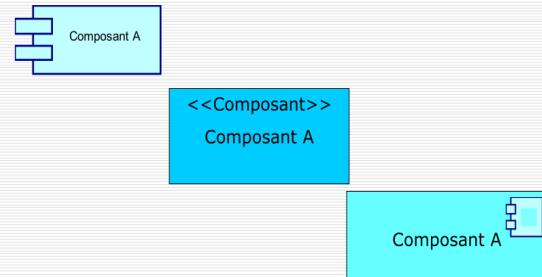
Composant

- **Unité autonome** faisant partie d'un **système** ou d'un **sous-système** qui **encapsule** un **comportement** (i.e. implémentation) et qui **offre** une ou plusieurs **interfaces publiques**.
- Partie constituante d'un système qui peut être **remplacée** ou/et **réutilisée**.
- Élément **d'implémentation** (un sous-système, un fichier exécutable, une classe d'implémentation (i.e. non abstraite), etc.) muni **d'interface(s)**.
- Chaque composant est le représentant d'une ou plusieurs classes qui implémentent un service à l'intérieur du système.
- **Granularité?** Un composant peut représenter quelque chose d'aussi fin qu'un **objet**, comme il peut représenter un **sous-système complexe**.

7 Aymen Sellaouti

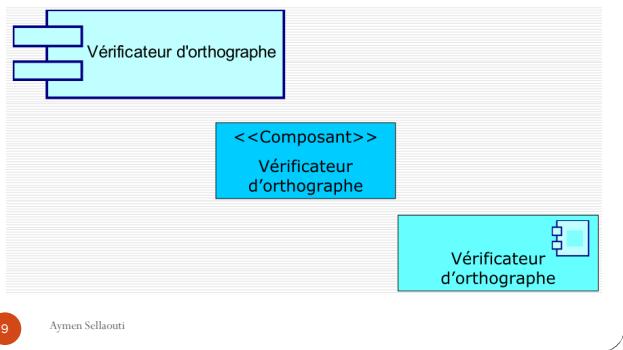
Représentation graphique

Trois représentations possibles :



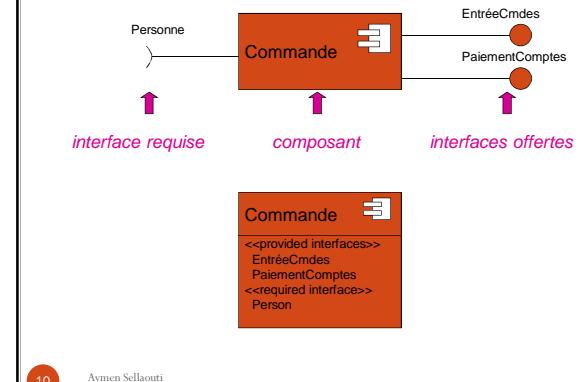
4 Aymen Sellaouti

Représentation graphique : exemple



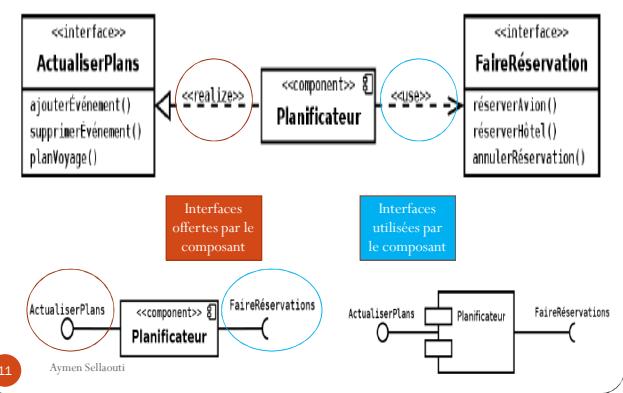
9 Aymen Sellaouti

Interfaces et composants



10 Aymen Sellaouti

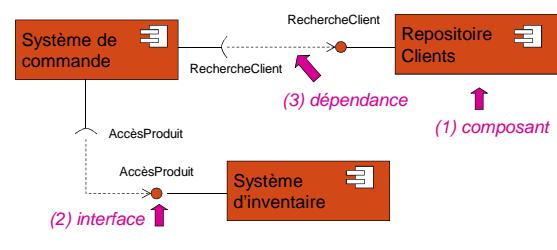
Interfaces et composants



11 Aymen Sellaouti

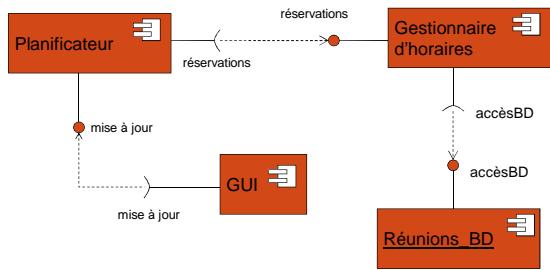
Composants et relations

- Une flèche de dépendance permet de mettre en relation des composants via les interfaces requises et celles fournies.



