

Modélisation

Analyse et Conception Orientée Objet avec UML.2

Unified Modeling Language (UML)
Langage de Modélisation unifié

Bouchra BERRADA

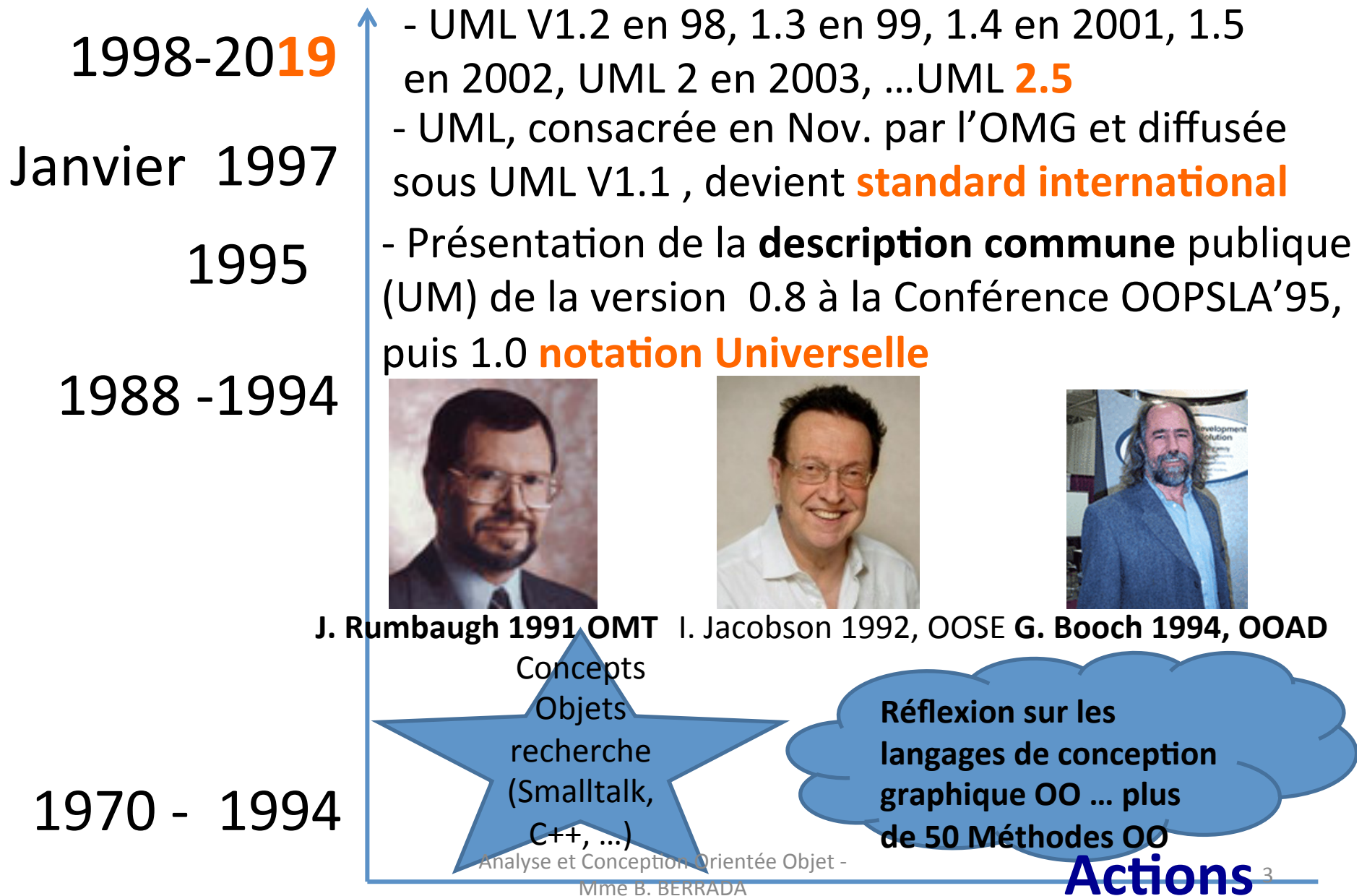
email : nouveau.berrada.bouchra@gmail.com

Sommaire

- Genèse d'UML
- Qu'est-ce UML ?
- Pourquoi UML ?
- Les Diagrammes UML 2
- Les point de vues UML 2

Temps

Genèse UML...



Qu'est-ce qu'UML ?

- Un langage graphique de modélisation objet
- Une famille de notation graphique s'appuyant sur un méta-modèle unique
- Un support méthodologique pour les concepteurs et développeurs pour formaliser l'analyse et la conception technique du logiciel orienté objet en particulier
 - Un standard car s'appuyant sur une norme structurante
- Une méthode ... ?

UML, Suite

METHODE = Langage de modélisation + ~~PROCESSUS~~

Par conséquent :

- UML est un langage de modélisation, pas une méthode
- UML est adaptable car peut être utilisé avec un processus de développement existant
- Deux processus : Unified Process (UP) et Rational Unified Process (RUP) sont associés à UML :
 - Guidé par les besoins des utilisateurs du système,
 - Itératif et incrémental
 - Centré sur l'architecture logicielle,
 - Orienté par la réduction des risques

Les processus étant fondés sur :

- Quatre concepts
- Deux dimensions

Pourquoi UML ?

- Le plus connu et utilisé dans le monde
- Le marché est important et s'accroît
- UML est adaptable
- UML, étant un standard ouvert, contrôlé par l'OMG, prône l'interopérabilité, et spécifiquement entre systèmes orientés objet
- UML est dans le domaine public

**UML est un langage standard
de modélisation orientée objet**

Approche Orientée Objet

Définition

Organisation d'un logiciel sous la forme d'une collection d'objets indépendants incorporant *structure de données* et *comportement*.

Objet :

- Entité identifiable du monde réel, ayant une existence physique ou pas

- Exemples :
 - *Ce transparent ;*
 - *Enseignant.e M.me X. YZ ;*
 - *Stratégie pédagogique ;*
 - *Texte de loi ;*
 - *Réunion*

Présentation Générale

Diagrammes UML

UML, dans sa version 2, propose 14 diagrammes

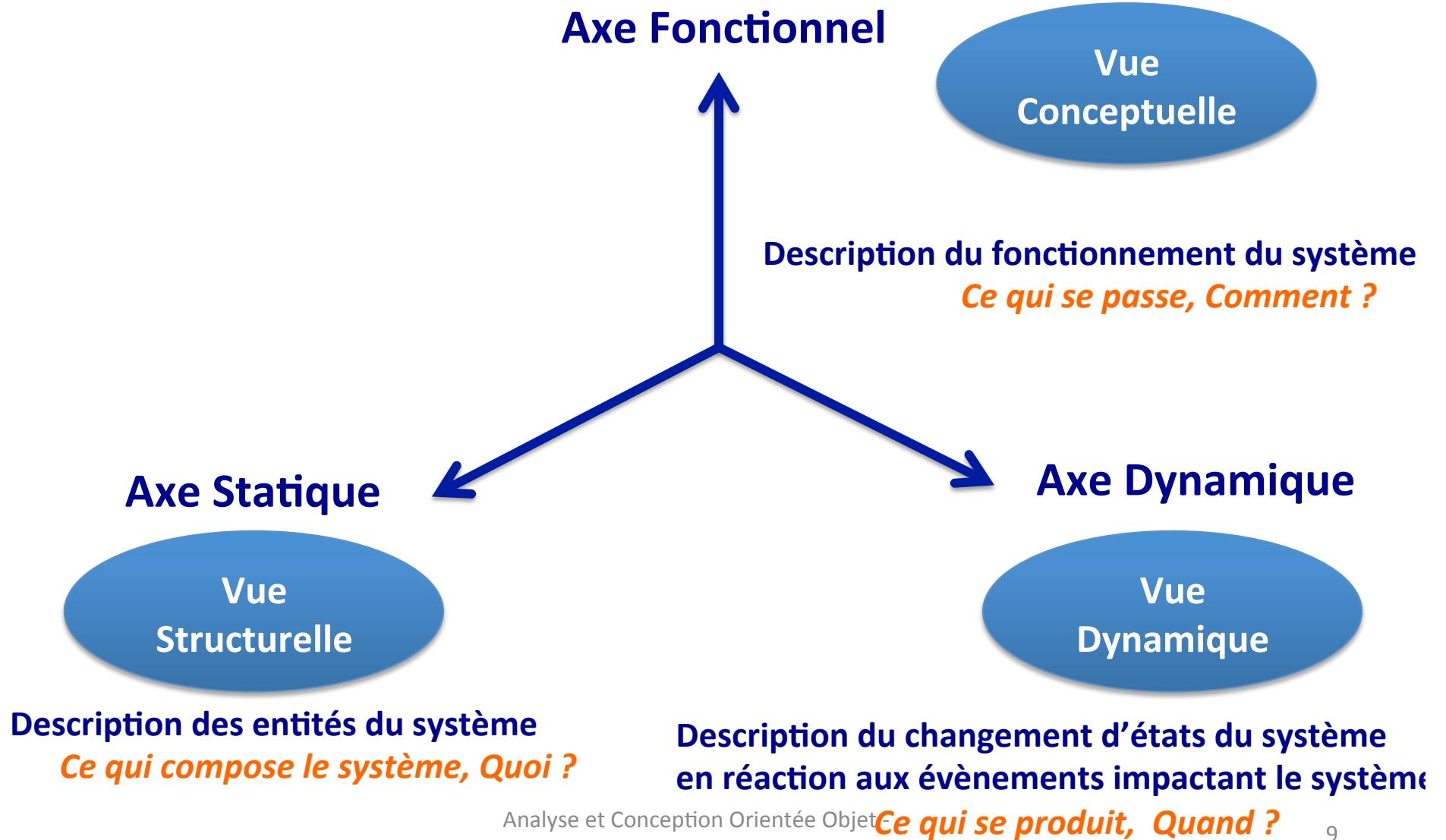
- **dépendants** hiérarchiquement et
- **complémentaires**, de façon à permettre la modélisation d'un projet tout au long de son cycle de vie.

Ces diagrammes sont regroupés dans **2 grands ensembles** :

1. **Diagrammes STRUCTURELS** : *représentant l'aspect statique (Structure Diagram)*
2. **Diagrammes COMPORTEMENTAUX** : *représentant l'aspect dynamique (Behavior Diagram)*

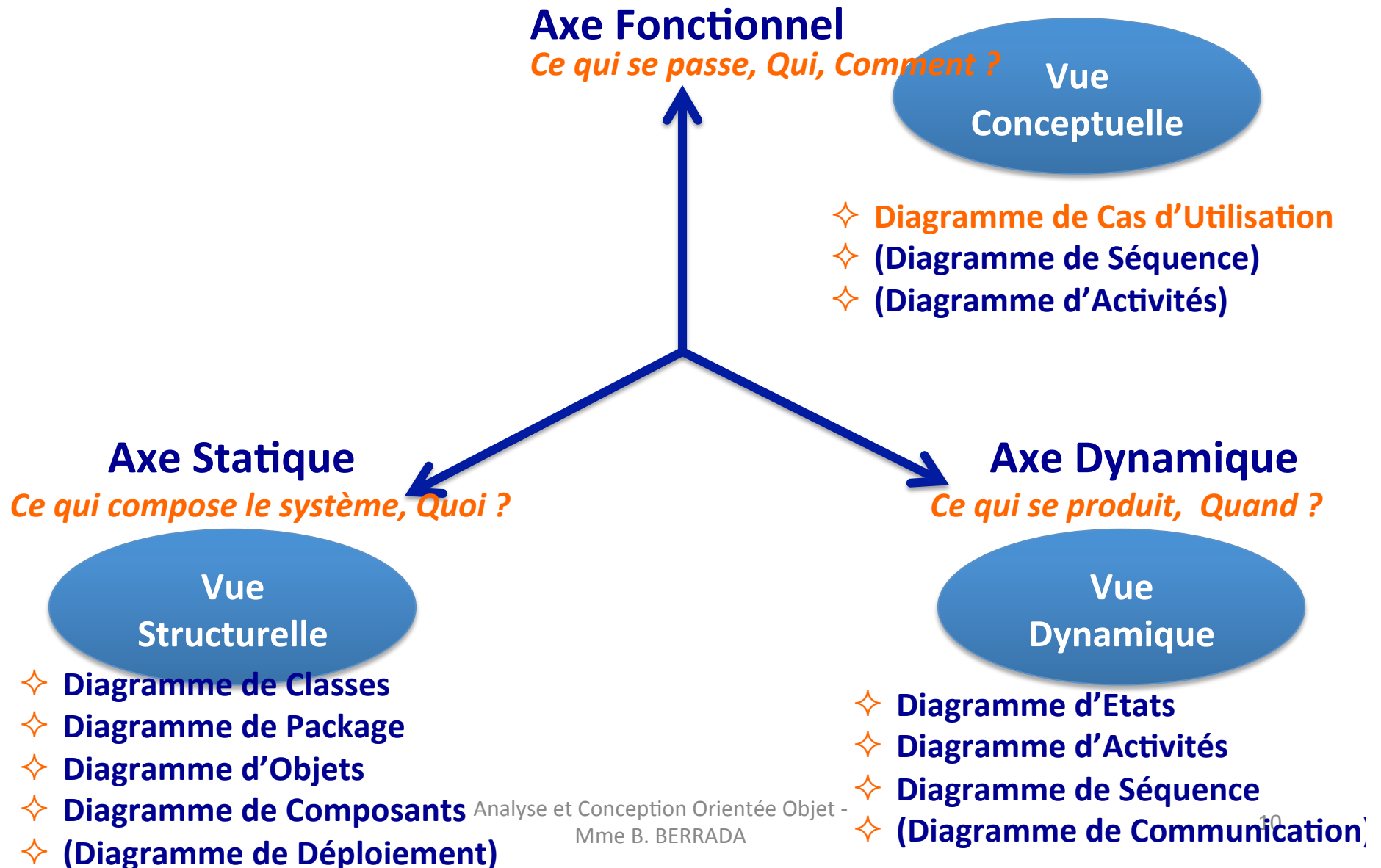
Diagrammes UML

3 Points de vue



Diagrammes UML

3 Points de vue



Description des Diagrammes UML 2

Diagramme	Finalité	Historique
Axe fonctionnel		
Cas d'utilisation (DCU) ou Use Case (UCD)	<p>Interactions entre les utilisateurs et le Système, illustrant les différentes fonctionnalités (scénarios).</p> <p>Il constitue l'un des diagrammes des plus structurants dans l'analyse d'un système.</p>	UML 1
Séquence (DSE)	Il décrit les Interactions de chaque scénario d'un CU en mettant l'accent sur la chronologie des opérations en interactions avec les objets.	UML 1, 2
Activité (DAC)	<p>Il donne une vision des enchaînements des activités propres à une opération ou à un CU.</p> <p>Il permet aussi de représenter les flots de contrôle et les flots de données.</p>	UML 1, 2
Analyse et Conception Orientée Objet - Mme B. BERRADA		11

Suite - Description des Diagrammes UML 2

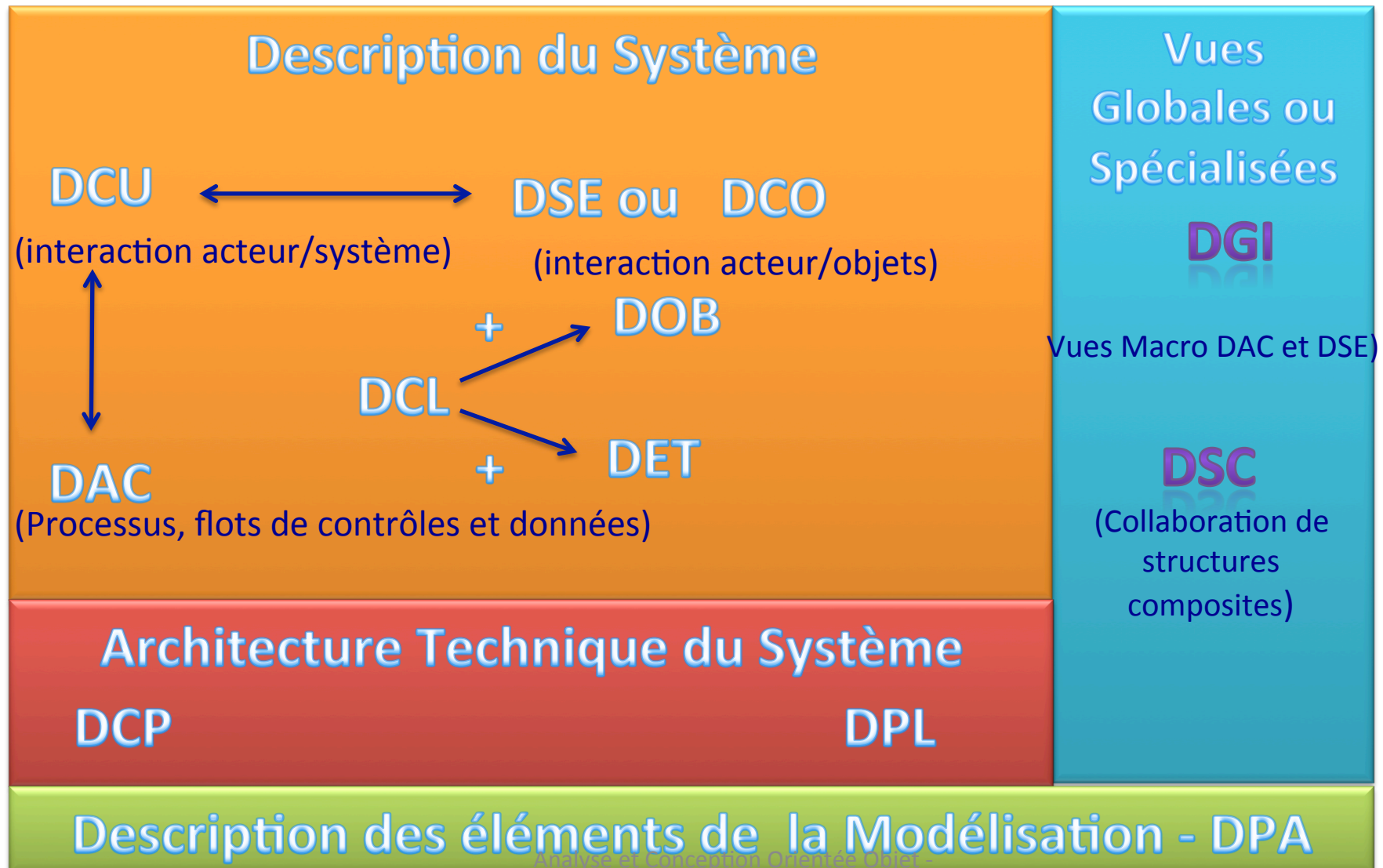
Diagramme	Finalité	Historique
Axe Statique (6 Diagrammes Structurels)		
Classes (DCL)	Description statique du système, intégrant dans chaque classe, ses caractéristiques et les relations statiques.	UML 1, 2
Objets (DOB)	Représentation des Instances de Classes et de leurs liens.	UML 1, 2
Packages (DPA)	Vue d'ensemble du système, constitué d'éléments homogènes du système (Classes, composants)	UML 2
Composant (DCP)	Représentation des constituants du logiciel au niveau implémentation d'un système.	UML 1, 2
Structure composite (DSP)	Description des collaborations d'instances constituants les fonctions du système à réaliser	UML 2
Déploiement (DPL)	Description de l'architecture physique avec vue centrée sur la répartition des composants - nœuds de traitements (device), ou nœud d'environnement d'exécution	UML 2

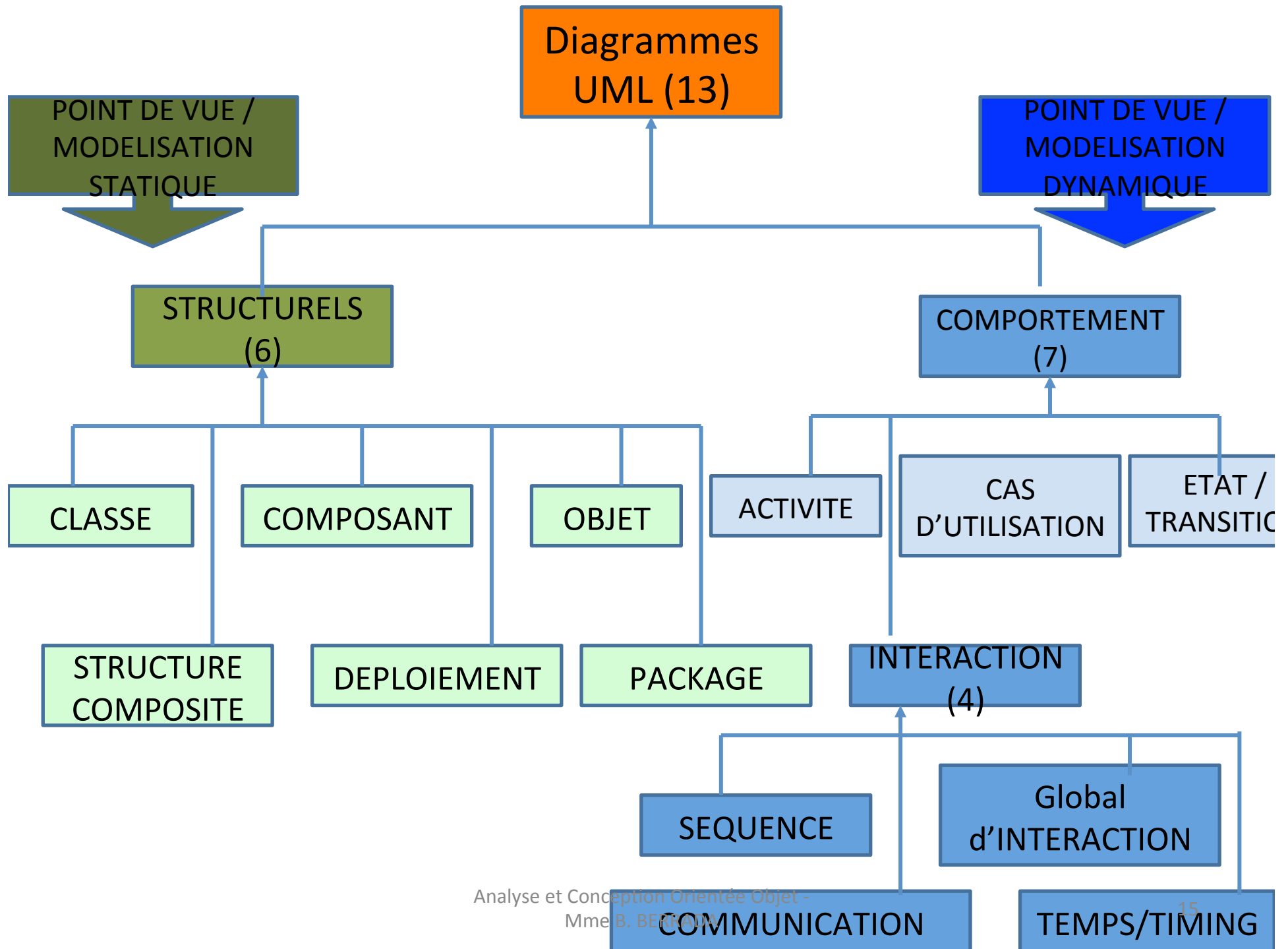
Suite - Description des Diagrammes UML 2

Diagramme	Finalité	Historique
Axe Dynamique (4 Diagrammes Comportementaux)		
Etat-Transition (DET) (Machine d'état)	Description des différents états des objets en réaction aux évènements.	UML 1, 2
Communication (DCO)	Représentation des scénarios de communications mettant l'accent sur les objets et les messages échangés.	UML 1, 2
Global d'Interaction (DGI)	Vue générales des interactions, décrites dans les DSE et flots de contrôles du DAC	UML 2
Temps ou Timing (DTP)	Représentation des états et les interactions d'objets dans un contexte où le temps a une forte influence sur le système à gérer.	UML 2

Suite - Description des Diagrammes UML 2

4 ensembles suivant leur finalité *





Structure du Cours

Construction privilégiant :

- ❖ **Trois points de vues classiques, en insistant sur le ou les diagrammes prépondérant.s**
 - **1. AXE FONCTIONNEL**
 - **DIAGRAMME DES CAS D'UTILISATION (DCU)**
 - **DIAGRAMME DE SEQUENCE (DES)**
 - **DIAGRAMME D'INTERACTION (DAC)**
 - **2. AXE STATIQUE**
 - **3. AXE DYNAMIQUE**
- ❖ **Une présentation détaillée d'UML 2, illustrant les concepts essentiels et les diagrammes les plus structurants par des exemples et études de cas.**

Modélisation Fonctionnelle

- Elle comprend 3 diagrammes dans la modélisation dynamique :

- **DIAGRAMME DES CAS D'UTILISATION (DCU)**

- DIAGRAMME DE SEQUENCE (DSE)

- DIAGRAMME D'INTERACTION (DAC)

DIAGRAMME DE CAS D'UTILISATION ou USE CASES :

Notions de base

- **Présentation et notions de base :**
 - Les diagrammes de cas d'utilisation ont été définis initialement par Ivar JACOBSON en 1992 dans sa méthode OOSE (Addison-Wesley, 92)
 - Ils constituent un moyen de recueillir et de décrire les besoins des acteurs du système
 - Ils décrivent l'interaction entre les acteurs (utilisateurs du cas) et le système, **selon le point de vue utilisateur**
 - Ils mettent en jeu 3 concepts importants :
 - ✓ **Acteur,**
 - ✓ **Cas d'utilisation ,**
 - ✓ **Interaction entre acteur et cas d'utilisation**

Etape 1.1 :

Acteur et Définition

■ ACTEUR : Définition

- **Un acteur représente un rôle joué par une entité externe :**
 - ✓ utilisateur humain (personne physique),
 - ✓ dispositif matériel ou autre système qui interagit avec le système étudié.

- **Une même personne physique peut se comporter en autant d'acteurs différents que le nombre de rôles qu'elle joue vis-a-vis du système**
 - ✓ **Exemple 1** : Opérateur de maintenance pour un GAB et Client de banque
 - ✓ **Exemple 2** : Dans un système de messagerie, l'administrateur peut être aussi l'utilisateur de cette messagerie.

- **Plusieurs personnes peuvent également jouer le même rôle, et donc agir comme un même acteur**
 - ✓ Exemple : (Tous les) Clients

Etape 1.1 : *Acteur - Formalisme*

▪ ACTEUR : Formalisme et illustration

La représentation graphique standard de l'acteur en UML peut être sous 3 formes :

1.



l'icône appelée stick man, avec le nom de l'acteur sous le dessin.
(Pour des acteurs humains)

Nom de l'acteur : Utilisateur, Administrateur

2.



Forme rectangulaire d'une classe stéréotypée, avec le mot clé « acteur »

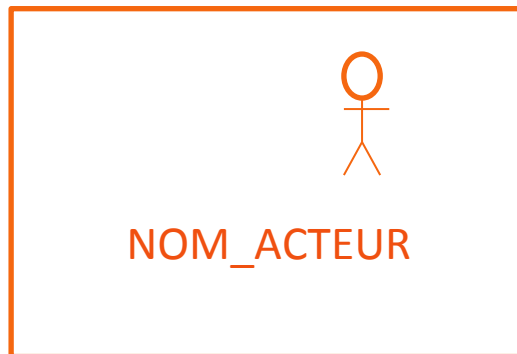
Stéréotype : Variation d'un élément de modèle existant

Etape 1.1 : *Acteur - Formalisme*

■ **ACTEUR : Formalisme et illustration**

La représentation graphique standard de l'acteur en UML peut être sous 3 formes :

3.



(2. et 3. Pour des acteurs non humains)

Etape 1.1 : *Acteur et Identification*

- **ACTEUR : Comment les identifier ?**

- **Les acteurs candidats sont systématiquement :**

- ✓ **Les utilisateurs humains directs** (tous les profils, tels que les administrateurs, opérateur de maintenance, etc.)

*Ils sont qualifiés **d'acteurs principaux**, pour qui le cas d'utilisation produit un résultat observable.*

Conventionnellement, Ils sont représentés à gauche du système étudié.

- ✓ **Les autres systèmes connexes** qui interagissent aussi avec le système étudié, par le biais de protocoles bidirectionnels (système informatique, système d'autorisation, etc).

Conventionnellement, Ils sont représentés à droite du système étudié.

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

Exercice 1 - Etape 1.2

Diagramme de Contexte

1.2 Comment représenter les acteurs (seuls) en interaction avec le système ?

- **Sous forme de boîte noire ou diagramme de contexte statique délimitant le domaine d'étude** en précisant :
 - ✓ Ce qui est à la charge du système ;
 - ✓ l'environnement extérieur au système étudié avec lequel ce dernier communique.

- **Il est notamment recommandé pour la définition des acteurs,** avant de commencer à s'intéresser à d'autres aspects, tels que les cas d'utilisation et les packages.

Exercice 1 - Etape 1.2

Diagramme de Contexte

➤ Ses composants sont :

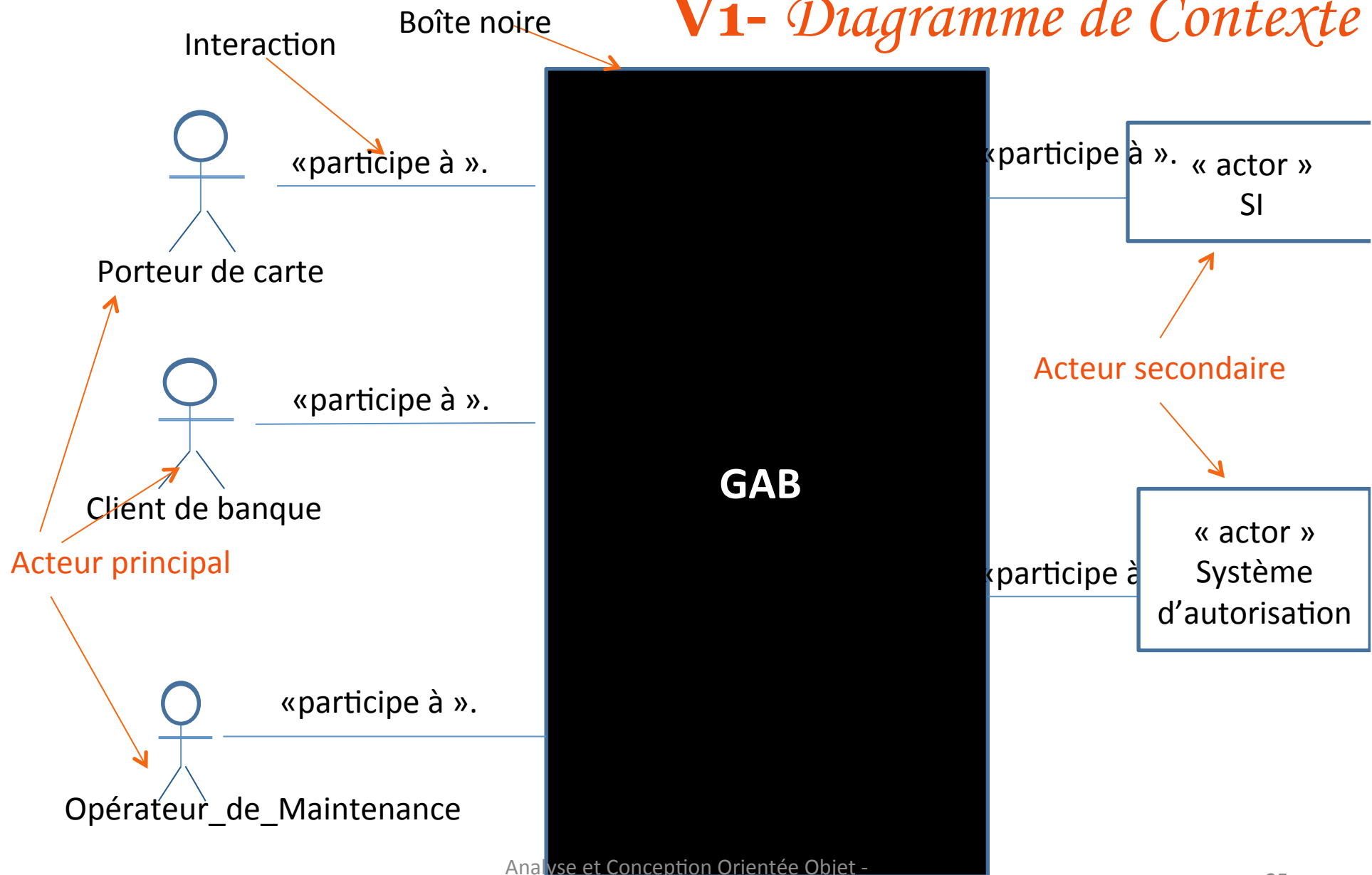
- ✓ Les acteurs externes.
- ✓ Un processus unique (Boîte noire*) symbolisant le Système Information étudié(SI) ;
- ✓ L'Echange entre le système étudié et son environnement.
Autour du système, nous devons donc relier les acteurs principaux et secondaires correspondants.

Remarque : On représente les interactions des acteurs avec le système étudié. Mais pas les interactions entre acteurs.

