

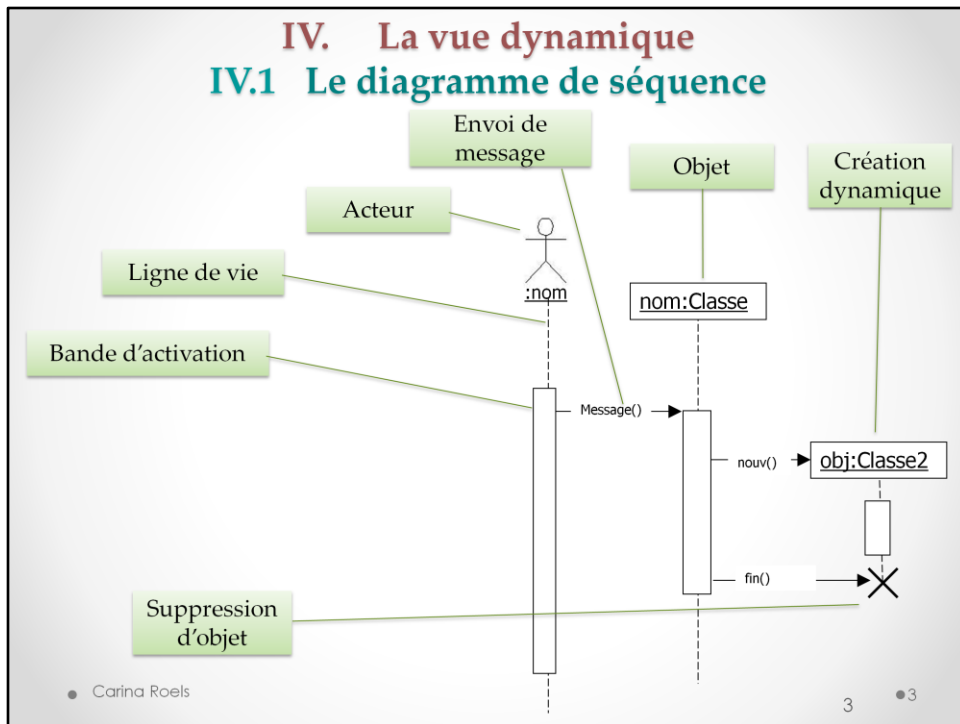
UML 2

- 4. La modélisation dynamique
 - Le diagramme de séquence
 - Le diagramme d'activité
 - Le diagramme d'état-transition
 - Le diagramme de collaboration

IV. Les diagrammes d'interaction

- Représentent une **interaction**.
Autrement dit, un ensemble d'objets et leurs **relations**, y compris les messages qu'ils peuvent échanger
- Représentent une vue **dynamique** du système
- 2 types de diagrammes d'interaction :
 - **Diagrammes de séquence** : mettent l'accent sur la **chronologique** des messages de collaboration d'instance
 - **Diagrammes de collaboration** : mettent l'accent sur l'**organisation** structurelle des éléments qui envoient et reçoivent des messages

Les diagrammes de séquences peuvent servir à illustrer un **cas d'utilisation**.



Un objet dispose d'une ligne de vie, représentée par une ligne verticale en pointillés.
 Un objet peut être créé dynamiquement.
 Dans ce cas, l'objet figure à l'endroit où la création a lieu (message de création reçu).

