UML
0 - Introduction : UML et la méthode

Bertrand LIAUDET

USAGE du document *: les paragraphes précédés d'un ( ) sont des présentations détaillées qu'on peut sauter dans un premier temps.*

SOMMAIRE

INTRODUCTION : UML ET LA METHODE 2

1. Introduction à la méthode 2

2. Introduction UML 2

Présentation UML 2
5 Exemples : 3
Vocabulaire, notions 8
Les 4 diagrammes plus importants pour conception 9
Syntaxe 9
Outils 10
Classification des diagrammes 11
Historique 13
Bilan 14
Perspectives 14
But du cours : un usage actuel d'UML ! 15

3. Classification détaillées des diagrammes UML 16

() Classification des 14 diagrammes UML 2.2 avec héritage 16
Les 4 principaux diagrammes d'UML pour ce cours 18
() 3 autres diagrammes importants d'UML 22
() Les 7 diagrammes d'UML secondaires 24

4. UML et le cycle en V 25

Les différents diagrammes selon l'étape de la conception 25
( ) Autre point de vue 27
( ) Autre point de vue 28

5. Bibliographie 29

Référence du cours 29
Ce cours 29
() Théorie UML 29
() Pratique 29
() Internet 29

Edition janvier-2025

INTRODUCTION : UML ET LA METHODE

Il est facile de décrire la méthode encore que
son application exige à coup sûr savoir et pratique.
La méthode est dénuée de sens tant qu'elle est déconnectée du rapport au savoir.

***USAGE du document** : les paragraphes précédés d'un ( ) sont des présentations détaillées qu'on peut sauter dans un premier temps.*

1. Introduction à la méthode

- Voir le cours d'introduction à la méthode
  - 1. (pour UML) Généralités sur la méthode
  - 2. (pour UML) Développement d'un logiciel : les 5 distinctions capitales (en partie)
  - 3. (pour UML) Le cycle en V (en partie)

2. Introduction UML

Présentation UML

Sigle

- **UML** : Unified Modeling Language : Langage de Modélisation Unifié
  - En réalité : **Des** langages de modélisation, unifiés (on peut utiliser certaines syntaxes d'un langage dans un autre langage).

Des langages (pas un langage !)

- **UML** propose un **ensemble de langages graphiques pour l'analyse des architectures et des processus d'un système.** Ces langages qui permettent de réaliser toutes les étapes de la conception.
  - L'UML permet d'avoir un **langage commun** (en réalité, plusieurs).

Méthode

- **L'UML** propose **des langages, pas une méthode : on peut en faire ce qu'on veut !**
  - Mais pour avoir une présentation concrète et pratique, on va suivre une méthode !

Diagramme

- Chaque analyse réalisée en UML s'appelle un « **diagramme** ».
  - Par exemple le diagramme des cas d'utilisation ou le diagramme des classes.
  - Un diagramme, c'est un modèle graphique.

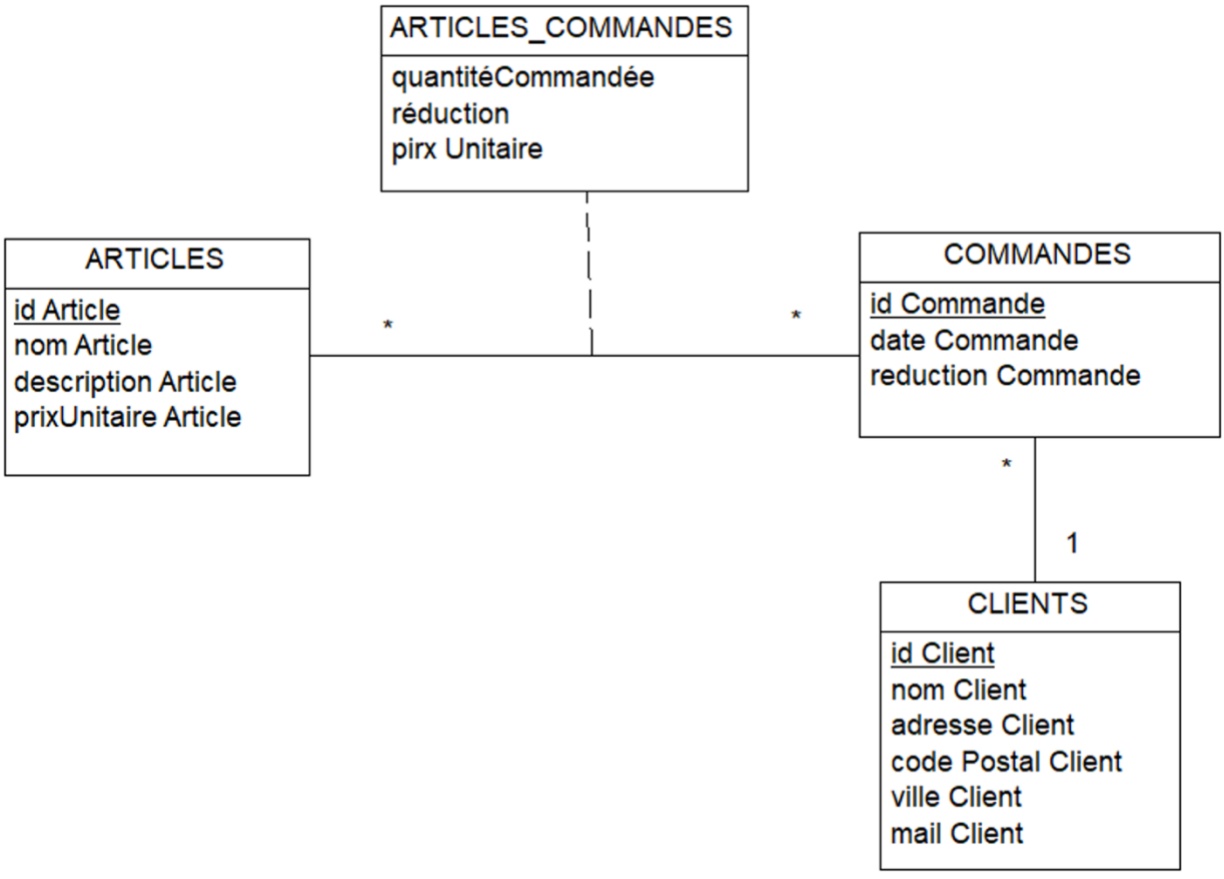
14 diagrammes différents, dont 4 utiles !

- UML propose **14 diagrammes différents** qui sont adaptés à une ou plusieurs étapes de la conception.
- On en utilisera que 4 pour faire toute la conception : Classes, Use Case, Séquence, Activités.

