Liens entre objets
Relations entre classes

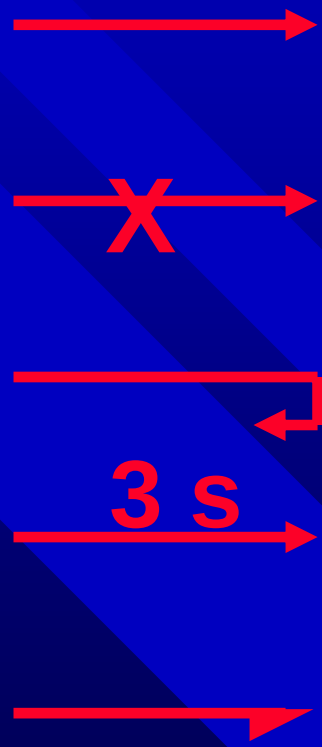
- Représentation des liens entre objets
- Diagrammes d'objets (instances et interactions)
- Passage du monde des objets au monde des classes
- Caractéristiques des relations entre classes

Diagrammes d'objets

- Un peu de structure, un peu de comportement
- Représentation de scénarii simples
- Représentation spatiale des objets
- Représentation des liens entre objets
- Limités à un petit nombre d'objets (comme toute représentation graphique)
- Les diagrammes qui ne montrent que de la structure sont appelés diagrammes d'instances

Différentes formes de synchronisation des messages

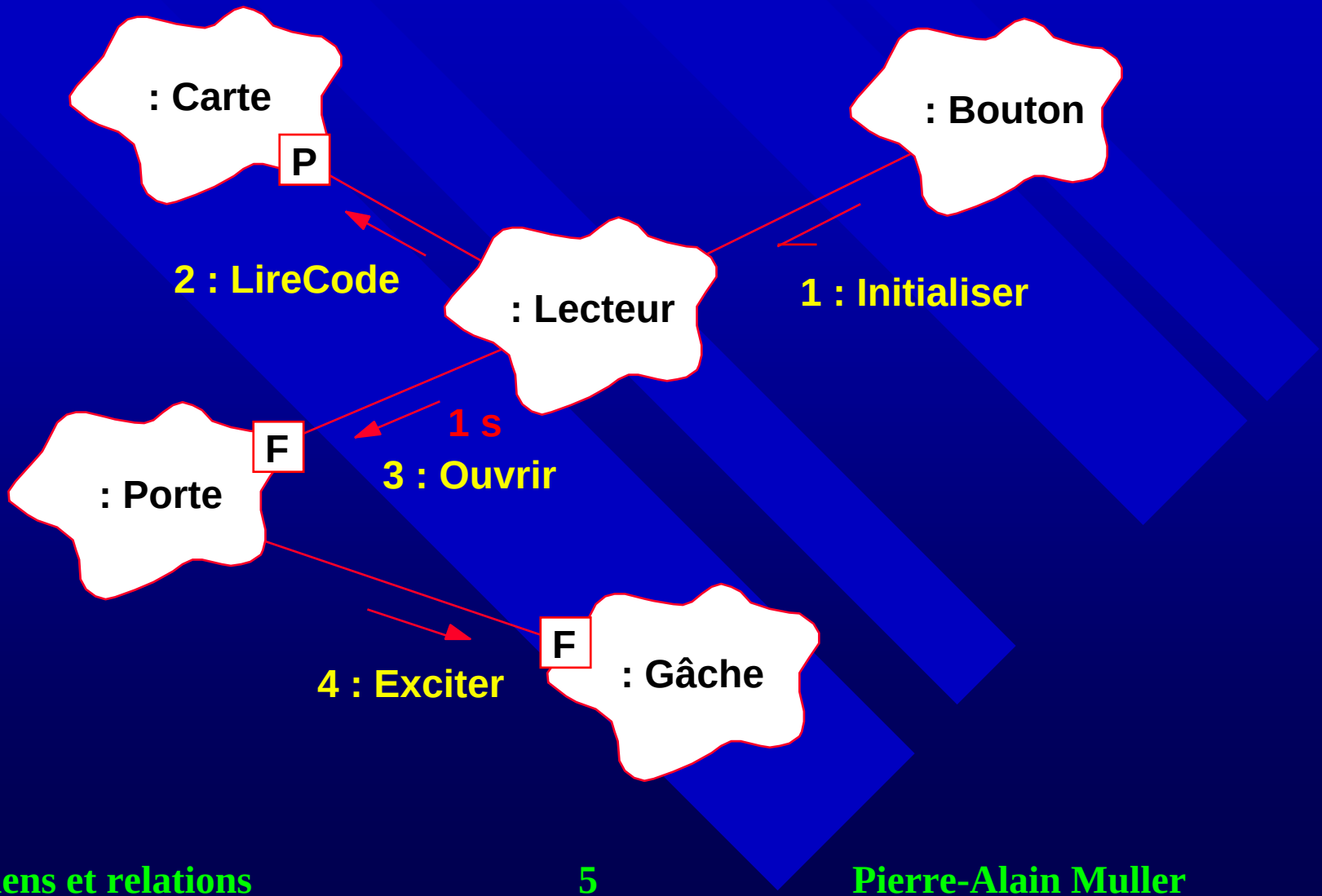
- Simple
- Synchrone
- Dérobante
- Minutée
- Asynchrone