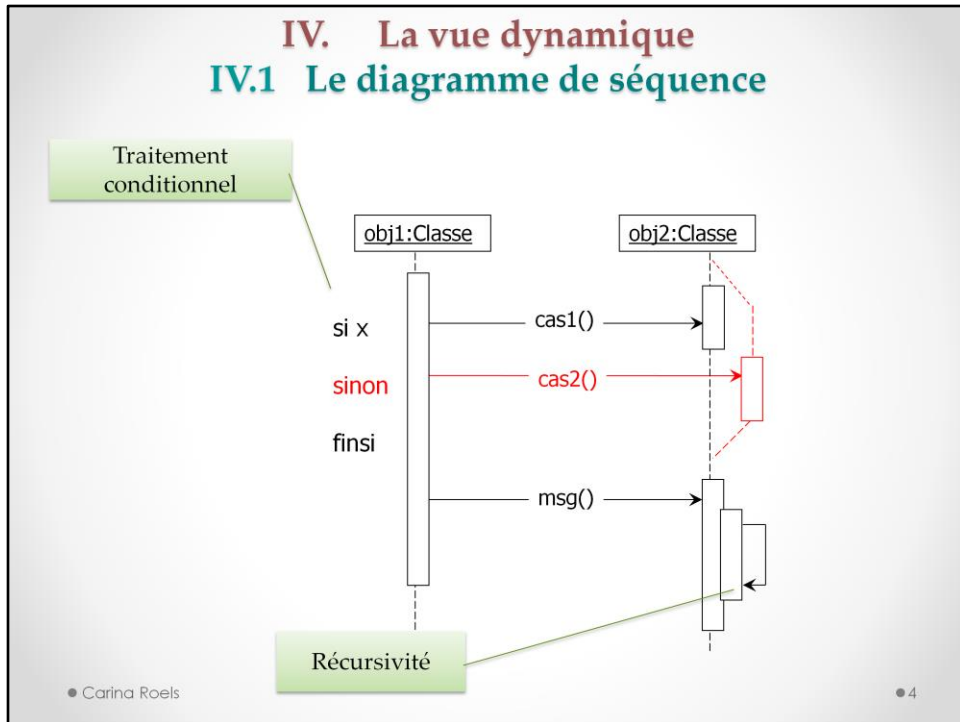
Les différentes périodes d'activité d'un objet sont mises en évidence grâce à une bande rectangulaire superposée à la ligne de vie de l'objet.

Période d'activité n'est pas l'équivalent de durée de vie!

Un objet est détruit uniquement lorsque précisé par une croix sur la ligne de vie.

IV. La vue dynamique

IV.1 Le diagramme de séquence



Comme montré dans l'exemple de diagramme, un objet peut être activé à de multiples reprises.

L'objet 1 a une bande d'activation continue.

L'objet 2 a 2 bandes activation au cours de sa vie :

- Le premier, qui est représenté par 2 bandes d'activation alternatifs (`cas1` et `cas2`)
- De deuxième, qui démontre une récursivité.

IV. La vue dynamique

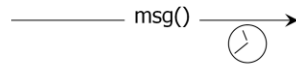
IV.1 Le diagramme de séquence

Types de messages

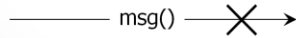
- **Message simple** : aucune caractéristique d'envoi ou de réception particulière



- **Message minuté** (timeout) : bloque l'expéditeur pendant un temps donné, en attendant la prise en compte du message par le récepteur. L'expéditeur est libéré si la prise en compte n'a pas eu lieu pendant le délai spécifié



- **Message synchrone** : bloque l'expéditeur jusqu'à prise en compte du message par le destinataire



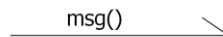
IV. La vue dynamique

IV.1 Le diagramme de séquence

Types de messages

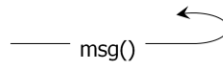
- **Message asynchrone** : n'interrompt pas l'exécution de l'expéditeur. Le message peut être pris en compte par le récepteur à tout moment ou ignoré

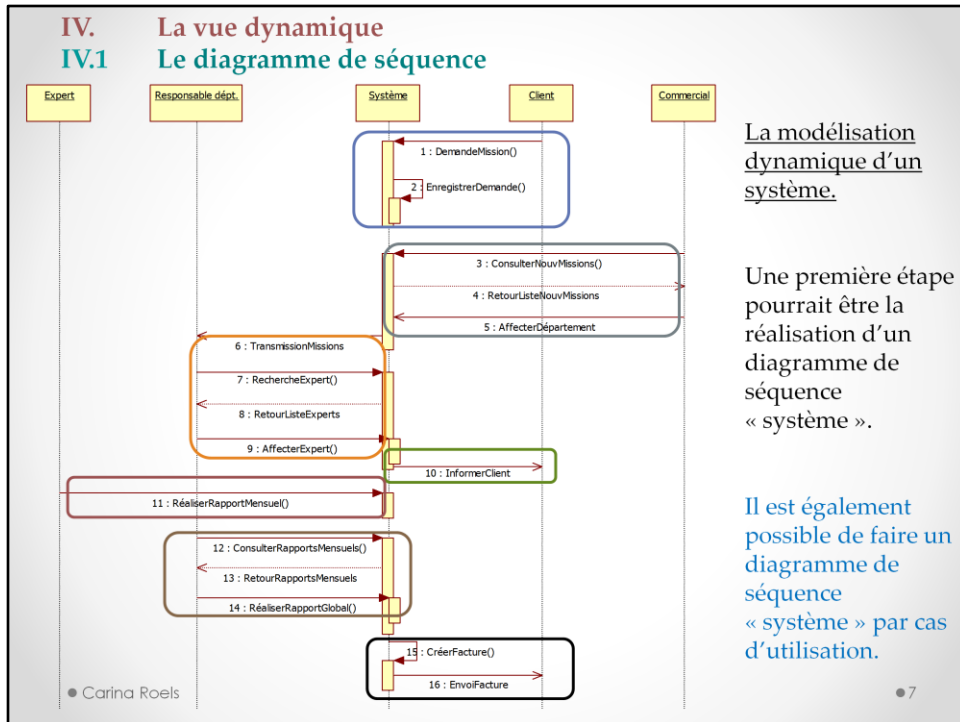
msg()



