Analyse, Conception Objet

Diagrammes de Collaboration

Une partie du matériau de ce cours est issue du cours de S.Galland (Stephane.Galland@emse.fr)

Octobre 2002

Sommaire

Définition

- Description des interactions entre les objets composant le système.
- Représentation se concentrant sur les relations d'interaction entre les objets.
- La dimension temporelle est ajoutée grâce à des numéros de séquence.
- Représente un ensemble de rôles joués par les objets dans un contexte particulier, ainsi que les liens entre ces objets.
- Les diagrammes de collaboration sont des diagrammes d'interaction comme les diagrammes de séquence.
- Le passage à un diagramme de séquences et inversement est simple.

Utilisation des diagrammes de collaboration

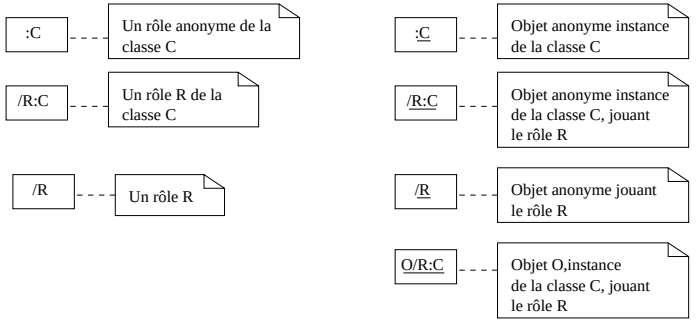
- Documentation des cas d'utilisation :
  - description permettant de réaliser les cas d'utilisation.
  - décrit le comportement du système pour chacun des scénarios accompagnant les cas d'utilisation.
  - Facilite la rédaction des diagrammes des classes, des diagrammes état-transition, ...
- Documentation conceptuelle :
  - description du comportement de classes et d'opérations.
  - **Remarque :** si les opérations ont une structure algorithmique, on préférera les décrire avec des diagrammes d'activités.
    Les diagrammes de collaboration sont plus adaptés quand une opération fait intéragir de nombreux objets.

Collaboration

- Définition des éléments utiles pour obtenir un résultat en spécifiant leurs rôles dans le contexte de la collaboration.
- Est composée de deux description :
  - description générale au niveau spécification qui représente :
    - les rôles des cas d'utilisations, des classes, des méthodes et des associations;
    - une interaction : *une séquence de messages partiellement ordonnés échangés entre les rôles.*
  - description spécifique au niveau instance qui représente :
    - une instance particulière d'une interaction composés d'objets et de liens respectant les rôles, et de stimulus (instances de messages) échangés entre ces objets.

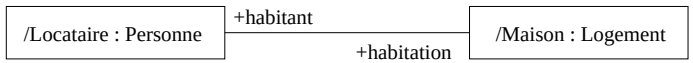
Collaboration : rôles

- Chaque élément d'une collaboration joue un rôle.
- Les rôles des classificateurs (classes, cas d'utilisation, ...) est représenté par un symbole de classe :



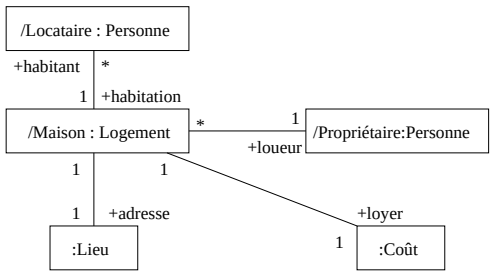
Collaboration : rôles (suite)

- Les rôles des associations sont des textes respectant la syntaxe des étiquettes d'associations (diagrammes de classes).



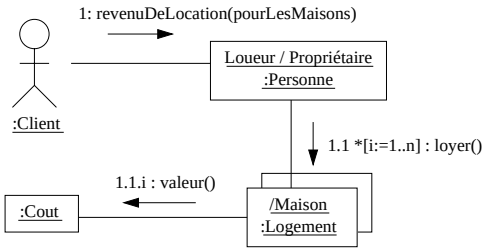
Collaboration : niveau spécification

- La collaboration forme un graphe de rôles liés par des rôles d'associations.
- En général, une collaboration au niveau spécification représente un contexte.



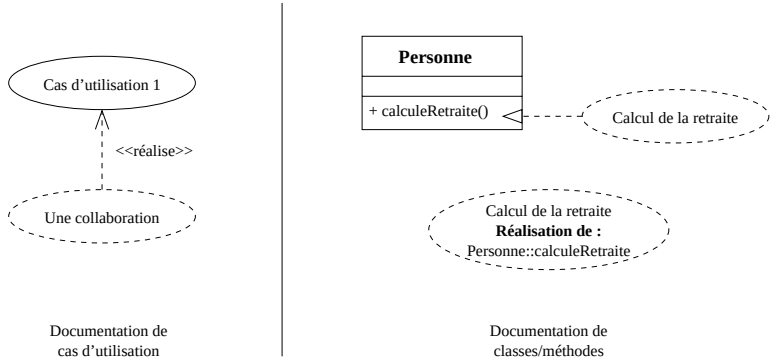
Collaboration : au niveau instance

- Diagramme représentant une instance du diagramme au niveau spécification avec des stimulus.
- Stimulus : instance d'un message envoyé d'un objet vers un autre.



