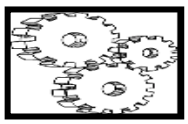
1

Analyse et conception de Systèmes Informatiques Orientés objets en UML

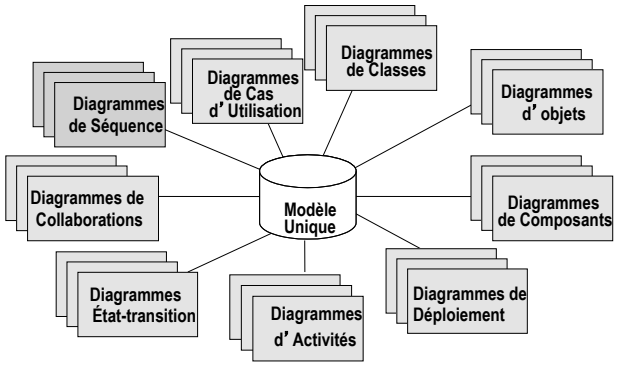
Abdelhak-Djamel SERIAI
seriai@lirmm.fr

2

Les vues dynamiques



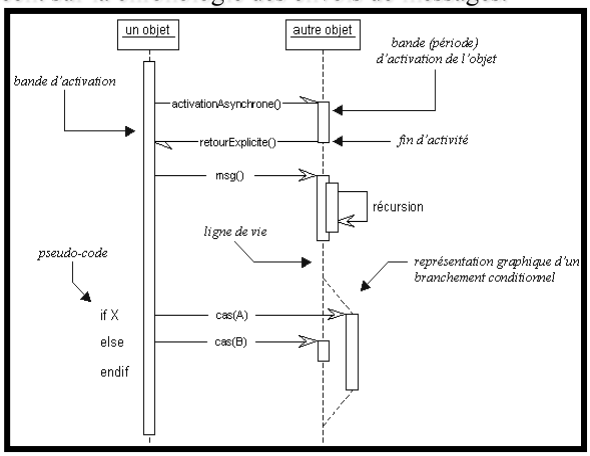
3



4

DIAGRAMME DE SEQUENCE (1)

- Sémantique
  - Permettent de représenter des collaborations entre objets selon un point de vue temporel
    - On y met l'accent sur la chronologie des envois de messages.



5

DIAGRAMME DE SEQUENCE (2)

- Sémantique
  - Contrairement au diagramme de collaboration, on n'y décrit pas le contexte ou l'état des objets
    - La représentation se concentre sur l'expression des interactions
  - L'ordre d'envoi d'un message est déterminé par sa position sur l'axe vertical du diagramme ;
    - Le temps s'écoule "de haut en bas" de cet axe
  - La disposition des objets sur l'axe horizontal n'a pas de conséquence pour la sémantique du diagramme
  - Peuvent servir à illustrer un cas d'utilisation

6

DIAGRAMME DE SEQUENCE (3)

- Types de messages

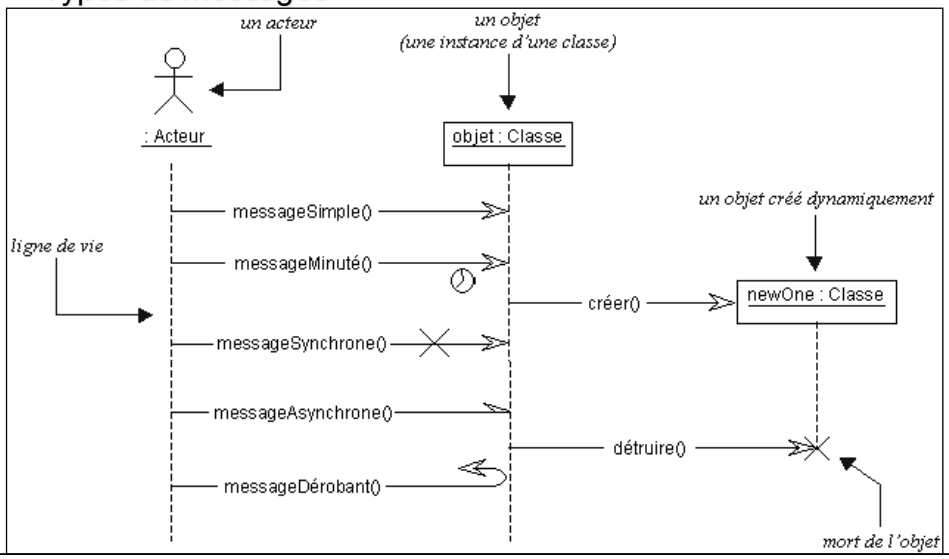


DIAGRAMME DE SEQUENCE (4)

- Types de messages
  - Message simple :
    - Message dont on ne spécifie aucune caractéristique d'envoi ou de réception particulière
  - Message minuté (timeout) :
    - Bloque l'expéditeur pendant un temps donné (qui peut être spécifié dans une contrainte), 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.