- **Message dérobant** : n'interrompt pas l'exécution de l'expéditeur et ne déclenche une opération chez le récepteur que s'il s'est préalablement mis en attente de ce message

msg()





Chaque groupe de messages correspond à un cas d'utilisation précis.
On introduit, ici, la notion de la chronologie dans l'utilisation du système.

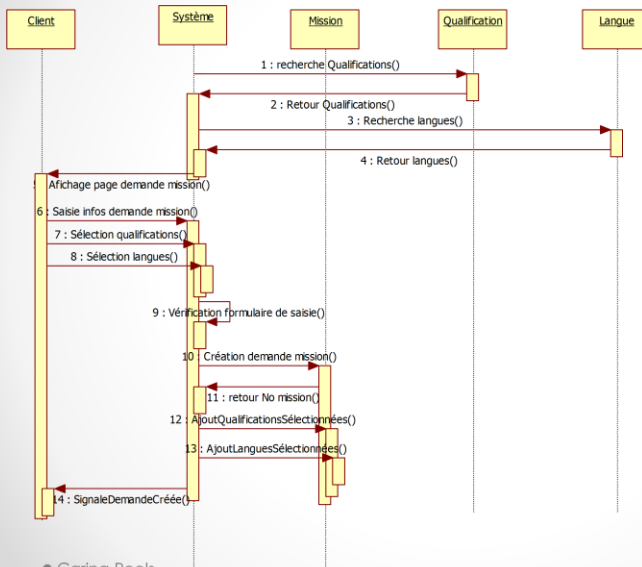
Le diagramme de l'exemple a été réalisé avec l'outil StarUML.

- Le formalisme (les stéréotypes) d'UML 2 n'est pas respecté par l'outil StarUML.
- Une flèche à pointe pleine = un appel (call)
- Une flèche en pointillés = un retour après un appel
- Une flèche à pointe non pleine = un envoi de message
- Une flèche réflexive = une action interne

Par la suite, il est intéressant de réaliser un diagramme de séquence par cas d'utilisation afin de mettre en évidence les interactions entre les objets du système.

IV. La vue dynamique

IV.1 Le diagramme de séquence



Scénario nominal du cas d'utilisation «Demande Mission »

Les scénarios alternatifs du cas d'utilisation «Demande Mission » sont présentés de la même façon, dans un diagramme séparé

Ou

En utilisant des cadres d'interaction.

IV. La vue dynamique

IV.1 Le diagramme de séquence

Il est possible de préciser les spécificités d'un ensemble de messages, sous forme de **cadre d'interaction**:

alt	fragments multiple alternatifs (si alors sinon)
opt	fragment optionnel
par	fragment parallèle (traitements concurrents)
loop	le fragment s'exécute plusieurs fois
region	région critique (un seul thread à la fois)
neg	une interaction non valable
ref	référence à une interaction dans un autre diagramme
sd	fragment du diagramme de séquence en entier

IV. La vue dynamique

IV.2 Le diagramme d'activité

Description

Ce diagramme permet de mettre en évidence la dimension temporelle, tant pour le scénario nominal que pour les scénarios alternatifs et d'erreur d'un cas d'utilisation.