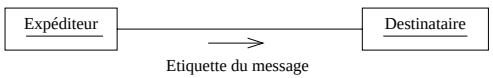
Collaboration : représentation condensée

- Dans UML, une collaboration est représentée comme suit :



Messages

- Les objets communiquent en échangeant des messages représentés sous forme de flèches.
- Les messages sont étiquettés par le nom de l'opération ou du signal invoqué.
- L'envoi d'un message nécessite que le récepteur puisse réaliser l'opération.

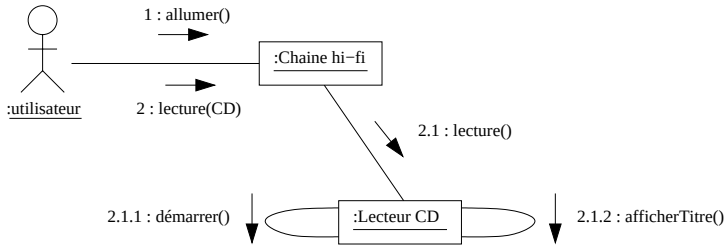


Messages : étiquettes

- Les étiquettes décrivent les messages auxquels elles sont attachées.
- Syntaxe générale:
  `[synchronisation] ['['garde']'] [séquence] [itération] [résultat :=] nom_message ['('arguments')']`
- `nom_message` : nom de l'opération ou du signal invoqué par l'intermédiaire de ce signal
- `garde` : condition booléenne et optionnelle (représentée entre corchets) autorisant ou non l'envoi d'un message.

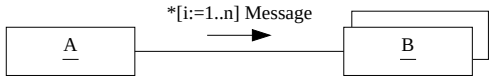
Messages : séquence

- Ensemble de numéros ordonnant l'envoi des messages (`1` puis `2` puis `3` ...)
- Numérotation englobante (cas d'appels de procédure) : `2` (appel initial) puis `2.1` (premier appel imbriqué) puis `2.2` (second sous-appel) puis `3` (appel du même niveau que le numéro `2`).



Messages : itération

- Itération séquentielle : envoi séquentiel de *n* instances du même message.
  Syntaxe : `*` [clause d'itération]
- Itération parallèle : envoi parallèle de *n* instances du même message.
  Syntaxe : `*||` [clause d'itération]

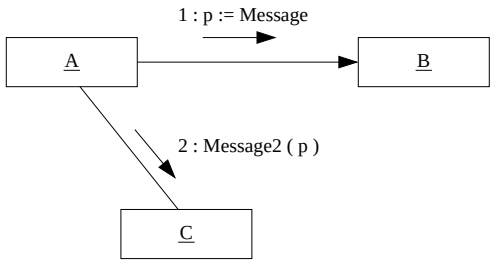


Messages : arguments

- Liste des paramètres du message séparés par des virgules.
- Les arguments et le nom de l'action déterminent sans ambiguïté l'action à réaliser.
- Les arguments peuvent contenir des valeurs retournées par des messages envoyés précédemment.
- Exemples :
  `Afficher ( x, y )` – affiche les valeurs x et y
  `Soustraire( Aujourd'hui, DateDeNaissance )` – calculer le nombre de jours entre deux dates

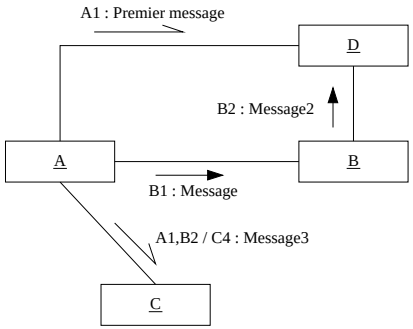
Messages : résultat

- Le résultat est constitué d'une liste de valeurs retournées par le message.
- Ces valeurs peuvent être utilisées comme paramètres des autres messages.



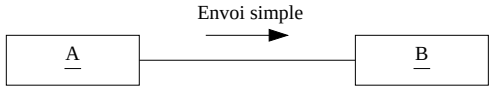
Messages : synchronisation

- Synchronization de message : envoi d'un message ssi d'autres messages ont déjà été envoyés.
- Syntaxe : `message1, message2 ... /`



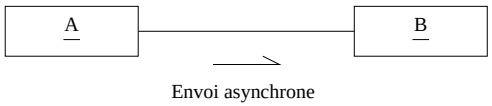
Messages : appel de procédure

- Dans un appel de procédure (flot de contrôle emboîté), la séquence emboîtée doit se terminer pour que la séquence englobante reprenne le contrôle.
- Les appels de procédure sont représentés par des flèches à pointe triangulaire.



