UML – Unified Modeling Language 1/4 : introduction

Yannick Prié
Département Informatique – Faculté des Sciences et Technologies
Université Claude Bernard Lyon 1
2009-2010

UML ?

Sondage sept 09

- Analyse contexte / situation pour préparer la POO
- méthode de modélisation des donnéees. Méthode où l'on va représenter les actions, les utilisateurs, les besoins, les données et définir les interactions entre.
- ensemble de méthodes ayant pour objectif la modélisation de SI.
- permet de modéliser un problème. Il est ainsi plus facile, aprèsmodélisation, de rentrer dans la partie programmation d'un projet.
- sert à délimiter, à comprendre un projet informatique. Il est le pont entre l'énonciation des besoins utilisateurs et le SI à créer.
- L destiné à modéliser des SI, des événements, des traitements, des processus entre acteurs
- système de modélisation de données. Sert à représenter un SI et à rendre ce SI compréhensible par tous.
- permet de représenter un système informatique en fonction des besoins d'un client. Langage conçu pour être compris par tous !
- façon de représenter les données sous forme de schémas de différents types afin de mieux comprendre le problème.
- méthode de modélisation, permet de modéliser un problème. Utile pour l'analyse du problème. Permet de trouver une solution adaptée.
- outil de modélisation des concepts de l'OO sous forme de graphe, montre les classes et leurs relations

UML ?

Sondage sept 09

- méthode d'analyse qui comprend différents schémas. Permet de spécifier les objectifs, les processus, les acteurs, du projet
- Langage pour modéliser un modèle OO destiné à être codé (classes + relations)
- Méthode de modélisation qui permet de définir les différents cas d'utilisation, les différentes classes et leurs interactions
- permet de modéliser de plusieurs manières un SI et les interactions qu'il engendre. Propose des diagrammes qui permettent d'avoir une vision schématique de ce que devra faire le SI et son fonctionnement
- décrit un SI industriel, formé de différentes parties qui décrivent le SI sous des angles différents
- langage de modélisation, permet de modéliser un ensemble de classes ou de décrire le fonctionnement global d'un cas précis. Etape indispensable à la POO
- permet de modéliser les besoins de manière claire et précise. On utilise pour cela les diagrammes de classes, seq.
- représentation graphique du futur programme qui va être développé, utile pour le travail en équipe, permet aussi d'expliquer un programme à une personne n'ayant aucune connaissance en informatique
- permet de schématiser le fonctionnement voulu pour notre système en utilisant une notation standardisée et compréhensible pour un informaticien

UML ?

Sondage sept 09

- permet de modéliser un projet en partant des besoins utilisateur jusqu'à la réalisation du projet
- notation qui sert à modéliser une situation
- normalisation qui permet de décrire la structure des classes et leur implémentation ainsi que les interactions ... permet de réaliser sur le papier des systèmes compréhensibles par tous
- méthode d'analyse objet permettant de modéliser, de représenter par le biais de diagrammes (acteurs, CU, seq...) ce qui doit être couvert par le périmètre de l'application
- langage de modélisation unifié pour représenter un SI à travers les objets qui le composent
- langage souvent vu comme une méthode. comporte 14 diagrammes.
- méthode de représentation de la POO qui permet de visualiser la structure de classe et les liens qui les unissent, permet donc de représenter le SI
- permet de modéliser, par le biais de différents diagrammes les besoins des utilisateurs. Permet aussi de modéliser de manière concise le SI qui sera nécessaire pour répondre aux besoins des utilisateurs (diag classes)
- langage de modélisation, permet de décrire la structure, le fonctionnement et l'organisation d'un programme de manière compréhensible
- représentation du projet. Sert à visualiser l'ensemble du projet de façon schématique.
- méthode pour schématiser les besoins sous forme graphique

UML ?