5 Exemples :

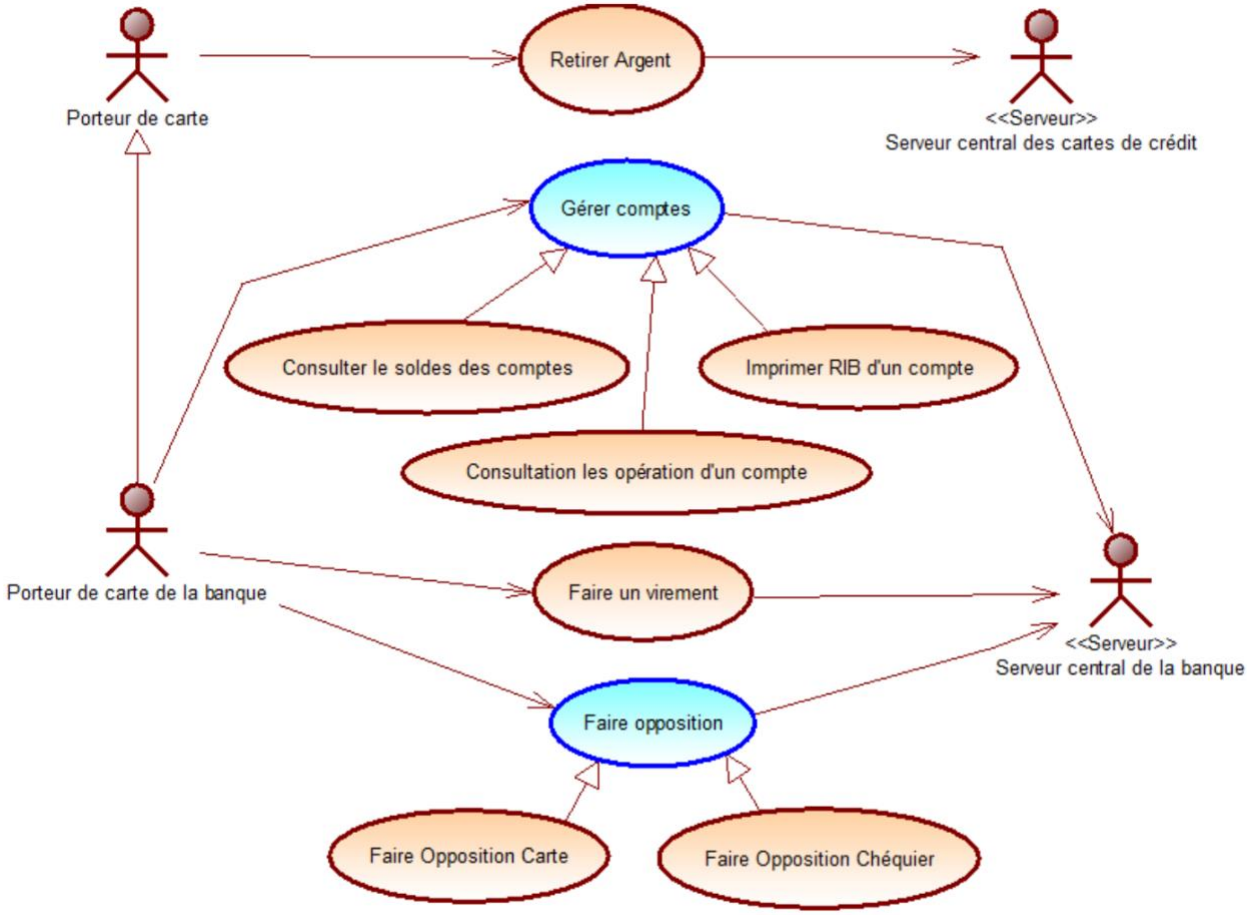
1 : Diagramme de Classes métier

Exemple d'un pattern de diagramme de Classes métier pour gérer des commandes



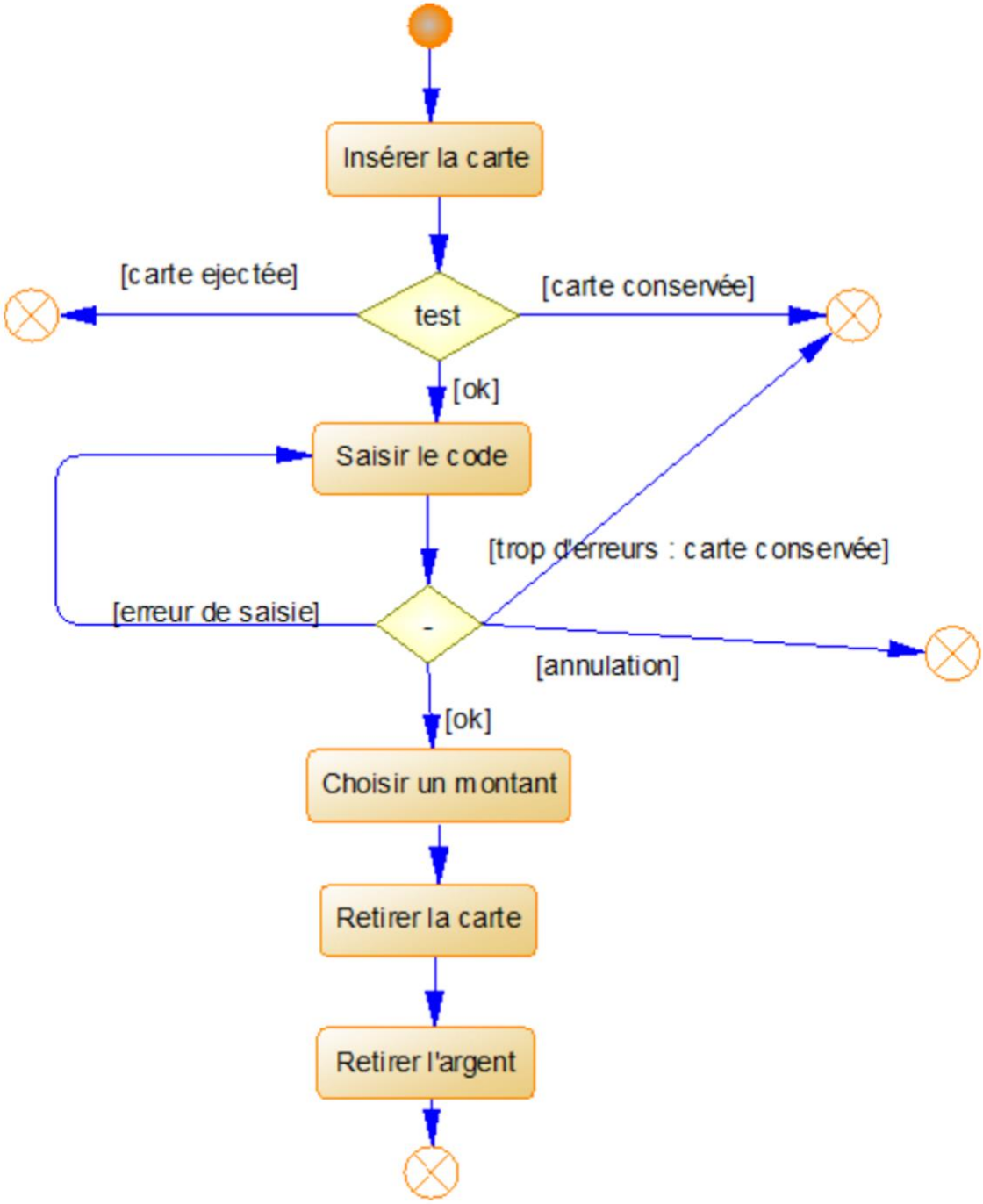
2 : Diagramme de Cas d'Utilisation (Use case) :

Exemple d'un pattern de cas d'utilisation pour un GAB (guichet automatique de banque) :



