UML
Diagrammes de Collaboration

Collaboration, le lien entre modèle externe et interne

Collaboration et cas d'utilisation

Collaboration et diagramme de classe

De l'analyse à la conception objet en UML

- L'analyse fonctionnelle conduit à un modèle fonctionnel : comportement externe conforme aux besoins du client.

- La conception objet conduit à un modèle du comportement interne du logiciel et de sa structuration

Des cas d'utilisation aux diagramme de classe

Contenu de ce cours

- Diagrammes de collaboration
- D'un diagramme de collaboration à un diagramme de classes

Notion de collaboration

- Principe de modélisation objet
  - Les objets sont indépendants
  - Chaque objet est responsable de certaines activités
  - Pour réaliser une activité, il faut le concours collaboratif d'objets
  - Les objets collaborent via des interactions

  Pas de système hiérarchique : système basé sur l'individu « objet »

- Collaboration
  - Modélise une activité réalisée par un ensemble d'objets
  - Peut représenter un scénario d'un cas d'utilisation du point de vue du fonctionnement interne du système

Diagrammes de collaboration

- Types de diagrammes
  - Diagrammes statiques : diagrammes d'objets
    - Exprime les liens entre objets impliqués dans une collaboration
  - Diagrammes dynamiques : diagrammes de collaboration
    - Interactions entre objets
    - Interactions entre rôles d'objets
- Composants
  - Objets
  - Rôles
  - Classes
  - Acteurs
  - Liens