Sondage sept 09

- le modèle UML permet permet la compréhension du système par l'ensemble des acteurs / clients ou développeurs
- ensemble de méthodes et de diagrammes qui permettent de modéliser de manière précise un problème informatique
- méthode de modélisation qui sert à représenter un SI en en décrivant toutes les fonctionnalités à l'aide de plusieurs diagrammes
- méthodologie permettant de concevoir un outil
- bonne idée à la base, mais normes officielles tellement strictes et complexes que peu de monde les connaît réellement et qu'il ne doit pas y avoir deux personnes qui les appliquent de la même façon
- outil de modélisation pour la POO. Normes strictes. On peut facilement comprendre les classes à créer et les interactions entre les classes.
- méthode permettant de représenter une application sous une forme simple et compréhensible par tous. Elle permet de créer un base pour la modélisation du prolème.
- langage permettant de modéliser différentes choses comme des CU, des scénarii, des classes. Permet d'avoir une vision globale sur un projet.
- méthode d'analyse qui permet de schématiser les besoins de l'utilisateur et le fonctionnement d'un SI pour en tirer la structure des classes objet à mettre en place, ainsi que les interactions entre celles-ci. UML fait donc le lien entre la demande initiale et le développement à proprement parler.

Objectifs de ce cours

- Pourquoi UML ?
- Petite histoire d'UML
- Principes généraux
- Quelques diagrammes

UML en un transparent

- UML = Unified Modelling Language
- Unification
  - de nombreux langages de modélisation graphique OO des années 1990,
  - de diagrammes et de principes de modélisation à succès
- Défini par l'OMG (Object Management Group)
- Un métamodèle et divers diagrammes dont
  - diagrammes de classes
  - diagrammes de séquences
- Utilisé pour concevoir, comprendre, échanger, à propos de systèmes d'information
- UML n'est pas une méthode

Plan

- **Histoire d'UML**
- Diagrammes et modélisation
- Généralités sur les diagrammes

Un foisonnement de méthodes

- Fin 80 / début 90
  - orientation de plus en plus marquée vers l'objet
- Conséquence naturelle, mise en place de méthodes
  - OOD : Object Oriented Design (Booch, 1991)
  - HOOD : Hierarchical Object Oriented Design (Delatte et al., 1993)
  - OOA : Object Oriented Analysis (Schlaer, Mellor, 1992)
  - OOA/OOD : (Coad, Yourdon, 1991)
  - OMT : Object Modeling Technique (Rumbaugh, 1991)
  - OOSE : Object Oriented Software Engineering (Jacobson, 1992)
  - OOM : Object Oriented Merise (Bouzeghoub, Rochfeld, 1993)
  - Fusion (Coleman et al., 1994)
- Bilan
  - de nombreuses méthodes (>50)
  - ayant des avantages et des inconvénients
  - des concepts assez proches, des notations différentes

Vers une unification

- 1994
  - tentative de normalisation de l'OMG, sans effet
  - Rumbaugh (OMT) rejoint Booch (OOD) chez Rational Software
    - objectif : créer une méthode en commun (méthode unifiée)
- 1995
  - présentation de la version 0.8 de la méthode
  - arrivée de Jacobson (OOSE) chez Rational
- 1996
  - implication de l'OMG (sous pression des industriels pour favoriser l'interopérabilité des modèles)
  - langage unifié UML 0.9 (Unified Modeling Language),
- 1997
  - UML 1.0 sort chez Rational
  - UML 1.1 adopté par l'OMG comme standard officiel

Evolutions d'UML

- 1997-2003
  - adoption par les entreprises
  - UML 1.1 à UML1.5 : modifications/améliorations
- 2005
  - UML 2.0
  - quelques nouveaux diagrammes
  - changements importants au niveau du méta-modèle, pour permettre d'utiliser UML pour la programmation
- 2007
  - UML 2.1.1
  - À suivre

Unified Modeling Language

- Combinaison de principes à succès
  - modélisation de données (E/A), de l'activité, objet, en composants…
- Objectif
  - visualiser / spécifier / construire / documenter les artefacts de la conception d'une application
- La norme elle-même
  - méta-modèle et familles de diagrammes
- Utilisation
  - pas de méthode préconisée
  - pas de spécification technologique

Plan

- Histoire d'UML
- **Diagrammes et modélisation**
- Généralités sur les diagrammes

Modèles, vue et diagrammes UML

- Modèle
  - abstraction d'un système composée d'un ensemble d'éléments de modèle
  - ce qui est construit par et ce qui est perçu au travers des diagrammes (par le concepteur, le lecteur)
  - conforme au méta-modèle UML
- Vue
  - projection d'un modèle suivant une perspective qui omet les éléments non pertinents pour cette perspective. Elle se manifeste dans des diagrammes
  - ex. : vue statique, vue fonctionnelle…
- Diagramme
  - présentation graphique d'éléments de visualisation représentant des éléments de modèle (graphe)
  - ex. : diagramme de classes, de séquences…