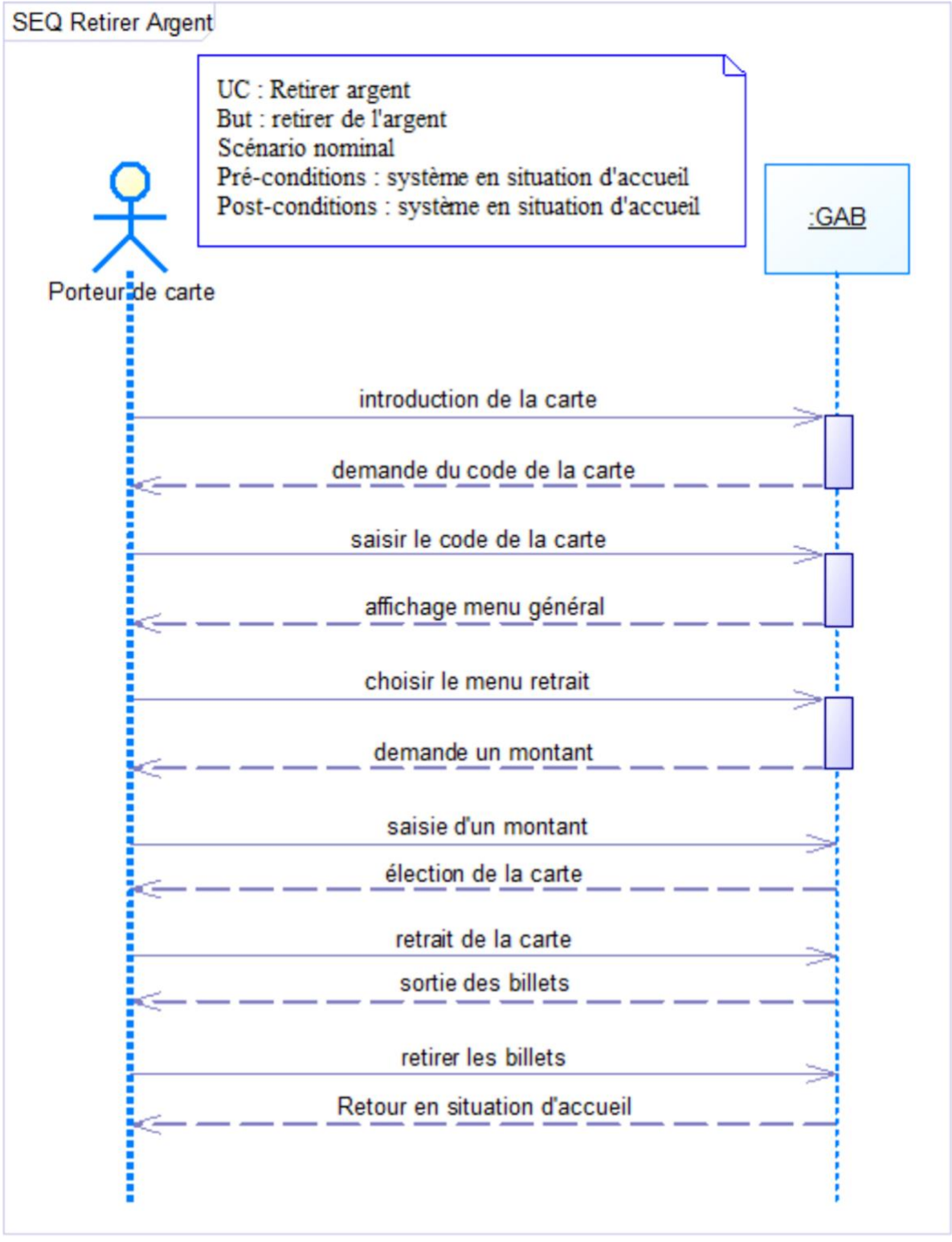
3 : Diagramme d'Activités

Exemple du diagramme d'activités pour un retrait d'argent sur un GAB :



4 : Diagramme de Séquence système

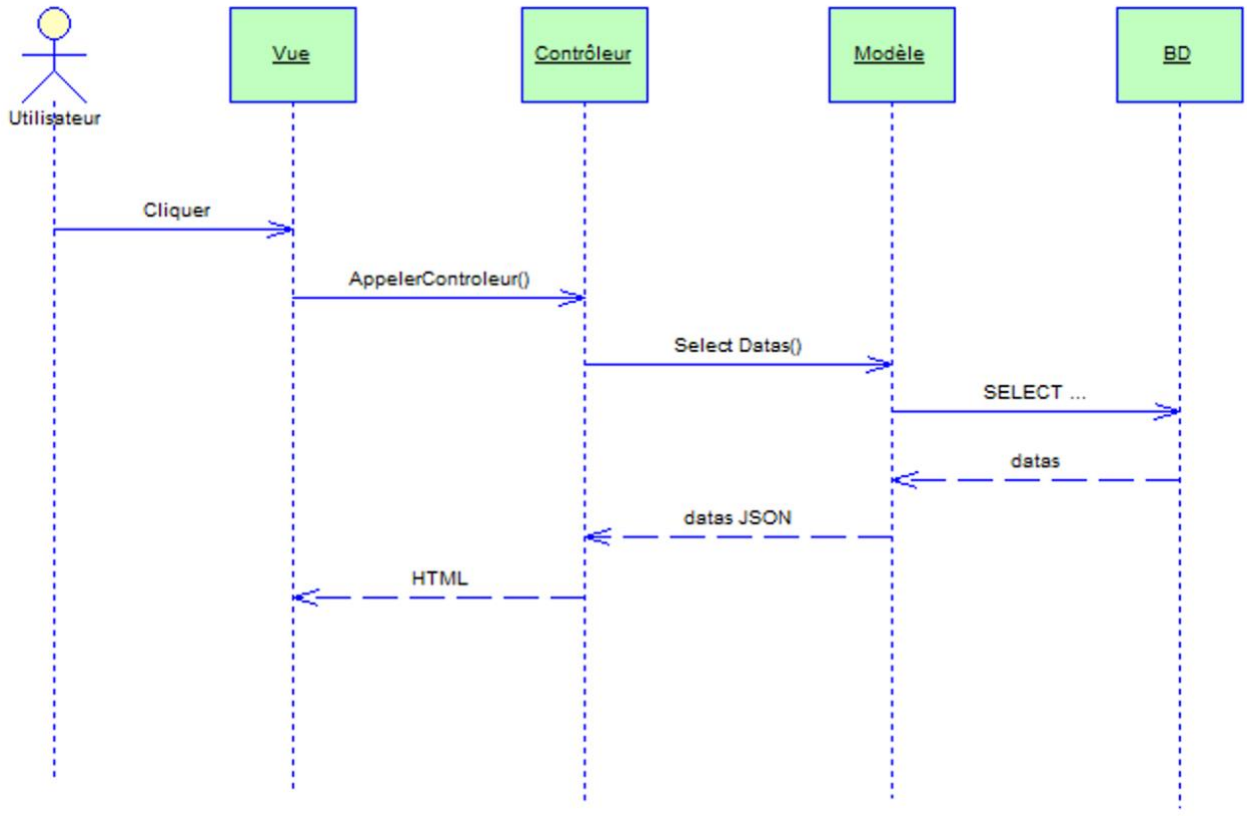
Exemple d'un diagramme de séquence système pour un retrait d'argent sur un GAB :



5 : Diagramme de Séquence MVC

Principe général d'un diagramme de séquence MVC :

1. L'utilisateur clique sur une vue (son écran).
2. La vue appelle un contrôleur.
3. Le contrôleur appelle une méthode « select datas() » d'un modèle.
4. Le modèle fait une requête SELECT … à la BD
5. La BD retourne des datas
6. Le modèle retourne les datas au format JSON
7. Le contrôleur fabrique une nouvelle vue et retourne du HTML
8. Retour en 1 : l'utilisateur voit une nouvelle vue et peut re-cliquer.



Vocabulaire, notions

- **Architecture** : organisation structurelle. Plutôt statique et non séquentiel.
- **Algorithme** : description du déroulement dans le temps d'une action. Dynamique et séquentiel. On peut aussi dire : « processus » ou « procédure ».
- **Modèle** : Représentation simplifiée d'un système ou d'un phénomène pour en faciliter l'analyse.
- **Modélisation** : Processus de création d'un modèle pour étudier ou simuler un système réel.
- **Système** : Ensemble d'éléments interconnectés qui fonctionnent ensemble pour atteindre un objectif commun. Concrètement, un système, pour nous, c'est un programme, une application ou un serveur. A noter qu'un serveur, c'est un programme sans interface utilisateur.
- **Système d'information** : Système utilisé pour collecter, stocker, mettre à jour, traiter et diffuser des informations afin de rendre des services aux utilisateurs et soutenir la prise de décision par les décideurs.

Les 4 diagrammes plus importants pour conception

- Le **diagramme de cas d'utilisation** : pour la description statique des usages du système (il peut être utilisé pour représenter l'arborescence des scénarios UX).

- Le **diagramme de classes** : pour la description statique des classes métier du système. Les classes métier correspondent aux tables de la BD.

- Les **diagrammes de séquence système** : pour les descriptions du comportement (dynamique) de **chaque scénario nominal de chaque cas d'utilisation.** Utilisés dans ce cas d'un point de vue fonctionnel. On peut aussi utiliser les diagrammes de séquence d'un point de vue technique.