Diagramme d'objets

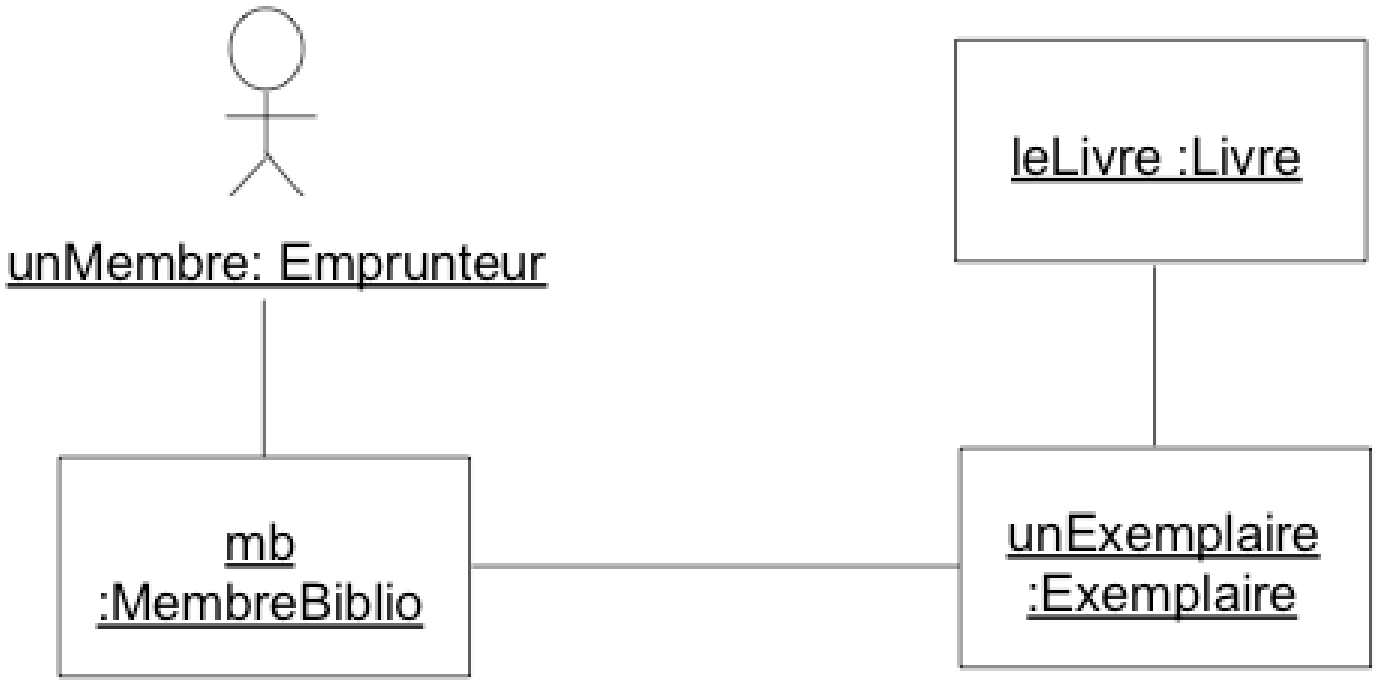
