

COO

La Notation UML

Cours d'Isabelle Mirbel

retouches mineures par Ph. Collet (septembre 2012)

L3-MIAGE Année 2012-13

ORGANISATION

□ Plan général

- ✓ Analyse et conception en UML
- ✓ Spécification (langage naturel, assertions et OCL)

□ Evaluation (contrôle continu)

- ✓ 1 note de contrôle en TD (25 %)
 - ✓ interrogation sur papier de 10 minutes en TD régulièrement, 3 à 4 fois sur l'ensemble du module
- ✓ 1 projet à rendre en équipe (35 %)
- ✓ 1 contrôle sur table en janvier (40 %) :
 - ✓ 2h seuls documents autorisés : mémentos UML et OCL (fournis lors du contrôle)

↙ http://miageprojet2.unice.fr/User:PhilippeCollet/Conception_orientée_objets



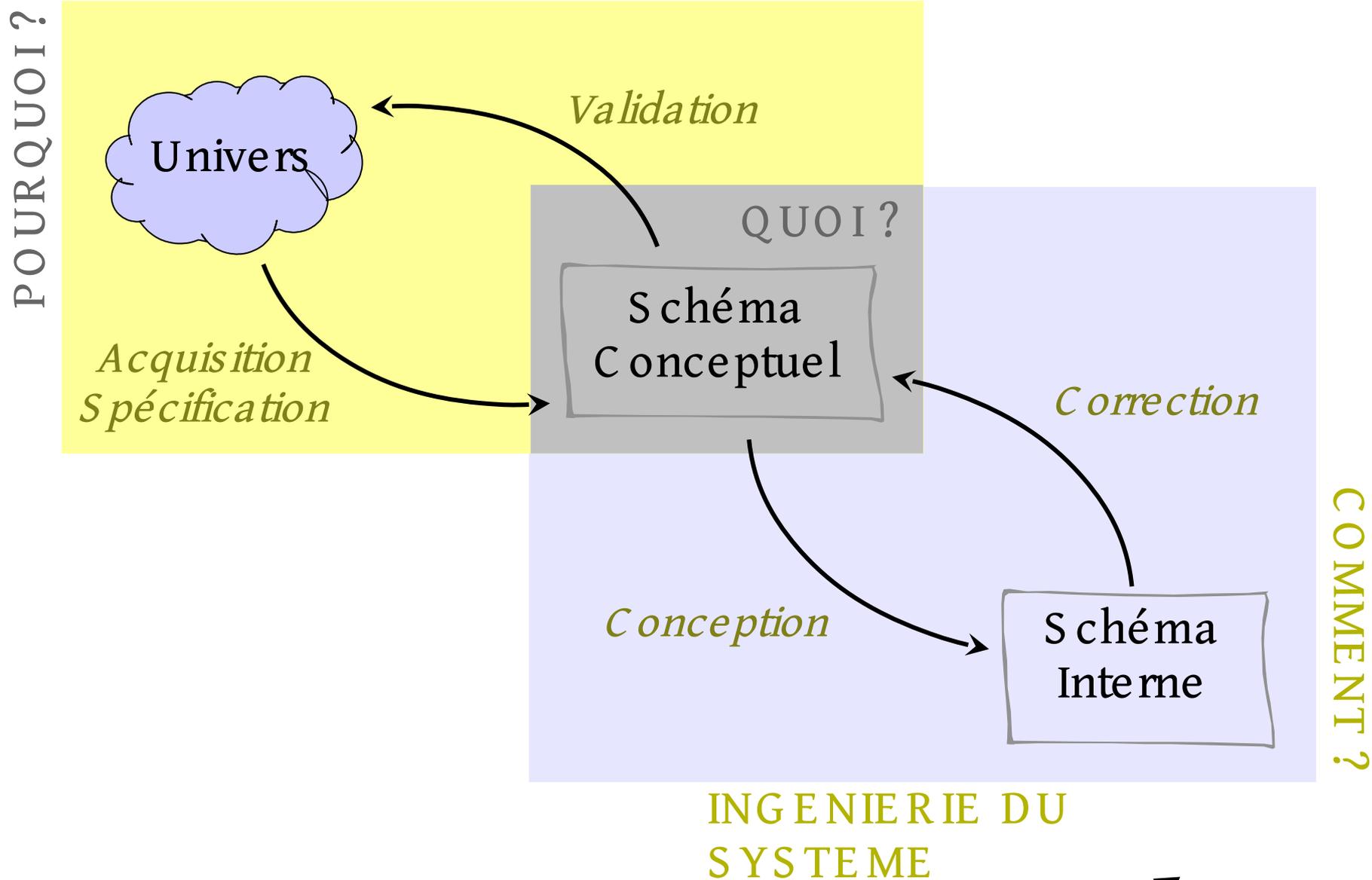
P L A N

- ▶ ■ Introduction
- Les diagrammes structurels
- Les diagrammes comportementaux
- Le diagramme de paquetage
- Conclusion

INTRODUCTION

□ L'ingénierie des besoins

INGENIERIE DES BESOINS



INTRODUCTION

□ MODELE

- ✓ Notation pour représenter les objets du monde réel
- ✓ Formalisme défini pour la spécification d'applications informatiques

□ METHODE

- ✓ Règles d'utilisation de la notation, guide, démarche cohérente
- ✓ Collection organisée de concepts & de prescriptions
- ✓ Supportée par des outils informatiques
- ✓ Pour développer de façon efficace & systématique



INTRODUCTION

- DES MODELES POUR QUOI FAIRE ?
 - ✓ Améliorer la communication
 - ✓ Partager l'information
 - ✓ Lever les ambiguïtés
 - ✓ Penser avant de coder (complexité grandissante des applications)
 - ✓ Proposer une vue abstraite
 - ✓ Anticiper des problèmes fonctionnels & techniques
 - ✓ Permettre une meilleure planification



INTRODUCTION

- DES MODELES COMMENT ?
 - ✓ Choix de la façon de modéliser influence
 - ✓ Comment le problème est appréhendé
 - ✓ Comment la solution est trouvée

 - ✓ Chaque modèle doit être utilisé avec
 - ✓ Des niveaux de précision différents
 - ✓ Des niveaux d'abstraction différents

 - ✓ Utilisation de plusieurs modèles nécessaire



▣ MODELISER, POURQUOI ? (suite)

- ✓ Réfléchir au système à développer
- ✓ Capturer les décisions de conception
- ✓ Produire un travail éventuellement réutilisable
- ✓ Permettre d'organiser l'information dans les systèmes de grande ampleur

- ✓ Explorer plusieurs solutions à un problème
- ✓ Mener à bien le développement de systèmes complexes
- ✓ Support tout au long du processus de développement
- ✓ Extraction des spécifications essentielles
- ✓ Spécification complète du système à développer et exemples de situations types possibles



INTRODUCTION

□ GENESE

- ✓ Les méthodes de programmation
 - ✓ OOD (G. Booch - 1991)
 - ✓ HOOD (B. Delatte, M. Heitz, J.F. Muller – 1993)

- ✓ Les extensions de méthodes spécifiques de développement d'applications temps réel
 - ✓ OOA (S. Shlaer, S.Mellor - 1988/1992)
 - ✓ OOSE (I. Jacobson, M. Christerson, P. Jonson, G. Overgaard – 1992)

- ✓ Les extensions de méthodes générales de conception de systèmes d'information
 - ✓ OOAD (T. Coad & E. Yourdon - 1991)
 - ✓ OMT (J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, W. Lorensen – 1991)



INTRODUCTION

1995 1996 1997 1998 1999 2000 2001 2003 2004 2005 2006 2007 2008 2009 2010

Rational

UML consortium

OMG

U
M
L
0
8

U
M
L
0
9

U
M
L
1
0

U
M
L
1
1

U
M
L
1
2

U
M
L
1
3

U
M
L
1
4

U
M
L
1
5

U
M
L
2
0

U
M
L
2
1
1

U
M
L
2
2

U
M
L
2
3

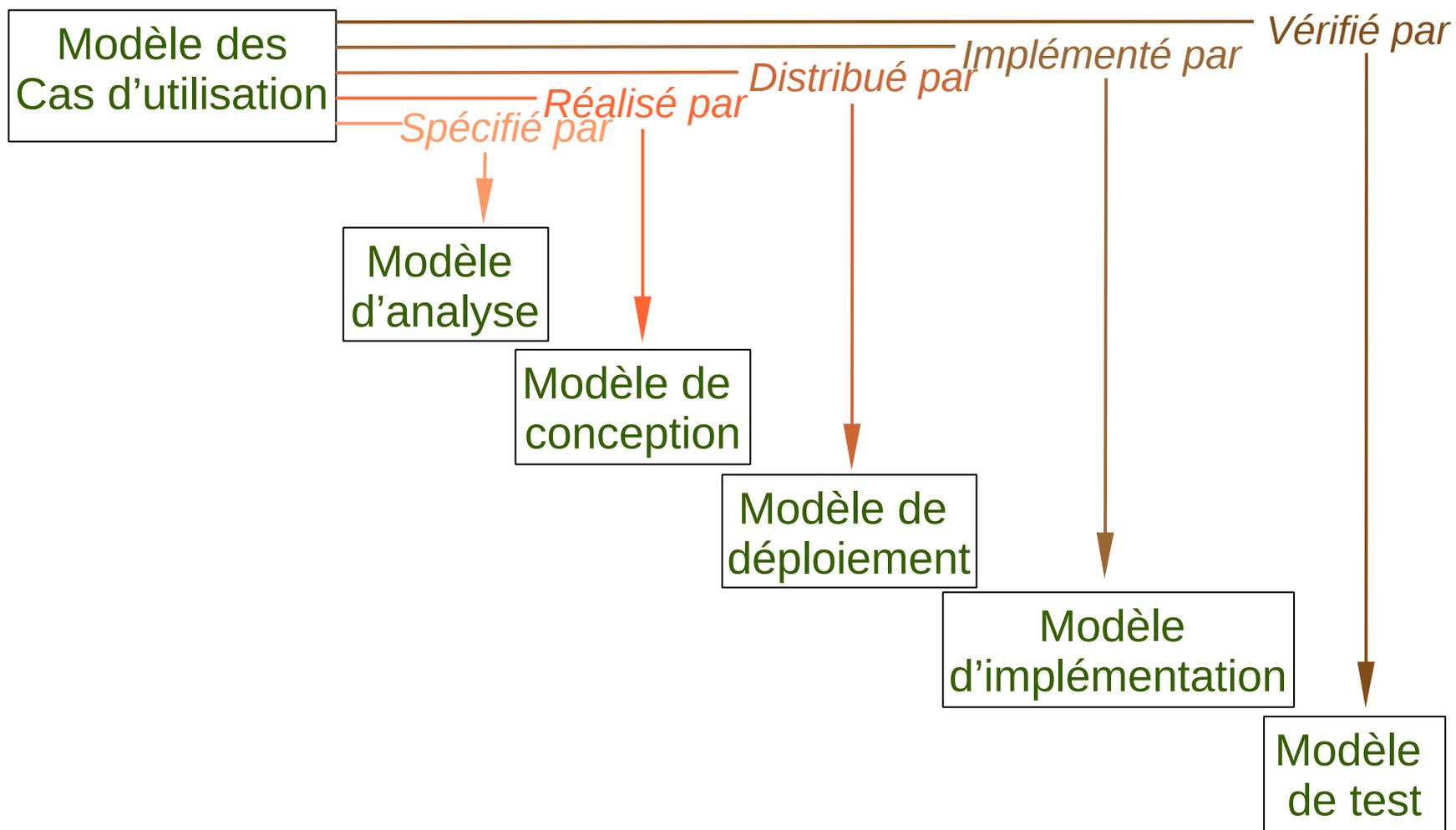


INTRODUCTION

- ✓ UML est une notation (et pas un modèle)
 - ➔ UML ne propose pas de démarche
- ✓ Auteurs : Booch, Jacobson & Rumbaugh
- ✓ Version officielle 2.3, en cours de validation 2.4
- ✓ Méthodologie recommandée dirigée par les cas d'utilisation
 - ✓ Analyse systématique des besoins des utilisateurs
 - ✓ Guide tout au long du processus de développement



INTRODUCTION



▣ LES DIFFERENTS DIAGRAMMES

✓ Diagrammes structurels

- ✓ Diagramme de **classe**
- ✓ Diagramme d'**instance** ou d'**objet**
- ✓ Diagramme d'**architecture** ou de **structure composite**
- ✓ Diagramme de **composant**
- ✓ Diagramme de **déploiement**
- ✓ Diagramme de **cas d'utilisation**



▣ LES DIFFERENTS DIAGRAMMES (suite)

✓ Diagrammes comportementaux

✓ Diagramme de machine d'état (*ex diagramme état - transition*)

✓ Diagramme d'activité

✓ Diagramme d'interaction

✓ Diagramme de séquence

✓ Diagramme de communication (*ex diagramme de collaboration*)

✓ Diagramme global d'interaction

✓ Diagramme de timing

✓ Diagramme de paquetage



INTRODUCTION

❑ CHANGEMENTS 1.x / 2.x

- ✓ Éléments des **diagrammes de séquence** pour permettre une meilleure modularité (notamment / cas d'utilisation)

- ✓ Éléments des **diagrammes d'activité** (plus conformes aux notations répandues dans le domaine de la modélisation métier)

- ✓ Repositionnement du concept de **composants**
 - ✓ Diagramme de composant
 - ✓ Diagramme d'architecture

- ✓ Nouveaux diagrammes
 - ✓ Diagramme de temps
 - ✓ Diagramme global d'interaction

- ✓ Intégration des **profils**

- ✓ Unification / **MOF**
- ✓ Restructuration du **méta-modèle**



INTRODUCTION

- ✓ Méta – méta modèle
 - ➔ *Méta-classe, méta-attribut, ...*
- ✓ Méta modèle
 - ➔ *Classe, attribut, ...*
- ✓ Modèles
 - ➔ *Client, fournisseur, facture, ...;*
- ✓ Objets
 - ➔ *Dupont, Martin, 22342. ...*



INTRODUCTION

✓ Les tags

UseCase

Class

Object

Activity

Component
Deployment

Sd

Intover

Comm

Timing

State

Package



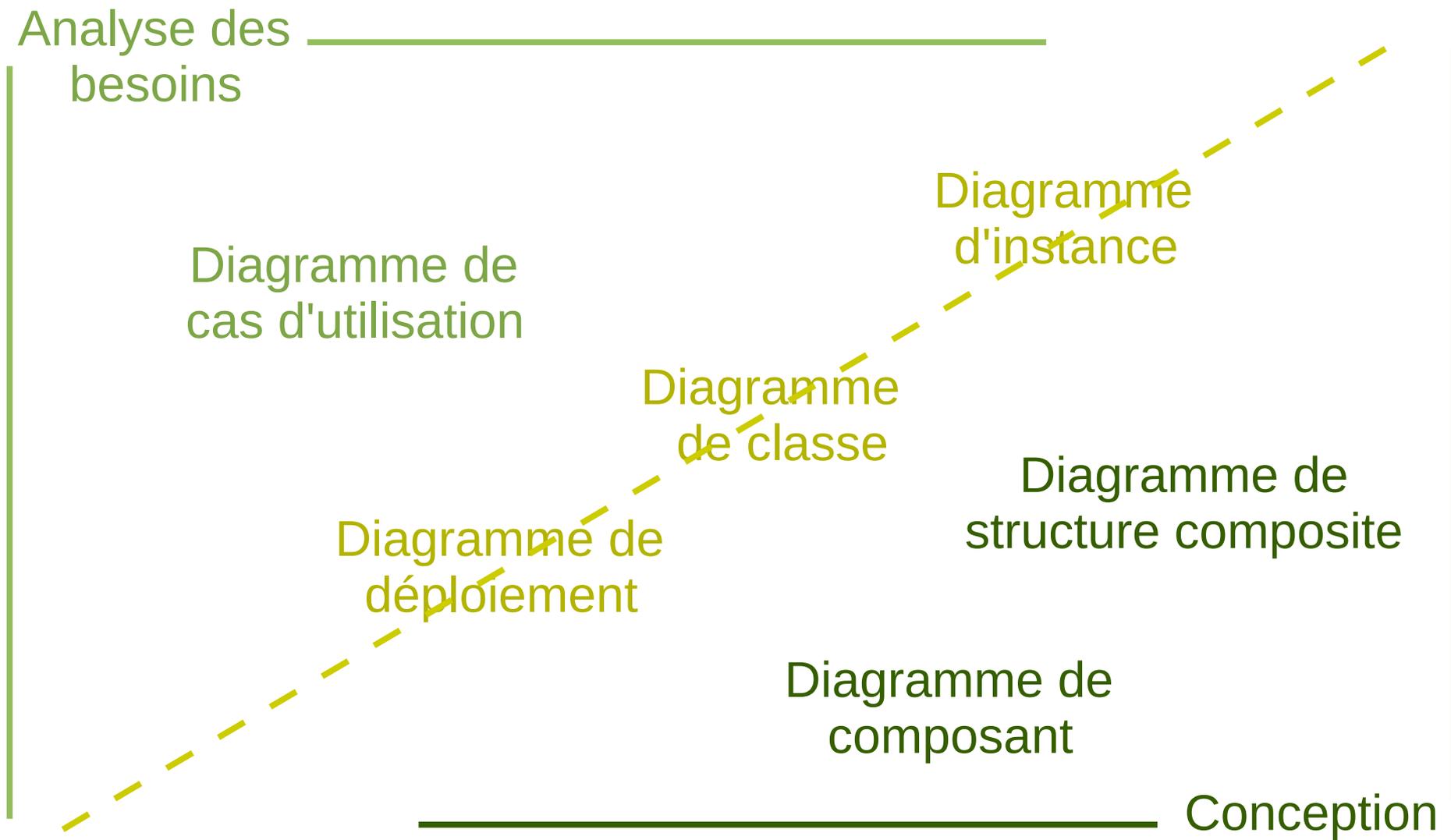
P L A N

- Introduction
- ▶ ■ Les diagrammes structurels
- Les diagrammes comportementaux
- Le diagramme de paquetage
- Conclusion

LES DIAGRAMMES STRUCTURELS

- ▶ Introduction
- Diagramme de cas d'utilisation
- Diagramme de classe
- Diagramme d'architecture
- Diagramme d'instance
- Diagramme de composant
- Diagramme de déploiement

LES DIAGRAMMES STRUCTURELS INTRODUCTION



LES DIAGRAMMES STRUCTURELS

- Introduction
- ▶ □ Diagramme de cas d'utilisation
- Diagramme de classe
- Diagramme d'architecture
- Diagramme d'instance
- Diagramme de composant
- Diagramme de déploiement



LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

- Des cas d'utilisation pour quoi faire ?
 - ✓ Spécifier le comportement d'un système du point de vue de l'utilisateur (besoins externes)
 - ✓ Décrire les limites du système & ses relations avec son environnement
 - ✓ Comprendre les besoins de l'utilisateur
 - ✓ noyés dans une grande quantité d'informations,
 - ✓ exprimés de façon non structurée,
 - ✓ sans forte cohérence (oublis, contraires, imprécisions)
 - ✓ Vue synthétique & claire de l'application



LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

□ Quoi spécifier ?

- ✓ Un **nouveau** système
- ✓ Les fonctionnalités offertes par un système **existant**

- ✓ Les cas d'utilisation sont aussi utiles pour :
 - ✓ **Évaluer** l'utilité et des fonctionnalités d'un système.
 - ✓ Améliorer la **communication** dans/entre les équipes
 - ✓ **Valider** des spécifications



LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

▣ ACTEUR

- ✓ Entités externes
- ✓ Interactions directes avec le système
- ✓ Personne, Matériel externe, Autres systèmes



Nom

➔ Comment identifier les acteurs ?

- ✓ *Qui utilise, installe, démarre, arrête, maintient l'application ?*
- ✓ *Qui donne et/ou reçoit des informations de l'application ?*
- ✓ *Quels autres systèmes utilisent les services proposés par l'application ?*
- ✓ ...
- ✓ Les acteurs ne font pas partie de l'application
 - ➔ Pas de développement associé
- ✓ Hiérarchies de généralisation & interfaces possibles entre acteurs



LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

□ CAS D'UTILISATION

Verbe
À l'infinitif

✓ Services demandés par l'utilisateur du système

➔ Comment identifier les cas d'utilisation ?

✓ *Qu'est ce que l'acteur attend de l'application ?*

✓ *Est-ce que l'application enregistre des informations ? Quels acteurs les créent, les consultent, les modifient ou les détruisent ?*

✓ *Est-ce qu'un acteur indique au système des changements dans son état ?*

✓ *Y-a-t-il des événements externes que l'application doit connaître ? Quels acteurs l'en informent ?*

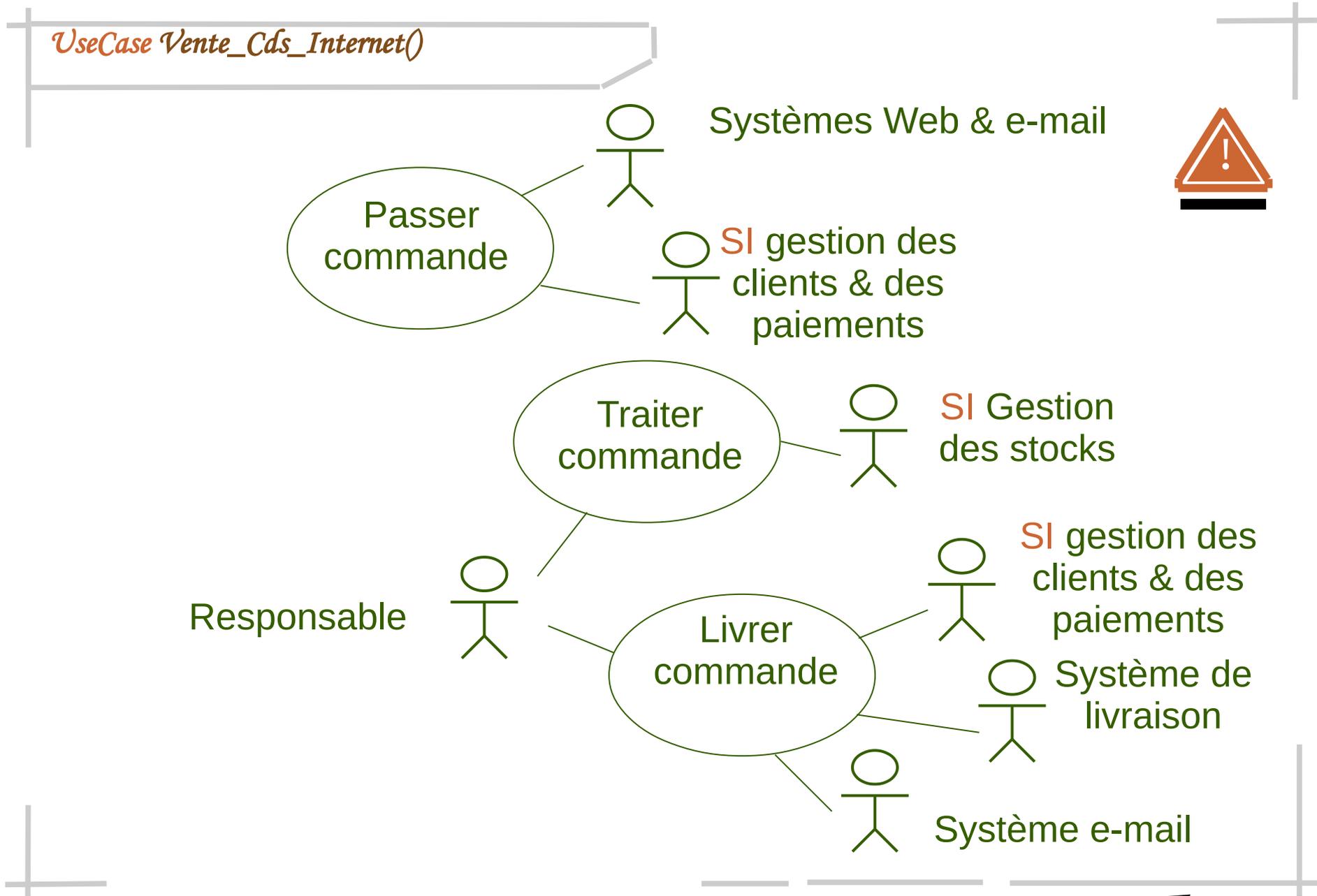
✓ ...

- ✓ Un cas est exécuté sans être durablement interrompu dans le temps
- ✓ Modélisation des fonctionnalités de l'application comme des boîtes noires
- ✓ Structuration selon l'utilisation qui en est faite (processus métier)
- ✓ Déclenchement & résultat(s) observables depuis l'extérieur du système



LES DIAGRAMMES STRUCTURELS

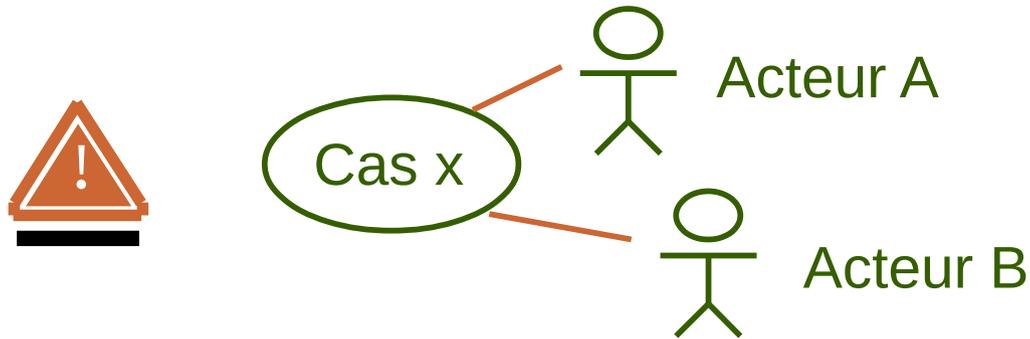
DIAGRAMMES DE CAS D'UTILISATION



LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

▣ ACTEURS & CAS D'UTILISATION



- ✓ Les deux acteurs sont nécessaires au fonctionnement du cas d'utilisation
- ✓ Le cas d'utilisation ne peut pas fonctionner sans les deux acteurs

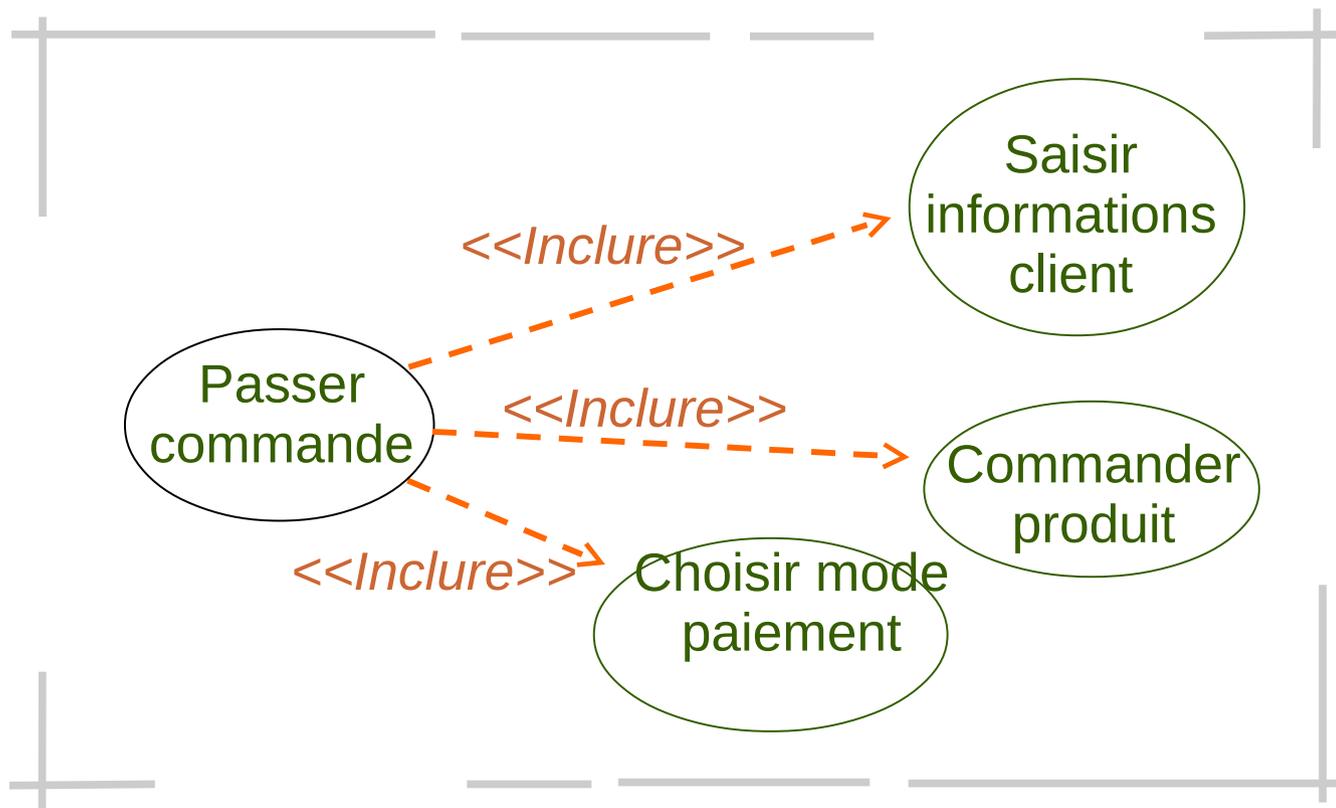


LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

□ RELATION D'INCLUSION (include)

- ✓ le cas d'utilisation *source* comprend/contient également le comportement décrit dans le cas d'utilisation *inclus*

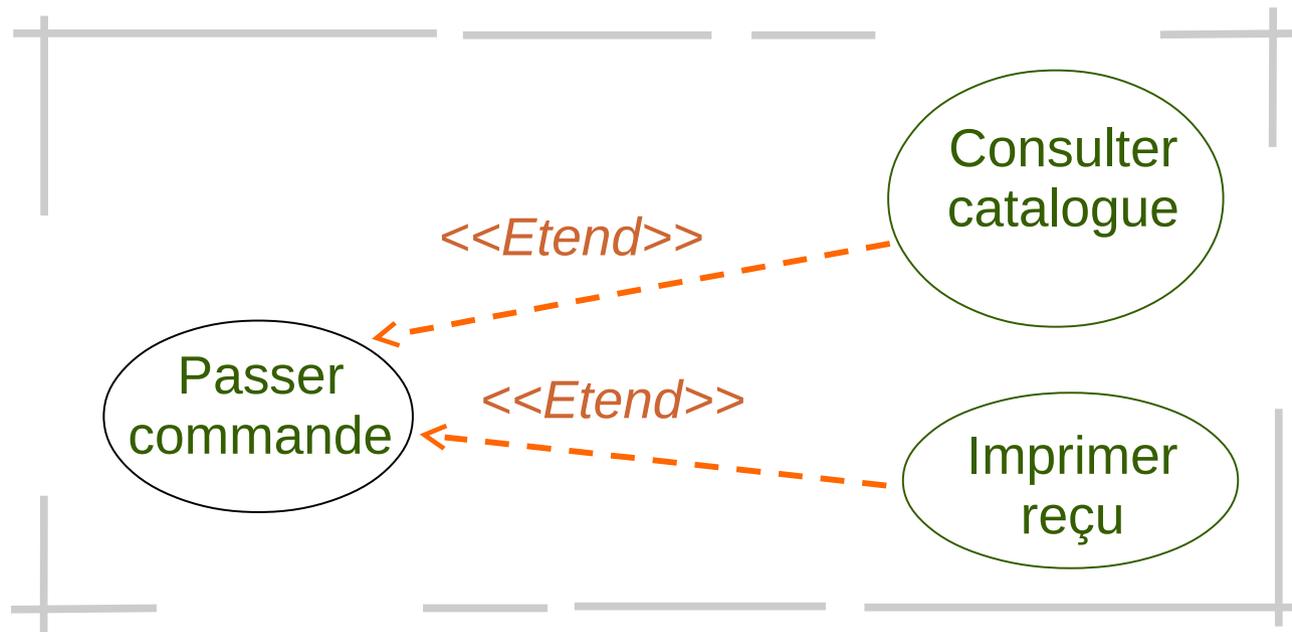


LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

RELATION D'EXTENSION (extend)

- ✓ le cas d'utilisation *source* est étendu par le comportement du cas d'utilisation en *extension*



Comme dans le transparent précédent, le cas d'utilisation *source* est "Passer Commande"

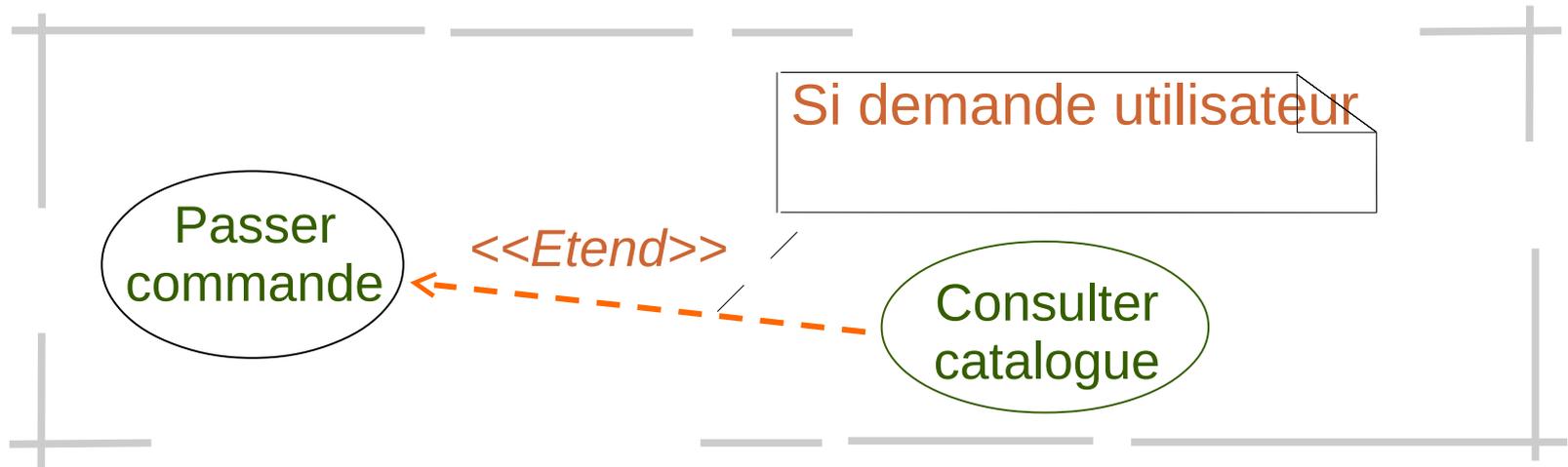


LES DIAGRAMMES STRUCTURELS

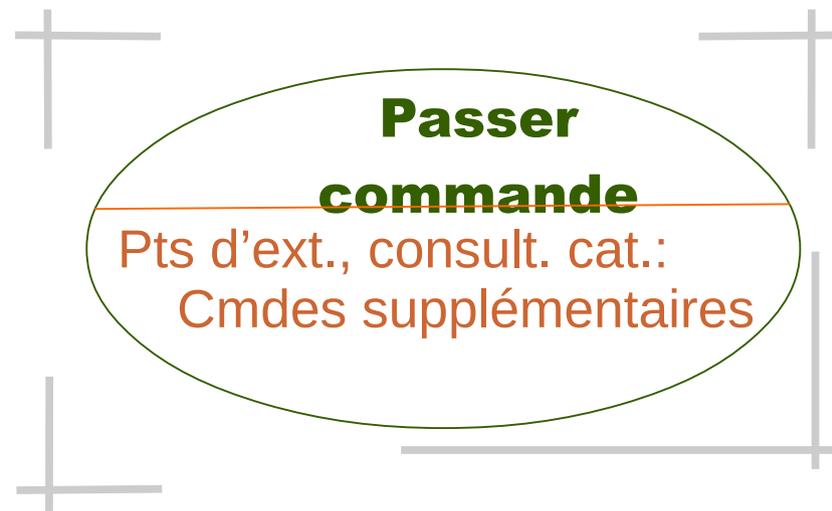
DIAGRAMMES DE CAS D'UTILISATION

□ RELATION D'EXTENSION (suite)

- ✓ **Condition** d'extension nécessaire



- ✓ Liste éventuelle des **points d'extension** pour préciser où est réalisée l'extension

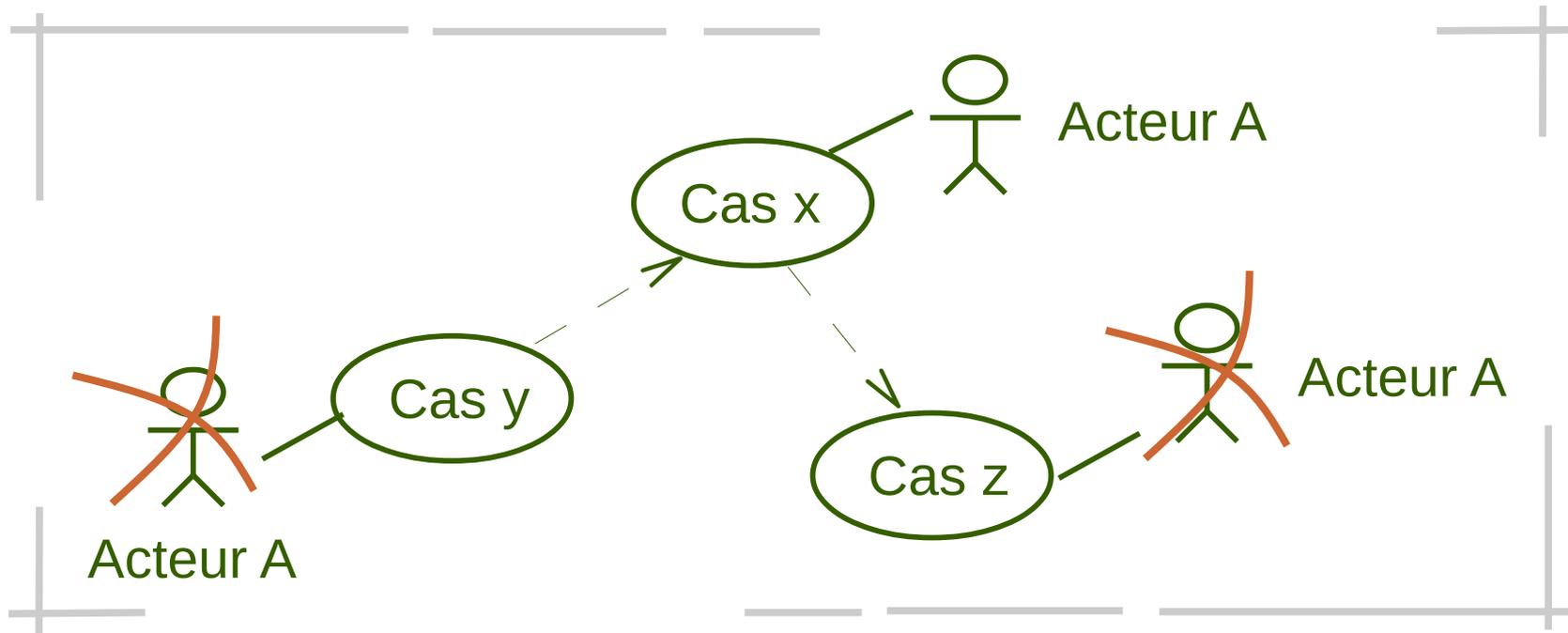


LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

▣ ACTEURS & CAS D'UTILISATION (suite)

- ✓ Il ne faut **pas répéter** les acteurs des cas d'utilisation principaux au niveau des cas d'utilisation **étendus** ou **inclus**



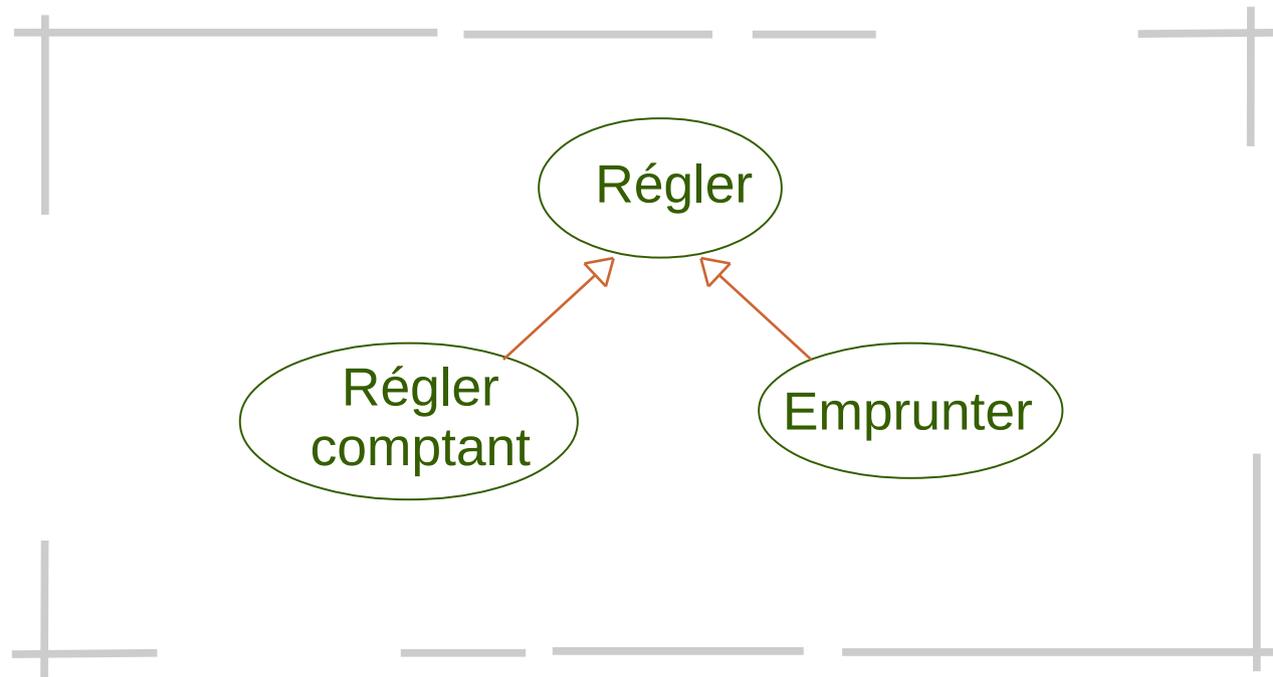
- ✓ **Répéter** les acteurs des cas d'utilisations principaux indique que les cas d'utilisation **inclus** ou **étendus** sont aussi **accessibles directement** (i.e. Ils font partie des cas d'utilisations principaux)

LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

□ GENERALISATION ENTRE CAS D'UTILISATION

- ✓ Pour définir des formes plus spécifiques de traitement
- ✓ Implique l'**héritage** des caractéristiques et des associations



LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

□ DESCRIPTION DETAILLEE DES CAS D'UTILISATION

- ✓ Pas de notation ou plan standardisés en UML
- ✓ Mais beaucoup d'informations à structurer...
- ✓ Plusieurs propositions dans la littérature

□ Exemple : Passer Commande (version très simplifiée...)

- ✓ Le **client** saisit ses informations
- ✓ Le **système** valide et enregistre les informations
- ✓ Le **système** propose les produits disponibles
- ✓ Le **client** sélectionne les produits à commander
- ✓ Le **système** affiche le récapitulatif de la commande
- ✓



LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

❑ DESCRIPTION DETAILLEE DES CAS D'UTILISATION

- ✓ Chaque cas d'utilisation doit être décrit en détail
- ✓ Commencer par les cas d'utilisation prioritaires

❑ Les informations à décrire :

- ✓ Quand le cas d'utilisation commence, **pré-conditions**
- ✓ Quand le cas d'utilisation se termine, **post-conditions**
- ✓ Le chemin correspondant au **déroulement normal**
= les interactions entre le système et les acteurs
- ✓ Les **variantes** possibles et les cas d'erreurs
- ✓ Les éventuels besoins **non fonctionnels**



LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

□ DESCRIPTION DETAILLEE : Exemple de rubriques

- ✓ Acteur principal : client
- ✓ **Préconditions** : Le client n'est pas connecté, ...
- ✓ Début (ou événement déclencheur) : Le client arrive sur la page de commande
- ✓ **Postconditions** : La commande du client est enregistrée et transmise au service de traitements, le paiement électronique est validée par le SI de paiement, ...
- ✓ Fin (ou événement de terminaison) : ...
- ✓ **Scénario nominal** (*ou déroulement normal*) :
 1. *Le système affiche les champs d'information à remplir...*
 2. *Le client remplit...*
 3.



LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

□ DESCRIPTION DETAILLEE : Exemple de rubriques (suite)

✓ **Scénario alternatif / variante / extension :**

✓ **A1** : Le client est déjà enregistré dans le système

✓ Démarrage à l'étape 2 du scénario nominal

1. Le client fournit son adresse électronique et son mot de passe

2. ...

✓ **A2** : Le client veut modifier les informations déjà enregistrées

✓ ...

✓ **Exception:**

✓ **E1** : Le client annule la commande

✓ **Contraintes non fonctionnelles:**

✓ Sécurité et confidentialité des échanges

✓ ...



LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

□ ORGANISATION DES DIAGRAMMES DE CAS D'UTILISATION

- ✓ Un (ou un ensemble) de **diagramme de plus haut niveau**
 - ✓ Placement des **acteurs**
 - ✓ Pas de raffinements des cas d'utilisation
 - ↳ Généralement pas de dépendances entre cas d'utilisation (sauf cas d'utilisation de plus haut niveau)

- ✓ Pour chaque cas d'utilisation du diagramme de plus haut niveau, si nécessaire, **décomposition dans un diagramme de plus bas niveau**
 - ✓ Inutile de répéter les acteurs
 - ✓ Répéter le cas d'utilisation à raffiner
 - ✓ **Relations de dépendance** entre cas d'utilisation obligatoires

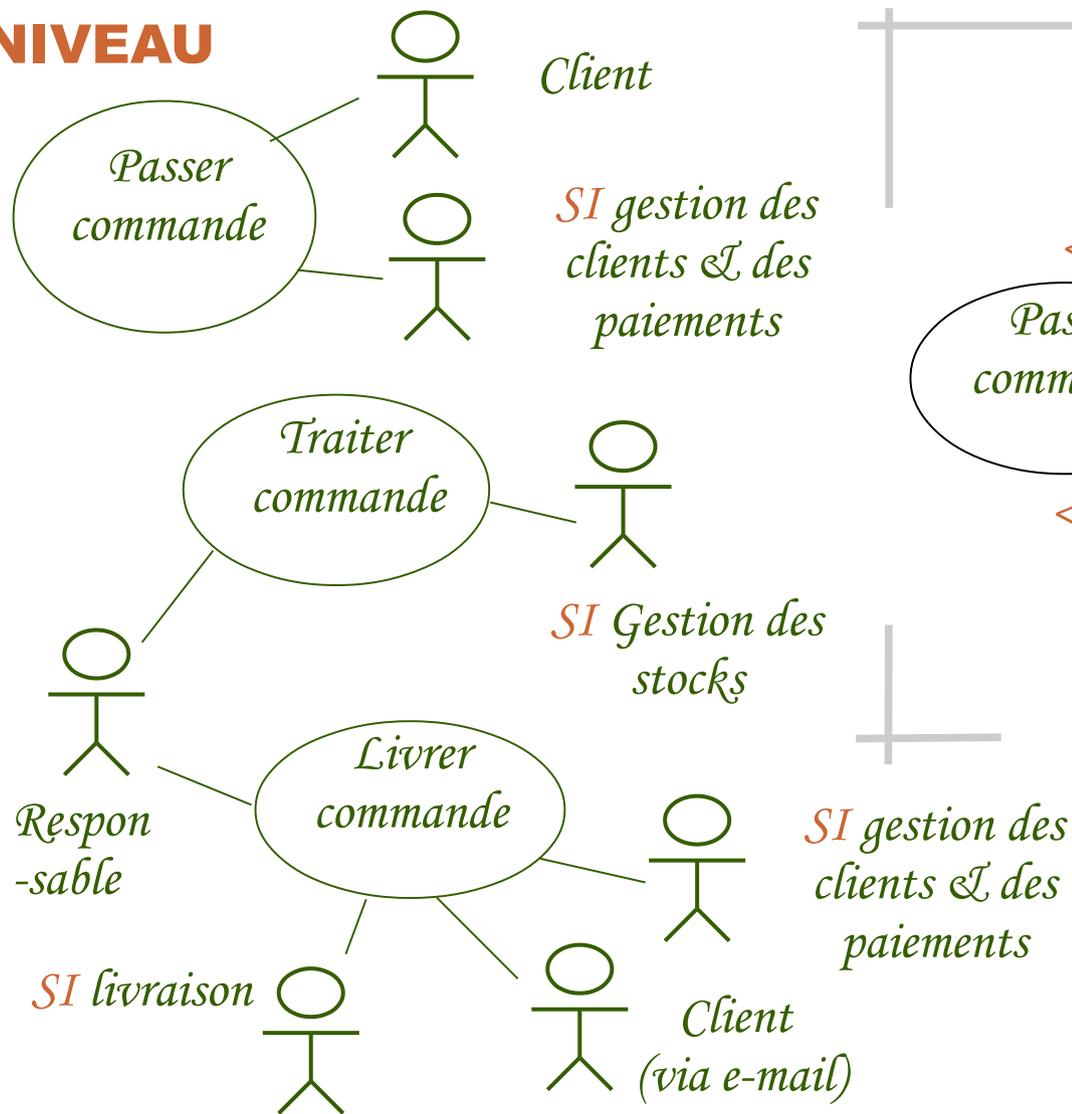
- ✓ Plusieurs niveaux de diagramme possibles



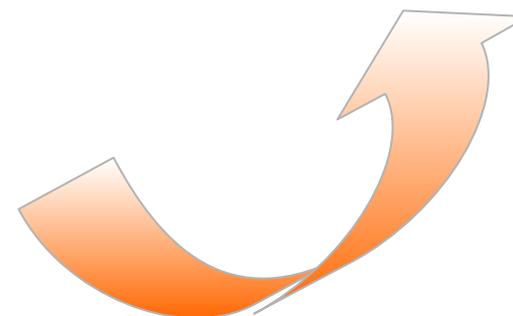
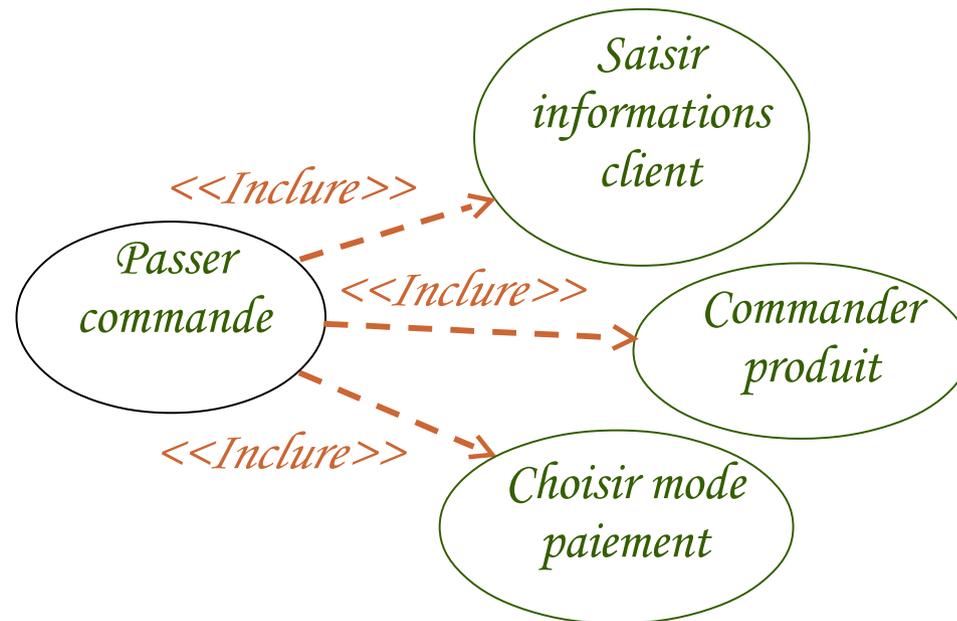
LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

DIAGRAMME DE PLUS HAUT NIVEAU



RAFFINEMENT



LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

❑ QUELQUES CONSEILS

- ✓ Un diagramme de cas d'utilisation doit toujours rester **simple**
- ✓ Donner un **nom parlant (verbe à l'infinitif)** à chaque cas d'utilisation
- ✓ Compléter la spécification du cas d'utilisation
 - ✓ Par une **description textuelle**
 - ✓ Eventuellement à l'aide du **diagramme d'activité**
- ✓ Ne pas modéliser à un niveau de détail trop précis

~~Entrer le nom du client~~

~~Entrer le prénom du client~~

- ✓ Ne pas modéliser en dehors de l'application

~~Choix des articles commandés~~

~~...~~



LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

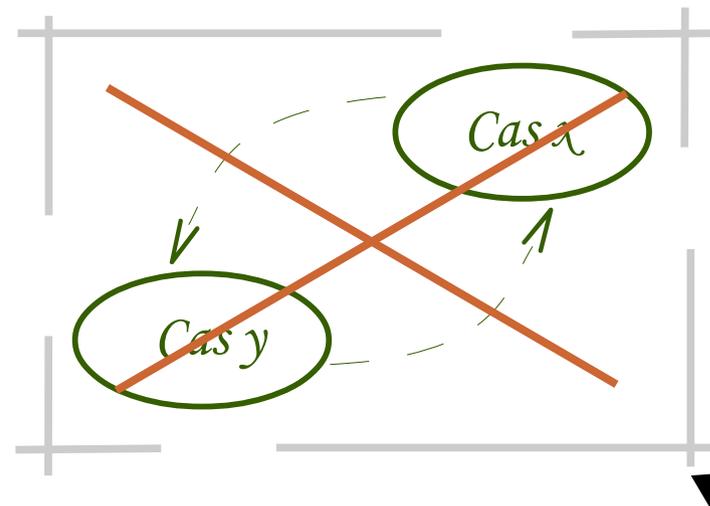
▣ QUELQUES CONSEILS (suite)

- ✓ Fragmenter les cas d'utilisation si
 - ➔ Interactions trop complexes
 - ➔ Isolation de parties indépendantes possible

- ✓ Il doit toujours y avoir au moins un acteur par cas d'utilisation
 - ➔ Explicite pour les cas d'utilisation de plus haut niveau
 - ➔ Via les relations d'extension & d'inclusion pour les autres

- ✓ Présentation des cas d'utilisation
 - ➔ Par séquence de déclenchement / acteur
 - ➔ Par hiérarchisation
 - ➔ Par domaine d'activité

- ✓ Eviter
 - ➔ De décomposer les cas d'utilisation / classes
 - ➔ Les cycles



LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

□ QUELQUES CONSEILS (suite)

- ✓ Représentation du comportement **normal** de l'application mais aussi
 - ➔ Des **variations** possibles
 - ➔ Des situations **exceptionnelles**

- ✓ Les cas d'utilisation doivent générer un **résultat**

- ✓ Les cas d'utilisation doivent traduire ce que fait l'application
 - ✓ Vision **claire**
 - ➔ Organiser les cas d'utilisations & les diagrammes

 - ✓ Vision complète
 - ➔ Représenter un **maximum** de choses
 - ➔ En **respectant** & en utilisant la notation choisie



LES DIAGRAMMES STRUCTURELS

DIAGRAMMES DE CAS D'UTILISATION

□ CAS D'UTILISATION & SCENARIOS

- ✓ Les scénarios sont
 - ✓ Des « instances » de cas d'utilisation
 - ✓ Des exemples de fonctionnement

- ✓ Formalisation des scénarios (UML)
 - ✓ Diagramme d'activité
 - ➔ Représentation de plusieurs scénarios dans un même diagramme
 - ✓ Diagramme de séquence ou de communication
 - ➔ Représentation d'un nombre réduit de scénario
 - ➔ Permet de faire le lien avec les objets

- ✓ Le langage naturel
 - ✓ Pouvoir d'expression élevé mais manque de précision
 - ✓ Permet difficilement d'exprimer les boucles, synchronisations, parallélisme, ...



LES DIAGRAMMES STRUCTURELS

- Introduction
- Diagramme de cas d'utilisation
- ▶ □ Diagramme de classe
- Diagramme d'architecture
- Diagramme d'instance
- Diagramme de composant
- Diagramme de déploiement

LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE CLASSE

► Généralités

Classes, attributs & opérations

Associations, cardinalité, qualifieur & rappels

Composition

Généralisation

Contraintes

Dépendances & interfaces

Quelques conseils



LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE CLASSE

MODELE DE CAS D'UTILISATION

Langage de l'utilisateur
Vue externe du système
Cas d'utilisation

Contrat utilisateur/développeur
Peut être redondant/inconsistant
Capture les fonctionnalités

Spécifie des cas d'utilisation

MODELE DE CLASSE

- ✓ Langage du développeur
- ✓ Vue interne du système
- ✓ Classes
- ✓ Vient des développeurs
- ✓ Ni redondant / ni inconsistant
- ✓ Réalisation des fonctionnalités
- ✓ Spécifie des réalisations de cas d'utilisation



LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

□ GENERALITES

- ✓ Structure statique du modèle (classes & relations)
- ✓ Nécessite des choix de représentation
 - ➔ *Informations utiles*
- ✓ Nécessite des choix de conception
 - ➔ *Technique performante pour stocker/accéder aux données, les modéliser*
- ✓ Classes & associations intéressants pour le système d'information
 - ➔ Objectif du système d'information implique la connaissance sur ces classes & ces liens



LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE CLASSE

Généralités

- ▶ Classes, attributs & opérations

Associations, cardinalité, qualifieur & rappels

Composition

Généralisation

Contraintes

Dépendances & interfaces

Quelques conseils



LES DIAGRAMMES

STRUCTURELS

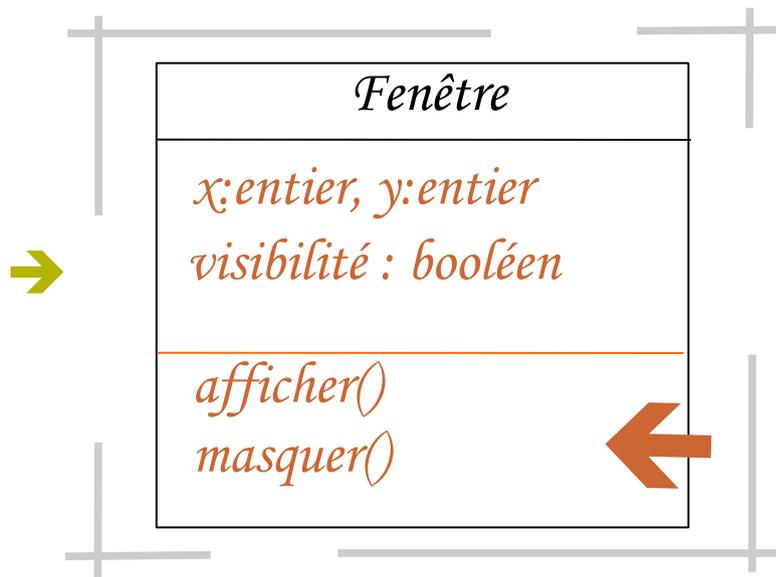
DIAGRAMME DE CLASSE

CLASS

E Notation
sans
détails



Notation
détaillée



Traitements
fonctionnels



Compartiment(s)
additionnel(s)



Règles de gestion,
Responsable,
Exceptions, ...



LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

■ CLASSE (suite)

- ✓ Comment trouver les classes ?
 - ✓ Abstraction des objets du monde réel
 - ✓ Regroupement d'attributs
 - ✓ Sémantiquement proches
 - ✓ Ayant le même cycle de vie
- ✓ Pas de redondance d'attributs dans le diagramme
- ✓ Une seule valeur d'attribut par instance



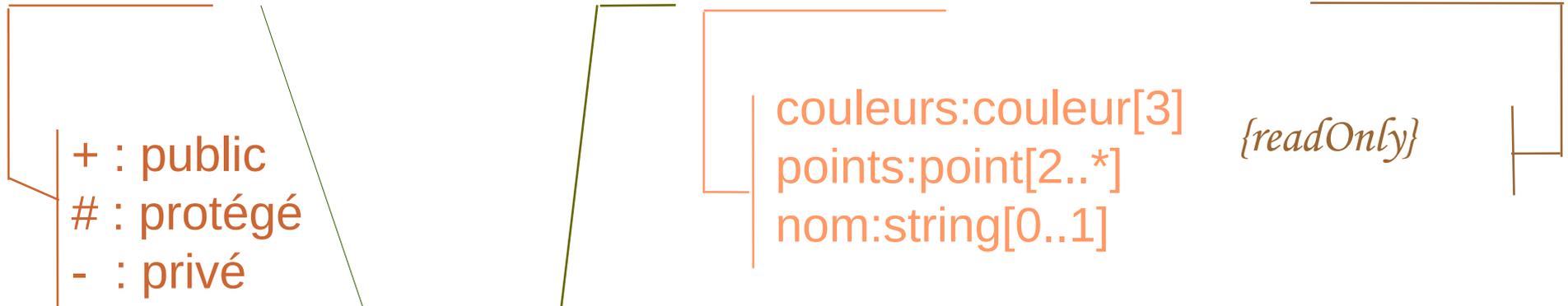
LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

 ATTRIBUT

Visibilité / nom_attribut : type [multiplicité] = valeur_initiale {propriétés}



Attribut dérivé

✓ type primitif : description de valeurs primitives dépourvues d'identité: nombres, chaînes, énumération, ...

✓ référence à une classe

Niveau
conception

✓ Attribut abstrait

✓ Attribut de classe ou static

✓ Attribut dérivé

→ *nom* ... mais aussi {abstrait}

→ nom

→ /nom



LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

▣ ATTRIBUT (suite)

- ✓ Comment **trouver** les attributs ?
 - ✓ Informations **pertinentes** du domaine d'application
- ✓ Comment **placer** les attributs ?
 - ✓ Dans la classe qui modélise l'objet du monde réel que décrit l'attribut
 - ✓ **Une seule valeur** par instance de la classe
- ✓ Un seul endroit pour chaque attribut dans le diagramme de classe
- ✓ Eviter les types non **primitifs** **Classes & Associations**



Classes & Associations



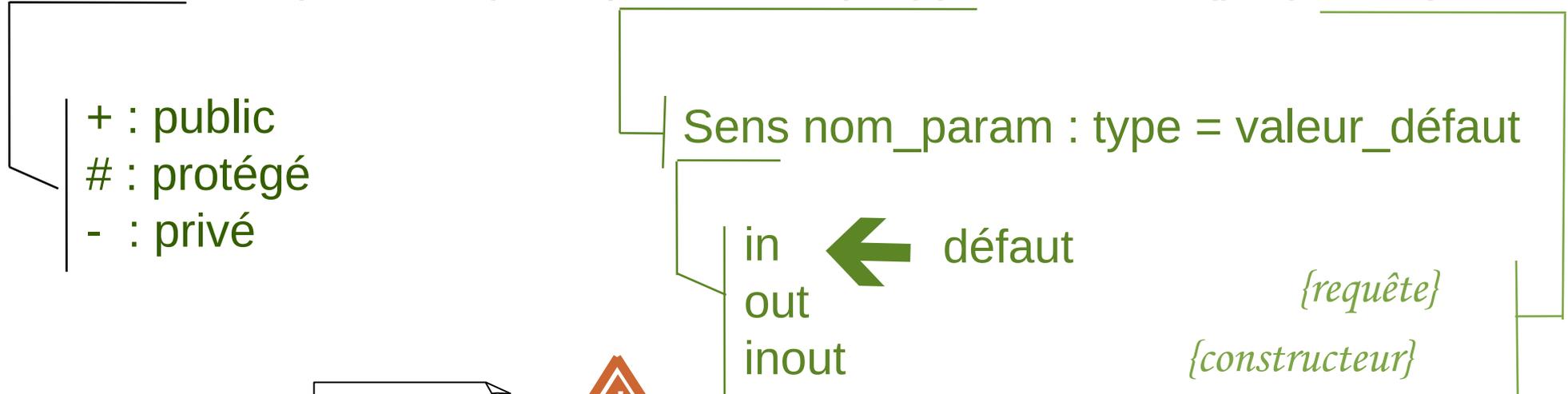
LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

 OPERATION

visibilité **nom_opération** (liste_paramètres) : **type_retourné** {propriétés}



✓ Note



➔ attachée à une opération pour informations (texte, algorithme, ...)

<<precondition>>
 {condition1}

- ✓ Opération abstraite ➔ *nom* ... mais aussi {abstraite}
- ✓ Opération de classe ou static ➔ nom
- ✓ Opération **non** polymorphe ➔ {leaf}



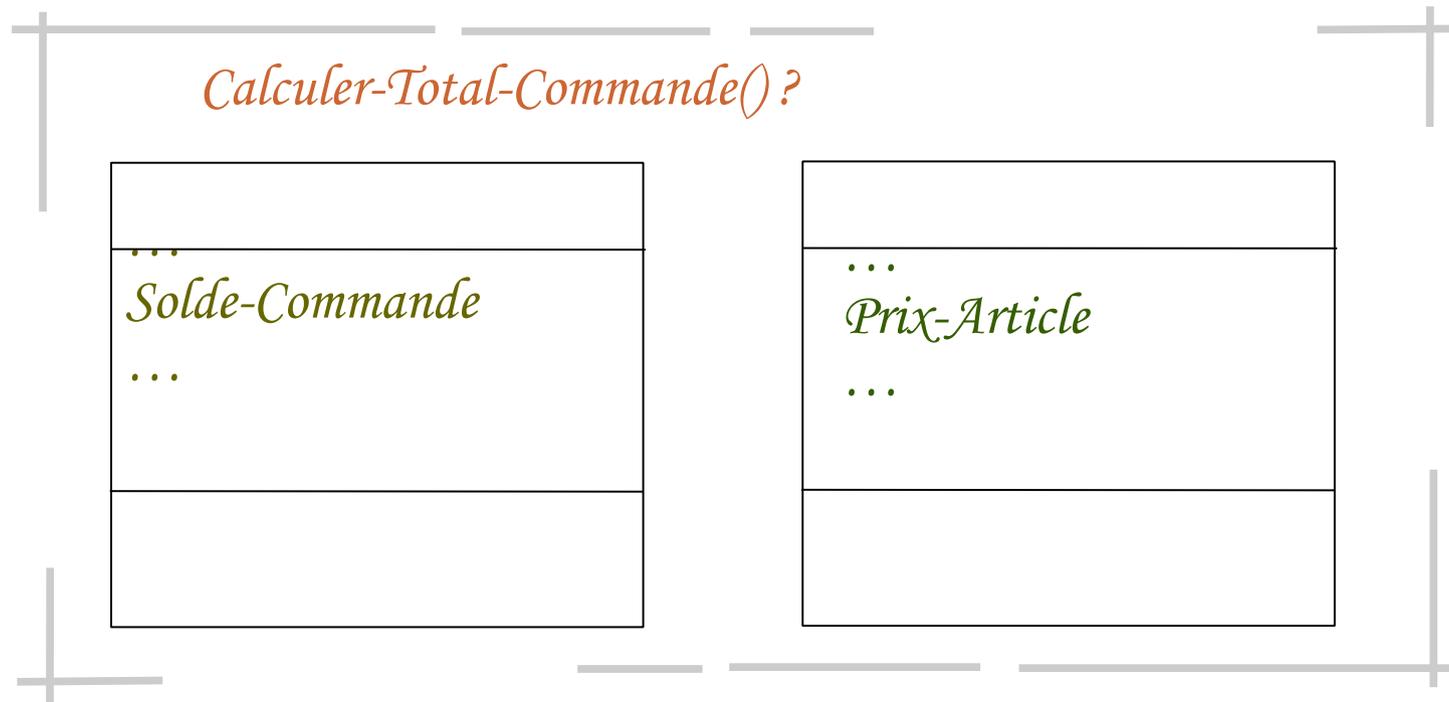
LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

❑ OPERATION (suite)

- ✓ Comment trouver les opérations ?
 - ✓ Traitements **fonctionnels** du domaine d'application
- ✓ Comment placer les opérations ?
 - ✓ Dans la classe où se trouvent les attributs créés ou modifiés



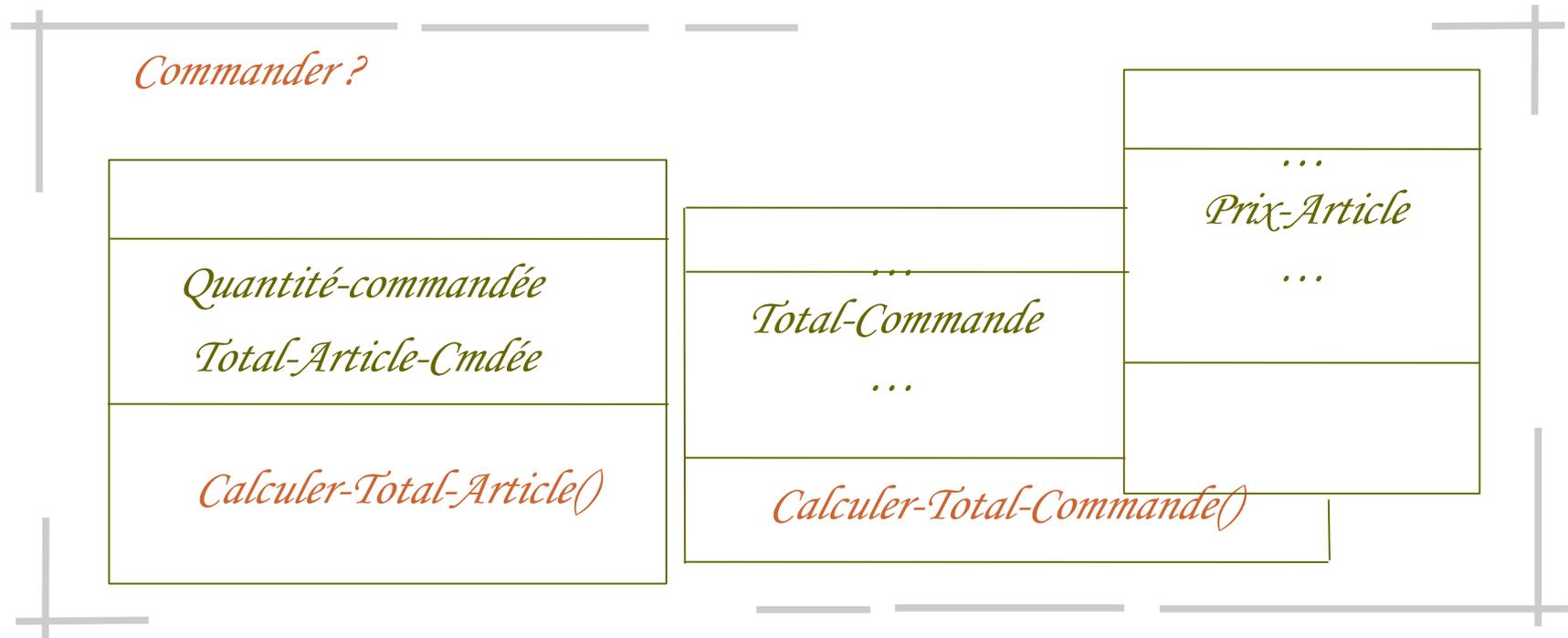
LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

❑ OPERATION (suite)

- ✓ Comment placer les opérations ? (suite)
 - ✓ Un traitement fonctionnel peut être traduit par plusieurs opérations si les attributs sont dans des classes différentes



- ✓ Cohérence attributs/opérations
- ✓ Respecter les principes de l'encapsulation



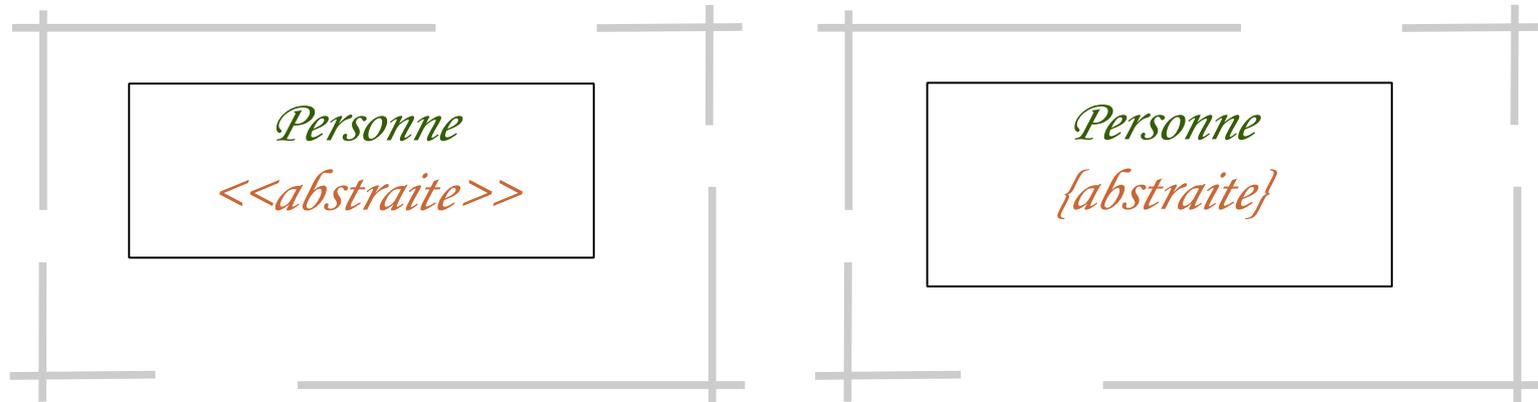
LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

CLASSE (suite)

- ✓ Classe abstraite
 - ✓ Pas d'instance
 - ✓ Attributs & opérations possibles
 - ✓ Abstrait ou non



LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE CLASSE

Généralités

Classes, attributs & opérations

- ▶ Associations, cardinalité, qualifieur & rappels

Composition

Généralisation

Contraintes

Dépendances & interfaces

Quelques conseils



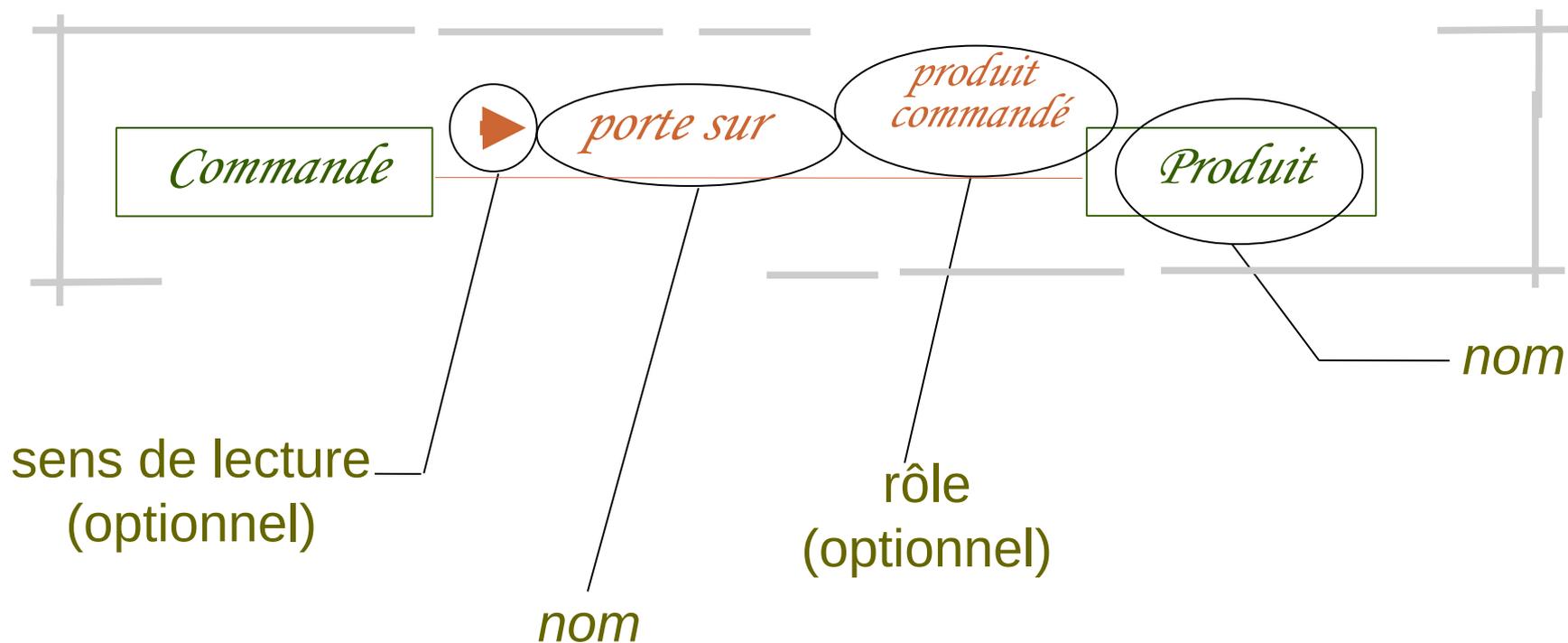
LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

 ASSOCIATION

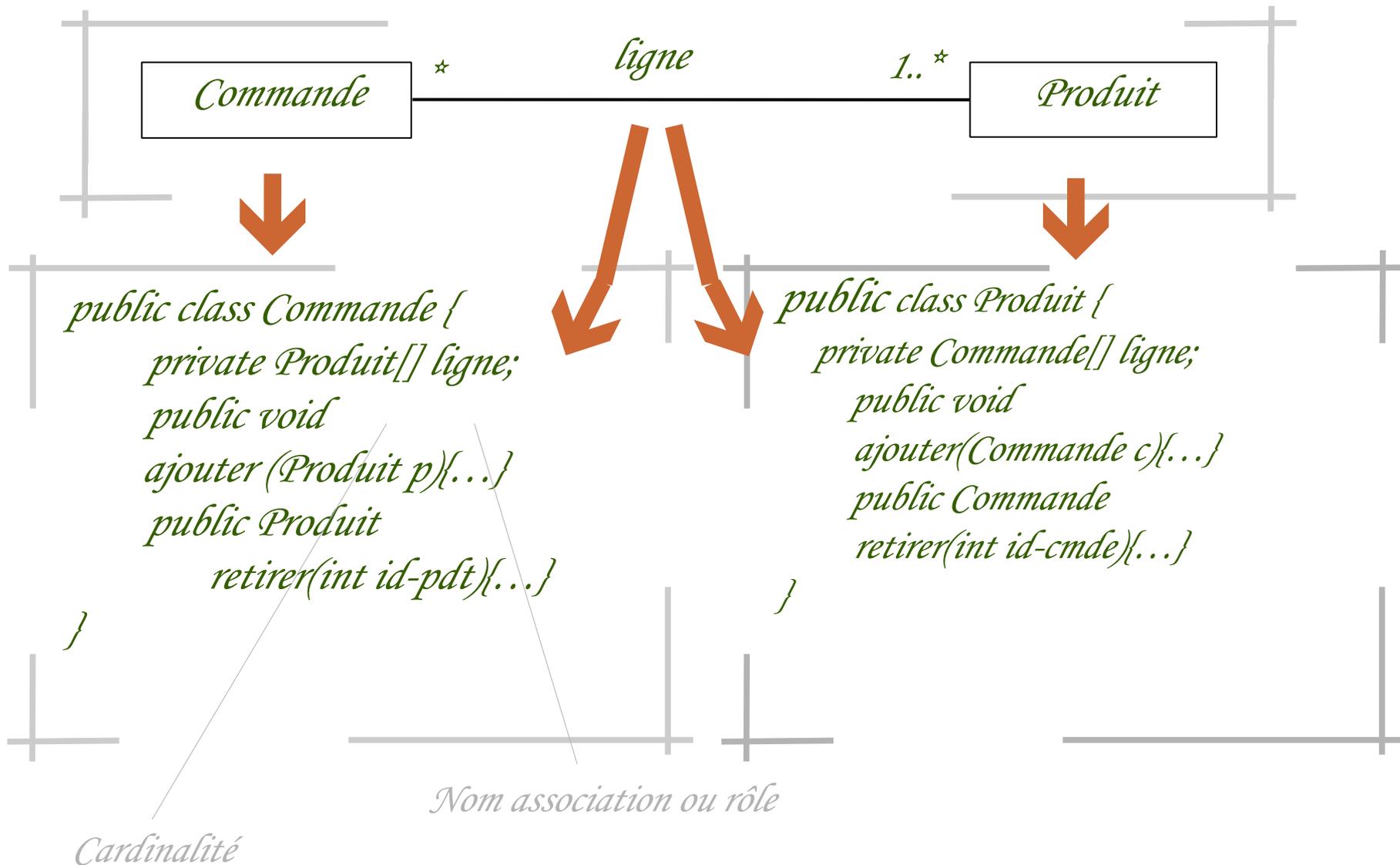
✓ Relation **structurelle** entre classes d'objets



LES DIAGRAMMES

STRUCTURELS

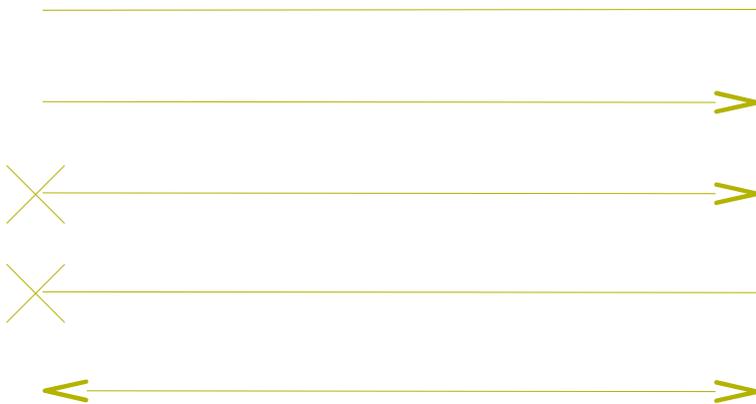
DIAGRAMME DE CLASSE

 DE LA CONCEPTION A L'IMPLEMENTATION


LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

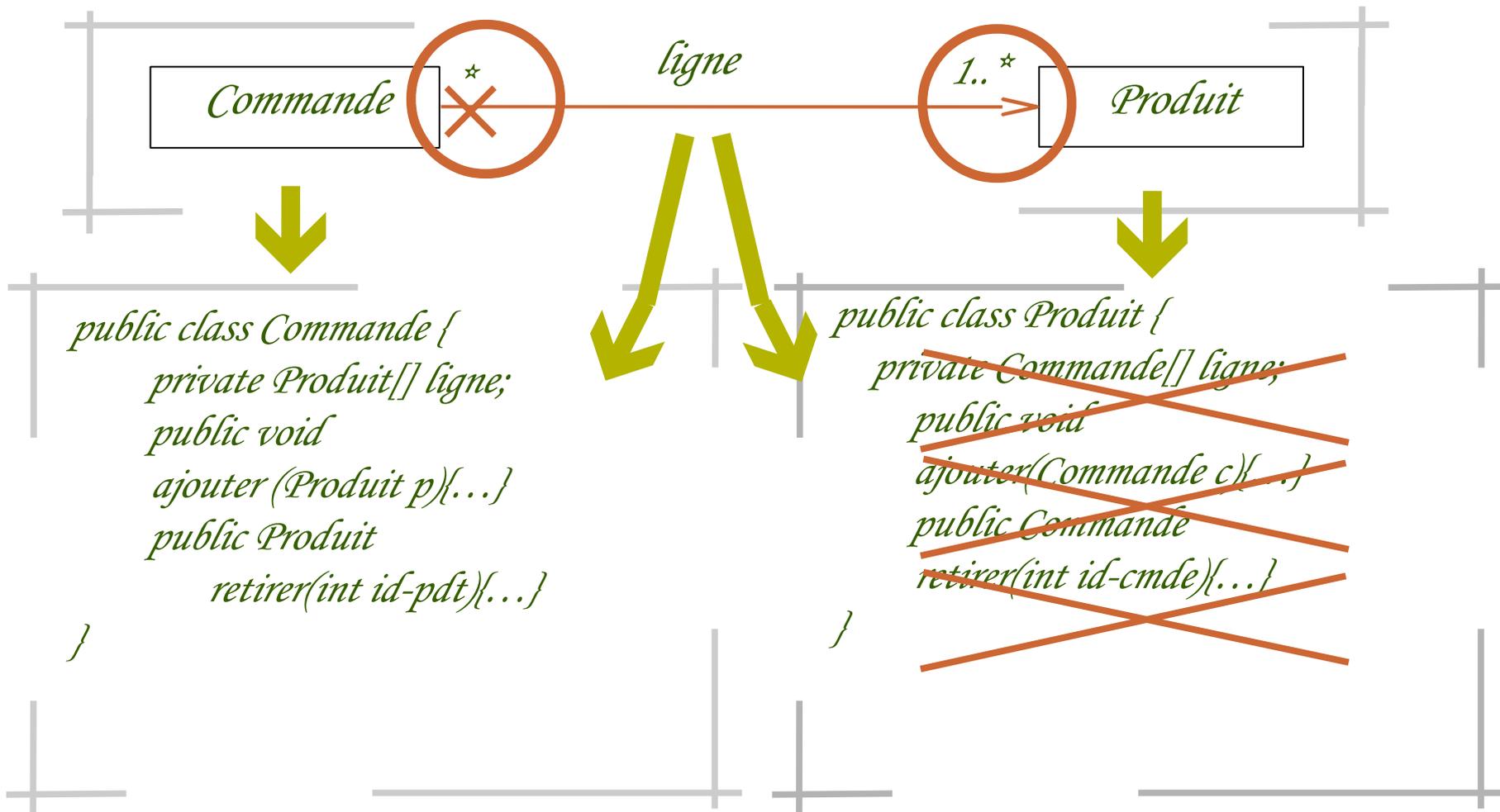
 ASSOCIATION (suite)


LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

DE LA CONCEPTION A L'IMPLEMENTATION (suite)



LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

□ DE LA CONCEPTION A L'IMPLEMENTATION (suite)



```

public class Commande {
    private Produit[] ProduitsCommandés;
    public void
    ajouter (Produit p){...}
    public Produit
    retirer(int id-pdt){...}
}

```



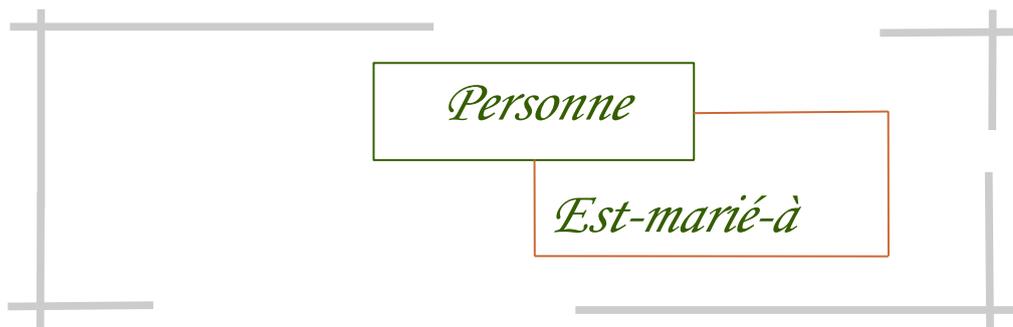
LES DIAGRAMMES

STRUCTURELS

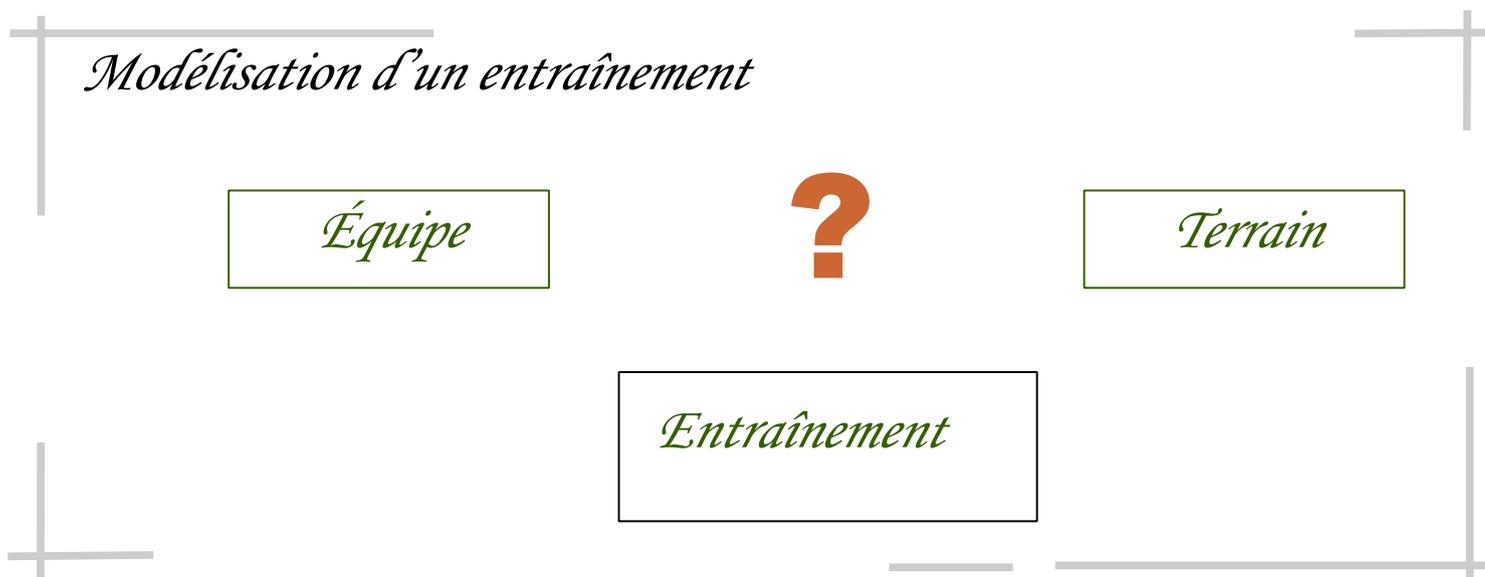
DIAGRAMME DE CLASSE

- ASSOCIATION (suite)

- ✓ Association **réflexive**



- ✓ Association ...

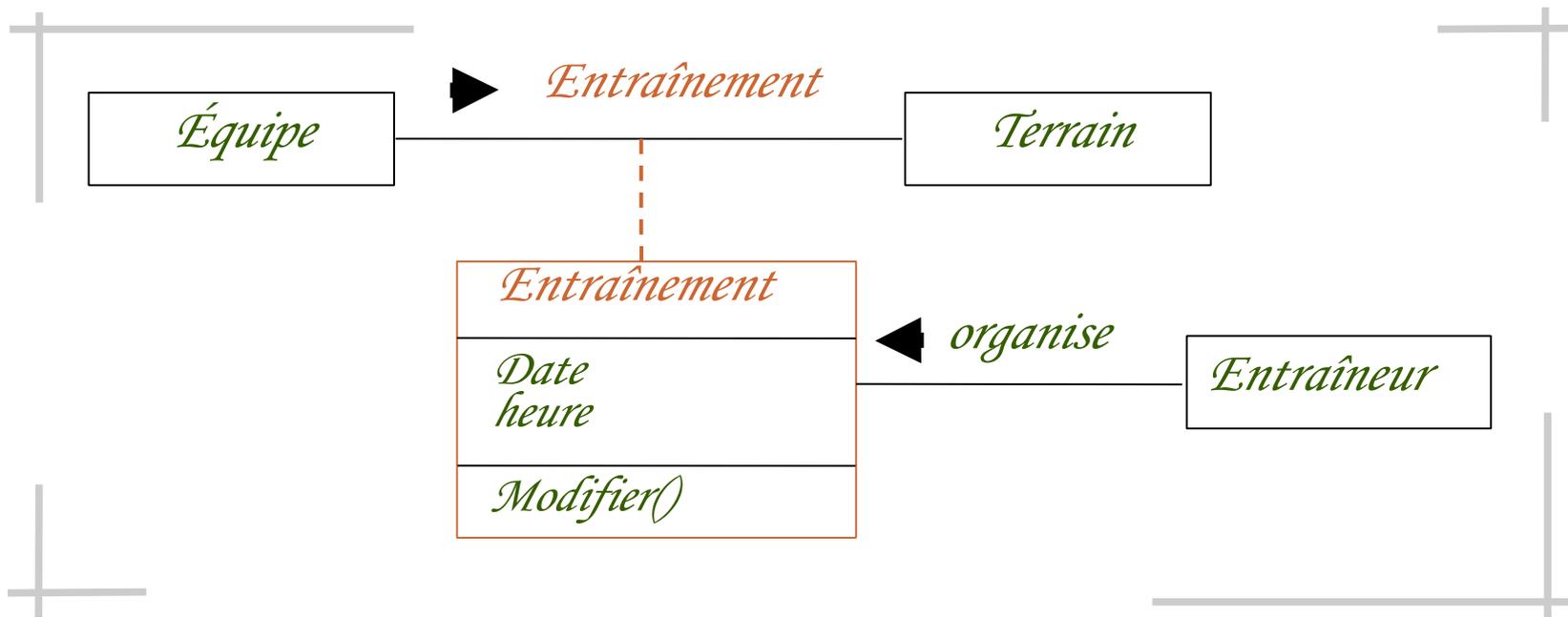


LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE CLASSE

- ASSOCIATION (suite)

✓ Association porteuse d'attributs : Classe-association



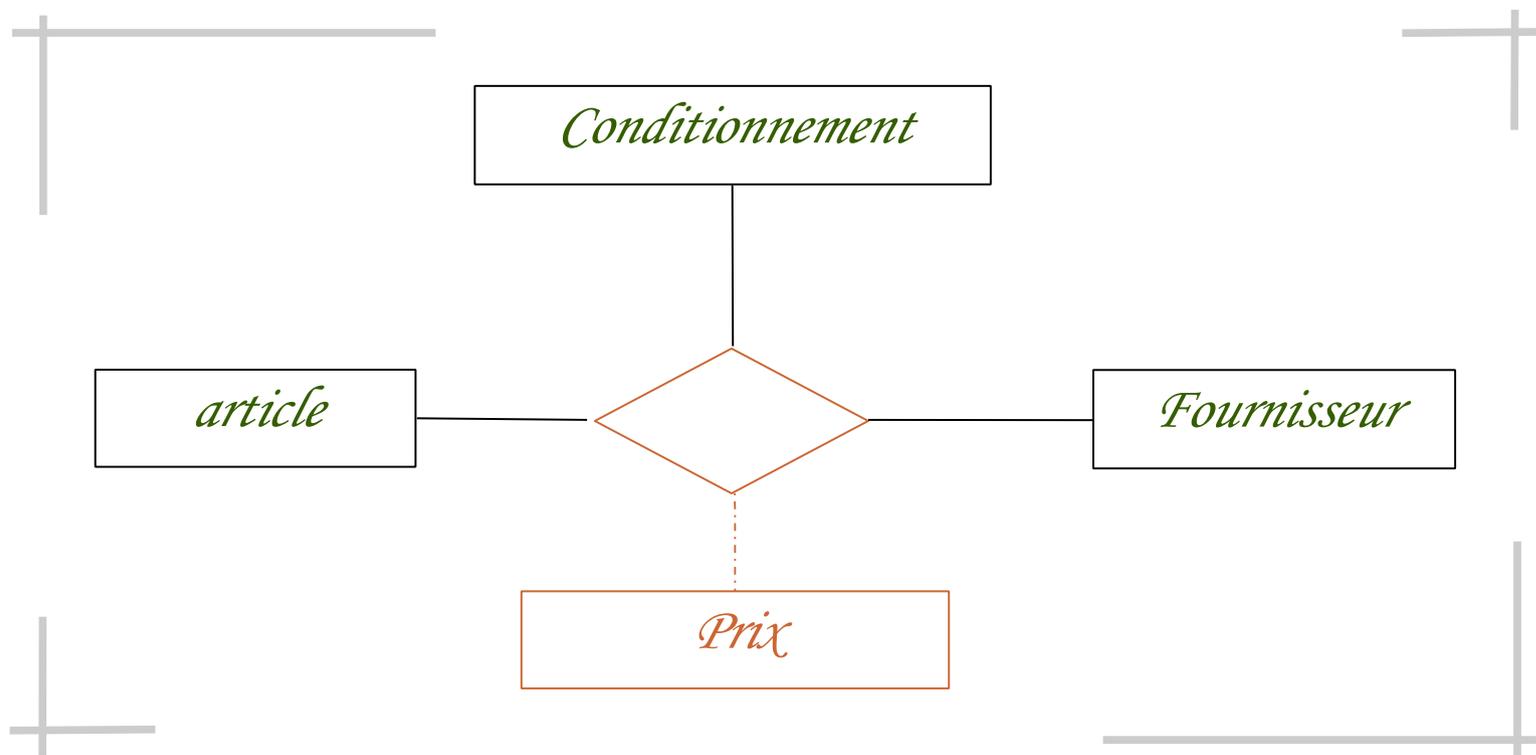
LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

■ ASSOCIATION (suite)

✓ Association **n-aire** : au moins 3 classes



➔ généralement porteuse d'attributs

LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE CLASSE

CARDINALITE



✓ **Cardinalité** : minimale & maximale

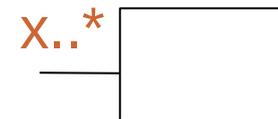
- exactement x



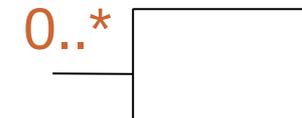
- de x à y ($y > x$)