- Les **diagrammes d'activités** : pour les descriptions des comportements (dynamiques) de **tous ls scénarios de chaque cas d'utilisation.** Utilisés dans ce cas d'un point de vue fonctionnel. On peut aussi utiliser les diagrammes de séquence d'un point de vue technique ou pour des diagrammes d'activités à travées.

Syntaxe

- L'UML propose un ensemble de **langages formalisés** : les symboles graphiques ont une signification précise. **Il y a donc une syntaxe à connaître !**

Outils

- Pour la conception UML, on peut :
  - ⇒ Faire ça **sur papier** ou utiliser **Draw.io** = http://diagrams.net/ : logiciel de dessin JavaScript en accès libre.
  - ⇒ Faire ça **avec un « modeler UML »** : ce sont des logiciels qui permettent, plus ou moins, de :
    - Travailler sur **l'ensemble des diagrammes** de façon cohérente
    - **Générer du code** automatiquement à partir des modèles.
    - Faire du **reverse engineering** (faire les modèles à partir du code).

- Il existe de nombreux modeler UML permettant de produire une conception UML.
  - StarUML,
  - BoUML,
  - Eclipse – Papyrus
  - SAP – PowerDesigner (anciennement powerAMC)
  - Rational Software Modeler : descendant de « **Rational Rose** », le premier né. Aujourd'hui propriété d'IBM.
  - Etc.
  - PowerDesigner aujourd'hui propriété de SAP (anciennement PowerAMC)

- **chatGPT** (arrivé fin 2022) ne produit pas de graphique (de base), mais peut aider dans l'analyse et peut générer du code à partir de graphique.

- Pour ce cours, les élèves devront utiliser un modeler pour le projet, celui qu'ils veulent en fonction de leurs usages. ChatGPT peut servir à l'occasion. Draw.io est pratique mais manque de cohérence entre les diagrammes.

- Les exemples et exercices seront montrés au tableau ou probablement sous PowerDesigner.

Classification des diagrammes

Il y a plusieurs classifications possibles

Classification des diagrammes UML – 1 : structure vs comportement :

- **Les diagrammes de structure** décrivent l'organisation structurelle des composants du logiciels.
  - Organisation structurelle veut dire : statique (ça ne décrit pas un comportement mais un état).
  - Organisation structurelle veut dire : architecture.
    - ⇒ Par exemple : des menus, des classes, fichiers, dossiers, packages, machines, etc.

- **Les diagrammes de comportement** décrivent les comportements du logiciel et/ou de ses parties.
  - Comportement veut dire : dynamique (ça ne décrit pas un état mais un déroulement dans le temps = une séquence).
    - ⇒ Par exemple des cas d'utilisation, séquences, activités, états-transitions, etc.
    - ⇒ A noter que les diagrammes de cas d'utilisation décrivent une architecture des cas d'utilisation. Un cas d'utilisation est un comportement, mais le diagramme des cas d'utilisation est plutôt une structure, une architecture.

Classification des diagrammes UML – 2 : statique vs dynamique = modèle vs algo :

- **les diagrammes statiques (modèle)** décrivent une situation statique.
  - L'organisation statique c'est aussi ce qu'on appelle « l'architecture ».
    ⇒ Par exemple des classes, des objets, des cas d'utilisation, des fichiers, des dossiers, des packages, des machines, etc.

- **les diagrammes dynamiques (algorithme)** décrivent un déroulement dans le temps, autrement dit une séquence (ou un film). Ils décrivent dont une dynamique.
  ⇒ Par exemple les diagrammes de séquences, d'activités, d'états-transitions, etc.
  ⇒ A noter que les diagrammes de cas d'utilisation décrivent une architecture des cas d'utilisation. Un cas d'utilisation est un comportement, mais le diagramme des cas d'utilisation est plutôt une structure, une architecture.

Classification des diagrammes UML – 3 : fonctionnel vs technique (et mixte) :

- **Les diagrammes fonctionnels** servent pour l'analyse fonctionnelle.
  ⇒ Par exemple le diagramme des cas d'utilisation, mais aussi les diagrammes de séquences et d'activités.
  ⇒ Les diagrammes de classes métier relèvent du fonctionnel.

- **Les diagrammes techniques** servent pour l'analyse technique.
  ⇒ Par exemple les diagrammes de classes techniques, les diagrammes d'objets, les diagrammes d'états-transitions, etc.

- **Les diagrammes mixtes** servent pour l'analyse fonctionnelle et pour l'analyse technique.
  ⇒ Par exemple le diagramme de séquence, le diagramme d'activités, le diagramme de classes.

Historique

Fin 80, Années 90 :

« **Crise du logiciel** ». Le paradigme procédural atteint ses limites avec l'accroissement exponentiel du développement logiciel.

Premières utilisations des langages orientés objets dans l'industrie.

Fort développement du C++ et du Java.

Plusieurs dizaines de méthodes objets. Signalons particulièrement : **OMT** (Object Modeling Technique, Rumbaugh), **OOD** (Object Oriented Design, Booch), **OOSE** (Object Oriented Software Engineering, Jacobson)

1995	**UML 0.8** de Booch, Jacobson et Rumbaugh. Première ébauche publique.
1996	UML 0.9 de Booch, Jacobson et Rumbaugh
1997	UML 1.0
1999	**RUP**, Le processus unifié de développement logiciel, Booch, Jacobson, Rumbaugh, Eyrolles, 1999. Le RUP, c'est une méthode.
1999	UML 1.3
2003	UML 1.5
2004	**UML 2**. UML 2.0 - Guide de référence, Booch, Jacobson et Rumbaugh, CampusPress, 2004

➔ On voit que l'UML est intimement lié à la P.O.O.

➔ Les diagrammes de classes et d'objets sont toujours très utiles quand on fait de la P.O.O : c'est technique.

➔ Aujourd'hui, on utilise l'UML pour la conception de base.

Bilan

- Le **diagramme de classe UML** reste un outil indispensable pour tous ceux qui ont compris que programmer objet, c'est d'abord concevoir objet.
  - ➔ De plus, il permet de faire de la modélisation de BD.

- Le **diagramme de cas d'utilisation (use case, UC)** est un outil d'analyse fonctionnelle et d'architecture fonctionnelle très efficace.
  - ➔ Aujourd'hui, les UC sont remplacés, dans certaines situations, par les User Story agile (US) ou par les scénarios UX (User eXpérience).

- Les **diagrammes de séquences et d'activités** un permettent de détailler l'analyse.

Perspectives

- Principaux **concurrents d'UML** :
  - L'intuition
  - Le langage naturel
  - La facilité offerte par les **frameworks**
  - Les méthodes agiles
  - L'approche UX
  - ChatGPT et l'IA

- Critique d'UML 2024 :
  - **https://jaystechbites.com/posts/2024/uml-alternatives-in-2024/**

But du cours : un usage actuel d'UML !

1 Diagramme des Classes métier

2 Diagramme des Cas d'utilisation détaillé : notion de scénario nominal, diagramme de séquence-système et diagramme d'activités.