- Proche d'un organigramme
- Donne une vision globale mais macroscopique d'un cas d'utilisation

IV. La vue dynamique

IV.2 Le diagramme d'activité

Formalisme

● Point de démarrage

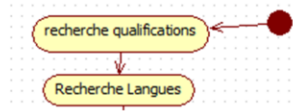
⊙ Point d'arrêt

Création Demande Mission

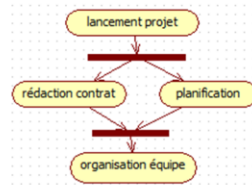
Action



Alternative



Transition



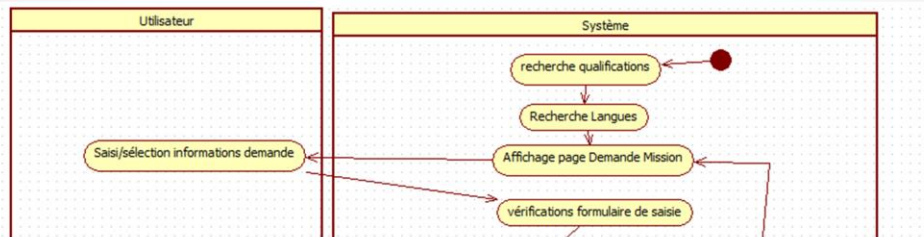
Synchronisation

IV. La vue dynamique

IV.2 Le diagramme d'activité

Formalisme

Couloirs d'activités (Swimlanes)

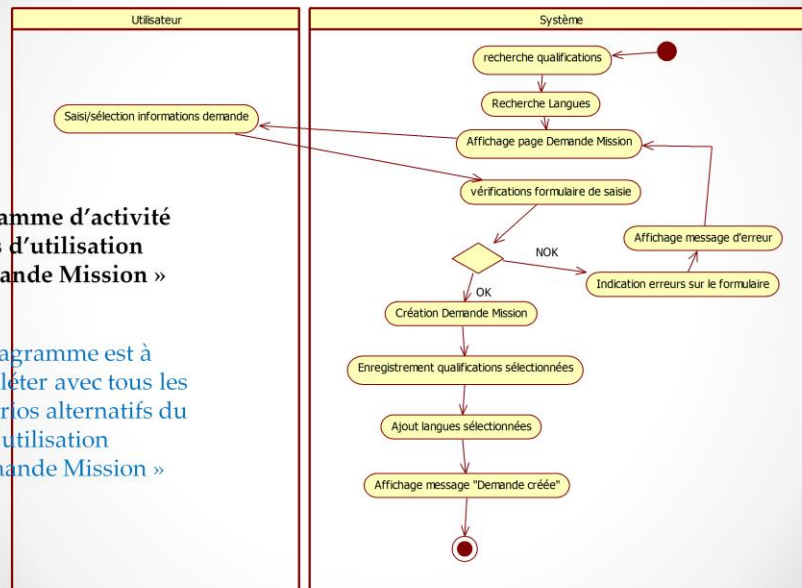


IV. La vue dynamique

IV.2 Le diagramme d'activité

Diagramme d'activité du cas d'utilisation «Demande Mission »

Ce diagramme est à
compléter avec tous les
scénarios alternatifs du
cas d'utilisation
«Demande Mission »



IV. La vue dynamique

IV.2 Le diagramme d'état-transition

Description

Si un objet passe par plusieurs états, on dit que l'objet a un cycle de vie. Tous les objets d'une même classe ont des cycles de vie qui ont la même structure. Cette structure est définie par le diagramme d'état-transition associé à la classe.

Les diagrammes d'états sont basés sur 3 notions :

- des états d'un objet (situation d'un objet définie par ses propriétés)
- des événements
- des comportements des objets (leurs actions et leurs activités).

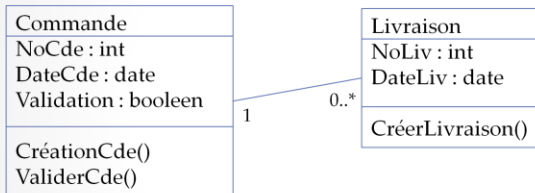
IV. La vue dynamique

IV.2 Le diagramme d'état-transition

Un **état** est une situation stable dans la vie de l'objet.

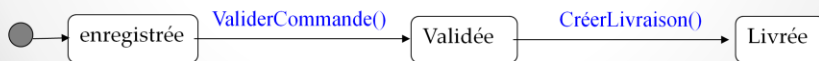
Un objet est toujours dans un état connu.