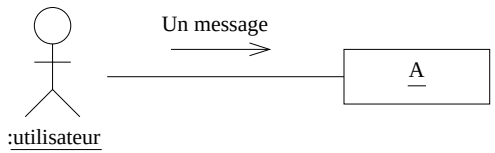
Messages : flot de contrôle asynchrone

- Dans un flot de contrôle asynchrone, il n'y pas de message englobé ou englobant
  $\Rightarrow$ pas d'obligation de terminaison d'autre messages pour continuer.
- Les messages asynchrones sont représentés par des demi-flèches.



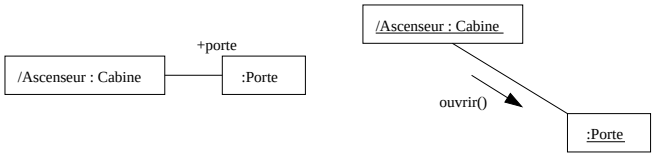
Messages : flot de contrôle à plat

- Cas particulier de messages asynchrones.
- Modélisation d'une progression non procédurale souvent utilisée pour les messages entre un acteur et le système.
- Les messages sont représentés par des flèches simples.



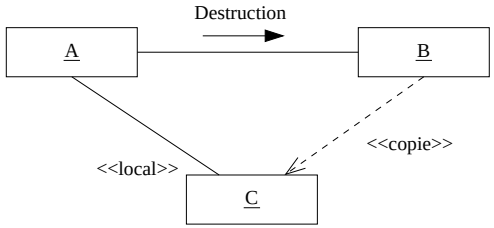
Interactions

- Définit la communication entre les instances des éléments d'une collaboration.
- Plusieurs interactions peuvent s'appliquer à la même collaboration pour exprimer divers comportements.
- Le contexte d'une interaction comprend les arguments, les variables locales, l'état des objets ainsi que les liens entre les objets qui participent à la collaboration.



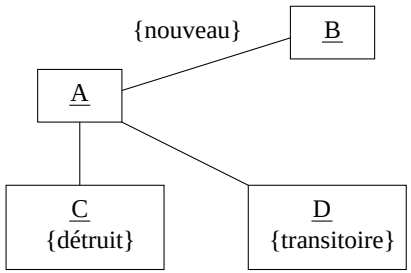
Interactions (suite)

- Les diagrammes d'interactions montrent les interactions entre les objets et les relations structurelles permettant ces interactions.



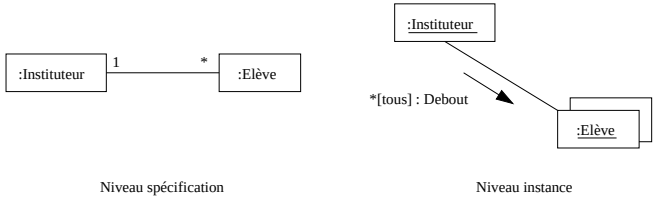
Interactions (suite)

- Les objets et les liens créés ou détruits au cours d'une interaction peuvent respectivement porter les contraintes `nouveau` ou `détruit`.
- Les objets créés, puis détruits au sein de la même interaction, portent la contrainte `transitoire`.



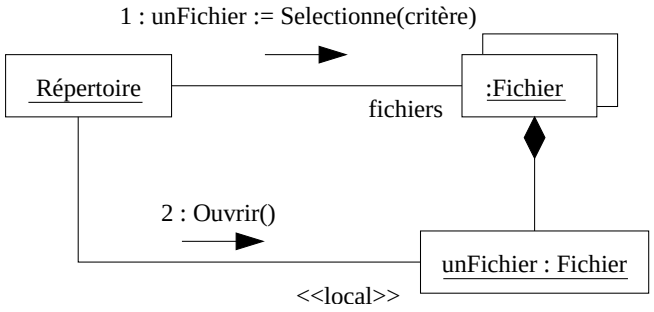
Interactions (suite)

- UML permet une représentation condensée d'un ensemble d'objets.
- Utile lorsque tous les objets de l'ensemble doivent être traités de manière uniforme.



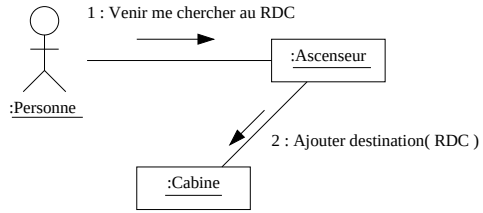
Interactions (suite)

- Possibilité de représenter un objet particulier appartenant à un groupe afin de lui appliquer un message particulier
- Représentation à l'aide d'une composition indiquant que l'objet fait parti de l'ensemble d'objets.



Représentation des acteurs

- La notation UML permet de faire figurer un acteur dans les diagrammes de collaboration.
- Ils permettent de représenter les intéractions déclenchées par un élément extérieur au système.
- Le premier message est envoyé par l'acteur.



Classes abstraites et interface

- Les classes abstraites et les interfaces peuvent figurer dans les diagrammes de collaboration.
- Elles représentent des informations complémentaires : liens polymorphes, réalisation d'interfaces, ...