3 Maquettage – Prototypage (papier ou avec outil de prototypage)
https://www.blogdumoderateur.com/tools/design/prototypage/

4 Structuration MVC en UML

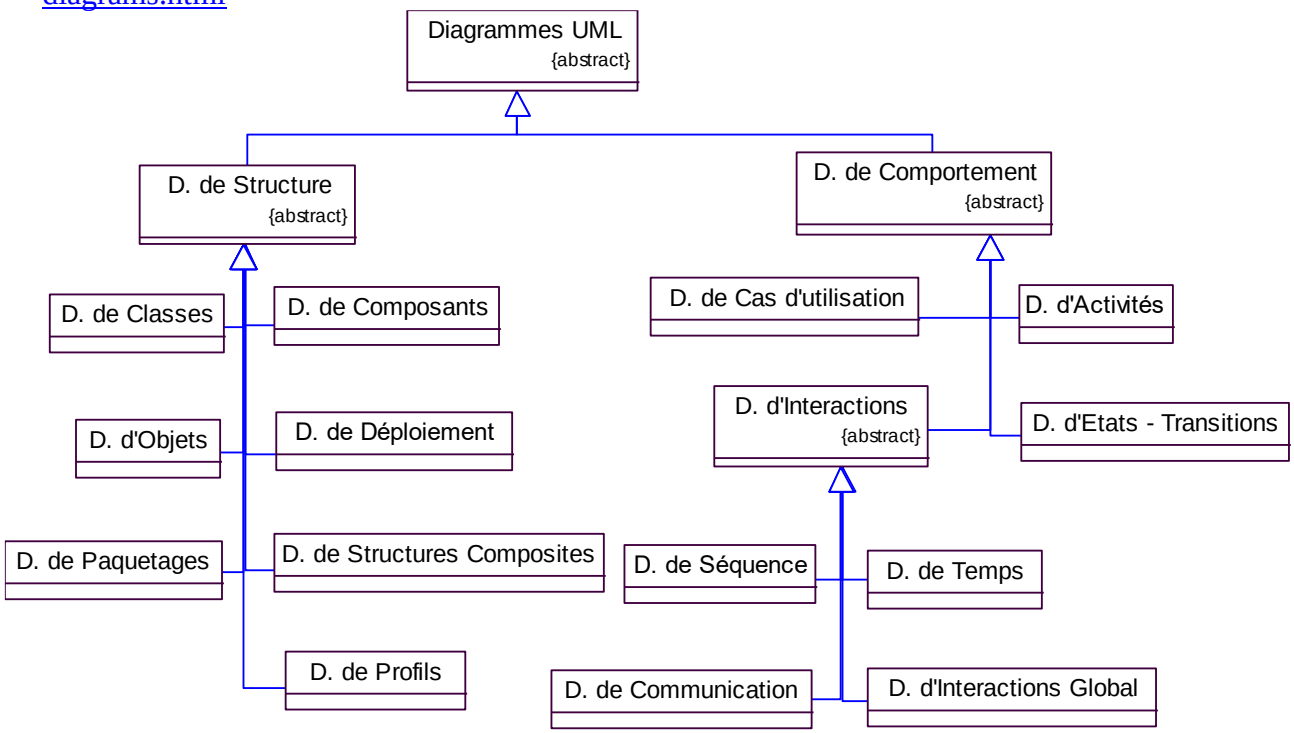
5 Simulation CRUD du SI et de la modélisation UML à partir du MVC

3. Classification détaillées des diagrammes UML

Il y a plusieurs classifications possibles. Comme toujours !

() Classification des 14 diagrammes UML 2.2 avec héritage

- Les diagrammes évoluent un peu avec l'usage : http://www.uml-diagrams.org/uml-24-diagrams.html



7 diagrammes de structures

http://www.uml-diagrams.org/uml-24-diagrams.html#structure-diagram
Les diagrammes de structures correspondent à la description des structures logiques et physiques du logiciel :

6 les classes (structures logiques), les packages, les objets
7 les fichiers (structures physiques)
8 les matériels (structures physiques).

Ils sont tous « statiques ».

7 diagrammes de comportement

http://www.uml-diagrams.org/uml-24-diagrams.html#behavior-diagram
Les diagrammes de comportements correspondent à la description des comportements du logiciel : ils sont **dynamiques** quand leur description correspond à une séquence :

9 diagramme d'activités,
10 diagramme de séquence,
11 diagramme de communication,
12 diagramme d'états-transitions.

Le diagramme des cas d'utilisation :

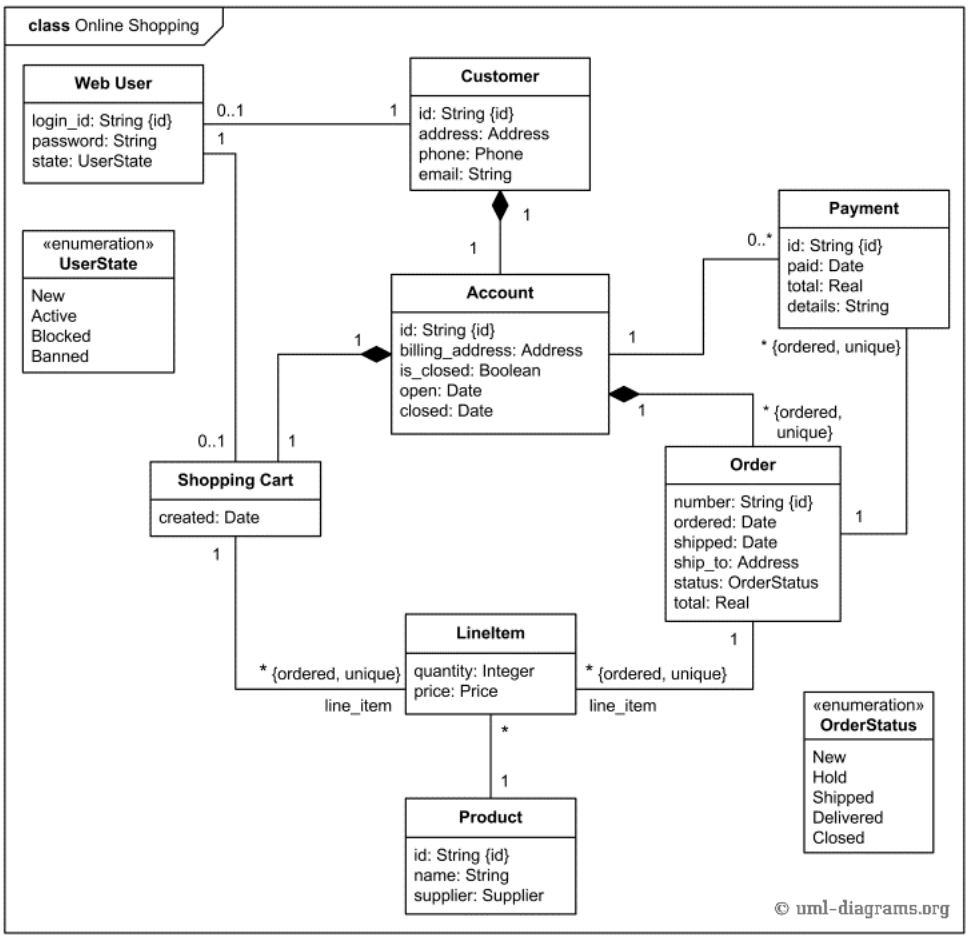
13 Il ne fait que lister les usages possibles, sans notion de séquence : il est donc **statique**.
14 En tant que statique, on pourrait le considérer comme un diagramme de structure (il présente la structure des usages).

Les 4 principaux diagrammes d'UML pour ce cours

- On montre un exemple pour donner une idée de ce à quoi cela ressemble esthétiquement.