A un instant donné, un objet est dans un et un seul état.



L'état d'un objet est caractérisé par

- des valeurs des attributs d'un objet.
- l'existence des associations/liens de cet objet aux autres objets.



• Carina Roels

• 15

Dans l'exemple :

- Une commande est créée dans l'état « enregistrée ».
 - Ceci est implémenté par la création d'une instance de la classe COMMANDE avec les attributs NoCde et DateCde valorisés et l'attribut Validation = FALSE.
- Elle passe à l'état « validée » par l'action « ValiderCommande ».
 - Ceci est implémentée par la modification de la valeur de l'attribut Validation = TRUE.
- Elle passe ensuite à l'état « Livrée » par l'action « CréerLivraison ».
 - Ceci correspond à la création d'une livraison liée à l'instance de commande.

La réalisation du diagramme état-transition peut nous aider à découvrir :

- Des attributs supplémentaires
- Des associations supplémentaires
- Des méthodes à ajouter

IV. La vue dynamique

IV.2 Le diagramme d'état-transition

Une **transition** correspond au passage d'un état vers un autre

déclenchée par un événement (méthode)

peut aussi être automatique, lorsqu'on ne spécifie pas l'événement qui la déclenche.

Il est aussi possible de conditionner une transition, à l'aide de "gardes" : il s'agit d'expressions booléennes, exprimées en langage naturel (et encadrées de crochets).

Une transition est représentée par une flèche de l'état d'origine au nouvel état.



IV. La vue dynamique

IV.2 Le diagramme d'état-transition

Les événements externes: sont produits par un acteur et destiné à un objet du système (instance d'une classe) → étiquette d'une transition → méthode dans classe destinataire.

Les événements résultats: sont produits par un objet du système à destination d'un acteur externe (non indiqué en tant qu'étiquette d'une transition).

Les événements temporels signalent l'arrivée d'une échéance temporelle; ils sont générés à la fin d'un délai d'attente ou à l'arrivée d'une date précise (date qui déclenche l'exécution de tâche par le système → étiquette d'une transition → méthode dans classe destinataire.

Les événements modificateurs sont produits par un objet du système à destination d'un autre objet du système → étiquette d'une transition → méthode dans classe destinataire.

Les événements propres sont générés par un objet pour lui-même → étiquette d'une transition → méthode.

● Carina Roels

● 17

Les événements externes:

Le nom d'un tel événement est préfixé par ARR_. Exemple : ARR_paiement, ARR_dde_DVD, ARR_commande.

Les événements résultats:

Le nom d'un tel événement est suffixé par « ENV ». Exemple : ENV_facture, ENV_relance

Les événements temporels : événement signalisant l'arrivée d'une échéance temporelle ; ils sont générés à la fin d'un délai d'attente ou à l'arrivée d'une date précise (date qui déclenche l'exécution de tâche par le système). Ces événements sont non porteur d'information : ils n'ont pas d'argument.

Le nom d'un événement temporel est par exemple : ARR_fin_délai_de_paiement(), ARR_fin_délai_livraison(), ARR_date_facturation().

Les événements modificateurs: événement produit par un objet du système à destination d'un autre objet du système.

Il n'y a pas de convention sur le nom d'un événement modificateur.

Les événements propres: événement généré par un objet pour lui-même

IV. La vue dynamique

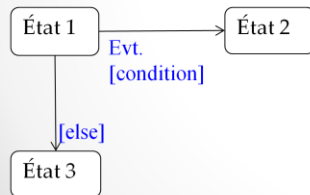
IV.2 Le diagramme d'état-transition

Formalisme 1/2

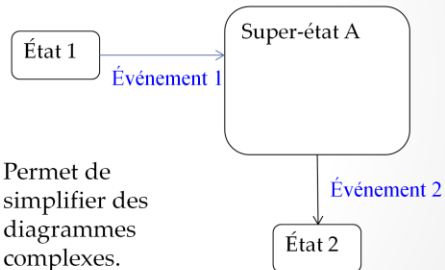
Un diagramme d'état-transition concerne 1 objet ou composant.



Transition conditionnelle



La notion de super-état



Permet de simplifier des diagrammes complexes. Le super-état est alors décrit dans un diagramme à part.