(Larman 2005)

Exemples de diagrammes

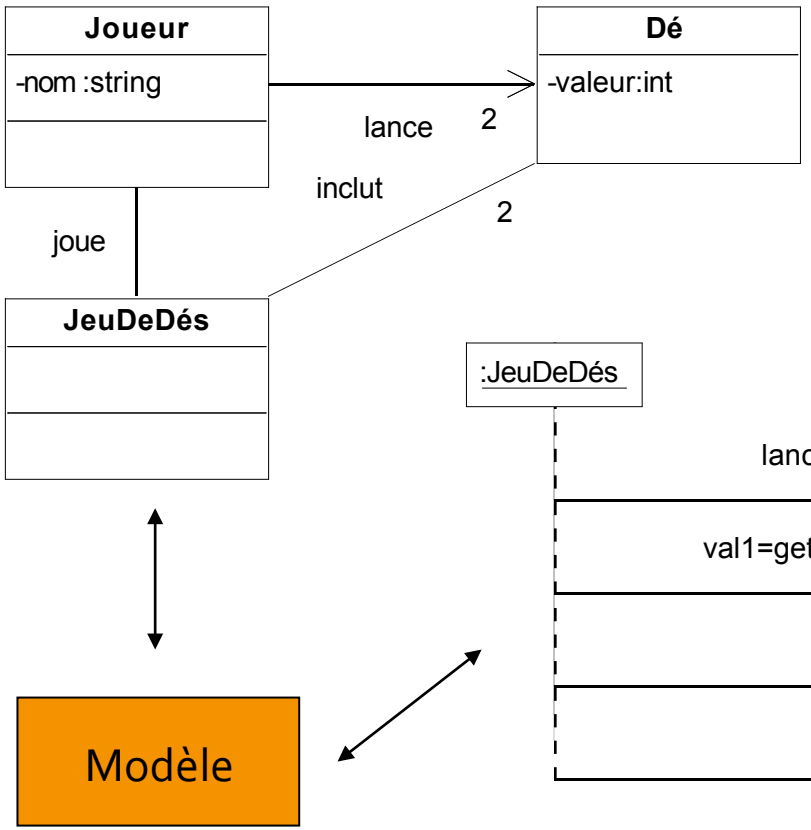


diagramme de classe, vue statique

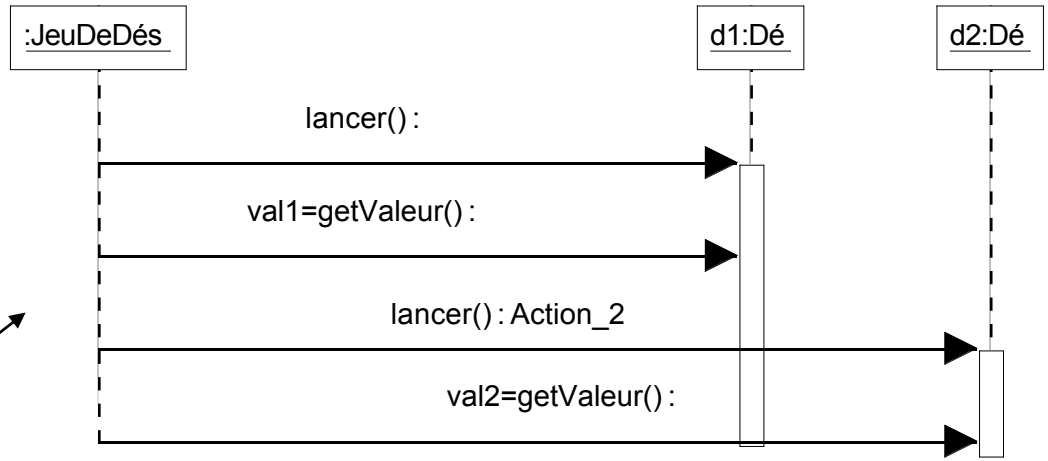
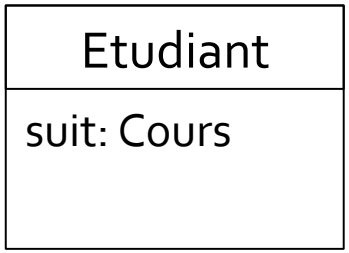