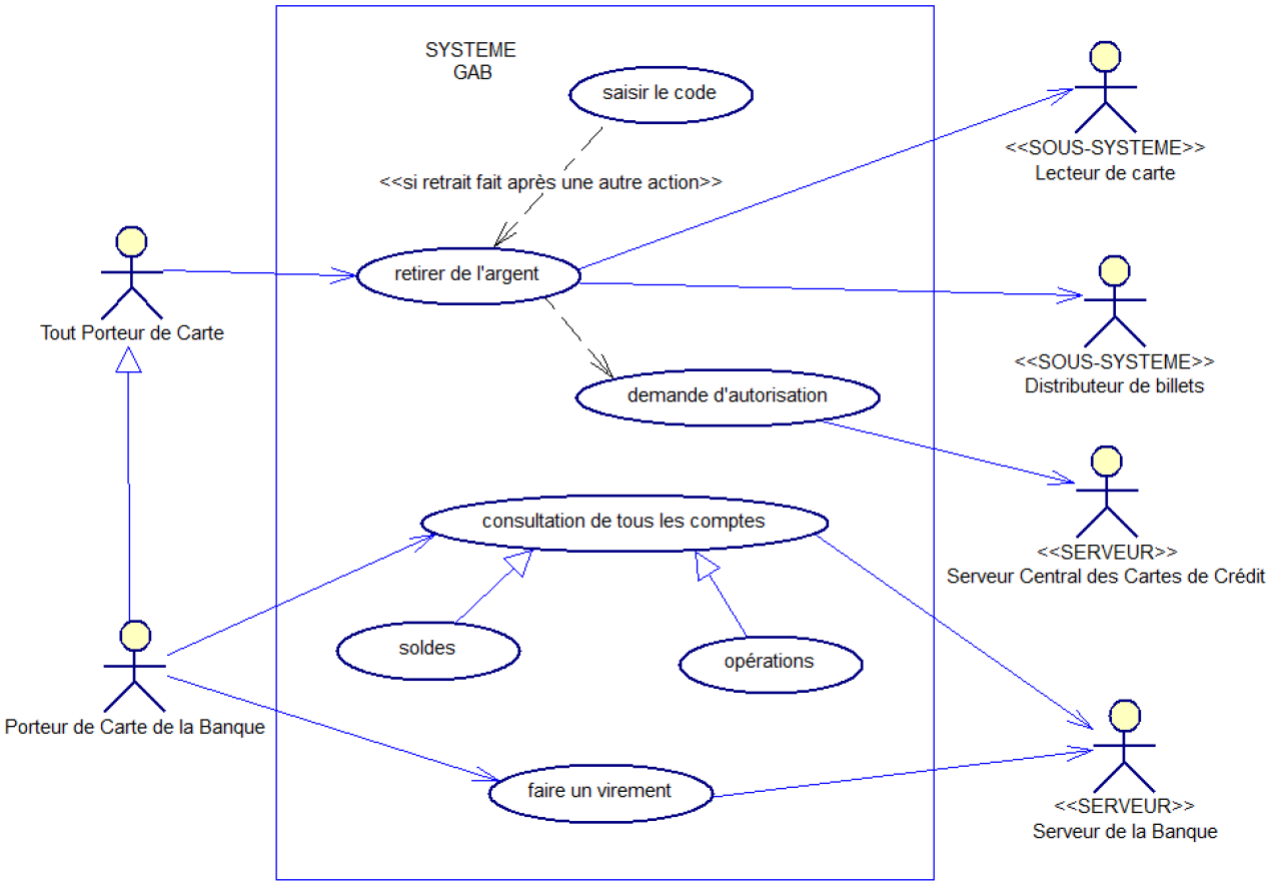
1 : Diagramme de classes

- Il représente la **structure statique** du système en termes de classes et de relations entre les classes.
- **Le diagramme des classes métier** correspond à la base de données. C'est fonctionnel.
- **Le diagramme des classes techniques** détaille toutes les classes, les méthodes, les interfaces et les paquetages. C'est technique.



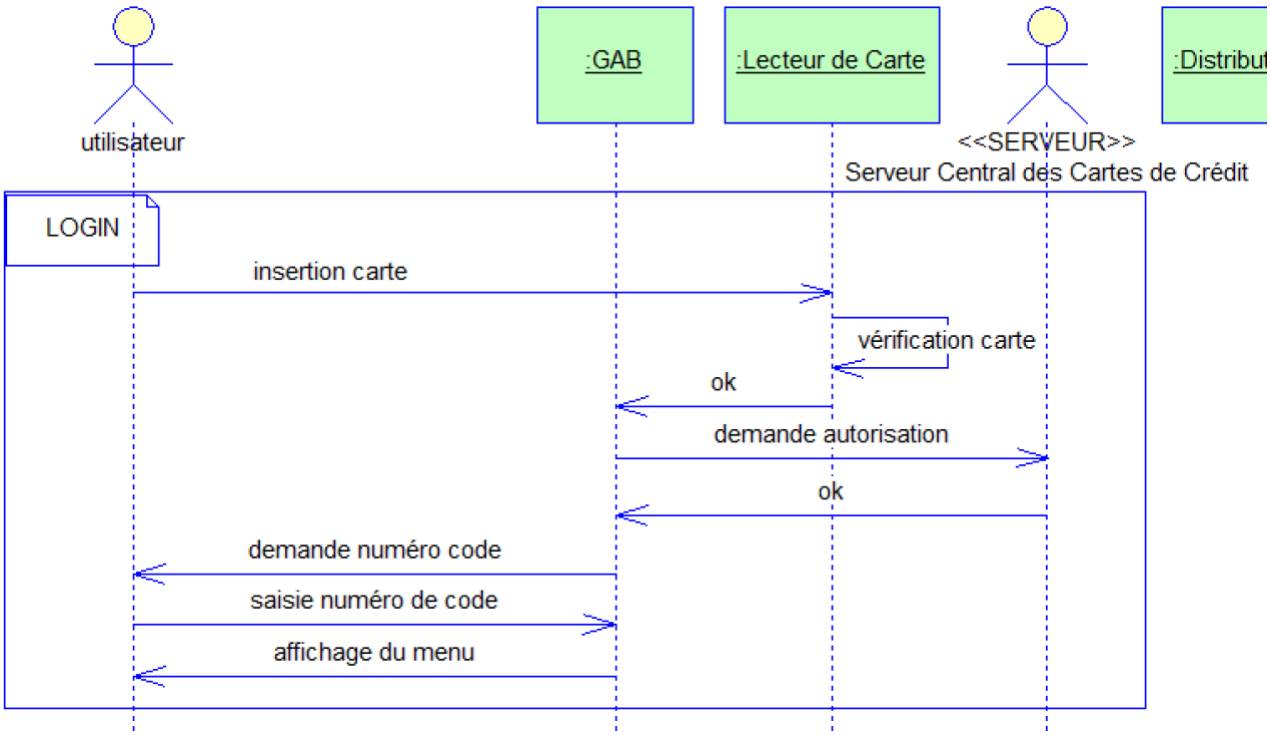
2 : Diagramme de cas d'utilisation

- Il représente les **usages du système** et les différents acteurs de ces usages.
- Un usage, c'est une fonctionnalité complète du système : ce que l'utilisateur est venu faire.



3 : Diagramme de séquence (cf. communication)

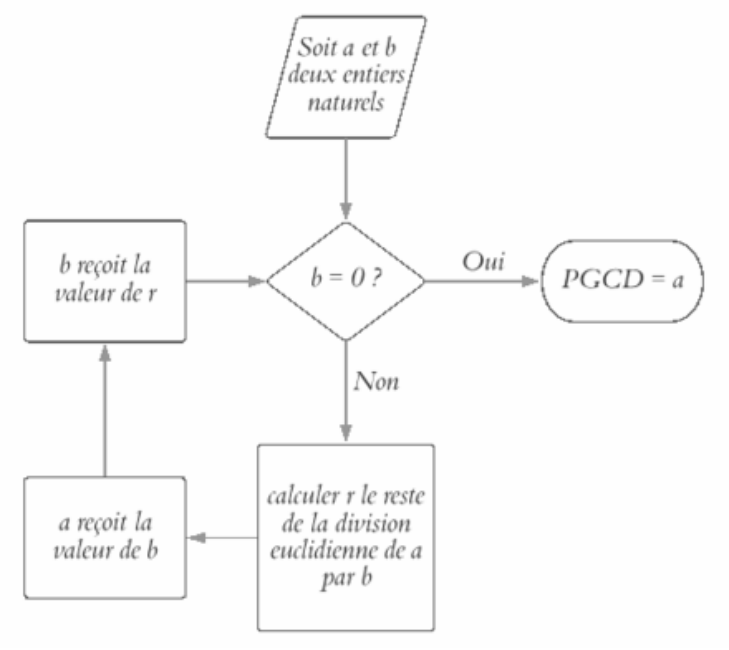
- C'est une espèce de diagramme de communication.
- Tout comme les diagrammes de communication, il montre **les interactions** entre des composants du système dans une séquence temporelle.
- **Le diagramme de séquence objet** montre des interactions entre des « objets » au sens de la POO : il montre les appels de méthode (=fonction) entre objets. C'est technique.
- **Le diagramme de séquence système** montre des interactions entre les « acteurs » et un « système » ou entre les « systèmes ». Il permet de représenter le déroulement des scénarios (instances des cas d'utilisation). C'est fonctionnel.