IV. La vue dynamique

IV.2 Le diagramme d'état-transition

Formalisme 2/2

Exemples pour un objet COMMANDE

Il est possible de distinguer l'action et l'événement qui déclenche la transition : événement / action



On peut également indiquer des actions qui sont propres à un état.

Actions 'standards' :

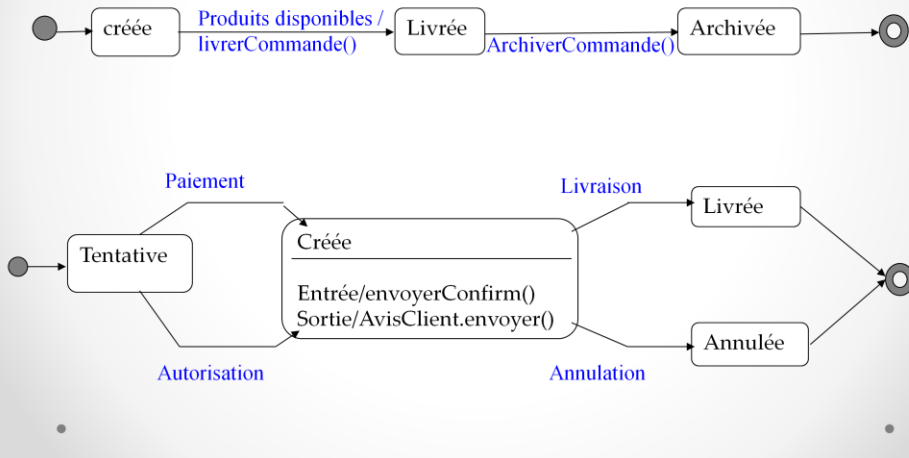
- **entry /action** (action exécutée lors de l'entrée dans l'état)
- **exit /action** (action exécutée lors de la sortie de l'état)
- **on événement /action** (action exécutée lorsque l'événement survient)
- **do /action** (action récurrente ou significative)



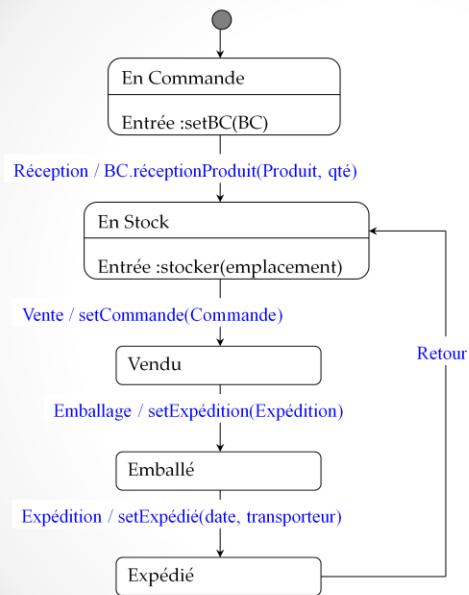
IV. La vue dynamique

IV.2 Le diagramme d'état-transition

Exemples de diagramme d'état d'un objet COMMANDE



Le diagramme d'état de
d'un objet PRODUIT



IV. La vue dynamique

IV.3 Le diagramme de collaboration

- Montre des **interactions entre objets** et les **états des objets** qui interagissent
- concerne des **objets reliés par des liens** et qui se connaissent dans une situation donnée
- **Représentation spatiale** d'une interaction

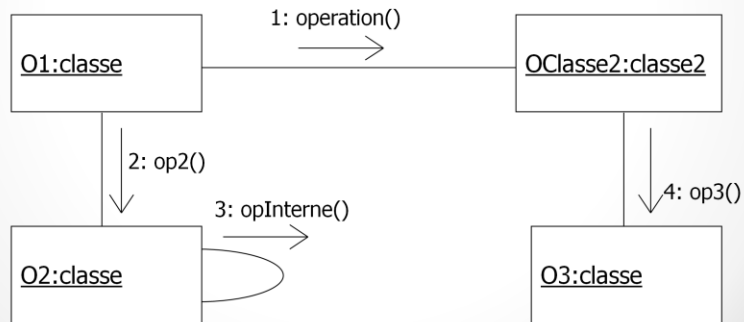
Diagramme de séquence	Diagramme de collaboration
Eléments : messages, objets	Eléments : messages, objets
<u>AVANTAGE :</u> Démontre la création, l'activation et la destruction d'un objet.	<u>AVANTAGE :</u> Aide à valider le diagramme de classes : mise en évidence de la nécessité de chaque association comme moyen de transmission de message