- au moins x



- 0 ou plusieurs

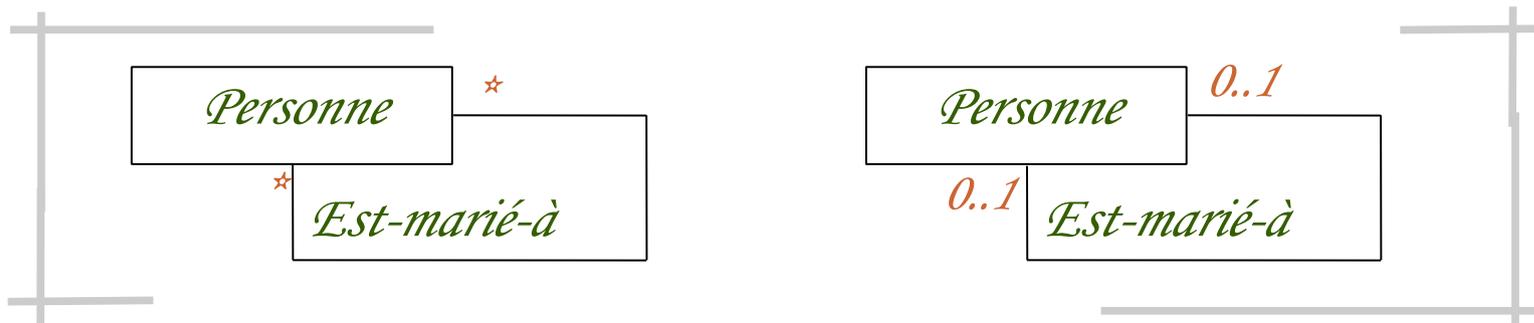
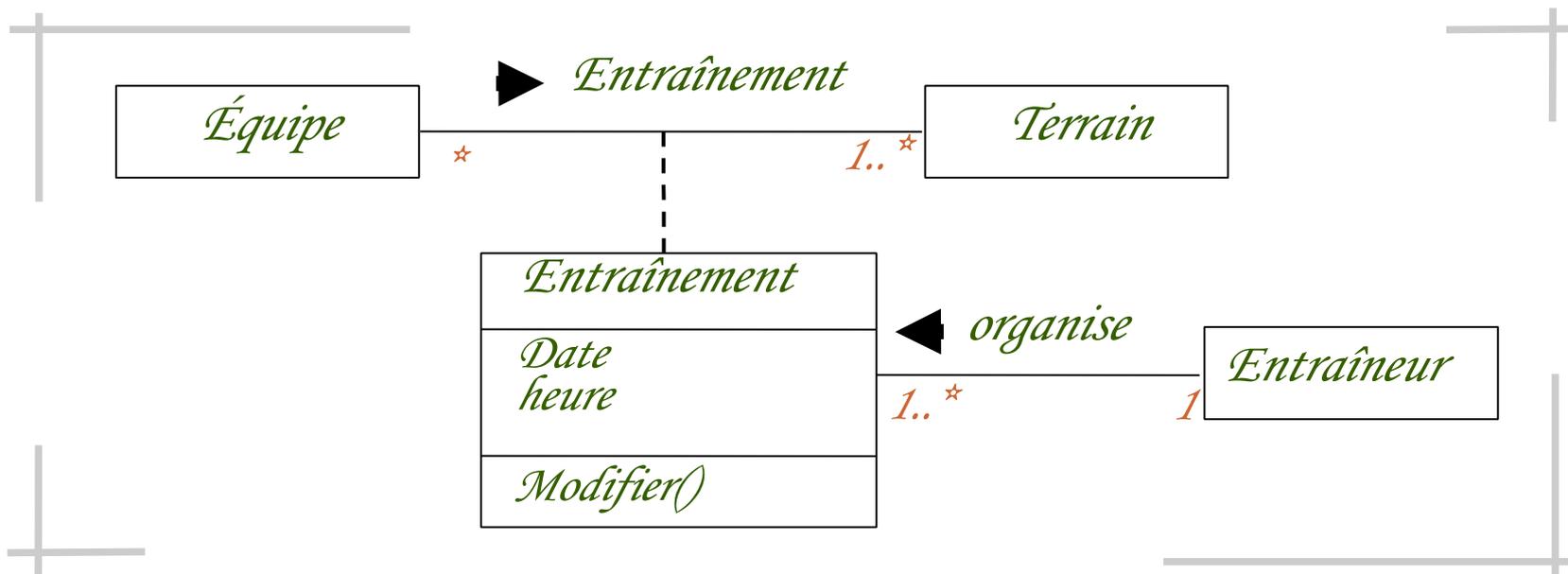


LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

 CARDINALITE (suite)

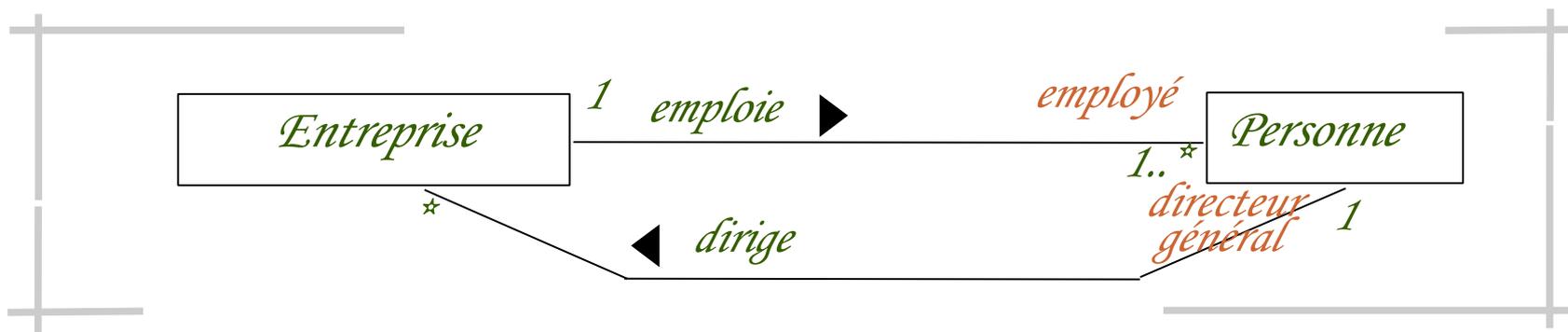
 Association réflexive

 Classe-association


LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

■ CARDINALITE (suite)

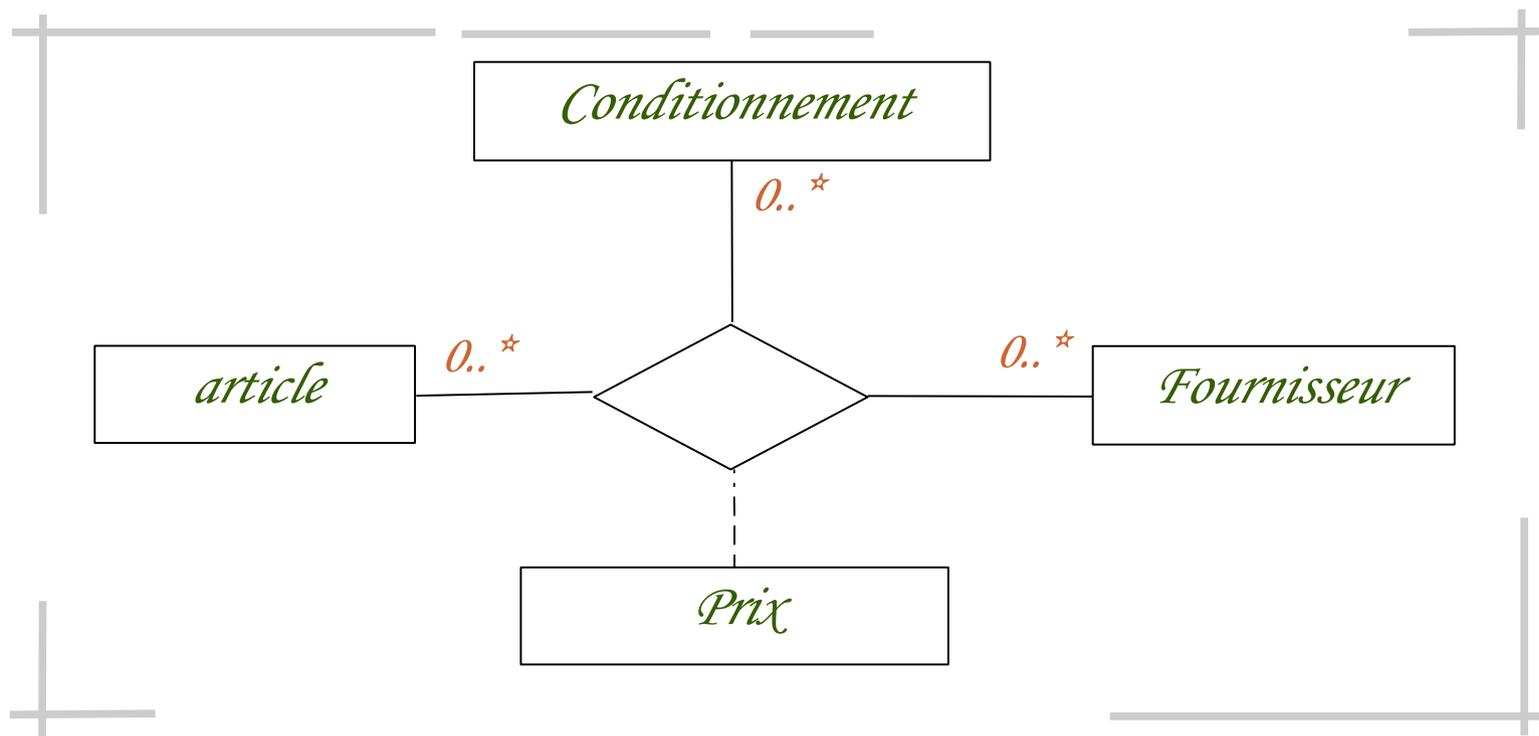


LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

 CARDINALITE (suite)

 Association n-aire


Sens de lecture

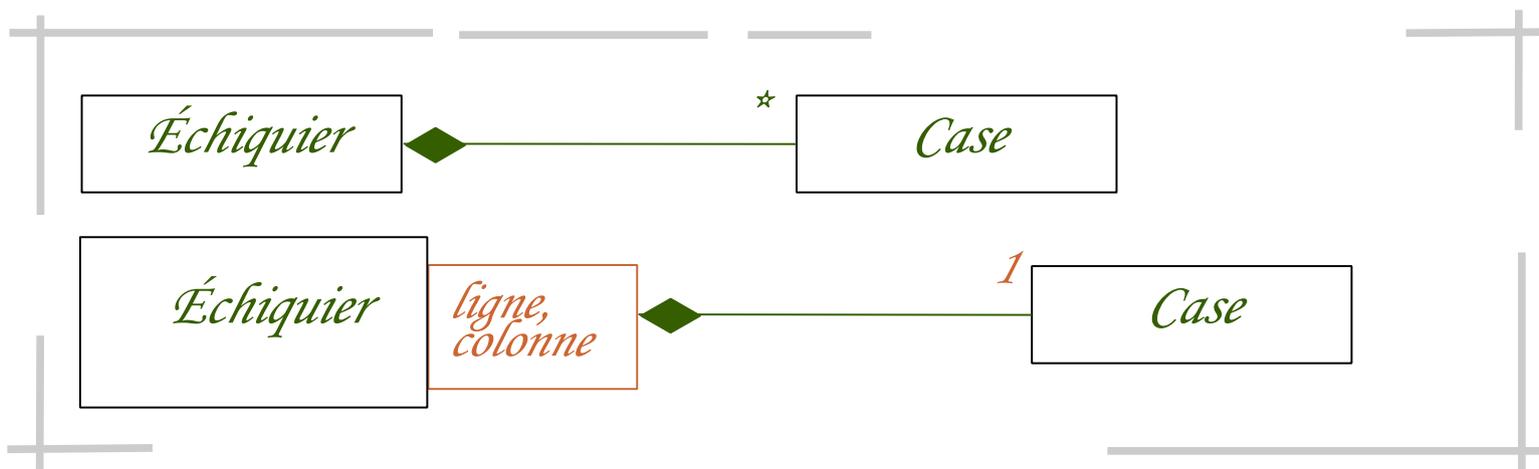
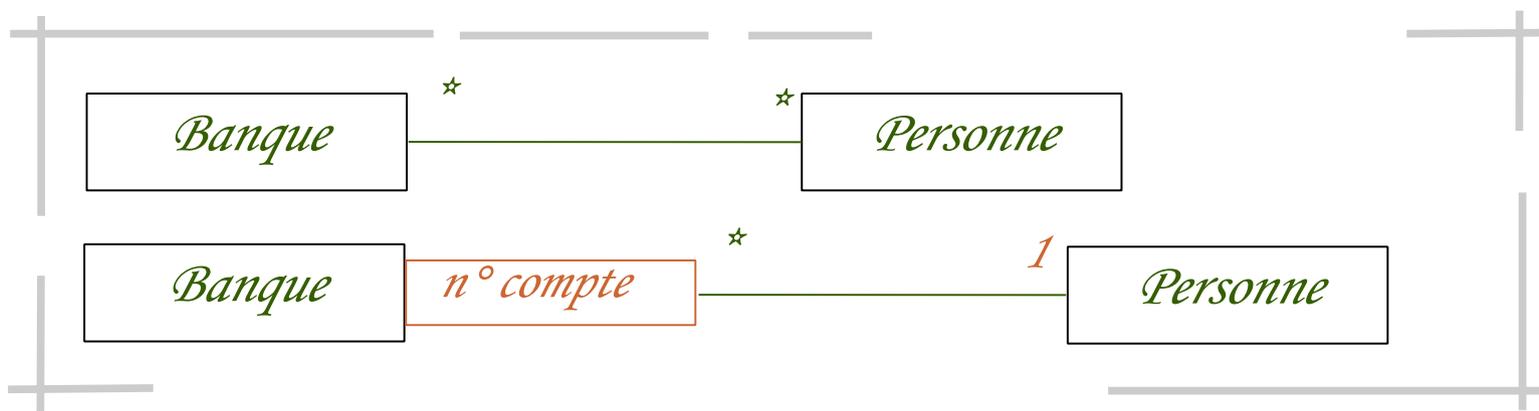


LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE CLASSE

■ QUALIFIEUR

- ✓ Pour partitionner un ensemble d'instances
- ✓ Conseillé pour des associations * _____ *



LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE CLASSE

▣ RAPPEL - UTILISATION DES ASSOCIATIONS

✓ Rappels

- ✓ Un seul endroit pour chaque attribut dans le diagramme de classe
- ✓ Pas de types autres que primitifs

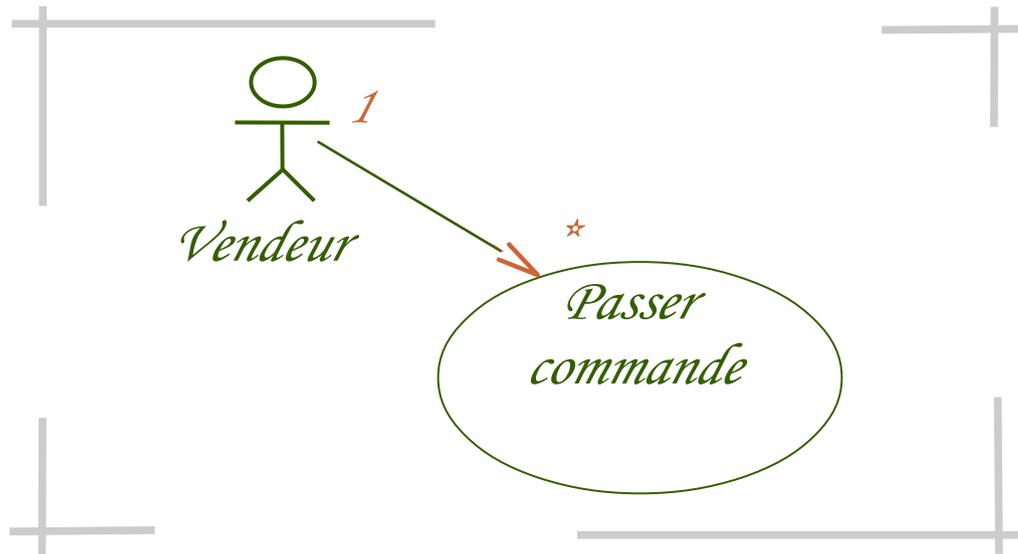
➔ Utiliser les associations



LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE CLASSE

▣ RAPPEL – CAS D'UTILISATION & ASSOCIATIONS



- ➔ Utiliser la navigation pour indiquer le sens d'utilisation de l'association
 - ✓ Entrant : l'acteur sollicite le système
 - ✓ Sortant : le système sollicite un acteur
 - ✓ 1 entrant obligatoire
 - ✓ Contrainte d'exclusion si plusieurs entrants



LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE CLASSE

Généralités

Classes, attributs & opérations

Associations, cardinalité, qualifieur & rappels

► Composition

Généralisation

Contraintes

Dépendances & interfaces

Quelques conseils

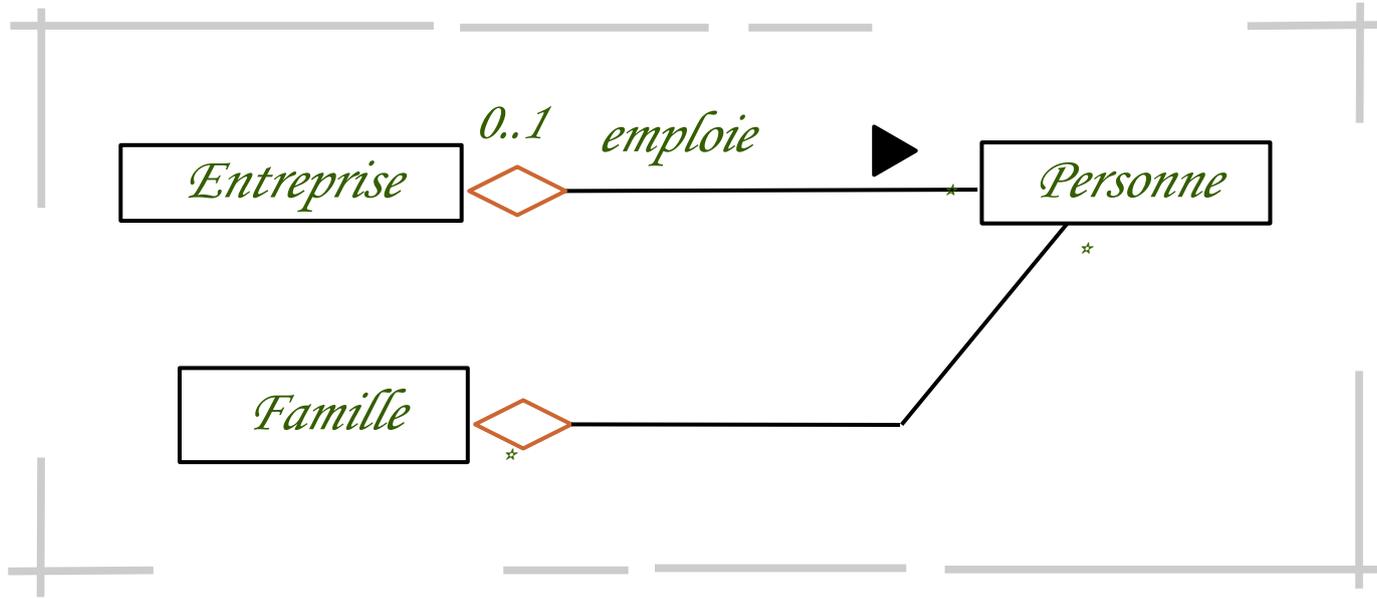


LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE CLASSE

COMPOSITION

- ✓ Agrégation
 - ✓ Pour enrichir la sémantique
 - ✓ Pas de contrainte
 - ✓ sur la valeur de multiplicité (partage possible)
 - ✓ sur les durées de vie des objets



LES DIAGRAMMES

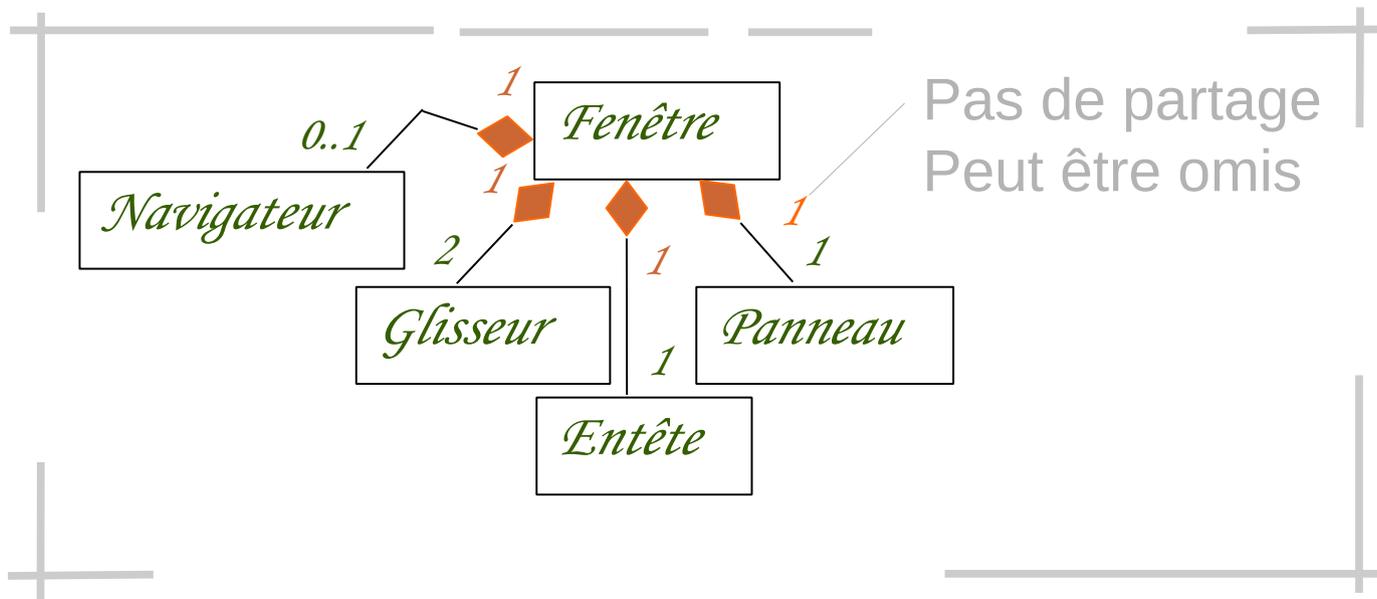
STRUCTURELS

DIAGRAMME DE CLASSE

 COMPOSITION (suite)

 Composition

-  Notion de possession (contient, est composé de, ...)
-  Durées de vie liées composants/composé



"Navigateur" n'est pas un browser web...

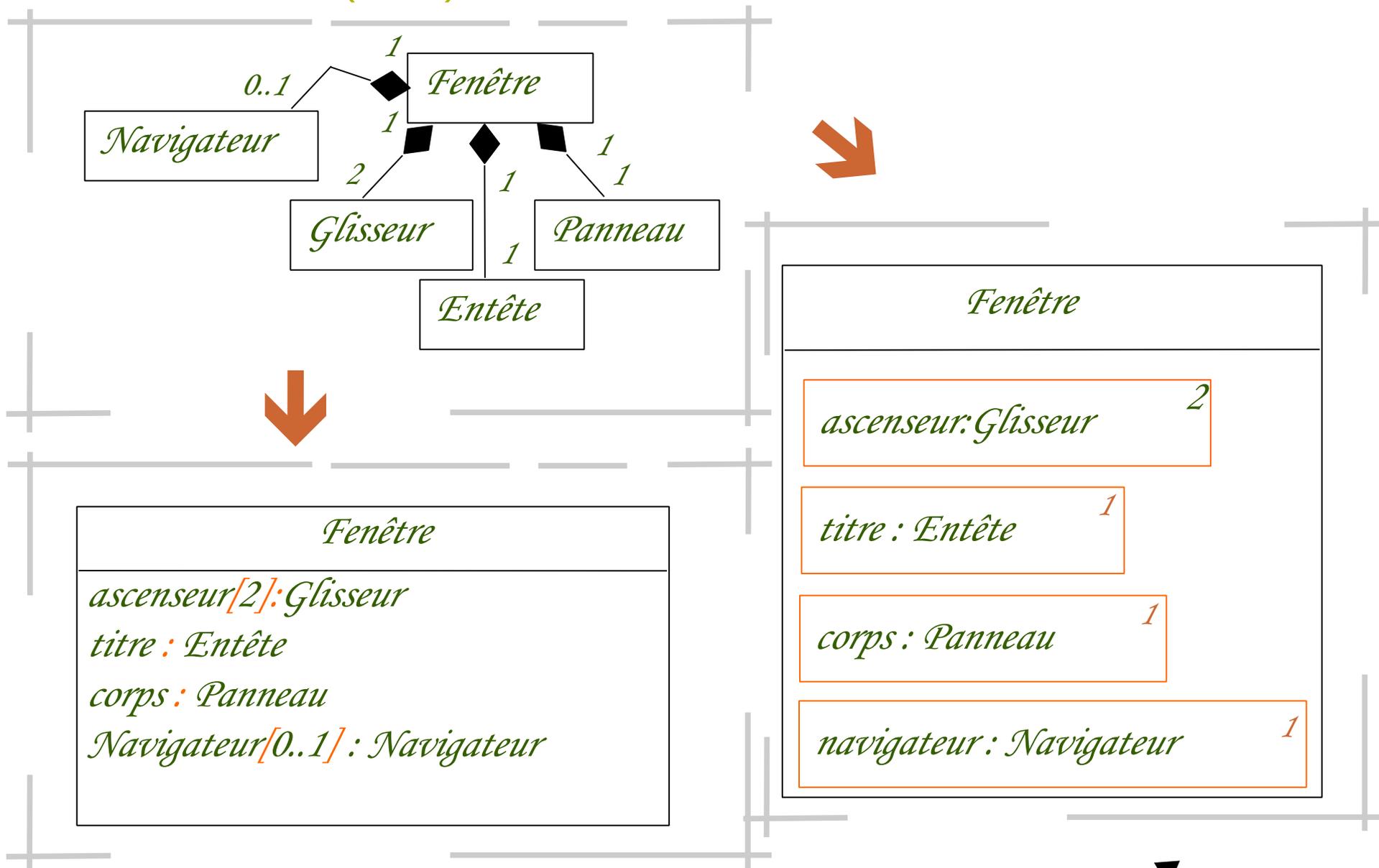


LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

COMPOSITION (suite)



LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE CLASSE

Généralités

Classes, attributs & opérations

Associations, cardinalité, qualifieur & rappels

Composition

▶ Généralisation

Contraintes

Dépendances & interfaces

Quelques conseils



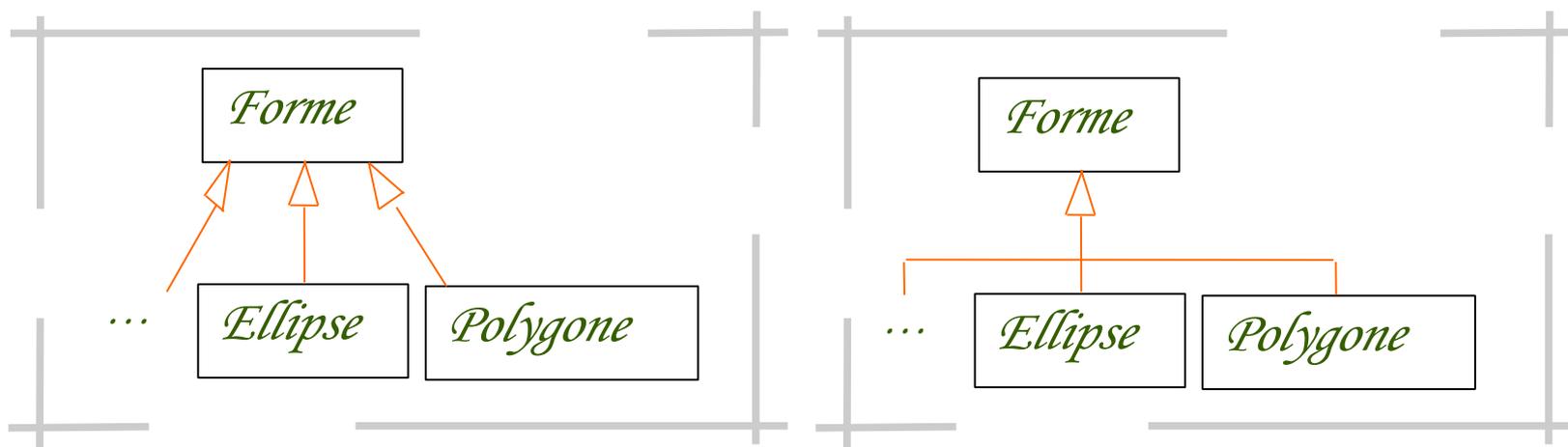
LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

□ GENERALISATION

- ✓ Transitive
- ✓ Peut être multiple



LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

□ GENERALISATION (suite)

- ✓ B **dérive** de A
- ✓ Toute instance de B **est** une instance de A pour tout traitement associé à A
- ✓ Toute instance de A **peut être substituée** par une instance de B

- ✓ B peut
 - ✓ avoir de **nouvelles propriétés** (en plus de celles héritées de A)
 - ✓ **redéfinir des propriétés** héritées de A
 - ✓ ne peut **pas** en supprimer



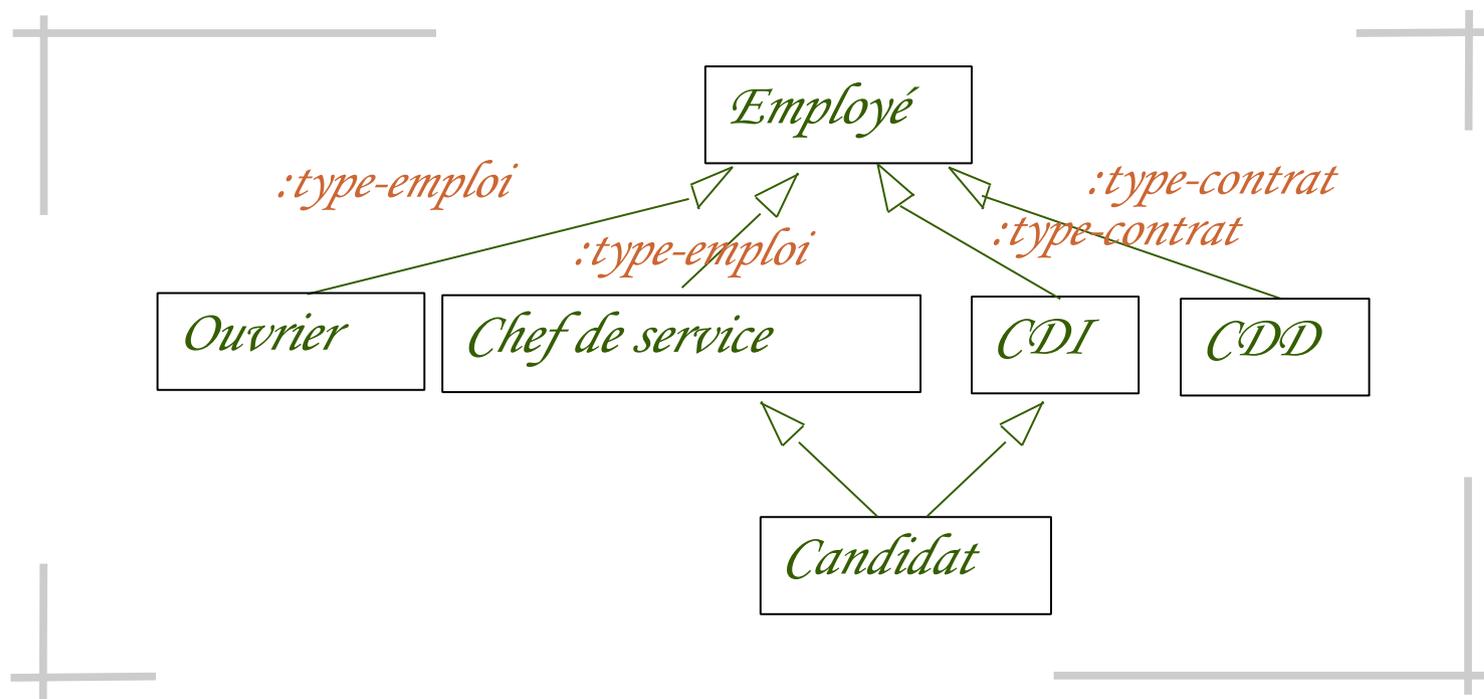
LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

■ GENERALISATION (suite)

✓ Notion de **jeu** de généralisation

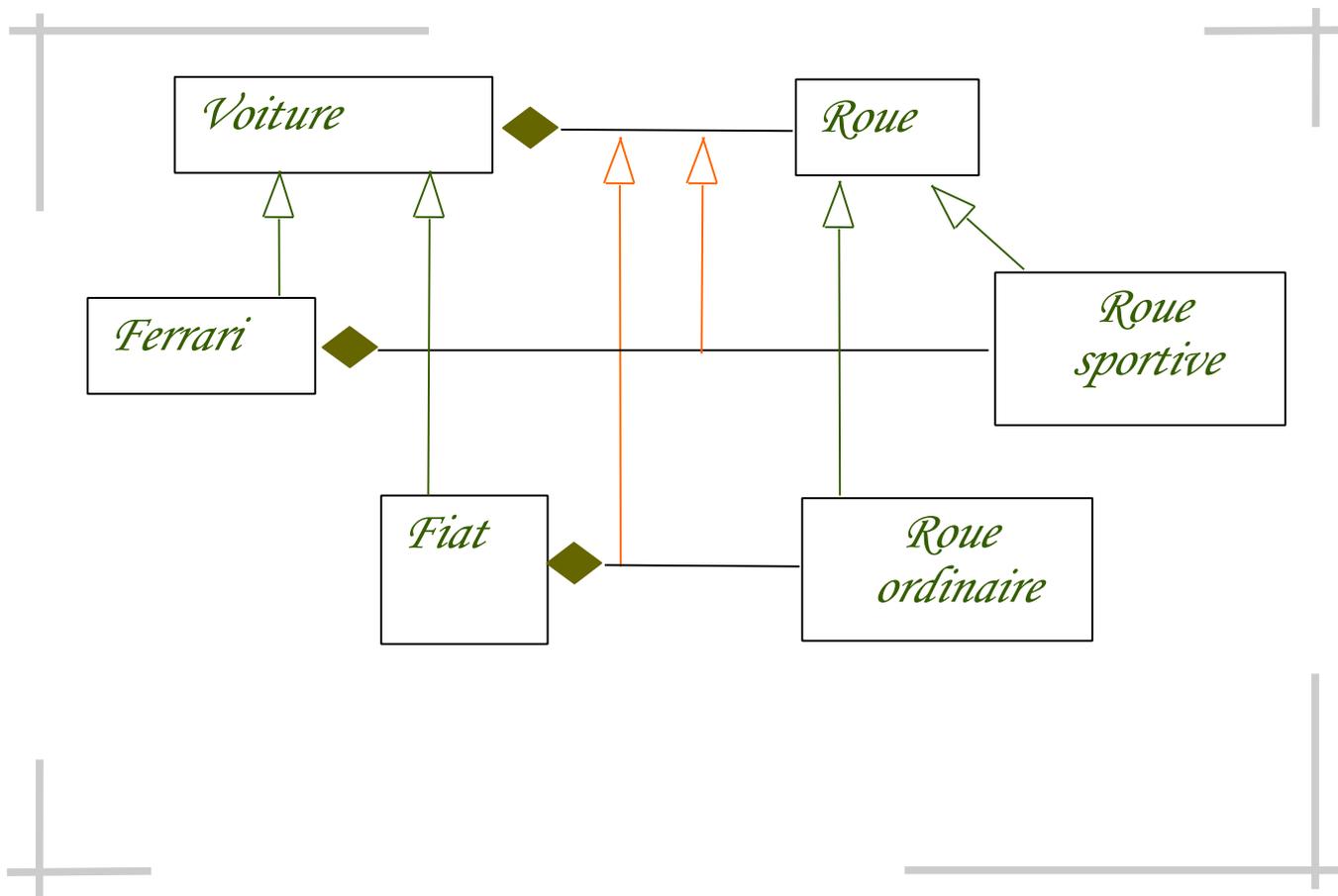


LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

■ GENERALISATION D'ASSOCIATION (suite)



LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE CLASSE

Généralités

Classes, attributs & opérations

Associations, cardinalité, qualifieur & rappels

Composition

Généralisation

▶ Contraintes

Dépendances & interfaces

Quelques conseils

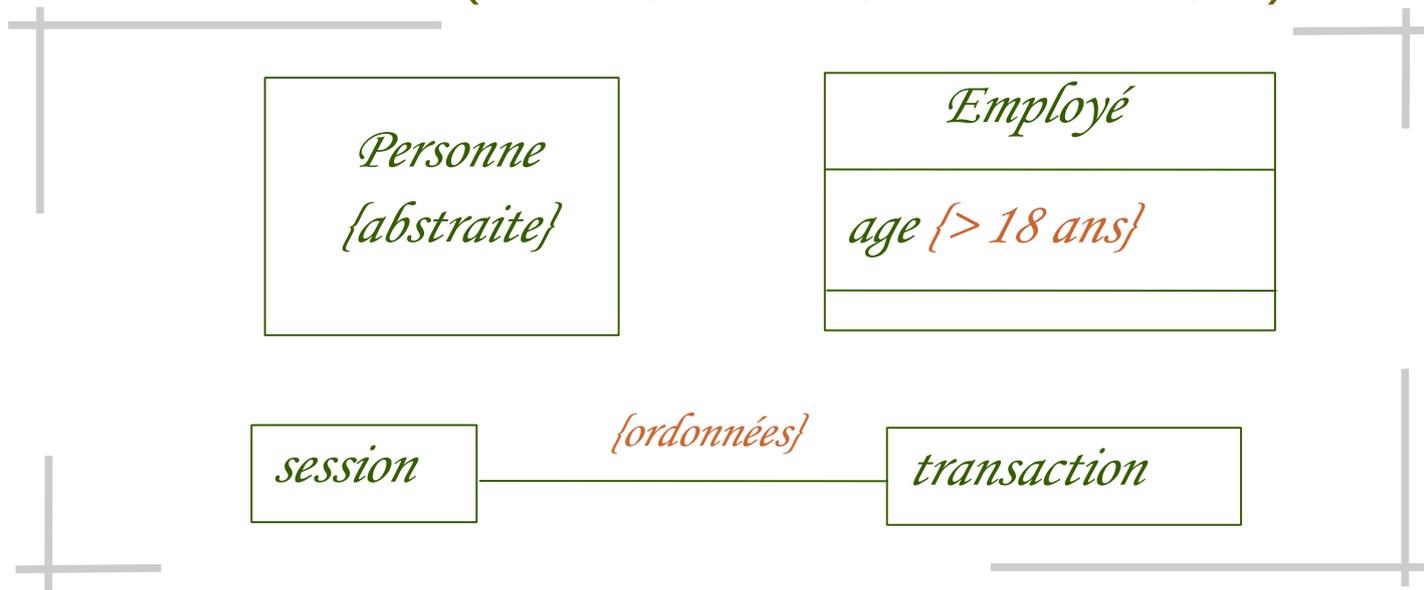


LES DIAGRAMMES STRUCTURELS

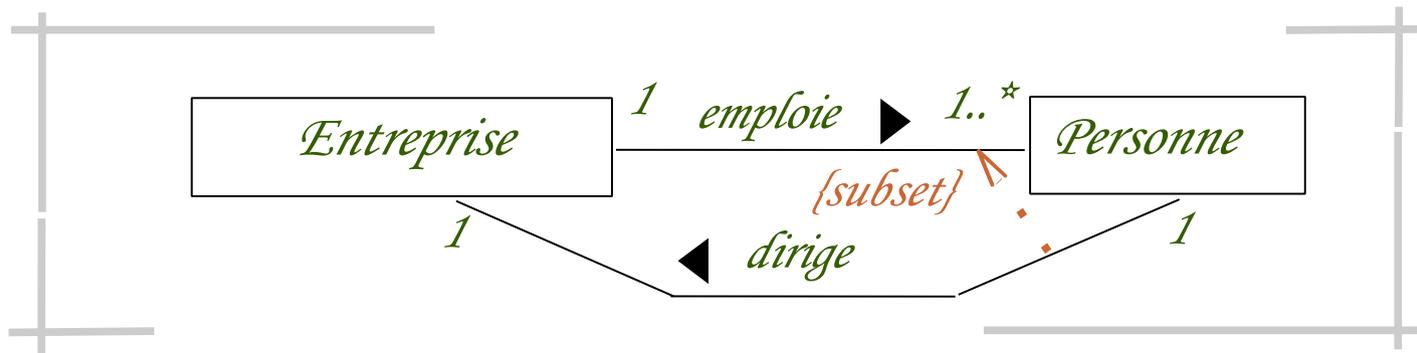
DIAGRAMME DE CLASSE

□ CONTRAINTES

- ✓ Sur un élément (classe, attribut, association, ...)



- ✓ Entre association



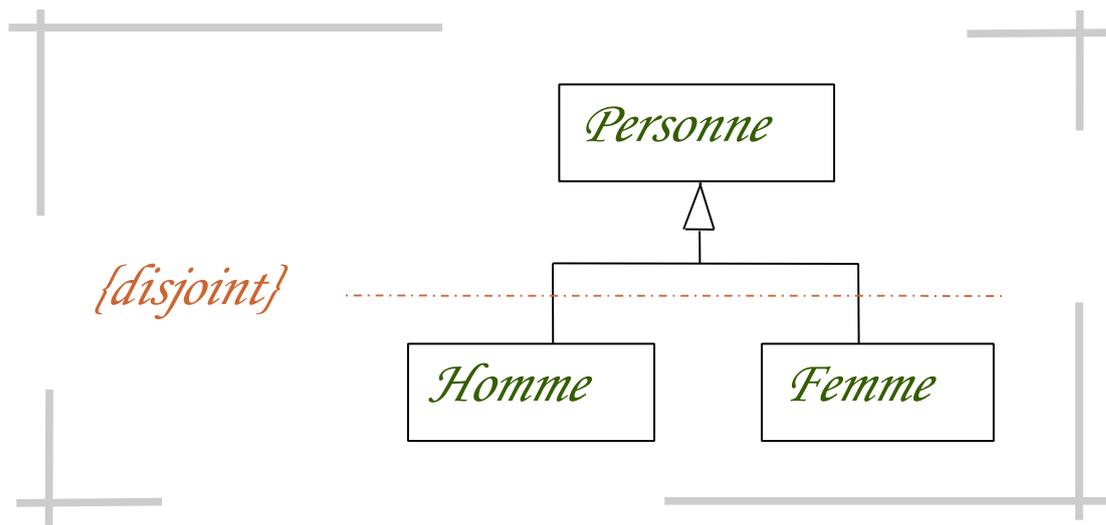
LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

□ CONTRAINTES (suite)

- ✓ Sur la généralisation



➔ {non-disjoint}, {complète}, {incomplète}, ...

LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE CLASSE

Généralités

Classes, attributs & opérations

Associations, cardinalité, qualifieur & rappels

Composition

Généralisation

Contraintes

▶ Dépendances & interfaces

Quelques conseils



LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

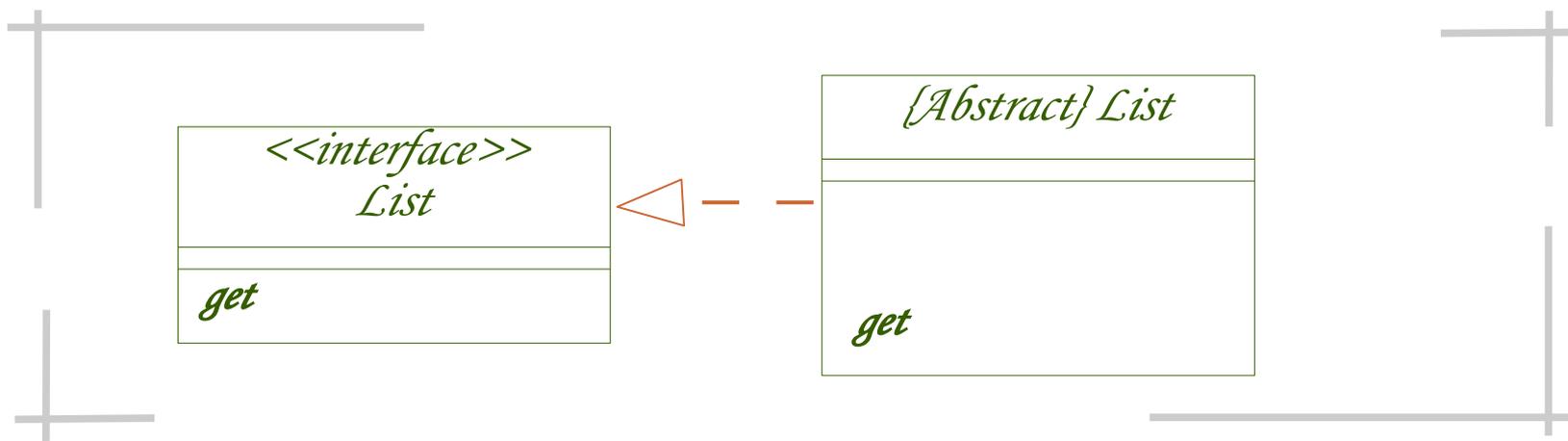
☐ DEPENDANCES



✓ La dérivation



✓ La réalisation



LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

▣ DEPENDANCES (suite)

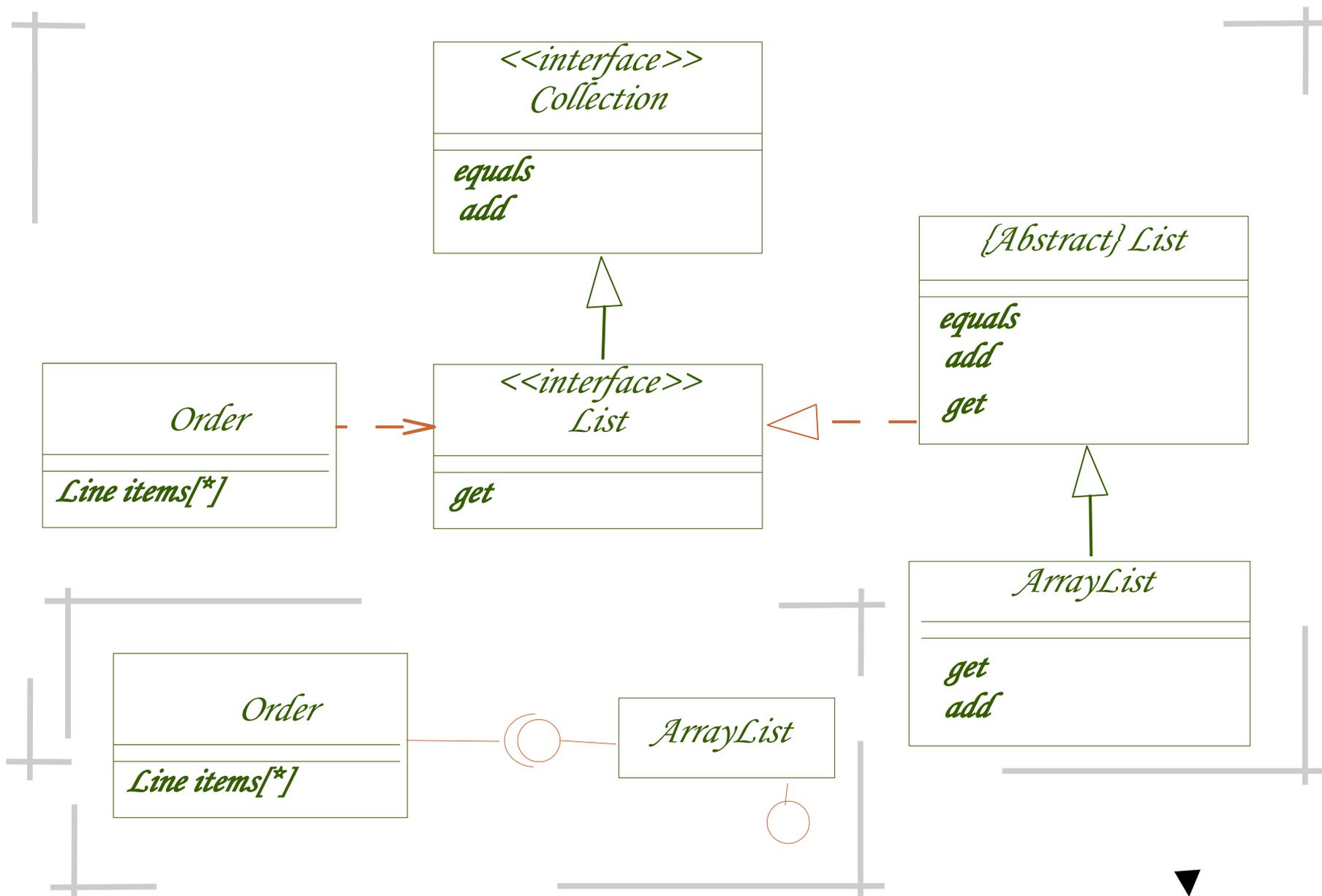
- ✓ <<call>>
- ✓ <<instanciate>>
- ✓ <<refine>>
 - ➔ *Analyse / conception*
- ✓ ...



LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE CLASSE

INTERFACES



LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE CLASSE

Généralités

Classes, attributs & opérations

Associations, cardinalité, qualifieur & rappels

Composition

Généralisation

Contraintes

Dépendances & interfaces

▶ Quelques conseils

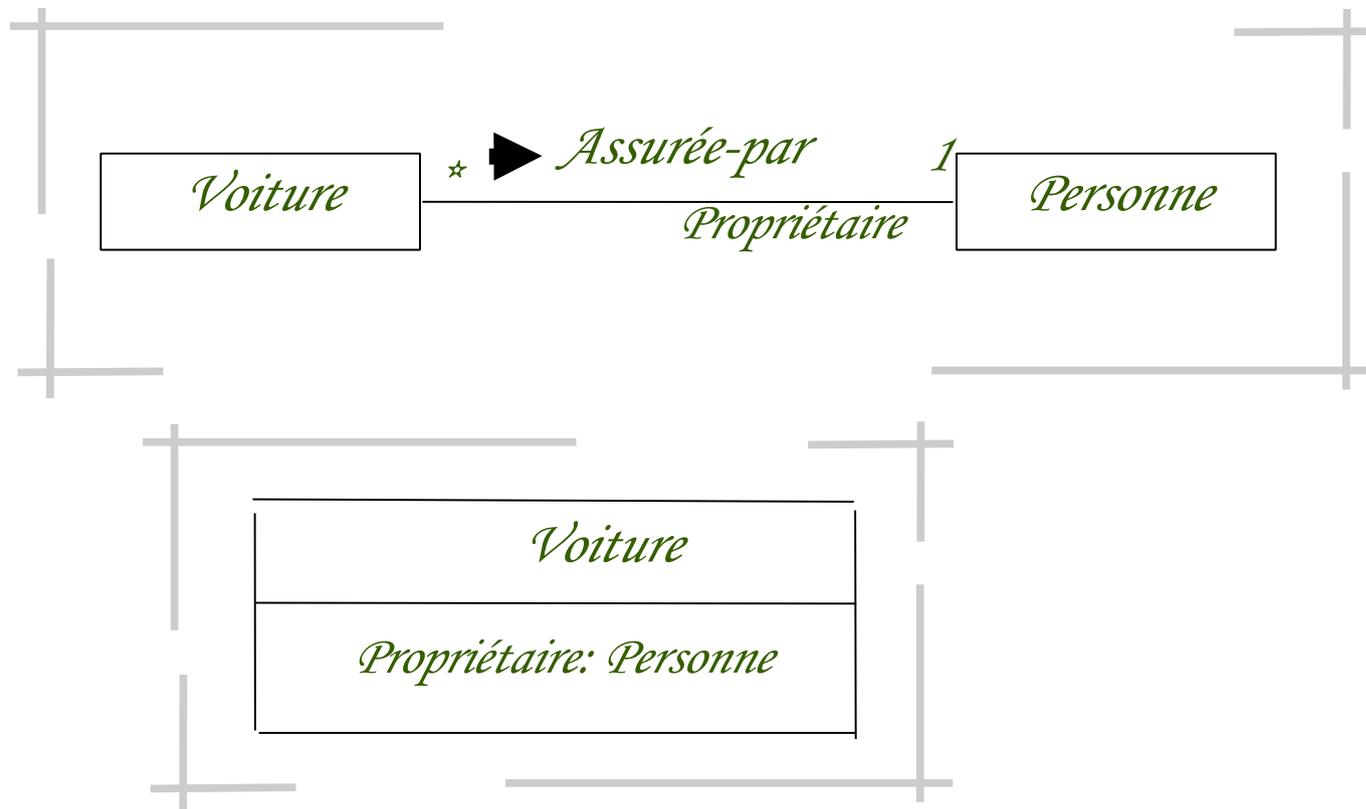


LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE CLASSE

▣ QUELQUES CONSEILS

- ✓ **Attribut vs Association**
 - ✓ Suivant la taille du schéma
 - ✓ Information à mettre en valeur
 - ✓ Etape du processus de développement



LES DIAGRAMMES STRUCTURELS

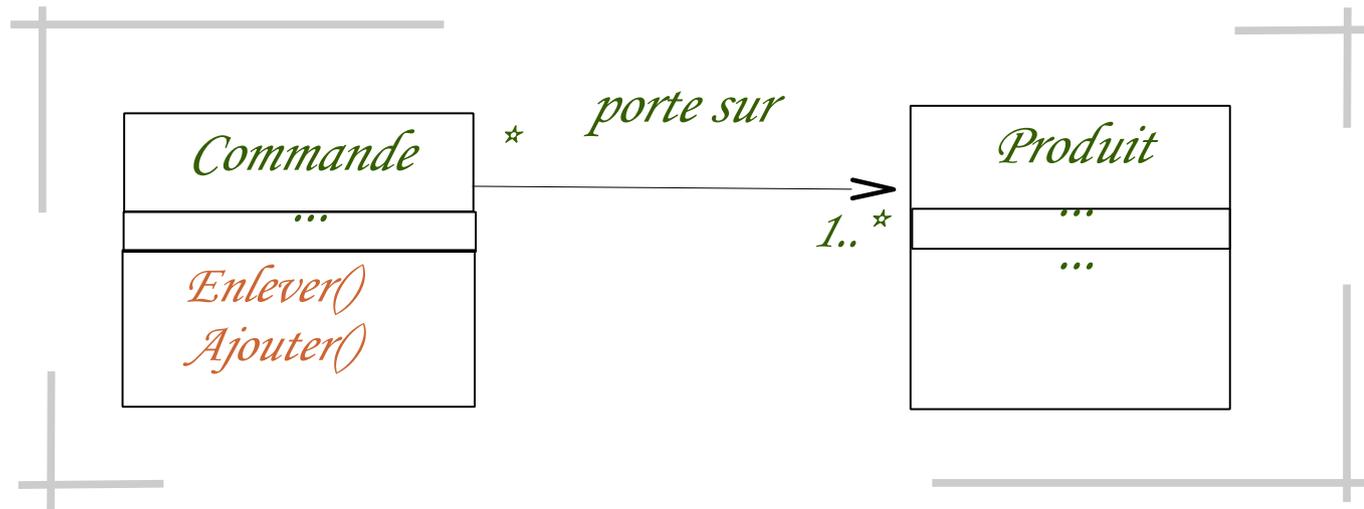
DIAGRAMME DE CLASSE

▣ QUELQUES CONSEILS (suite)



✓ opérations ajouter() & enlever() pour gérer les instances d'association (liens)

✓ Justifie & complète la navigabilité



LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE CLASSE

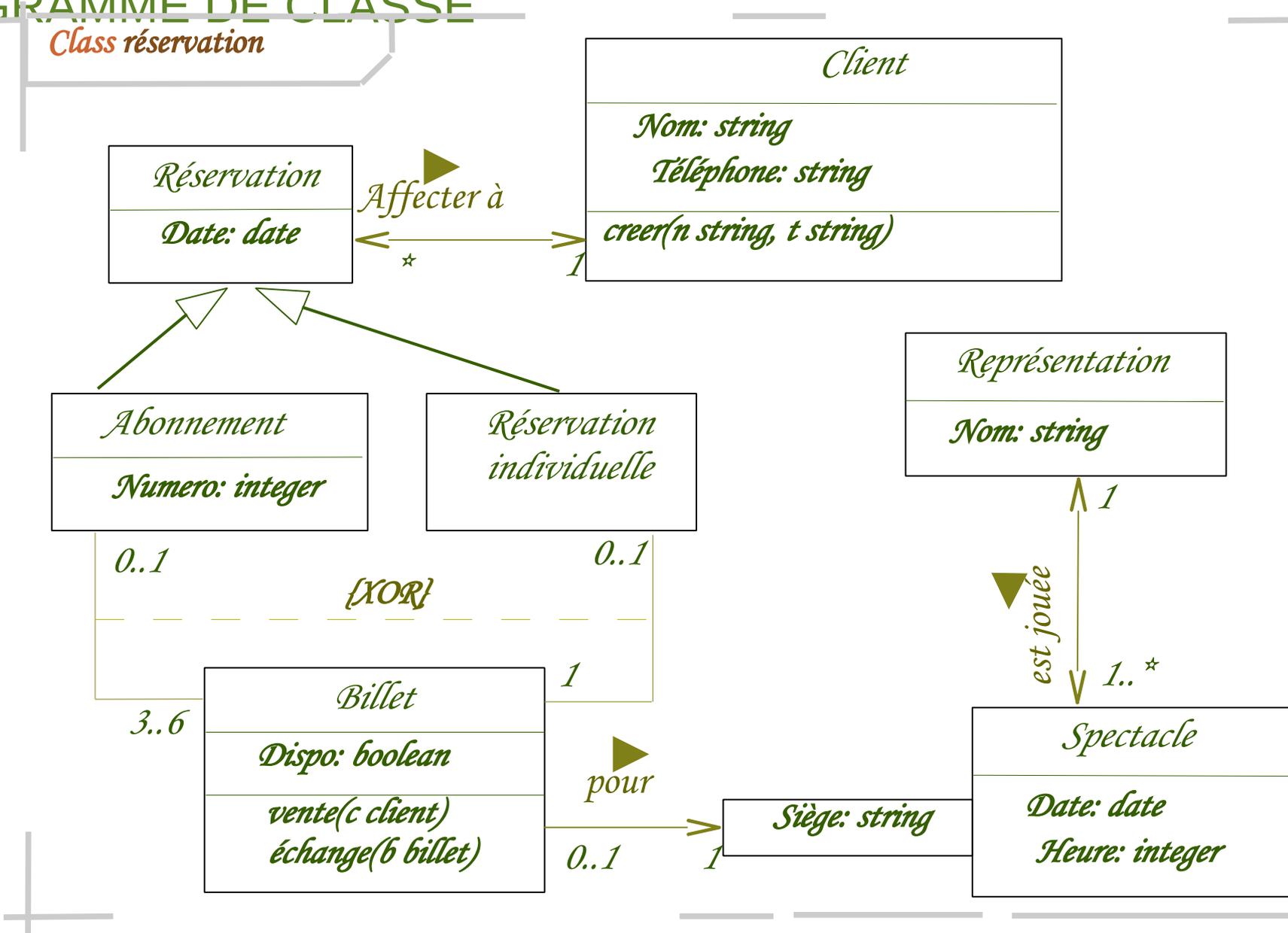
▣ QUELQUES CONSEILS (suite)

- ✓ Ne pas confondre classe & acteur
- ✓ Relation 1-1 possible mais à justifier
- ✓ Une/des classe application
 - ✓ Pour spécifier les listes d'objet accessibles
 - ✓ Mettre en relief les interfaces du systèmes
 - ✓ Etre cohérent avec les diagrammes d'interaction
- ✓ Un diagramme de classe doit
 - ✓ tenir dans 1 seule page
 - ✓ Être clair & lisible
 - ✓ Être **documenté**
- ✓ Une classe doit
 - ✓ Correspondre à une seule responsabilité
 - ✓ Représenter une abstraction pertinente
 - ✓ Etre bien nommée
 - ✓ Etre complètement décrite à un endroit du document



LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE CLASSE



LES DIAGRAMMES STRUCTURELS

- ❑ Introduction
- ❑ Diagramme de cas d'utilisation
- ❑ Diagramme de classe
- ▶ ❑ Diagramme d'architecture
- ❑ Diagramme d'instance
- ❑ Diagramme de composant
- ❑ Diagramme de déploiement

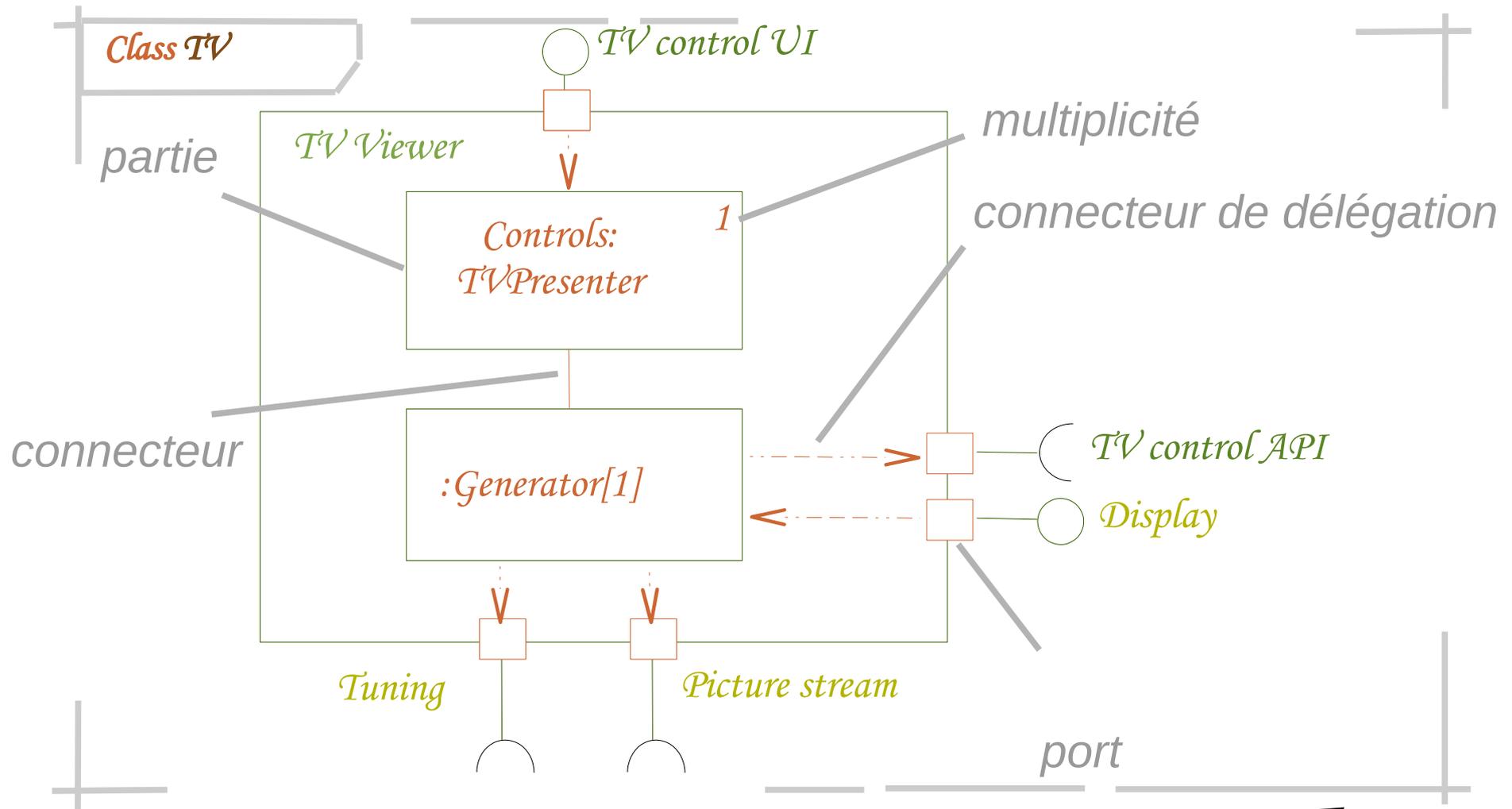
LES DIAGRAMMES

STRUCTURELS

DIAGRAMME D'ARCHITECTURE

UN EXEMPLE

- ✓ Nouveau dans la notation UML
- ✓ Pour montrer la structuration des classes
- ✓ Pas de frontière stricte avec le diagramme de classe



LES DIAGRAMMES STRUCTURELS

- ❑ Introduction
- ❑ Diagramme de cas d'utilisation
- ❑ Diagramme de classe
- ❑ Diagramme d'architecture
- ▶ ❑ Diagramme d'instance
- ❑ Diagramme de composant
- ❑ Diagramme de déploiement

LES DIAGRAMMES

STRUCTURELS

DIAGRAMME D'INSTANCE

□ GENERALITES

- ✓ Aussi appelé diagramme d'objet
- ✓ Pour montrer un contexte avant / après interaction
- ✓ Pour montrer un comportement au travers d'une séquence de diagrammes d'instance
- ✓ Pour faciliter la compréhension de structures de données complexes
- ✓ Image détaillée de l'état du système à un instant donné



LES DIAGRAMMES

STRUCTURELS

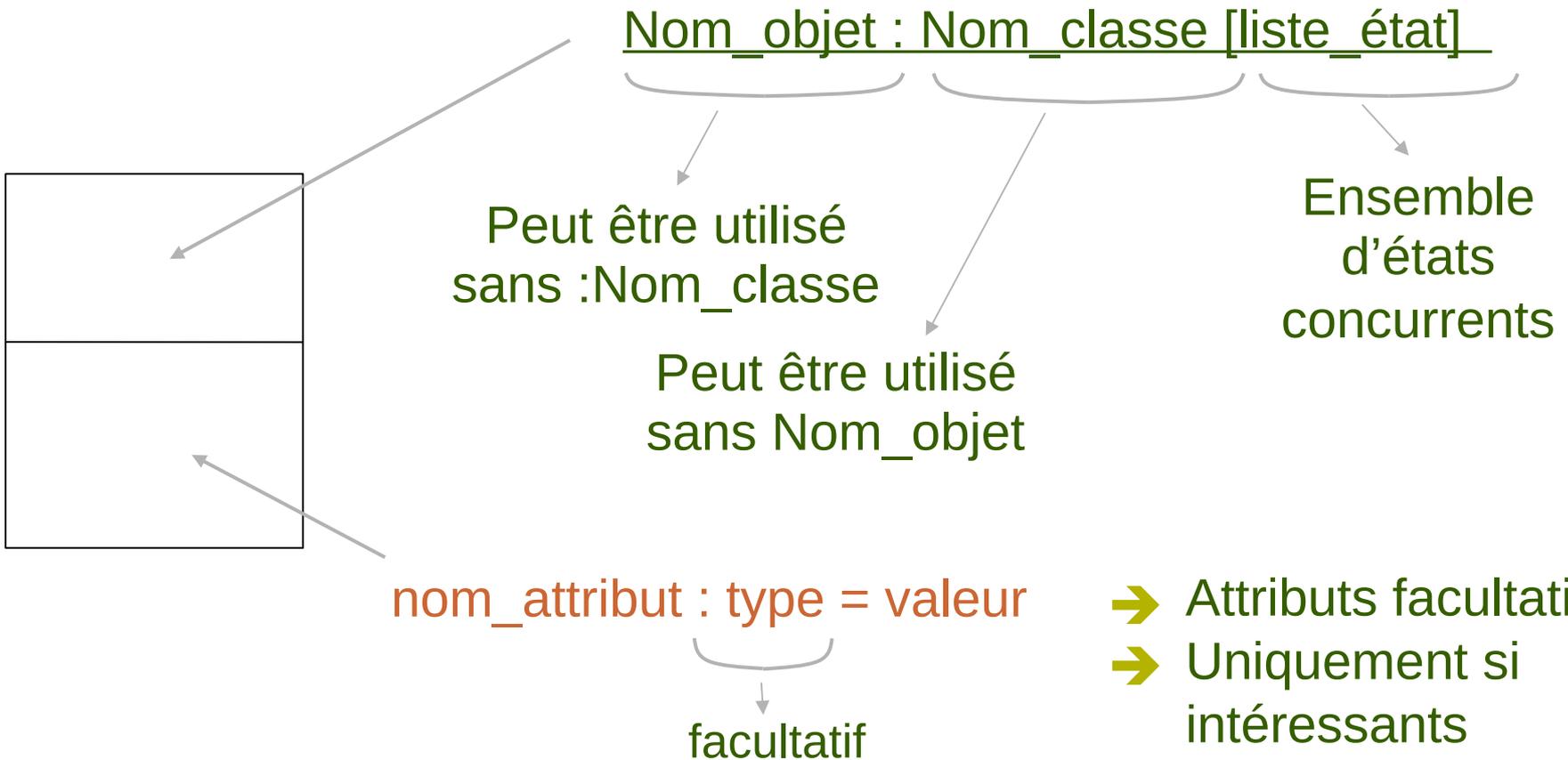
DIAGRAMME D'INSTANCE

▣ OBJET

✓ Instance particulière d'une classe

➔ Identité

➔ Valeurs d'attribut



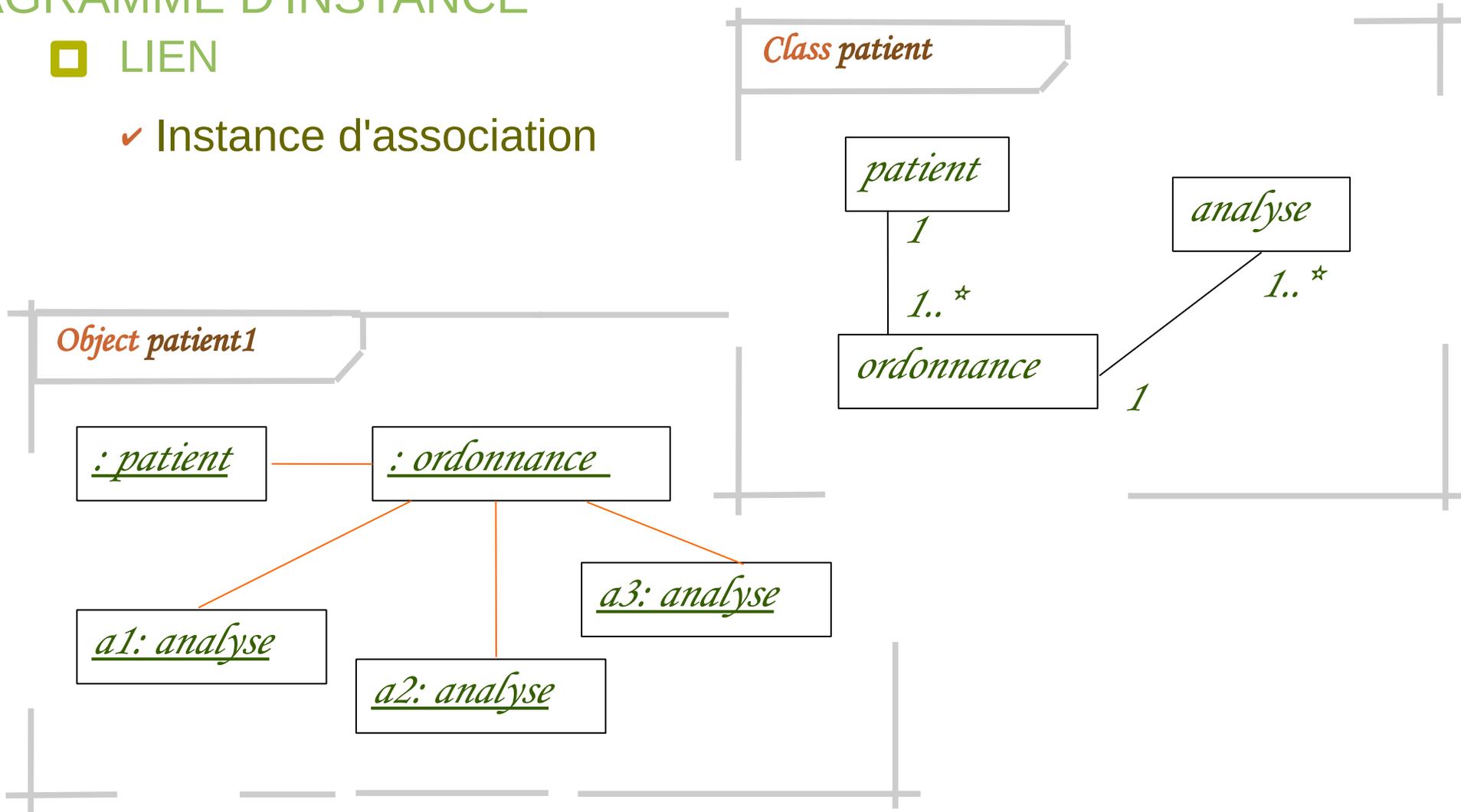
LES DIAGRAMMES

STRUCTURELS

DIAGRAMME D'INSTANCE

□ LIEN

✓ Instance d'association

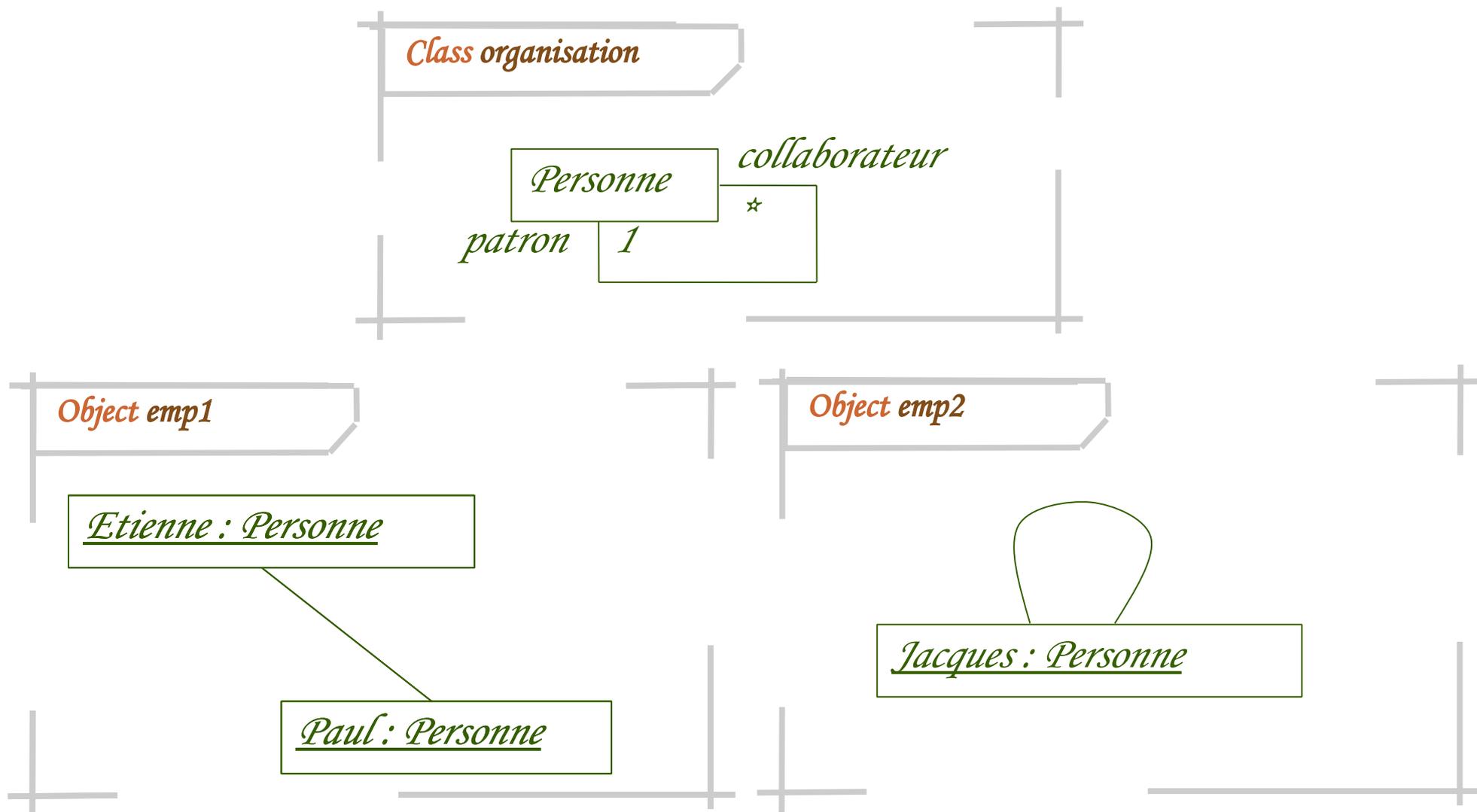


LES DIAGRAMMES

STRUCTURELS

DIAGRAMME D'INSTANCE

□ AUTRES EXEMPLES



LES DIAGRAMMES

STRUCTURELS

DIAGRAMME D'INSTANCE

▣ OBJETS COMPOSITES

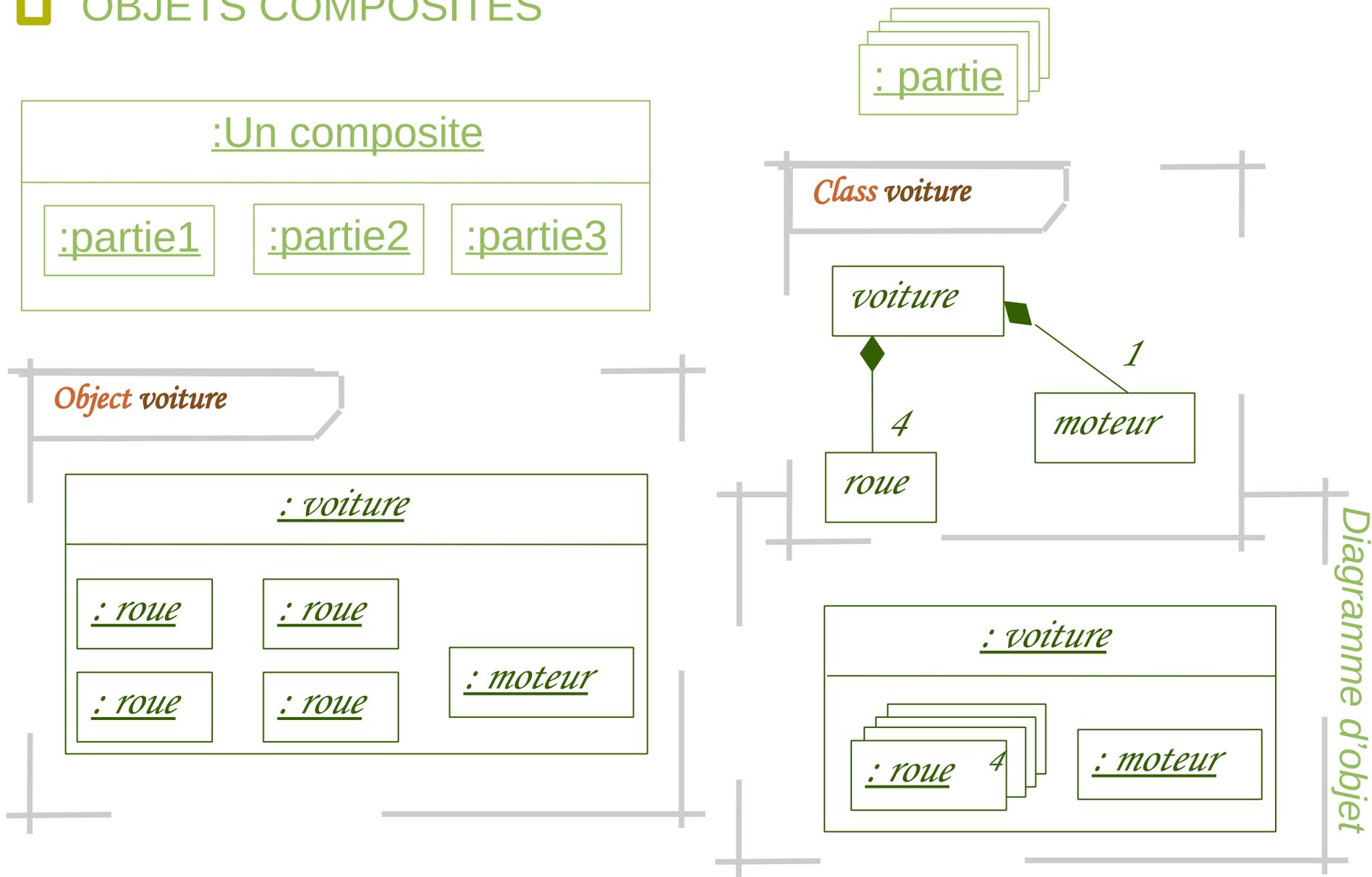


Diagramme d'objet

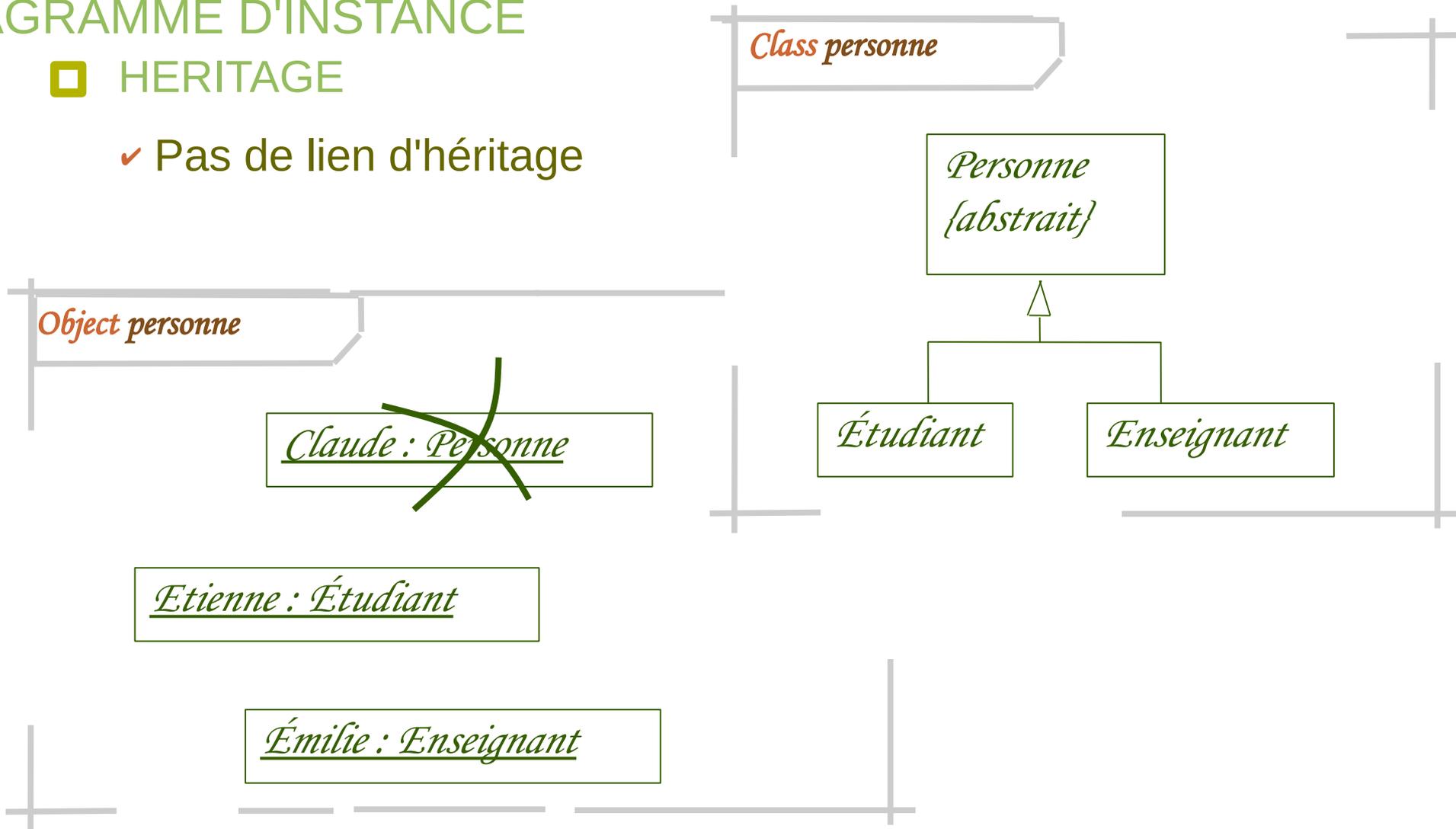
LES DIAGRAMMES

STRUCTURELS

DIAGRAMME D'INSTANCE

□ HERITAGE

✓ Pas de lien d'héritage



LES DIAGRAMMES STRUCTURELS

- Introduction
- Diagramme de cas d'utilisation
- Diagramme de classe
- Diagramme d'architecture
- Diagramme d'instance
- ▶ □ Diagramme de composant
- Diagramme de déploiement

LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE COMPOSANT

□ GENERALITES

- ✓ Gestion de la complexité
- ✓ Mise en évidence
 - ✓ des unités logicielles
 - ✓ des dépendances entre unités
 - ✓ des interfaces de chaque unité

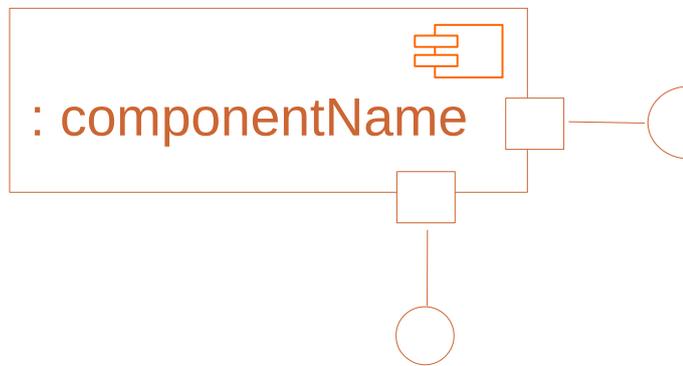


LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE COMPOSANT

❑ COMPOSANT

- ✓ Distinction classe structurée / composant floue



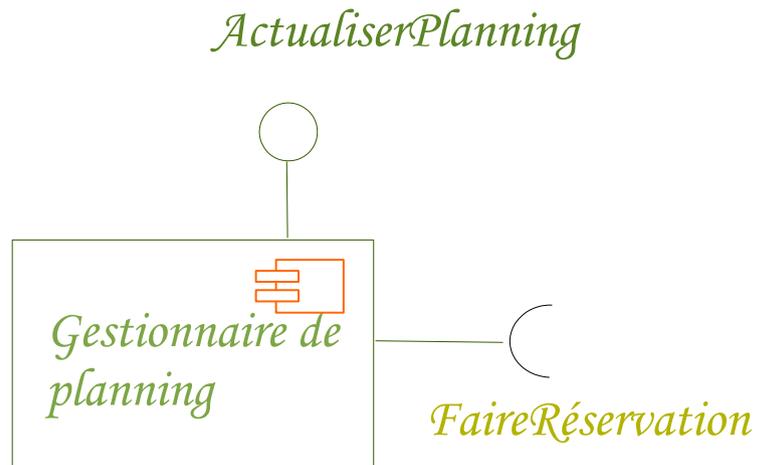
- ✓ Chaque composant doit avoir
 - ✓ Un minimum de communication avec les autres composants
 - ✓ Une interface claire et précise



LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE COMPOSANT

COMPOSANT (suite)



LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE COMPOSANT

□ COMPOSANT & CAS D'UTILISATION

✓ Un cas d'utilisation ne peut être placé que dans un seul composant

✓ Décomposition des cas d'utilisation

✓ Utilisation de la relation d'inclusion



✓ Cohérence cas d'utilisation / composants

✓ Cohérence acteurs / composants

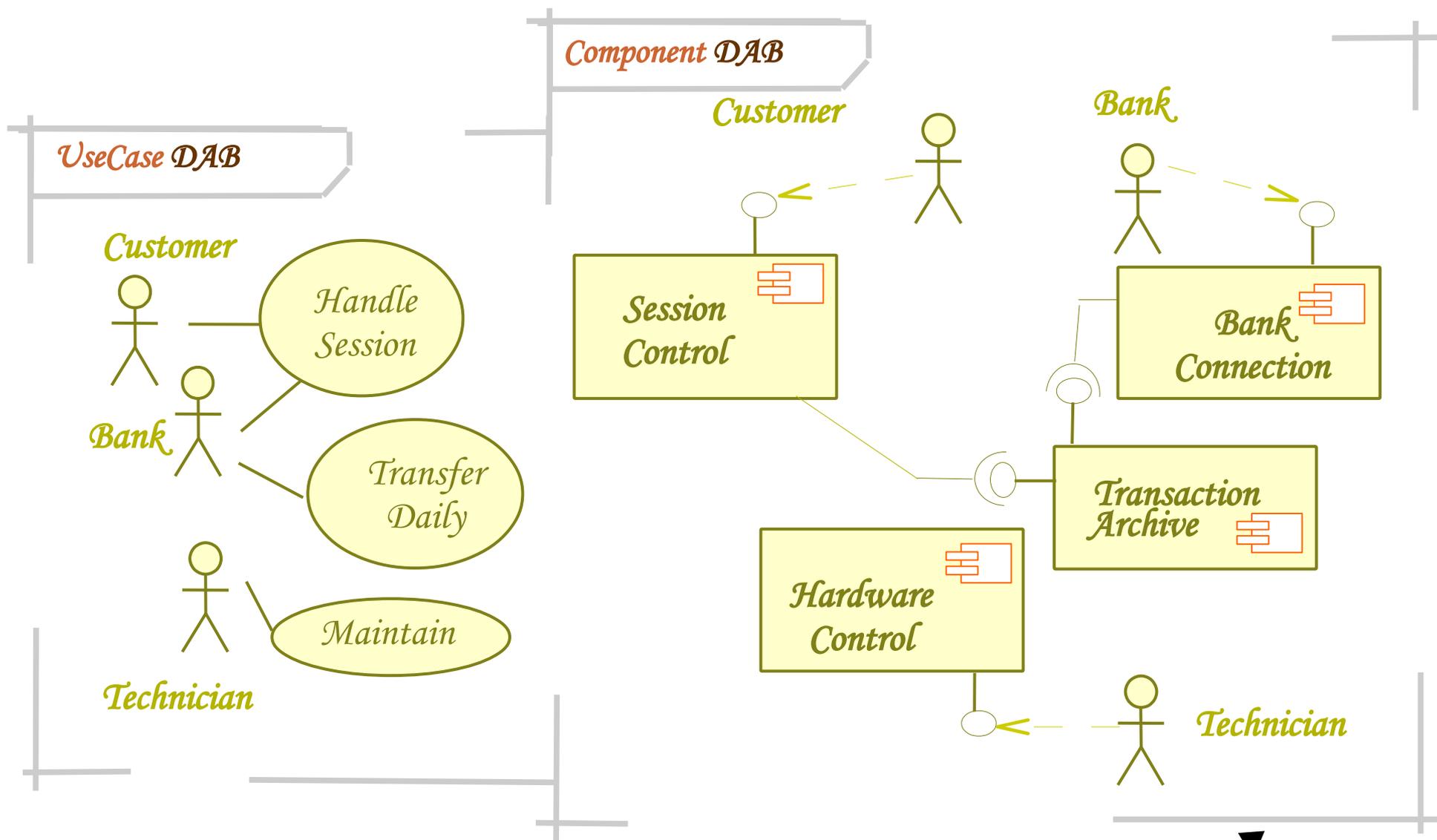


LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE COMPOSANT

COMPOSANT & CAS D'UTILISATION (suite)



LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE COMPOSANT

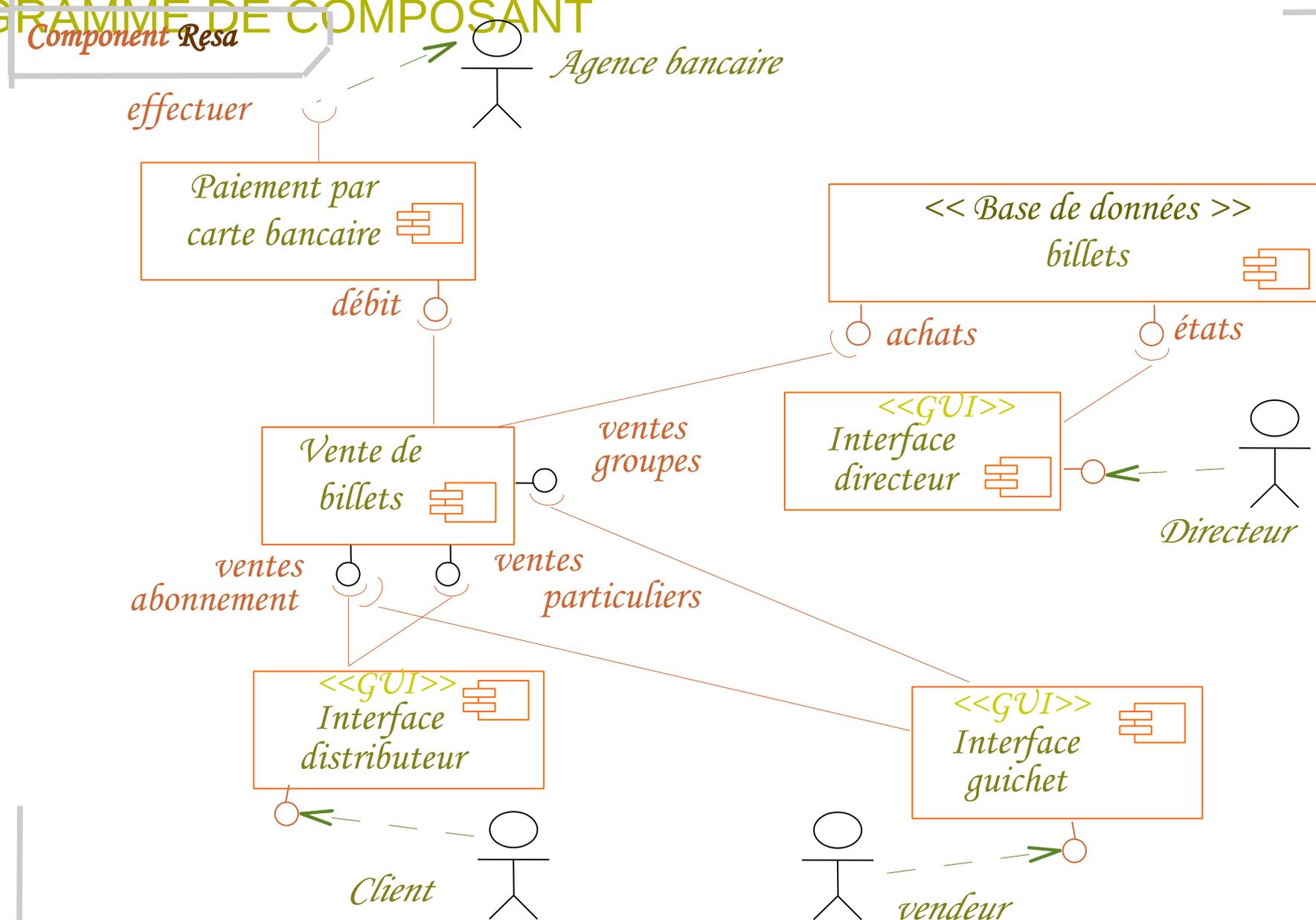
❑ COMPOSANT & CLASSE

- ✓ Toute les classes doivent être placées dans les composants
 - ✓ Une classe ne doit apparaître qu'à un seul endroit
 - ✓ Les dépendances entre composants doivent supporter
 - ✓ les associations entre classes (diag. de classe)
 - ✓ l'envoi de messages (diag. d'interaction)
- ✓ Les interfaces doivent rassembler
 - ✓ la liste des opérations publiques des classes du composants
 - ✓ éventuellement la liste des attributs publics



LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE COMPOSANT



LES DIAGRAMMES STRUCTURELS

- Introduction
- Diagramme de cas d'utilisation
- Diagramme de classe
- Diagramme d'architecture
- Diagramme d'instance
- Diagramme de composant
- ▶ □ Diagramme de déploiement

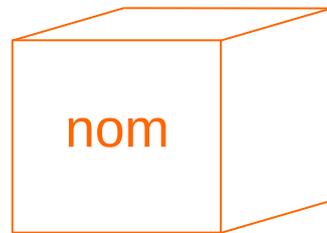
LES DIAGRAMMES

STRUCTURELS

DIAGRAMME DE DEPLOIEMENT

□ NOEUD & CONNEXION

- ✓ Pour montrer :
 - ✓ Disposition physique des différents matériels
 - ✓ Répartition des composants sur les matériels
 - ✓ Relations entre composants & matériel



nœud



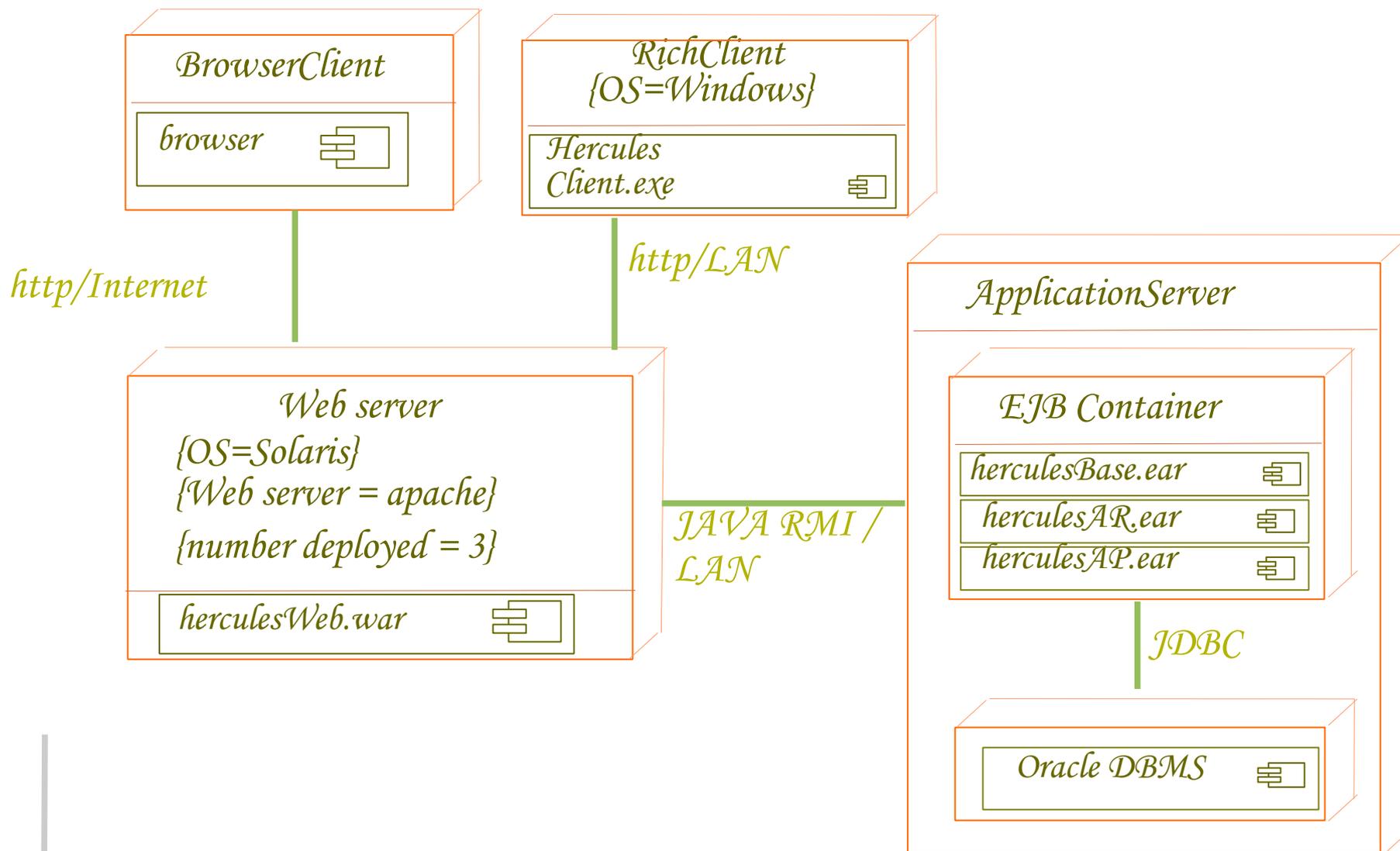
connexion



LES DIAGRAMMES STRUCTURELS

DIAGRAMME DE DEPLOIEMENT

Deployment hercule



P L A N

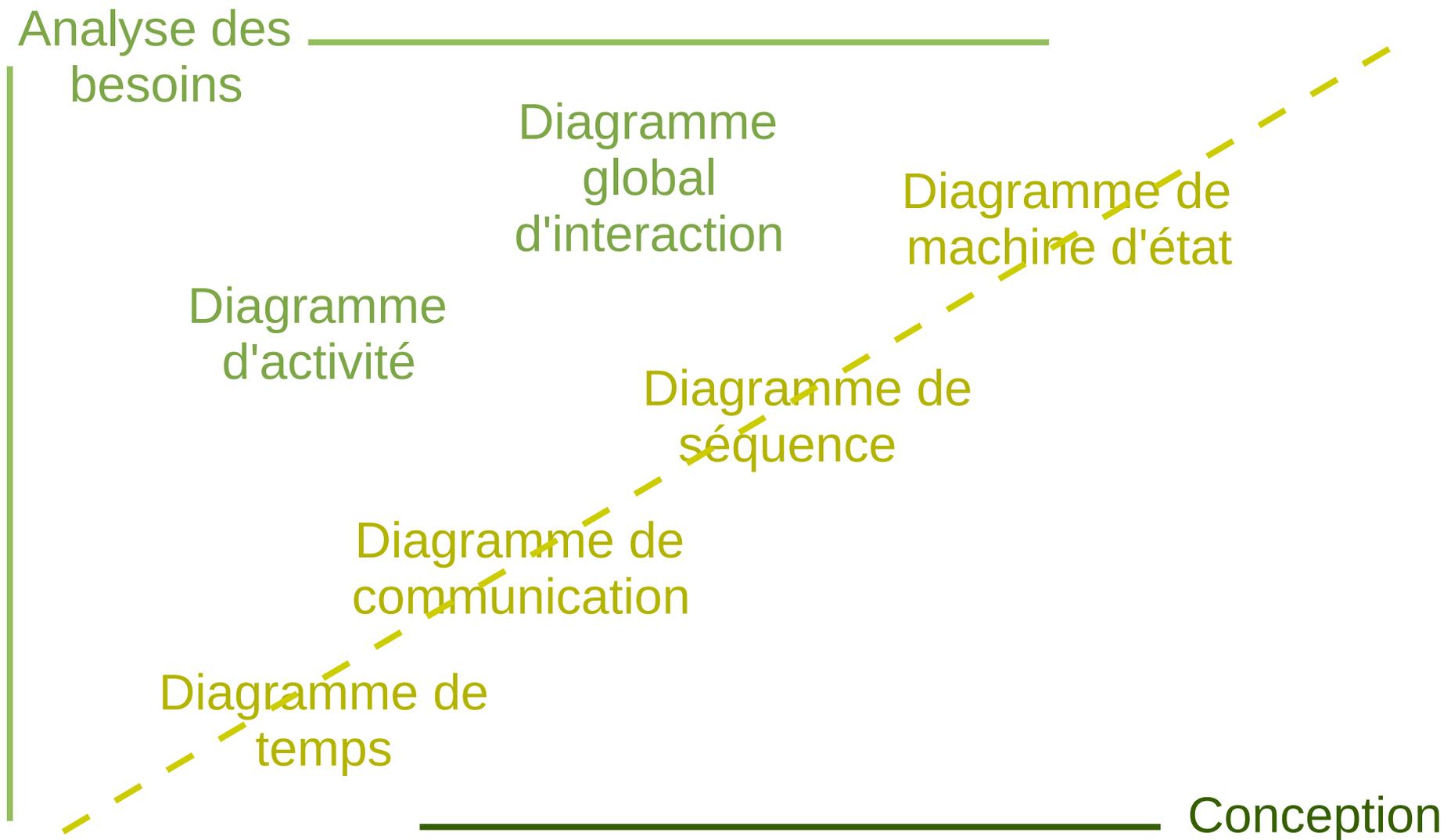
- Introduction
- Les diagrammes structurels
- ▶ ■ Les diagrammes comportementaux
- Le diagramme de paquetage
- Conclusion

LES DIAGRAMMES COMPORTEMENTAUX

- ▶ Introduction
- Diagramme d'activité
- Diagrammes d'interaction
 - Diagramme de séquence
 - Diagramme de communication
 - Diagramme global d'interaction
 - Diagramme de temps
- Diagramme de machine d'état

LES DIAGRAMMES COMPORTEMENTAUX

INTRODUCTION



LES DIAGRAMMES COMPORTEMENTAUX

- Introduction
- ▶ □ Diagramme d'activité
- Diagrammes d'interaction
 - Diagramme de séquence
 - Diagramme de communication
 - Diagramme global d'interaction
 - Diagramme de temps
- Diagramme de machine d'état

LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME D'ACTIVITE

□ UN APERCU DU DIAGRAMME D'ACTIVITE

- ✓ Représentation peu détaillée
- ✓ Flot d'activité / objets
- ✓ Inspirée des réseaux de Pétri
- ✓ Recommandée pour
 - ➔ Nature procédurale
 - ➔ Exécutions en parallèle
- ✓ Permet de représenter plusieurs scénarios à la fois

- ✓ Représentation des processus métier
- ✓ Compléments aux cas d'utilisation (enchaînements logiques)



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME D'ACTIVITE

□ ACTION / ACTIVITE

- ✓ permet de réaliser un but précis (modification de l'état du système ou retour de valeur)
- ✓ atomique
- ✓ ne peut être interrompue

Nom, [paramètres en entrée], [paramètres en sortie],
[préconditions], [postcondition]

□ ACTIONS PARTICULIERES

début ●

fin ⊙

Flot final exceptionnel

□ ACTIONS PRIMITIVES



→ Jeu prédéfini

accept call, accept event, destroy Object, ...