DIAGRAMME DE SEQUENCE (5)

- Types de messages
  - Message asynchrone
    - N'interrompt pas l'exécution de l'expéditeur.
    - Le message envoyé peut être pris en compte par le récepteur à tout moment ou ignoré (jamais traité).
  - 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.

DIAGRAMME DE SEQUENCE (6)

- Activation d'un objet
  - Il est possible de représenter de manière explicite les différentes périodes d'activité d'un objet
    - au moyen d'une bande rectangulaire superposée à la ligne de vie de l'objet
  - On peut aussi représenter des messages récursifs
    - En dédoublant la bande d'activation de l'objet concerné
  - Pour représenter de manière graphique une exécution conditionnelle d'un message,
    - On peut documenter un diagramme de séquence avec du pseudo-code et représenter des bandes d'activation conditionnelles.

DIAGRAMME DE SEQUENCE (7)

- Activation d'un objet

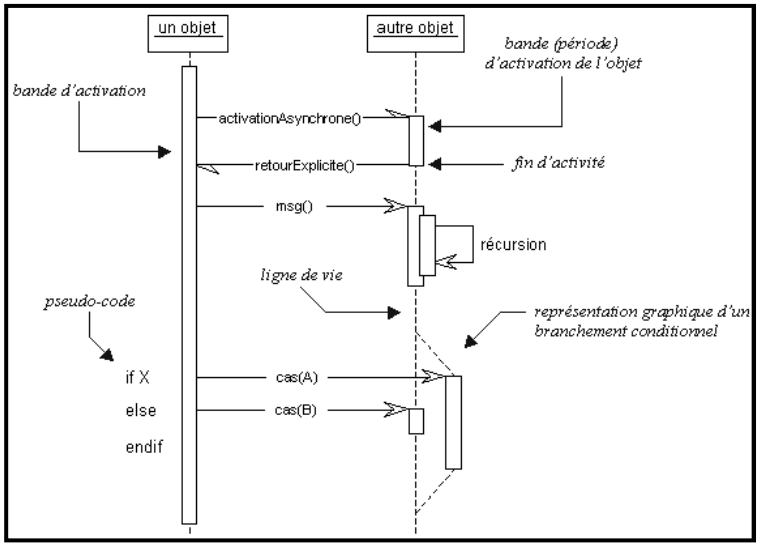
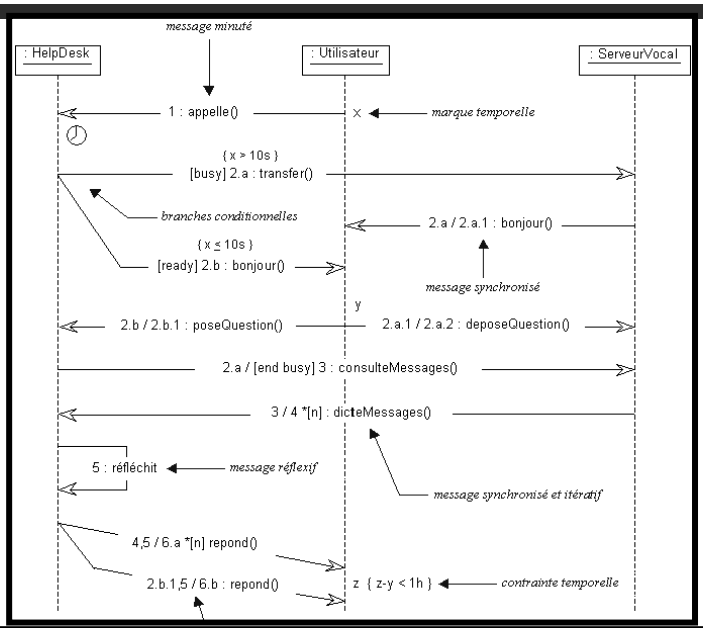