Règles de visibilité entre objets

- Global G
- Paramètre P
- Champs F
- Local L

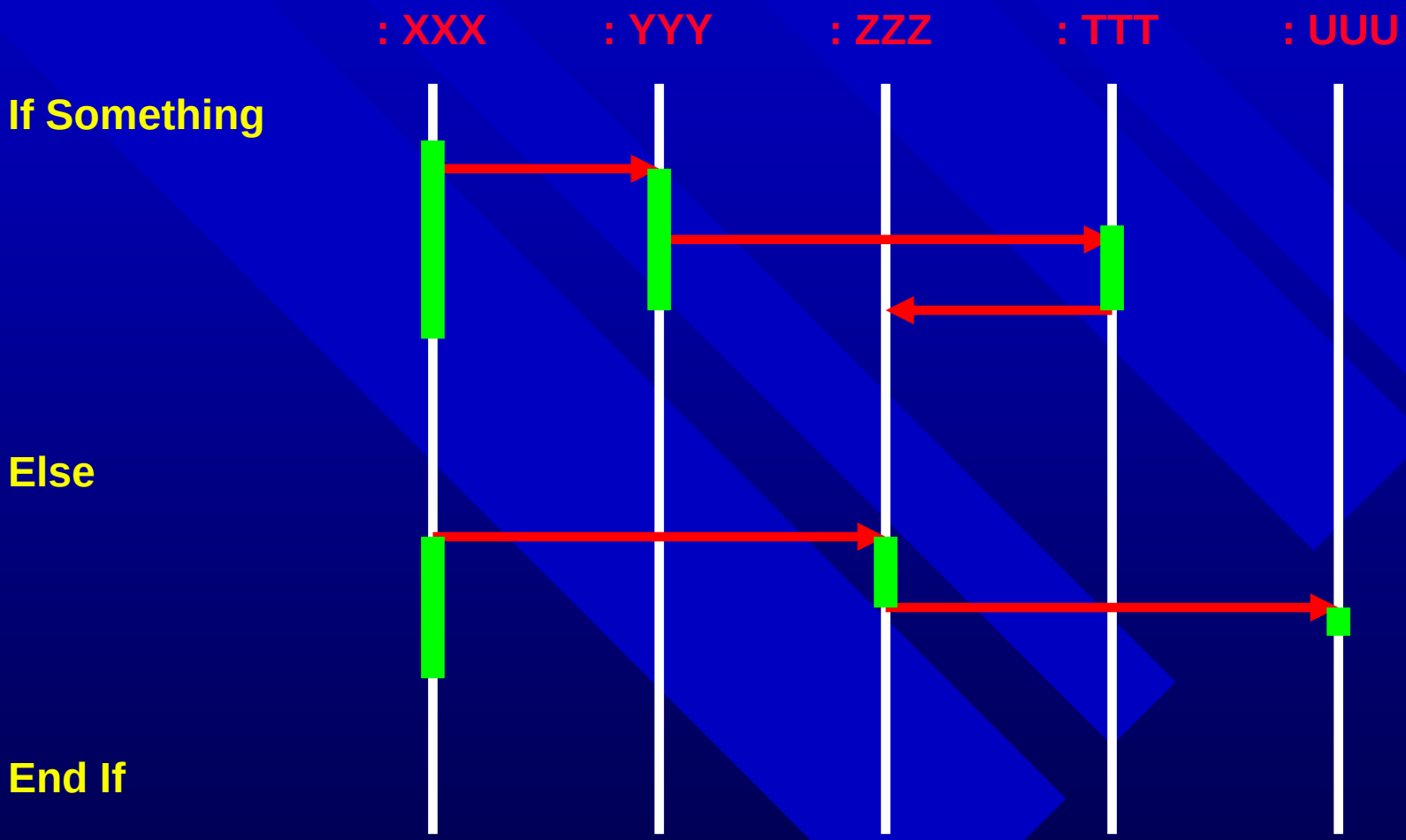
Exemple de diagramme d'objets



Diagrammes d’interactions

- Diagrammes complémentaires des diagrammes d’objets
- Pas de structure, beaucoup de comportement
- Représentation de scénarii plus complexes
- Qualifiables par du pseudo-code
- Prise en compte des alternatives

Exemple de diagramme d'interaction



Liens entre le monde des objets et le monde des classes

- Un diagramme d'objet montre la réalité a un instant donné
- Un diagramme de classe montre une abstraction de la réalité, valable d'un point de vue général
- Les classes permettent de factoriser ce qui est commun à un ensemble d'objets
- De même les relations entre classes permettent de factoriser les liens entre objets
- Chaque objet appartient de façon permanente à une classe

Correspondances entre diagrammes de classes et diagrammes d'objets