diagramme de séquences, vue dynamique

Eléments de modélisation, de visualisation et sémantique

- Les éléments de visualisation sont utilisés dans les diagrammes visuels, les éléments de modélisation forment le modèle lui-même
- La syntaxe de UML dit comment composer les éléments. La sémantique du modèle dit ce qu'ils signifient en termes objets.
- Par exemple
  - dans un diagramme « dc1 » on a mis les éléments de visualisation
    - un conteneur composé de trois parties
    - le texte « Etudiant » dans la partie du haut,
    - le texte « suit : Cours » dans la partie médiane
  - ce qui signifie
    - qu'on a les éléments de modélisation suivant
      - classes « Etudiants », « Cours »
      - l'attribut « suit » associé à la classe « Cours »

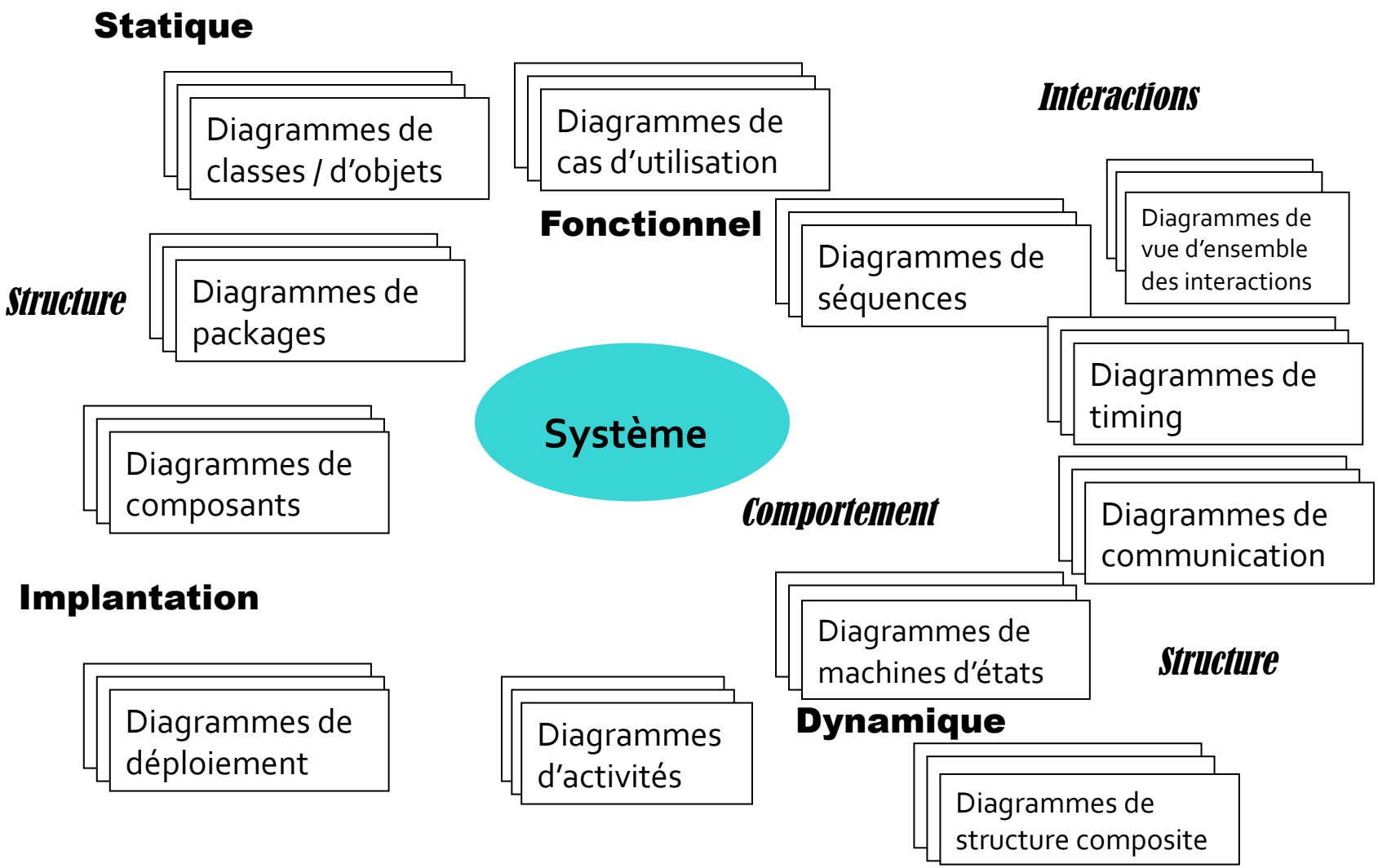


- La sémantique du langage spécifie ce que signifie un modèle objet
  - qu'une classe peut avoir des instances
  - qu'une instance a des attributs
  - qu'un instance peut être valeur d'attribut
  - etc.

Objectifs d'UML (1997)

- *Montrer les limites d'un système* et ses fonctions principales (pour les utilisateurs) à l'aide des cas d'utilisation et des acteurs
- *Illustrer* les réalisations de CU à l'aide de diagrammes d'interaction
- *Modéliser la structure statique* d'un système à l'aide de diagrammes de classes, associations, contraintes
- *Modéliser la dynamique*, le comportement des objets à l'aide de diagrammes de machines d'états
- *Révéler l'implantation physique* de l'architecture avec des diagrammes de composants et de déploiement
- Possibilité d'étendre les fonctionnalités du langage avec des stéréotypes
- Un langage utilisable par l'homme et la machine : permettre la génération automatique de code, et la rétro-ingénierie

Panorama des diagrammes



(Fowler/Mellor)

3 modes d'utilisation d'UML

- Esquisse
  - conception / communication
  - incomplétude
- Plan
  - exhaustivité
  - outils bidirectionnels
- Programmation
  - model Driven Architecture / UML exécutable
  - implantation automatique
  - réaliste ?

Focus sur les diagrammes

Focus sur le méta-modèle

Conception et UML

- Différentes façons de voir UML : différentes façons de penser
  - la conception