DIAGRAMME DE SEQUENCE (8)

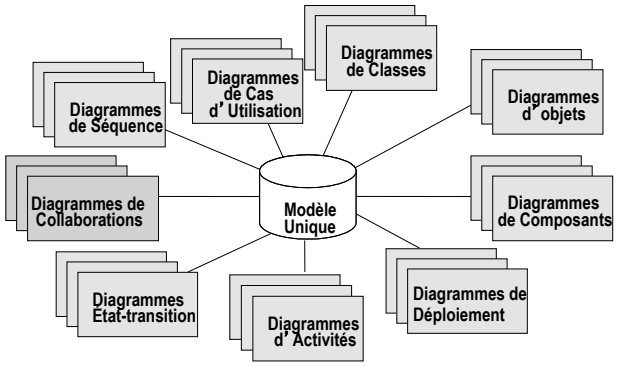
- Remarques
  - Ne confondez pas la période d'activation d'un objet avec sa création ou sa destruction.
    - Un objet peut être actif plusieurs fois au cours de son existence
  - Le pseudo-code peut aussi être utilisé pour indiquer des itérations (avec incrémentation d'un paramètre d'un message par exemple)
  - Le retour des messages asynchrones devrait toujours être matérialisé, lorsqu'il existe
  - Préférez l'utilisation de contraintes à celle de pseudo-code
  - Un message réflexif ne représente pas l'envoi d'un message, il représente une activité interne à l'objet (qui peut être détaillée dans un diagramme d'activités) ou une abstraction d'une autre interaction (qu'on peut détailler dans un autre diagramme de séquence).

DIAGRAMME DE SEQUENCE (9)

- Exemple



13



14

LA COLLABORATION (1)

- Symbole de modélisation "collaboration"
- Les collaborations sont des interactions entre objets, dont le but est de réaliser un objectif du système (c'est-à-dire aussi de répondre à un besoin d'un utilisateur)
- L'élément de modélisation UML "collaboration", représente les classes qui participent à la réalisation d'un cas d'utilisation.
- Attention : ne confondez pas l'élément de modélisation "collaboration" avec le diagramme de collaboration, qui représente des interactions entre instances de classes.

LA COLLABORATION (2)

- **Exemple**

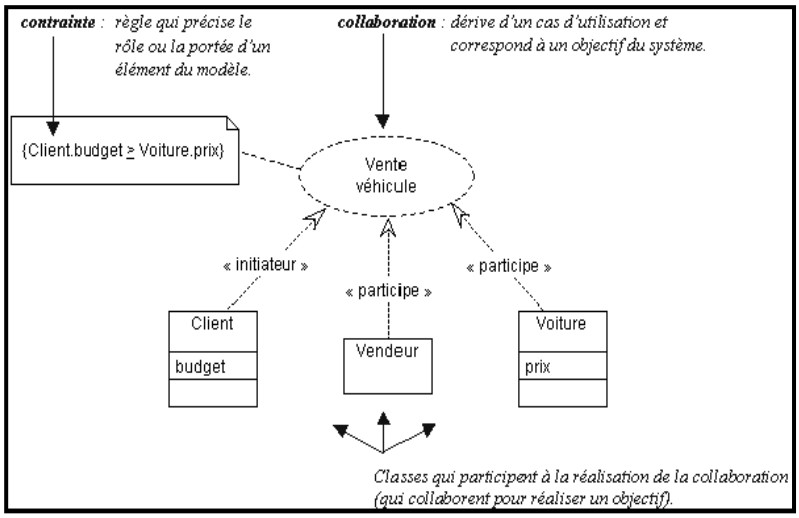
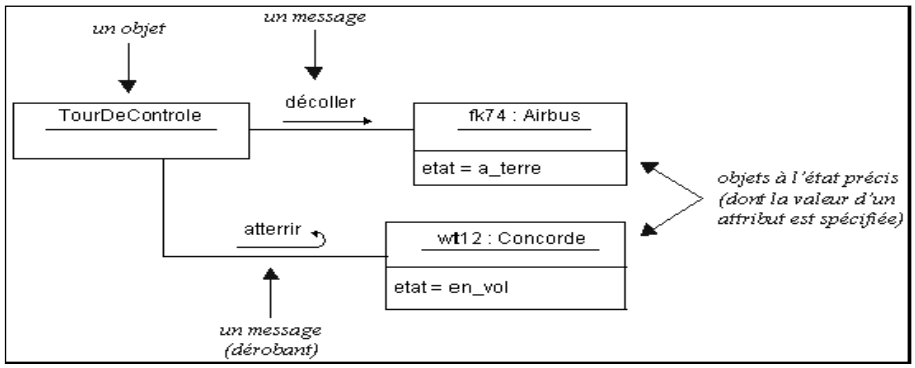