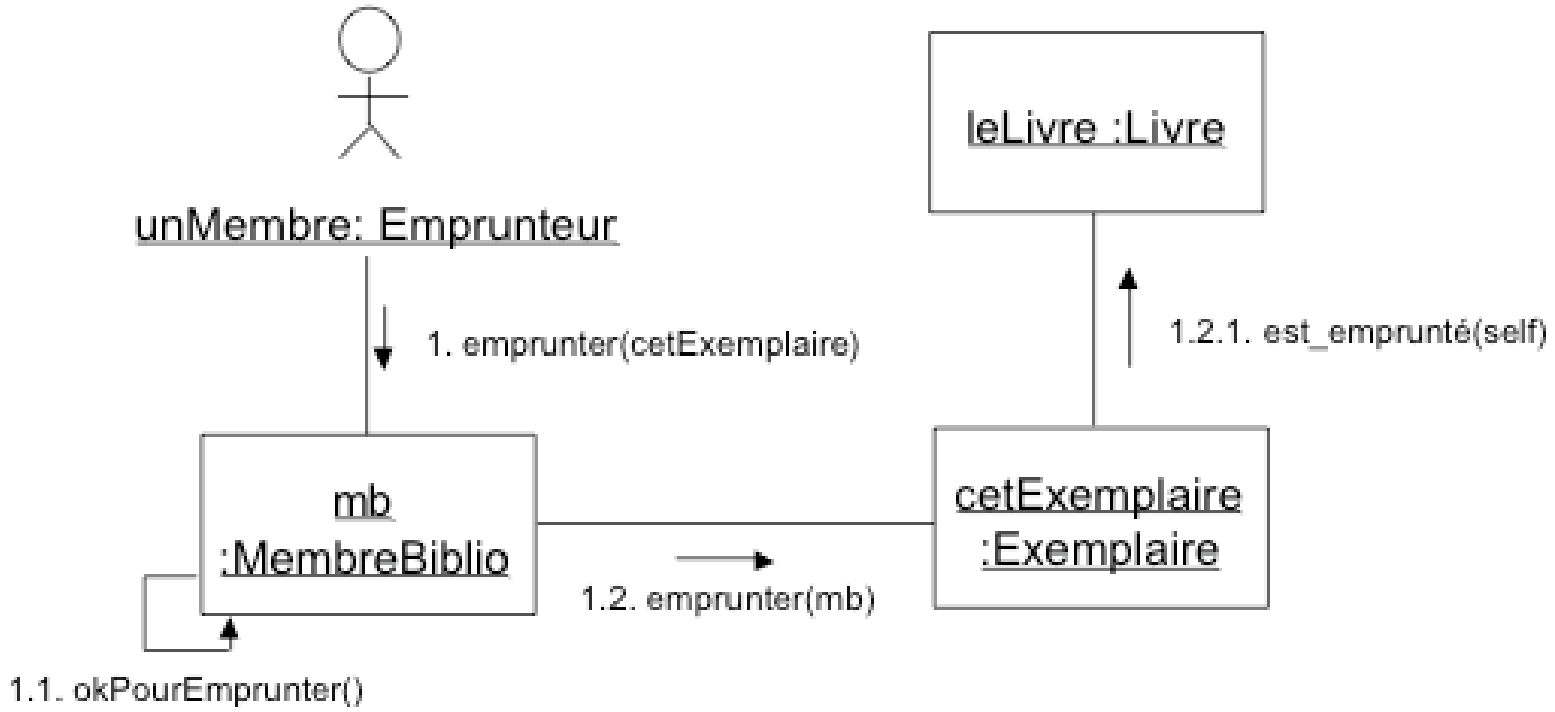
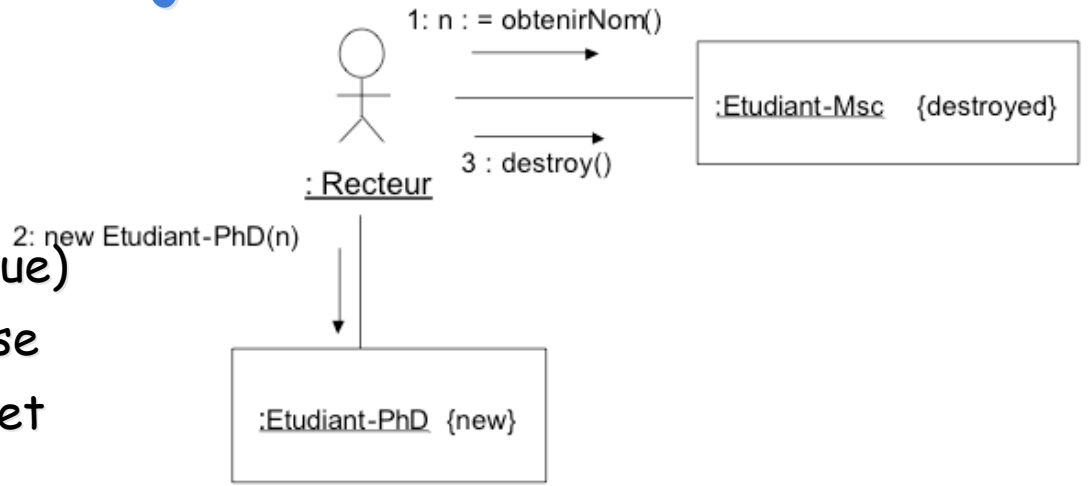