  - l'objectif et l'efficacité d'un processus de génie logiciel
- donc
  - essayer de comprendre le point de vue de l'auteur pour chaque publication / site sur UML

- UML n'est pas une méthode…
- … mais des principes de conception orientée objet sont sous-jacents
  - aux diagrammes
  - aux façons de les présenter
- donc
  - difficile de présenter uniquement les diagrammes
  - on parlera aussi de méthode, de bonnes pratiques

Généralités sur les méthodes OO

- Grandes caractéristiques
  - itératives (vs cascade)
    - analyse et conception tout au long du projet, pas seulement au début
  - centrées sur les cas d'utilisation
    - besoins réels
  - centrées sur l'architecture
- Découpage d'un projet en activités
  - besoins : comprendre dans quoi s'insère le système et ce qu'il doit faire
  - analyse : fonctionnement du système à hau niveau
  - conception : fonctionnement logiciel
  - réalisation : codage
  - tests, déploiement…

À propos de ces cours

- Présentation de UML 2, quelques points de UML 1
  - faites attention à la syntaxe quand vous rencontrez un diagramme
- Présentation = synthèse de nombreuses lectures
  - mixe syntaxe et utilisation
  - synthèse personnelle des bonnes pratiques présentées
- Présentation d'UML non exhaustive
  - ce cours contient beaucoup de choses utiles
  - pour plus de précisions : livres de référence
  - pour la description exacte (syntaxe et sémantique) : http://www.omg.org/uml
- UML et le web
  - beaucoup de sites web parlent d'UML
  - on trouve du bon et du moins bon

UML et la règle

- Deux types de règles pour l'utilisation d'UML
  - normatives
    - comment il faut faire, comité d'experts : normes
  - descriptives
    - comment les gens font, usages, modes : conventions dans l'utilisation
    - peuvent être en contradiction avec la norme (surtout pour UML2)
- Règles
  - utiliser le sous-ensemble d'UML qui vous convient
  - droit de supprimer n'importe quel élément d'un diagramme
  - droit d'utiliser n'importe quel élément d'un diagramme dans un autre
    - ce qui compte pour les auteurs d'UML, c'est le méta-modèle, pas les diagrammes
  - liberté de dessiner ce que l'on veut
    - surtout en mode esquisse, sur papier ou au tableau