* : représentation d'un système sans considérer son fonctionnement interne

Exercice 1 - Etape 1.2

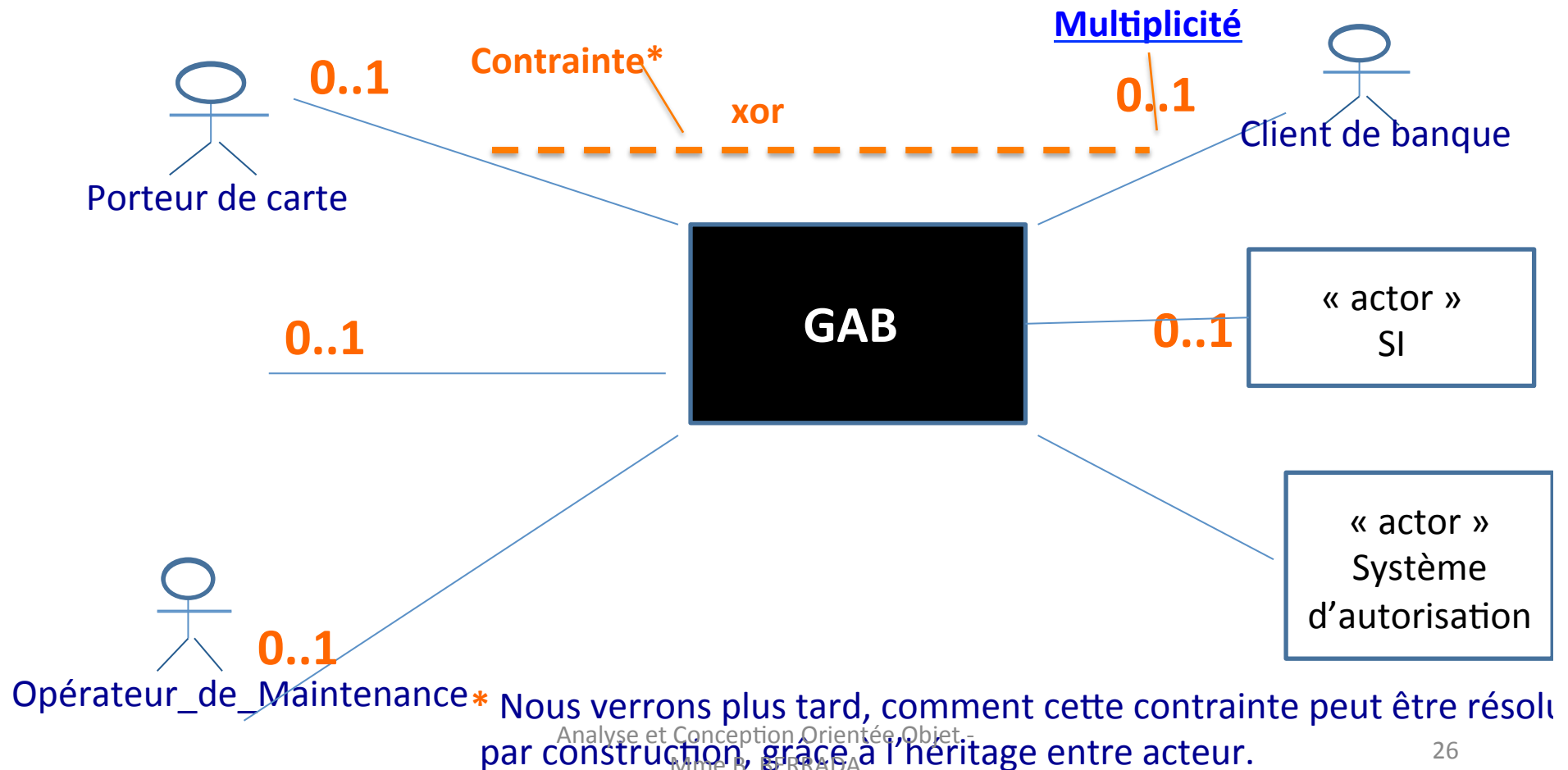
V1- Diagramme de Contexte



Exercice 1 - Etape 1.2

V2 - Diagramme de Contexte – Cardinalités et Contrainte

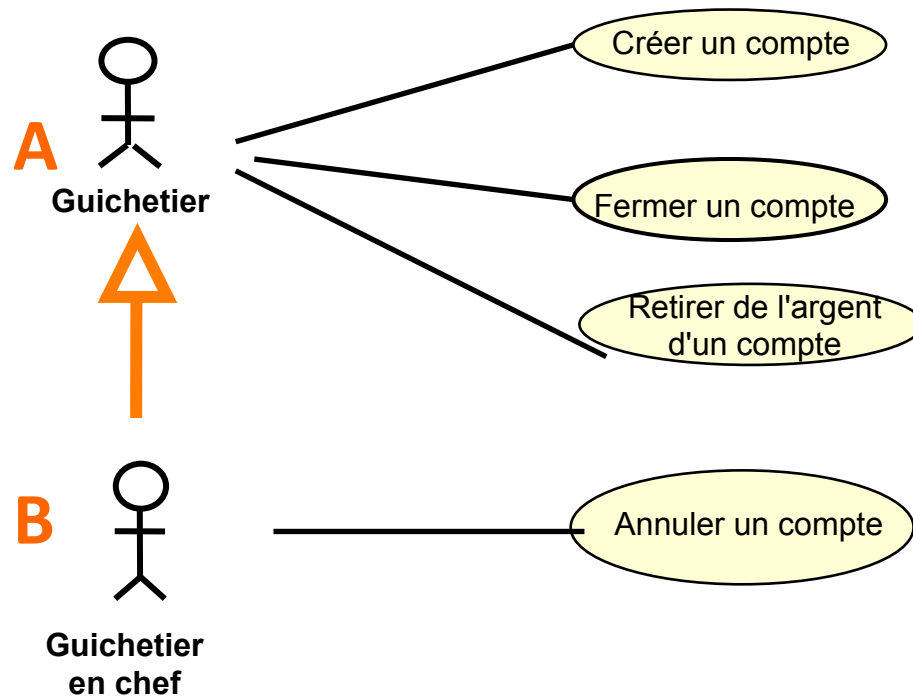
1. Le GAB est un système fondamentalement mono-utilisateur : à tout instant, il n'y a qu'une instance de chaque acteur (au maximum) connectée au système.
2. Les acteurs Porteur de carte et Client de la banque sont mutuellement exclusifs



Relations entre acteurs

Relation acteur <-> acteur

- Une seule relation la généralisation/spécialisation

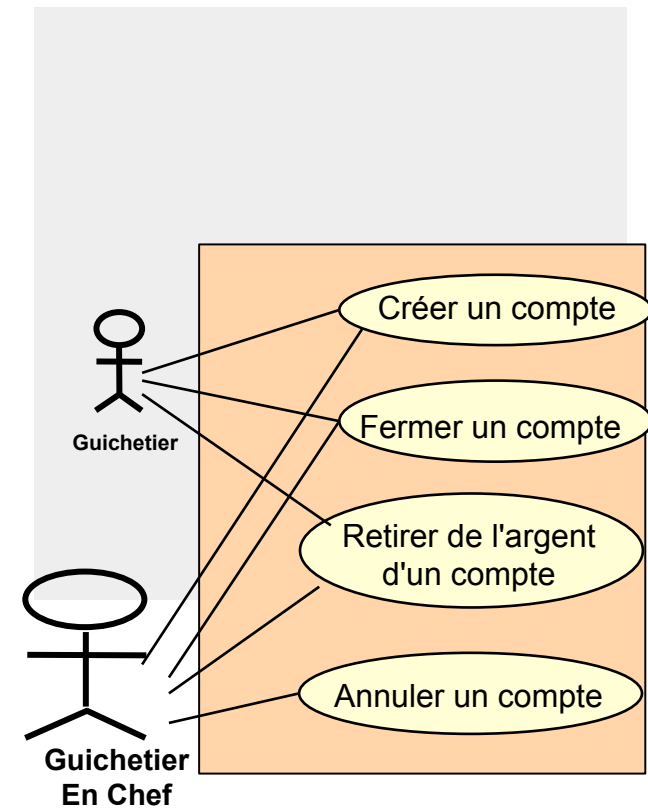
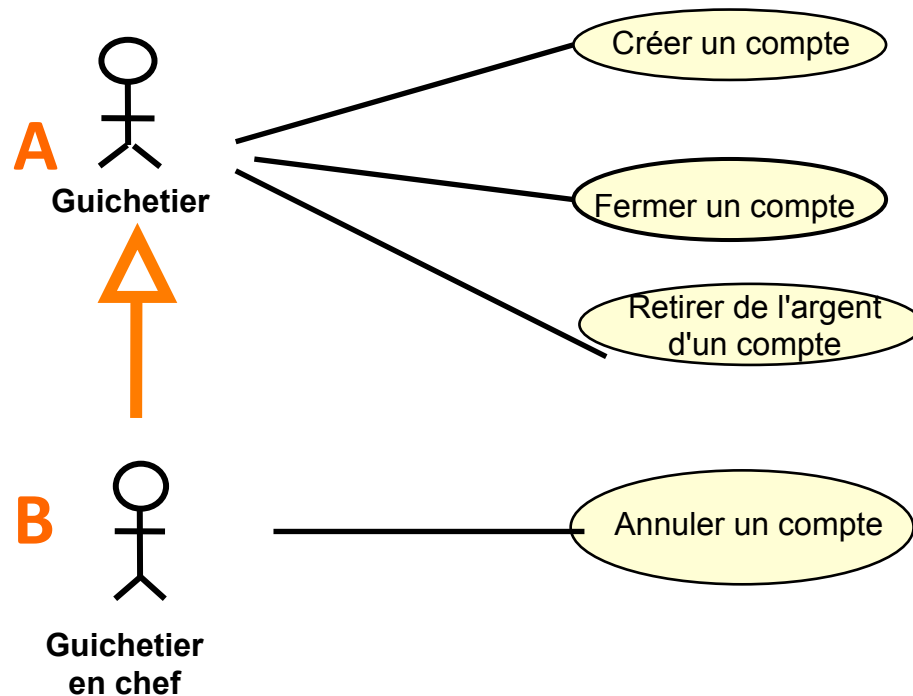


Les acteurs spécialisés héritent alors des associations de l'acteur ancêtre

Relations entre acteurs

Relation acteur <-> acteur

- Une seule relation la généralisation/spécialisation

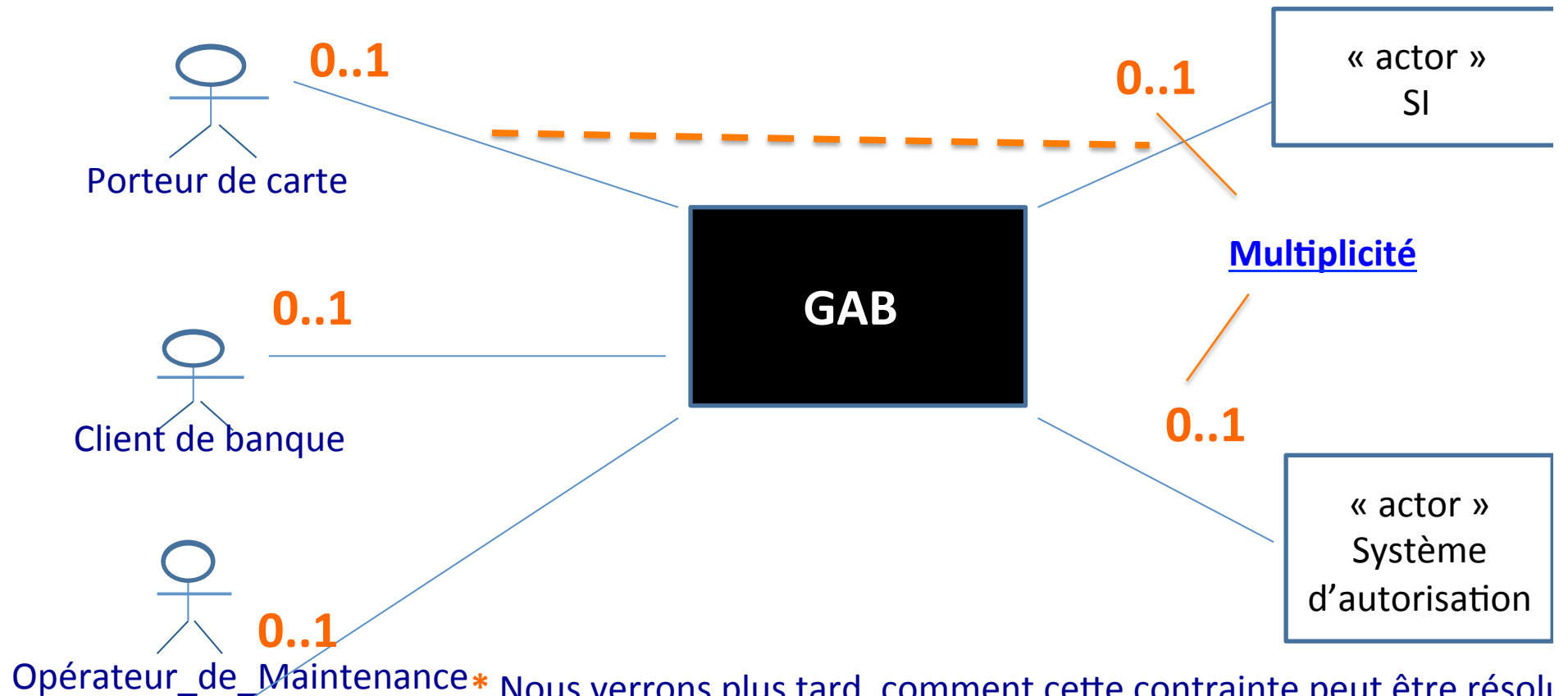


Les acteurs spécialisés héritent alors des associations de l'acteur ancêtre

Exercice 1 - Etape 1.2

V2 - Diagramme de Contexte – Cardinalités et Contrainte

1. Le GAB est un système fondamentalement mono-utilisateur : à tout instant, il n'y a qu'une instance de chaque acteur (au maximum) connectée au système.
2. Les acteurs Porteur de carte et Client de la banque sont mutuellement exclusifs

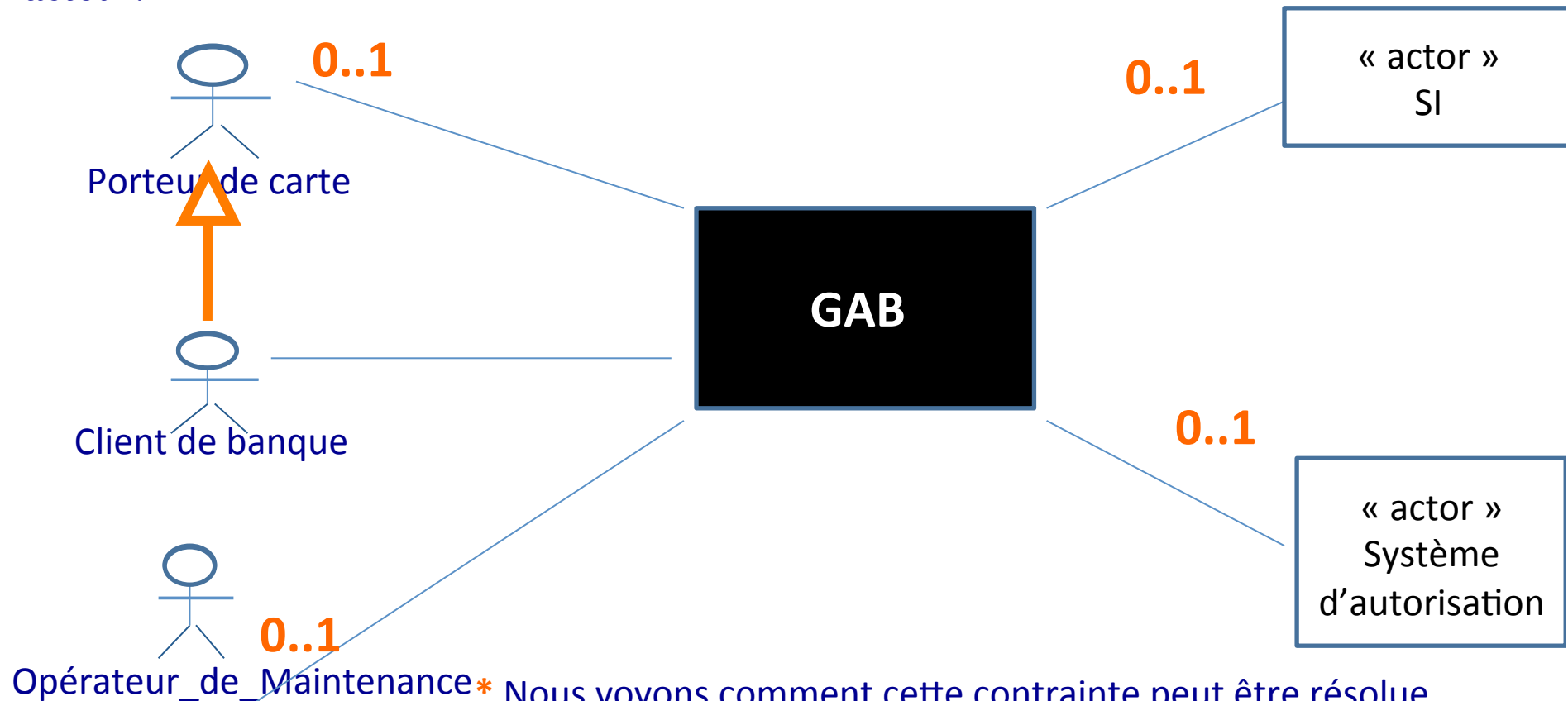


* Nous verrons plus tard, comment cette contrainte peut être résolue par construction, grâce à l'héritage entre acteur.

Exercice 1 - Etape 1.2

V2 - Diagramme de Contexte – Cardinalités et Contrainte

2. Une **autre** solution plus élaborée, consiste à considérer que *Client Banque* est une spécialisation de *Porteur de carte*, comme cela est illustré dans la figure suivante. Le problème précité d'exclusivité est ainsi résolu par construction, grâce à l'héritage entre acteur.



* Nous voyons comment cette contrainte peut être résolue par construction, grâce à l'héritage entre acteur.

CAS D'UTILISATION :

Identification

Cas d'Utilisation : Comment les identifier ?

L'objectif est le suivant :

- L'ensemble des cas d'utilisation doivent décrire exhaustivement les exigences **fonctionnelles** du système.
- Chaque cas d'utilisation correspond donc à une fonction métier du système, selon un point de vue d'un de ses acteurs.
- **Pour chaque acteur, il convient de :**
 - ✓ Rechercher les différentes fonctions métier utiles à l'utilisation du système (tâches utilisateur)
 - ✓ Déterminer dans le cahier des charges, les services fonctionnels attendus du système

CAS D'UTILISATION :

Définition et Notation

Cas d'Utilisation : Définition

- Un Cas d'Utilisation (use case) correspond à un **nombre d'actions** ou fonctionnalités (scenario) que le système devra exécuter en réponse à un **besoin d'un acteur**.
- Un cas d'utilisation doit produire un résultat observable et intéressant pour un ou plusieurs acteurs particuliers.



Le cas d'utilisation est représenté par un ovale dans lequel figure son nom.

- Un cas d'utilisation doit être relié à au moins un acteur.
- Un cas d'utilisation est Nommé par un verbe à l'infinitif, suivi d'un complément, du point de vue de l'acteur, par exemple : Envoi d'un message

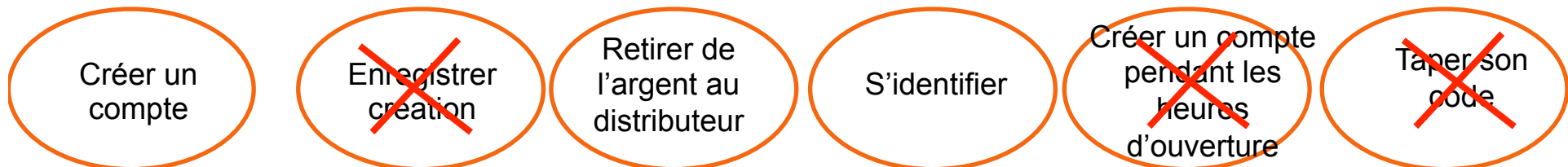
CAS D'UTILISATION :

Notation

- **Cas d'Utilisation : Notation**



- **Exemples**



CAS D'UTILISATION :

Intérêt

- Les cas d'utilisation (CU) représentent le dialogue **entre l'acteur et le système** de manière abstraite
- Un CU correspond à une fonction du système **visible par l'acteur**
- Un CU est une **séquence d'actions** réalisées par le système produisant un résultat observable
- Ensemble de scénarios au sein d'une description unique
Regroupe un ensemble de scénarios correspondant à un même but

DIAGRAMME DE CAS D'UTILISATION :

Intérêt (1)

- **Intérêt :**

- Déterminer les besoins fonctionnels du système
- Permettre aux utilisateurs d'exprimer leurs attentes et besoins
- Permettre d'impliquer les utilisateurs dès les premiers stades de développement
- Constituer une base pour les tests fonctionnels

CAS D'UTILISATION

Relations entre éléments de DCU

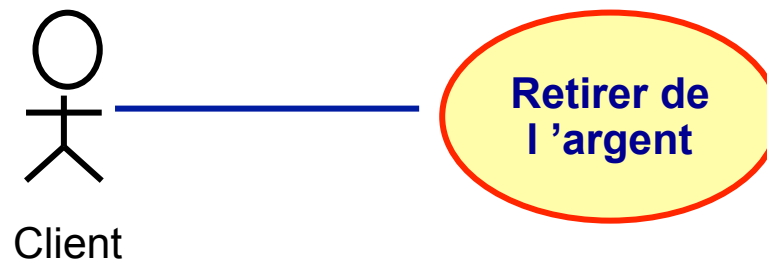
- **3 types de relations :**
 - Communication
 - Dépendances stéréotypées (inclusion et extension)
 - Généralisation/Spécialisation

CAS D'UTILISATION

Relations entre éléments de DCU

Relation acteur <-> cas d'utilisation : communication

- Point de vue **besoin** : représente la possibilité d'atteindre un but
- Point de vue **système** : représente un échange de messages



CAS D'UTILISATION :

Relation d'inclusion

Relations cas d'utilisation <-> cas d'utilisation

- **inclusion** (« *include* »)

- Le cas d'utilisation incorpore **explicitement** et de manière **obligatoire** un autre cas d'utilisation à l'endroit spécifié



Signifiant qu'une instance de A contient le comportement décrit par B ou encore
B est une partie *obligatoire* de A

- Cette relation permet ainsi de décomposer des comportements et de définir des comportements partageables entre plusieurs cas d'utilisation.

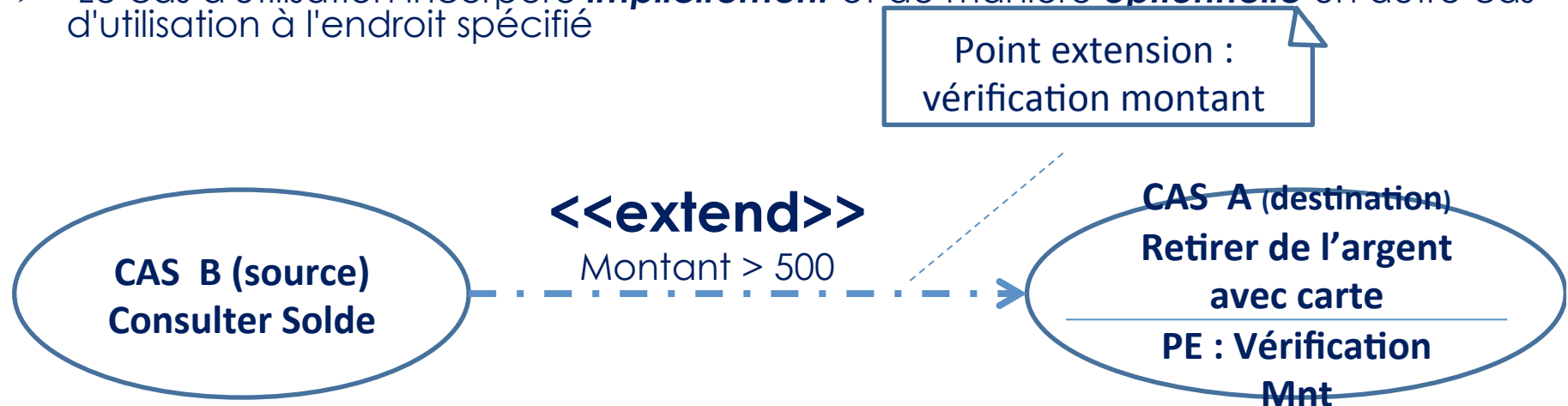
CAS D'UTILISATION :

Relation d'extension

Relations cas d'utilisation <-> cas d'utilisation

■ extension (« extend »)

- Le cas d'utilisation incorpore **implicitement** et de manière **optionnelle** un autre cas d'utilisation à l'endroit spécifié



Signifiant qu'une instance de A peut être étendue par le comportement décrit par B

- Si une condition d'extension existe, il est possible de la préciser à proximité du mot clé.

Consultation de solde n'intervient que si le retrait dépasse 500 dirhams

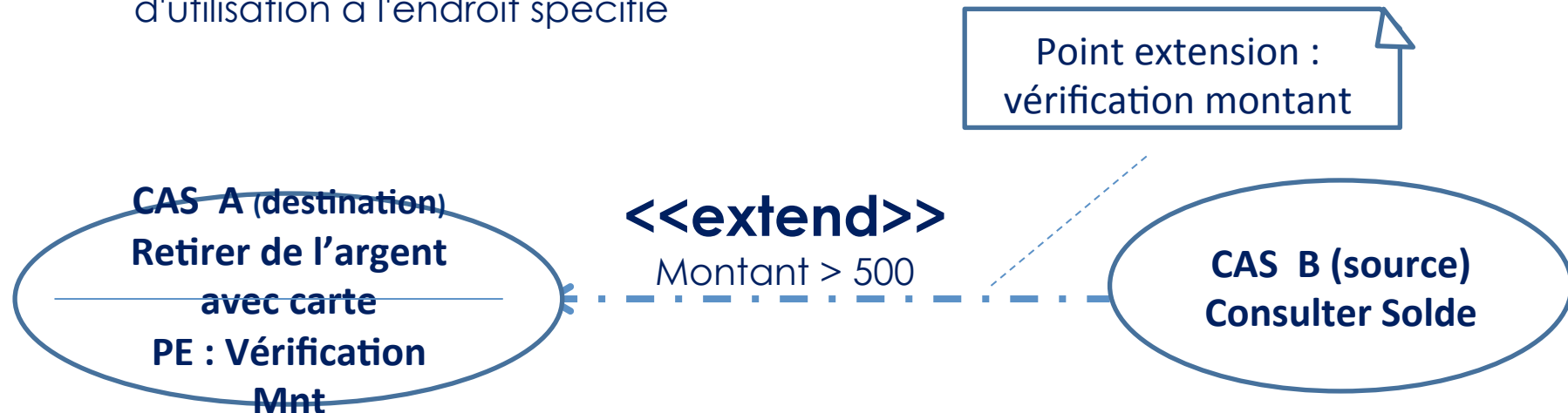
CAS D'UTILISATION :

Relation d'extension

Relations cas d'utilisation <-> cas d'utilisation

- **extension** (« extend »)

- Le cas d'utilisation incorpore **implicitement** et de manière **optionnelle** un autre cas d'utilisation à l'endroit spécifié



Signifiant qu'une instance de A peut être étendue par le comportement décrit par B

- Si une condition d'extension existe, il est possible de la préciser à proximité du mot clé.

Consultation de solde n'intervient que si le retrait dépasse 500 dirhams

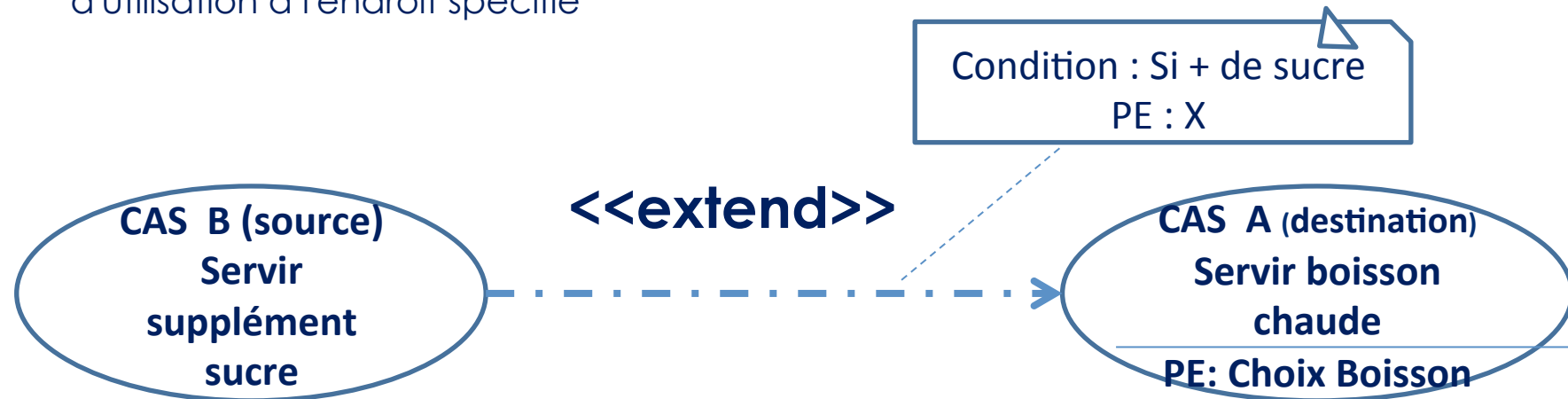
CAS D'UTILISATION :

Relation d'extension

Relations cas d'utilisation <-> cas d'utilisation

▪ Relations d'extension (« extend »)

- Le cas d'utilisation incorpore **implicitement** et de manière **optionnelle** un autre cas d'utilisation à l'endroit spécifié



Signifiant qu'une instance de A peut être étendue par le comportement décrit par B

- Si une condition d'extension existe, il est possible de la préciser à proximité du mot clé.

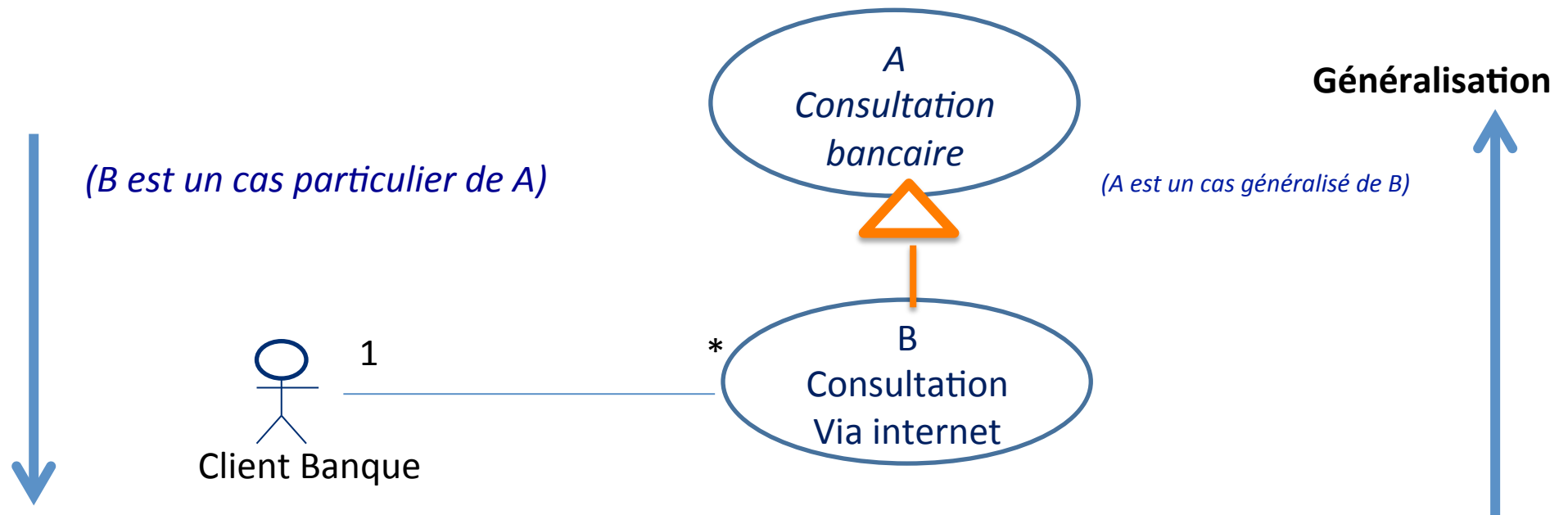
CAS D'UTILISATION :

Relation généralisation /spécialisation

Relations cas d'utilisation <-> cas d'utilisation

■ Relations de généralisation / spécialisation

- la généralisation est la relation entre un cas d'utilisation et un ou pls autres cas d'utilisation partageant des caractéristiques communes



Le CU parent généralisé, *abstrait*, ne s'instancie plus directement, mais seulement par le biais du cas spécialisé.

Spécialisation

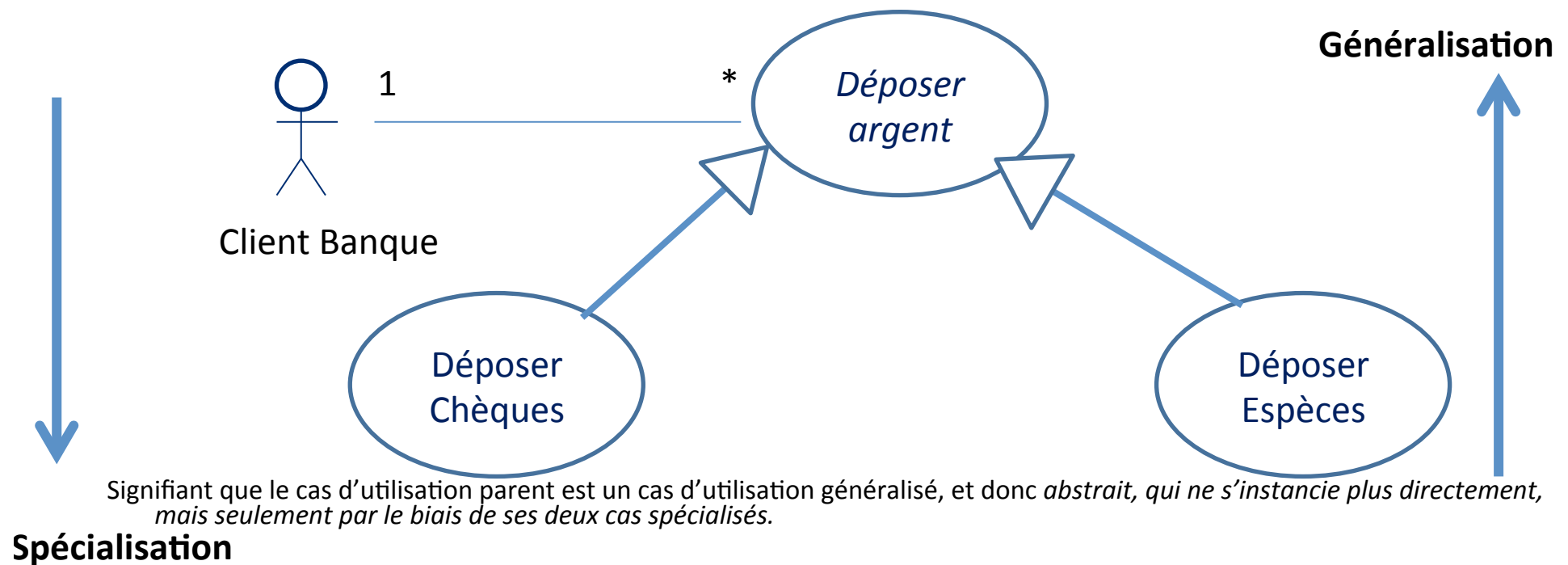
CAS D'UTILISATION :

Relation généralisation / spécialisation

Relations cas d'utilisation <-> cas d'utilisation

■ Relations de généralisation / spécialisation

- Conformément aux principes de spécialisation-généralisation présentée pour les classes, la généralisation est la relation entre un cas d'utilisation et de deux autres cas d'utilisation partageant des caractéristiques communes



Description textuelle des cas d'utilisation

- **La fiche de description textuelle d'un cas d'utilisation**

➤ La structuration suivante est souvent préconisée :

Sommaire d'identification (obligatoire)	Titre, Résumé, Dates de création et de modification, Version, Responsables, Acteurs, ...
Description des scénarios	le scenario nominal, les scenarios (enchaînements) alternatifs, les scénarios (enchaînements) d'erreur, mais aussi les pré-conditions et les post-conditions.
Exigences non-fonctionnelles (optionnel)	Fréquence, volumétrie, disponibilité, fiabilité, intégrité, confidentialité, performances, concurrences, etc. Précise aussi les contraintes IHM, comme les règles d'ergonomie, une charte graphique, etc.