Diagramme de collaboration entre objets



Objets

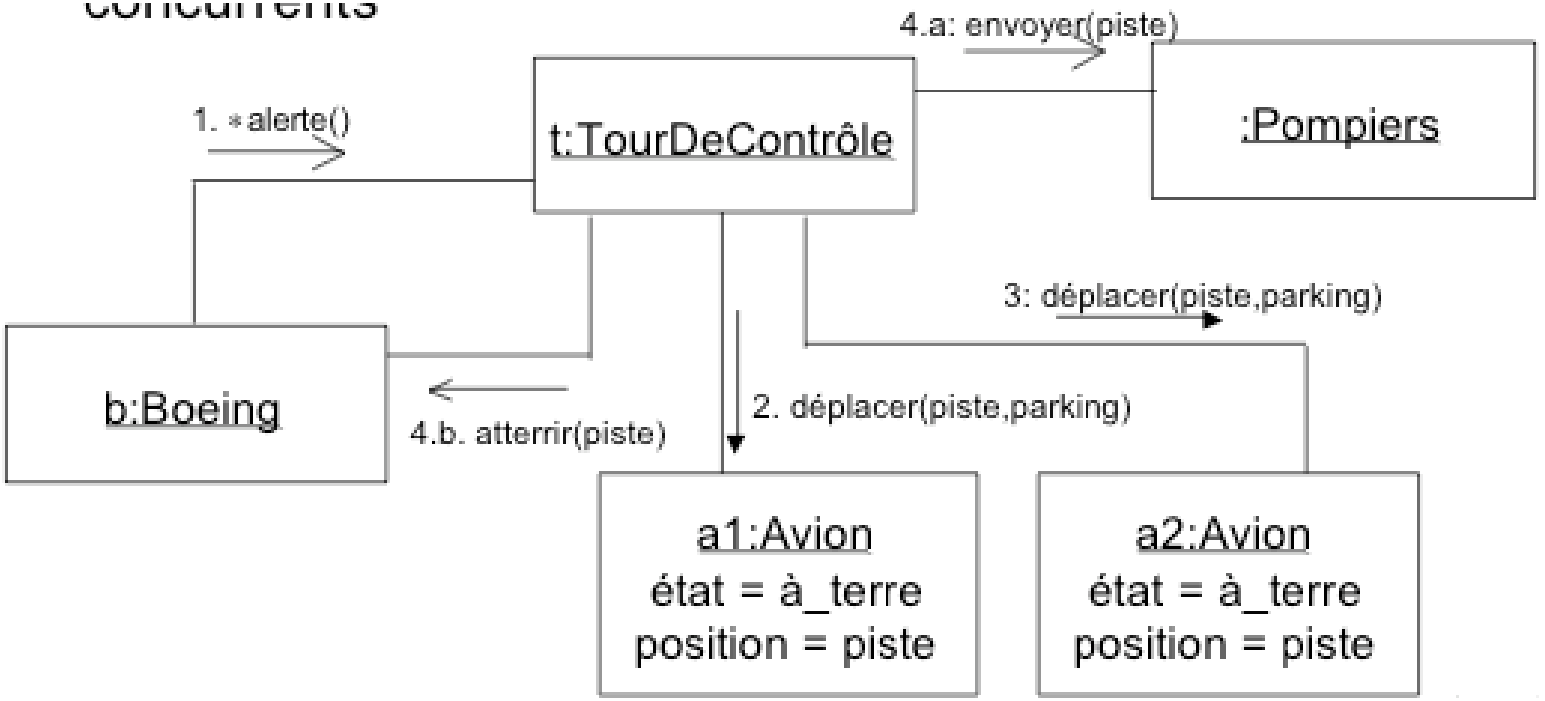


- Objet
  - Nom de l'objet
  - Classe de l'objet (si connue)

  Syntaxe : cetObjet:SaClasse
  - (facultatif) État de l'objet
    - Création/destruction
    - États intermédiaires

  Syntaxe : {new} {destroyed} {ouvert} {en cours}
- Acteur : objet « externe » participant à la collaboration
- Rôle : rôle d'un objet dans une collaboration
  - Syntaxe : cetObjet / sonRole : SaClasse (tous facultatifs)
  - Cas particulier : rôle unique de l'objet dans la collaboration → rôle non explicité

Exemple états objets



Liens

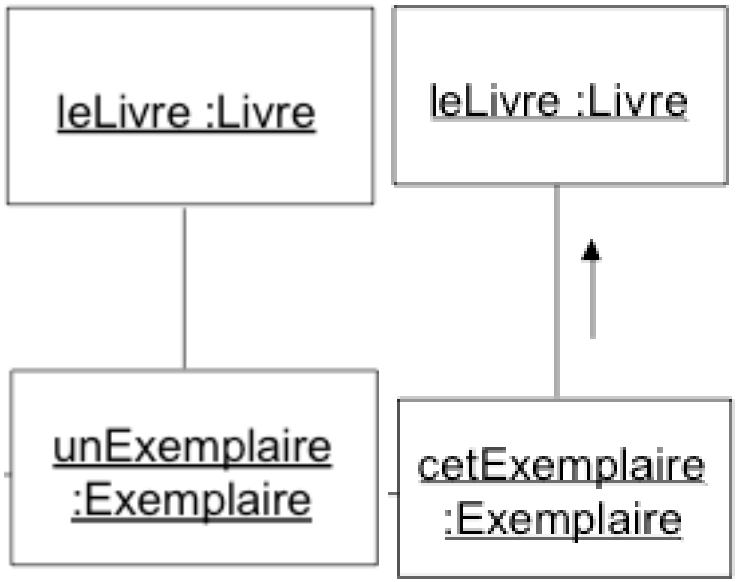
- Types de liens
  - Diagrammes d'objets

Interaction = instance d'association entre objets (instance de classe)

- Diagrammes de collaboration

Interaction = message entre objets.