IV. La vue dynamique

IV.3 Le diagramme de collaboration

Messages

- Unité de communication entre rôles
- Regroupent les flots de contrôle (appel de méthode) et les flots de données (valeurs)
- Les **messages** échangés par les objets sont représentés le long des liens
- L'**ordre d'envoi** des messages est matérialisé par un **numéro de séquence**



• Carina Roels

• 23

Synchronisation des messages

- UML permet de spécifier de manière très précise l'ordre et les conditions d'envoi des messages sur un diagramme dynamique
- Pour chaque message, il est possible d'indiquer :
 - les clauses qui conditionnent son envoi
 - son rang (son numéro d'ordre par rapport aux autres messages)
 - sa récurrence
 - ses arguments

• Carina Roels

• 24

- **pré / [cond] séq * | | [iter] : r := msg (par)**
- **pré** : prédécesseurs (liste de numéros de séquence de messages)
Indique que le message ne sera envoyé que lorsque tous ses prédécesseurs le seront
- **[cond]** : expression booléenne.
Permet de conditionner l'envoi du message
- **séq** : numéro de séquence du message.
Indique le rang du message, c'est-à-dire son numéro d'ordre par rapport aux autres messages.
Il est possible de représenter le niveau d'emboîtement des messages et leur précedence, à l'aide de chiffres séparés par des points.
Exemple : l'envoi du message 1.3.5 suit immédiatement celui du message 1.3.4 et ces deux messages font partie du flot (de la famille de messages) 1.3.
Pour représenter l'envoi simultané de deux messages, il suffit de les indexer par une lettre.
Exemple : l'envoi des messages 1.3.a et 1.3.b est simultané.
- **iter** : récurrence du message.
Permet de spécifier l'envoi séquentiel (ou en parallèle, avec " | ") de messages. Il est aussi possible de spécifier qu'un message est récurrent en n'utilisant que "*" ou "* | |".
- **r** : valeur de retour du message.
Permet d'affecter la valeur de retour d'un message
- **msg** : nom du message.
- **par** : paramètres du message

Exemples:

- 3 : bonjour() *Ce message a pour numéro de séquence "3".*
- [heure = midi] 1 : manger() *Ce message n'est envoyé que s'il est midi.*
- 1.3.6 * : ouvrir() *Ce message est envoyé de manière séquentielle un certain nombre de fois.*
- 3 / *||[i := 1..5] : fermer() *Représente l'envoi en parallèle de 5 messages. Ces messages ne seront envoyés qu'après l'envoi du message 3.*
- 1.3, 2.1 / [t < 10s] 2.5 : age := demanderAge(nom, prenom) *Ce message (numéro 2.5) ne sera envoyé qu'après les messages 1.3 et 2.1, et que si "t < 10s".*
La valeur de retour est affectée à âge. Le message prend nom et prénom en paramètres

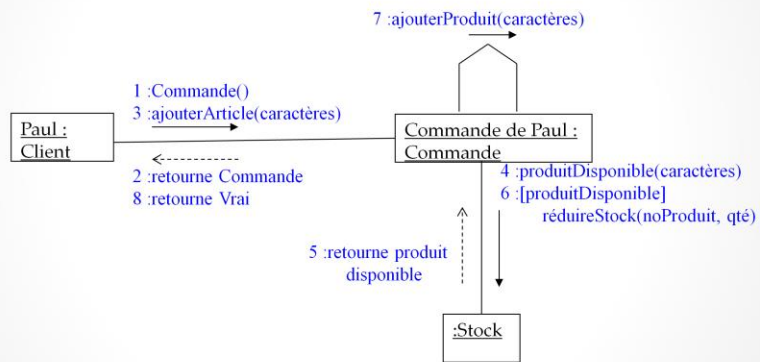
- 1.3 / [disk full] 1.7.a * : deleteTempFiles()
- 1.3 / [disk full] 1.7.b : reduceSwapFile(20%)

Ces messages ne seront envoyés qu'après l'envoi du message 1.3 et si la condition "disk full" est réalisée. Si cela est le cas, les messages 1.7.a et 1.7.b seront envoyés simultanément. Plusieurs messages 1.7.a peuvent être envoyés.