Description textuelle des cas d'utilisation :

2. Description des scénario

➤ **SCENARIO ? Il décrit le déroulement d'un CU**

**Le scénario est au CU (cas d'utilisation)
ce qu'est l'objet à la classe**

- ✓ Un cas d'utilisation propose un comportement nominal (scénario nominal ou idéal)
- ✓ Un cas d'utilisation propose aussi un ou pls cas alternatifs représentant un cheminement particulier dans le cas d'utilisation (que se passe t-il **si** ..?)
- ✓ Un cas d'utilisation propose aussi des situations exceptionnelles

L'idéal est de créer suffisamment de scénarios pour couvrir l'essentiel du cas d'utilisation

Description textuelle des cas d'utilisation :

2. Description des scénario

- **Pré-Condition** : Si certaines conditions particulières sont requises avant l'exécution du cas
- **Scénario Nominal** : Il s'agit du scénario principal qui doit se dérouler sans incident et qui permet d'aboutir au résultat souhaité
- **Scénarios Alternatifs** : Les autres scénarios, secondaires ou correspondant à la résolution d'anomalies. *Le lien avec le scénario principal se fait par une numérotation hiérarchisée (1.1a, 1.1b,...) pour rappeler le numéro de l'action concernée*
- **Scénario d'erreur** : Il termine brutalement le cas 'utilisation en échec
- **Post-Condition** : Par symétrie, si certaines conditions doivent être réunies après l'exécution du cas.

Description textuelle des cas d'utilisation

La description textuelle des cas d'utilisation est indispensable car elle seule permet de :

- Communiquer facilement avec les utilisateurs
- S'entendre sur la terminologie métier employée.

En revanche, le texte présente des désavantages car difficile de montrer :

- La succession des enchaînements
- Le moment où les acteurs secondaires sont sollicités.

De même que la maintenance des évolutions est aussi fastidieuse.

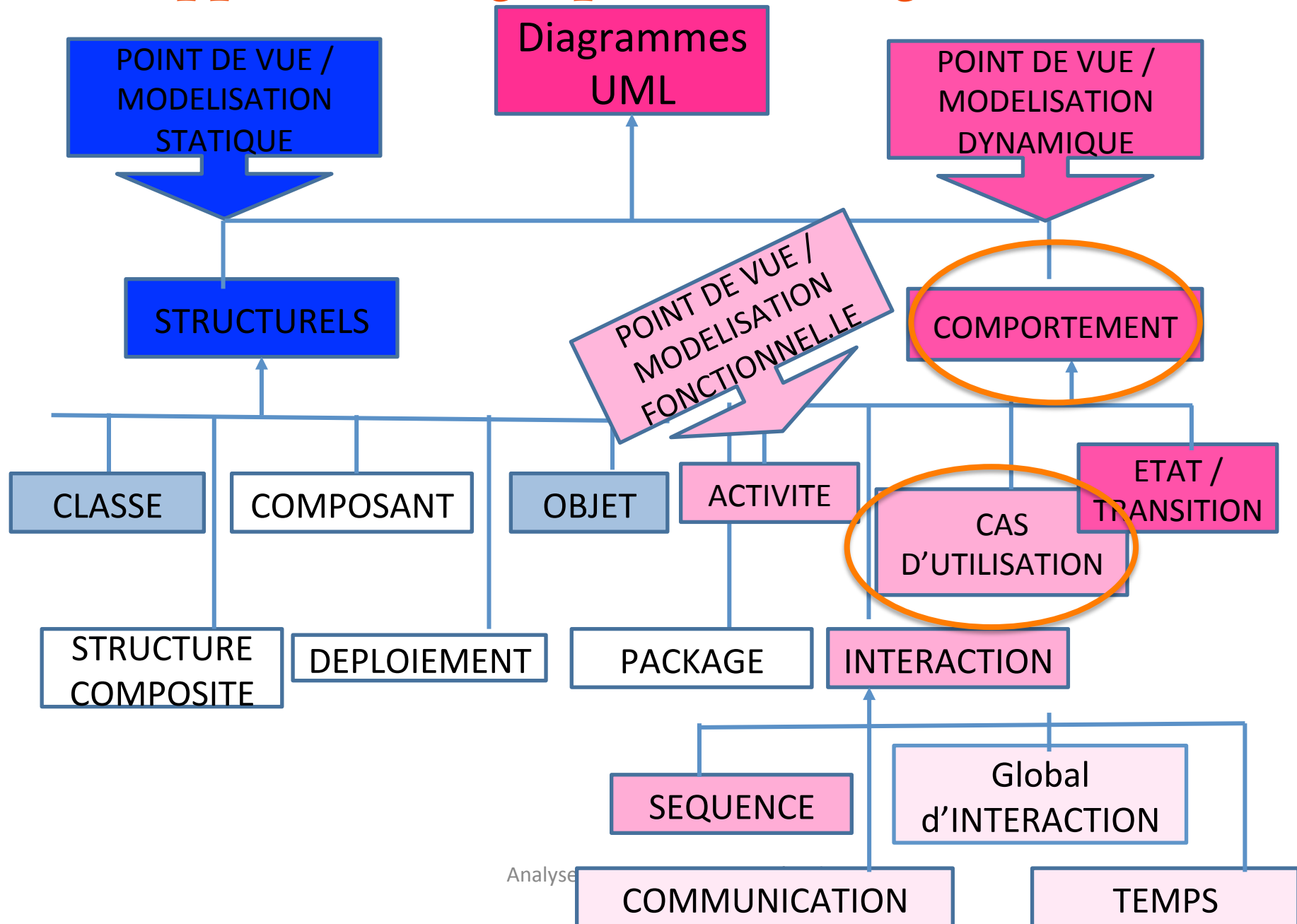
Description dynamique des cas d'utilisation

Pour les cas d'utilisation, on peut utiliser une description dynamique :

1. Diagramme de séquence

2. Diagramme d'activité

Rappel - Cartographie des Diagrammes UML 2



*Dans la cartographie des diagrammes UML 2,
Vous avez étudié ...*

Dans une première partie :

- ♦ **Les diagrammes de cas d'utilisation** modélisant à **QUOI** sert le système, en organisant les interactions possibles avec les acteurs ;

Dans une seconde partie, vous étudierez quelques-uns des diagrammes d'interaction permettant de :

- ♦ Modéliser **comment** les objets communiquent entre eux (point focal : échange de messages) ;
- ♦ Offrir une vue plus holistique du comportement d'un jeu d'objets.

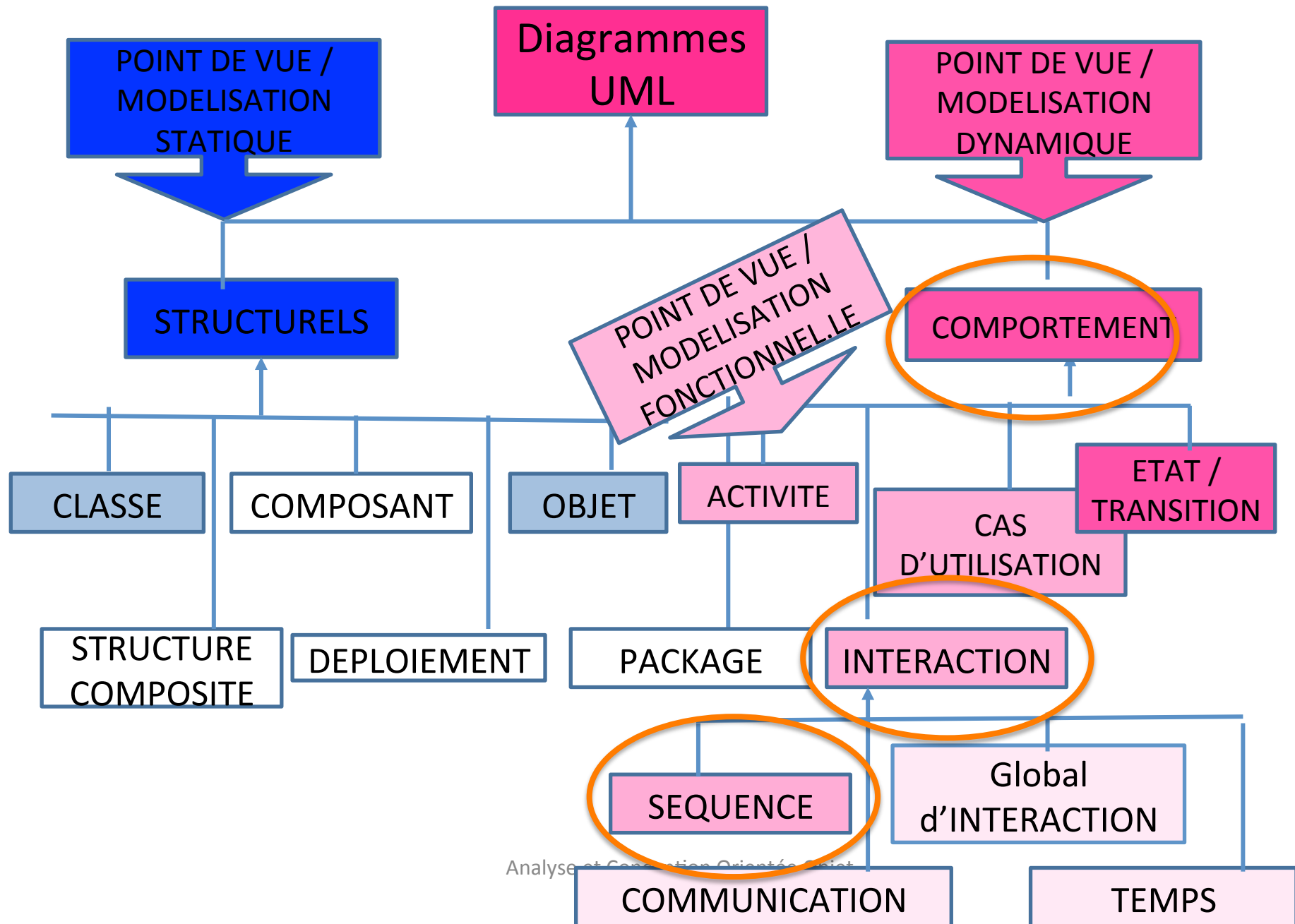
Diagrammes d'Interactions

La norme 2.0 reclasse les diagrammes de la norme 1.X et met les 4 diagrammes suivants dans la catégorie des diagrammes d'interaction :

- ◆ **Diagrammes de séquence**
- ◆ **Diagramme de communication**
- ◆ Diagramme d'interaction globale
- ◆ Diagrammes de temps

Ces types de diagramme tendent à mettre l'accent spécifiquement sur la **séquence** des opérations plutôt que sur les données qui sont véhiculées, bien que certains ajouts (décorations) permettent de le faire.

Rappel - Cartographie des Diagrammes UML 2



Description dynamique

Diagramme de Séquence (DSE) :

Le DSE est une forme de diagramme comportemental, dérivé des scénarios de OMT

L'**objectif** du DSE est de représenter les interactions entre objets en indiquant la **chronologie des échanges**.

Grâce à ces informations, vous pouvez déterminer plus précisément pourquoi deux objets sont liés et voir comment les ils s'utilisent mutuellement.

Cette représentation est réalisée par cas d'utilisation en considérant :

- Forme générique : décrivant **tous** les déroulements possibles d'un scénario et contenant des **branchements**, des **conditions**, et des **boucles**;
- Forme d'instanciation : décrivant **un aspect spécifique** d'un scénario et ne contenant aucun **branchements**, et aucune **condition** ou **boucle**.

En général, un DSE capture le comportement d'un seul scénario.

Description dynamique

Autrement dit :

- Les diagrammes de séquences permettent de décrire **COMMENT** les éléments du système interagissent entre eux et avec les acteurs :
 - ◆ Les objets au coeur d'un système interagissent en s'échangeant des messages ;
 - ◆ Les acteurs interagissent avec le système au moyen d'IHM (Interfaces Homme-Machine).

Description dynamique

Diagramme de Séquence (DSE):

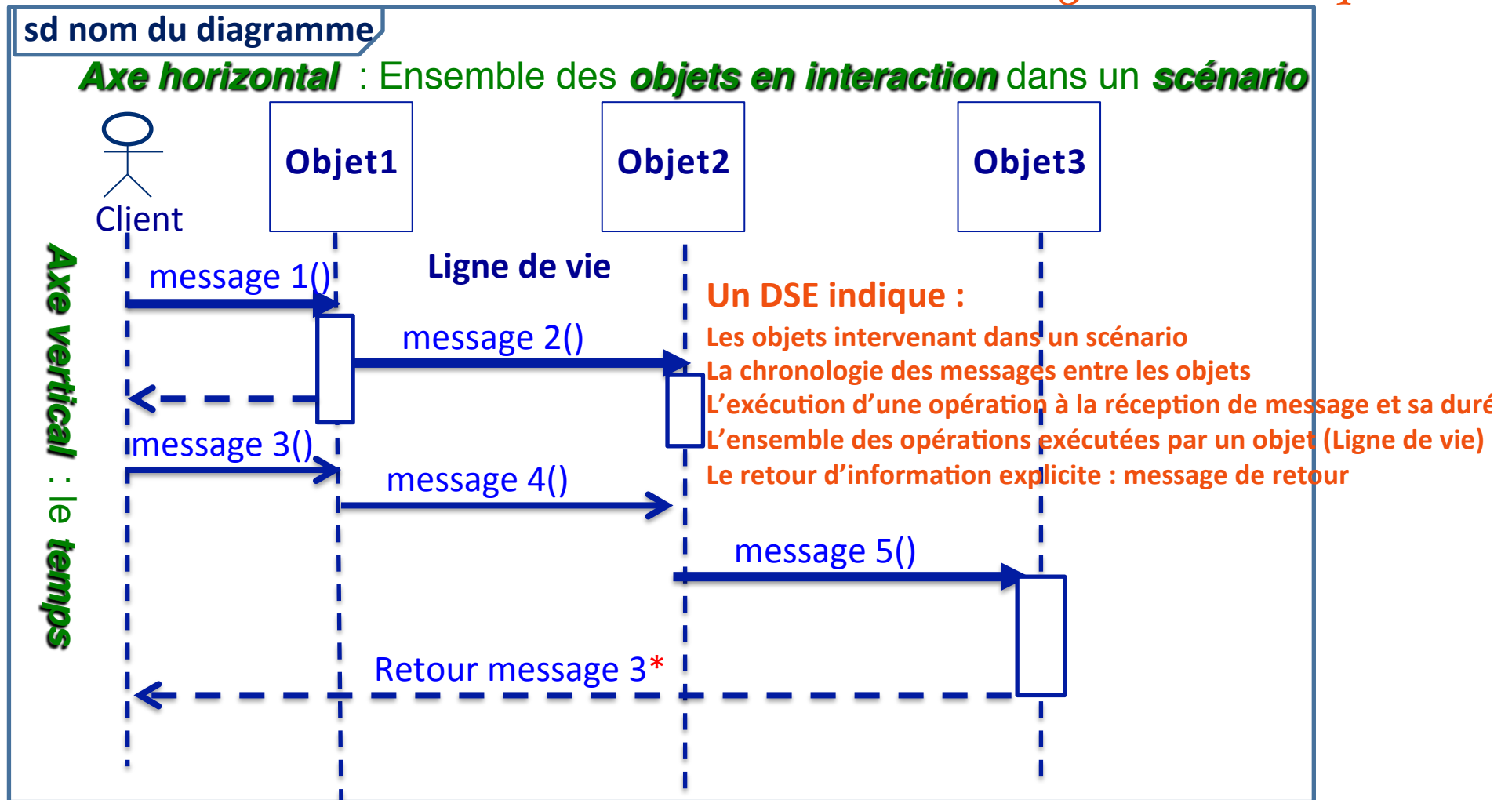
Formalisme général du cadre d'un diagramme de séquence :



sd : abréviation de « sequence diagram »

Description dynamique

Formalisme du diagramme de séquence



* Les messages asynchrones pouvant être reçus dans un ordre différent de l'ordre d'envoi

Description dynamique *Diagramme de Séquence (DSE):*

Pour illustrer les concepts de base et les opérations d'un DSE, prenons l'exemple d'un scenario de traitement d'une commande, spécifiant :

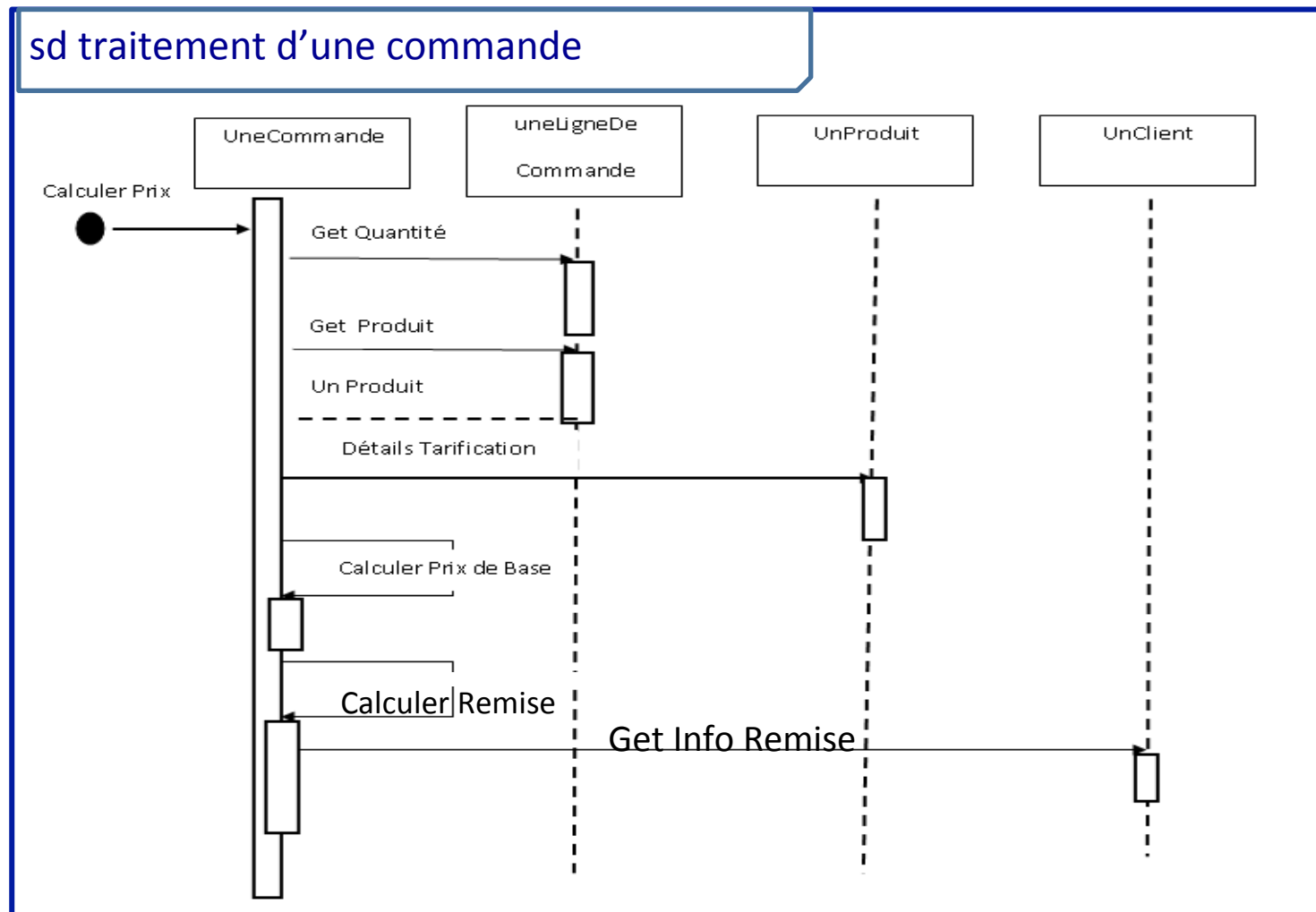
Pour calculer le prix d'une commande, l'objet commande doit :

- ✓ Lire toutes les lignes de la commande
- ✓ Déterminer leur prix, en fonction des tarifications des produits.

Après avoir traité chaque ligne, il doit ensuite :

- ✓ Calculer une remise globale, qui dépend des règles associées à chaque client.

Implémentation du scénario exemple par un DSE

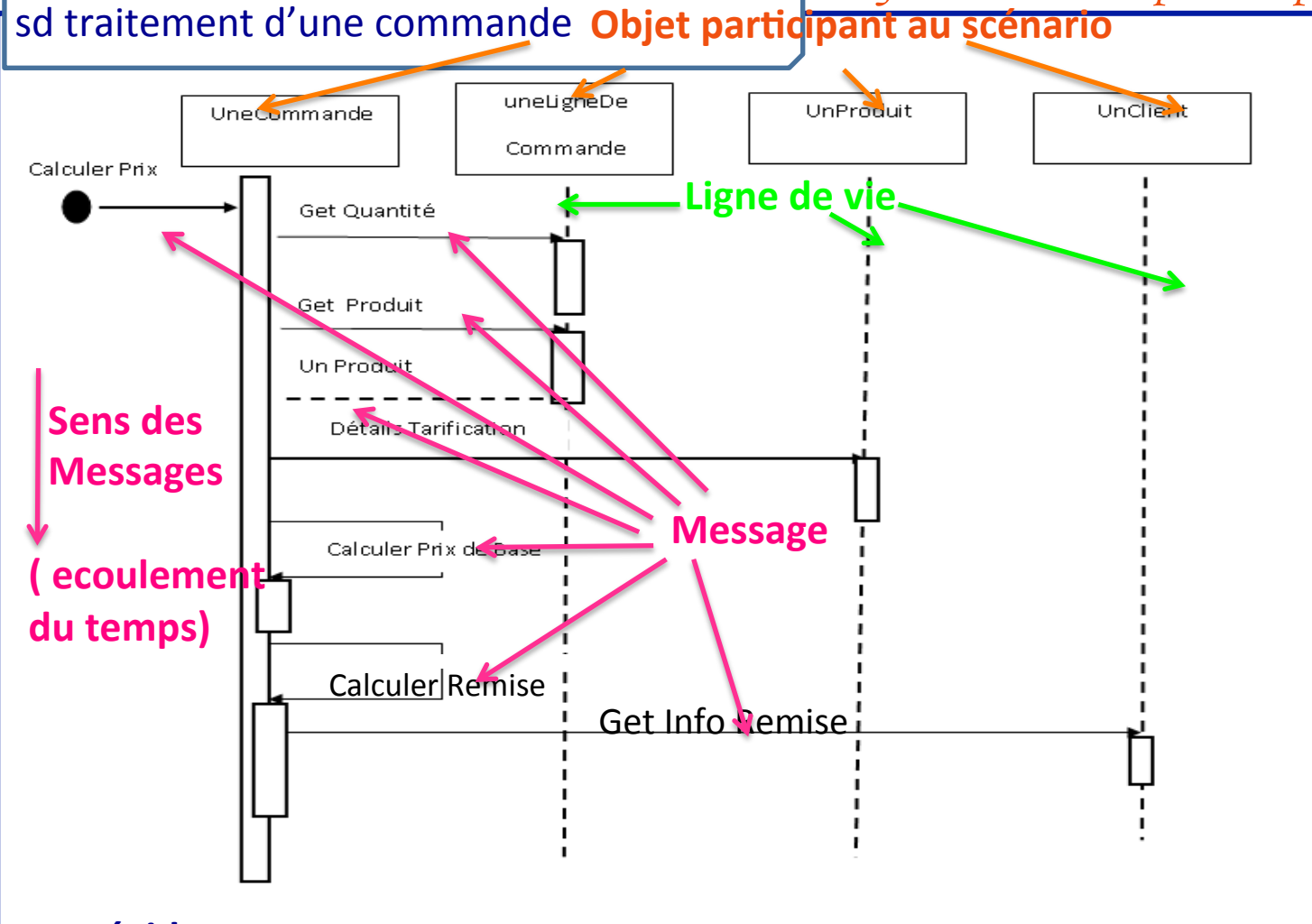


Le DES met bien en évidence :

- L'interaction d'un objet participant au scénario
- Chaque participant ayant une ligne de vie, qui parcourt verticalement la page
- L'ordre des messages se faisant de haut en bas.

Implémentation du scénario exemple par un DSE

Identification des composants primaires

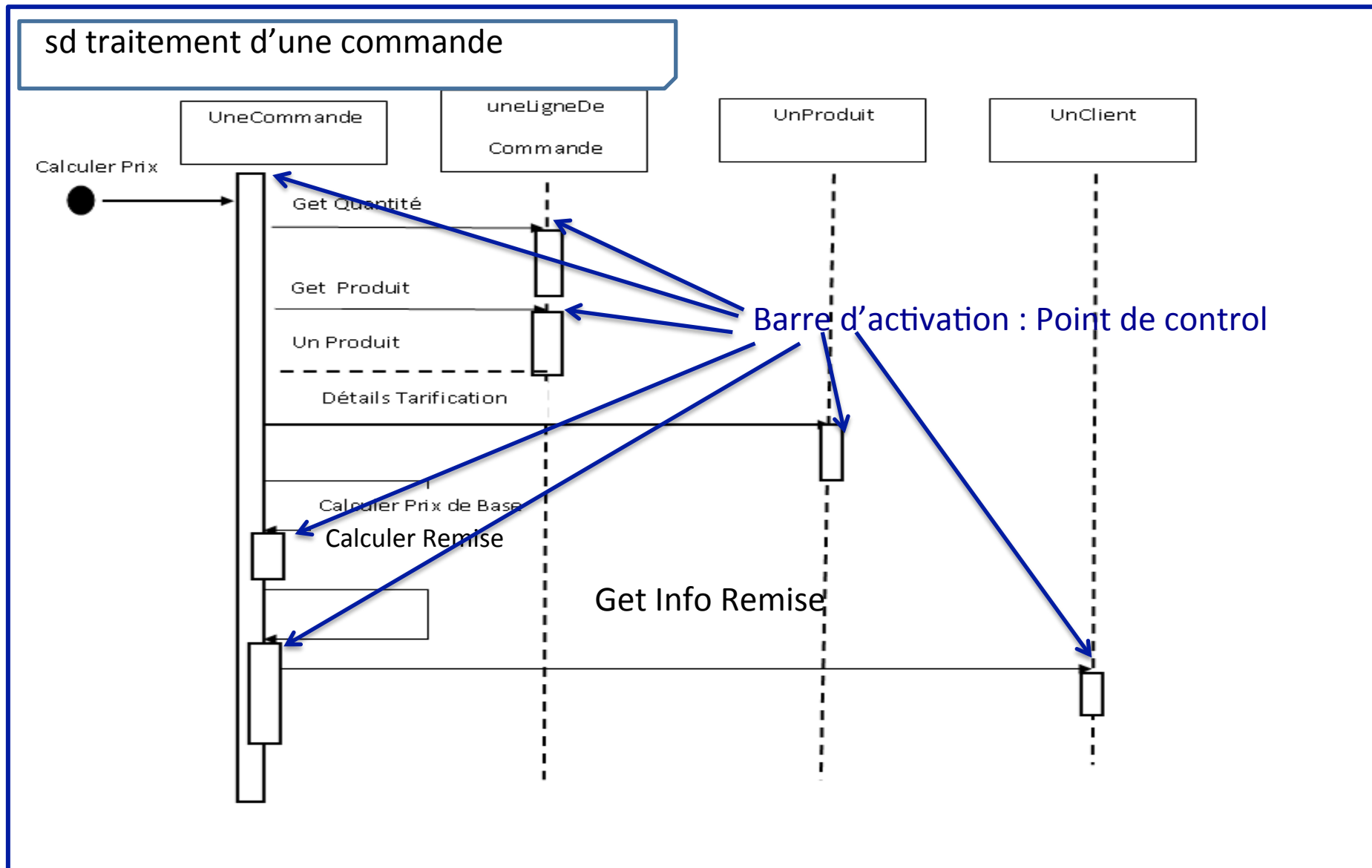


Le DSE met bien en évidence :

- L'interaction d'un objet participant au scénario
- Chaque participant ayant une ligne de vie, qui parcourt verticalement la page
- L'ordre des messages se faisant de haut en bas.

Implémentation du scenario exemple par un DSE

Composants optionnels



Description dynamique

Sémantique des composants d'un DSE

Définition des composants primaires d'un DSE

➤ **1. Les objets (Object)** : participants à l'interaction

Les objets apparaissent toujours dans la partie supérieure, ce qui facilite l'identification des classes (en supposant que la notation des classes est utilisée).

Représentation graphique :

UML



UML 2



Les composants primaires (suite) :

- **2. La ligne de vie de l'objet (Object lifeline) :** représentant
- ✓ la vie d'un objet dans le contexte de la séquence **d'événements**
 - ✓ La durée de l'activité du participant dans l'interaction.

Formalisme :

|
|
|
|

Un événement se produit généralement lors de la réception explicite d'un signal ou d'un message, lorsqu'une condition devient vraie, écoulement d'une période de temps (expression temporelle)

- ✓ Chaque ligne de vie comporte une activation : point de contrôle (durée de vie de l'activité du participant dans l'interaction)
 - Le point de contrôle correspond au temps pendant lequel l'une des méthodes du participant est au sommet de la pile
 - Les barres d'activation sont facultatives, mais elles clarifient le comportement.

Les composants primaires (suite) :

- **3. Les messages** définissent une communication particulière entre deux instances, présentes sur des lignes de vie.
 - ◆ Représentés par des flèches directionnelles véhiculant une information (message) envoyée entre les objets ;
 - ◆ Les messages anticipent qu'une action sera entreprise ;
 - ◆ Dans la plupart des cas, la réception d'un message est suivie de l'exécution d'une méthode d'une classe (classe réceptrice) ;
 - ◆ L'ordre relatif des messages est matérialisé par l'axe vertical qui représente l'écoulement du temps ;
 - ◆ Le retour de message doit être matérialisé, lorsqu'il existe.

Plusieurs types de messages existent, les plus communs sont :

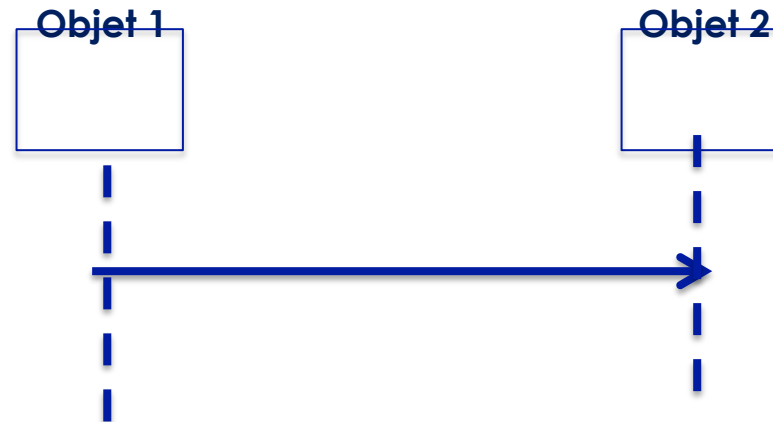
- ◆ L'envoi d'un signal ou d'un événement ;
- ◆ L'invocation d'une opération ;
- ◆ La création ou la destruction d'une instance.

Description dynamique

Sémantique des composants d'un DSE

➤ **3.1 Message asynchrone :**

- ✓ L'émetteur n'attend pas la réponse à son message, il poursuit l'exécution de ses opérations.



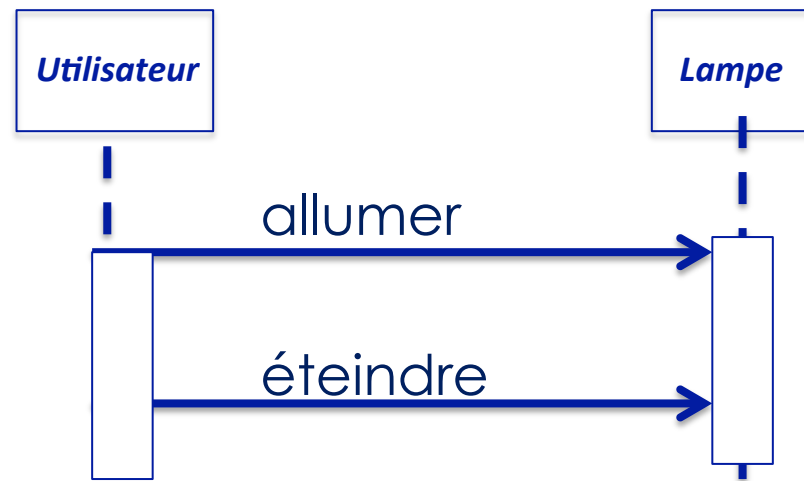
- ◆ Une interruption ou un événement sont de bons exemples de signaux.
- ◆ Ils n'attendent pas de réponse et ne bloquent pas l'émetteur qui ne sait pas si le message arrivera à destination, le cas échéant quand il arrivera et s'il sera traité par le destinataire.
- ◆ Lors de l'envoi de message asynchrone, le retour doit être représenté s'il existe.

Description dynamique

Sémantique des composants d'un DSE

➤ **3.1 Exemple de message asynchrone :**

- ✓ Représentation de l'allumage et de l'extinction de la lampe par un utilisateur



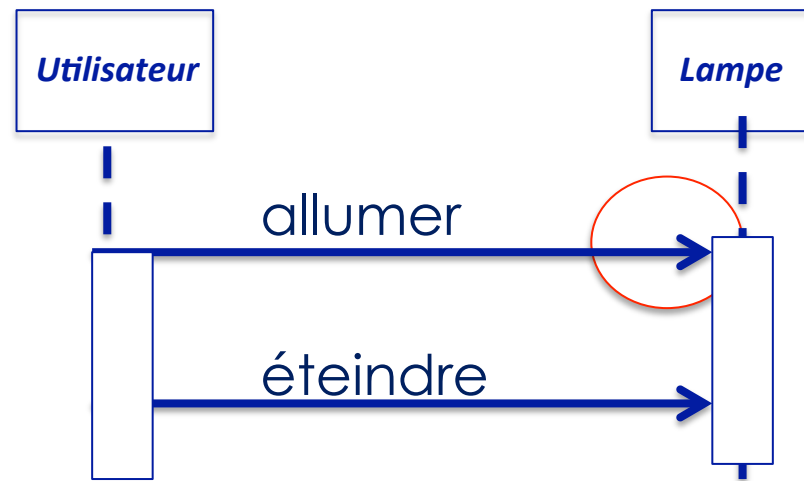
- ◆ Ces deux messages sont asynchrones :
 - L'utilisateur n'attend pas un message de retour de la lampe lui indiquant si elle est allumée (ou éteinte dans le second cas).