- Syntaxe graphique
  - Instance d'association : fil simple.
  - Message : flèche ajoutée au fil.



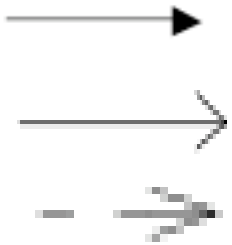
Message

- Représente une demande faite par l'objet émetteur à l'objet destinataire
- Le destinataire doit pouvoir comprendre le message

Message = méthode de l'objet destinataire

Nécessité d'une association entre les classes correspondantes des objets.

- Syntaxe graphique
  - Message synchrone
  - Message simple ou asynchrone
  - Retour de message



Message : syntaxe détaillée

[prédécesseurs "/"] [["[" garde "]"] [numéro_de_séquence]
["*" ["||"]["[" itération "]"]] ":"] [résultat ":="]
message "(" [paramètres] ") »

- Garde : condition d'envoi du message
- Numéro de séquence : ordre du message dans la collaboration
- Prédécesseurs : liste des numéros de séquence des message devant être envoyés avant le message
- Itération : type d'itération (informel et précisé par * ou //)
- Message : méthode invoquée
- Paramètres : paramètre effectifs de la méthode

SANS OUBLIER : économie de moyens et KISS

Messages et types d'objets

- Messages synchrones asynchrones

Objets actifs versus passifs

  - passifs : il faut leur envoyer un message pour qu'ils s'activent
  - actifs : constamment en activité, autonomes = peuvent faire des calcul et envoyer des messages sans être requis par un autre.

- Message synchrones

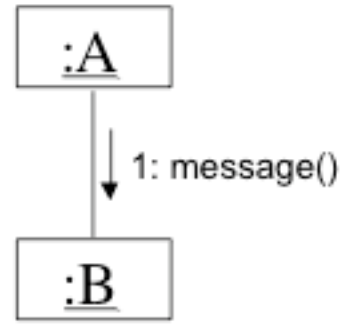
L'envoyeur perd le contrôle et reste actif en attente de réponse

- Messages asynchrones

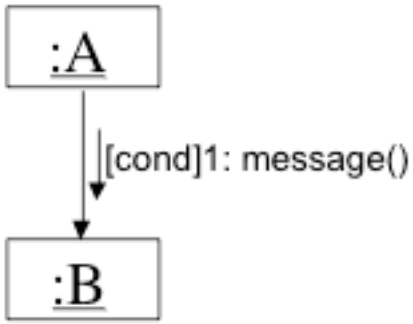
Envoi de message sans attente de réponse

  - objet actif : garde le contrôle
  - objet passif : perd le contrôle et s'inactive