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME D'ACTIVITE

▣ TRANSITION / FLOT DE CONTROLE

COMPLETION : lorsque l'action source est terminée, le flot de contrôle passe via la transition à l'action but

→ Pas d'événement sur la transition

EXCEPTION : action source est interrompue, le flot de contrôle passe via la transition à l'action but

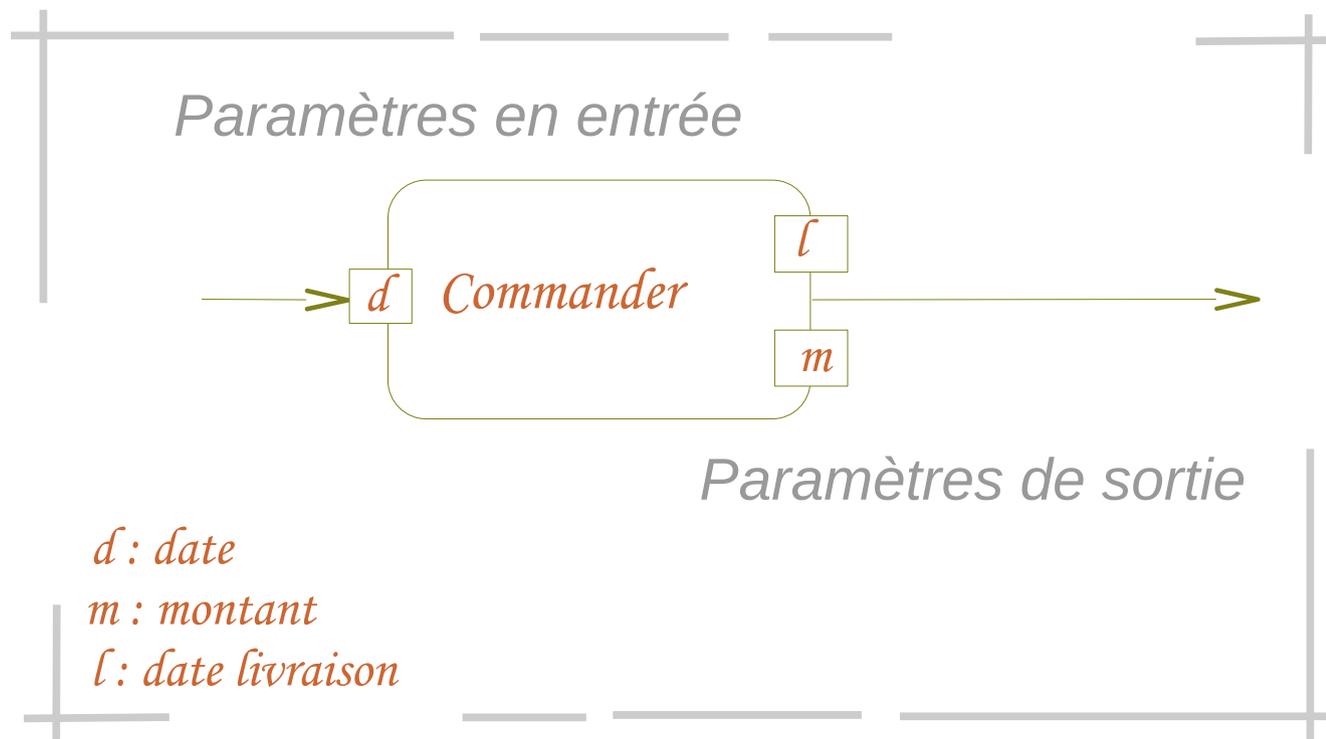
→ Événement obligatoire sur la transition



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME D'ACTIVITE

▣ PARAMETRE *pin*

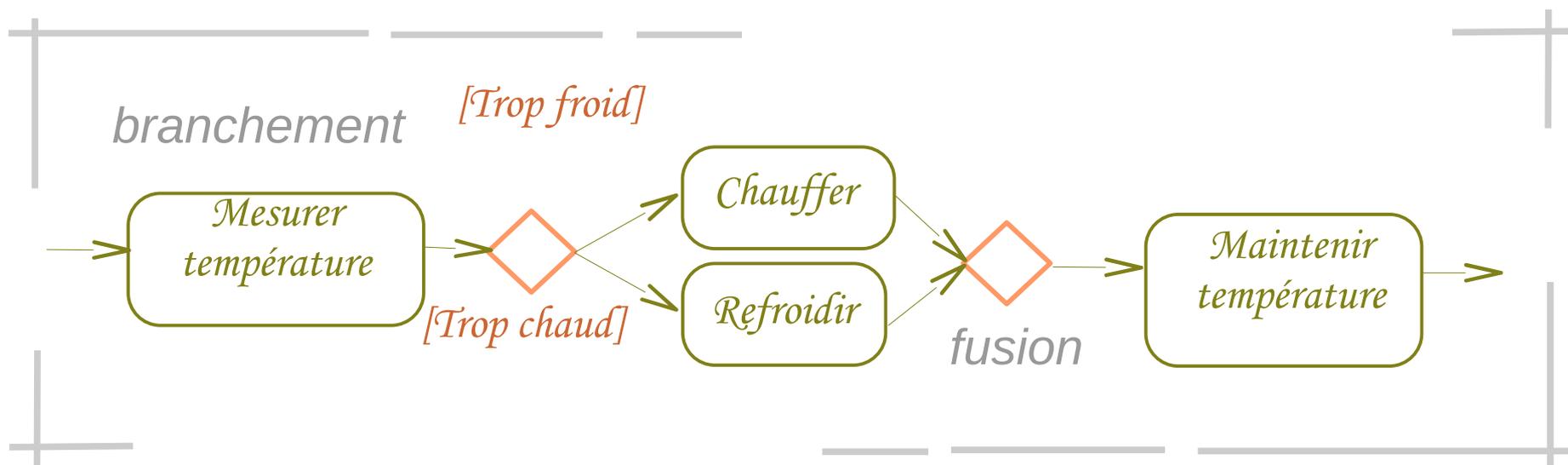


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME D'ACTIVITE

▣ BRANCHEMENT / FUSION

- ✓ Exécutions alternatives
- ✓ Gardes exclusives

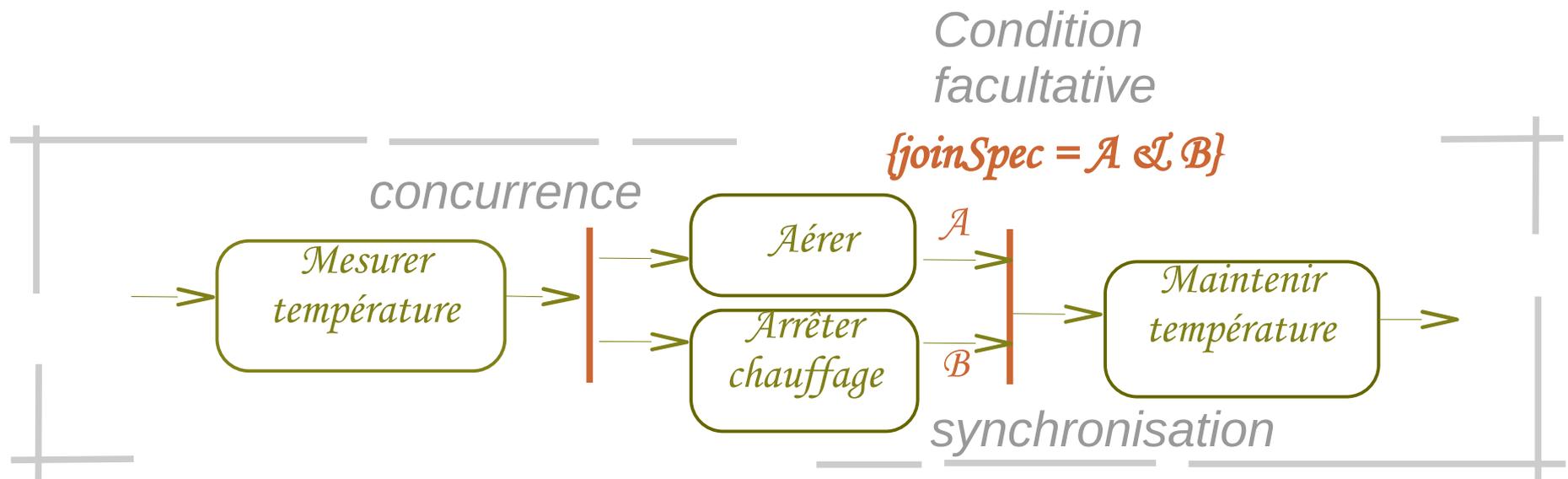


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME D'ACTIVITE

□ CONCURRENCE / SYNCHRONISATION

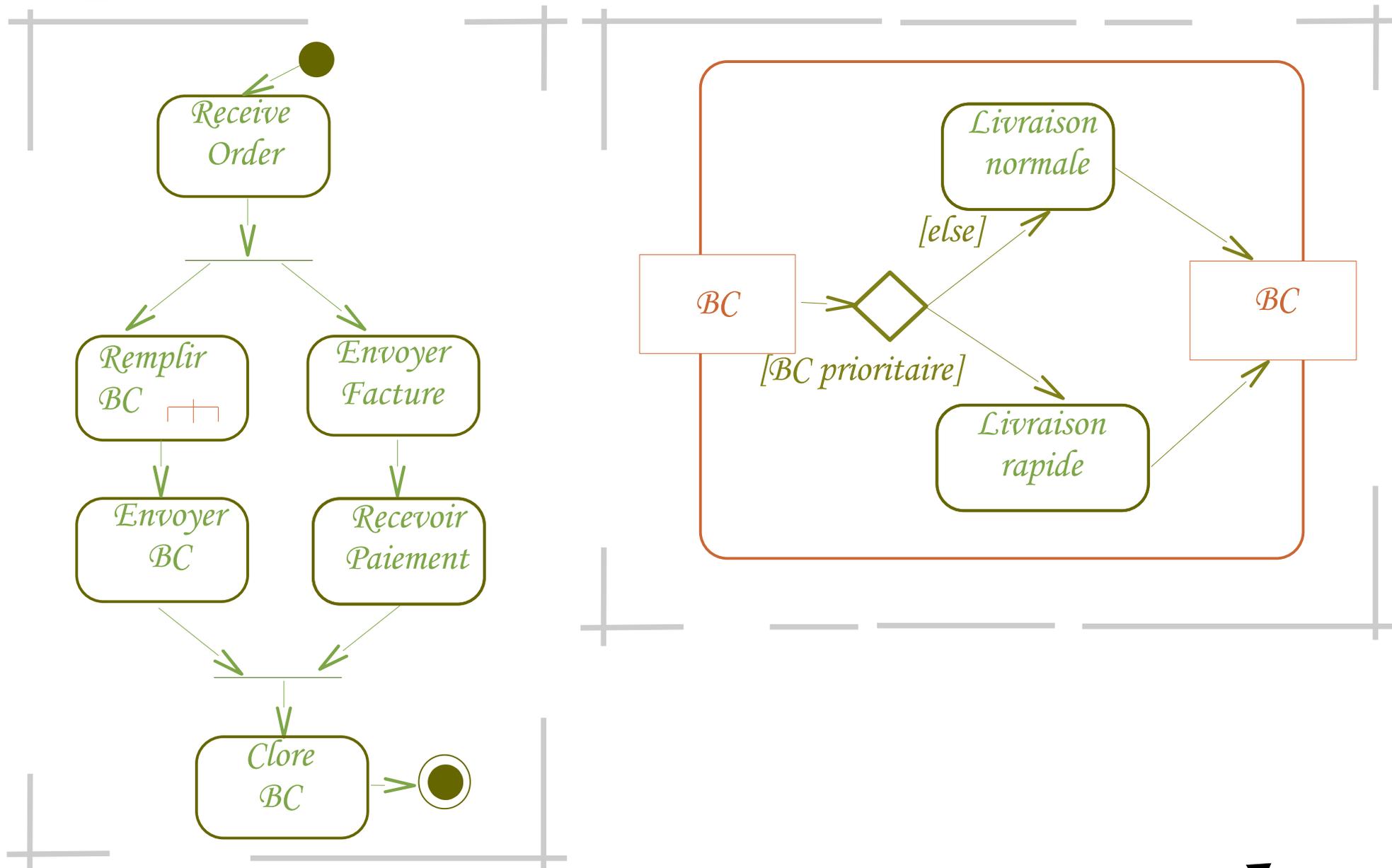
- ✓ Chaque transition est un flôt de contrôle indépendant
- ✓ Exécutions concurrentes



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME D'ACTIVITE

IMBRICATIONS ENTRE ACTIONS

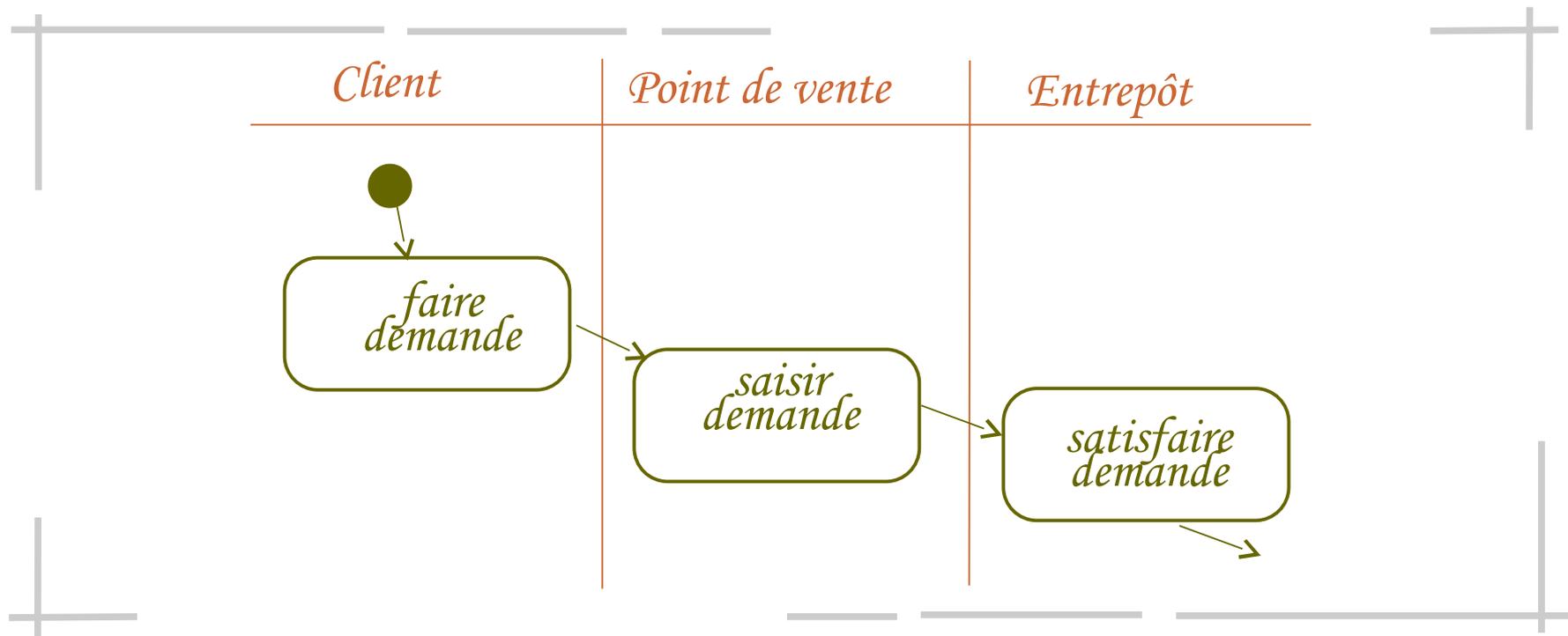


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME D'ACTIVITE

▣ RESPONSABILITES (ou partitions ou ligne de visibilité)

- ✓ Indiquer qui (?) est responsable de chaque action
- ✓ Une action n'appartient qu'à un seul responsable



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME D'ACTIVITE

▣ RESPONSABILITES (suite)

- ✓ Nécessite de bien préciser
 - ➔ Le contexte d'utilisation
 - ➔ Ce que l'on entend par responsabilité
 - ➔ *Acteur / ressource / objet / composants /*

- ✓ Recommandé pour la description de processus métiers (workflow) distribués

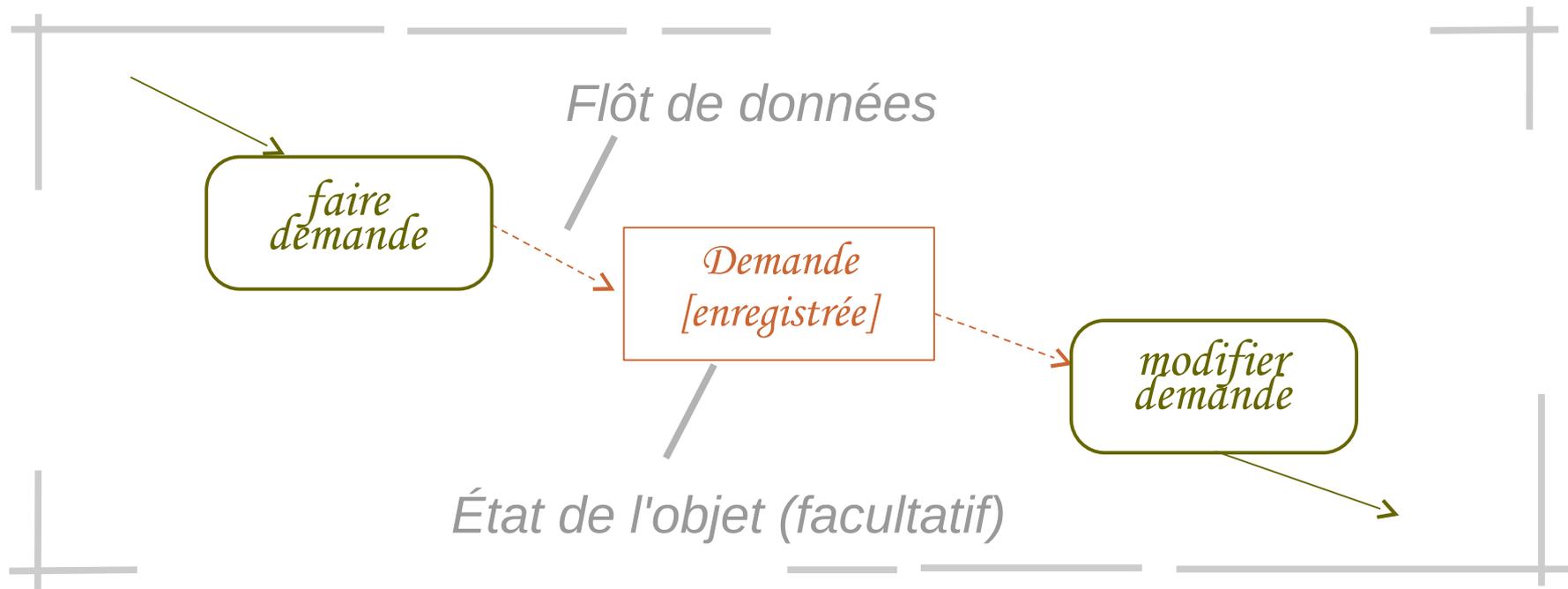


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME D'ACTIVITE

▣ OBJETS

- ✓ Pour faire le liens avec les objets
- ✓ Objets représentés comme flots en entrée ou en sortie des actions
- ✓ Peut aussi être modélisé à l'aide de paramètres (pins) explicites d'entrée et de sortie des activités



- ✓ Difficilement utilisable si plusieurs objets impliqués
- ✓ Recommandé pour les objets du domaine métier



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME D'ACTIVITE

▣ QUELQUES CONSEILS

✓ A utiliser pour documenter les cas d'utilisation

➔ Les cas d'utilisation **en extension** doivent apparaître dans le cadre d'un branchement / fusion qui explicite la **condition** d'extension

➔ Distinguer les cas d'utilisation des tâches qui les constituent

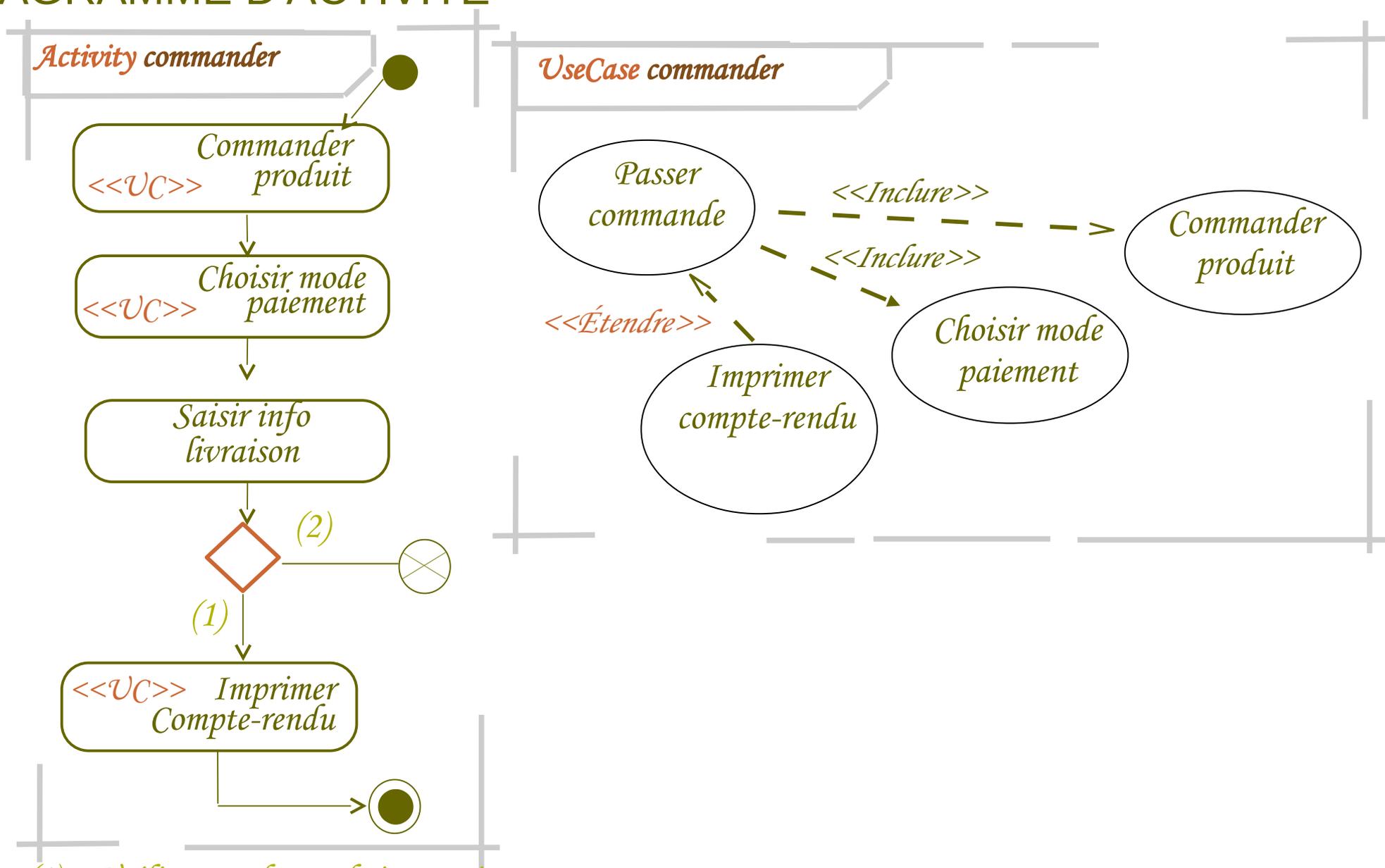
Actions / Activités

➔ Stéréotypes



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME D'ACTIVITE



(1) Utilisateur demande impression

(2) Utilisateur ne demande pas l'impression



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME D'ACTIVITE

▣ QUELQUES CONSEILS (suite)

- ✓ A utiliser à bon escient
- ✓ Peuvent devenir trop complexes
- ✓ Décorrélés des objets

▣ ENCORE A DECOUVRIR

- ✓ Les exceptions
- ✓ Les jetons
- ✓ ...



LES DIAGRAMMES COMPORTEMENTAUX

- Introduction
- Diagramme d'activité
- ▶ □ Diagrammes d'interaction
 - Diagramme de séquence
 - Diagramme de communication
 - Diagramme global d'interaction
 - Diagramme de temps
- Diagramme de machine d'état

LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME D'INTERACTION

- ✓ Diagramme de **séquence**
 - ✓ Diagramme de **communication**
- ← Alternatives
- ✓ Diagramme global d'**interaction**
 - ✓ Diagramme de **temps**
- ← Nouveaux

LES DIAGRAMMES COMPORTEMENTAUX

- Introduction
- Diagramme d'activité
- Diagrammes d'interaction
 - ▶ ■ Diagramme de séquence
 - Diagramme de communication
 - Diagramme global d'interaction
 - Diagramme de temps
- Diagramme de machine d'état



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ GENERALITES

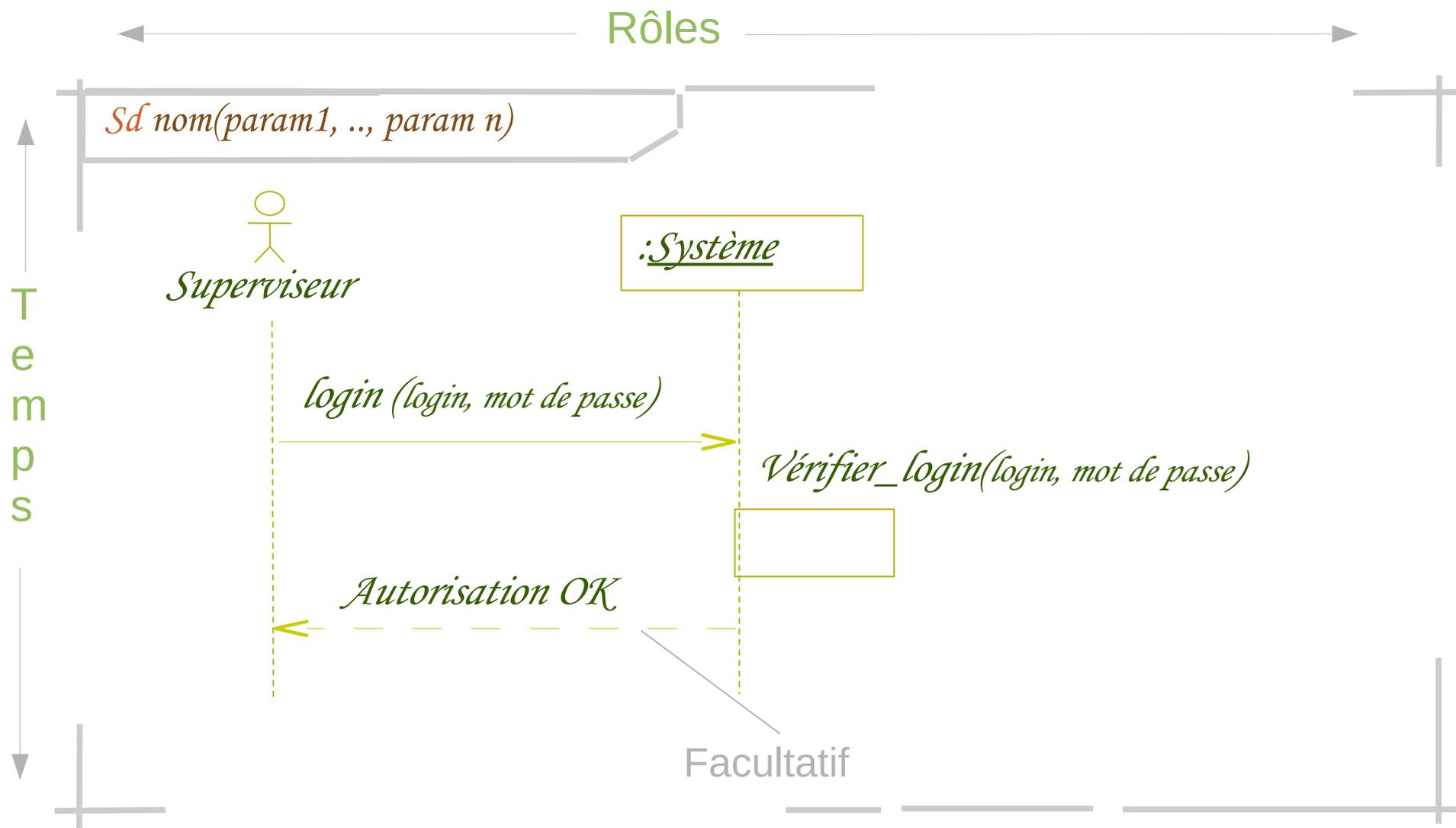
- ✓ Représentation des interactions entre entités
- ✓ Description d'un scénario
 - ✓ Flot normal
 - ✓ Flot exceptionnel
- ✓ Point de vue temporel
- ✓ Documente un (ou plusieurs) cas d'utilisation
- ✓ Utiles en phase d'analyse, de conception & de test



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ STRUCTURE



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ ROLES

➔ **nom** **selecteur** : **type**

Rôle dans
l'interaction

Type & nom : l'un des
deux peut être omis

Optionnel, permet
d'identifier 1 élément si
multiplicité > 1

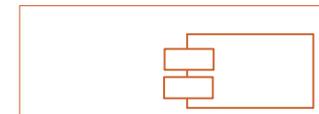
Objet

:Superviseur

Acteur


Nom

Composant


Nom



Ne pas **confondre** acteur / classe

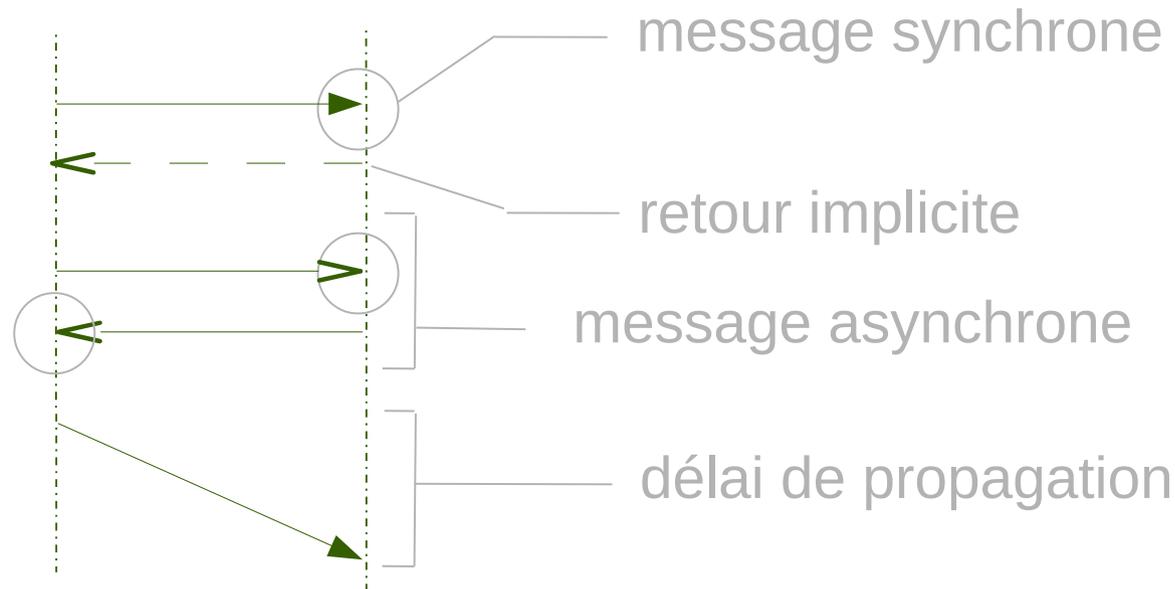


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

□ MESSAGES

- ✓ Des messages pour :
 - ✓ Envoyer un message/signal ($\rightarrow \dashrightarrow$ acteur)
 - ➔ événements du domaine d'application
 - ✓ Appeler une méthode ($\rightarrow \dashrightarrow$ objets / composants)
 - ➔ appels d'opération



➔ Valeur-retournée := nom-message(args)

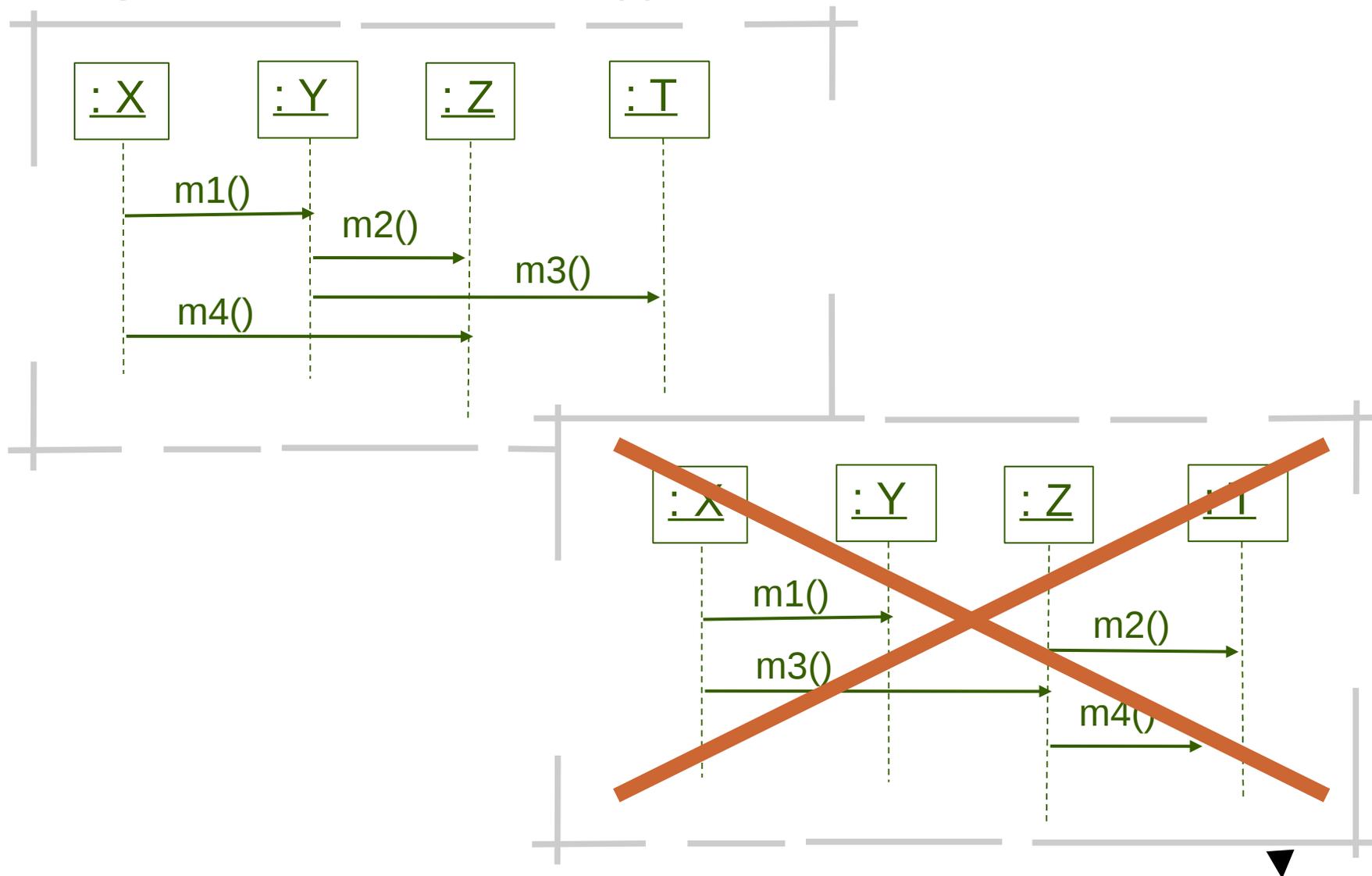


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

MESSAGES (suite)

✓ Soigner la succession des appels

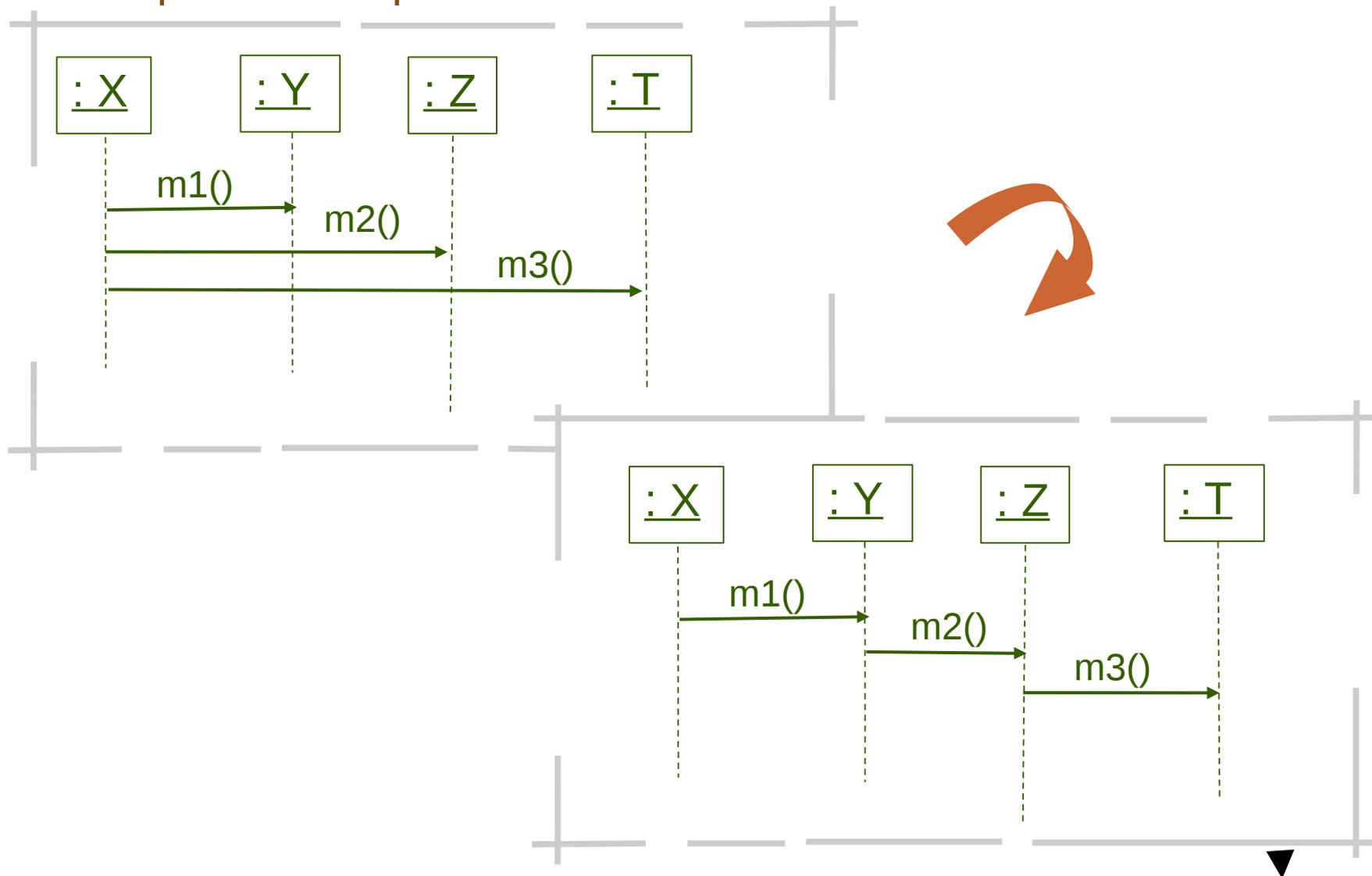


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

MESSAGES (suite)

✓ Encapsuler les opérations

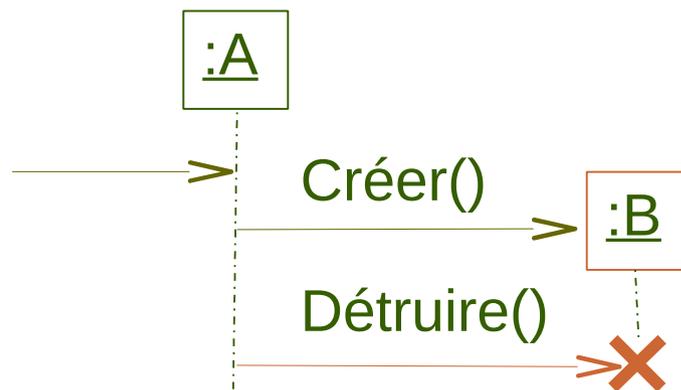


LES DIAGRAMMES COMPORTEMENTAUX

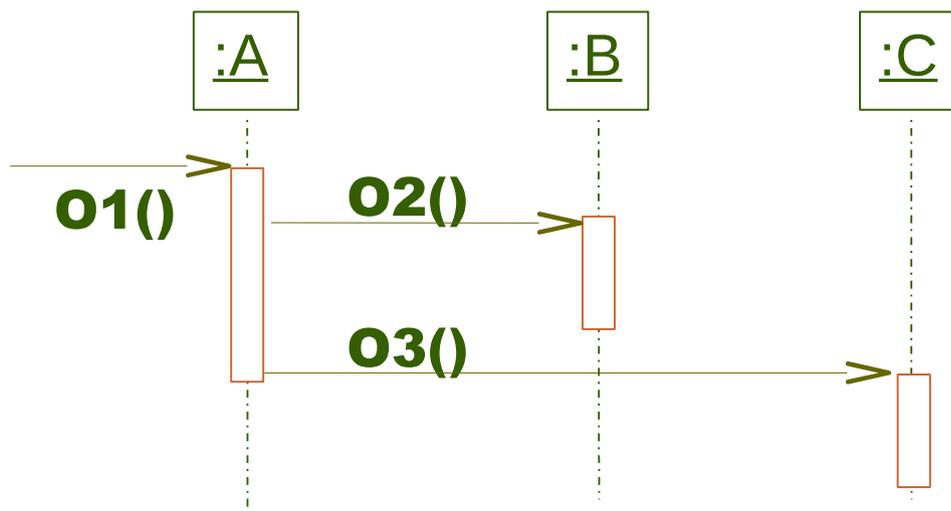
DIAGRAMME DE SEQUENCE

▣ OBJETS

✓ Création & destruction



✓ Durée d'activation

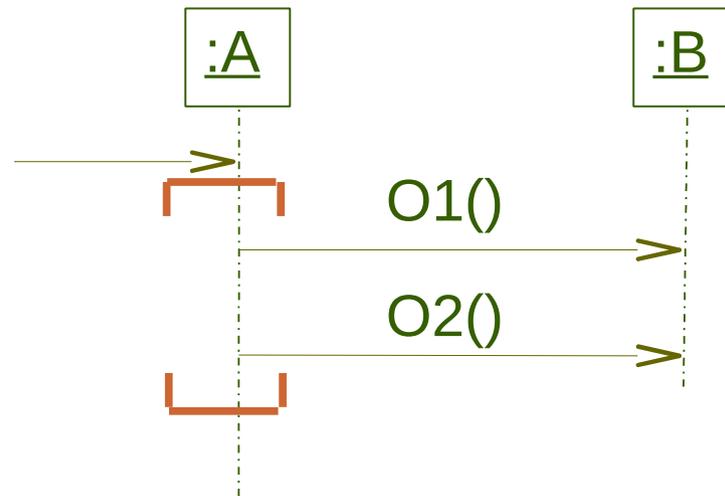


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ COREGION

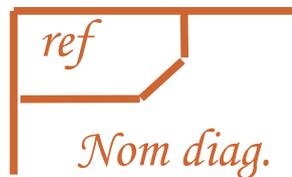
✓ Pour indiquer que l'ordre des évènements n'a pas d'importance