Description dynamique

Sémantique des composants d'un DSE

➤ **2.1 Exemple de message asynchrone :**

- ✓ Représentation de l'allumage et de l'extinction de la lampe par un utilisateur



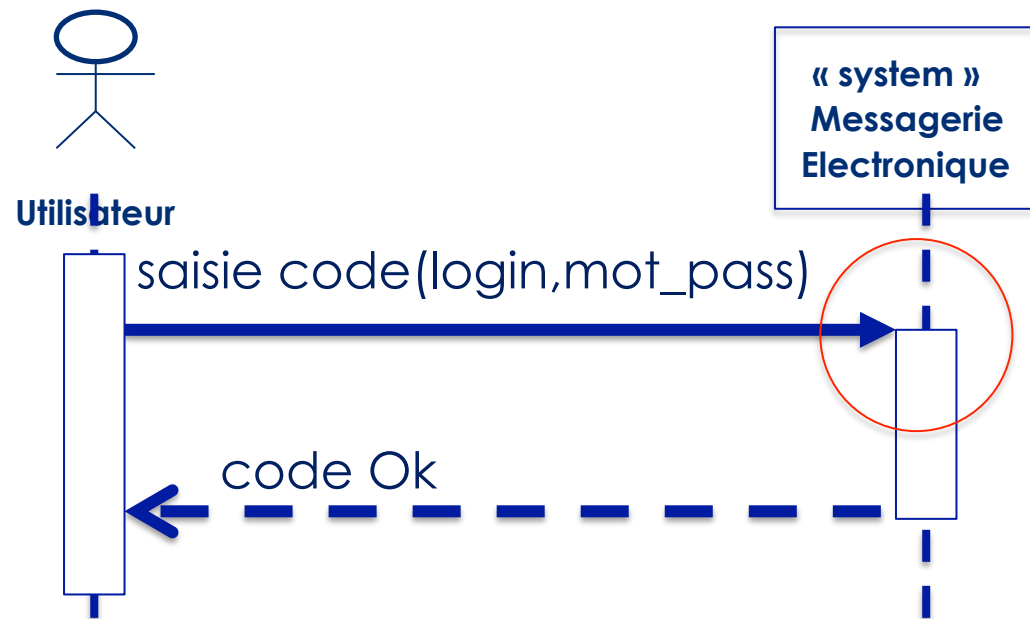
- ◆ Ces deux messages sont asynchrones :
 - L'utilisateur n'attend pas un message de retour de la lampe lui indiquant si elle est allumée (ou éteinte dans le second cas).

Description dynamique

Sémantique des composants d'un DSE

➤ 3.2 Message synchrone :

- ✓ L'émetteur reste en attente de la réponse à son message, avant de poursuivre l'exécution de ses opérations.



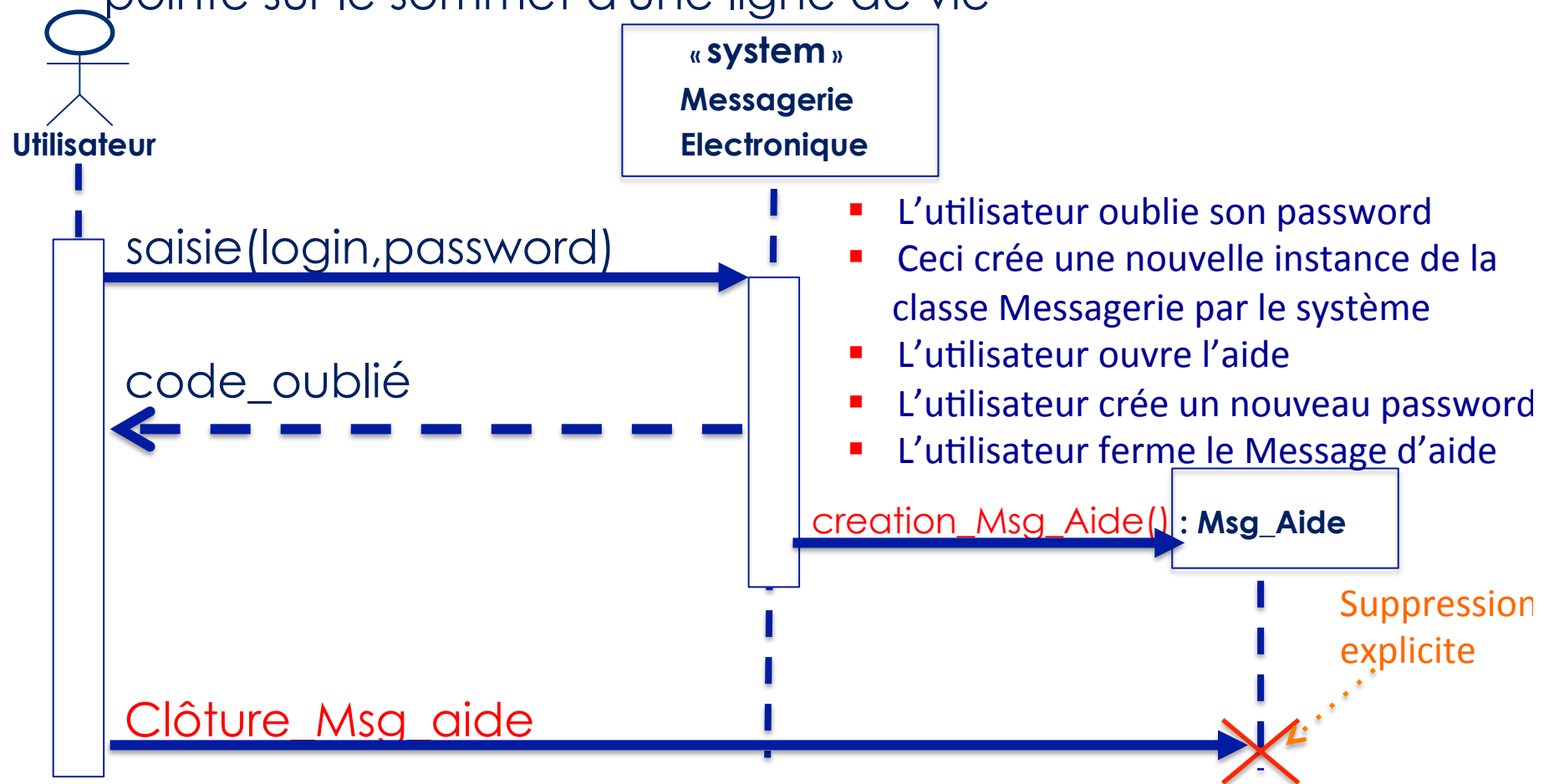
- ◆ L'invocation d'une opération est le type de message le plus utilisé en programmation objet ;
- ◆ Dans la pratique, la plupart des invocations sont synchrones, l'émetteur reste alors bloqué le temps que dure l'invocation de l'opération ;
- ◆ Le message de retour peut ne pas être représenté car il est inclus dans la fin d'exécution de l'opération de l'objet destinataire du message; le retour de message est implicite.

Description dynamique

Sémantique des composants d'un DSE

➤ 3.3 Message de création et de suppression d'instance :

- ✓ La création d'un objet est matérialisée par une flèche qui pointe sur le sommet d'une ligne de vie



Description dynamique

Sémantique des composants d'un DES

➤ 3.3 Message de suppression d'instance :

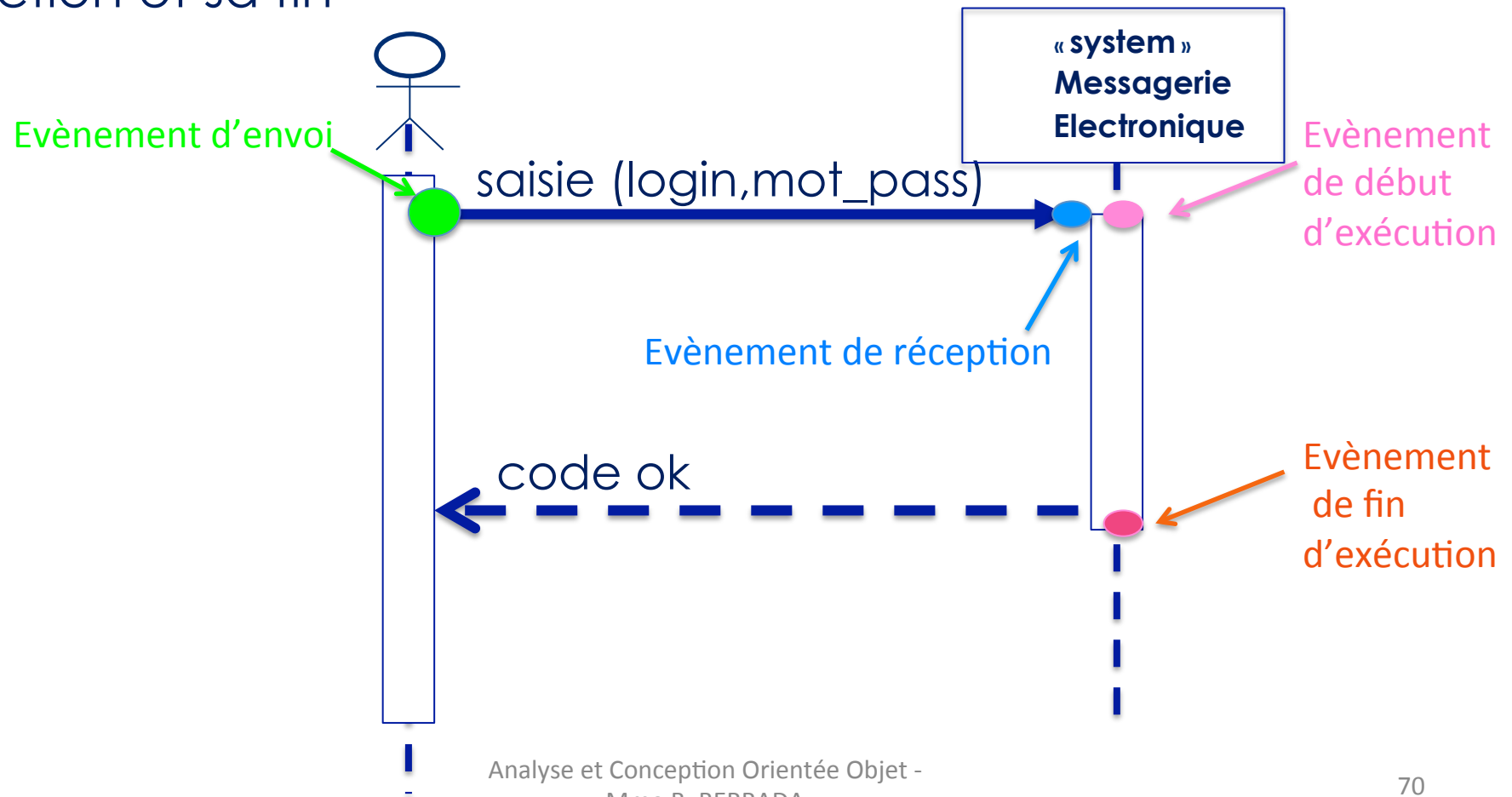
- ◆ La suppression d'un objet (participant) est indiquée par un **X** ;
- ◆ Une flèche de message entrant signifie qu'un objet en supprime explicitement un autre ;
- ◆ Un **X** à la fin d'une ligne de vie, montre que l'objet se supprime lui-même.
- ◆ Il est toujours utile d'utiliser un X pour indiquer qu'un objet n'est plus nécessaire, et qu'il est prêt à être collecté dans un environnement disposant d'un ramasse-miette ;
- ◆ Un **X** est également approprié aux opérations de clôture, pour signifier également que l'objet n'est plus utilisable.

Description dynamique

Sémantique des composants d'un DSE

➤ 3.4 Message et événements :

UML permet de séparer clairement l'envoi du message, sa réception, ainsi que le début de l'exécution de la réaction et sa fin



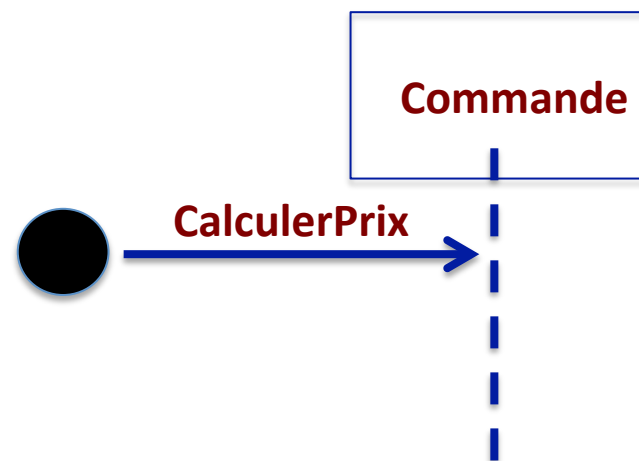
Description dynamique

Sémantique des composants d'un DSE

➤ **3.5 Message trouvé :**

Un message trouvé est tel que l'événement de réception est connu, mais pas l'événement d'émission.

Une flèche partant d'une petite boule noire représente un message trouvé, comme illustré :



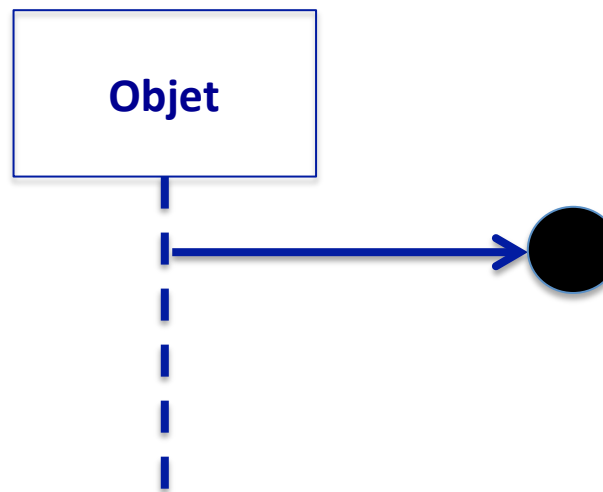
Description dynamique

Sémantique des composants d'un DSE

➤ **3.6 Message perdu :**

Un message perdu est tel que l'événement d'envoi est connu, mais pas l'événement de réception.

Il se représente par une flèche qui pointe sur une petite boule noire tel qu'illustré ci-dessous :

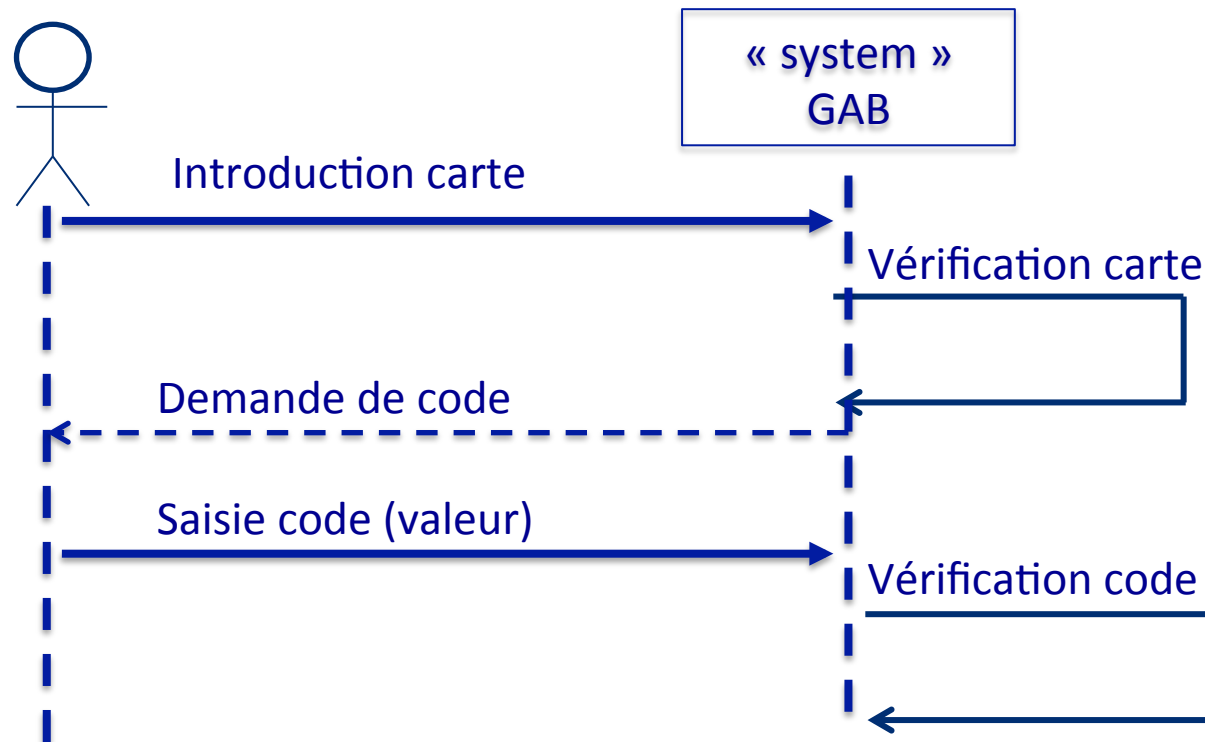


Description dynamique

Diagramme de Séquence (DSE):

➤ 3.7 Message réflexif :

Il représente un message que s'envoie un objet à lui-même (self-call):



Il représente une activité interne à l'objet ou une abstraction d'une autre interaction (qu'on peut détailler dans un autre diagramme de séquence)

Diagramme de Séquence

Fragment d'interaction

Les cadres d'interactions sont utilisés pour distinguer des sous-ensembles qui constituent des Fragments d'interactions.

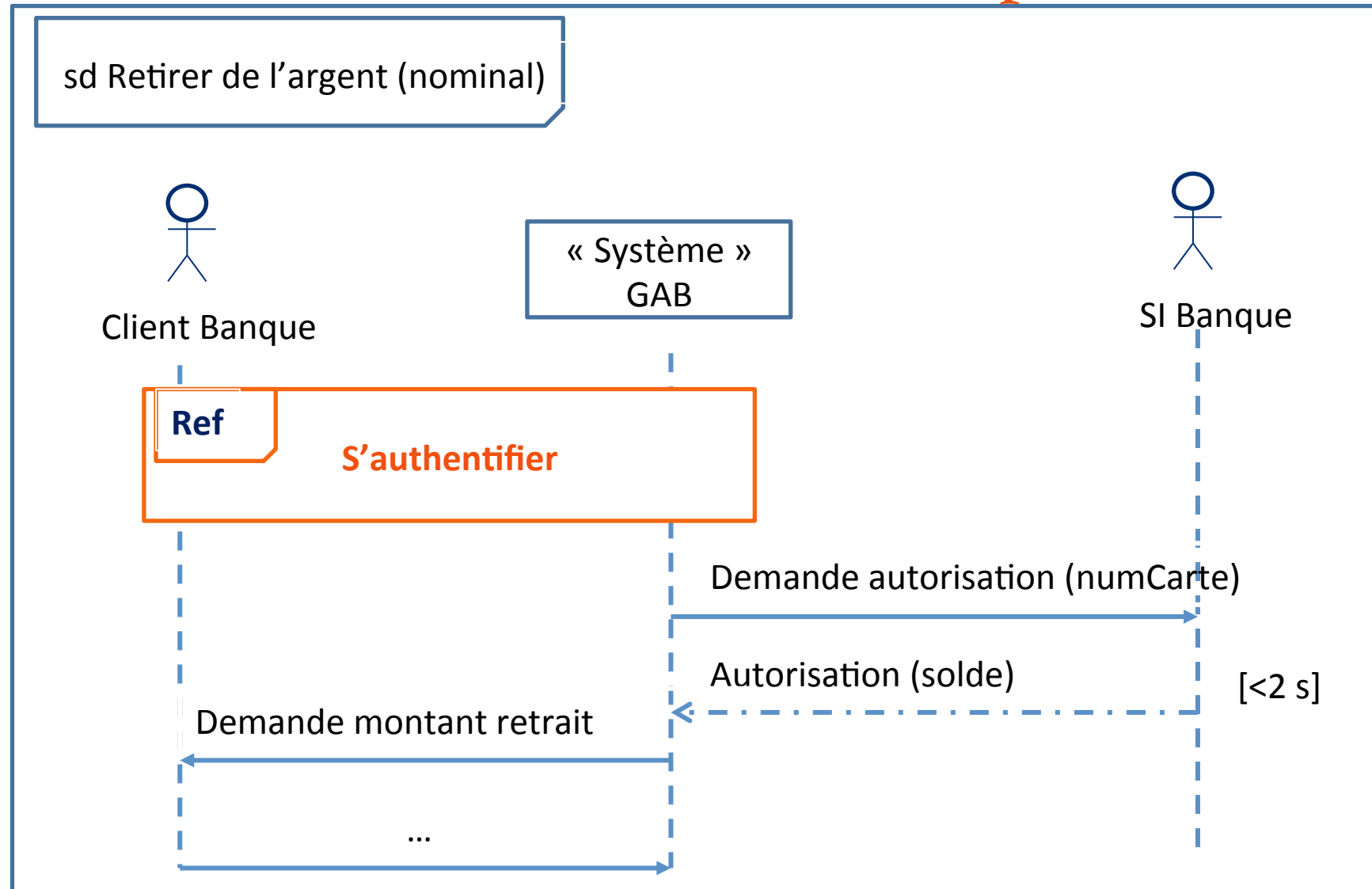
Une indication dans le coin gauche, dans les cas précédents, ref ou opt portant le nom de l'interaction.

Treize opérateurs de fragments d'interaction (combiné) existent :

ref, opt, loop, alt, par, strict/weak, break, ignore/consider, critical, et assertion

Exemple de Fragment d'interaction

Opérateur REF



Exemple de Fragment d'interaction

Opérateur alt

L'opérateur **alt** correspond à une instruction de test avec une ou plusieurs alternatives possibles. Seul le fragment dont la condition est vraie s'exécutera

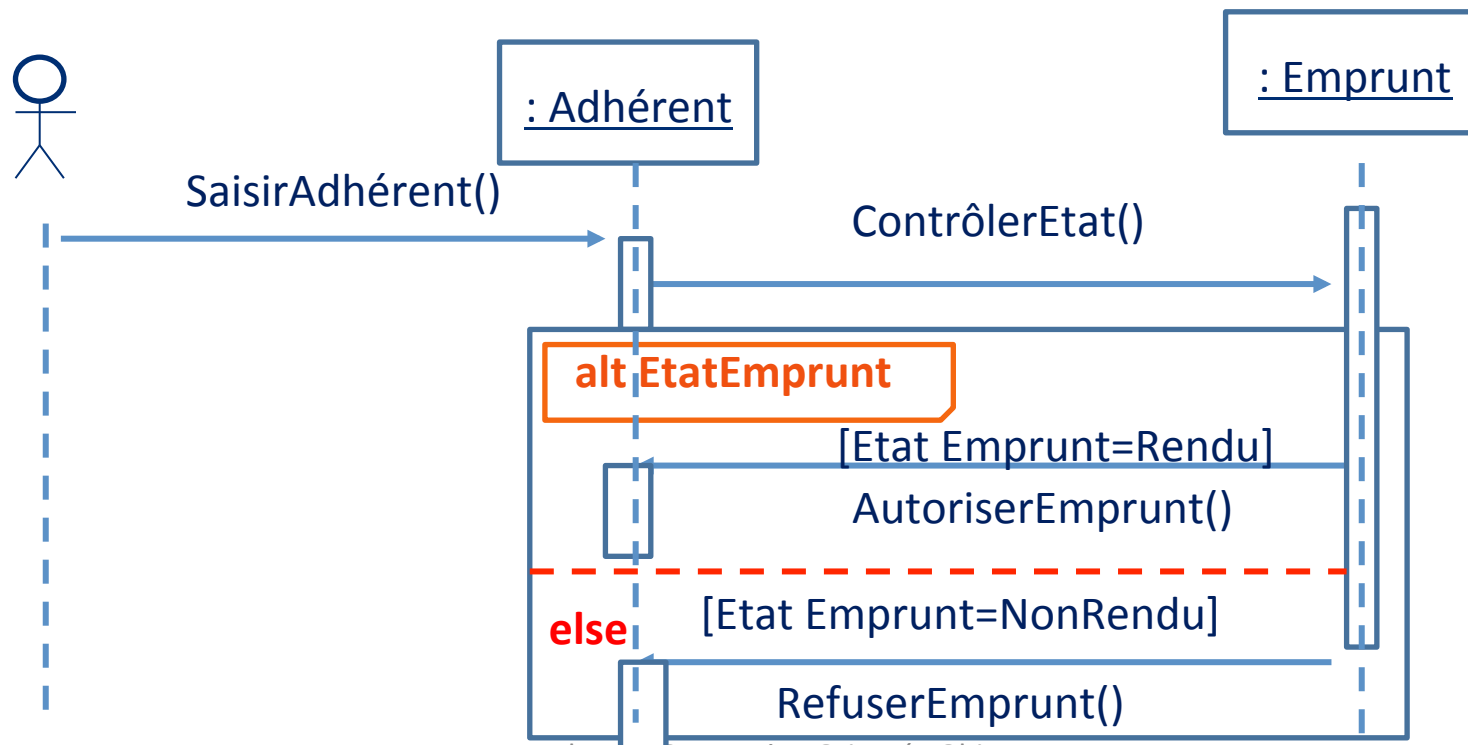


Diagramme de Séquence

Fragment d'interaction

L'opérateur **loop** : correspond à une instruction de boucle qui permet d'exécuter une séquence d'interaction tant qu'une condition n'est pas satisfaite. Une garde indique la condition de l'iteration.

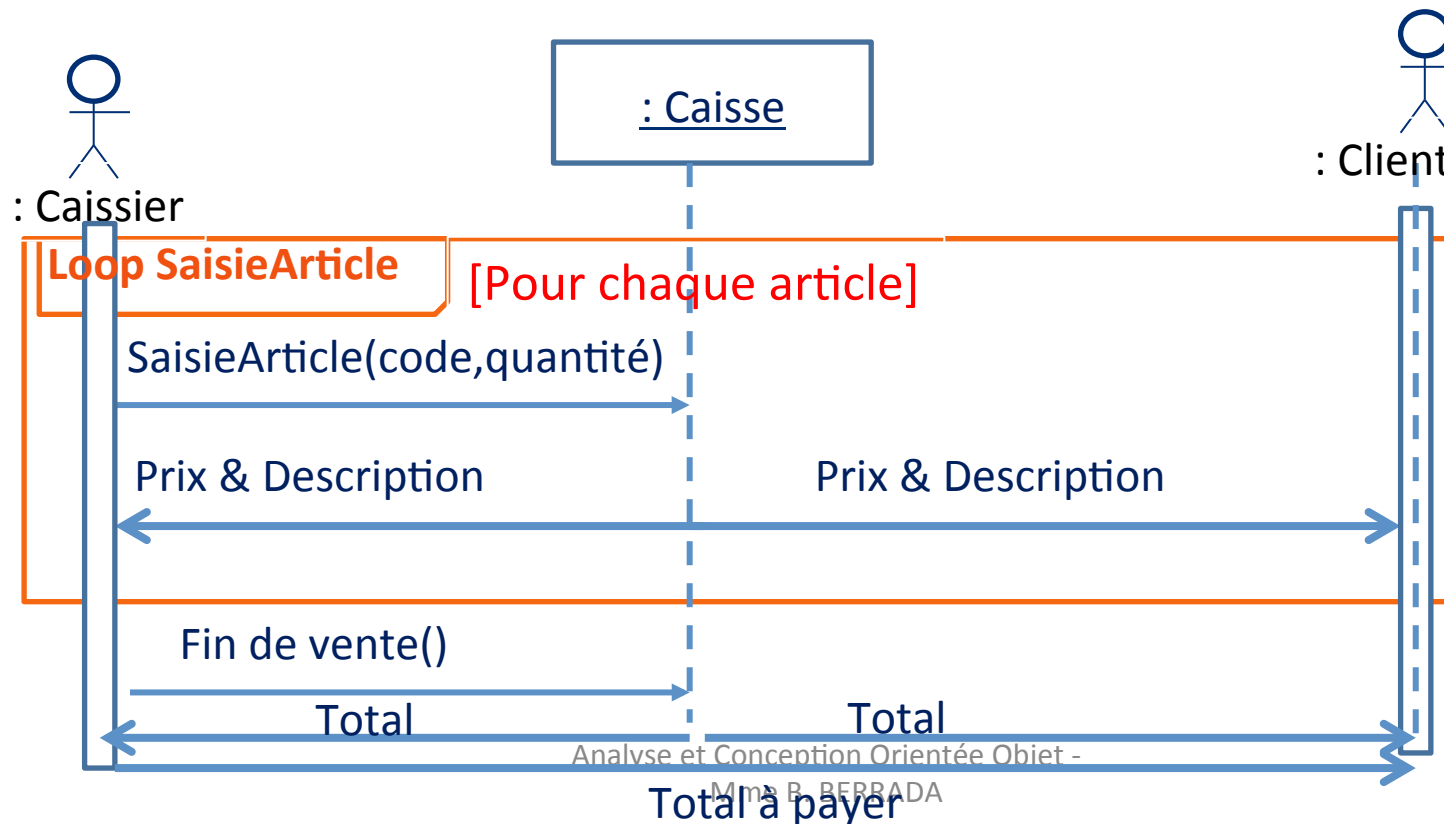


Diagramme de Séquence

Fragment d'interaction : Opérateur par

par : correspond à une instruction qui permet de représenter deux séries d'interactions qui se déroulent en parallèle.

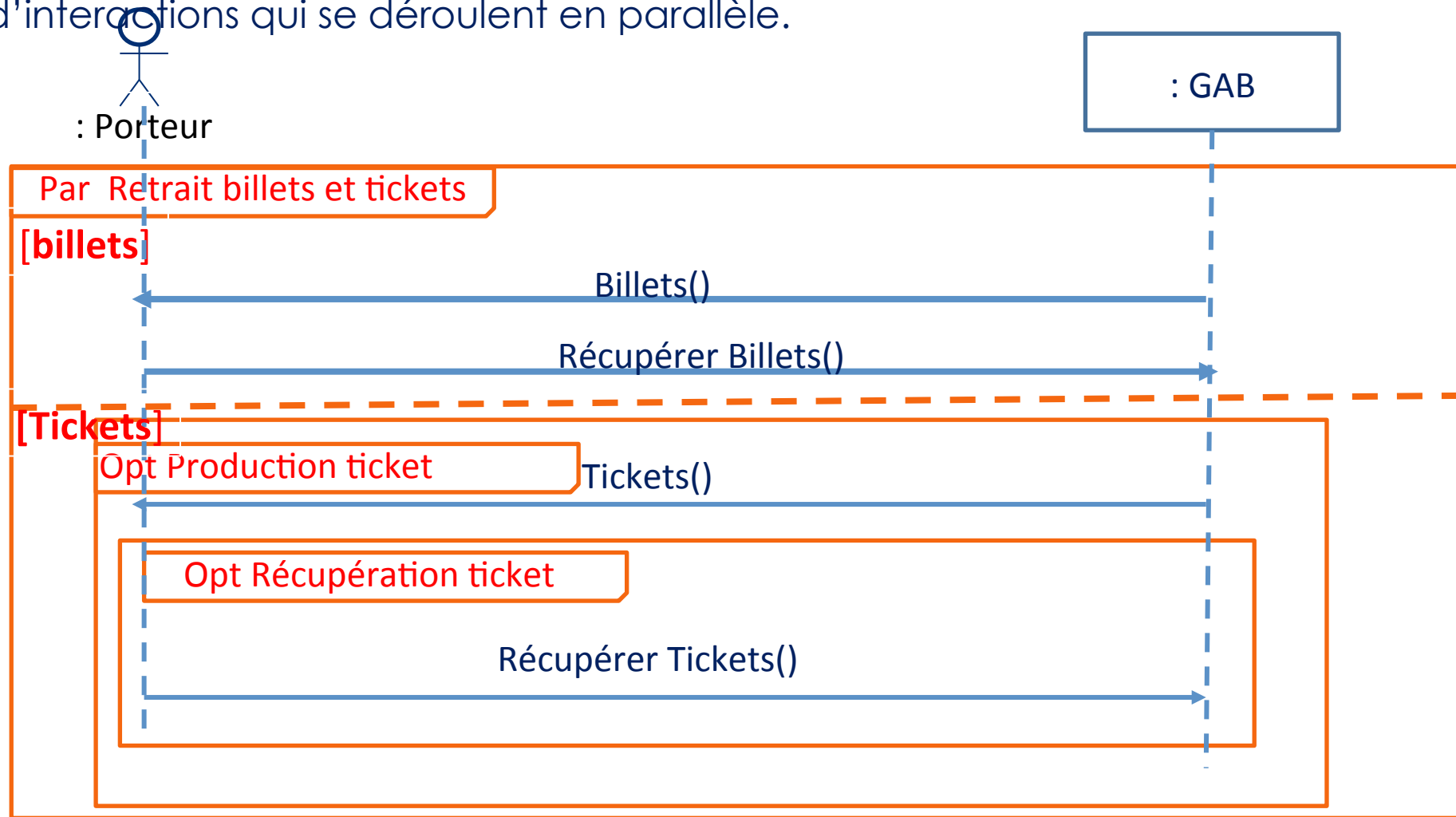


Diagramme de Séquence

Fragment d'interaction

Les autres opérateurs :

strict/weak sequencing : permettent de préciser que l'ordre des opérations(strict) ou pas important

Break : permet de représenter une situation exceptionnelle correspondant à un scénario de rupture (avec une condition de garde, par rapport au scénario général.

Consider/ignore : expriment respect. que des messages doivent être soit obligatoirement présents (consider), soit absents (ignore), sans incidence sur le déroulement des interactions.

Critical : indique une séquence d'interaction critique, donc **ne pouvant être interrompue** de l'importance des opérations traitées.

Diagramme de Séquence

Fragment d'interaction

Suite des autres opérateurs :

assertion : indique qu'une séquence d'interaction est l'unique séquence possible en considérant les messages échangés dans le fragment d'interaction. Toute autre configuration de message est invalide.

négative : indique le fragment représente une interaction invalide.