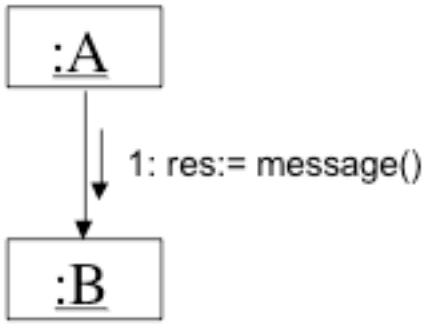
Résumé syntaxe



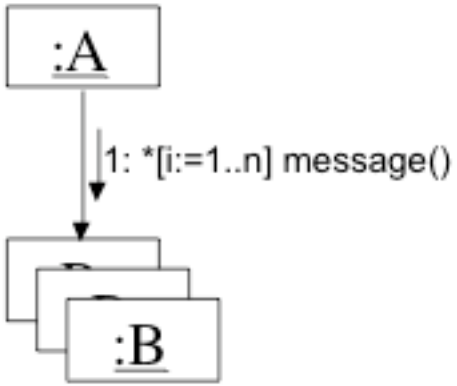
Message



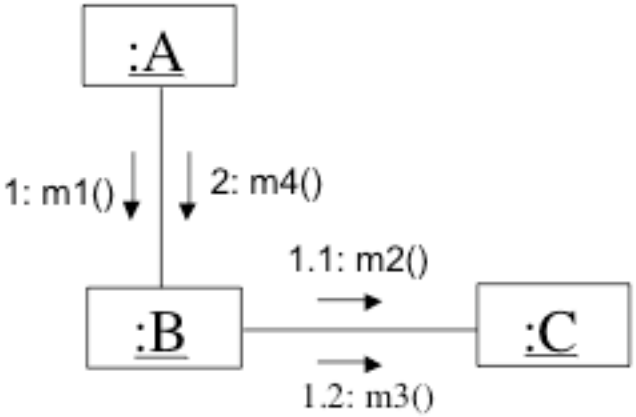
Message conditionnel



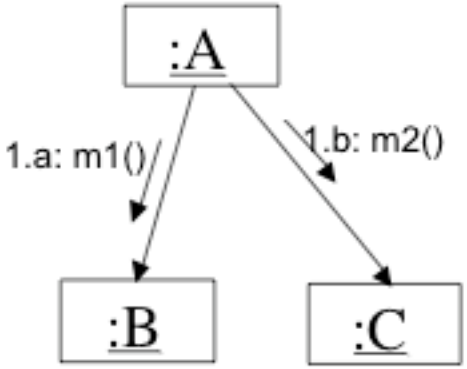
Message avec retour



Itération



Invocations synchrones



Concurrence

Diagrammes de collaboration et d'objets avec objecteering

Comment procéder ?

- Diagramme de collaboration

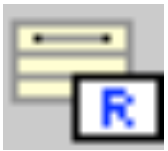
Attaché à une collaboration d'un cas d'utilisation