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

- ❑ **FRAGMENTS COMBINES / *combined fragment / inline frame***
 - ✓ Articulations entre interactions
 - ✓ Pour une description plus compacte
 - ✓ Composés d'opérandes et d'un opérateur
- ✓ Les opérateurs peuvent être combinés les uns avec les autres
- ✓ Les diagrammes peuvent se référencer les uns les autres
 - ✓ Pointeur / raccourci vers un diagramme de séquence complètement décrit
 - ✓ Pour factoriser des parties de comportements à plusieurs endroits



Paramètres des diagrammes

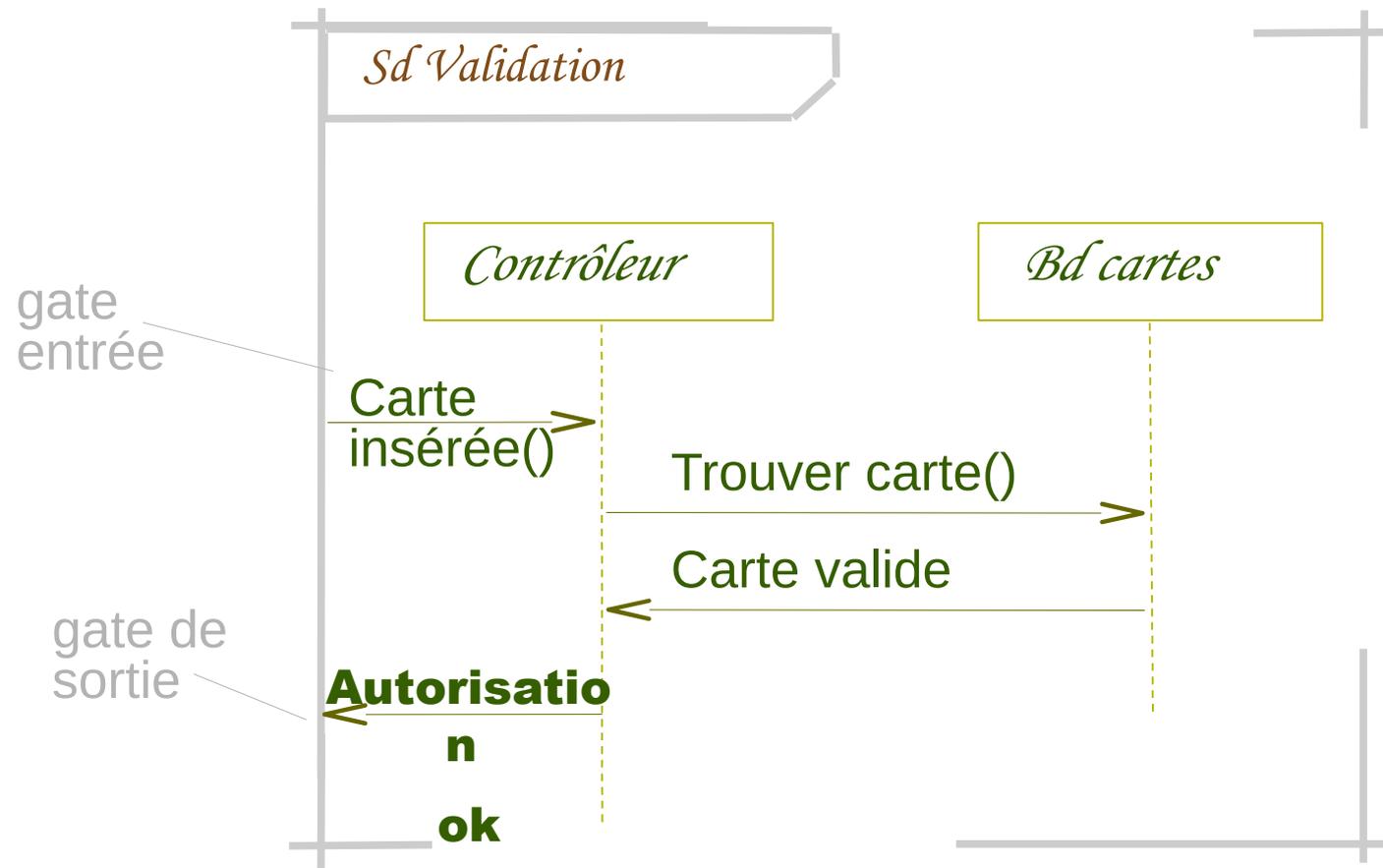


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

FRAGMENTS COMBINES (suite)

- ✓ Les ports (*Gate*) permettent de mieux spécifier le passage de paramètres entre diagrammes de séquence



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ LES OPERATEURS

- ▶ ✓ Les opérateurs de base
 - ✓ ALT / OPT
 - ✓ LOOP
 - ✓ PAR
 - ✓ SEQ WEAK/STRICT

- ✓ Les opérateurs pour les situations exceptionnelles
 - ✓ BREAK

- ✓ Les opérateurs pour les tests
 - ✓ CRITICAL
 - ✓ IGNORE/CONSIDER
 - ✓ NEGATIVE
 - ✓ ASSERT

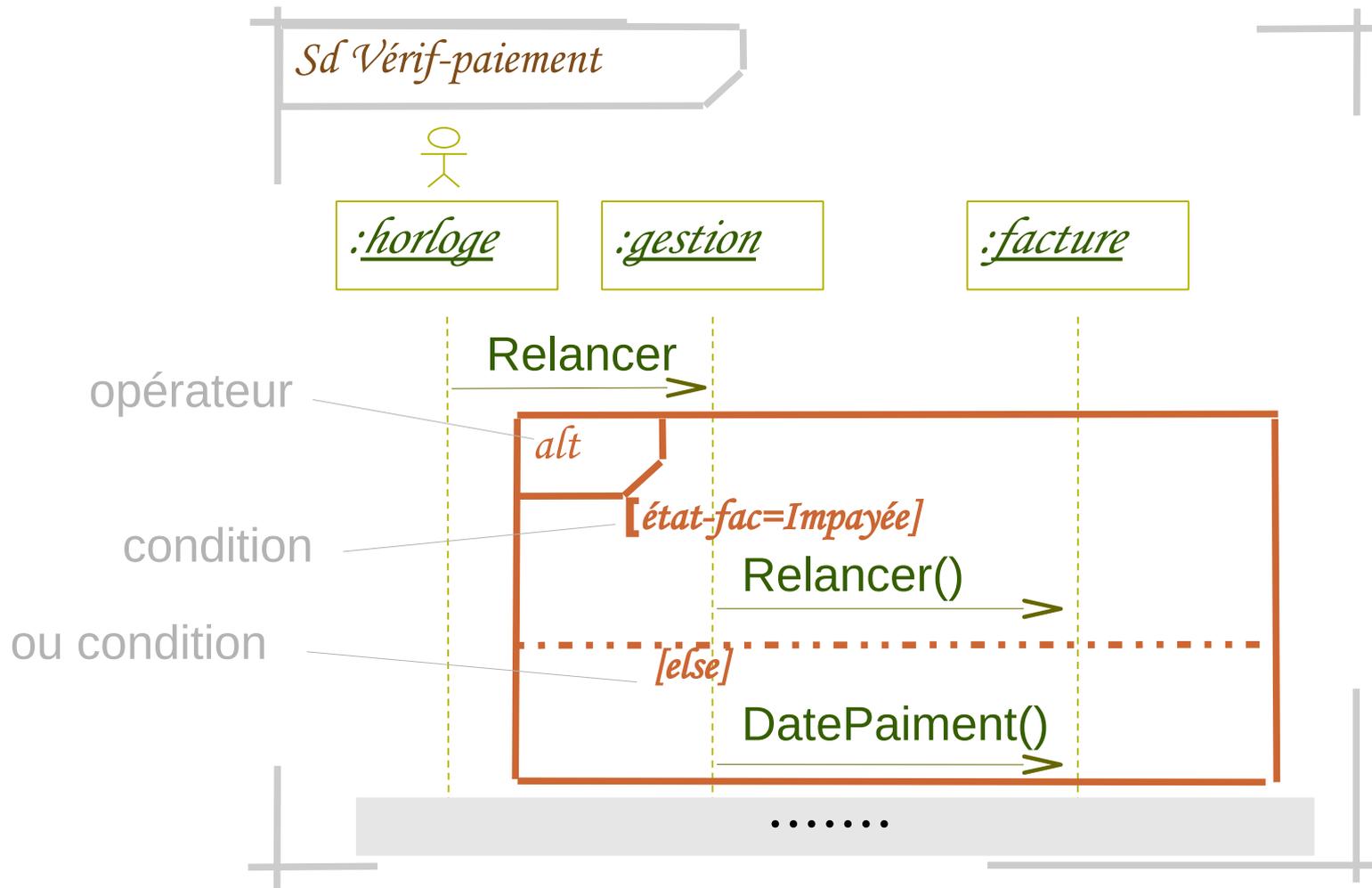


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ OPERATEUR *ALTERNATIVE*

- ✓ Pour représenter un choix, une alternative
- ✓ Condition implicite ou explicite

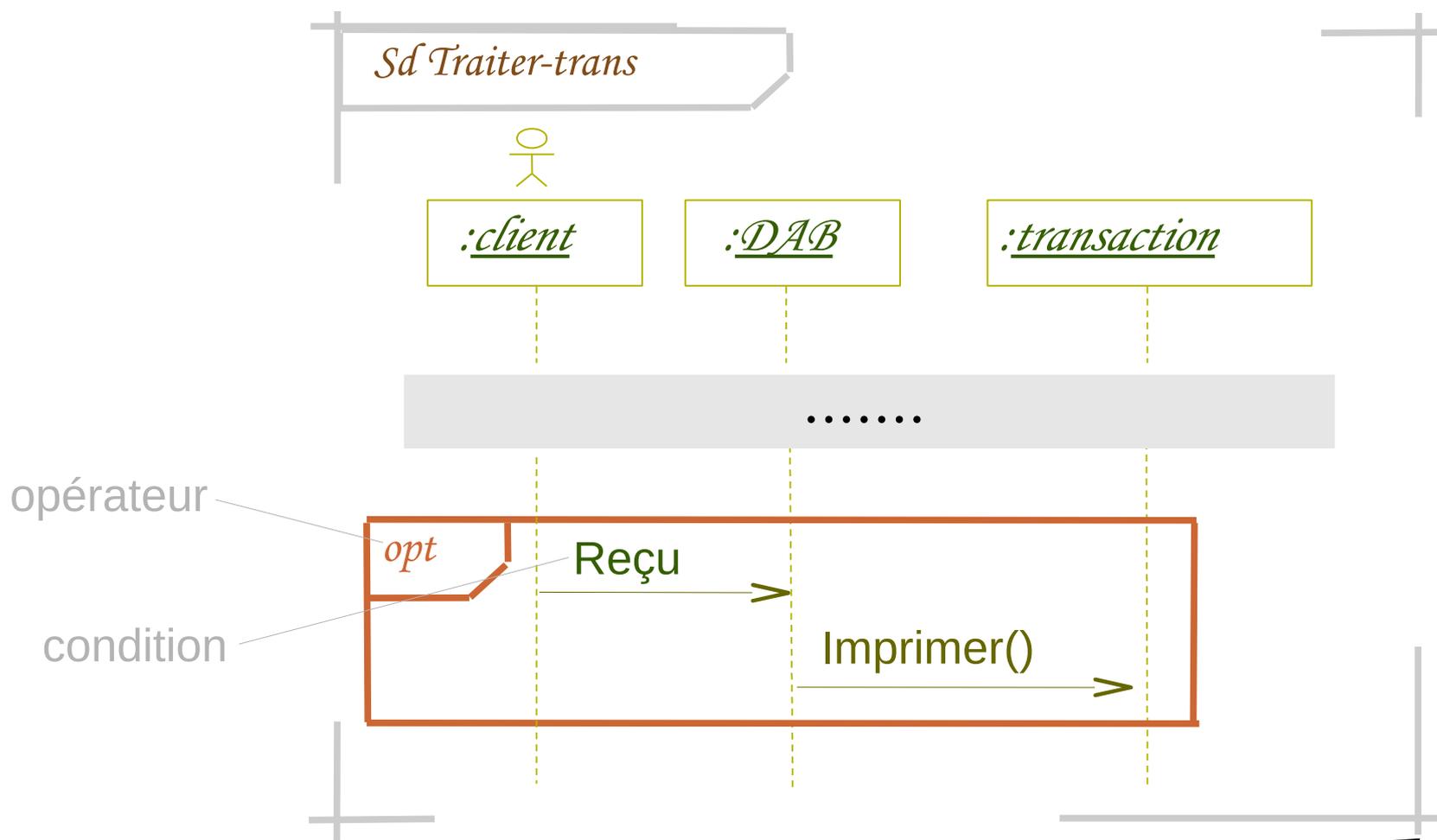


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ OPERATEUR *OPTION*

- ✓ Pour représenter un fragment optionnel
- ✓ Correspond à un *alt* sans *else*

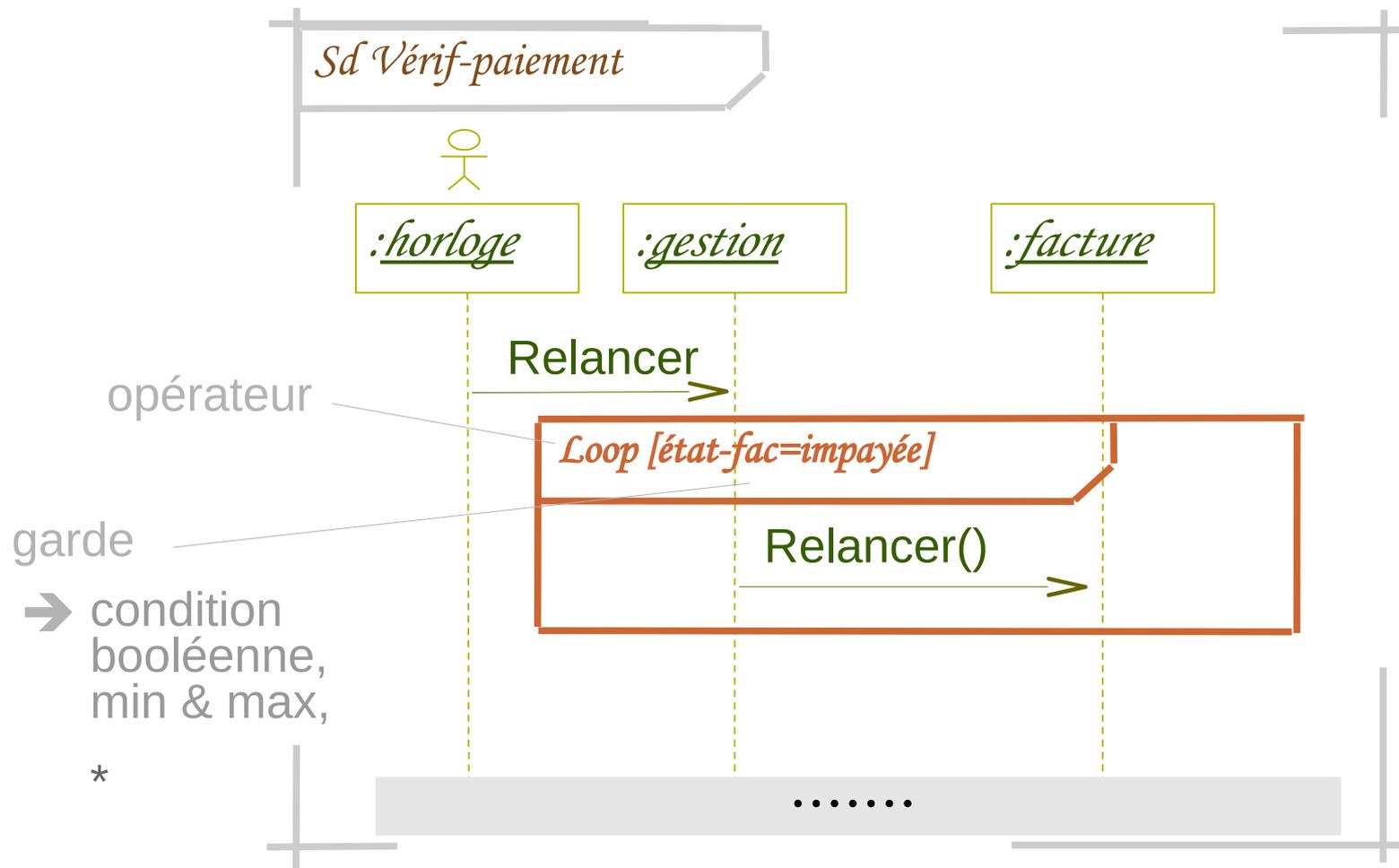


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

OPERATEUR *LOOP*

- ✓ Pour indiquer qu'un ensemble d'interactions s'exécute en boucle

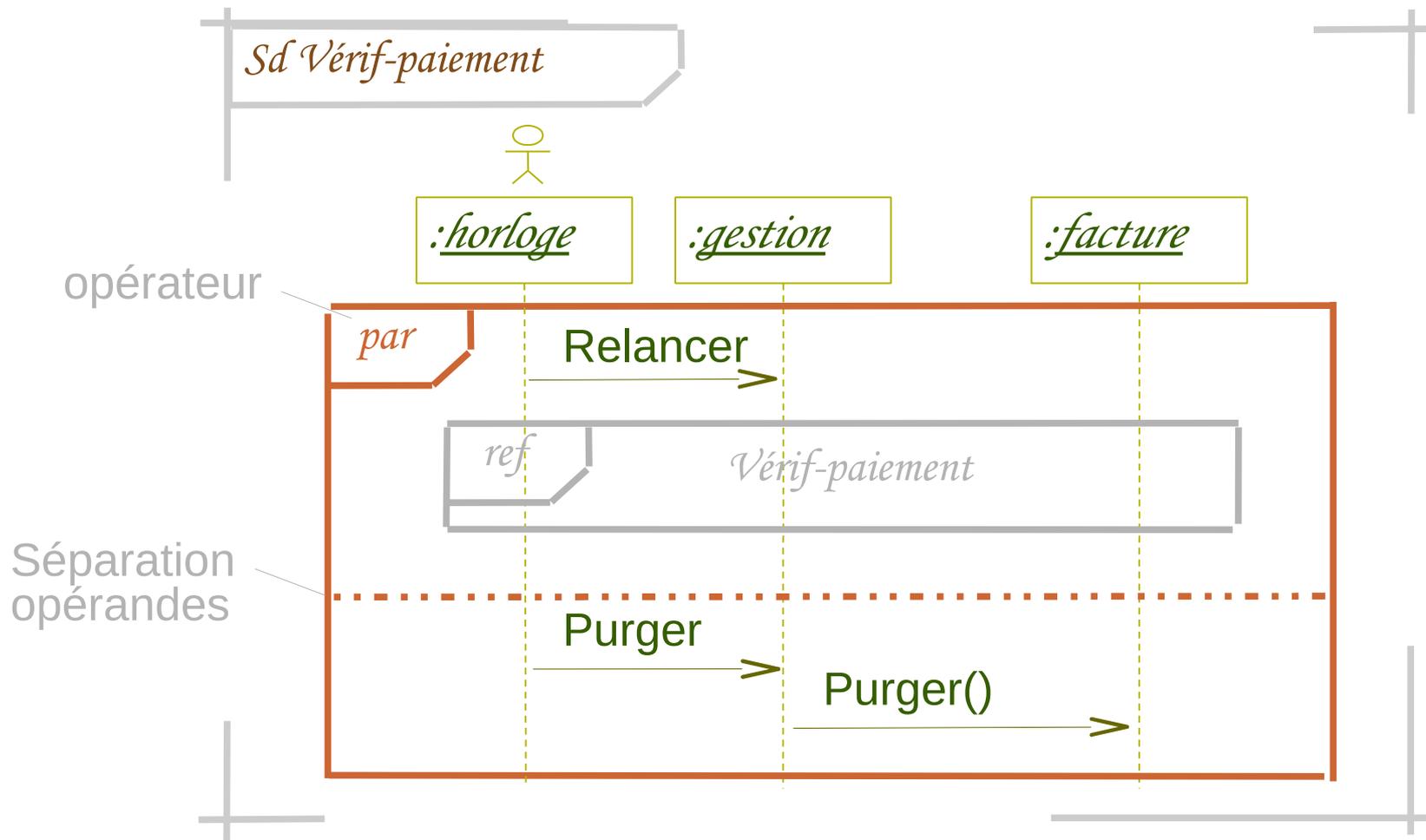


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ OPERATEUR PARALLELE

- ✓ Interactions réalisées en parallèle
- ✓ Ordre imposé dans chaque opérande respecté



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ OPERATEUR *STRICT / WEAK SEQUENCING*

✓ Pour préciser l'ordre des interactions entre entités indépendantes

✓ Opérateur *WEAK*

→ Les interactions qui s'opèrent entre entités indépendantes n'ont pas d'ordre particulier

→ Mode par défaut

✓ Opérateur *STRICT*

→ L'ordre est celui décrit sur le diagramme

→ A l'intérieur des opérandes

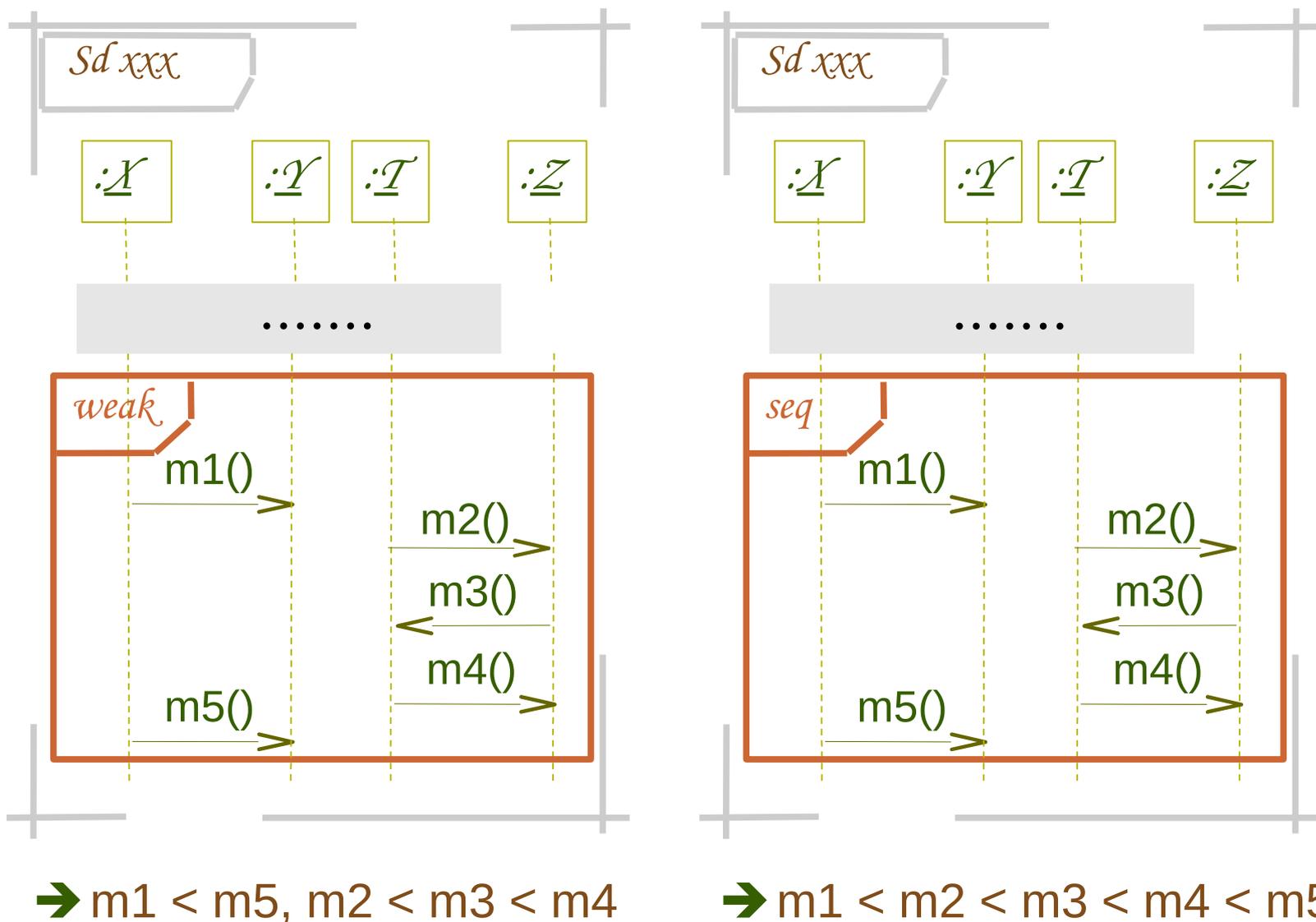
→ Entre les opérandes



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ OPERATEUR *STRICT* / *WEAK SEQUENCING* (suite)



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ LES OPERATEURS

- ✓ Les opérateurs de base
 - ✓ ALT / OPT
 - ✓ LOOP
 - ✓ PAR
 - ✓ SEQ WEAK/STRICT

- ▶ ✓ Les opérateurs pour les situations exceptionnelles
 - ✓ BREAK

- ✓ Les opérateurs pour les tests
 - ✓ CRITICAL
 - ✓ IGNORE/CONSIDER
 - ✓ NEGATIVE
 - ✓ ASSERT

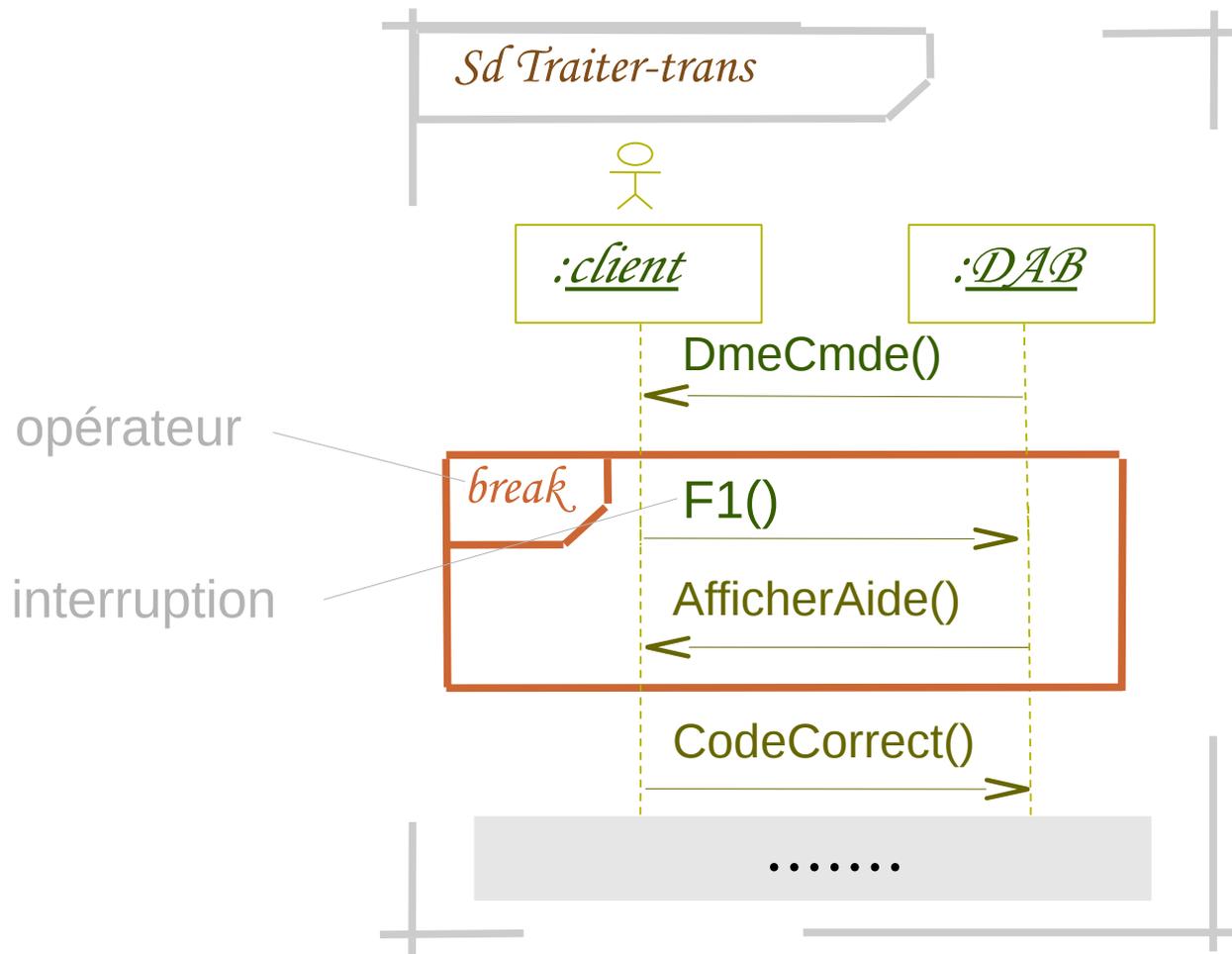


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ OPERATEUR *BREAK*

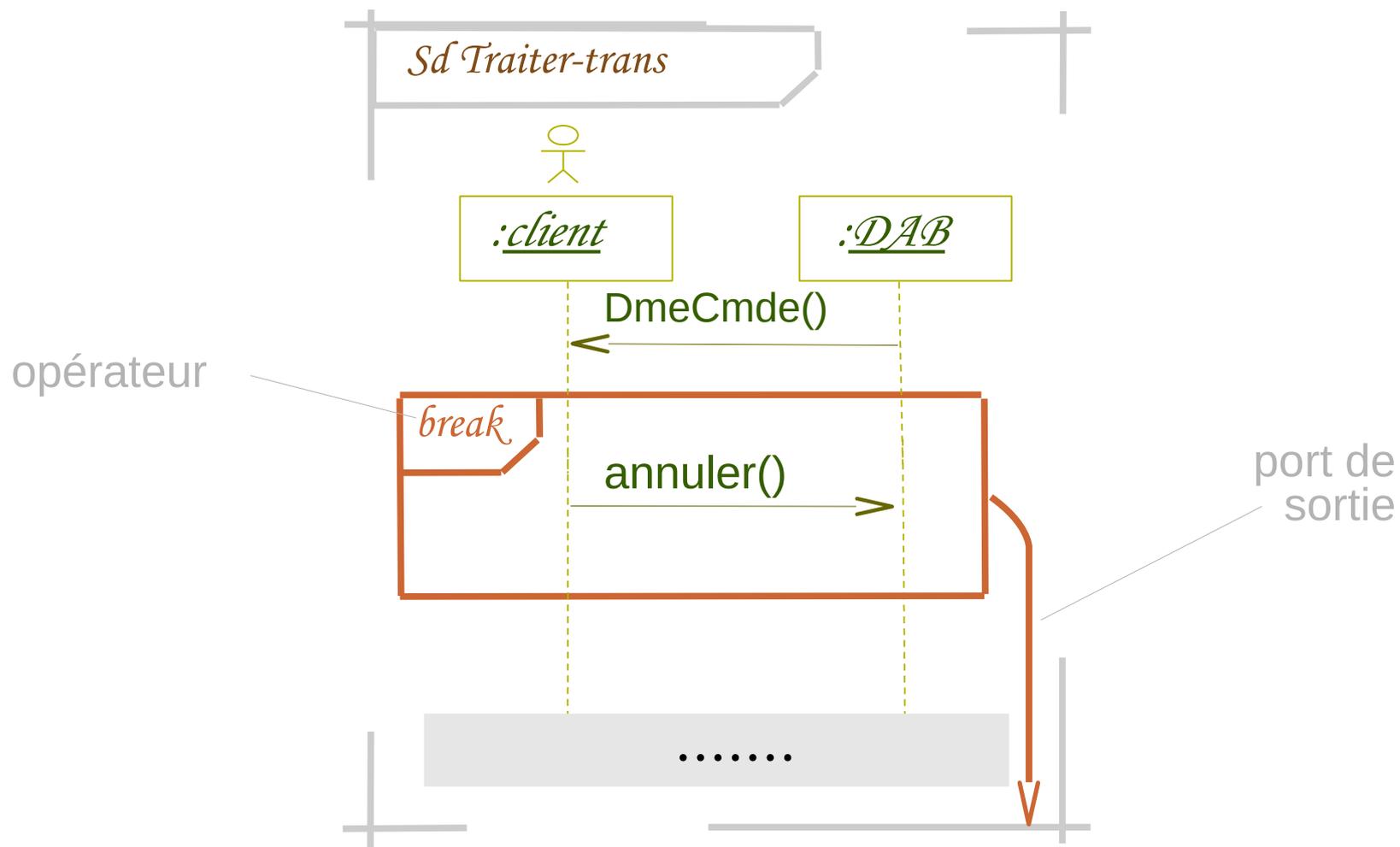
- ✓ Pour représenter des situations exceptionnelles
- ✓ Implique une interruption du flôt normal des interactions



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ OPERATEUR *BREAK* (suite)



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ LES OPERATEURS

✓ Les opérateurs de base

- ✓ ALT / OPT
- ✓ LOOP
- ✓ PAR
- ✓ SEQ WEAK/STRICT

✓ Les opérateurs pour les situations exceptionnelles

- ✓ BREAK

▶ ✓ Les opérateurs pour les tests

- ✓ CRITICAL
- ✓ IGNORE/CONSIDER
- ✓ NEGATIVE
- ✓ ASSERT

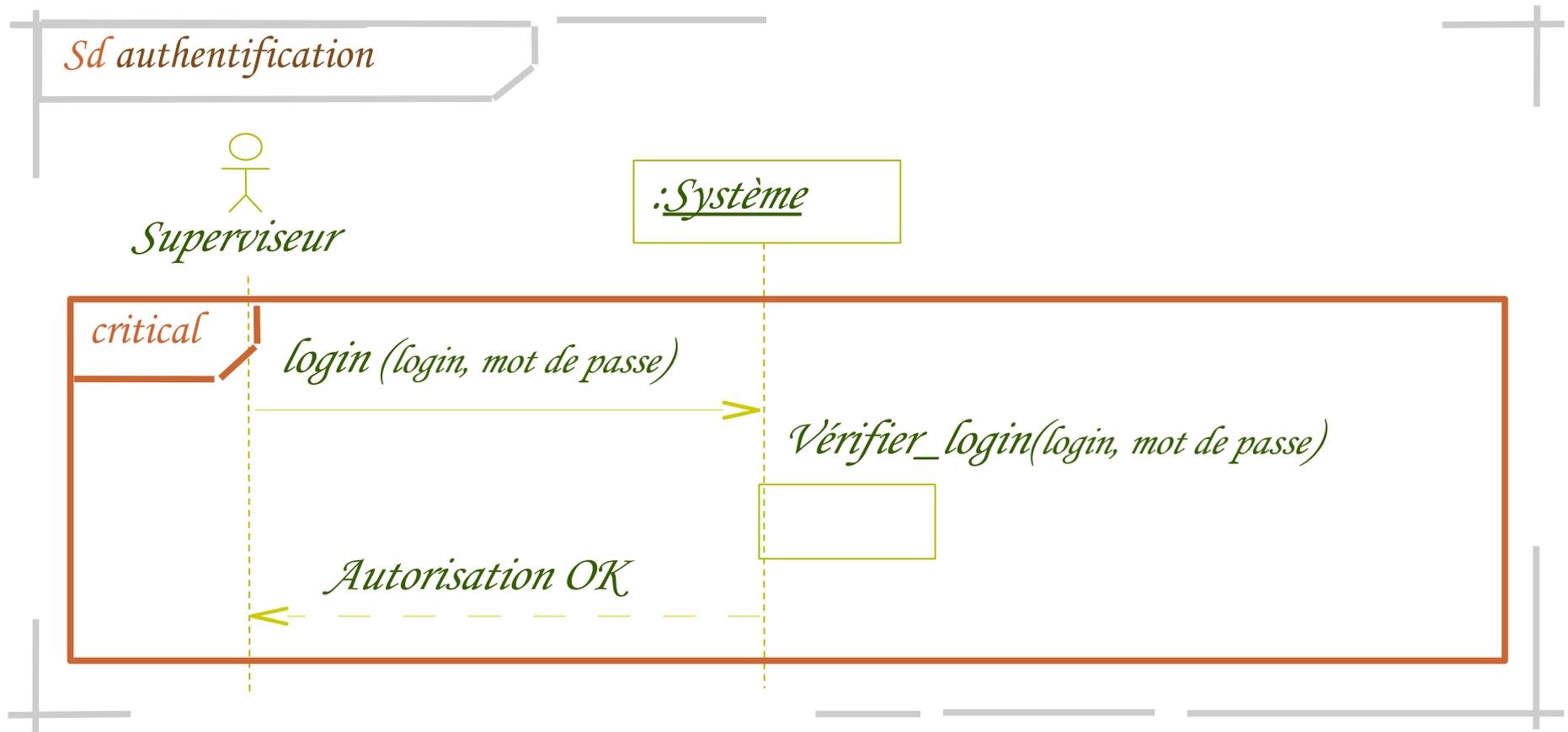


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ OPERATEUR *CRITICAL*

- ✓ Section dans laquelle les interactions ne peuvent pas être interrompues
- ✓ Indique un traitement atomique des interactions



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ OPERATEURS *IGNORE* & *CONSIDER*

✓ Pour indiquer que l'**absence** d'un ou plusieurs évènements n'a pas d'incidence sur la séquence (par opposition aux autres)

→ Opérateur *IGNORE*

✓ Pour indiquer que la **présence** d'un ou plusieurs évènements est obligatoire (par opposition aux autres)

→ Opérateur *CONSIDER*

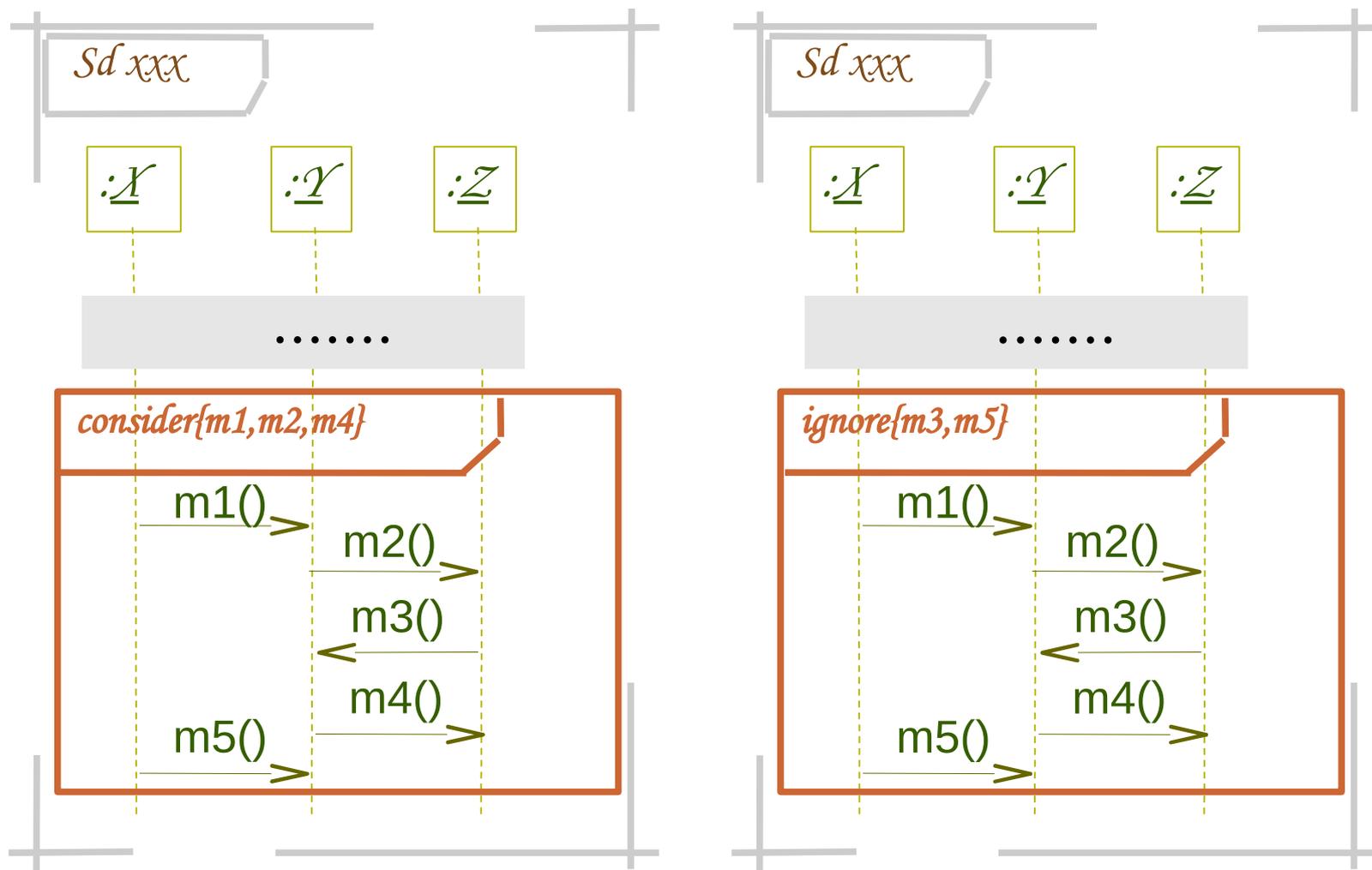
✓ Opérateurs plutôt dédiés au profil de test



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ OPERATEUR *IGNORE* & *CONSIDER* (suite)



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

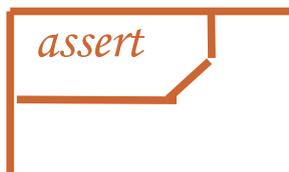
▣ OPERATEUR *NEGATIVE*

- ✓ Séquence d'interaction invalide



▣ OPERATEUR *ASSERTION*

- ✓ Pour désigner l'unique séquence possible
- ✓ Généralement utiliser en combinaison avec ignore / consider
- ✓ Opérateur plutôt dédié au profil de test

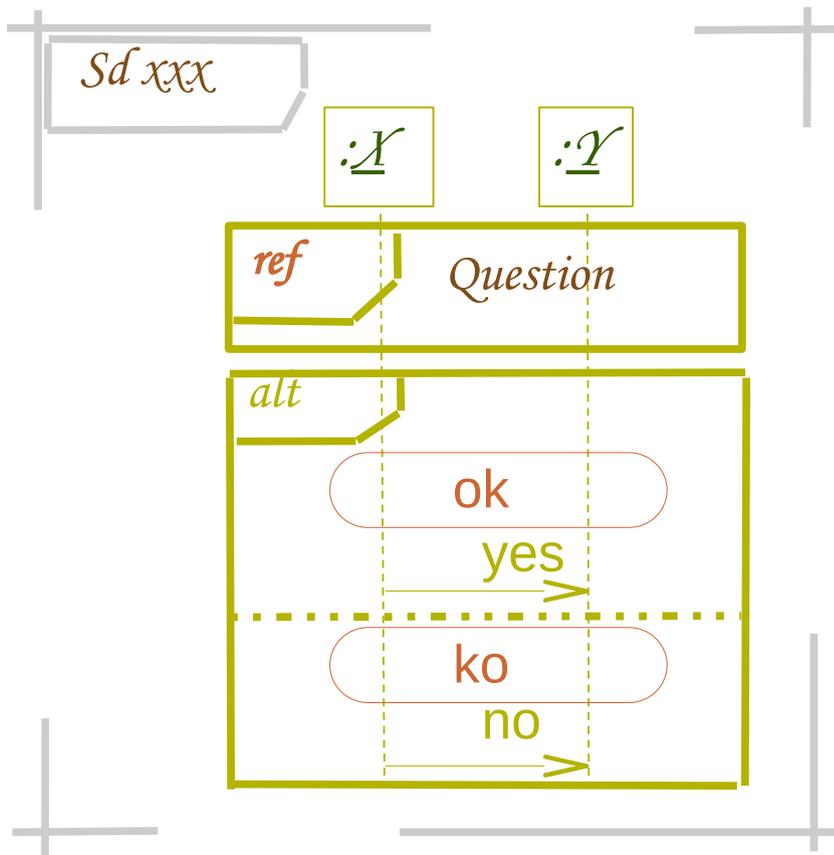


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

□ CONTINUATION

- ✓ Equivalent à des labels indiquant des endroits spécifiques dans le flôt de contrôle

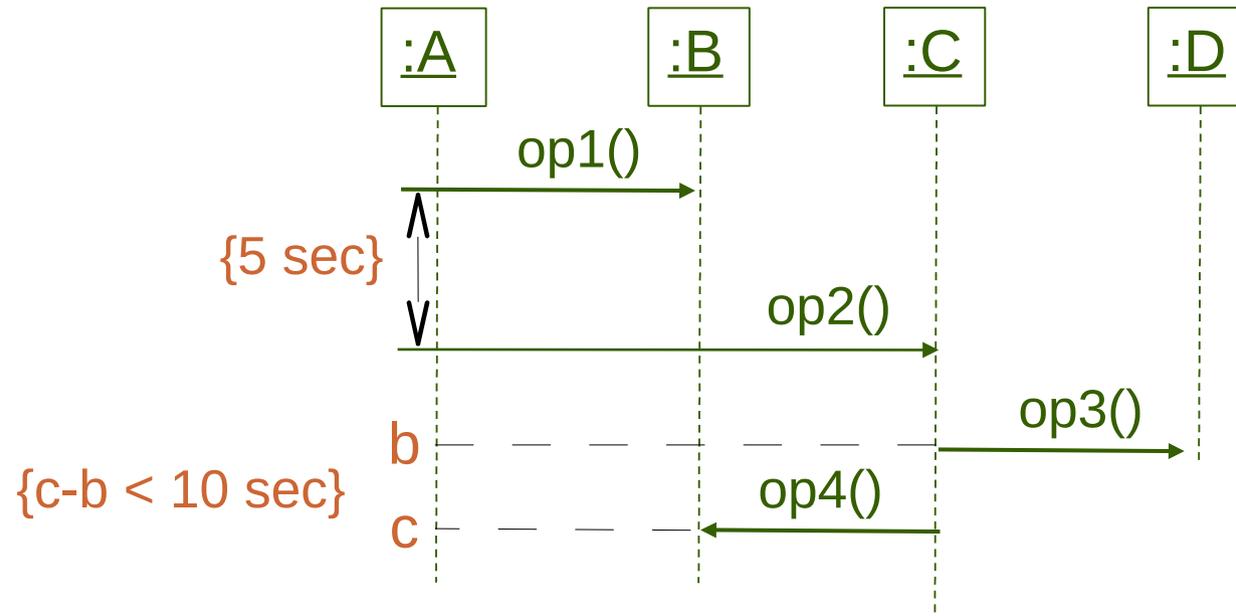


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

□ CONTRAINTES DE TEMPS

- ✓ Contrainte de durée
- ✓ Contrainte temporelle

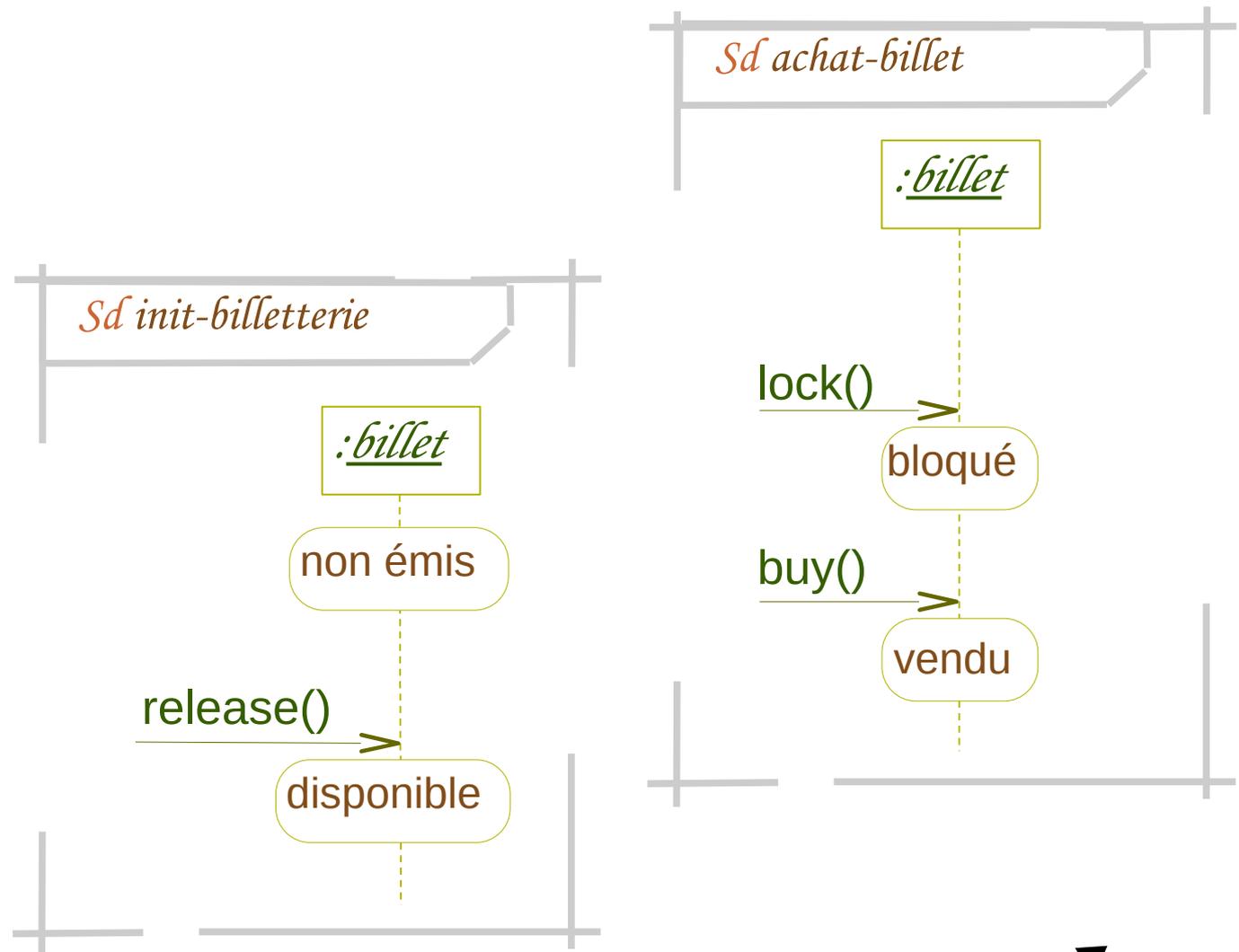
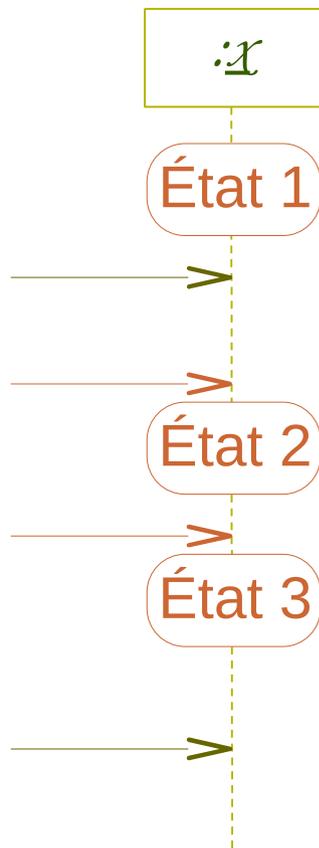