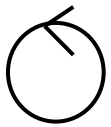
Plan

- Histoire d'UML
- Diagrammes et modélisation
- **Généralités sur les diagrammes UML**

Mots-clés (classifiers)

- Mots-clés
  - pour regrouper en familles des éléments similaires d'un modèle pour ne pas multiplier les symboles différents dans les diagrammes
- Ornements textuels
  - associés à des éléments du modèle
  - certains mots-clé sont prédéfinis par UML
- Notation
  - « mot-clé »
  - ex. « abstract » « interface » « à vérifier »

Stéréotypes

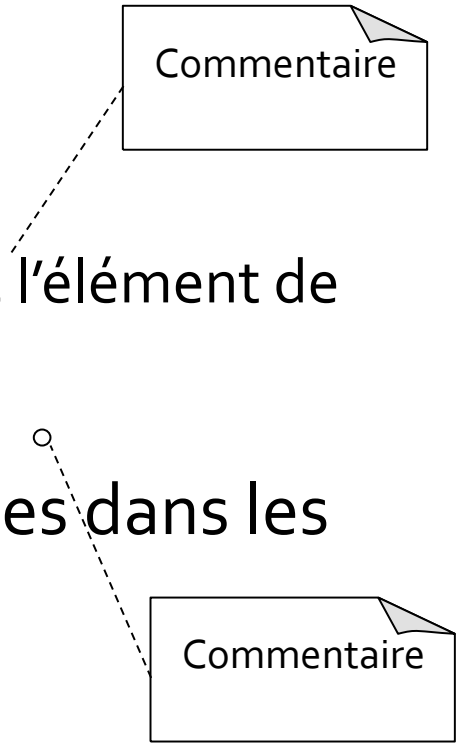
- Permet d'étendre le vocabulaire UML en dérivant des nouveaux éléments à partir d'éléments existants
  - par ajout de valeur étiquetées (`nom=valeur`)
- Notation : « stéréotype »
- Certains sont prédéfinis par UML
  - Eg. « constructor » « getter » « entity » « boundary » « control »
- Possibilité d'associer une icône
  - forme visuelle déterminée
  - ex. : pour « control » 

Contraintes

- Relation sémantique quelconque
  - concernant un ou plusieurs éléments du modèle
  - définissant des propositions devant être maintenues à Vrai pour garantir la validité du système modélisé
- Notation : {contrainte}
  - contenu formel ou informel
  - à côté des éléments concernés
  - ex. {frozen}, {jamais détruit !}, {x - y < 10}
- Certaines sont prédéfinies
  - ex. xor, ordered
- D'autres créées par l'utilisateur
  - langue, pseudo-code, OCL…

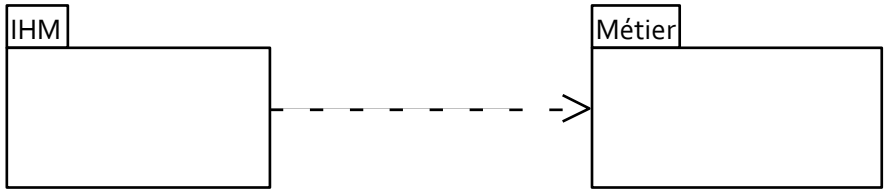
Commentaires

- Commentaire
  - annotation quelconque associée à un élément du modèle
  - pas de sémantique pour le modèle
- Notation : note
  - rectangle avec coin replié, lien pointillé avec l'élément de visualisation concerné
  - cercle en bout de ligne : plus précis
- Il existe des mots-clé prédéfinis utilisables dans les commentaires
  - ex. « besoin », « responsabilité »



Dépendances

- Relation sémantique faible
  - relation d'utilisation unidirectionnelle entre deux éléments
- Notation
  - flèche pointillée de l'élément source vers l'élément cible, éventuellement stéréotype / mot-clé pour préciser le type de dépendance