- Diagramme d'objet

Attaché à un package

Dans tous les cas : gestion des accès inter-packages

- Outils Objecteering



Rôle d'objet ou objet

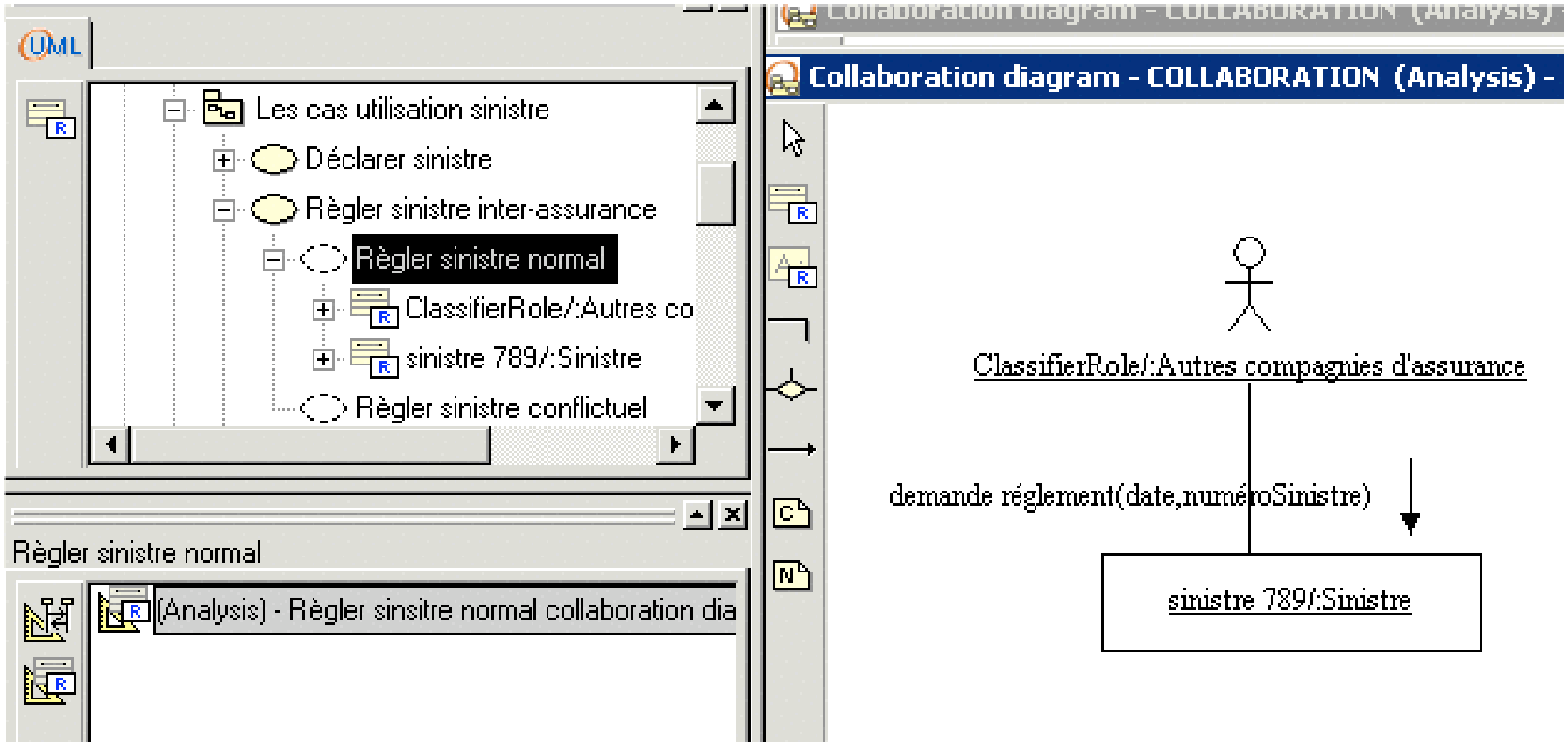
Attribut d'un objet

Lien entre objets

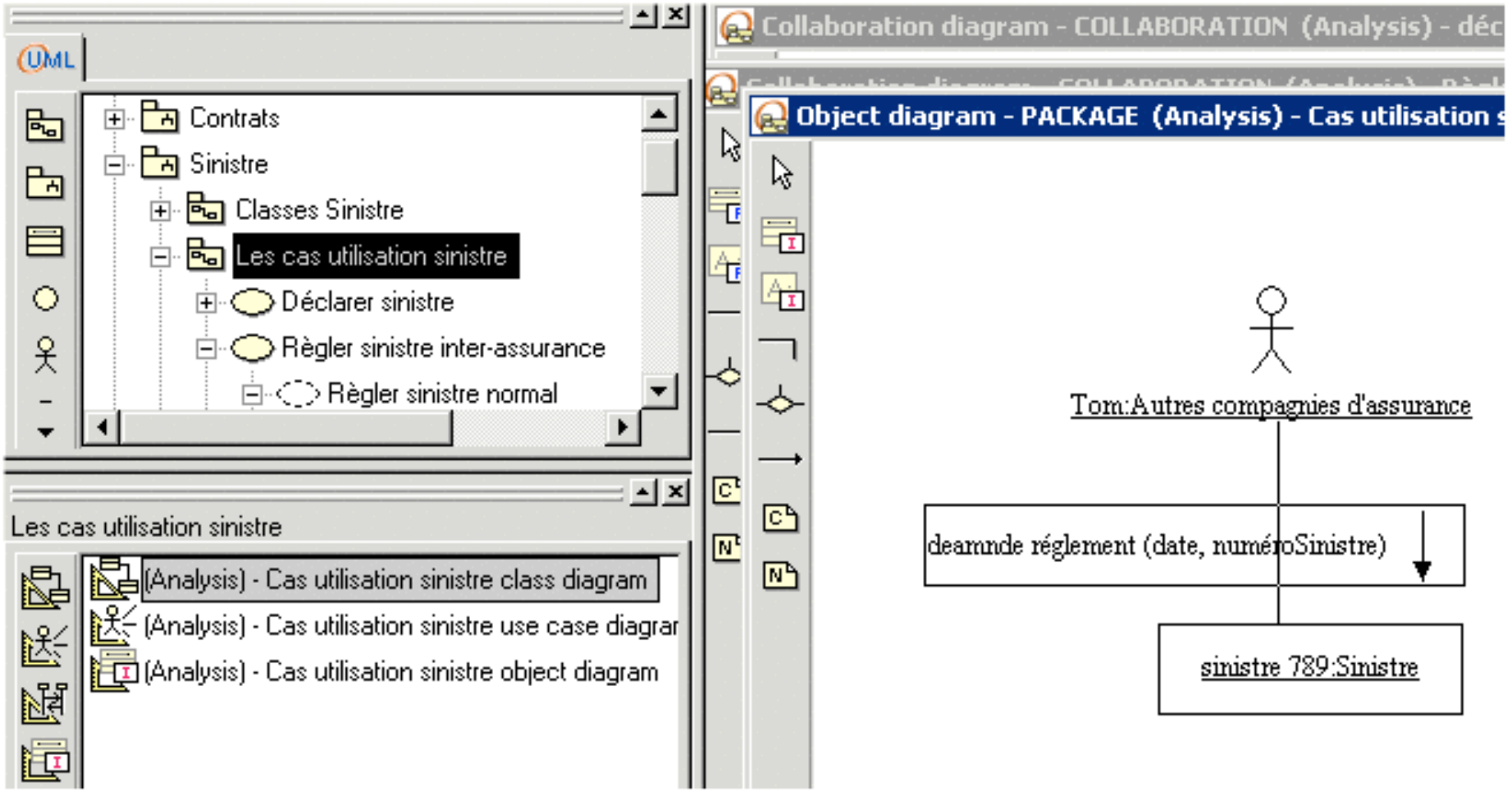
Message sur un lien

NB : idem avec « I » pour les diagrammes d'objets

Exemple diagramme collaboration Objecteering



Exemple diagramme objets Objecteering



Exemple Collaboration Assurance

- Déclaration de sinistre

Des diagrammes de cas d'utilisation aux diagrammes de classe

- LA question de base de la conception objet.
  - Comment faire pour « trouver » les classes modélisant le fonctionnement interne du logiciel?
- Un processus semi-formel
  - Partir du fonctionnement externe : les cas d'utilisation et leurs scénarios externes.
  - Pour chaque scénario d'un cas d'utilisation, détailler sur un exemple (en utilisant des rôles d'instances d'objets) le fonctionnement interne.
  - En généralisant les rôles instances d'objets et leurs communications, en déduire les classes et leurs relations.
- Choix d'une architecture : modèle organique.
  - Classes « métiers »
  - Classes d'interface
  - Classes d'« analyse » : utilitaires ou spécifiques au fonctionnement logiciel

Du scénario aux classes

Transformation très simple :

- Un objet de scénario => une classe
- Un message entre objets => une méthode de la classe correspondant à l'objet

- Reste à détailler
  - Objets : attributs, méthodes
  - Liens entre objets : généralisation, relation, agrégation, dépendance

Une autre façon de modéliser une collaboration : diagrammes de séquence

- Similaire au diagramme de collaboration
  - Une vue orientée temps versus orienté objet
  - Plus facile à lire sur la vie des objets
    - Création
    - Destruction
    - Activité
    - Parallélisme et déroulement du temps
  - Moins fine sur la composition des objets
    - Attributs

NB : le même formalisme que celui utilisé pour décrire l'interaction Acteur / Système du scénario

Exemple Assurance
Collaboration par diagramme de séquence

- Déclaration de sinistre

Exemple Assurance
Diagramme de classes

- Déclaration de sinistre

Exemple Assurance
Diagramme de collaboration avec classes

- Déclaration de sinistre