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ OBJETS (suite)

✓ Modélisation des changements d'état



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE SEQUENCE

▣ QUELQUES CONSEILS

- ✓ Toujours donner le **contexte** du diagramme
 - **cas d'utilisation**
- ✓ Indiquer précisément le **but** du scénario
- ✓ Bien préciser
 - ✓ l'**acteur** qui **déclenche** le scénario
 - ✓ le **résultat observable** de l'exécution du cas d'utilisation
- ✓ Faire apparaître un objet **interface/application** entre l'acteur & les objets du système



- ✓ Ne pas confondre
 - ✓ cas d'utilisation / diagramme de séquence
- ✓ **Cohérence** diagramme de classe / diagramme de séquence

LES DIAGRAMMES COMPORTEMENTAUX

- Introduction
- Diagramme d'activité
- Diagrammes d'interaction
 - Diagramme de séquence
 - ▶ ■ Diagramme de communication
 - Diagramme global d'interaction
 - Diagramme de temps
- Diagramme de machine d'état

LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE COMMUNICATION

- ✓ **Interactions** entre objets
- ✓ Décrit un **scénario**
- ✓ Structure **spatiale** statique

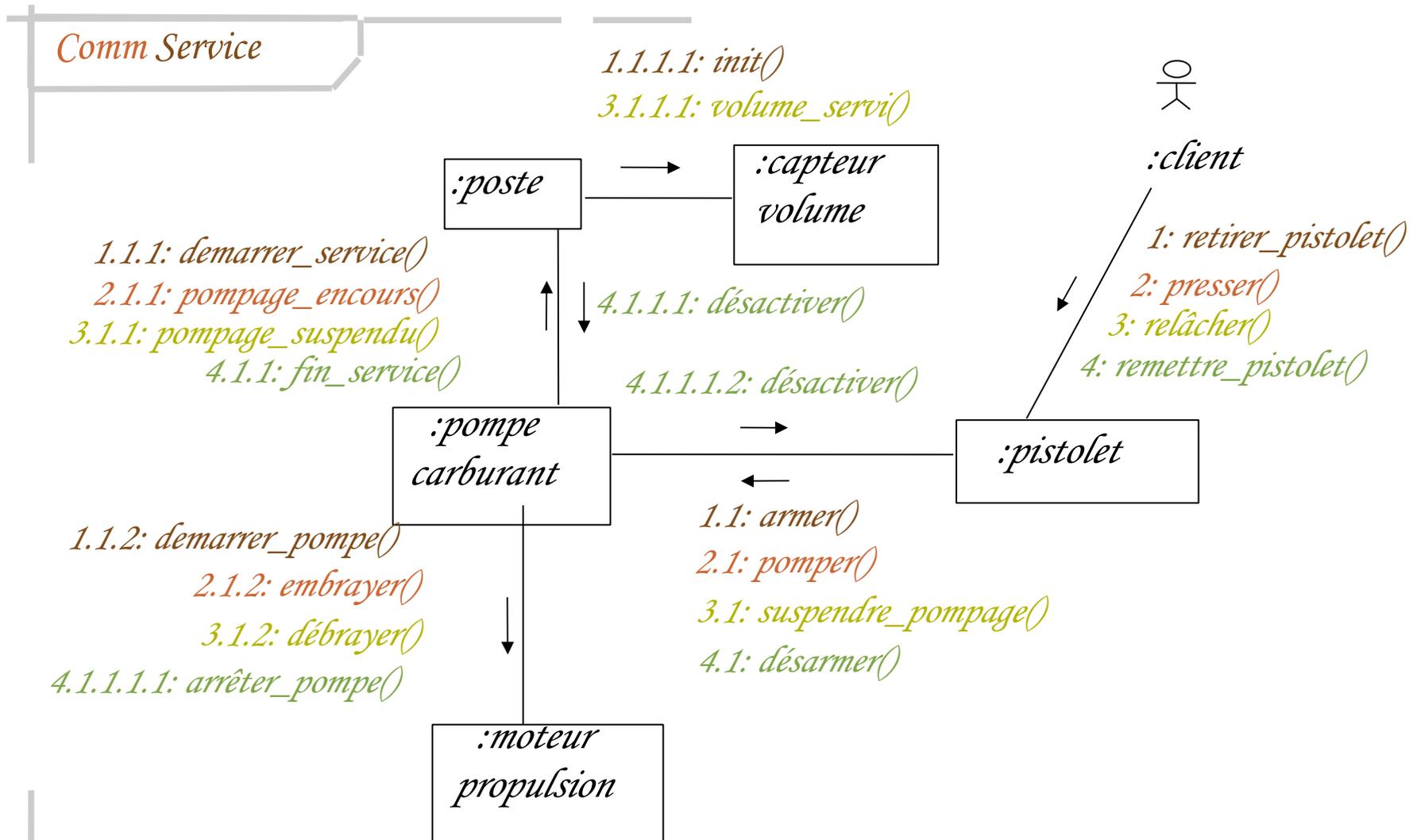
- ✓ Documente les **cas d'utilisation**
- ✓ Représentation **précise** des interactions entre objets



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE COMMUNICATION

EXEMPLE



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE COMMUNICATION

▣ ROLE

➔ **nom** **selecteur** : **type**

Rôle dans
l'interaction

Type & nom : l'un des
deux peut être omis

Optionnel, permet
d'identifier 1 élément si
multiplicité > 1

Objet

:Superviseur

Acteur


Nom

Composant


Nom



Ne pas **confondre** acteur / classe



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE COMMUNICATION

□ MESSAGE

- ✓ Des messages pour :
 - ✓ Envoyer un message/signal (\rightarrow \neg acteur)
 - ➔ *événements du domaine d'application*
 - ✓ Appeler une méthode (\rightarrow \neg objets / composants)
 - ➔ *appels d'opération*

 message synchrone

 retour implicite

 message asynchrone

➔ Valeur-retournée := nom-message(args)



LES DIAGRAMMES COMPORTEMENTAUX

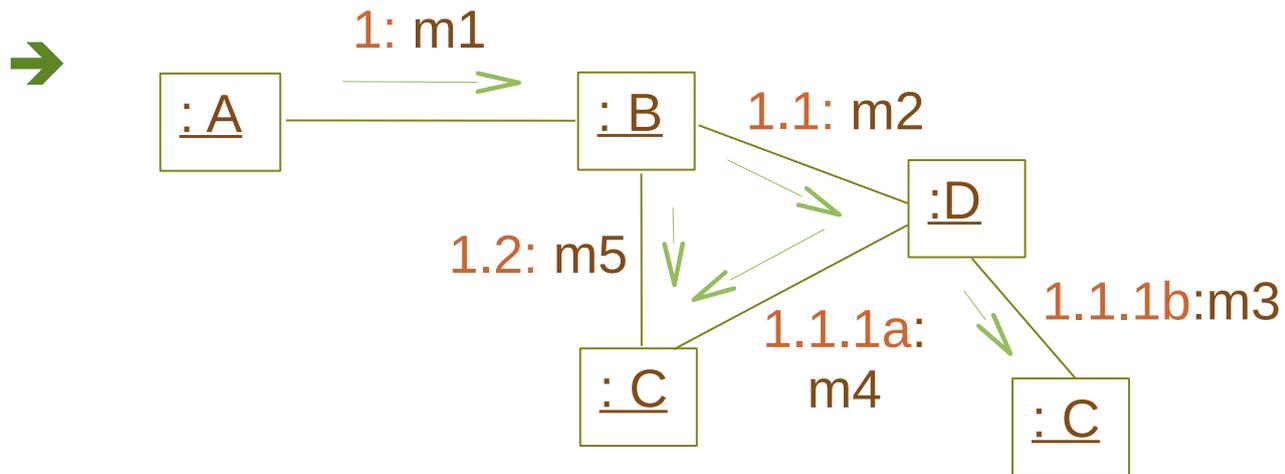
DIAGRAMME DE COMMUNICATION

MESSAGE (suite)

Séquence : *Condition* Valeur-retournée := nom-message(args)

- ➔ Itération : **[clause_itération]*
- ➔ Itération en parallèle : **//[clause_itération]*
- ➔ Condition : *[condition]*

**[facture impayée] relancer(params)*

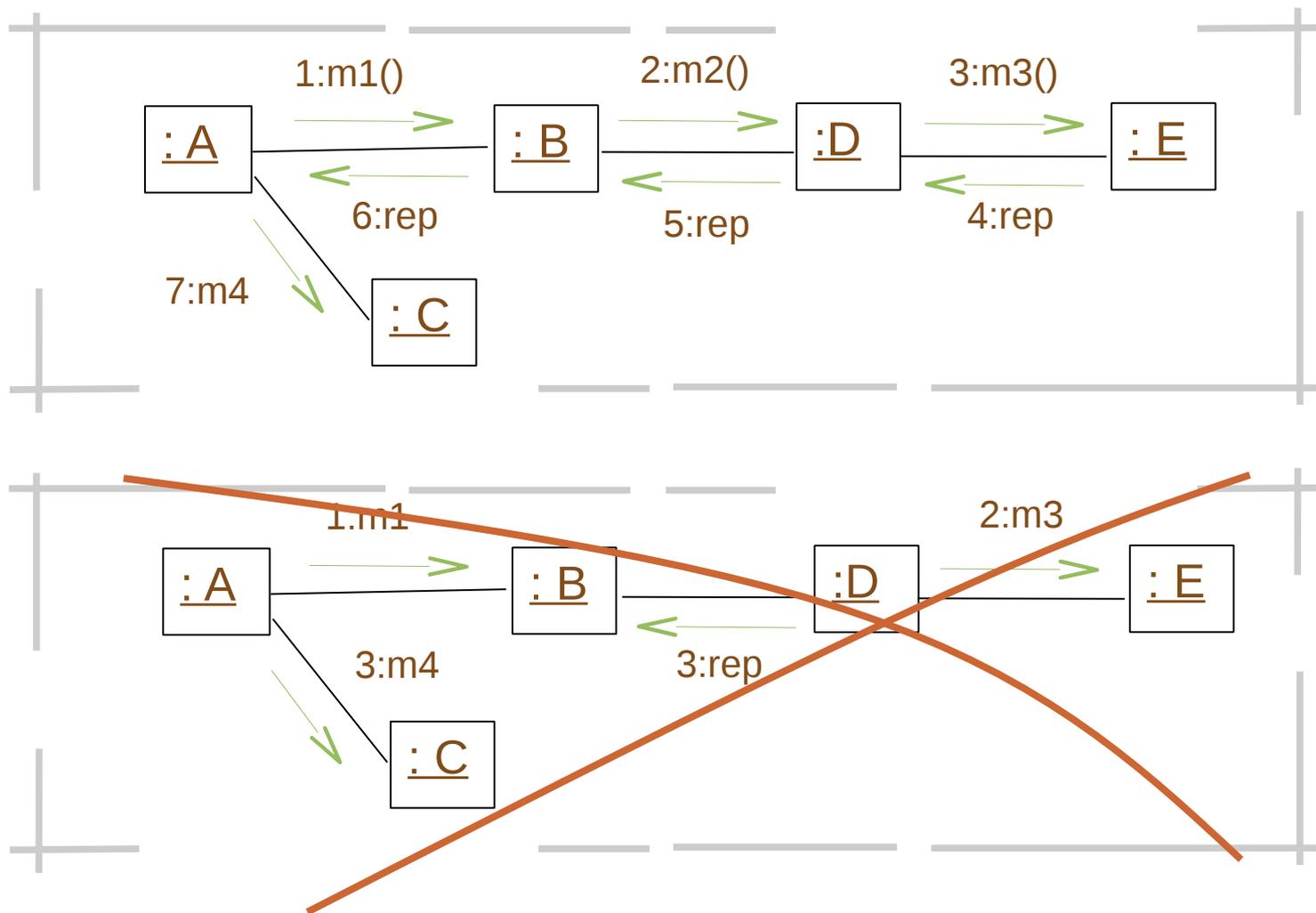


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE COMMUNICATION

MESSAGE (suite)

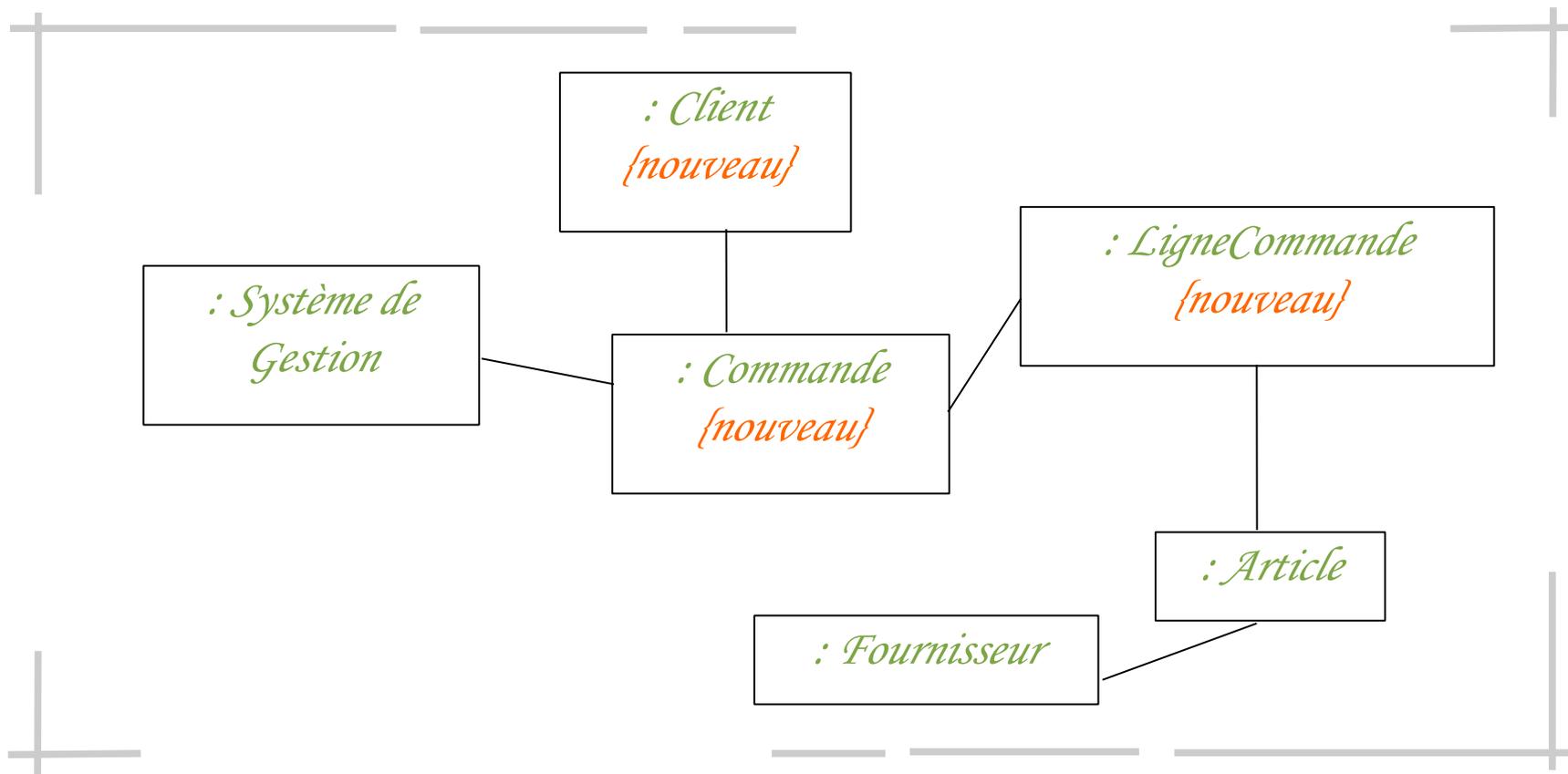
✓ Attention à la succession des appels



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE COMMUNICATION

□ CREATION & DESTRUCTION DE ROLES

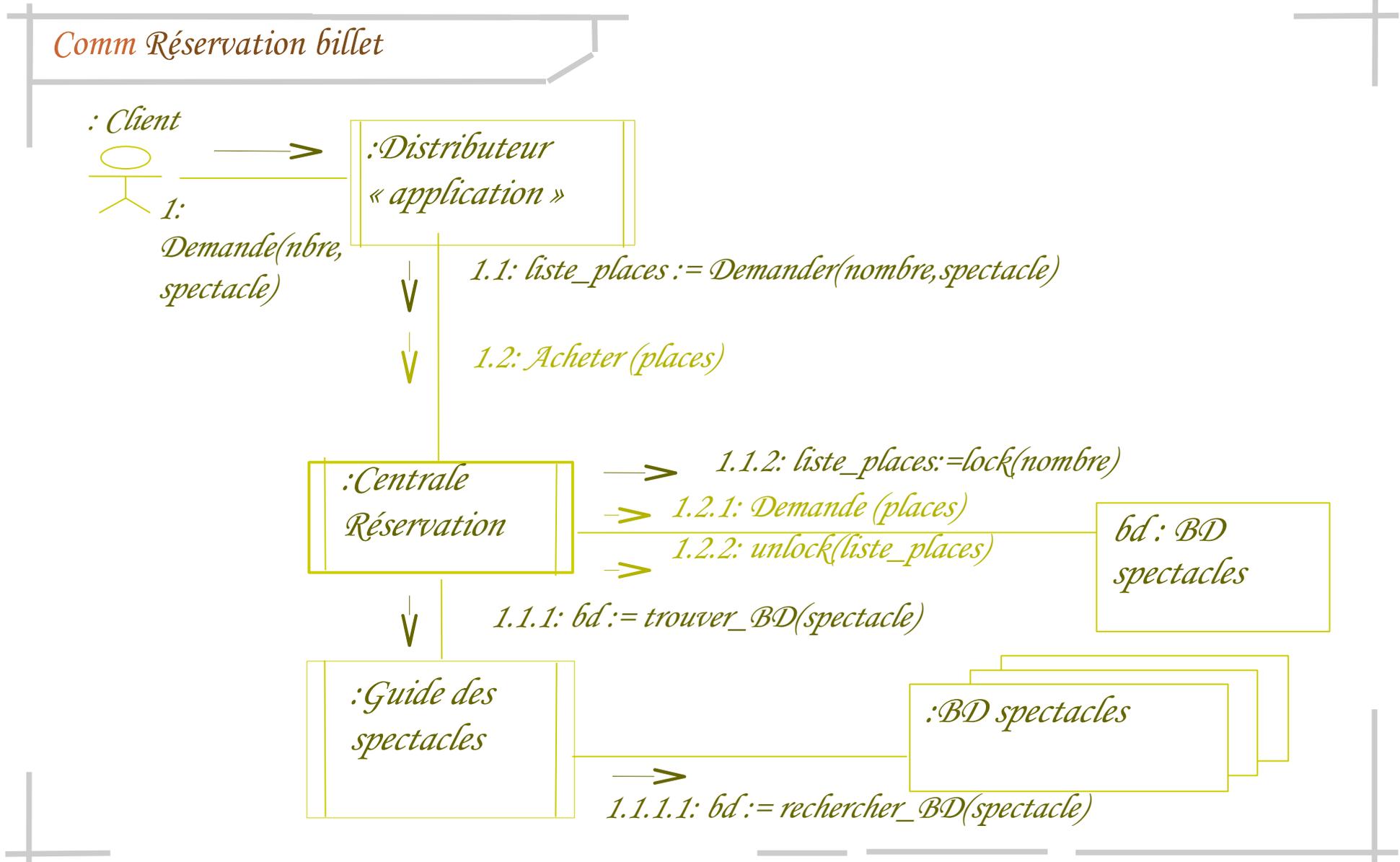


Et aussi : **{transitoire,détruit}**

LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE COMMUNICATION

EXEMPLE



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE COMMUNICATION

□ DIAGRAMME DE SEQUENCE VS DIAGRAMME DE COMMUNICATION

- ✓ Diagramme de **séquence**
 - ✓ pour les spécifications liées au **temps**
 - ✓ pour les scénarios **complexes**
 - ✓ permet la modularité de la représentation
- ✓ Diagramme de **communication**
 - ✓ pour comprendre les effets d'une instance
 - ✓ pour décrire un traitement **procédural**
 - ✓ pas de modularité de la représentation
 - ✓ Représentation moins poussée (opérateurs de test, représentation des exceptions)

□ DIAGRAMME D'INTERACTION

- ✓ Un scénario à la fois
- ✓ Peut être très précis & détaillé
- ✓ Le nombre de diagrammes peut devenir (trop) important
- ✓ Risque de redondances & d'inconsistances



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE COMMUNICATION

□ SCENARIOS

- ✓ en langage naturel
 - ✓ pour l'utilisateur
 - ✓ déblayer le terrain
- ✓ formalisés
 - ✓ pour spécifier plus précisément
 - ✓ pour aider au travail d'analyse

➔ Ne pas trop formaliser

➔ Diagrammes de séquence & de communication équivalents

✓ Choisir judicieusement les scénarios présentés

➔ Un diagramme d'**activité** pour décrire l'ensemble des **situations normales**

➔ Des diagrammes de **séquence** et/ou de **communication** pour décrire les situations **exceptionnelles** intéressantes



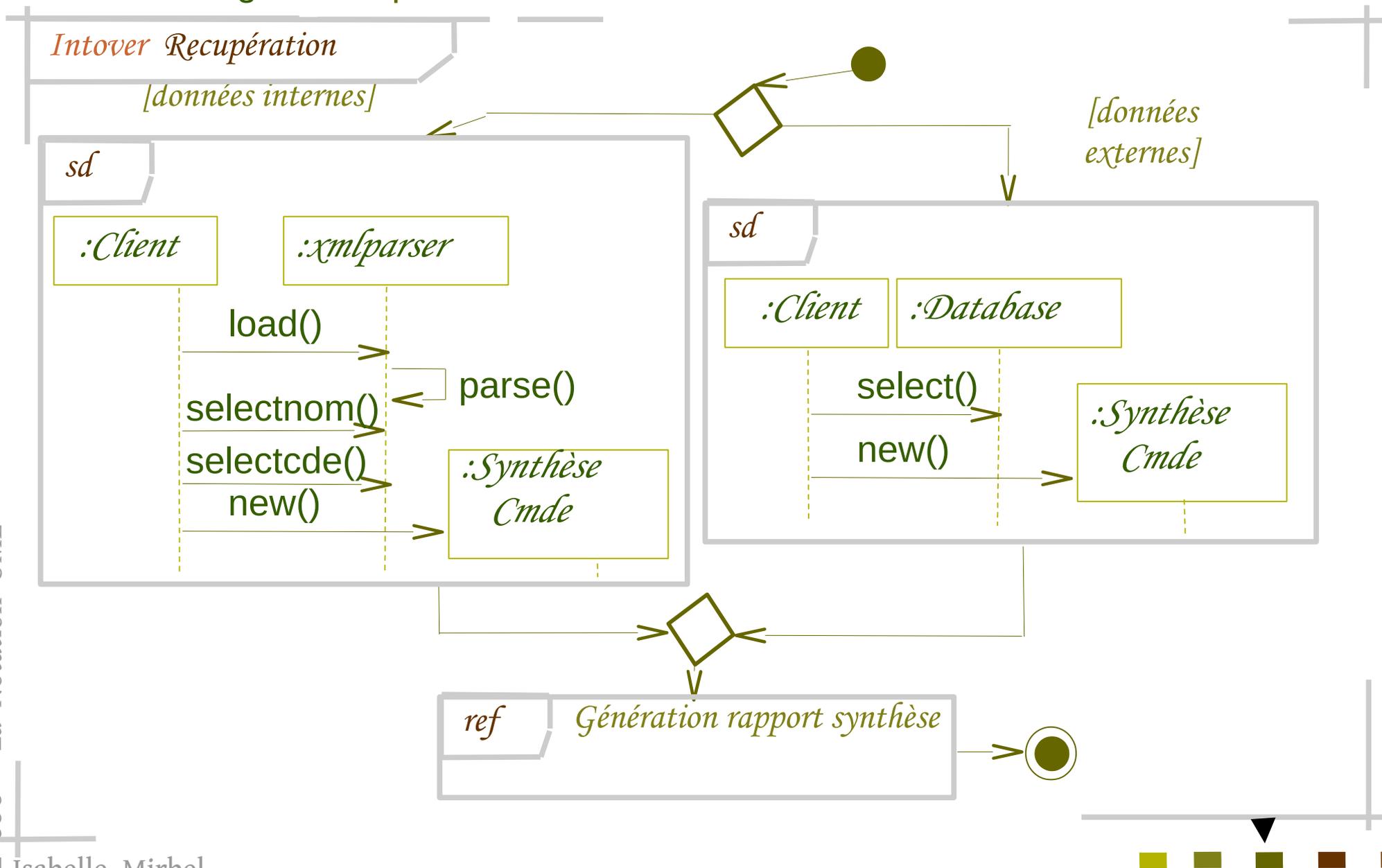
LES DIAGRAMMES COMPORTEMENTAUX

- Introduction
- Diagramme d'activité
- Diagrammes d'interaction
 - Diagramme de séquence
 - Diagramme de communication
 - ▶ ■ Diagramme global d'interaction
 - Diagramme de temps
- Diagramme de machine d'état

LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME GLOBAL D'INTERACTION

- ✓ Nouveau dans UML 2.0
- ✓ Mélange de séquence & activité



LES DIAGRAMMES COMPORTEMENTAUX

- Introduction
- Diagramme d'activité
- Diagrammes d'interaction
 - Diagramme de séquence
 - Diagramme de communication
 - Diagramme global d'interaction
 - ▶ ■ Diagramme de temps
- Diagramme de machine d'état

LES DIAGRAMMES COMPORTEMENTAUX

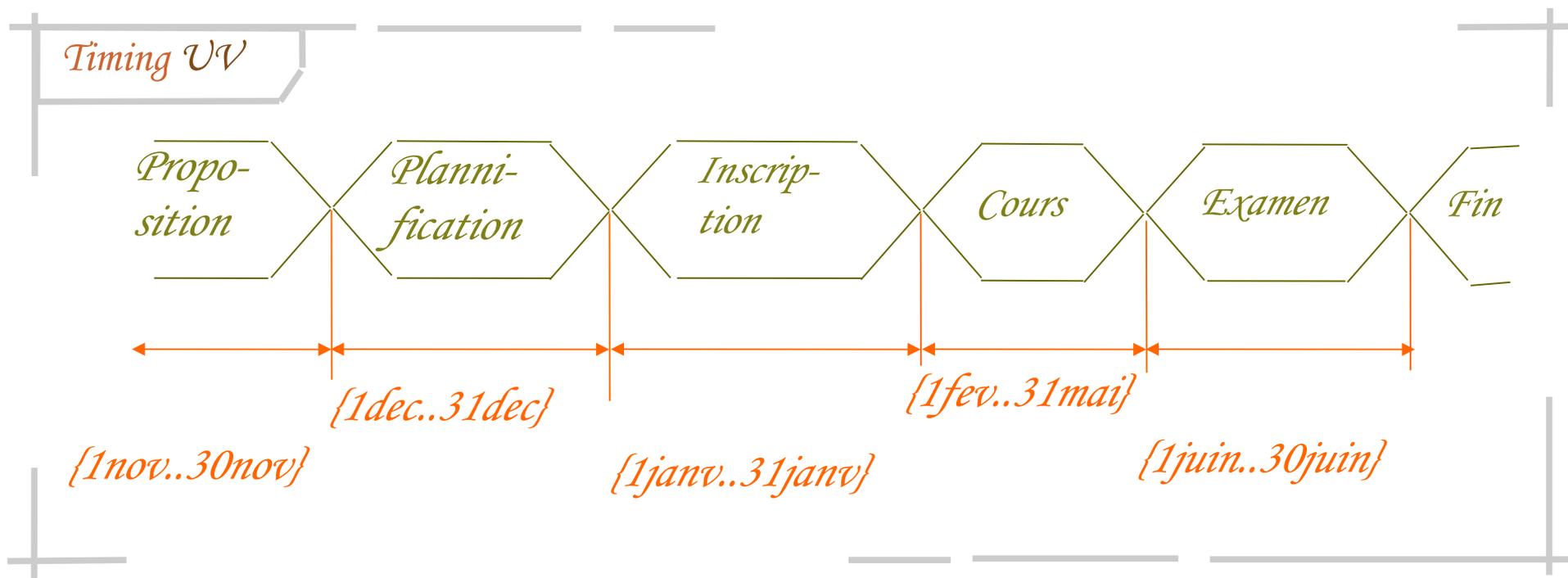
DIAGRAMME DE TEMPS

- ✓ Nouveau dans UML 2.0
- ✓ Pour montrer les contraintes de temps entre changements d'états
- ✓ A faire après les diagrammes de machine d'état
- ✓ Permet de montrer les changements d'états sur un scénario qui se déroule dans le temps
- ✓ Vient du domaine du hardware
- ✓ Plutôt pour les applications temps réel



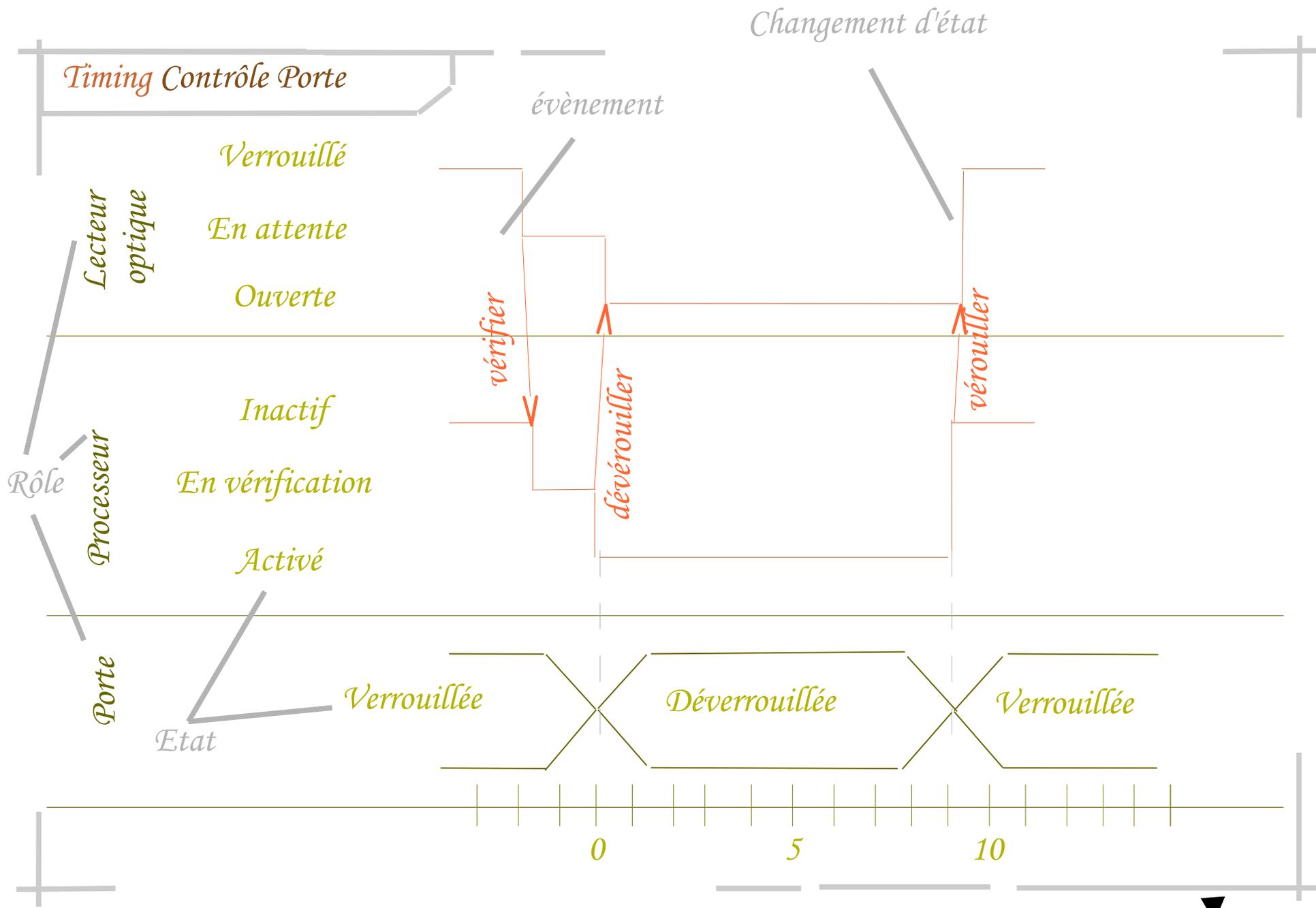
LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE TEMPS



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE TEMPS



COO - La Notation UML



LES DIAGRAMMES COMPORTEMENTAUX

- Introduction
- Diagramme d'activité
- Diagrammes d'interaction
 - Diagramme de séquence
 - Diagramme de communication
 - Diagramme global d'interaction
 - Diagramme de temps
- ▶ □ Diagramme de machine d'état

LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

- ▶ Généralités

 - Etat, activités, événement & transition

 - Evènement

 - Décompositions séquentielle & concurrente

 - Quelques conseils



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

- ✓ Représentation du comportement interne d'une classe
 - ➔ Pas de diagramme si la classe n'a pas d'états différents
 - ➔ Un diagramme est attaché à une seule classe

- ✓ Chaque objet traité de façon isolée
 - ➔ Événements de communication avec l'extérieur
 - ➔ Réponse aux événements

- ✓ Pour représenter un comportement précis
- ✓ Déconseillé pour une vue générale du système
- ✓ Représentation du cycle de vie des objets de chaque classe



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

- ✓ Pour montrer les états remarquables des objets du système
 - ➔ Pour relancer des factures
 - ➔ Permettre la destruction d'objets de façon automatique
 - ➔ ...

- ✓ Pour autoriser / interdire des traitements sur les objets
 - ➔ Un livre est-il toujours empruntable ?
 - ➔ ...



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

Généralités

- ▶ Etat, activités, événement & transition

Evènement

Décompositions séquentielle & concurrente

Quelques conseils



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

□ ETAT

- ✓ Ensemble des valeurs qui impliquent la même réponse à l'arrivée d'un événement
- ✓ Période de la vie d'un objet durant laquelle il
 - ✓ satisfait certaines conditions
 - ✓ attend un/des événement(s)
 - ✓ réalise éventuellement une activité



□ ETATS PARTICULIERS

début ●

fin ○

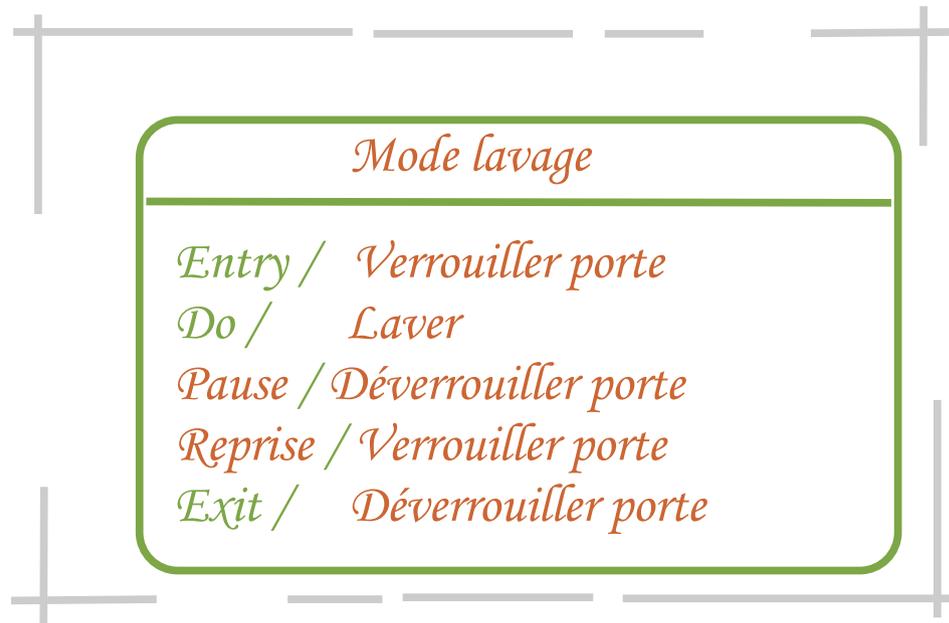


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

□ ACTIVITES

- ✓ Opérations associées à l'entrée dans l'état
 - ➔ Initialisations...
- ✓ Opérations réalisées dans l'état
 - ➔ Execution en continu
- ✓ Opérations associées à la sortie de l'état
 - ➔ Nettoyage...
- ✓ Opérations internes associées à un évènement quelconque



Activités dans diag.
de machine d'état
≠
Activités dans diag.
d'activités



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

□ EVENEMENT

✓ Toute influence extérieure est représentée par un événement

- ✓ Point dans le temps
- ✓ Pas de durée
- ✓ Notion d'instance d'événement

- ✓ Asynchrone
- ✓ Atomique
- ✓ Uni-directionnel

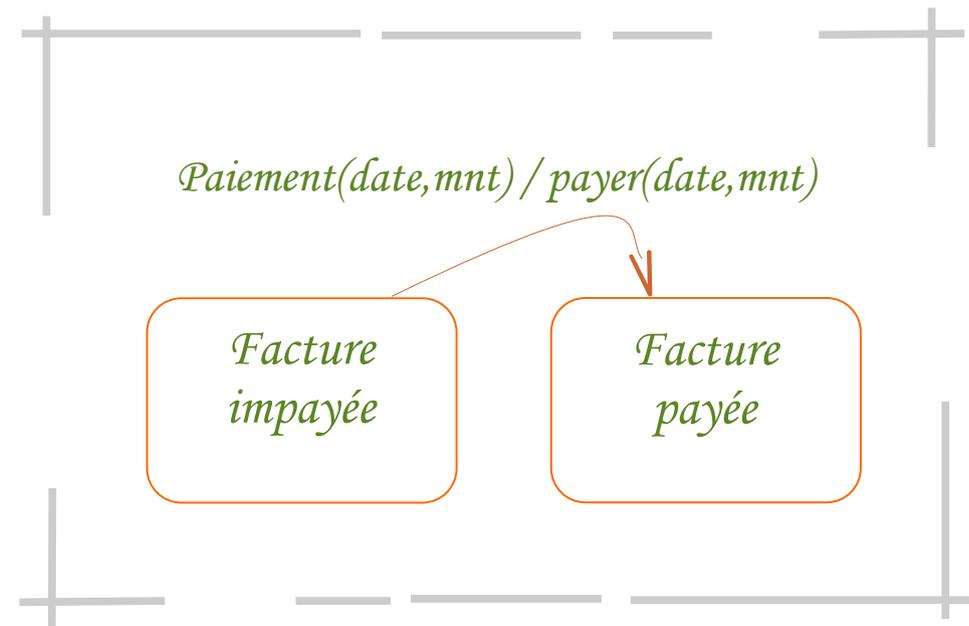


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

□ TRANSITION

- ✓ Pour connecter 2 états
- ✓ Réponse de l'objet à un événement



- ✓ Un événement ne peut activer qu'une seule transition
- ✓ Une transition correspond à un seul événement
- ✓ L'opération peut porter sur/utiliser
 - ✓ les paramètres de l'événement
 - ✓ les attributs de la classe



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

□ TRANSITION (suite)

- ✓ Syntaxe complète d'une transition

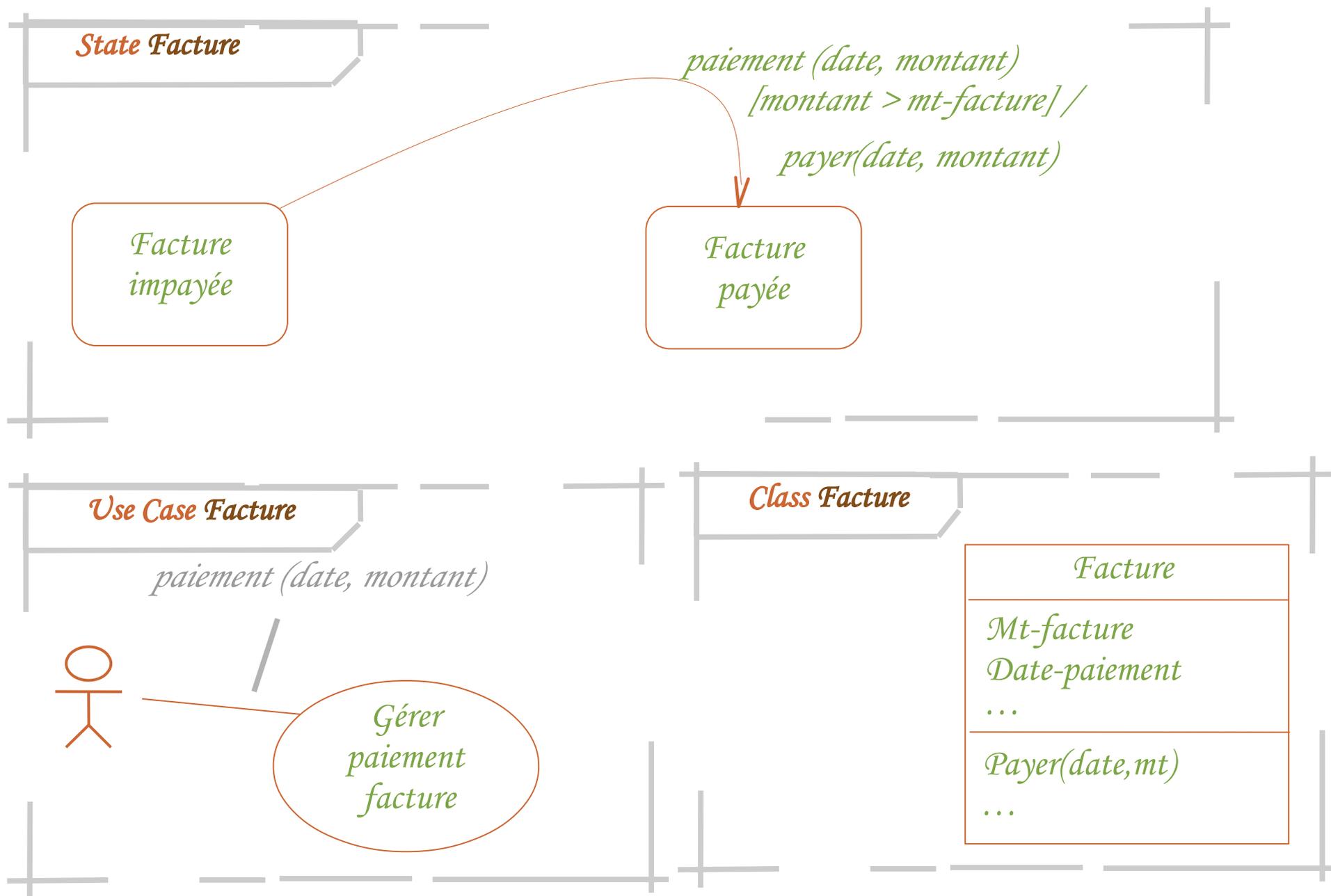
Evénement (args) garde / Action(args)

- ✓ Entre []
- ✓ Expression booléenne
- ✓ Fonction
 - ✓ des paramètres de l'événement
 - ✓ des valeurs de l'objet
 - ✓ de l'état (**in / not in**) d'un objet accessible



LES DIAGRAMMES COMPORTEMENTAUX

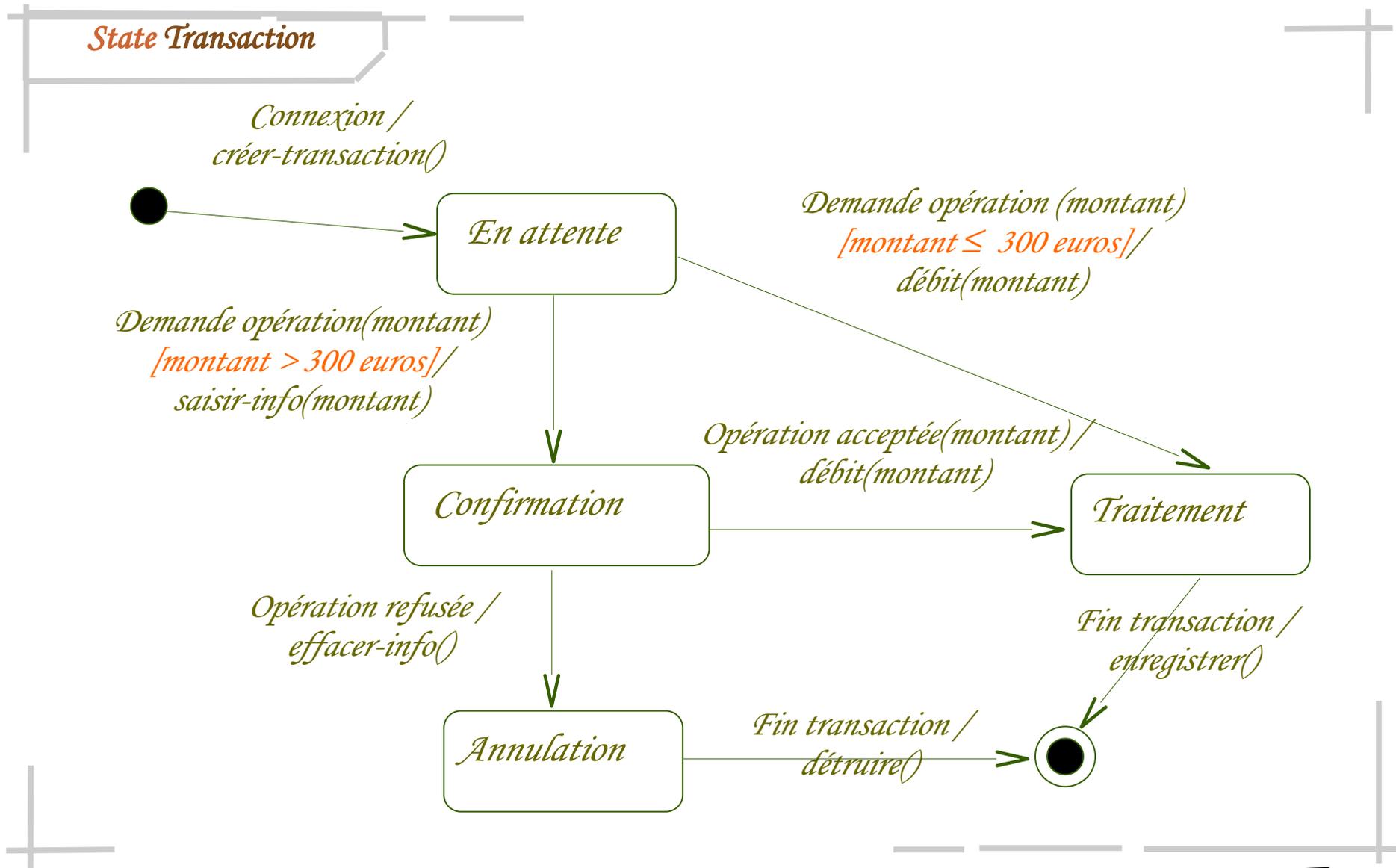
DIAGRAMME DE MACHINE D'ETAT



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

- ✓ Un événement donné peut permettre de quitter l'état initial pour plusieurs autres → Gardes exclusives



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

Généralités

Etat, activités, événement & transition

▶ Précisions sur les évènements

Décompositions séquentielle & concurrente

Quelques conseils



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

□ EVENEMENT (suite)

- ✓ Appel d'une opération
 - ✓ Signal/appel de la forme:
 - ➔ `NomÉvénement(param, param, param, ...)`
 - ✓ Paramètre de la forme:
 - ➔ `NomParam:type`
- ✓ Réception d'un signal
- ✓ Condition qui devient vraie
 - ➔ `When expression-booléenne`



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

□ EVENEMENT (suite)

✓ Écoulement d'une période de temps *Paiement / payer(date, mnt)*

➔ After (5 secondes)

➔ After (10 sec après être sorti de l'état 1)

✓ Évènement *ANY*



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

□ EVENEMENT (suite)

- ✓ Description des événements
 - ✓ sous forme de classe
 - ➔ pour donner plus d'informations
 - ➔ pour mieux gérer la cohérence entre diagrammes état-transition
 - ✓ avec généralisation
 - ➔ pour hiérarchiser les différents événements
 - ✓ en utilisant des stéréotypes
 - ➔ pour représenter des signaux, ...