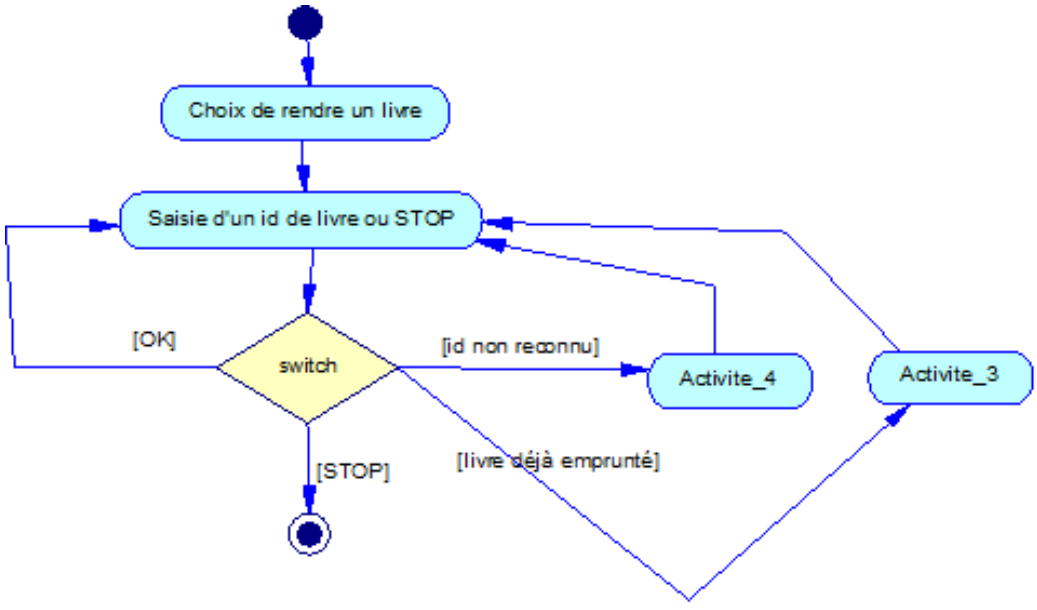
4 : Diagramme d'activité (cf. états-transition)

- Le diagramme d'activités correspond aux **anciens organigrammes** (algorithme graphique).
  - ⇒ Il permet de représenter toutes les activités algorithmiques de programmation impérative classique (de l'affectation au programme) avec des tests et des boucles.
  - ⇒ Le diagramme d'activités est une variante du diagramme d'états-transitions qui met en avant les activités (point de vue dynamique) et les bifurcations plutôt que les états (point de vue statique) et les transitions d'un état à un autre.

Anciens organigrammes :



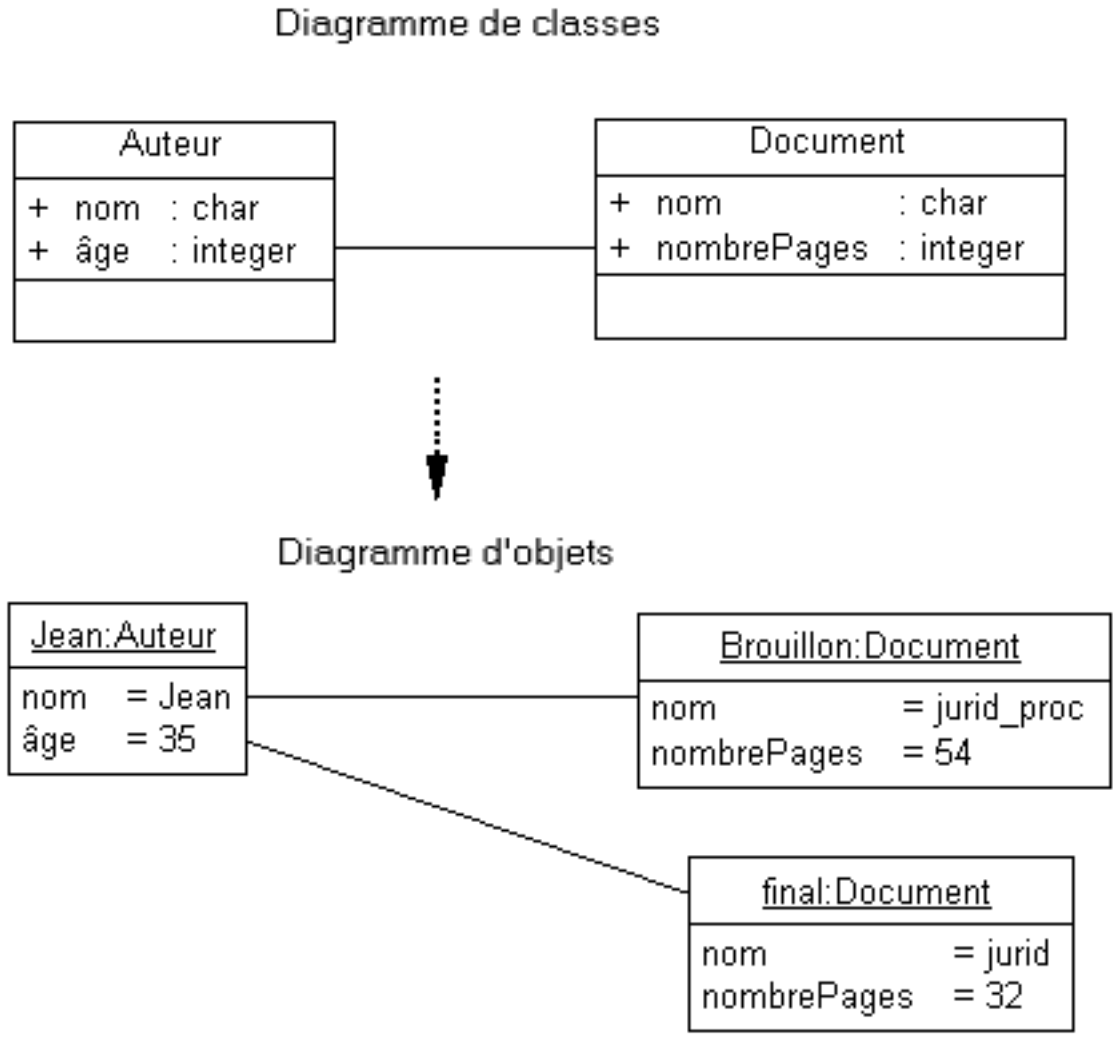
Usages avec les Use Cases



() 3 autres diagrammes importants d'UML

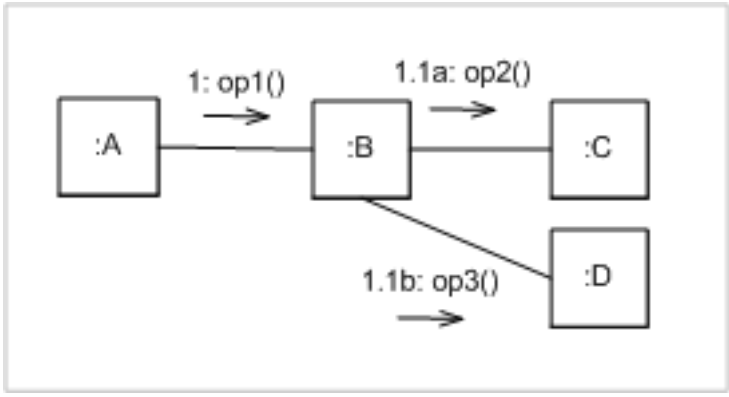
5 : Diagramme d'objets (cf. classes)

Il représente **un état du système** des objets à un instant donné.
On y représente les objets, qui sont des instances des classes, avec les valeurs effectives pertinentes des attributs et les liens pertinents entre les objets.
C'est un diagramme technique qui présente une instance du diagramme de classes à un instant donné de la vie du système.