12 Aymen Sellaouti

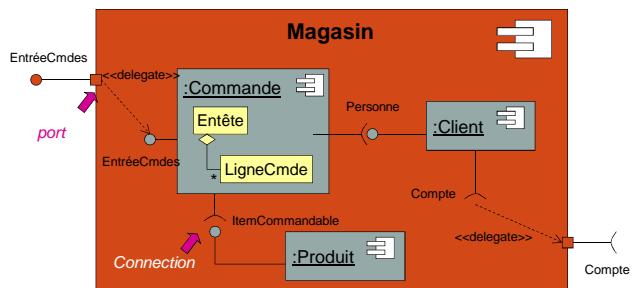
Composants et relations



13 Aymen Sellaouti

Composants : Vue de la structure interne

- Il est parfois utile de pouvoir montrer la structure interne d'un composant.



14 Aymen Sellaouti

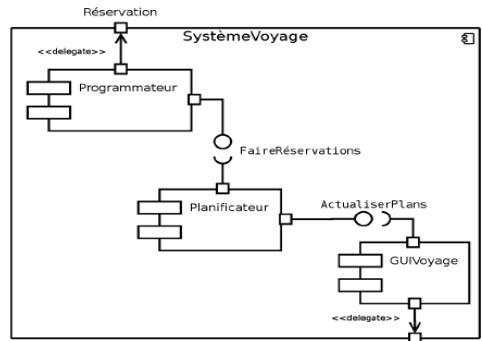
Port



- Un port est un point de connexion entre un classeur et son environnement
- L'utilisation des ports permet de modifier la structure interne d'un classeur sans affecter les clients externes

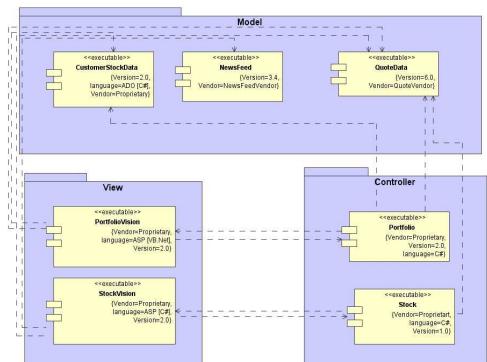
15 Aymen Sellaouti

Exemple d'un composant complexe



16 Aymen Sellaouti

Diagramme de composants décrivant une architecture MVC



17 Aymen Sellaouti

Méthodologies d'analyse et de conception logicielles

Diagrammes de déploiement

UML
Unified Modeling Language
aymen.sellaouti@gmail.com



3 Aymen Sellaouti

Cliquez pour ajouter un titre
Diagrammes de composants(Component diagram)
&
Diagrammes de déploiement(Deployment diagram)

Sous-titre
None

IFT2255 :
Génie logiciel
Chapitre 5. Conception
Section 5. UML et conception architecturale
Julie Vachon

Génie logiciel et méthodes de conception orientées objet
UML Modélisation statique
A. ABDELLATIF
abdelaziz.abdellatif@fai.mn.tn
2007-2008

Introduction

- Représente la structure physique du système informatique et la répartition des composantes logicielles sur ce système.
- Définis la répartition des composants sur les nœuds d'exécution physiques
- Constitue un moyen pour représenter les différentes architectures selon laquelle le nouveau logiciel peut être déployé :
 - Architecture centralisée
 - Architecture client/serveur
 - Architecture 3 tiers
 - Architecture n-tiers

4 Aymen Sellaouti

5 Aymen Sellaouti

Éléments de base

- Le diagramme de déploiement se base sur les concepts suivants :

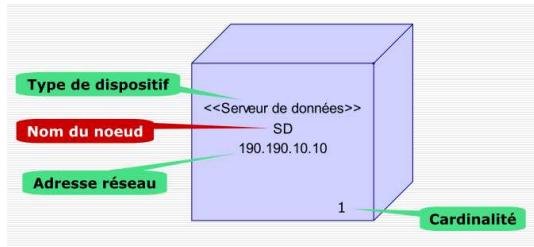
- **Nœud** : Description d'un dispositif matériel
- **Composant** : Composante logicielle.
- **Association de nœud** : Relation entre deux nœuds.
- **Instance de nœud** : C'est une occurrence d'un dispositif matériel.