- Un lien entre deux objets indique que les deux objets se connaissent
- La direction du flot de contrôle permet de préciser l'orientation du lien
- Un lien entre deux objets de deux classes différentes implique une relation entre les deux classes
- Les relations entre classes expriment ce qui peut exister d'une façon générale
- Les liens entre objets expriment ce qui existe dans un cas particulier

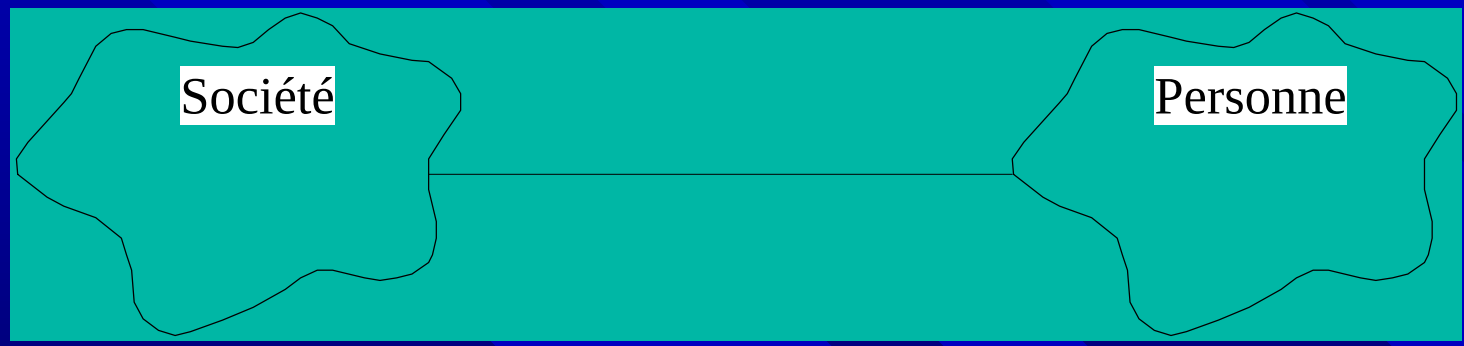
Les relations entre classes

- L’association
- L’aggrégation
- L’utilisation
- L’héritage
- L’instantiation de générique
- La métaclasse

La relation d'association

- Relation bidirectionnelle non raffinée, à faible contenu sémantique
- Exprime le fait que deux classes sont liées
- Employée uniquement en phase d'analyse, de façon transitoire
- Permet de faire progresser la réflexion
- Transformée plus tard en une autre relation