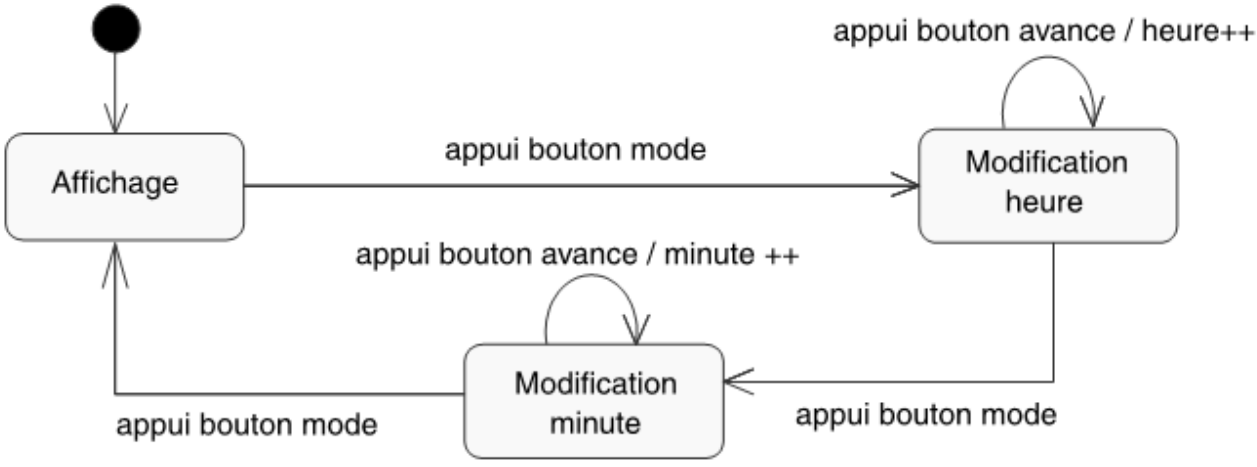
6 : Diagramme de communication (cf. séquence)

Tout comme les diagrammes de séquence, il montre les interactions (les appels de méthodes) entre les objets (les instances des classes) : **c'est un diagramme technique.**
A la différence d'un diagramme de séquence, la séquence temporelle des interactions n'est pas visualisée graphiquement. Par contre il montre les relations structurelles entre les objets.



7 : Diagramme d'états-transition (cf. activité)

Il montre les évolutions d'un état particulier du système. Il est centré sur la présentation des différents états et des transitions entre ces états.



() Les 7 diagrammes d'UML secondaires

8 : Diagramme de paquetages

Les packages permettent de regrouper les classes ou n'importe quel autre élément de modélisation.

9 : Diagramme de composants

Il représente les composants physiques d'une application (fichiers, bibliothèques, etc.).

10 : Diagramme de déploiement

Il représente le déploiement des composants sur les dispositifs matériels.

11 : Diagramme de structures composites

Il représente la structure interne d'une classe : c'est très technique.

12 : Diagramme de profils

Un profil se représente comme un package, avec un stéréotype <<profil>> en plus.

13 : Diagramme d'interactions global

C'est un cas particulier de diagramme d'activités.

14 : Diagramme de temps

Les diagrammes de temps présentent les comportements et interactions de 2 éléments UML en fonction du temps.

4. UML et le cycle en V

Les différents diagrammes selon l'étape de la conception

<table>
<tr><th></th><th>Point de vue</th><th>Diagramme UML</th><th></th></tr>
<tr><td rowspan="3">ANALYSE FONCTIONNELLE</td><td rowspan="2">Statique – non objet
Analyse générale</td><td>Cas d'utilisation</td><td rowspan="4"></td></tr>
<tr><td rowspan="2">Dynamique – non objet
Analyse détaillée</td><td>Séquence</td></tr>
<tr><td>Activités</td></tr>
<tr><td>Etats-transitions</td></tr>
<tr><td rowspan="2">ANALYSE DES DONNEES</td><td>Statique – non objet</td><td>MEA
équivalent Classes</td><td rowspan="5">D O N N E E S</td></tr>
<tr><td>Statique – objet</td><td>Classes-métier</td></tr>
<tr><td rowspan="4">ANALYSE TECHNIQUE</td><td rowspan="2">Statique – objet</td><td>Classes</td></tr>
<tr><td>Objets</td></tr>
<tr><td rowspan="2">Dynamique - objet</td><td>Séquence</td></tr>
<tr><td>Communication</td></tr>
<tr><td rowspan="2">Dynamique
Plutôt non objet</td><td>Etats-transitions</td></tr>
<tr><td>Activités</td></tr>
</table>

Synthèse

Type d'analyse	Types de diagramme
Fonctionnelle	**Cas d'utilisation** **Séquence système** **Activités** États-transitions
Classes métier Classes de POO	**Classes** Séquence objet Communication Objets
Algorithmique classique	Activités États-transitions

- Les 4 diagrammes en bleu de ce tableau (cas d'utilisation, séquence système, activités et classes) sont les diagrammes de base de la conception de SI.

( ) Autre point de vue

Etapes	Point de vue		Diagramme	Paternité	Nb de diag.
Analyse fonctionnelle	Tout	Dynamique	Flux (contexte)	MERISE	1
	Tout	Statique	Cas d'utilisation	UML	1
	Parties	Dynamique	Séquence système	UML	* / UC
	Parties	Dynamique	Activité	UML	1 / UC
Analyse des données	Tout	Statique	Classes	UML – MERISE	1
Analyse technique	Tout	Statique	Classes	UML	1
	Parties	Statique	Objets	UML	*
	Parties	Dynamique	Séquence – Communication	UML	*
	Parties	Dynamique	Etats-transitions	UML	*
	Parties	Dynamique	Activités	UML	*

( ) Autre point de vue

	Diagramme	**Cf.**	**Fonctionnel**	**Données**	**Technique**	**Architecture**	**Objet**
1	Cas d'utilisation		**Oui**			Oui	
2	Séquence	Collab.	**Oui (scénarios)**		**Oui (séquence objet)**	Oui (séquence système)	Oui
3	Activités	Etats-trans.	**Oui (flots)**		(peu)		
4	Classes	Objets		**Oui**	**Oui**		Oui
5	Objets	Classes			**Oui**		Oui
6	Communication	Séquence			**Oui**		Oui
7	Etats-transitions	Activités			**Oui**		

5. Bibliographie

Référence du cours

- **UML 2, de l'apprentissage à la pratique – Cours et exercices, Laurent Audibert, 2009 (UML 2)**
  Ouvrage proche du cours proposé. Beaucoup d'exercices élémentaires. Application en java.
  Accès internet :
  http://laurent-audibert.developpez.com/Cours-UML/

Ce cours

- **http://bliaudet.free.fr/rubrique.php3?id_rubrique=29**

() Théorie UML

- **Modélisation objet avec UML, Pierre-Alain Muller et Nathalie Gaertner, Eyrolles, 2000, 2ème édition (MO.UML)**
  La référence française sur UML.
- **Introduction à UML, Tom Penders, OEM 2002.**
- **Mémento UML, Pascal Roques, Eyrolles, 2005 (UML 2)**
  La synthèse la plus concise !

() Pratique

- **Penser objet avec UML et Java, Michel Lai, Dunod, 2000.**
  Apprentissage de la conception objet et de l'UML associé avec des exemples concrets en Java. Très bonne introduction concrète.
- **UML 2 par la pratique, Pascal Roques, Eyrolles 2005.**
  Apprentissage d'UML par la pratique en suivant la méthode RUP.
- **UML2 en action, Pascal Roques et Franck Vallée, Eyrolles 2004.**
  Le même que le précédent en plus développé mais aussi plus abstrait (et plus cher !).
- **UML2 et les design pattern, Craig Larman, Pearson Education 2005.**
- **Modélisation UML avec Rational Rose 2000, Terry Quatrani, Eyrolles 2000.**
  UML et Rose.
- **UML 2, Benoît Charroux, Aomar Osmani, Yann Thierry-Mieg, Pearson Education, 2005**
  Quelques exercices intéressants.

() Internet

- **Il y en a partout !**