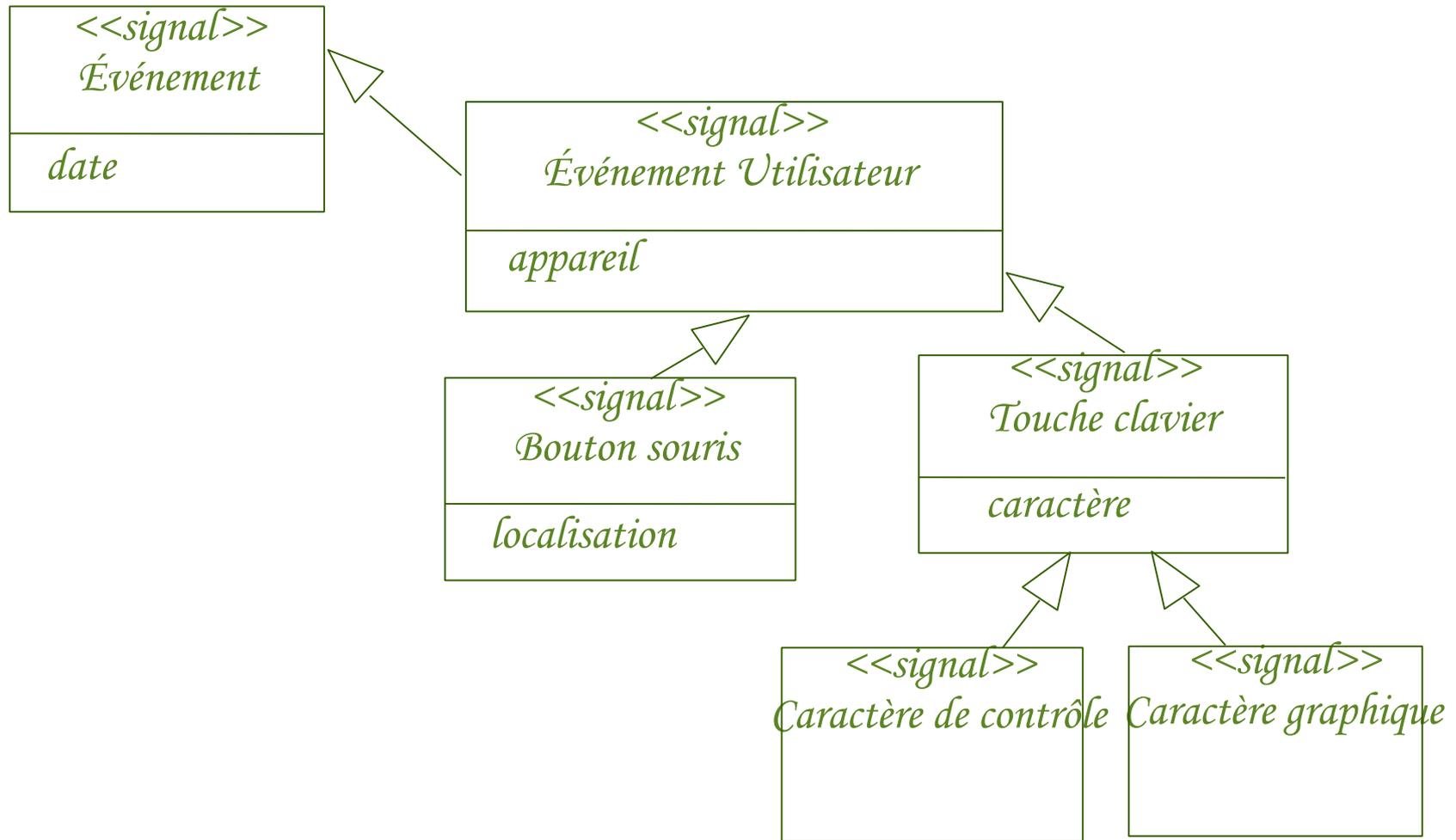
LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

□ SIGNAL

- ✓ Asynchrone
- ✓ Peut transporter des paramètres

- ➔ Diag. d'architecture
- ➔ Diag. de séquence
- ➔ Diag. d'état



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

Généralités

Etat, activités, événement & transition

Précisions sur les évènements

- ▶ Décompositions séquentielle & concurrente

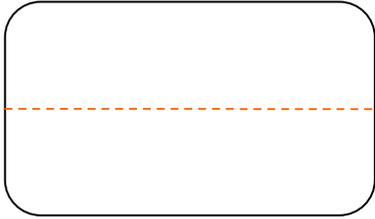
Quelques conseils



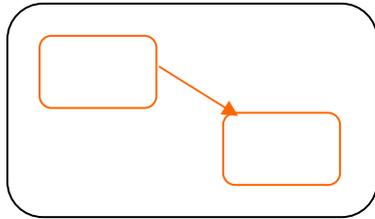
LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

□ DECOMPOSITION D'ETAT



✓ Sous-états concurrents



✓ Sous-états séquentiels

✓ Un événement peut être utilisé à plusieurs niveaux

➔ L'événement de plus haut niveau (imbrication minimum) l'emporte



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

POINTS D'ENTREE & SORTIE

✓ Sous-machine d'état



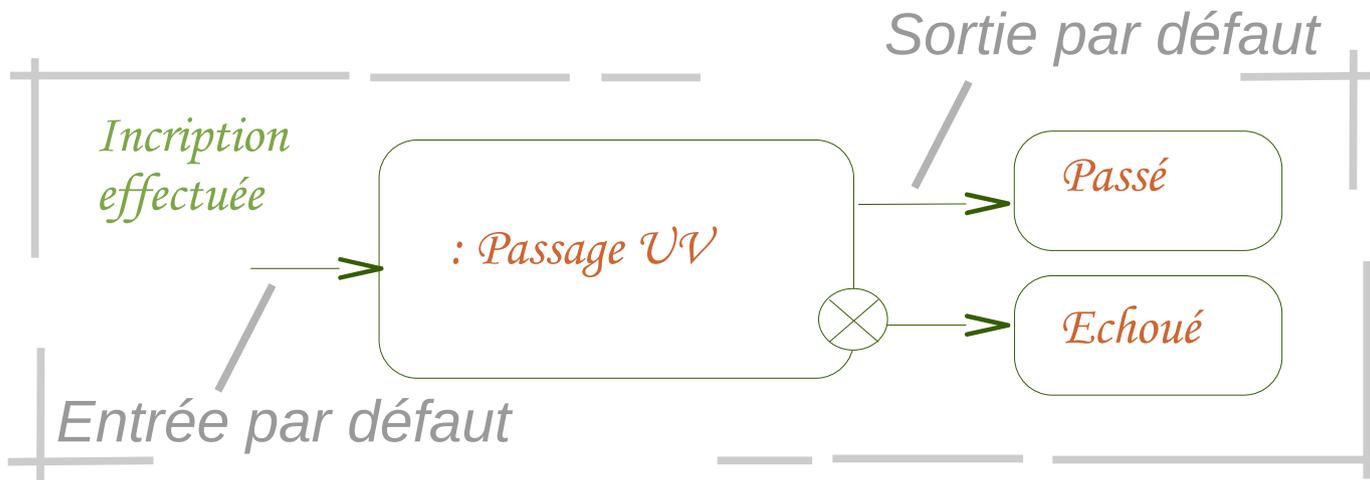
✓ Point d'entrée



✓ Point de sortie



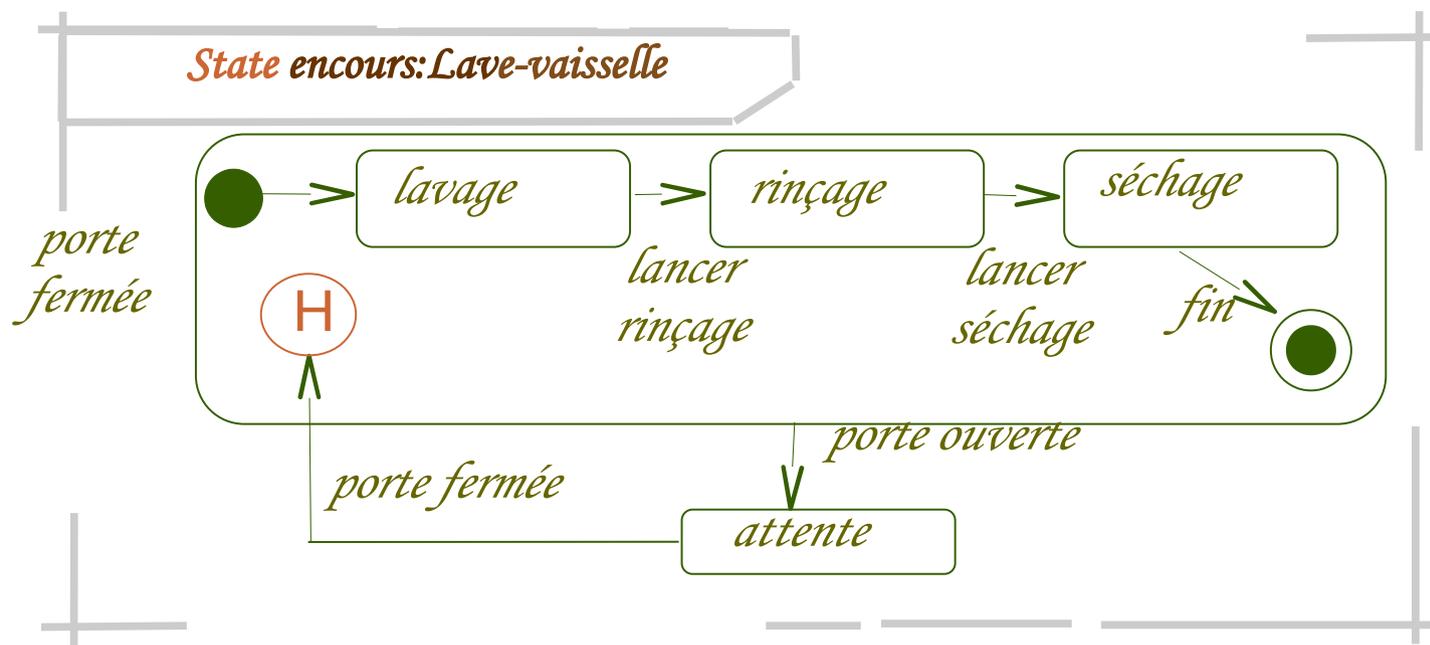
➔ Permet d'encapsuler les traitements & de rendre explicites les entrées & sorties exceptionnelles



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

□ HISTORIQUE D'ETAT



*H**

∇ profondeur d'emboîtement

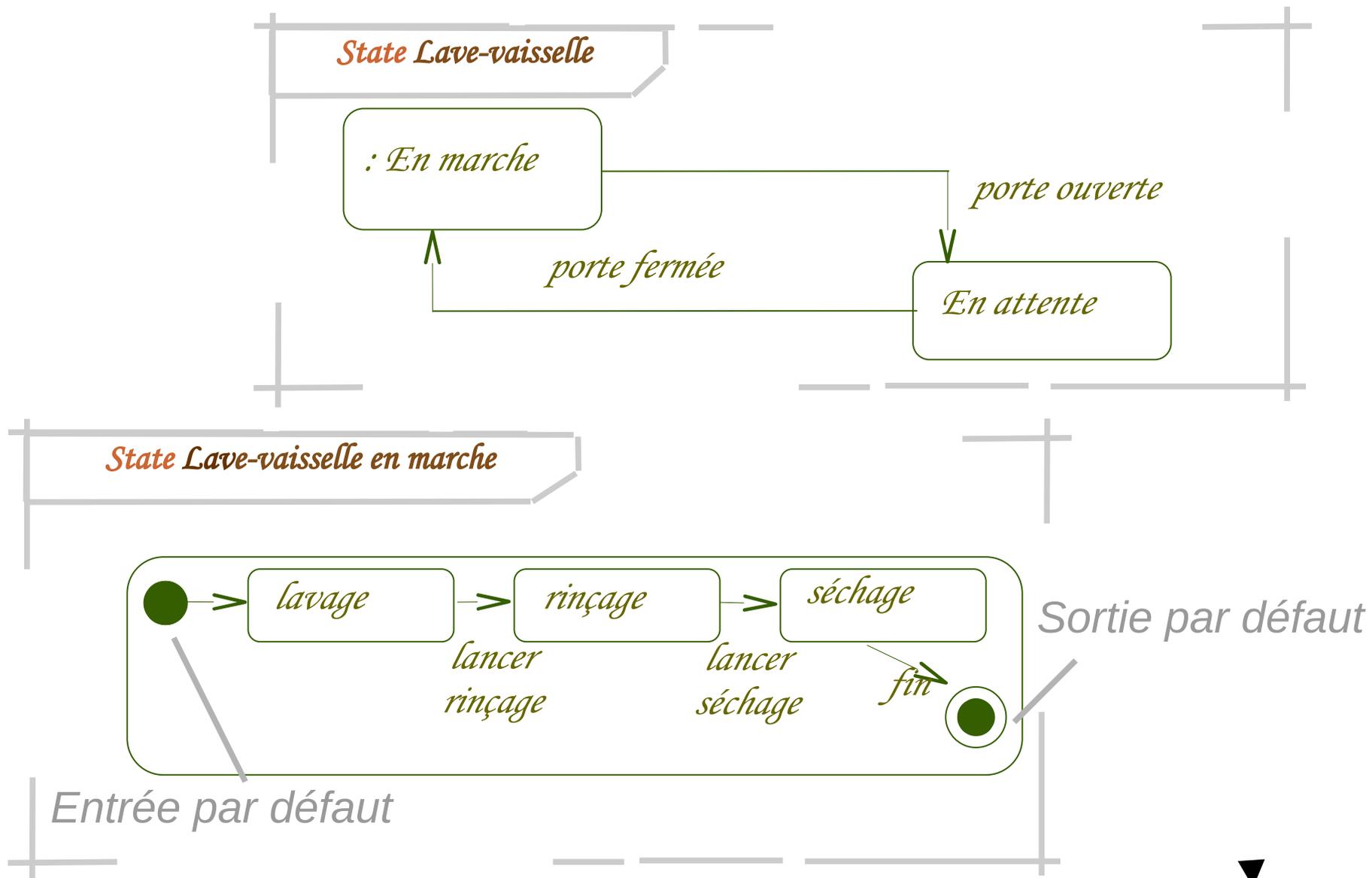


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

□ DECOMPOSITION SEQUENTIELLE

Description du fonctionnement d'un lave-vaisselle

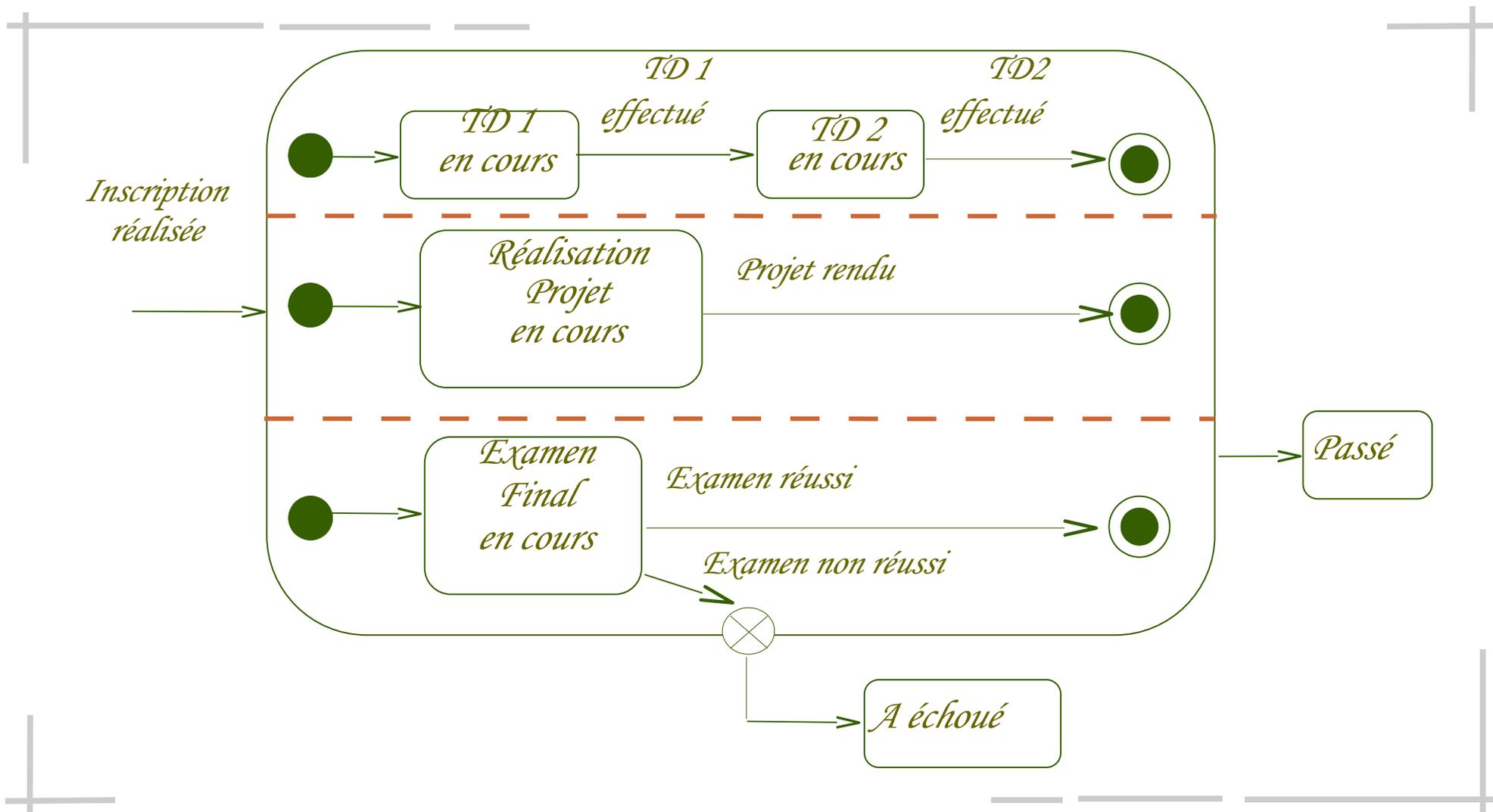


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

□ DECOMPOSITION CONCURRENTE

Description de la validation d'une unité de valeur



LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

Généralités

Etat, activités, événement & transition

Précisions sur les évènements

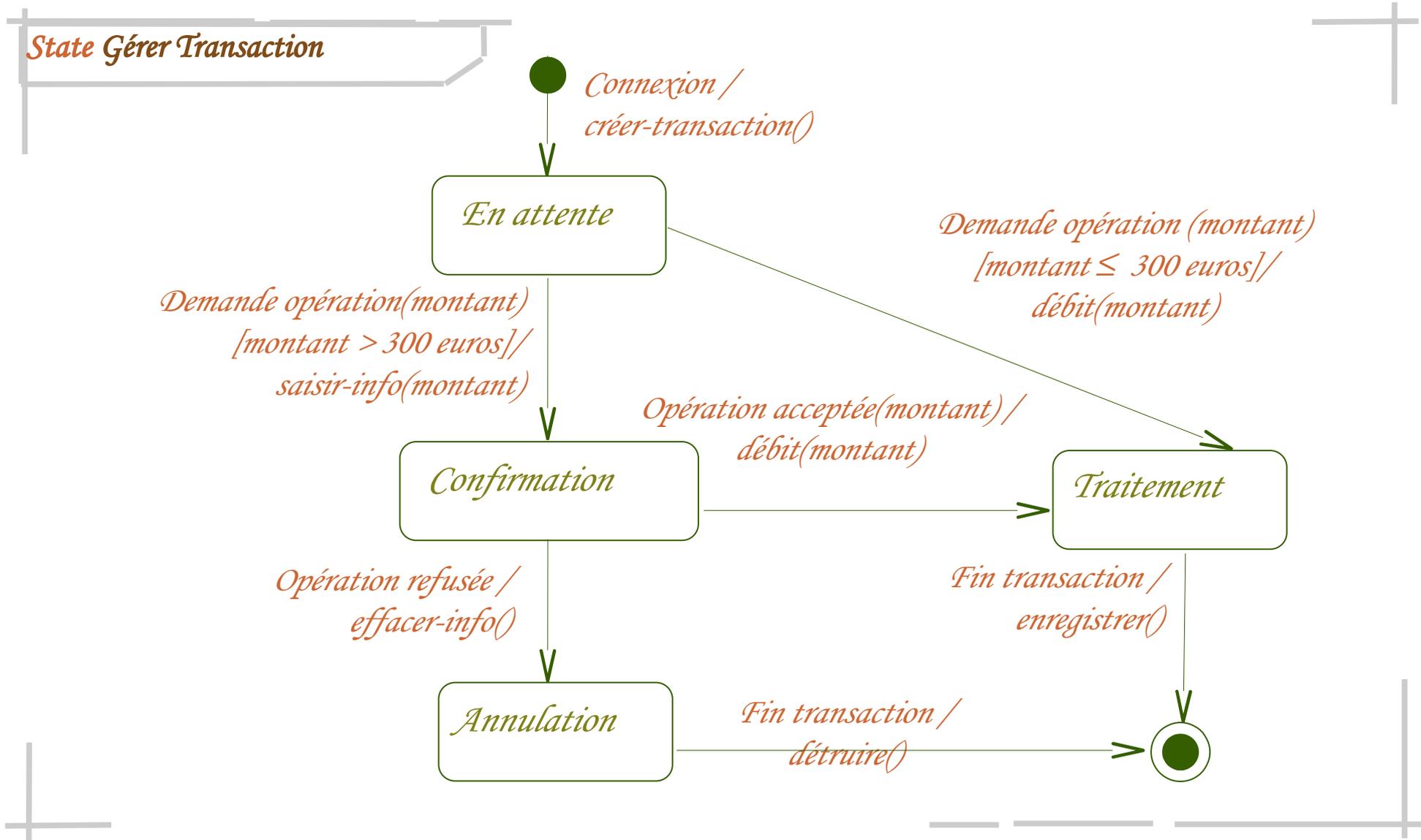
Décompositions séquentielle & concurrente

- ▶ Quelques conseils



LES DIAGRAMMES COMPORTEMENTAUX

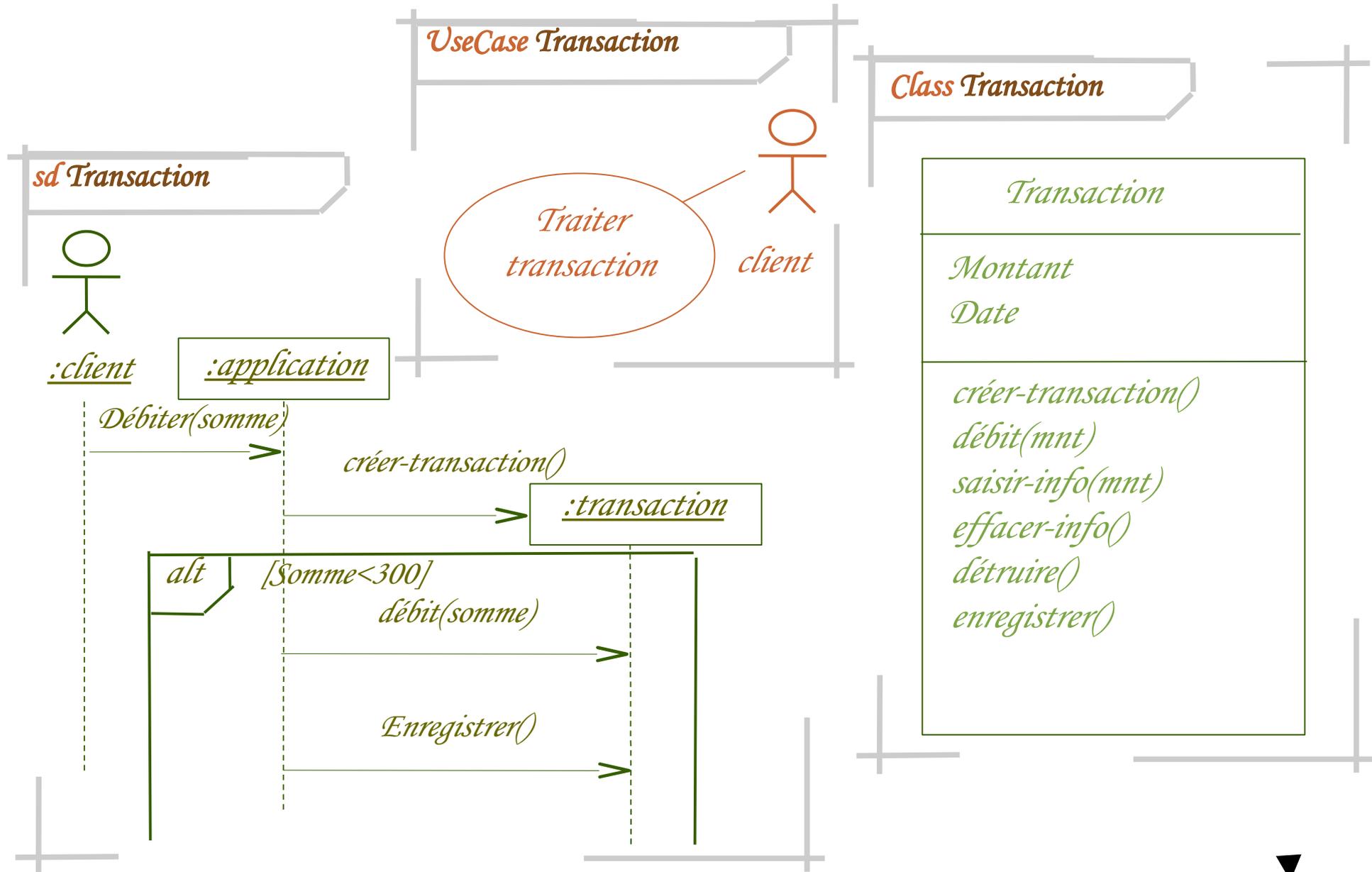
DIAGRAMME DE MACHINE D'ETAT

□ COHERENCE ENTRE DIAGRAMMES


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

COHERENCE ENTRE DIAGRAMMES (suite)

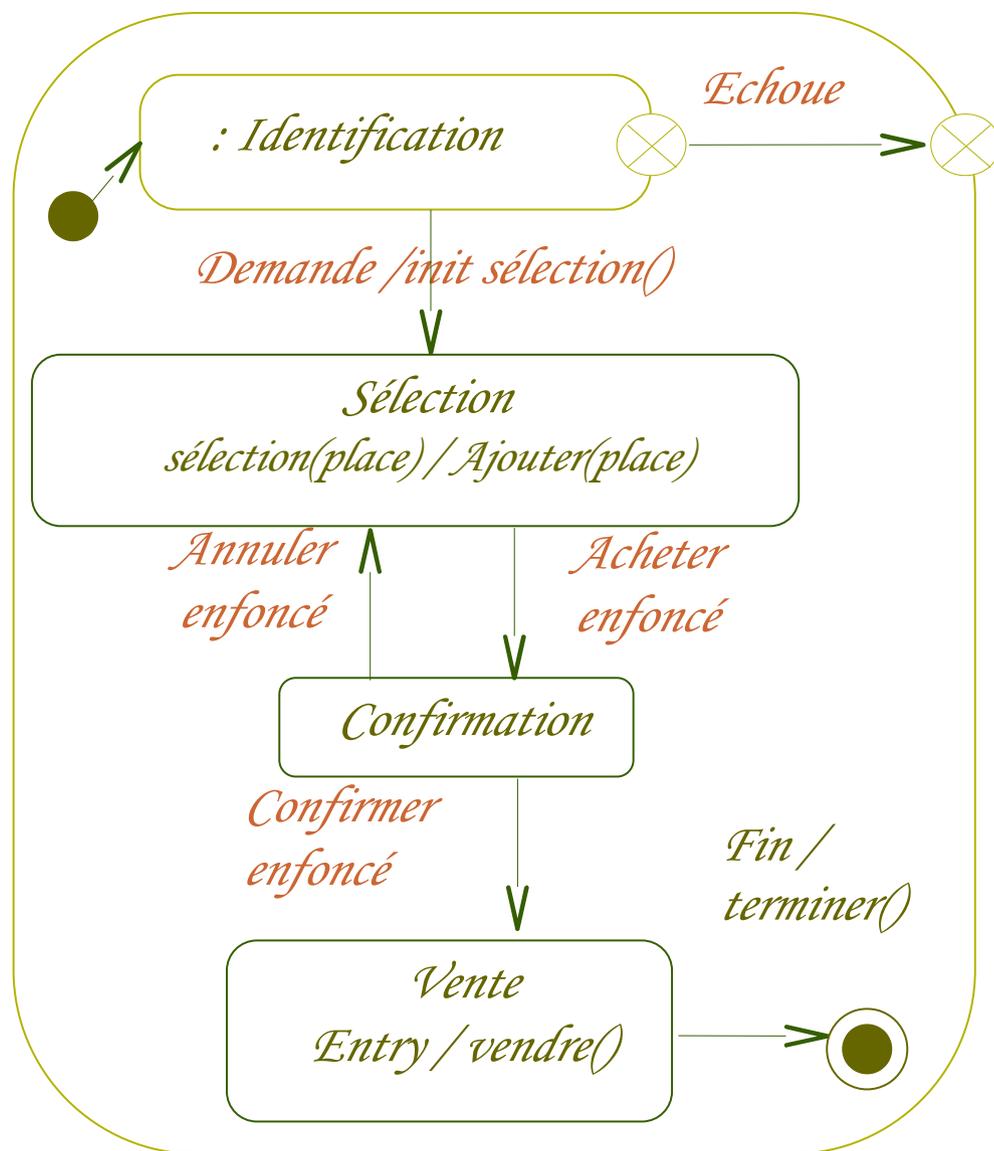
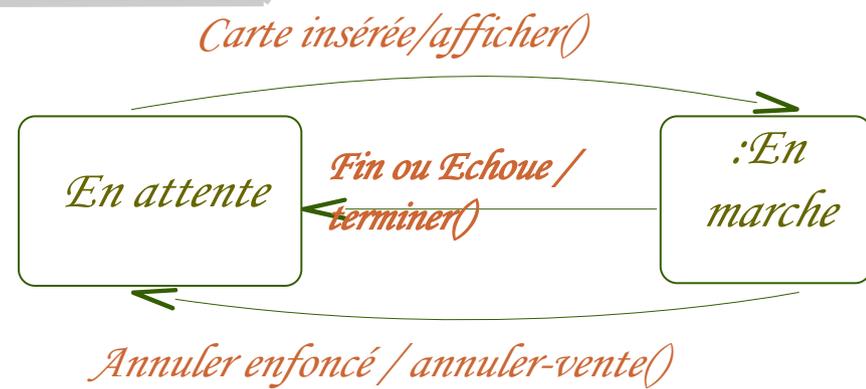


LES DIAGRAMMES COMPORTEMENTAUX

DIAGRAMME DE MACHINE D'ETAT

State DAB En Marche

□ EXEMPLE

*State DAB*

P L A N

- Introduction
- Les diagrammes structurels
- Les diagrammes comportementaux
- ▶ ■ Le diagramme de paquetage
- Conclusion

LE DIAGRAMME DE PAQUETAGE

► Paquetage

Dépendance

Paquetage type



LE DIAGRAMME DE PAQUETAGE

- ✓ Système complexe découpé en unités **plus petites**
 - ✓ pour permettre un travail **efficace** (quantité information)
 - ✓ pour permettre un travail **en parallèle** (minimiser interférences)

- ✓ Paquetages & dépendances
 - ✓ Fonctionnalités communes
 - ✓ Implémentations couplées
 - ✓ Points de vue communs

- ✓ Espace de nommage

- ✓ Reflètent l'**architecture** haut niveau du système



LE DIAGRAMME DE PAQUETAGE

Paquetage



- ✓ Diagrammes de classes
- ✓ Diagrammes de machines d'état
- ✓ Diagrammes de cas d'utilisation
- ✓ Diagrammes d'interaction
- ✓ ...
- ✓ autres paquetages

✓ Chaque élément

- ✓ n'est défini **qu'une seule fois**
- ✓ peut apparaître à plusieurs endroits



LE DIAGRAMME DE PAQUETAGE

Paquetage

▶ Dépendance

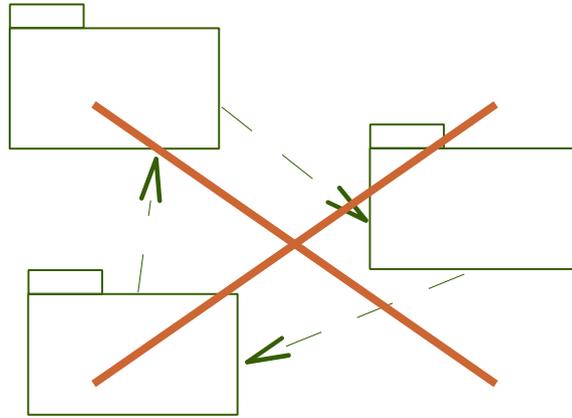
Paquetage type



LE DIAGRAMME DE PAQUETAGE

□ DEPENDANCES & PAQUETAGES

✓ Eviter les cycles



- ✓ Un package avec beaucoup de dépendances doit avoir une interface stable
- ✓ <<global>>

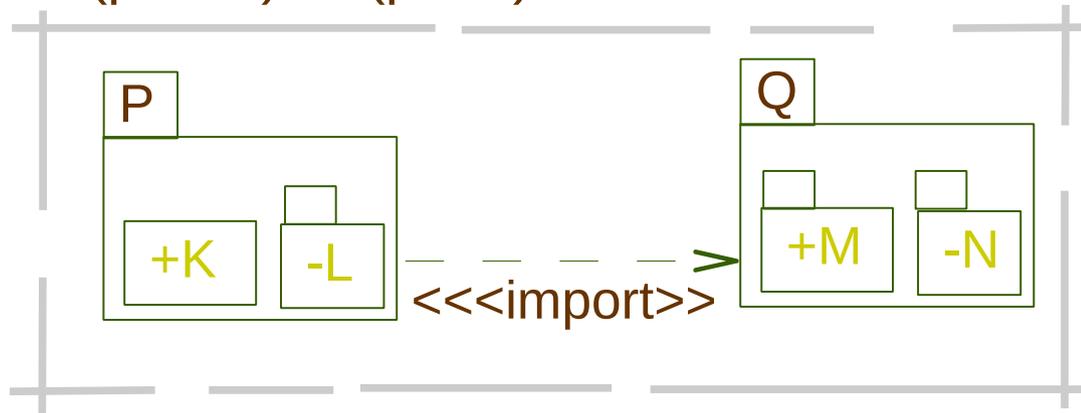
LE DIAGRAMME DE PAQUETAGE

□ DEPENDANCES & PAQUETAGES (suite)

- ✓ Import {Import nomqualifié}
- ✓ Access

Variante de la dépendance d'importation pour utiliser les noms d'un autre espace de nom pour en référencer des éléments

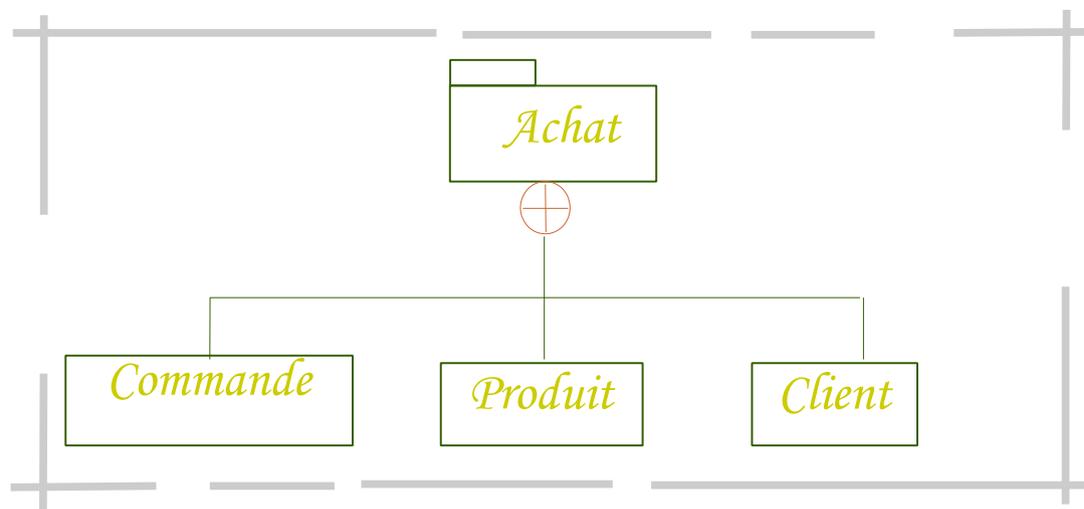
- ✓ + (public) / - (privé)



LE DIAGRAMME DE PAQUETAGE

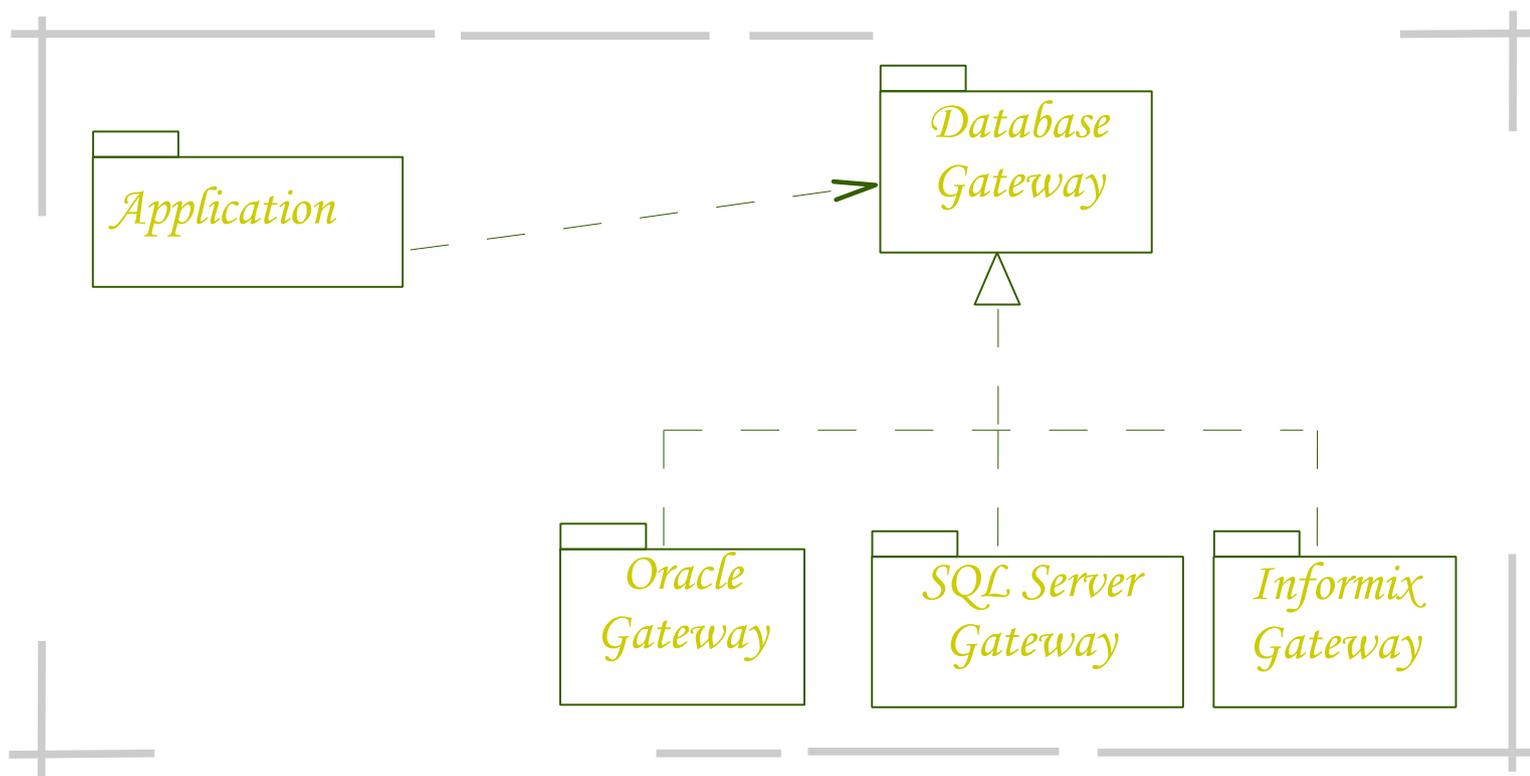
□ DEPENDANCES & PAQUETAGES (suite)

- ✓ Contenu d'un paquetage / notation externe



LE DIAGRAMME DE PAQUETAGE

EXEMPLE DE DEPENDANCES



LE DIAGRAMME DE PAQUETAGE

Paquetage

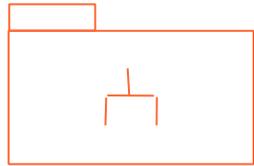
Dépendance

▶ Paquetage type



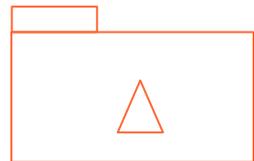
LE DIAGRAMME DE PAQUETAGE

□ PAQUETAGES TYPES



→ Systèmes / Sous-systèmes

- ✓ Spécification
- ✓ Réalisation



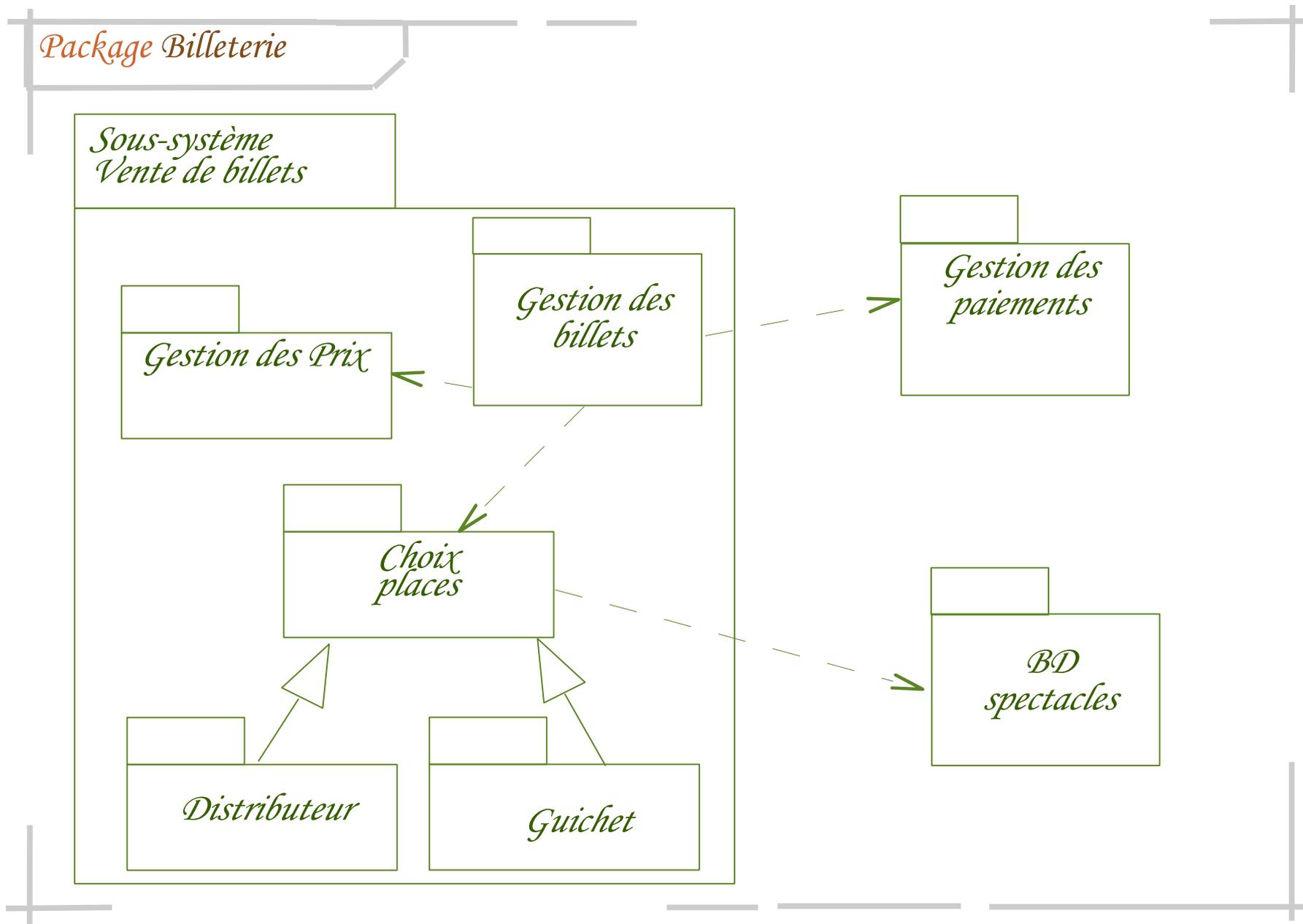
→ Modèles

- ✓ Abstraction de l'application pour la spécifier selon un certain point de vue, à un certain niveau d'abstraction & de détail

- ✓ Pour donner différentes vues de l'application
 - Phase du processus de développement
 - Destinataire de la modélisation
 - Aspect de l'application

- ✓ Combinaison des deux types de paquetage possible





P L A N

- Introduction
- Les diagrammes structurels
- Les diagrammes comportementaux
- Le diagramme de paquetage
- ▶ ■ Conclusion

CONCLUSION

- UML est un support à une approche systématique pour
 - ✓ Identifier les problèmes, les solutions et les objectifs
 - ✓ Analyser les flux d'information dans les organisations
 - ✓ Concenvoir un système d'information (informatique) pour
 - ✓ Résoudre un problème
 - ✓ Répondre à un besoin



CONCLUSION

▣ OBJECTIFS DE LA MODELISATION

- ✓ Communiquer une structure et un comportement souhaités
- ✓ Visualiser et contrôler l'architecture du système
- ✓ Mieux comprendre le système en construction
- ✓ Mettre en évidence des simplifications et réutilisations
- ✓ Documenter les décisions prises
- ✓ Mieux gérer les risques



CONCLUSION

□ PLUSIEURS DIAGRAMMES POUR COUVRIR

- ✓ Différents types de développements
- ✓ Différentes étapes de développement
 - ➔ Tous les diagrammes ne sont pas forcément nécessaires
 - ➔ Attention à la cohérence entre diagrammes

□ RASSEMBLER LES DIAGRAMMES POUR

- ✓ Consigner les résultats
- ✓ Expliquer & justifier les choix
- ✓ Permettre les évolutions



CONCLUSION

□ POURQUOI UML ?

✓ Activité d'analyse

- ➔ Pour clarifier un processus (vu de l'extérieur)
- ➔ Pour anticiper les tests (diagrammes d'interaction)

✓ Activité de développement

- ➔ Conception orientée objet pour
 - ✓ les langages orientés objet
 - ✓ les bases de données
 - ✓ ...
- ➔ Génération de code / rétro-conception

✓ Architecture système



CONCLUSION

□ UML POUR L'ANALYSE & LA CONCEPTION DE SI

- ✓ Evolution de la notation UML dirigée par
 - ➔ Les langages de programmation
 - ➔ Les technologies

- ✓ Phase d'analyse moins bien couverte que phase de conception
 - ➔ Langages spécifiques
 - ➔ Méthodologies adaptées



BIBLIOGRAPHIE

- ❑ UML 2 en action – Pascal Roques, Franck Vallée - Eyrolles

- ❑ UML Distilled: A Brief Guide to the Standard Object Modeling Language (2nd Edition) - Martin Fowler, Kendall Scott - Addison - Wesley

- ❑ UML pour les décideurs – Franck Vallée - Eyrolles

- ✓ Spécifications d'UML
 - ✓ Object Management Group: <http://www.omg.org>

- ✓ Logiciels pour mettre en œuvre UML
 - ✓ ArgoUML, BOUML, Magic Draw, Poseidon, Umbrello, Visual Paradigm, Violet...