Dynamique globale

« Interaction Overview Diagram »

Diagramme de vue globale/d'ensemble d'interaction

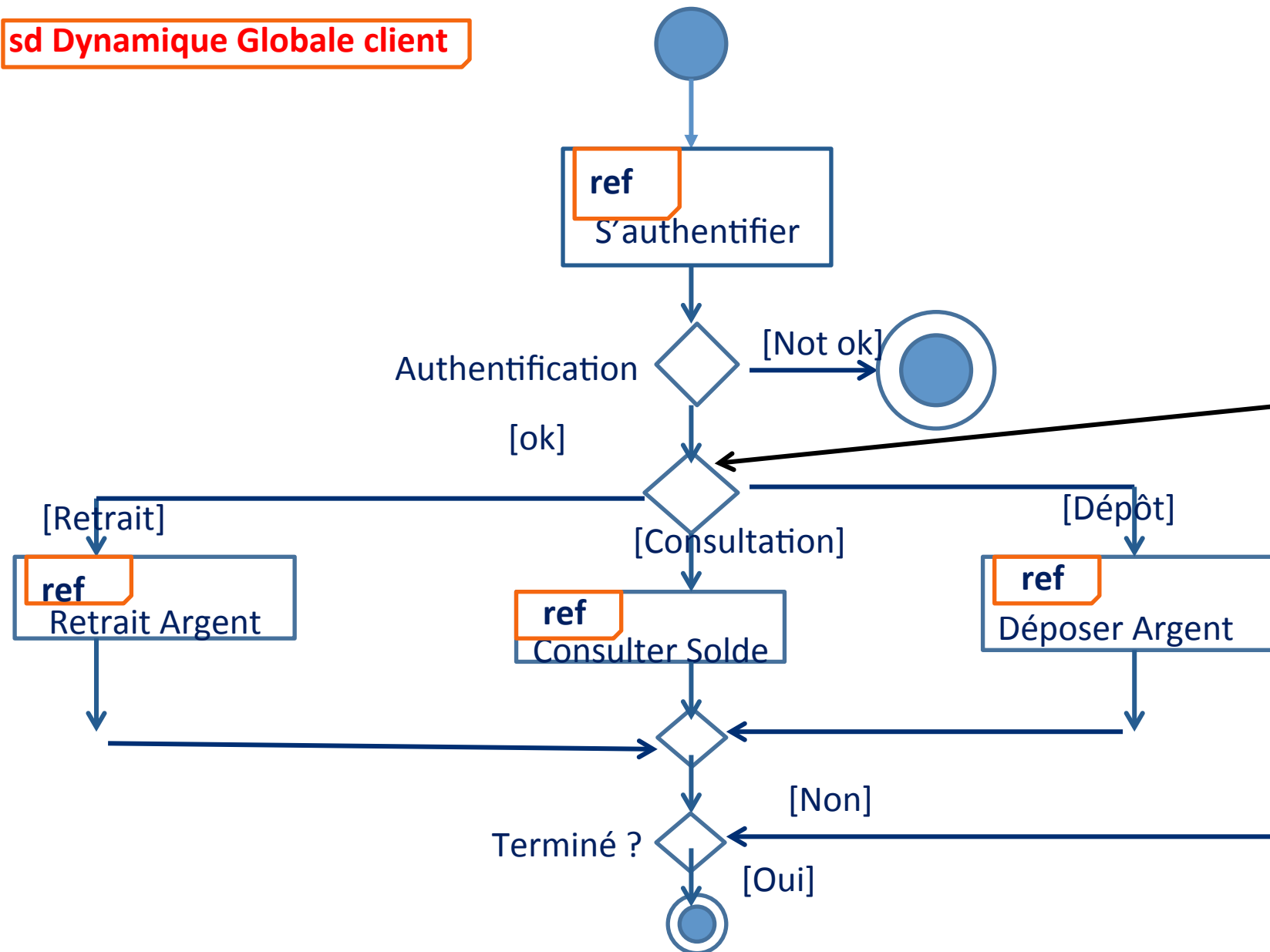
Ce diagramme est une fusion / combinaison du diagramme d'activité et du diagramme de séquence.

Il permet de représenter une vue générale des interactions décrites dans le diagramme de séquence et des flots de contrôle décrits dans les diagrammes d'activités.

Global Interaction Overview

Vue d'ensemble des interactions du client banque

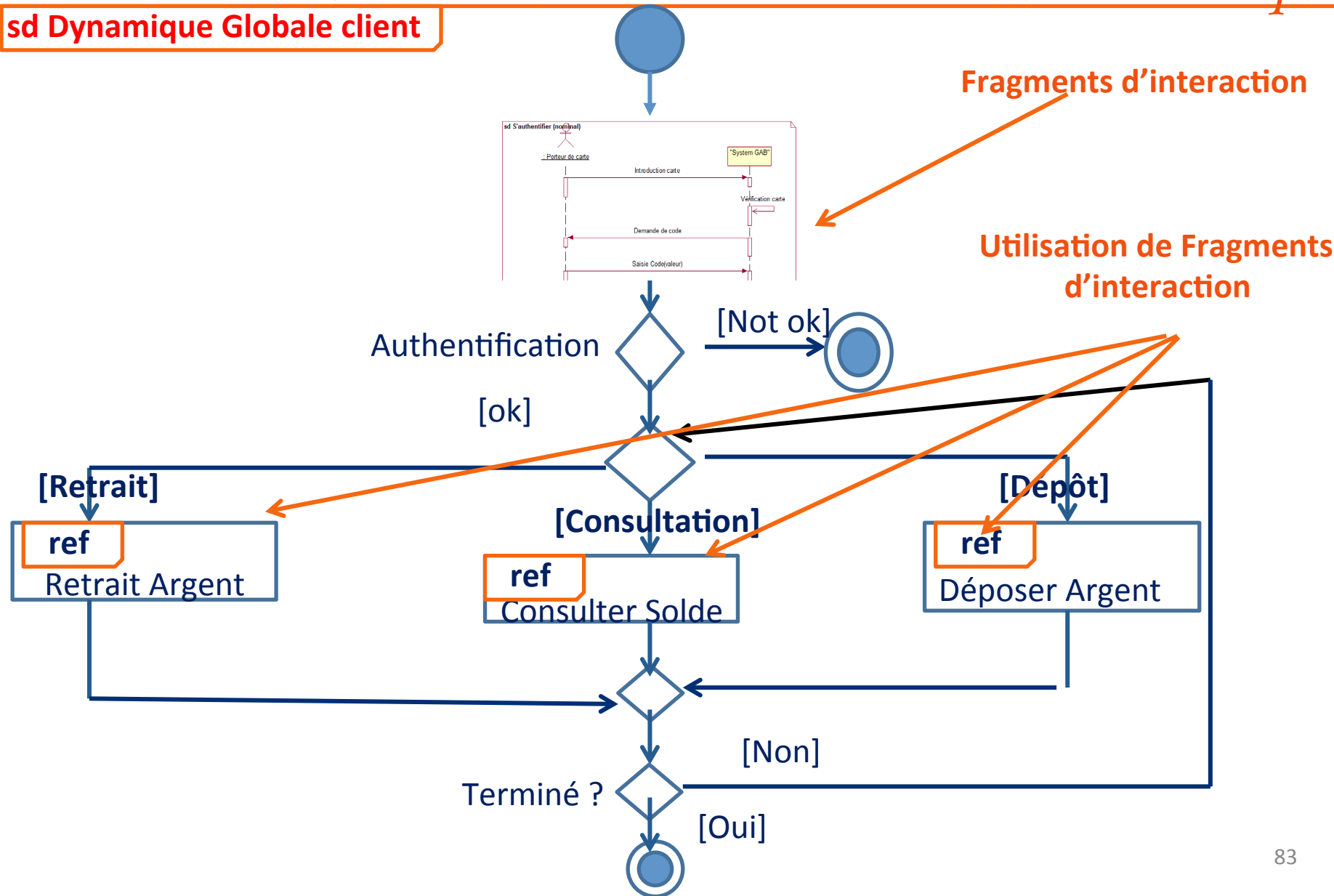
sd Dynamique Globale client



Global Interaction Overview

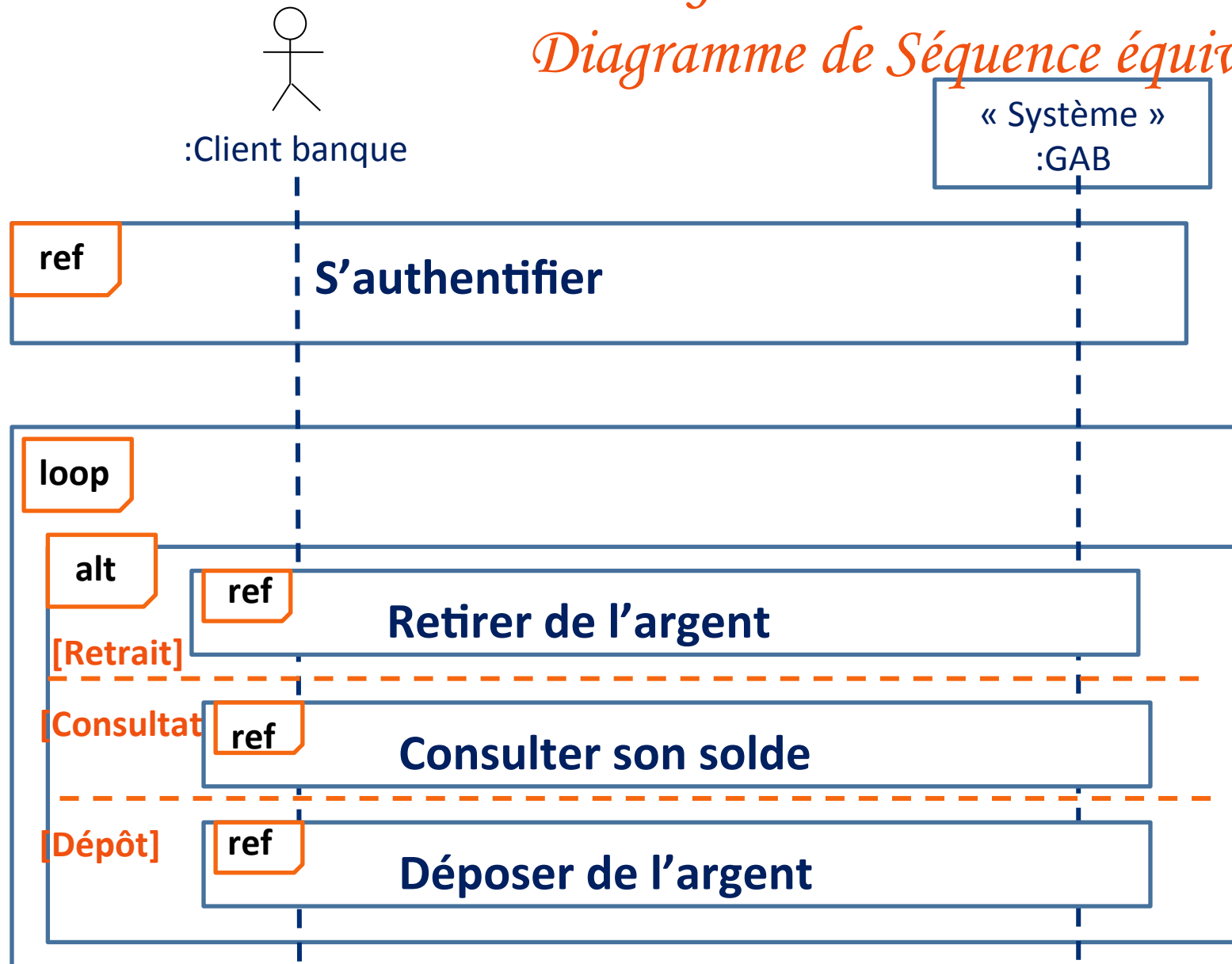
Vue d'ensemble des interactions du client banque

sd Dynamique Globale client



Global Interaction Overview

Diagramme de Séquence équivalent



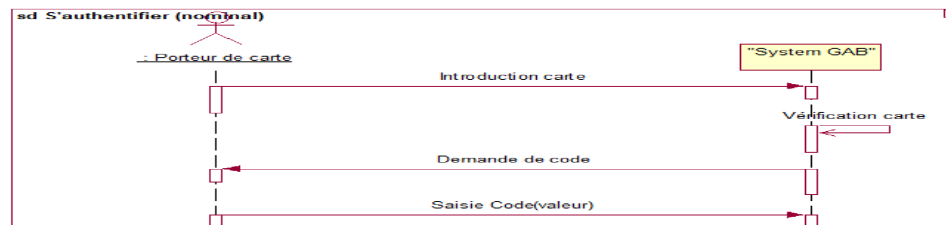
Global Interaction Overview

Les concepts manipulés ?

Le diagramme global d'interaction utilise les concepts du diagramme d'activité auquel on ajoute deux compléments:

➤ *Fragments d'interaction du diagramme de séquence.*

Il s'agit de la notion de fragment d'interaction vue dans le diagramme de séquence mais qui n'est pas détaillé à ce niveau :



➤ *Utilisations de fragments d'interaction à l'aide de l'opérateur ref, comme le montre l'exemple ci-dessous :*

ref

S'authentifier

Diagramme dynamique

Diagramme d'Activité

Le Diagramme d'activité décrit le comportement interne des opérations ou des cas d'utilisation.

C'est une technique intéressante pour :

- ***Représenter la logique comportementale***
- ***Décrire le processus métier***
- ***Représenter les enchaînements d'activités (workflow ou ordre d'exécution, ou encore règles de séquencement).***

Les concepts du diagramme d'activité (DAC)

Au cœur du DAC, 2 concepts essentiels :

1. ACTION : Elle correspond à un traitement qui modifie l'état du système.
Elle peut être appréhendée soit au niveau :

- Élémentaire, proche d'une instruction en termes de programmation
- Global, correspondant à une ou plusieurs opérations.

Formalisme



Nom de l'action

*N.B. : Un rectangle avec des coins sont arrondis
(comme pour les états-transitions)*

Exemple



Saisir commande

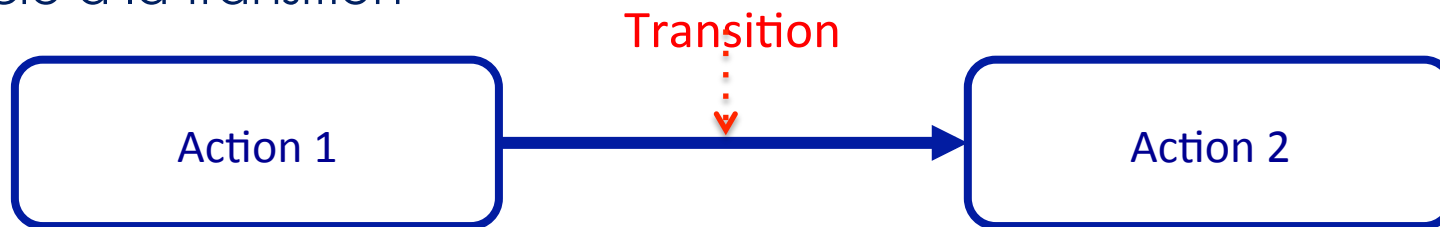
Diagramme d'activité

Action

Concepts particuliers au cœur du DAC : Action et Activité

Action : Transition et flot de contrôle

- Dès qu'une action est achevée, une **transition** automatique peut être déclenchée vers l'action suivante. Il n'y a donc pas d'événement* associé à la transition



- L'enchaînement des actions constitue le **flot de contrôle**

* **Rappel : Un événement** se produit généralement lors de la réception explicite d'un signal ou d'un message, lorsqu'une condition devient vraie, écoulement d'une période de temps (expression temporelle)

Diagramme d'activité

Activité

Concepts particuliers au cœur du DAC : Action et Activité

2. Activité : Elle représente le comportement d'une partie du système **en terme d'action et de transitions**




Une activité est composée de 3 types de nœuds :

- **Action**, symbolisée dans un nœud d'exécution
- **Nœud de contrôle**, symbolisé dans un nœud (initial, final, bifurcation, jonction, fusion, décision, flux de sortie)
- **Nœud d'objet**, symbolisé par objet généré par une action dans une activité et utilisé par d'autres actions (i.e. une valeur d'objet passée par copie (pin d'entrée et de sortie))

Diagramme d'activité


Activité : Type de Nœuds

Une activité est composée de 3 types de nœuds :

- **Action**, symbolisée dans un nœud d'exécution 
- **Nœud de contrôle**, symbolisé dans un nœud initial, final, bifurcation, jonction, fusion, décision, flux de sortie, pin d'entrée et de sortie)

 Nœud initial

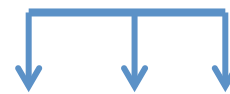
 Nœud Final

 Nœud de décision
(choix)

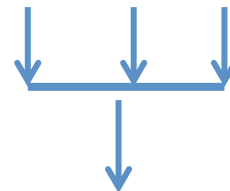
- **Nœud d'objet.**



Nœud de fin de flot



Nœud de bifurcation
(fourche)



Nœud de jonction
(Synchronisation)

Diagramme d'activité

Activité : Type de Nœuds

 Nœud initial : marque le début d'une activité

 Nœud Final : marque la fin d'une activité



Diagramme d'activité

Activité : Type de Nœuds



Nœud de décision

- Il n'a qu'un seul flot en entrée
- Il permet de faire un choix entre plusieurs flots sortants en fonction des conditions de chaque flot.
- On peut aussi utiliser que 2 flots de sortie :
 - ✓ Le premier correspondant à la condition vérifiée
 - ✓ Le second traitant le cas contraire (condition non vérifiée)

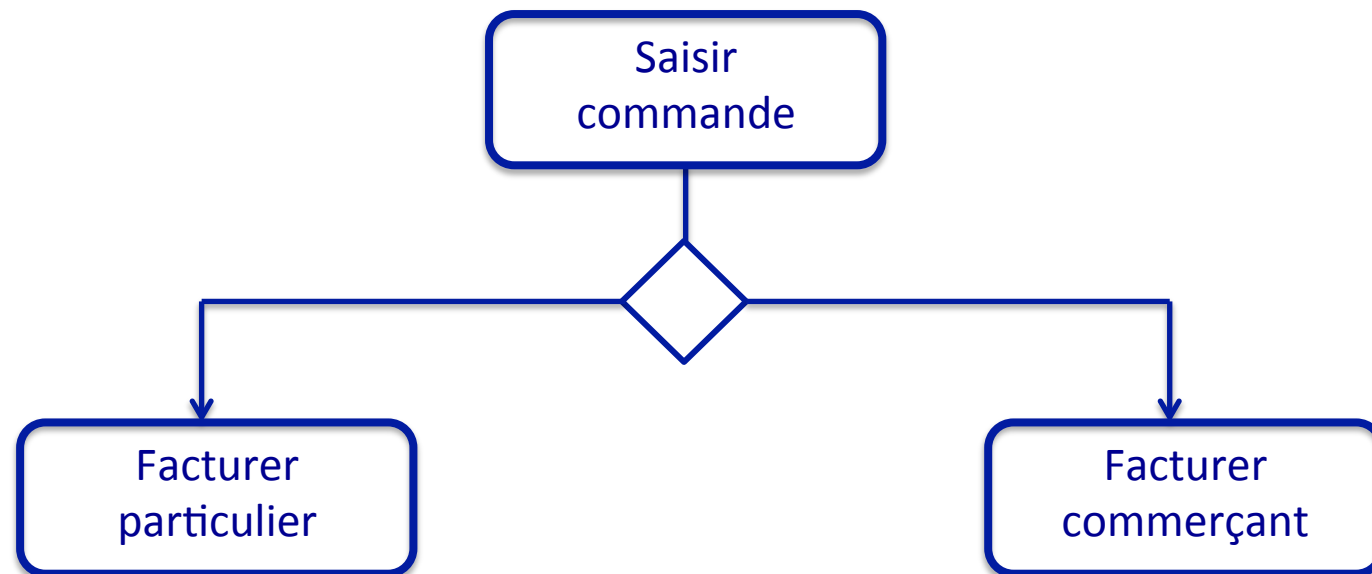


Diagramme d'activité

Activité : Type de Nœuds

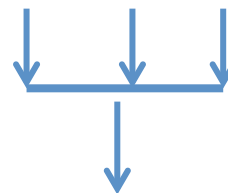
- Nœud initial : marque le début d'une activité
- ⦿ Nœud Final : marque la fin d'une activité
- ◇ Nœud de décision : permet de le faire un choix (choix)



Nœud de fin de flot



Nœud de bifurcation
(fourche)



Nœud de jonction
(Synchronisation)

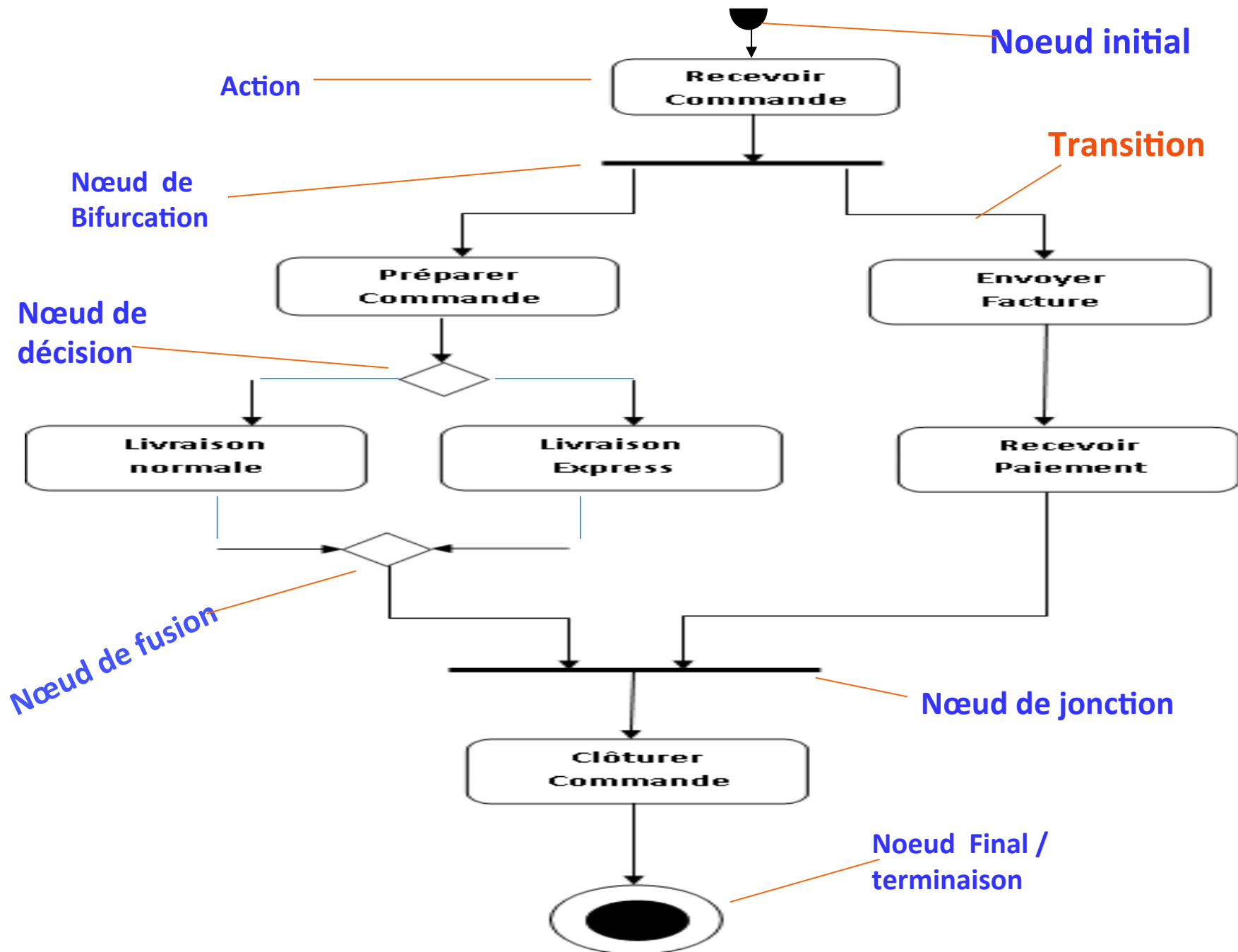
Description dynamique

Diagramme d'activité

Exemple de Diagramme d'activité :

Prenons en exemple un diagramme d'activités simple, pour représenter le processus métier de traitement d'une commande, depuis sa réception jusqu'à la clôture de cette commande.


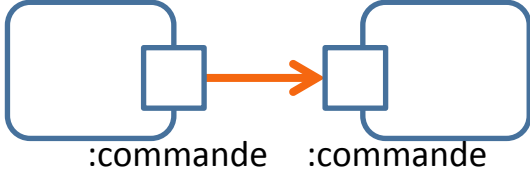
Le processus se traduit par le diagramme d'activité suivant :



Description dynamique

Diagramme d'activité

1. Diagramme d'activité et autres concepts particuliers :

- Pin d'entrée et de sortie : 
- Flots de données et Nœuds d'objet : 
- Décomposition des actions
- Partitions
- Signaux

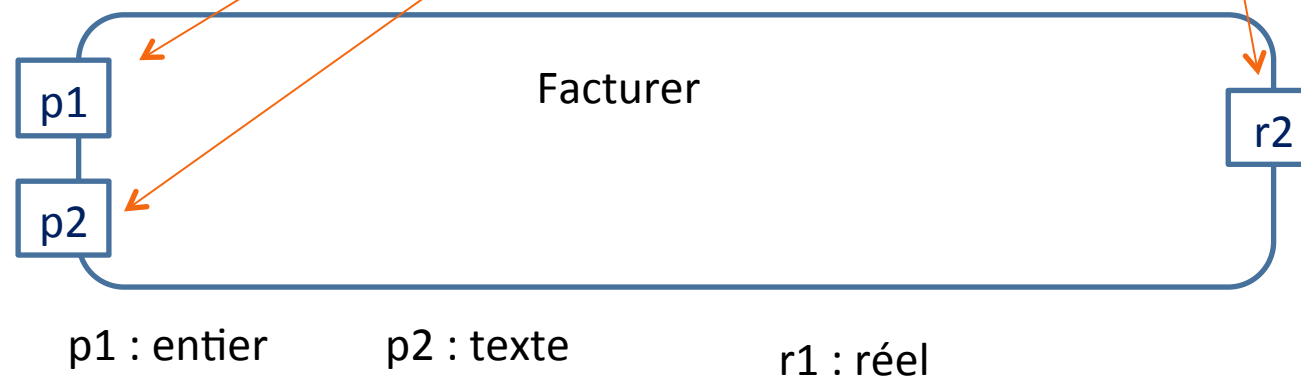
Description dynamique

Diagramme d'activité

➤ Pin d'entrée et de sortie :  (Paramètre)

Un pin d'entrée représente un paramètre que l'on peut spécifier en entrée ou en sortie d'une action

Un nom de donnée et un type peuvent être associés au PIN



Description dynamique

Diagramme d'activité

➤ **Flot de données : Définition**

UML 2 emploie le terme Flot (*Flow*) ou Arc pour décrire les connexions entre deux actions.

La forme d'arc la plus élémentaire est **une simple flèche** entre deux actions.



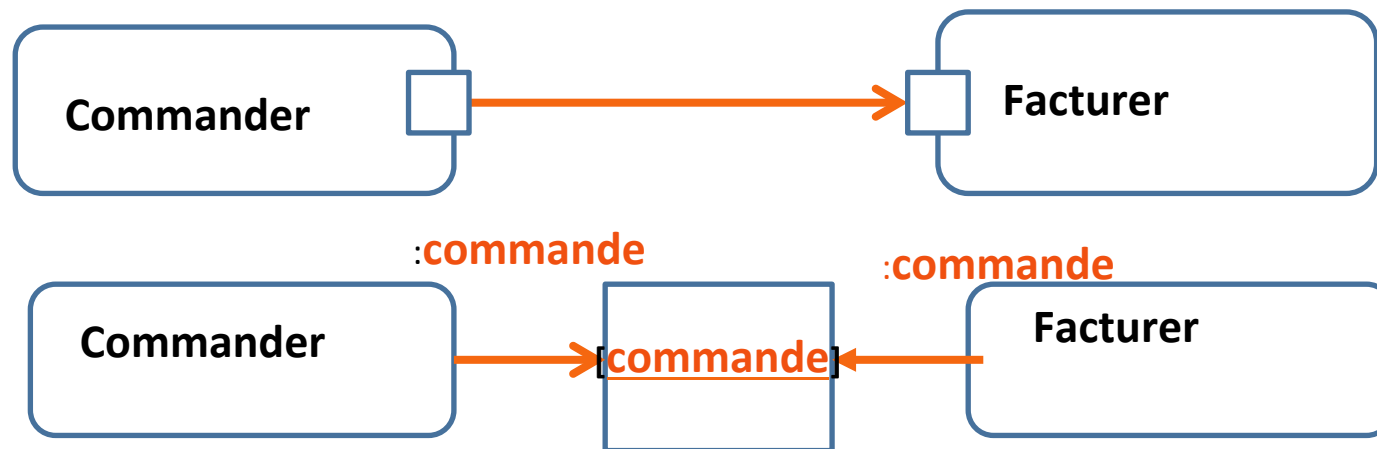
Description dynamique

Diagramme d'activité

➤ **Flots de données et Nœud d'objet**

Un nœud d'objet permet de représenter le Flot de données véhiculé entre les actions.

Les objets peuvent se représenter soit en utilisant le pin d'objet soit en représentant explicitement un objet



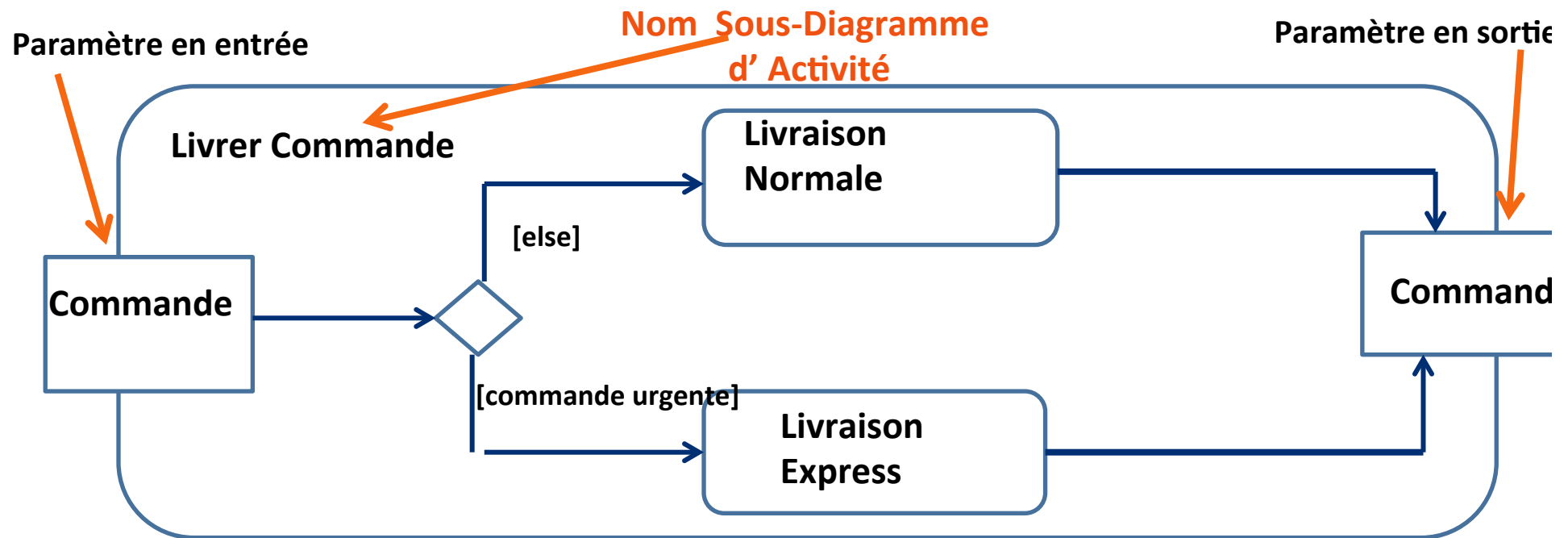
Description dynamique

Diagramme d'activité

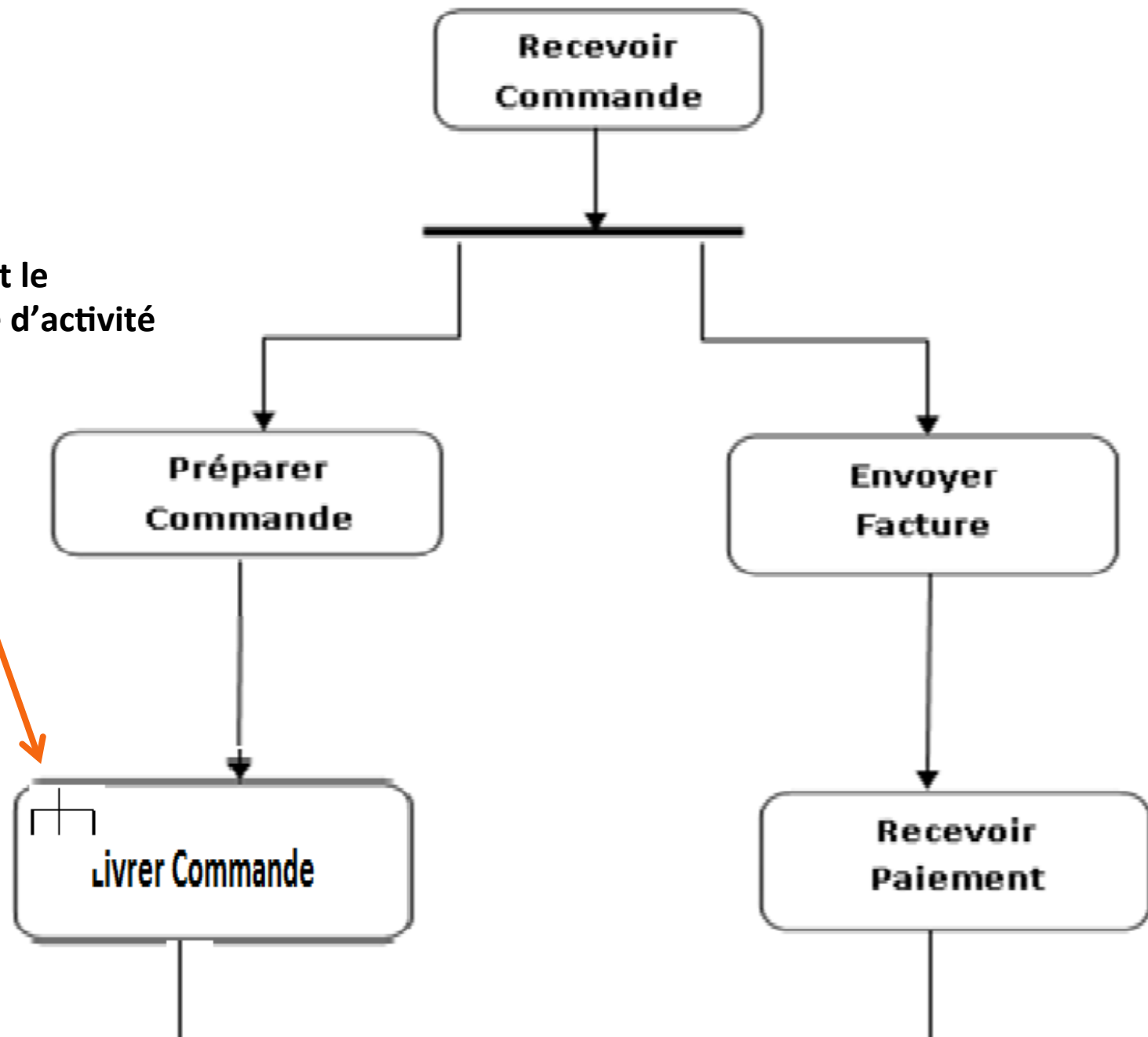
➤ Décomposition d'actions

Les actions peuvent être décomposées en sous-activités.