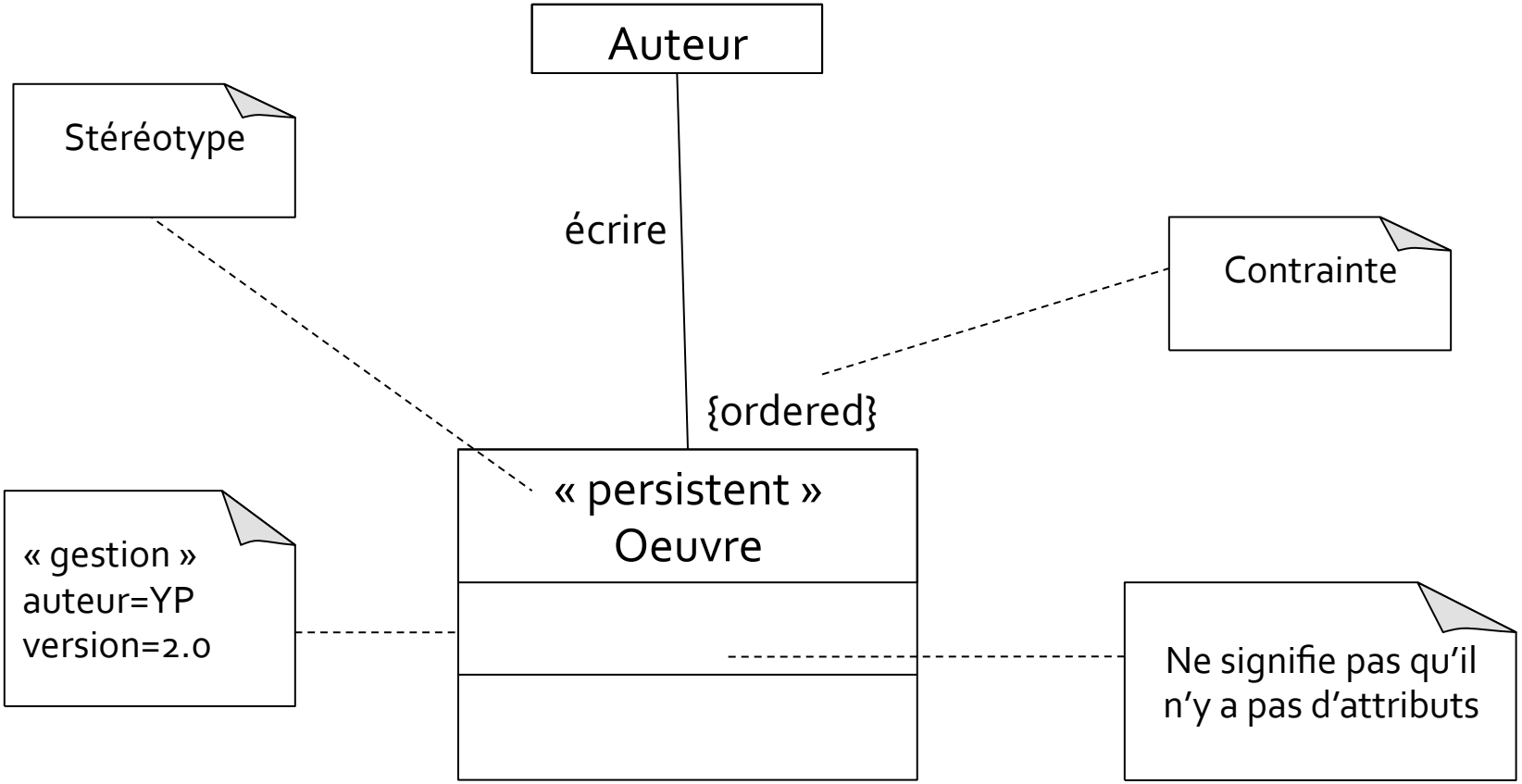
« le paquetage IHM est dépendant du paquetage Métier »

Conséquences :
- Toute modification dans le paquetage Métier *peut* avoir des conséquences sur le paquetage IHM
- Une modification dans le paquetage IHM n'a *au vu de cette dépendance* pas de conséquence sur le paquetage métier

Diagrammes

- Diagramme = association d' éléments de dessin
  - formes nœuds
  - relation de graphe
  - formes conteneurs
  - texte
- Sémantique graphique importante
  - graphe
  - contenant / contenu
  - Proximité
- Liberté pour le reste (positions…)
- N'importe quelle information peut être supprimée dans un diagramme
  - attention : pas de déduction due à l'absence d'un élément

Exemple général



A suivre

- UML 2/4 : diagrammes statiques
- UML 3/4 : diagrammes dynamiques et d'interactions
- UML 4/4 : concepts avancés