Diagramme de collaboration (1)

- C'est quoi ?



17

Diagramme de collaboration (2)

- Sémantique
  - Montrent des interactions entre objets
    - Les objets peuvent êtres des instances de classes et acteurs
  - Permettent de représenter le contexte d'une interaction
    - Car on peut y préciser les états des objets qui interagissent.

18

Diagramme de collaboration (3)

- 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.
  - La syntaxe d'un message est la suivante

[pré "/"] [["["cond"]"] [séq] ["*"["||"]["["iter"]"]] ":"] [r ":="] msg"("[par]")"

19

Diagramme de collaboration (4)

- Synchronisation des messages
  - pré : prédécesseurs
    - Liste de numéros de séquence de messages séparés par une virgule
    - Indique que le message courant ne sera envoyé que lorsque tous ses prédécesseurs le seront aussi
      - 3 / 5 : fermer()
        - Représente l'envoi du message numéro 5.
        - Ce messages ne sera envoyé qu'après l'envoi du message 3.

20

Diagramme de collaboration (5)

- Synchronisation des messages
  - cond : garde, expression booléenne.
    - Permet de conditionner l'envoi du message, à l'aide d'une clause exprimée en langage naturel
      - [heure = midi] 1 : manger() :
        - Ce message n'est envoyé que s'il est midi.

21

Diagramme de collaboration (6)

- Synchronisation des messages
  - séq : numéro de séquence du message
    - Indique le rang du message
      - c'est-à-dire son numéro d'ordre par rapport aux autres messages
    - 3 : bonjour()
      - Ce message a pour numéro de séquence "3".
    - Les messages sont numérotés à la façon de chapitres dans un document, à l'aide de chiffres séparés par des points.
      - 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.
      - L'envoi des messages 1.3.a et 1.3.b est simultané.

22

Diagramme de collaboration (7)

- Synchronisation des messages
  - iter : récurrence du message.
    - Permet de spécifier en langage naturel l'envoi séquentiel (ou en parallèle, avec "||") de messages.
    - Il est aussi possible de spécifier qu'un message est récurrent en omettant la clause d'itération (en n'utilisant que "*" ou "*||").
      - **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.

Diagramme de collaboration (8)

- Synchronisation des messages
  - r : valeur de retour du message.
    - Permet d'affecter la valeur de retour d'un message, pour par exemple la retransmettre dans un autre message, en tant que paramètre.
  - msg : nom du message
  - par : paramètres (optionnels) du message.
    - 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"
    - 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.

Diagramme de collaboration (9)

- Synchronisation des messages
  - Objets actifs (threads)
    - UML permet de représenter des communications entre objets actifs de manière concurrente
    - Permet de représenter des communications entre processus (threads)

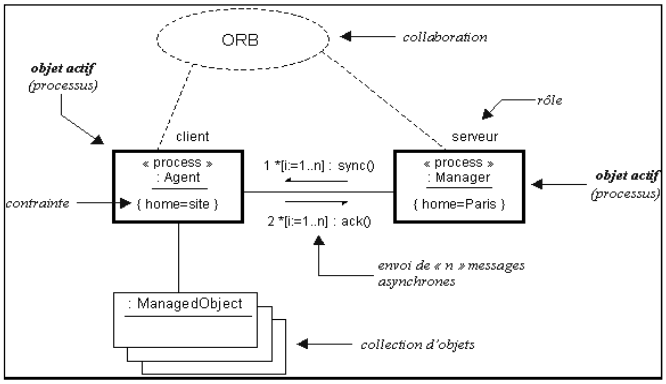


Diagramme de collaboration (10)

- Synchronisation des messages