La logique de la livraison peut-être définie en tant qu'activité-propre:



Râteau indiquant le
sous-diagramme d'activité



Description dynamique

Diagramme d'activité

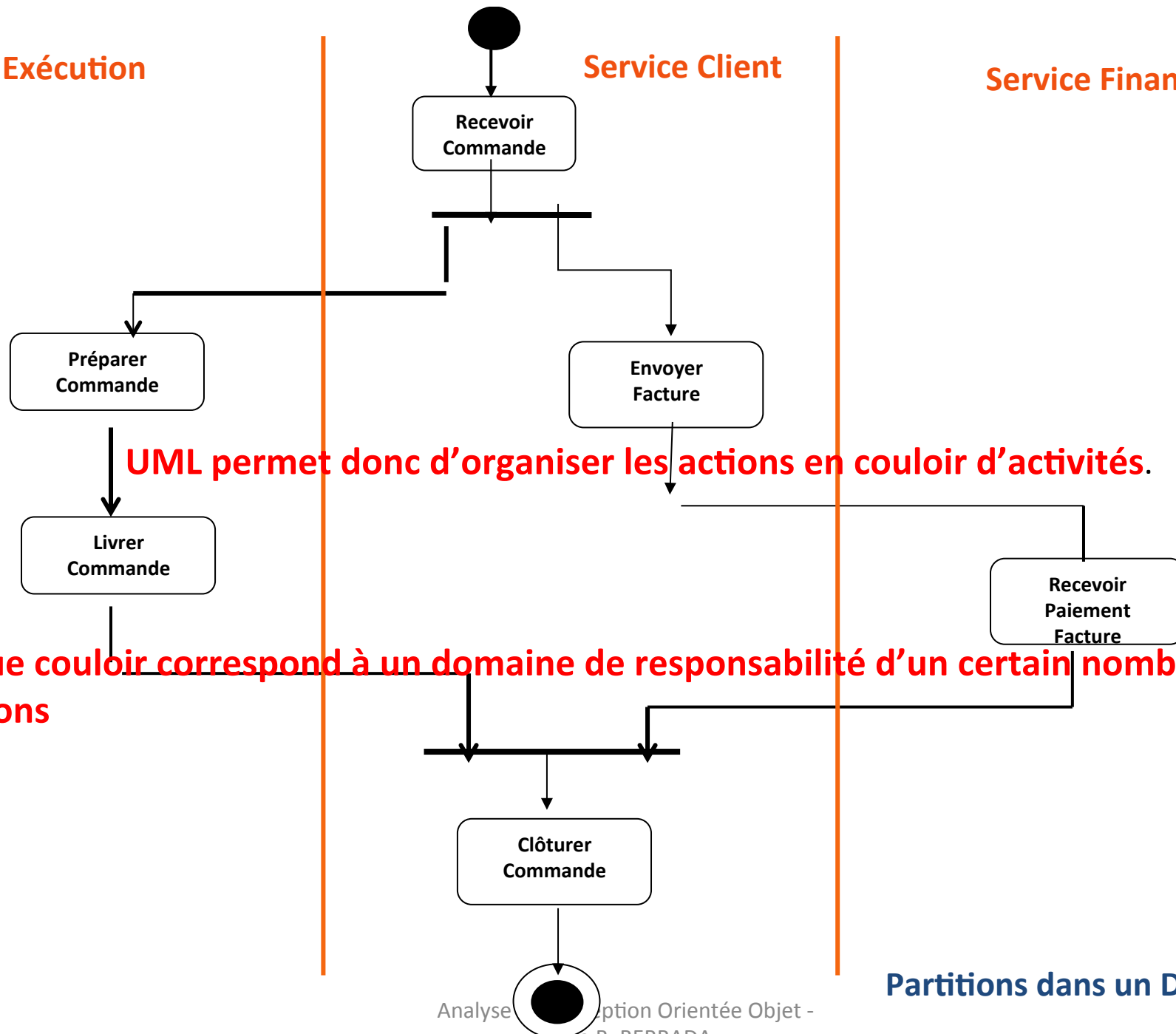
Partitions : Qui fait quoi ?

- Diviser un diagramme d'activité en partitions montrant quelles actions sont exécutées par une classe ou une unité organisationnelle.

Exécution

Service Client

Service Financier



Partitions dans un DAC

Description dynamique

Diagramme d'activité

Signaux temporels : Quand ?

Les diagrammes d'activités, comme les diagrammes de temps, peuvent représenter les actions de communications liées à certains événements.

Ils représentent les événements auxquelles les actions peuvent être liées.

Les types d'événements peuvent être :

- Signal
- Écoulement de temps



Description dynamique

Diagramme d'activité

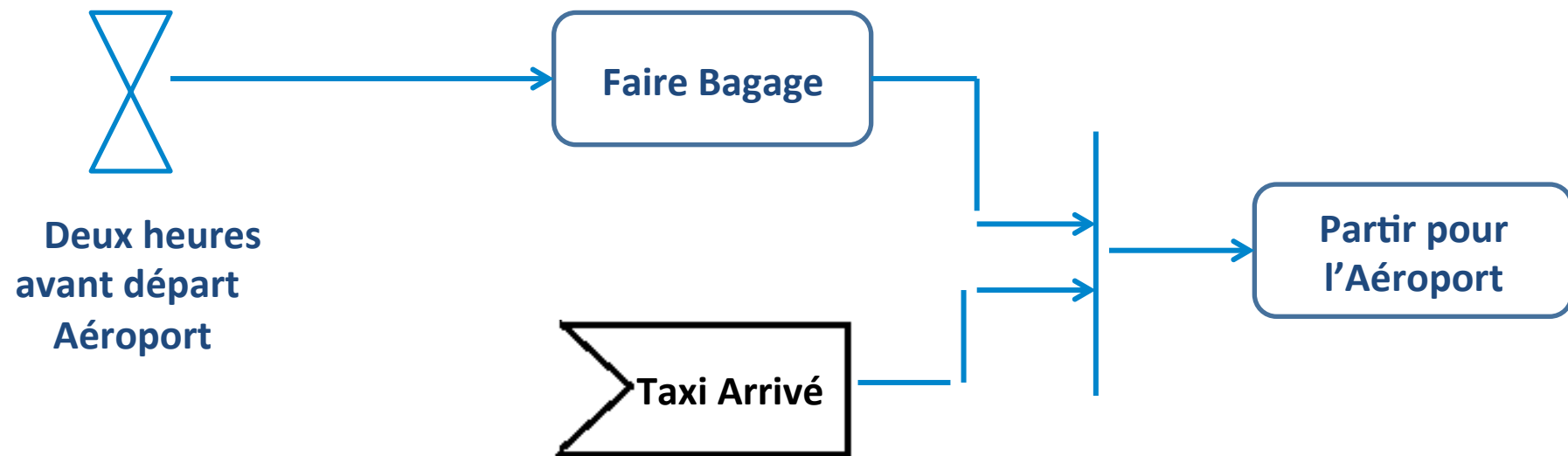
Exemple :

1. Nous devons avoir fait nos bagages deux heures avant de partir pour l'aéroport.
2. Nous ne pouvons pas partir avant que le taxi ne soit arrivé.
3. Si le taxi arrive avant que les bagages soient faits, il doit attendre avant de partir.

Description dynamique

Diagramme d'activité

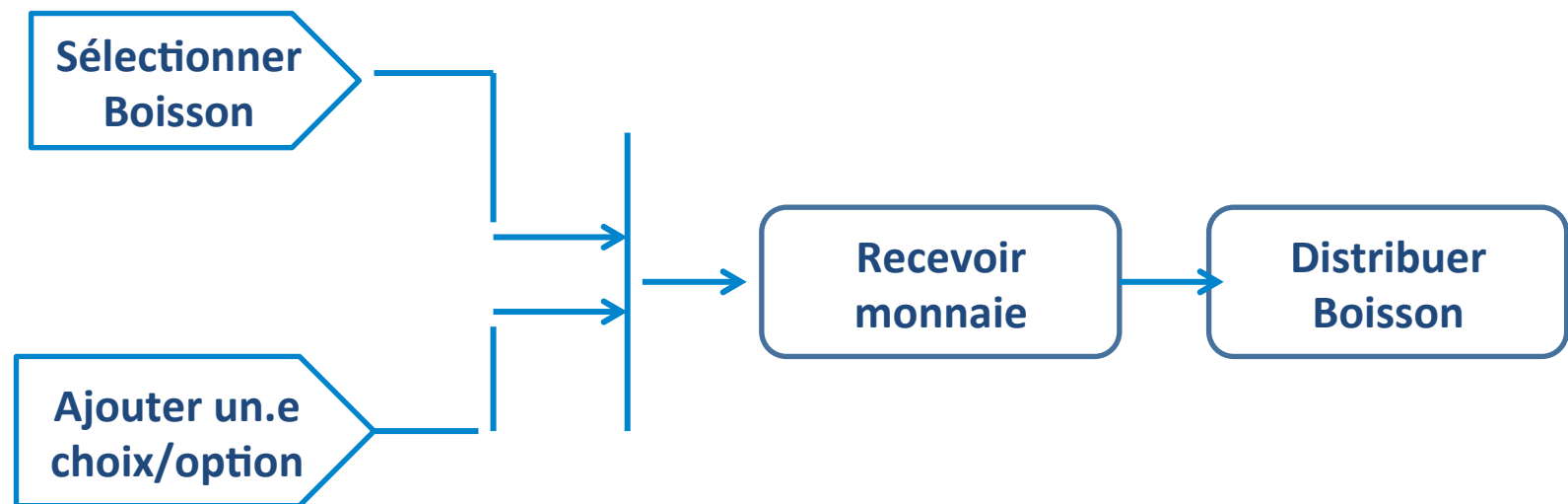
Résolution exemple :



Description dynamique

Diagramme d'activité

Exemple 2 : Distributeur de boisson



Description dynamique

Diagramme d'activité

Quand utiliser des diagrammes d'activités ?

La grande force des diagrammes réside dans le fait qu'ils permettent :

- ✓ La représentation du parallélisme : Workflow
- ✓ La représentation d'un Organigramme

Technique similaire : Réseau de pétri

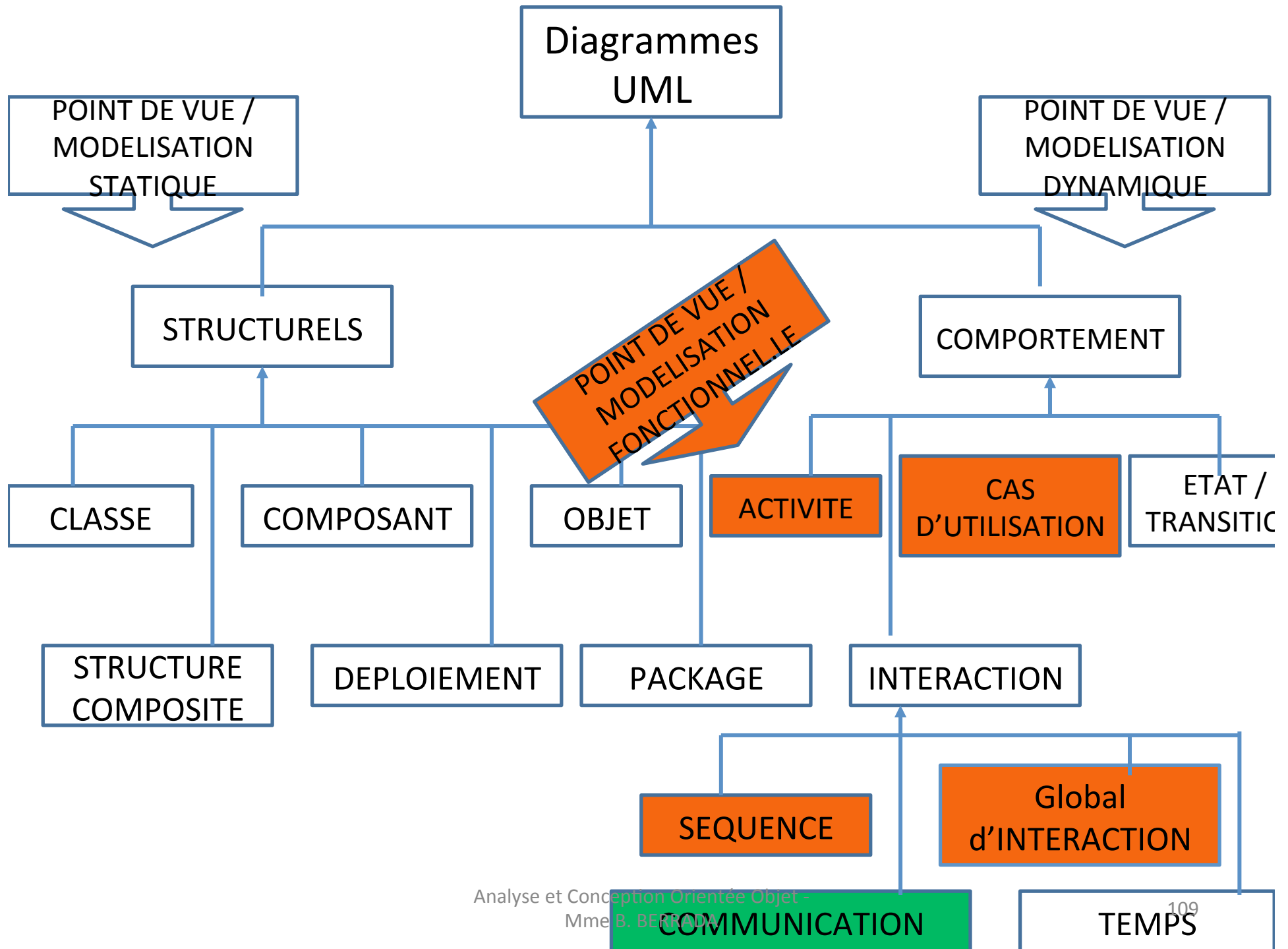


Diagramme de Communication

Le diagramme de communication (ex-collaboration), un diagramme particulier de diagramme d'interaction, met l'accent sur :

- les liaisons de données entre participants à l'interaction

A la différence du DSE, le DC permet de :

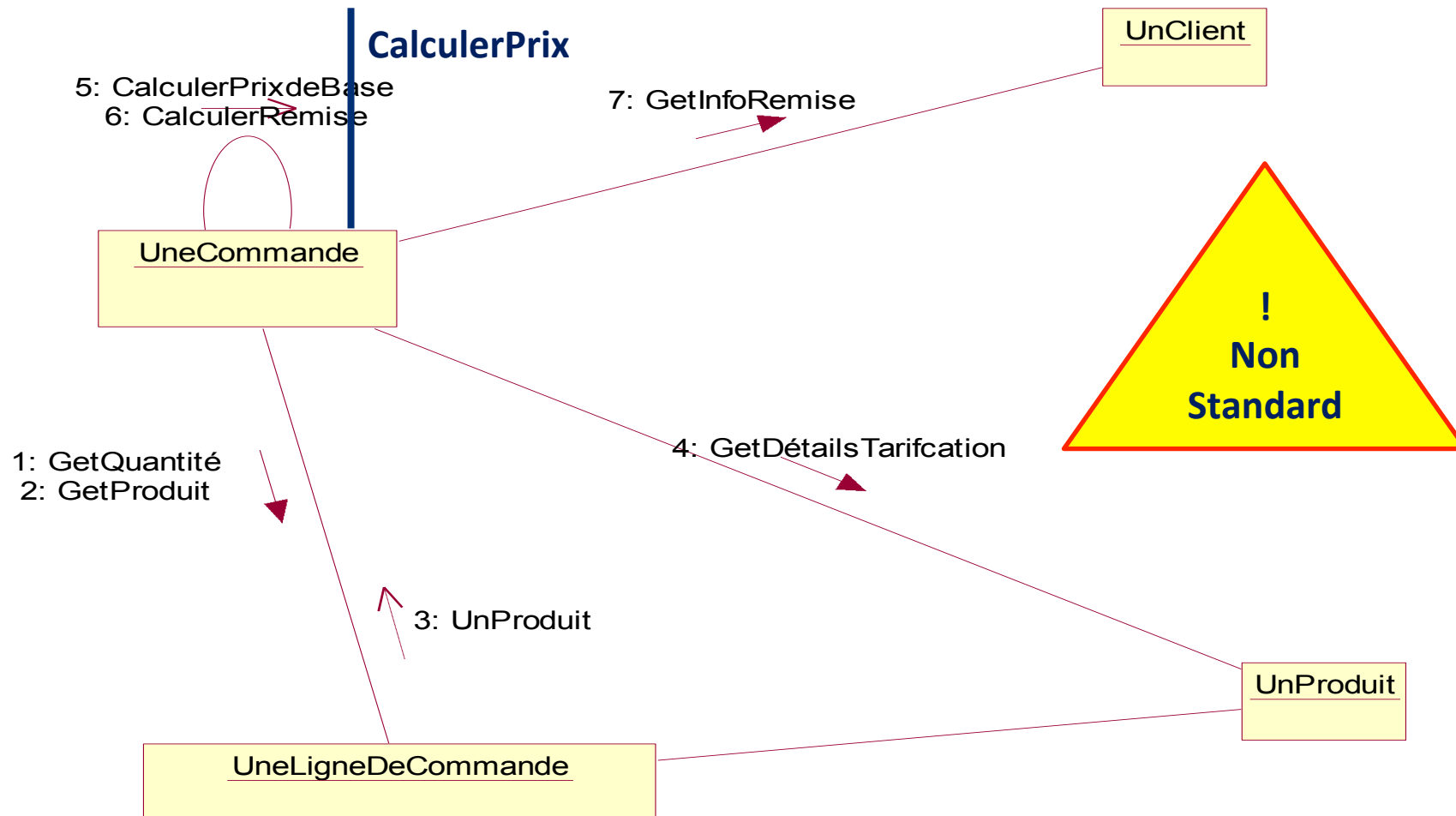
- Placer librement les participants;
- Tracer des liens pour montrer comment ils sont connectés
- Numéroter les messages pour suivre les séquences d'appel.

Le DC met davantage l'accent sur l'aspect spatial des échanges que sur l'aspect temporel

NB : Le DSE est transformé en DC à l'aide de la fonction F5

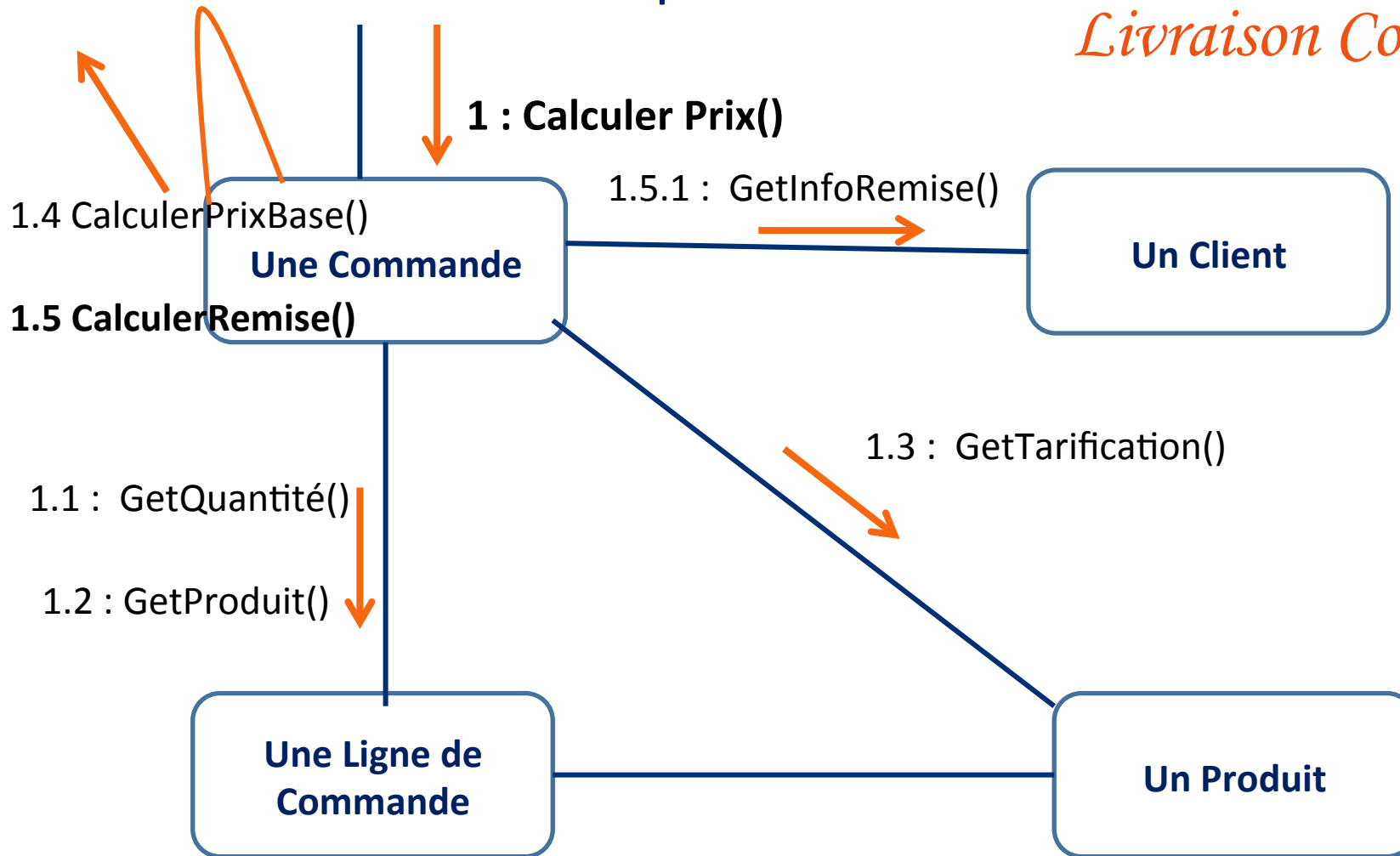
Diagramme de Communication Livraison Commande

Notation couramment utilisée



Notation décimale imbriquée

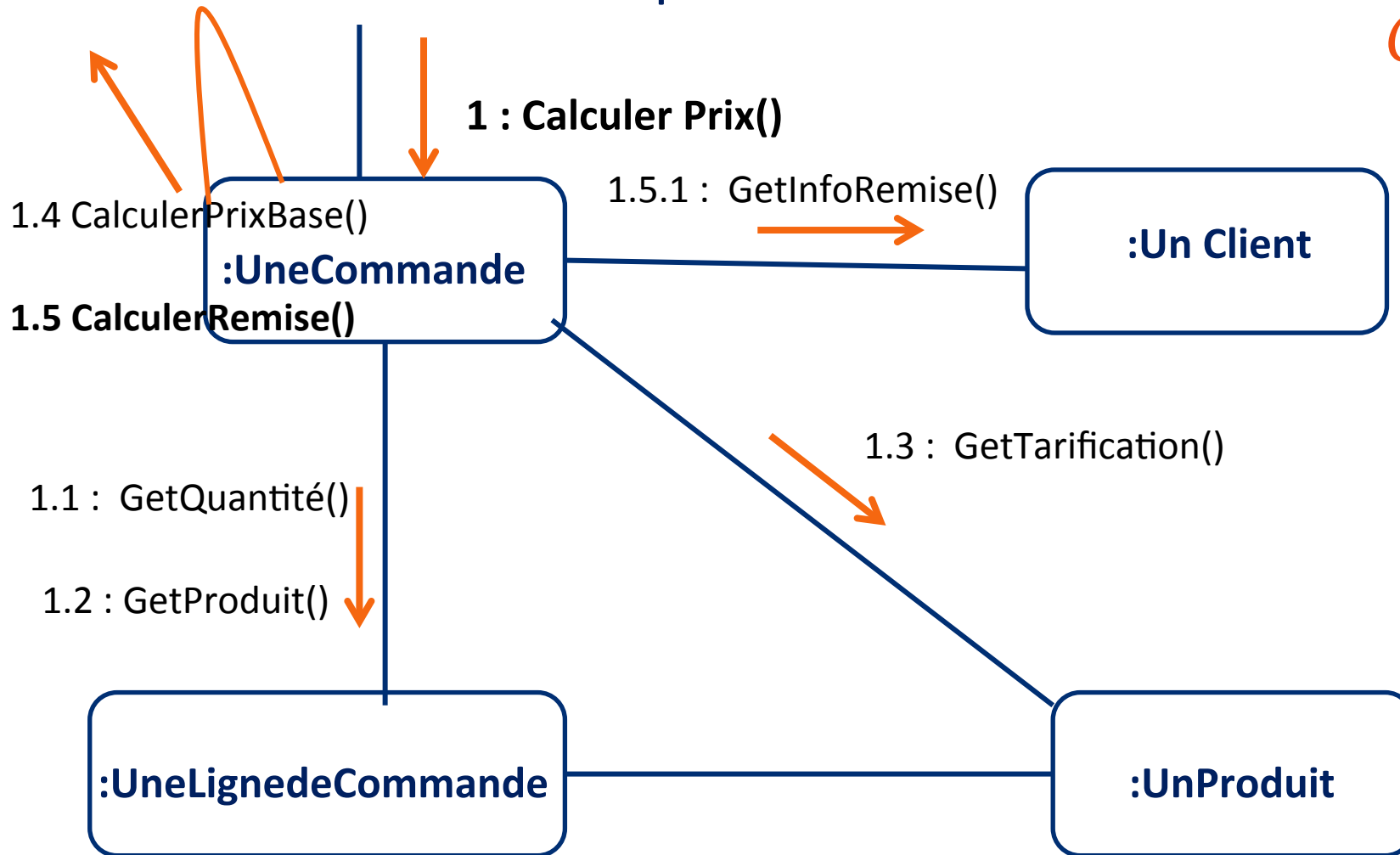
Diagramme de Communication Livraison Commande



Raison d'être : Résolution de l'ambiguïté liée aux auto-appels

Notation décimale imbriquée

Diagramme de Communication Concepts



1. Rôle : Participant à un échange correspondant à une ligne de vie dans le DS

2. Message : Appel à un appel d'opération effectuée par un rôle émetteur vers un Rôle récepteur

Diagramme de Communication

Concept : Rôle

1. Rôle :

- Chaque participant ayant un échange de message correspond à une ligne de vie du DSE
- Identifié par : **<nom de rôle> : type du message**

Exemple : **<nom de rôle> :UneCommande**, où l'un des deux est obligatoire avec le « : »

Le nom de rôle correspond au nom de l'objet et le nom du type à la classe

Diagramme de Communication

Concepts : Message

2. Message : Appel à une opération effectuée par un Rôle émetteur vers un Rôle récepteur

- Identifié par **<numéro> : nom ()**
- Syntaxe : [n° du message préc. reçu] . n° du message [itération] [condition] : nom du message

Exemple : **1.1.5.1a [si commande>10.000dhs] : CalculerRemise()**

Message précédent reçu
Indiquant la chronologie
du message

Message hiérarchique du message avec utilisation de lettre
Indiquant la simultanéité d'envoi de message

Condition indiquant message
à envoyer si commande>10000

Diagramme de Communication

Quand utiliser des diagrammes de communication ?

Ils sont à privilégier lorsque les liens entre participants doivent être mis en évidence.

De plus,

- ils sont plus faciles à modifier
- Ils constituent un bon moyen pour explorer des alternatives.

Les DSE sont par contre préférables lorsque il est besoin de mettre l'accent sur les séquences d'appels.