6 Aymen Sellaouti

Les nœuds

- Chaque dispositif matériel est représenté par un nœud.
- Un nœud est décrit par :
 - Un nom (obligatoire)
 - Un stéréotype (serveur, client, imprimante, routeur, ...)
 - Une adresse réseau
 - Une cardinalité

Représentation graphique d'un nœud



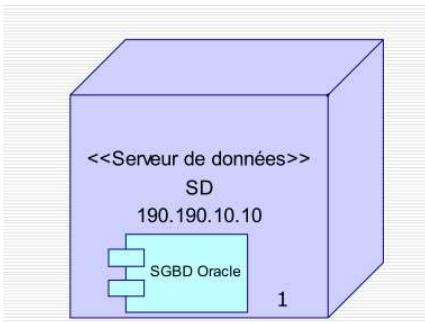
7 Aymen Sellaouti

Les composants

- Dans chaque nœud, on peut indiquer les composants logiciels qui vont y être installés.
- Les composants ont le même formalisme que celui du diagramme de composants.

8 Aymen Sellaouti

Représentation graphique d'un composant dans un nœud



9 Aymen Sellaouti

Les associations

- Les nœuds sont inter reliés par des associations.
- Chaque association entre deux nœuds représente l'existence d'un support de communication entre eux.
- Pour chaque association, on peut préciser :
 - Le nom
 - Le rôle de chaque côté
 - La cardinalité

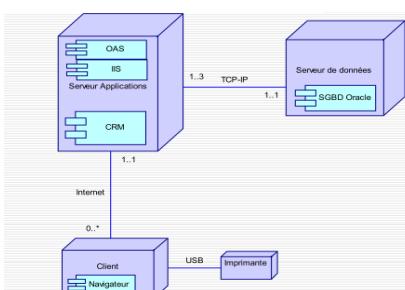
10 Aymen Sellaouti

Représentation graphique d'une association



11 Aymen Sellaouti

Exemple d'utilisation d'associations



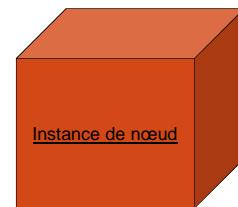
12 Aymen Sellaouti

Les instances de nœud

- Dans un diagramme de déploiement, on peut représenter les instances des nœuds.
- Cette possibilité est utilisée surtout pour représenter de façon exhaustive des parties du diagramme de déploiement.
- Les instances de nœuds se distinguent des nœuds par le fait que leurs noms sont soulignés.

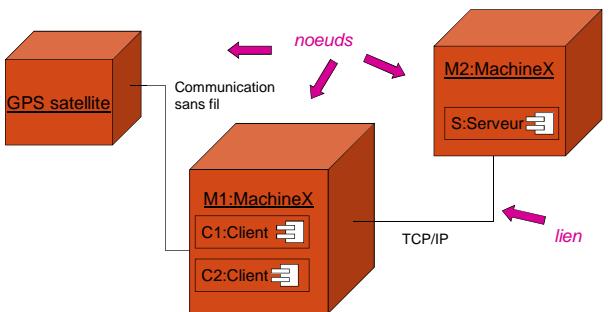
13 Aymen Sellaouti

Représentation graphique d'une instance de nœud



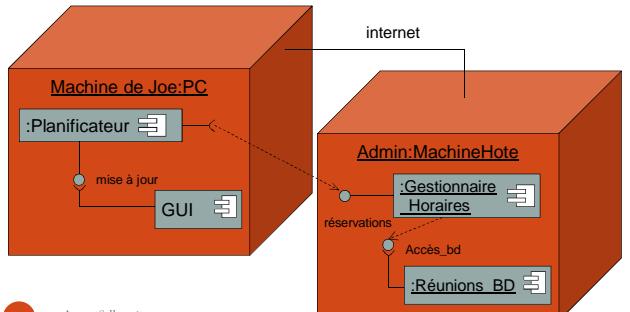
14 Aymen Sellaouti

Exemple d'une instance de nœud



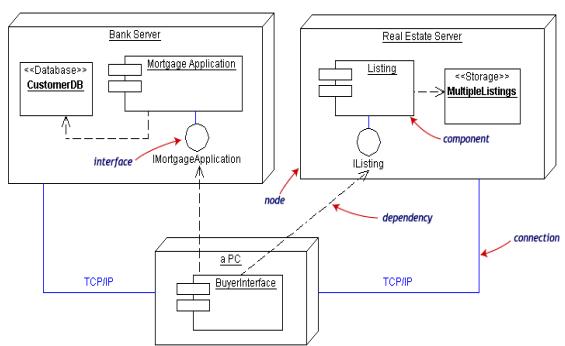
15 Aymen Sellaouti

Exemple



16 Aymen Sellaouti

Exemple



17 Aymen Sellaouti

Synthèse sur les diagrammes statiques

- Utiliser les **diagrammes de classes** pour décrire des composantes structurelles d'un SI.
- Utiliser les **diagrammes d'objets** pour illustrer et tester les diagrammes de classes.
- Utiliser les **diagrammes de composants** pour montrer la façon selon laquelle le logiciel sera découpé en composantes.
- Utiliser les **diagrammes de déploiement** pour envisager les différentes façons selon lesquelles le logiciel peut être exploité.

18 Aymen Sellaouti