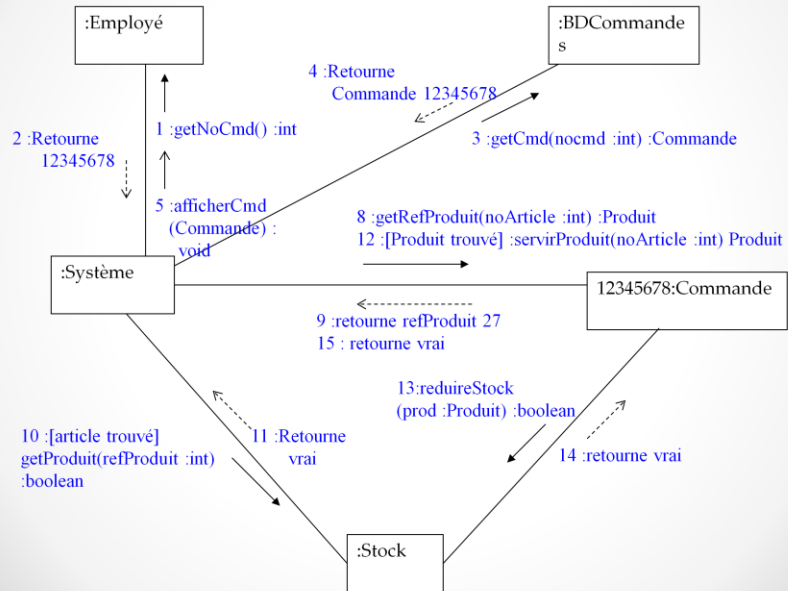
Types de message

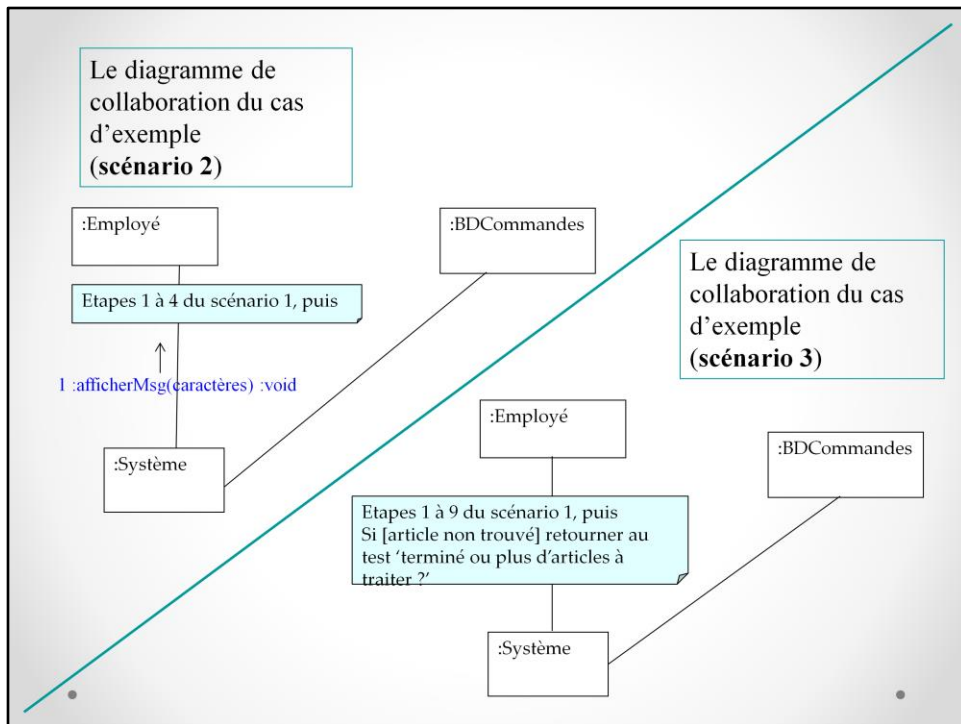
- Mêmes types de message que pour les séquences :
 - Message simple
 - Message minuté
 - Message synchrone
 - Message asynchrone
 - Message déroband

Un diagramme de collaboration simple



Le diagramme de collaboration du cas d'exemple (scénario 1)





Le diagramme de collaboration du cas d'exemple
(scénario 4)

Etapes 1 à 13 du scénario 1,
puis