Diagramme de Communication

Exemple : création d'une agence

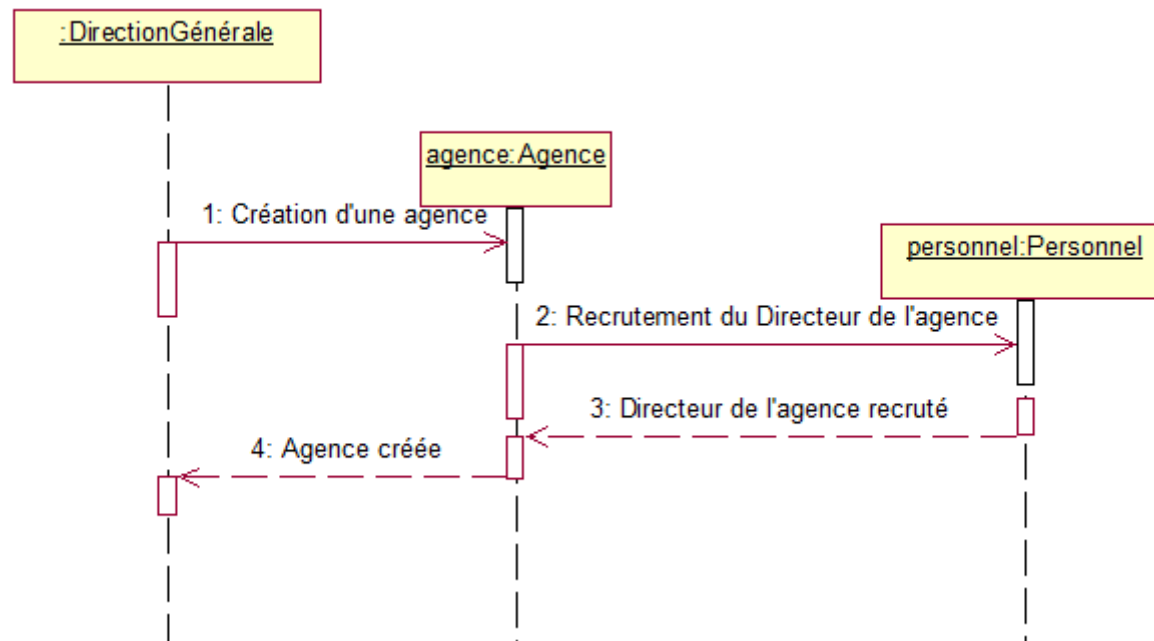


Diagramme de Communication

Exemple : création d'une agence

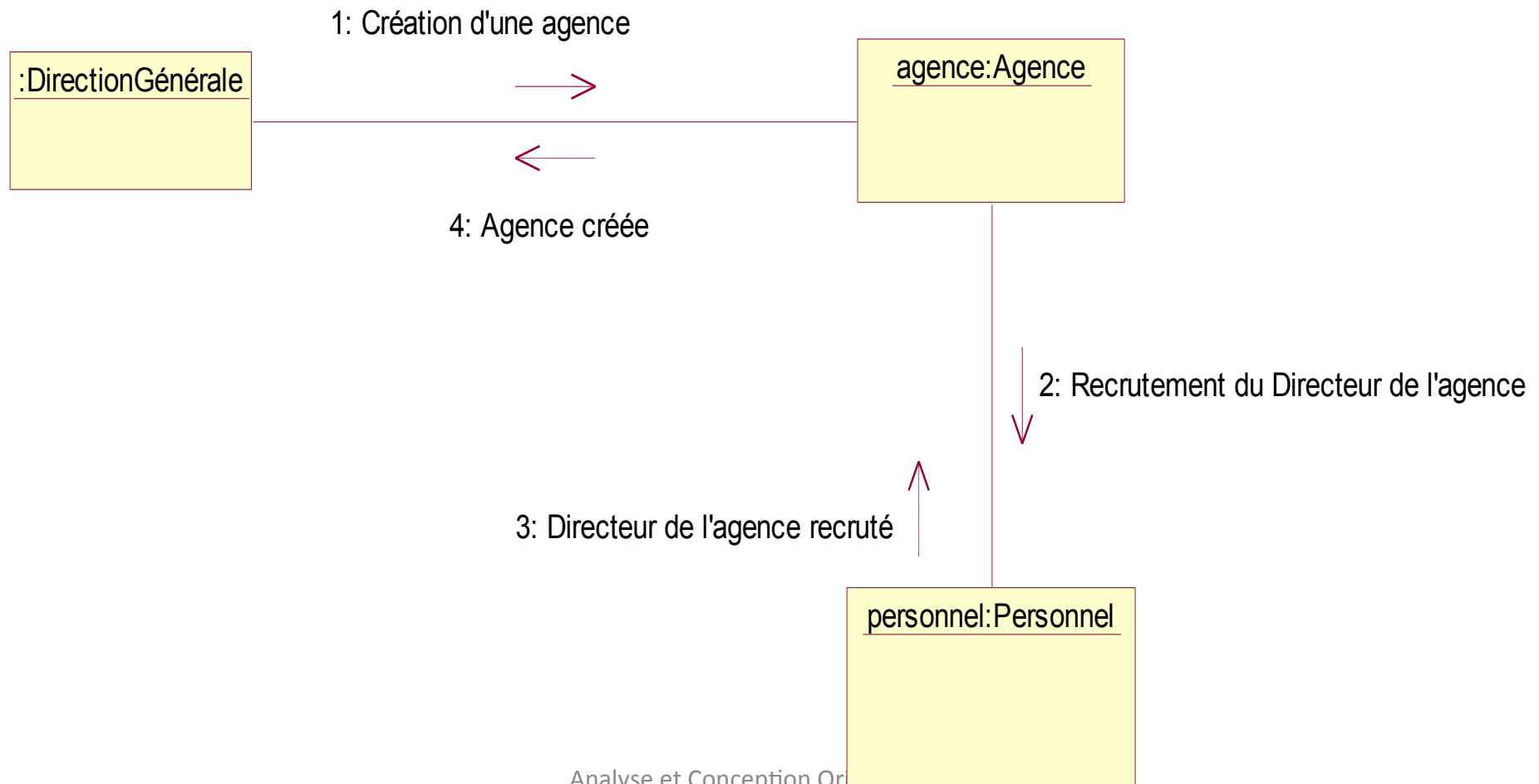


Diagramme de Communication

Exercice : Livraison Commande

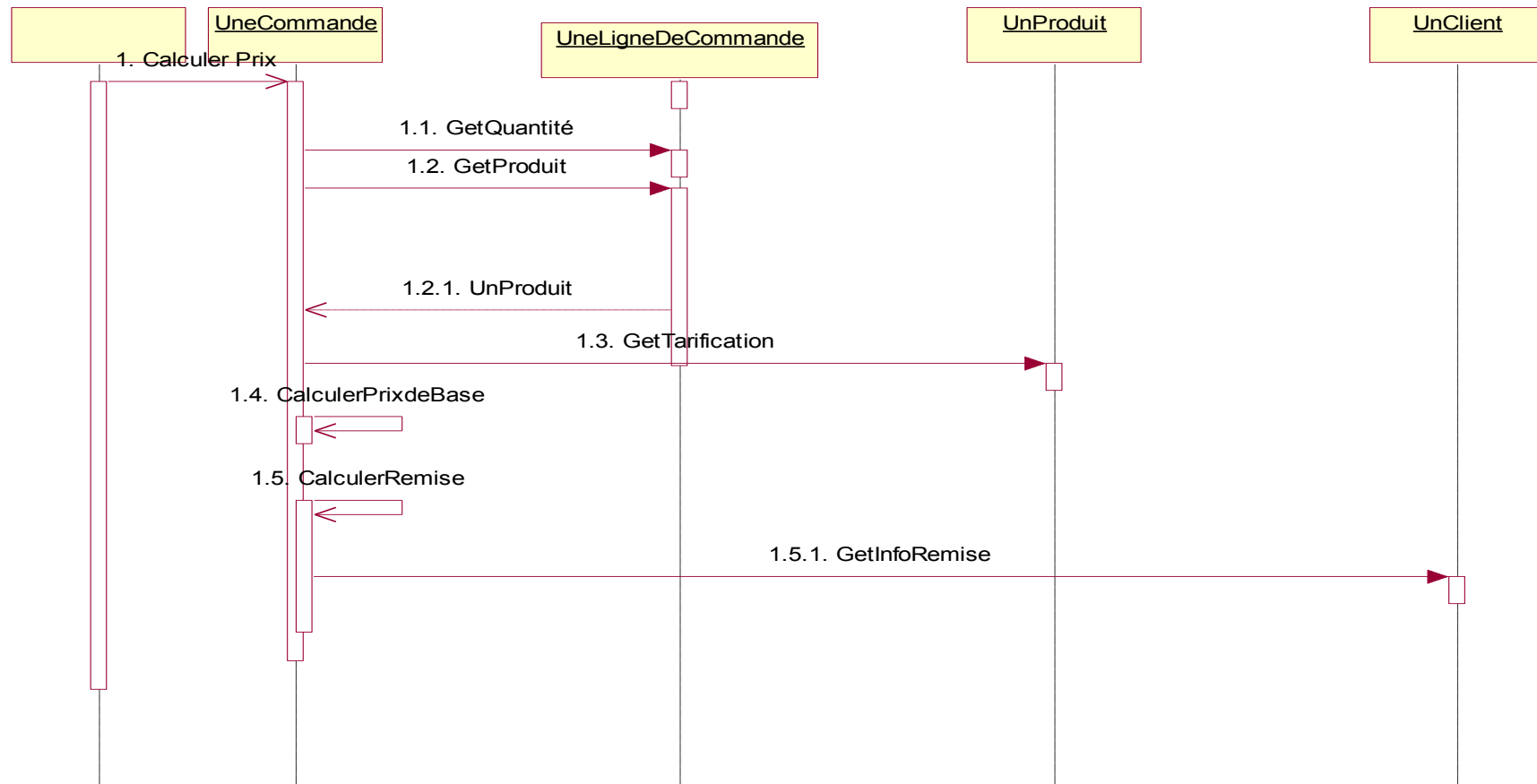
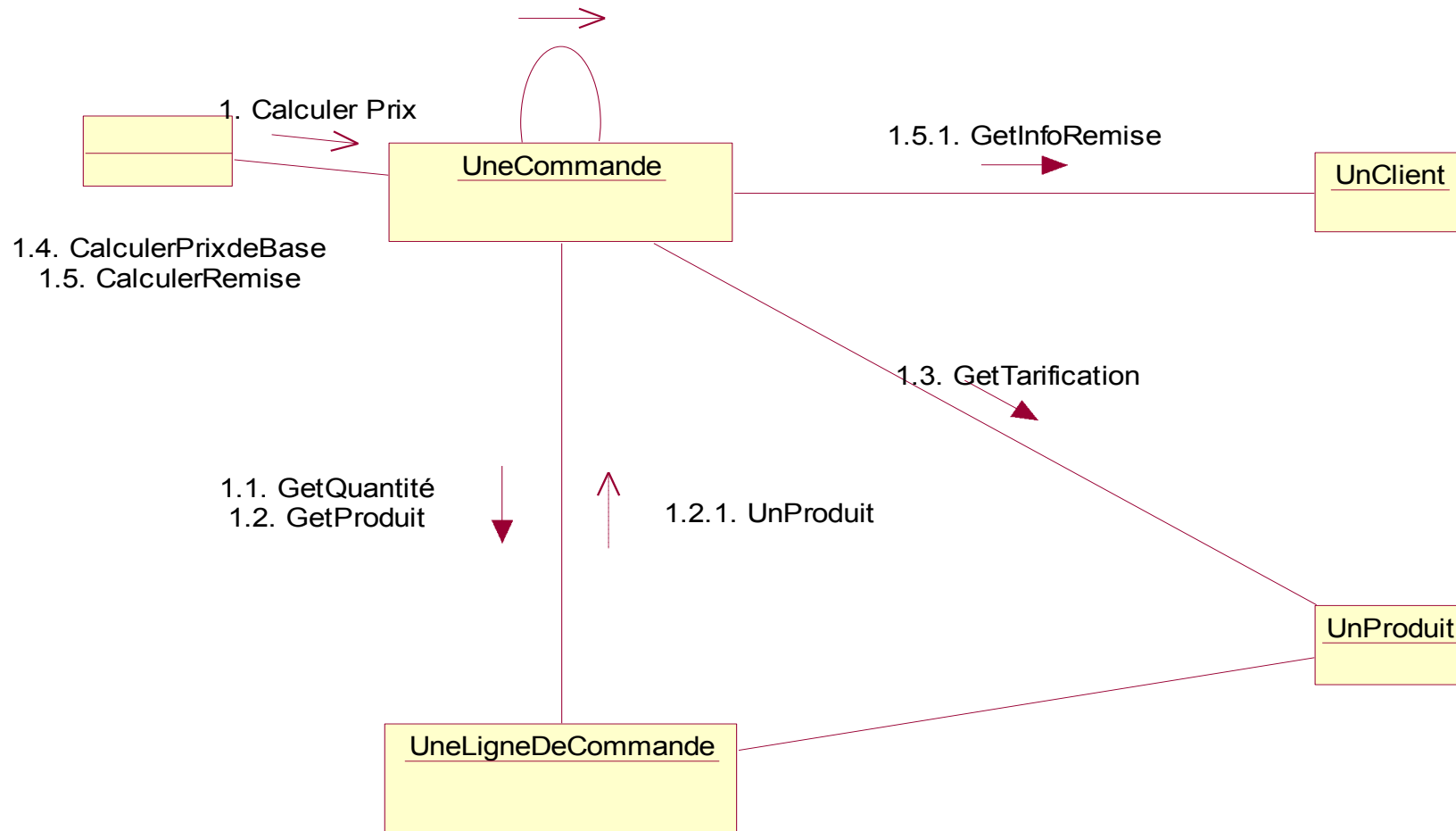


Diagramme de Communication

Exercice : livraison commande



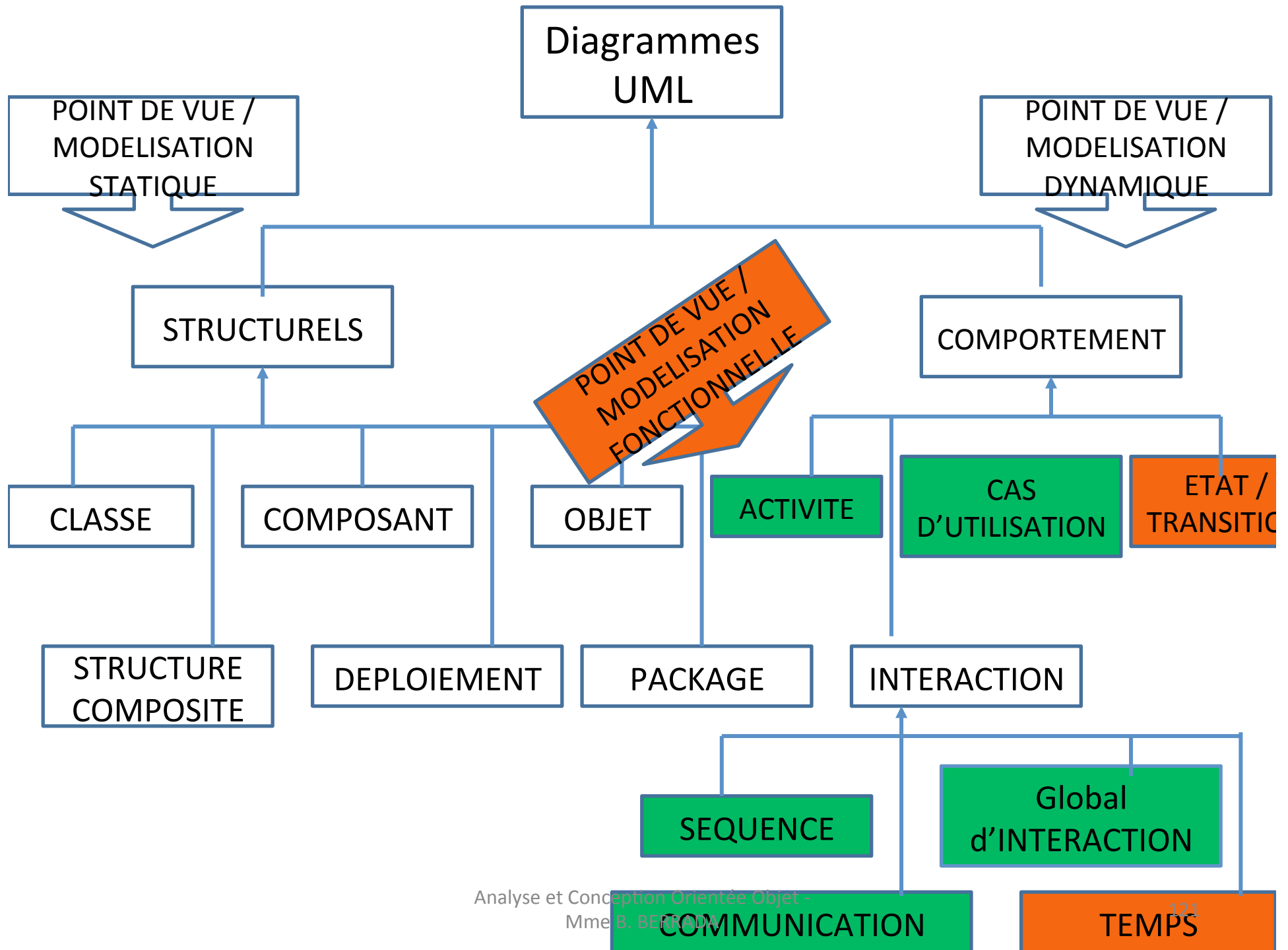


Diagramme de Temps (DTP)

Présentation générale :

Représente :

- Etats et interactions d'objets dans un contexte où le temps a une forte influence sur le comportement du système à gérer
- Changements d'états et des interactions entre objets liés à des contraintes de temps

Diagramme de Temps (DTP)

Concepts de base :

Trois concepts de base :

- **Ligne de vie** : Elle représente l'objet que l'on veut décrire. Elle se dessine de manière horizontale, et plusieurs lignes peuvent figurer dans un DTP
- **Etat ou ligne de temps conditionnée** : Différents états que peut prendre l'objet d'étude, listés en colonne et permettant de suivre le comportement de l'objet en ligne
- **Etats linéaires** : représentation de la succession des états est faite de manière linéaire à l'aide d'un graphisme particulier

Diagramme de Temps

Exemple 1

Représentation et exemples:

Soit à représenter le dispositif de chauffe d'un fer à repasser à vapeur au moment de sa mise en service selon les règles suivantes :

- La pompe à eau qui remplit la chambre de chauffe s'active dès que le témoin d'eau interne le demande ;
- La pompe à eau se désactive dès que le niveau d'eau nécessaire est atteint ;
- le chauffage de l'eau permettant de produire la vapeur, se met en action à la première mise en service du fer à repasser dès que le niveau d'eau de la chambre de chauffe est suffisant ;
- Le chauffage initial de l'eau dure 3 s permettant de produire la vapeur

Diagramme de Temps

Exemple de DT

Représentation en escalier ou sous forme de ligne:

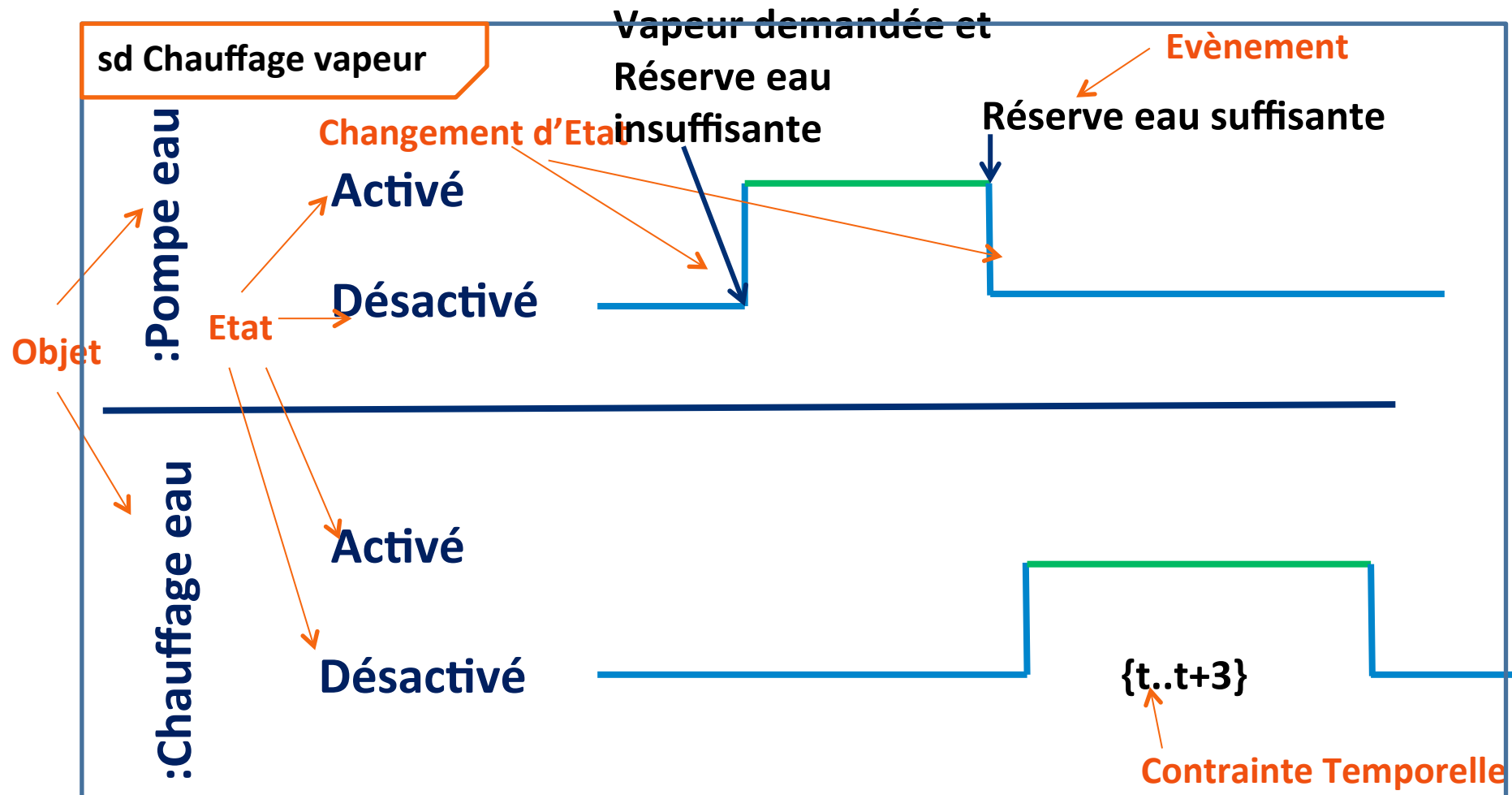


Diagramme de Temps

Exemple de DT

Représentation linéaire ou sous forme de zone:

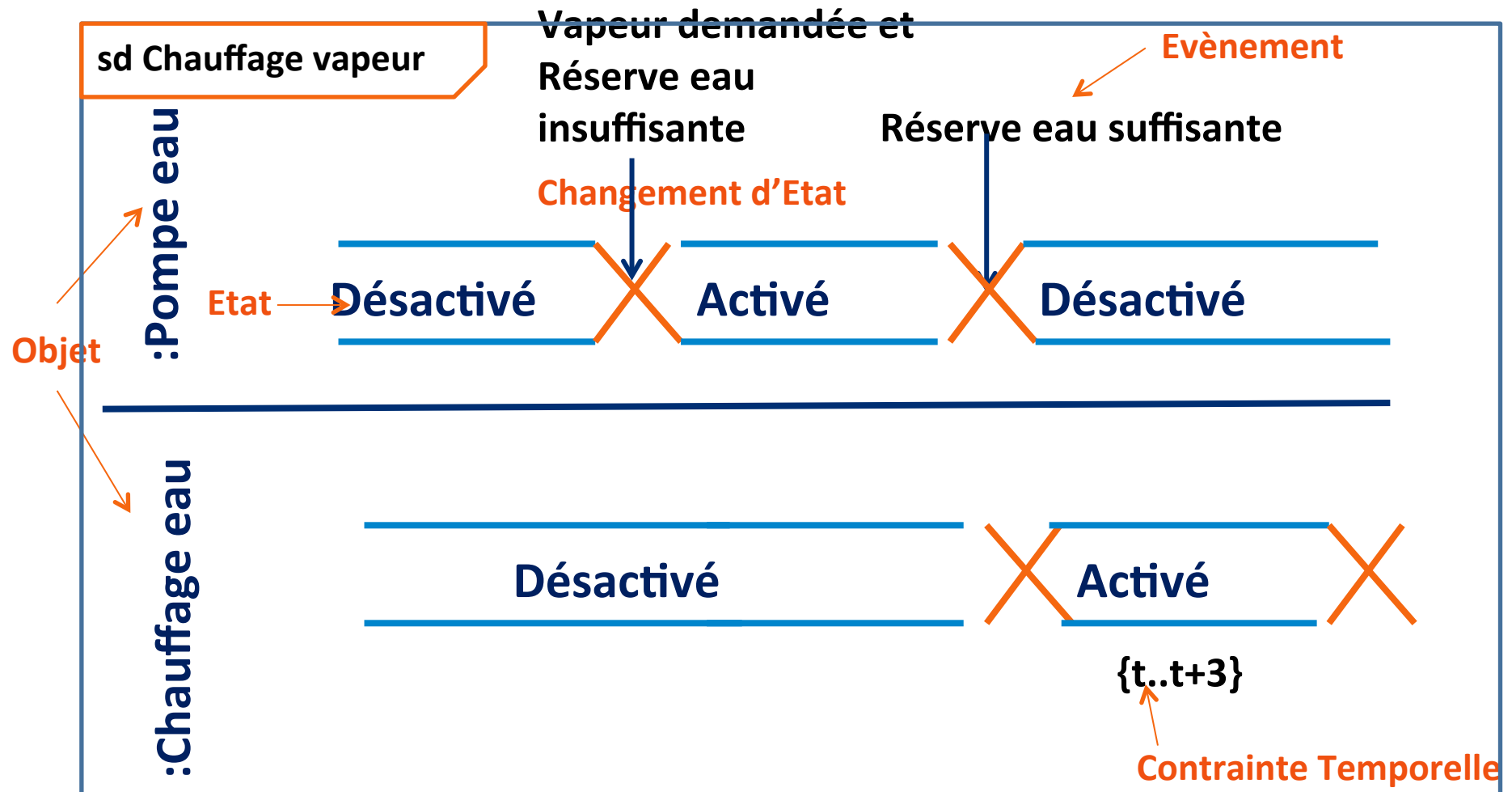


Diagramme de Temps

Exemple 2

Exemple 2:

Soit à représenter le dispositif d'une pompe et plaque chauffante d'une cafetière électrique.

Règles :

- Dix secondes au minimum doivent s'écouler entre l'activation de la pompe et celle de la plaque chauffante ;
- Quand le réservoir d'eau est vide, la pompe s'arrête mais la plaque peut continuer à fonctionner 15 minutes de plus.

Diagramme de Temps

Exemple de DT

Représentation en escalier ou sous forme de ligne:

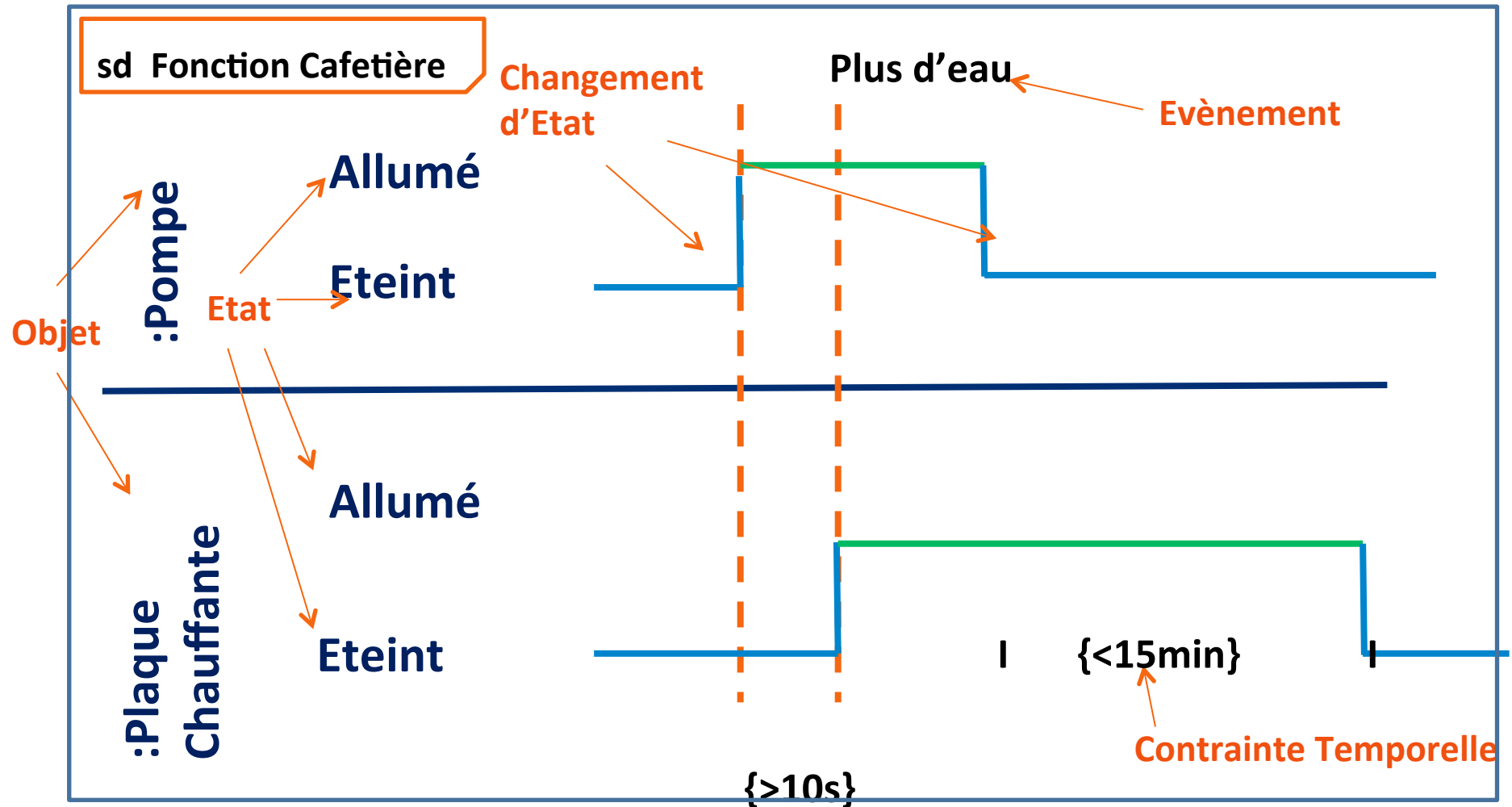
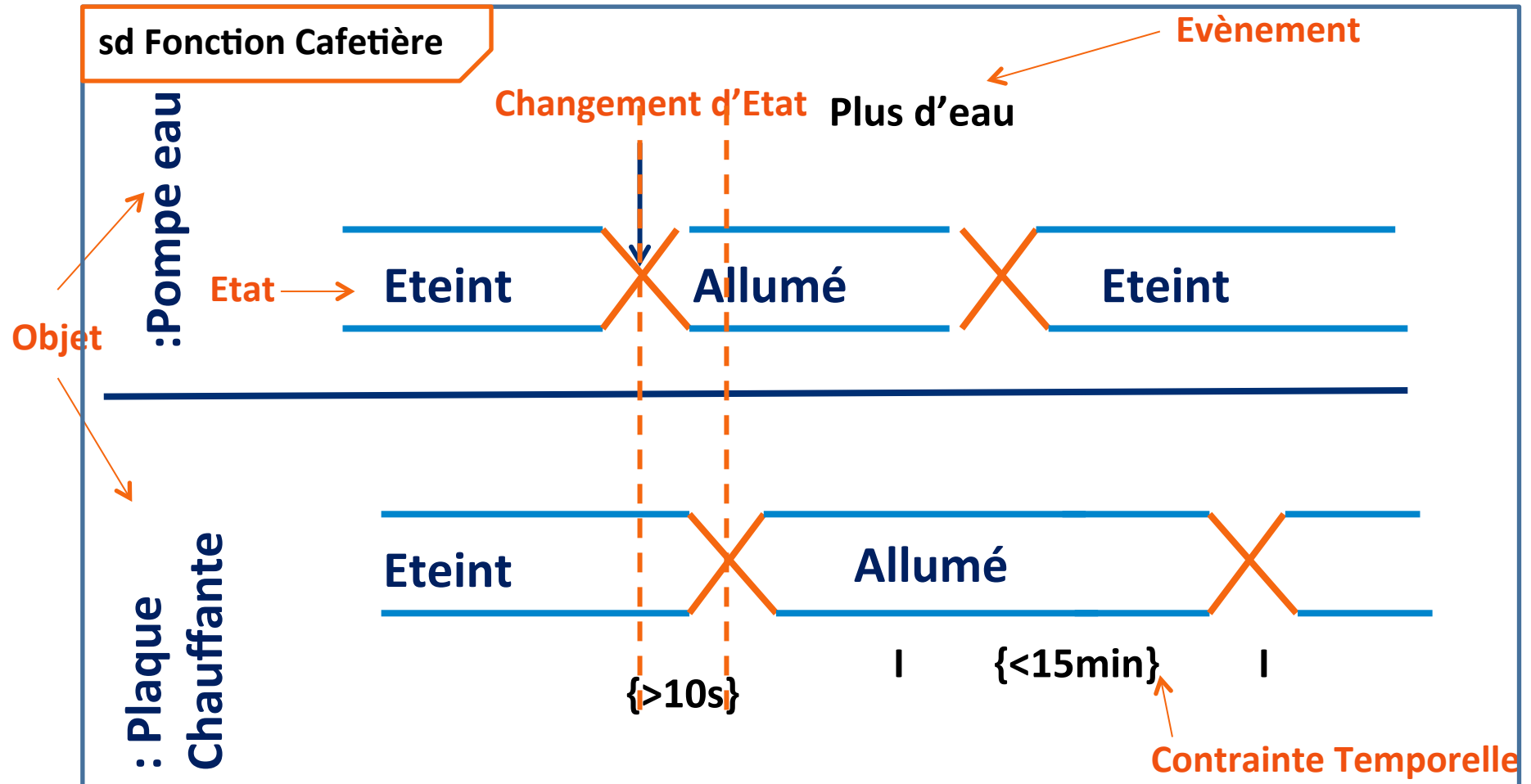


Diagramme de Temps

Exemple de DT

Représentation linéaire ou sous forme de zone:



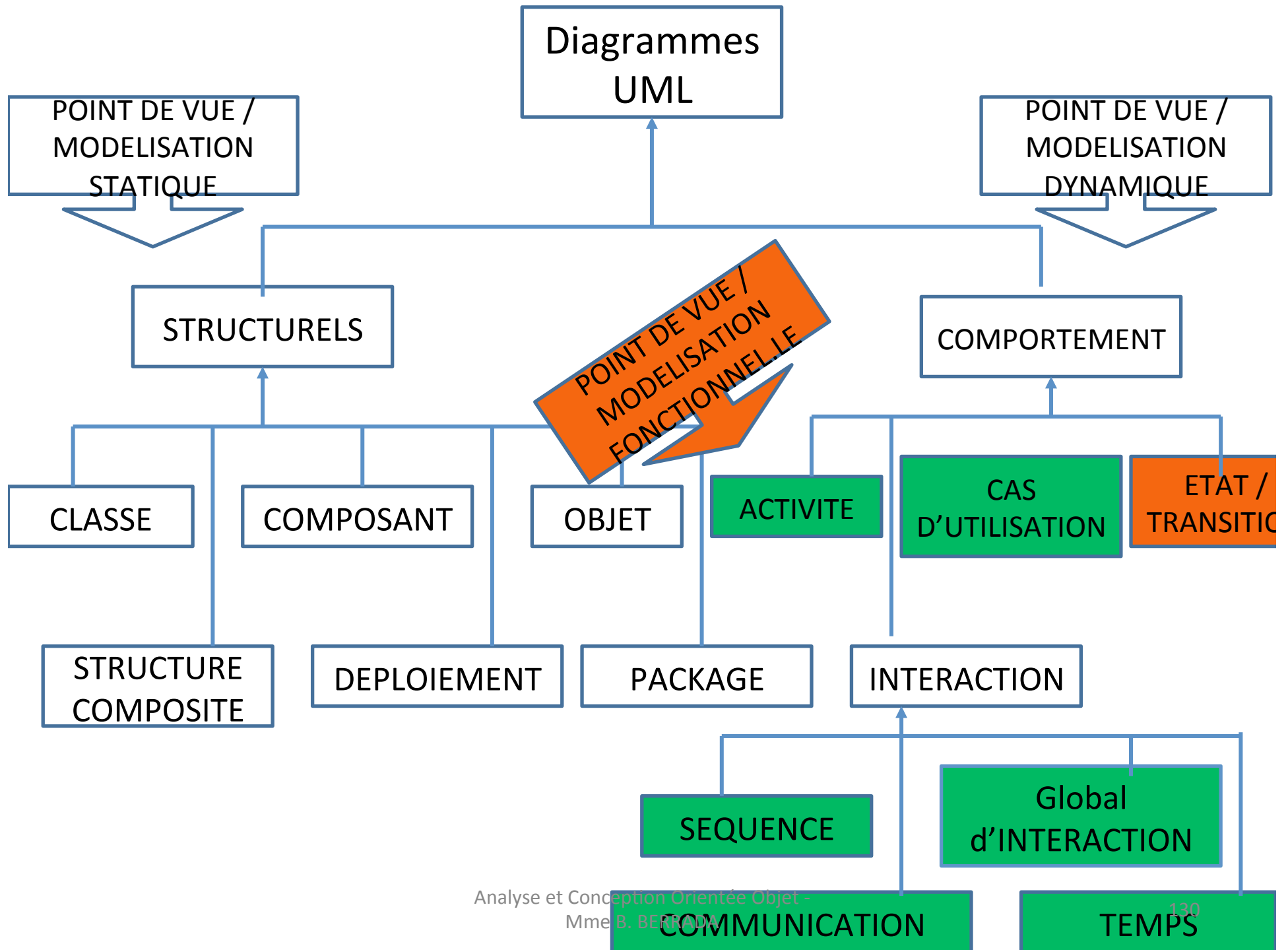


Diagramme d'Etat-Transition (DET)

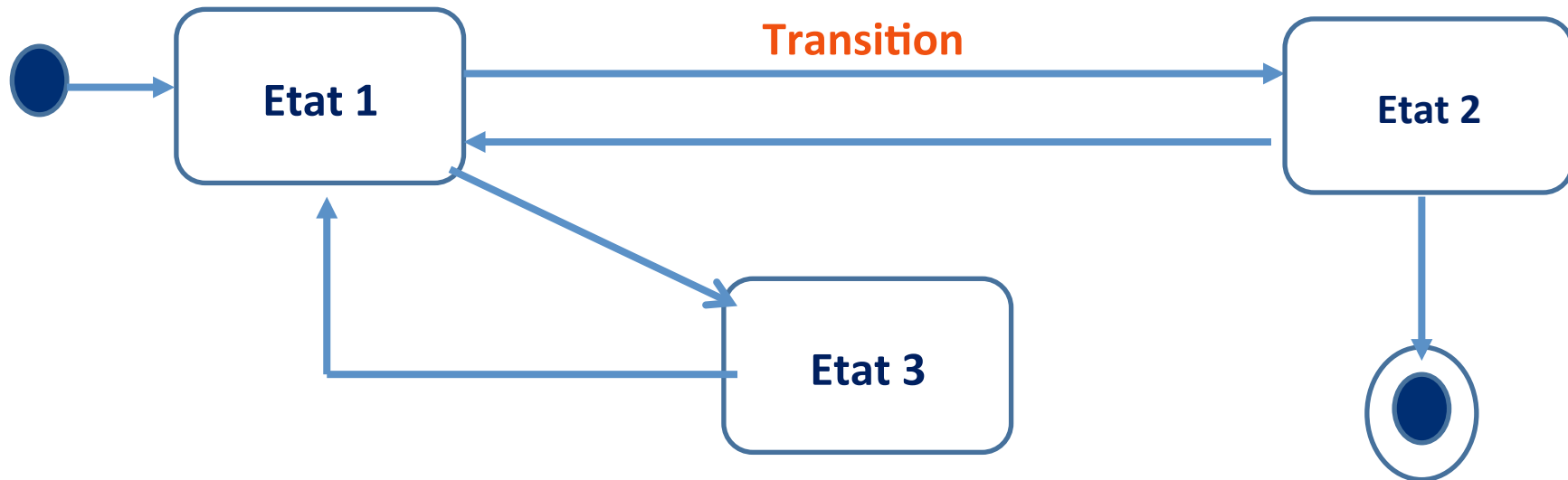
Présentation générale et concepts de base :

Les DET, depuis les années 60, constituent une technique courante permettant de :

- Décrire le comportement d'un système ;
- En particulier, pour mettre en évidence le comportement nominal **d'un seul objet** durant son cycle de vie au travers d'un enchaînement d'états caractéristiques de l'objet.

Diagramme d'Etat-Transition

Formalisme d'Etat/Transition :



➤ **Etat ? Une situation durant la vie d'un objet pendant laquelle :**

- ✓ Il attend un certain événement
- ✓ Il satisfait une certaine condition
- ✓ Il exécute une certaine activité

➤ **Un objet passe par une succession d'états durant son existence**

➤ **Un état a une durée finie**

➤ **Le DET comporte un Etat initial (création de l'instance) et un Etat final (destruction de l'instance)**

Diagramme d'Etat-Transition

Formalisme d'Etat/Transition :



- La transition décrit la réaction d'un objet lorsqu'un évènement se produit
- Une transition possède :
 - ✓ Un évènement déclencheur (réception de message, appel opération, passage du temps, chgt dans la satisfaction d'une condition)
 - ✓ Une condition de garde
 - ✓ Un effet (action ou activité)
 - ✓ Un Etat cible

Diagramme d'Etat-Transition

Exemple 1

Gestion commerciale de l'objet *Client* :

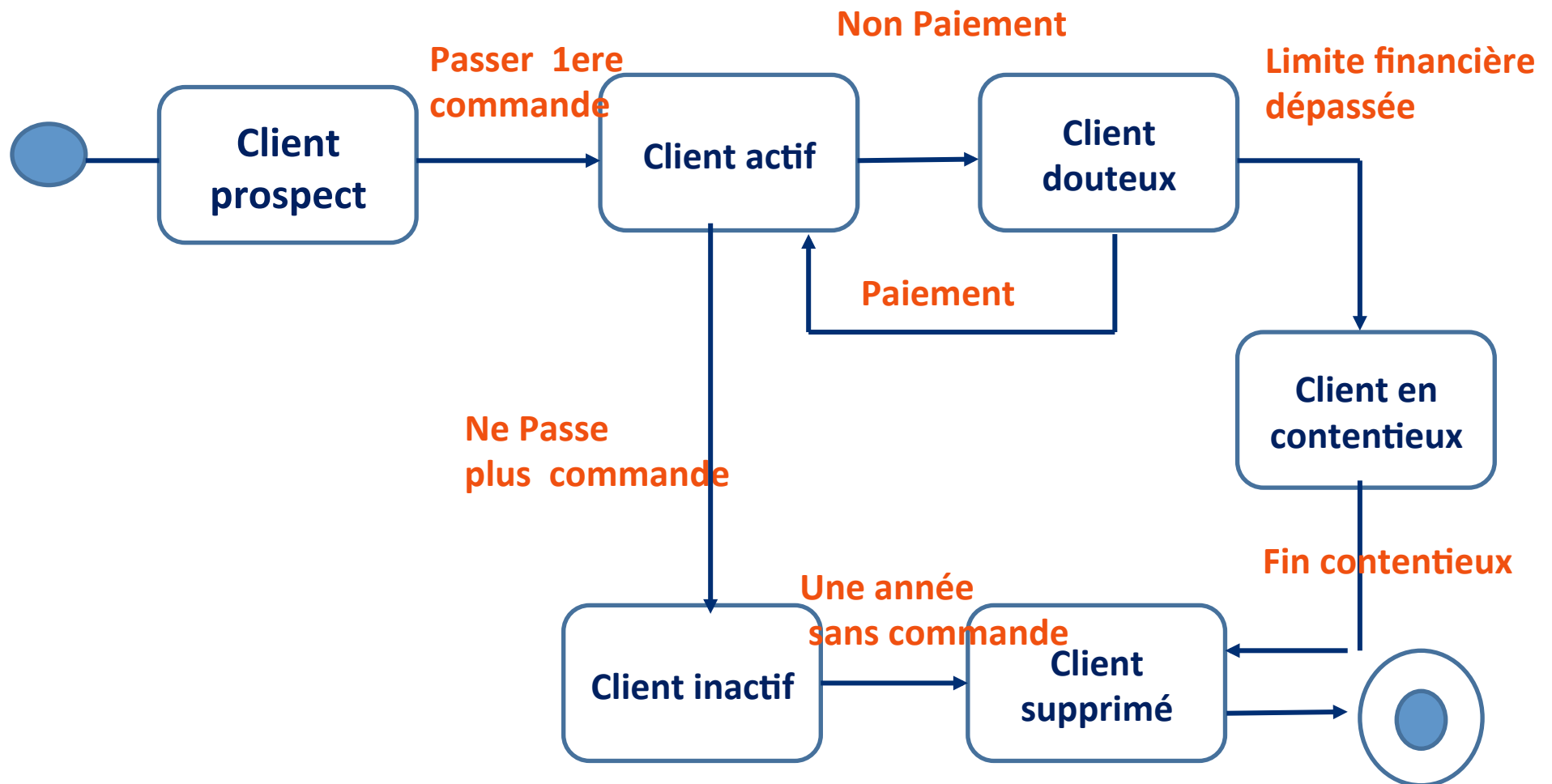


Diagramme d'Etat-Transition

Exemple 2

Ouverture/Fermeture d'un Coffre dans un château médiéval :

Dans ce château, les trésors sont conservés dans un coffre difficile à trouver.

Pour découvrir la serrure du coffre :

- Ôter une chandelle stratégique de son chandelier ;
- Pour révéler la serrure, la porte doit être fermée ;
- Dès que la serrure est visible, insérer la clef ;
- Le coffre n'est ouvert que si la chandelle est remise à sa place.

Diagramme d'Etat-Transition

Exemple

Ouverture/Fermeture d'un Coffre dans un château médiéval :

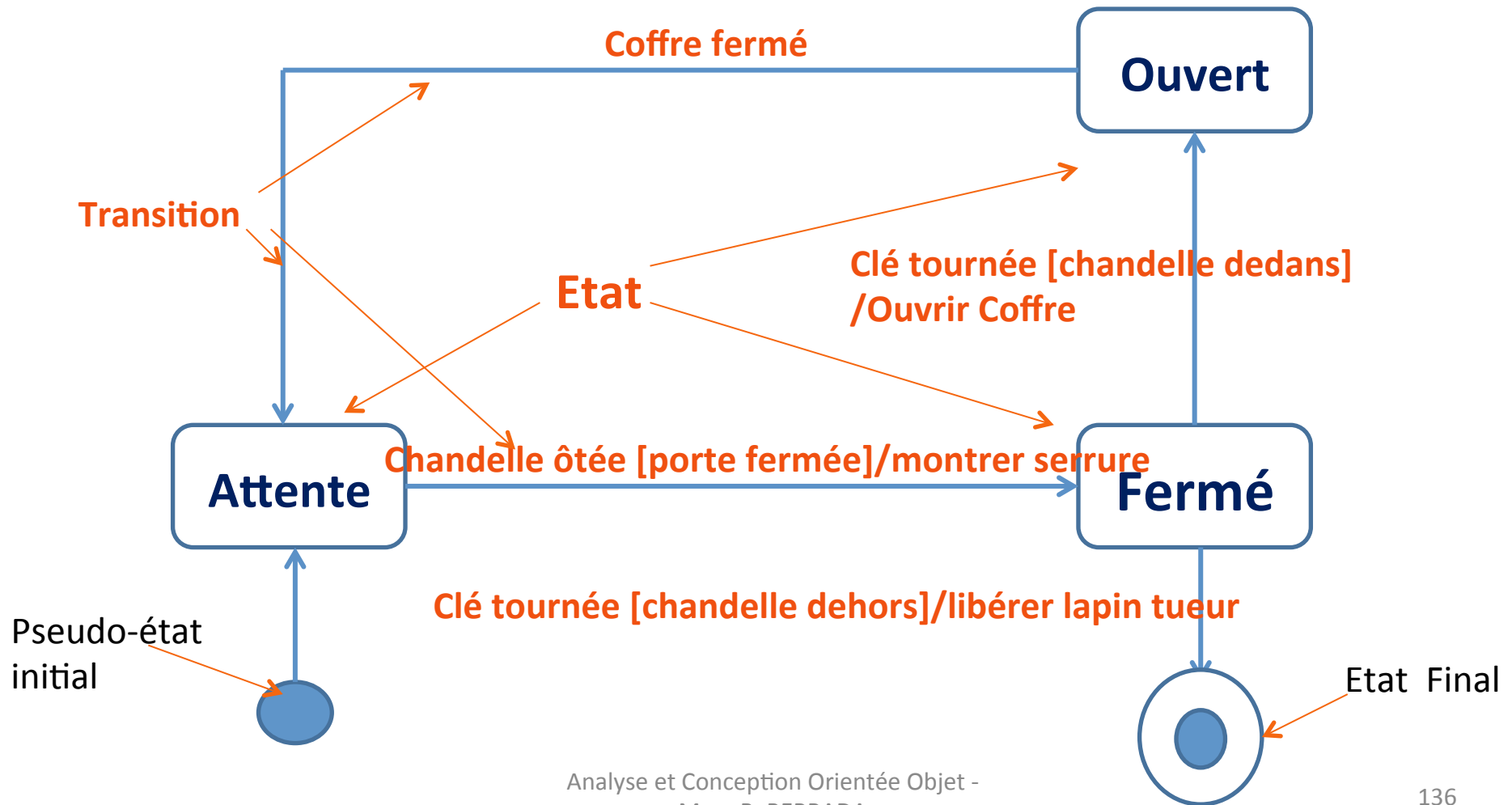


Diagramme d'Etat-Transition

- L'enchaînement de tous les états caractéristiques d'un objet constitue le diagramme d'état
- Un diagramme d'état débute toujours par un état initial. Et se termine par un ou plusieurs états finaux
- Sauf dans le cas où le diagramme d'états représente une boucle.

Diagramme d'Etat-Transition

Compléments

- **Composition et décomposition**
- **Point d'entrée et de sortie**
- **Point de jonction**
- **Point de choix**
- **Etat historique**
- **Etat d'activité**

Diagramme d'Etat-Transition

Composition et décomposition d'Etat

Il est possible de décrire un DET à plusieurs niveaux.

- **Premier niveau** : Etats élémentaires + Etats composites
- **Second niveau** : Description des Etats composites dans un autre diagramme

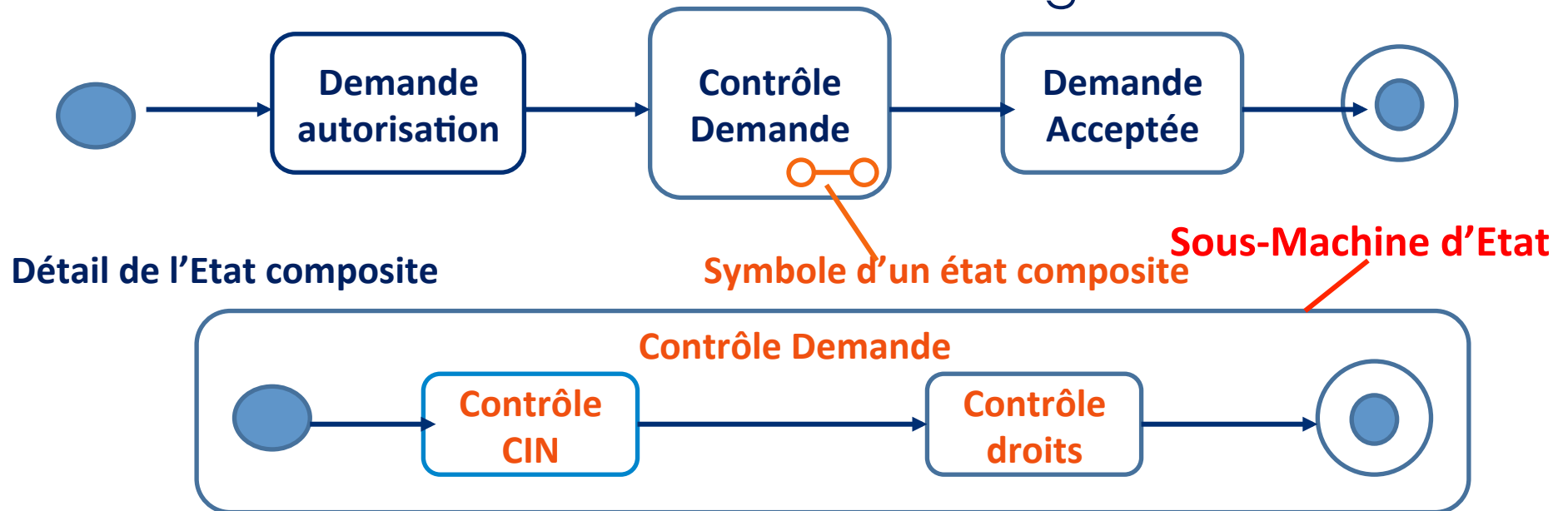


Diagramme d'Etat-Transition

Point d'Entrée et de Sortie

Sur une sous-machine d'Etat, il est possible de repérer un **point d'entrée** et un **point de sortie particuliers**.

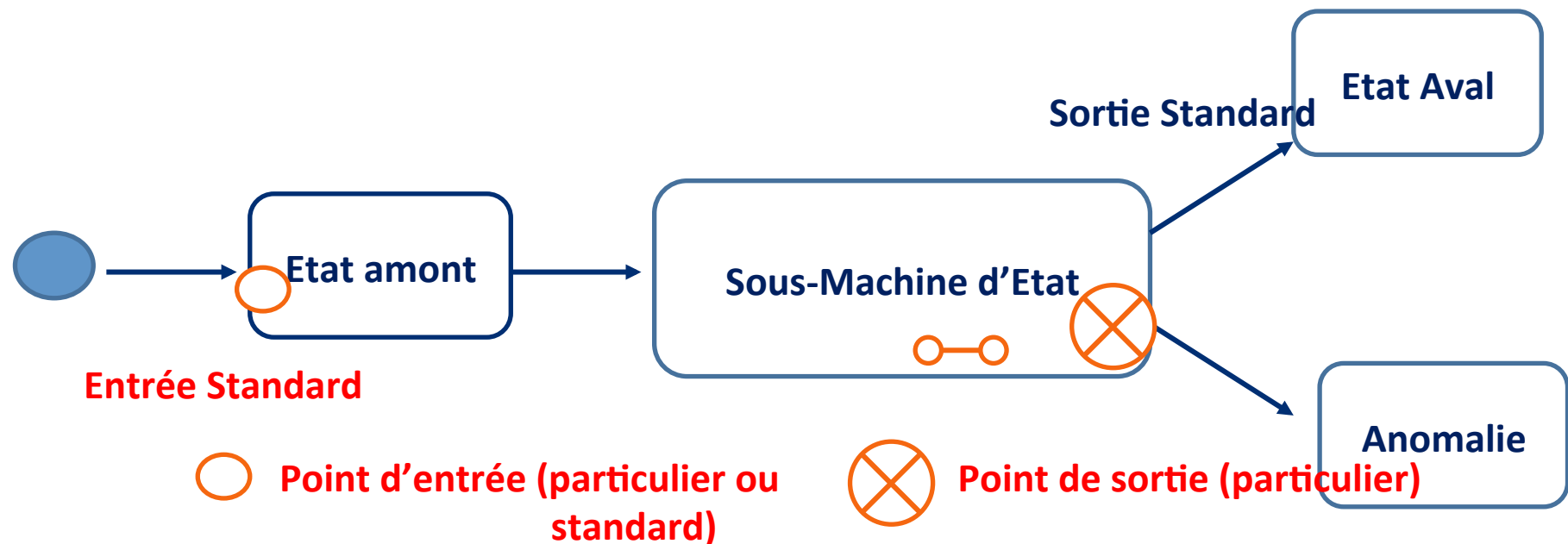
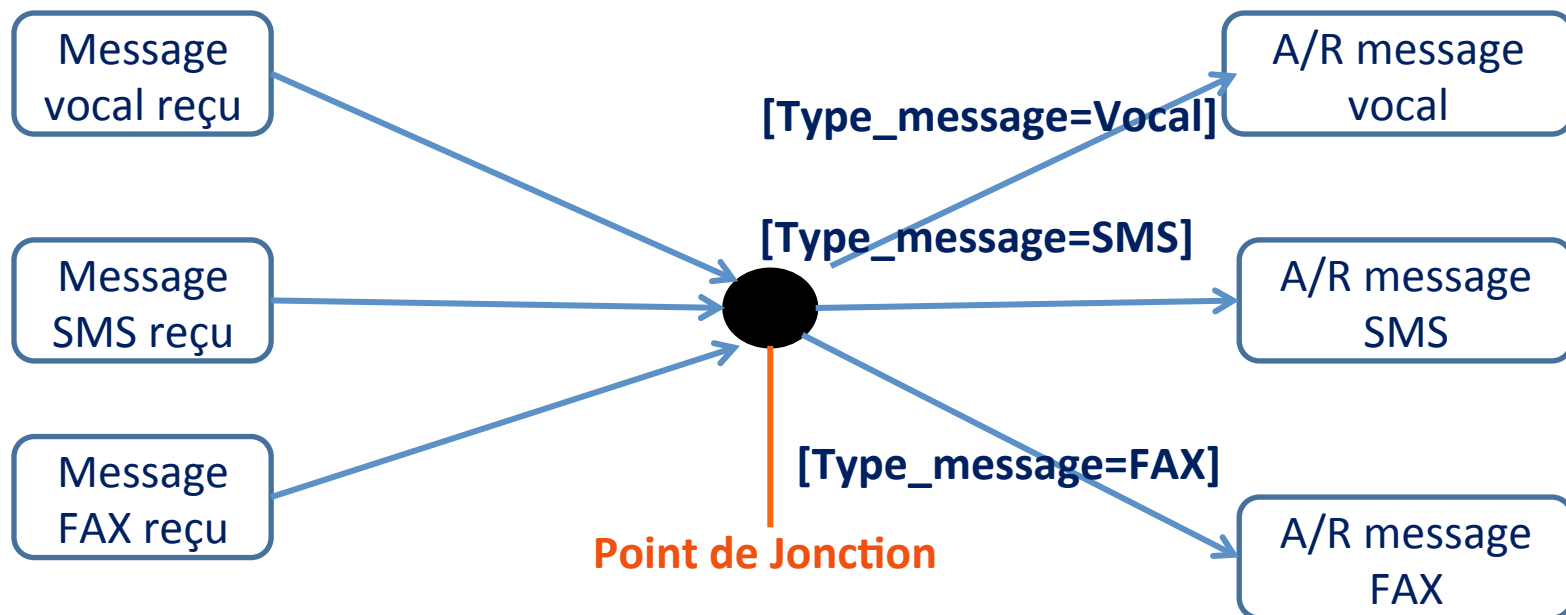


Diagramme d'Etat-Transition

Point de Jonction

Lorsque l'on veut relier plusieurs états vers d'autres, le point de jonction permet de décomposer une transition en deux parties, en indiquant si nécessaire, les gardes propres à chaque segment de la transition.



A l'exécution, **un seul parcours est emprunté**, c'est celui pour lequel toutes les conditions sont satisfaites.

Diagramme d'Etat-Transition

Point de Choix

Le point de choix se comporte comme un test de type :
si condition faire action 1 sinon faire action 2

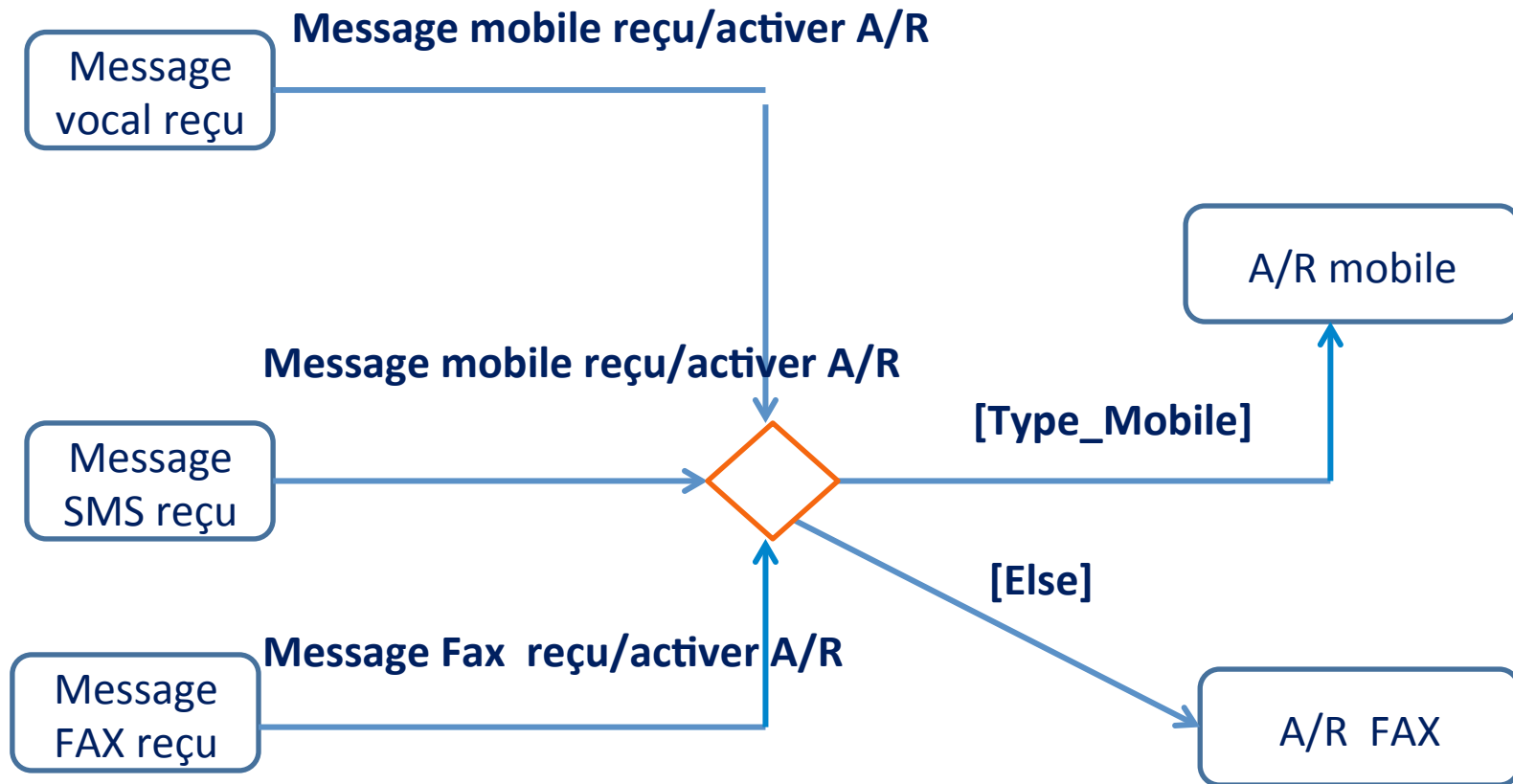


Diagramme d'Etat-Transition

Etat Historique

La mention de l'historisation d'un état composite permet de pouvoir indiquer **la réutilisation du dernier état historisé en cas de besoin.**

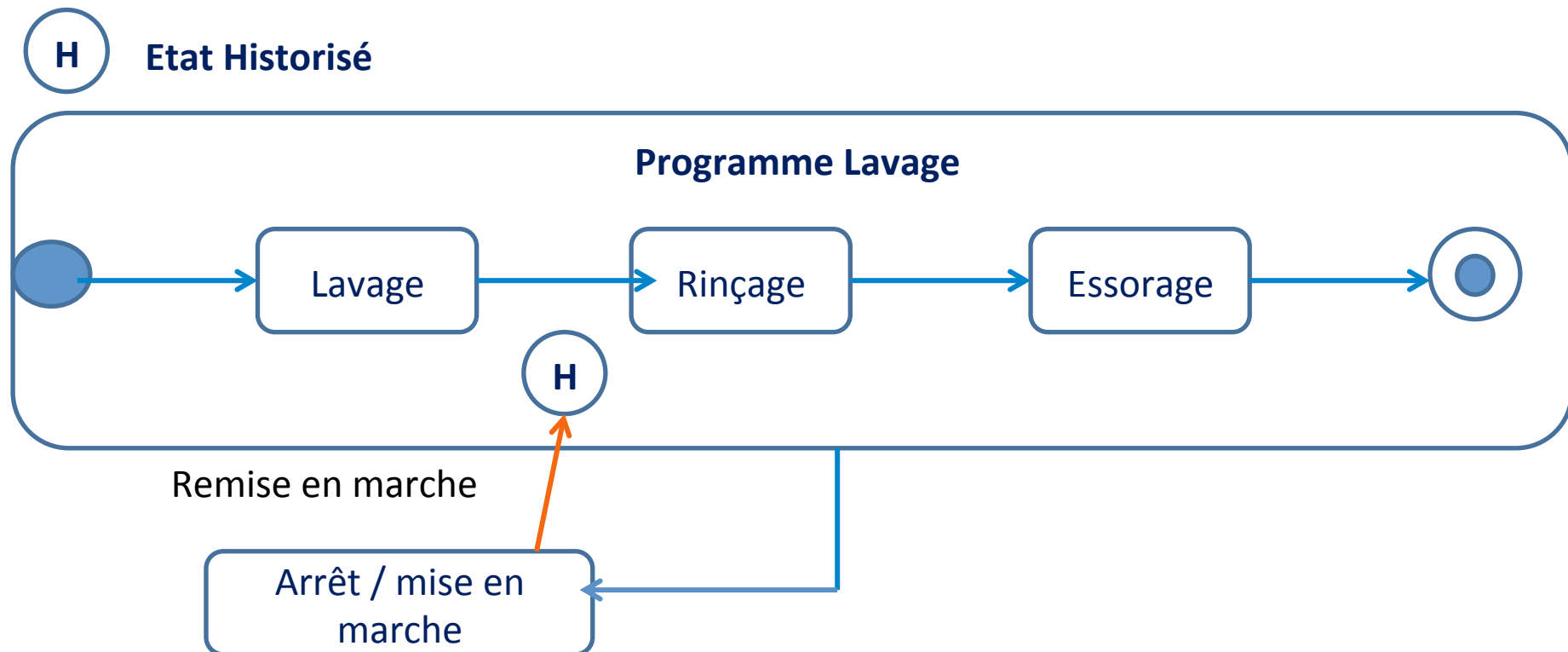


Diagramme d'Etat-Transition

Etat d'activité

Dans les états décrits jusqu'à présent, l'objet est inactif et attend l'évènement suivant avant de faire qqchose.

Il est possible de rencontrer des états dans lesquels l'objet exécute une activité durable.

Une activité durable à l'intérieur d'un état peut être soit :

- **Continue** : elle ne cesse que lorsque se produit un évènement qui fait sortir l'objet de l'état
- **Finie / automatique**: elle peut être interrompue par un évènement mais elle cesse d'elle-même au bout d'un certain temps

Diagramme d'Etat-Transition

Etat d'activité

Considérons un réveil du matin simplifié :

- Phrase 1 : On peut mettre l'alarme « on » ou « off »;
- Phrase 2 : Quant l'heure courante devient l'heure de l'alarme, le réveil sonne sans s'arrêter ;
- Phrase 3: On peut interrompre la sonnerie.

Diagramme d'Etat-Transition

Etat d'activité

Considérons un réveil du matin simplifié :

➤ Phrase 1 : On peut mettre l'alarme « on » ou « off »

Le réveil a 2 Etats distincts : Activé et Désactivé

Une simple action de l'utilisateur permet de passer d'un Etat à l'autre :

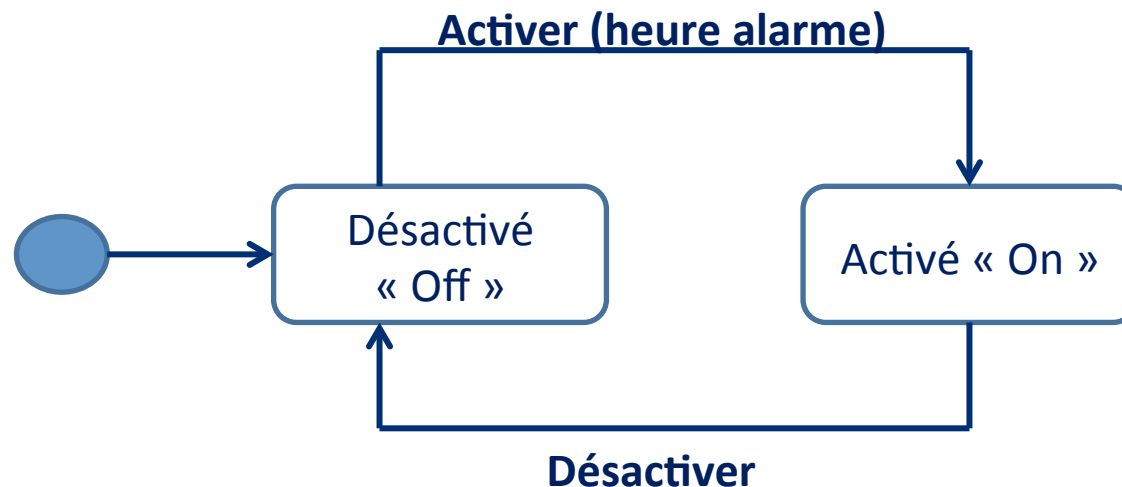


Diagramme d'Etat-Transition

Etat d'activité

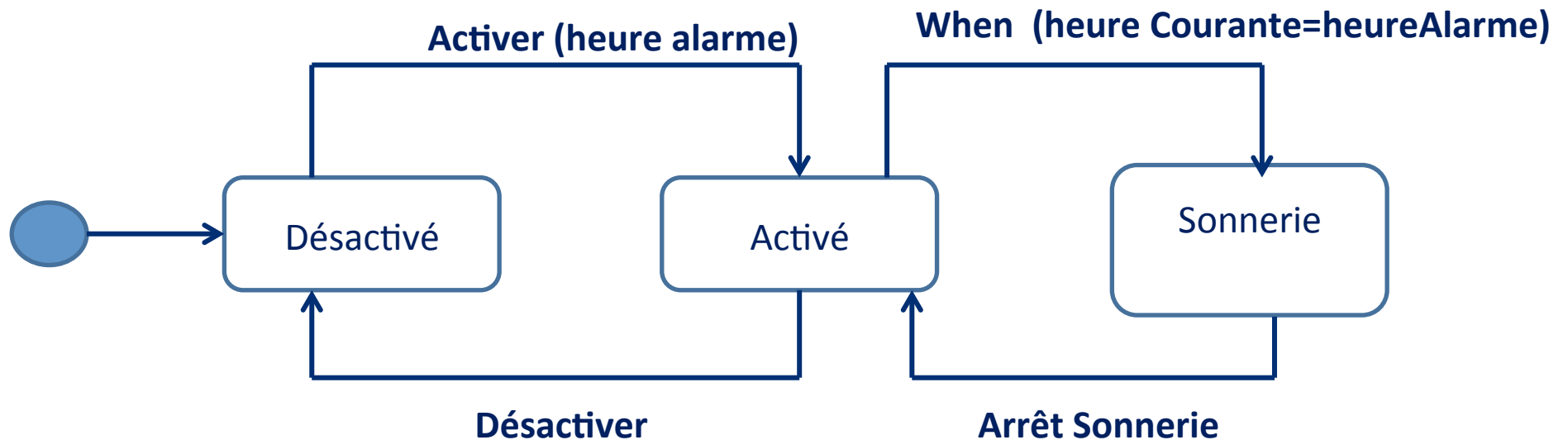
Considérons les deux autres phrases :

- Phrase 2 : Quant l'heure courante devient l'heure de l'alarme, le réveil sonne sans s'arrêter ;
- Phrase 3: On peut interrompre la sonnerie.

Le fait de sonner constitue un nouvel état pour le réveil. Il s'agit bien d'une période de temps durant laquelle le réveil effectue une activité (sonner) qui dure jusqu'à ce qu'un évènement vienne l'interrompre.

Diagramme d'Etat-Transition

Le passage de l'Etat Activé à l'Etat Sonnerie est déclenché par une transition « when » dite interne *Etat d'activité*



En revanche, le retour de l'Etat sonnerie à l'Etat Activé s'effectue sur un évènement utilisateur.

Diagramme d'Etat-Transition

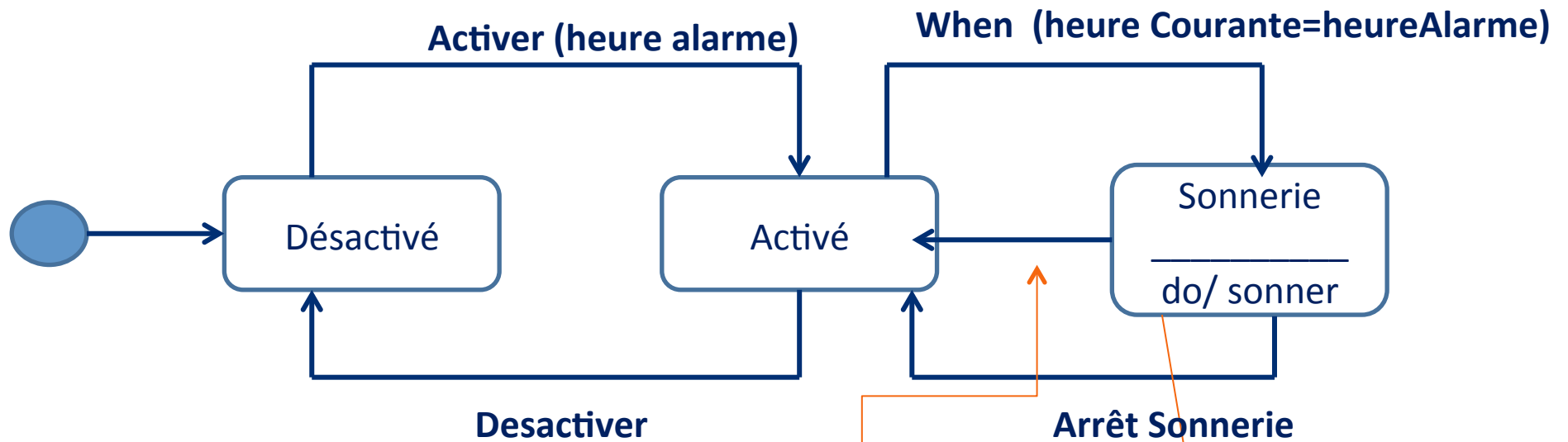
Etat d'activité

- Il y a une autre possibilité de sortie de l'Etat de sonnerie : quand le réveil arrête de sonner par lui-même, au bout d'un certain temps.
- Il suffit donc d'ajouter une activité durable Sonner à l'Etat sonnerie et une transition automatique en sortie de cet Etat.
- Le diagramme d'Etat complété est représenté dans le schéma suivant :

Diagramme d'Etat-Transition

Etat d'activité

Exemple d'un réveil matin



Transition automatique
En sortie de l'Etat signifiant
Que le réveil s'arrête tout seul
de sonner

Ajout de l'Activité durable
À l'état Sonnerie