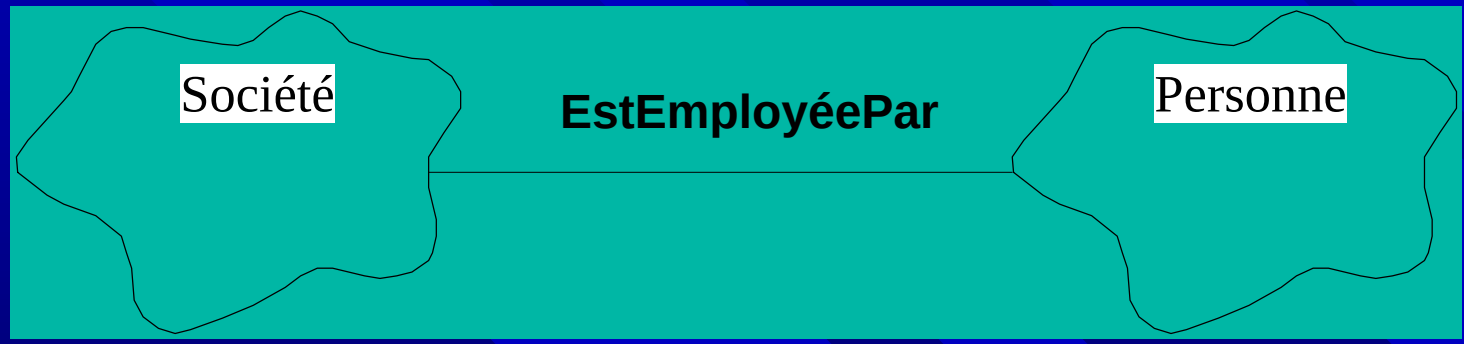
La relation d’association



Association nommée

- Information de clarification optionnelle
- Généralement un verbe conjugué, ou un participe passé
- L'information se lit de la gauche vers la droite
- Un peu en porte à faux avec les diagrammes de classes qui expriment surtout de la structure statique

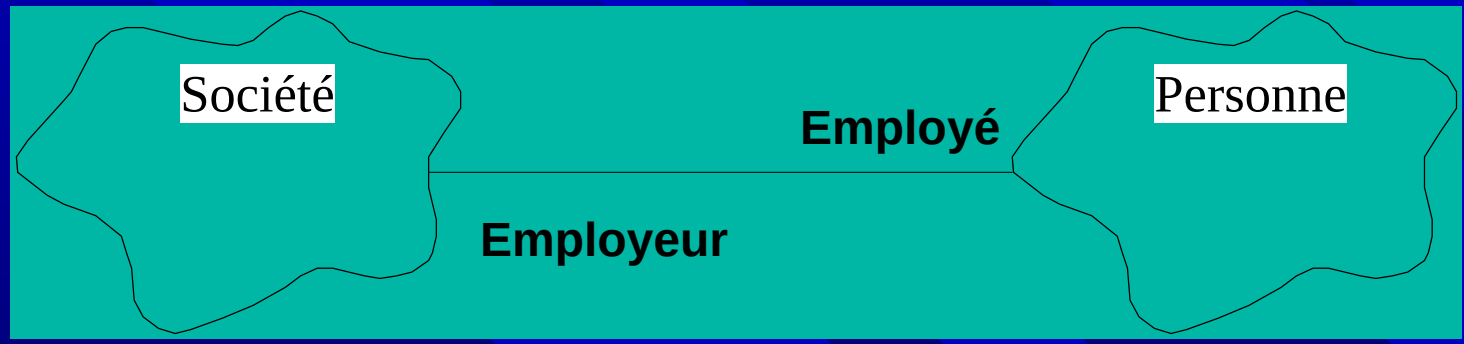
Association nommée



Association avec rôle

- Le rôle précise l'intérêt qu'une classe porte à une autre classe au travers d'une association
- Un rôle peut être spécifié à chaque extrémité de l'association
- Le rôle permet de faciliter la mutation de l'association vers une autre relation
- A recommander par rapport au simple nommage

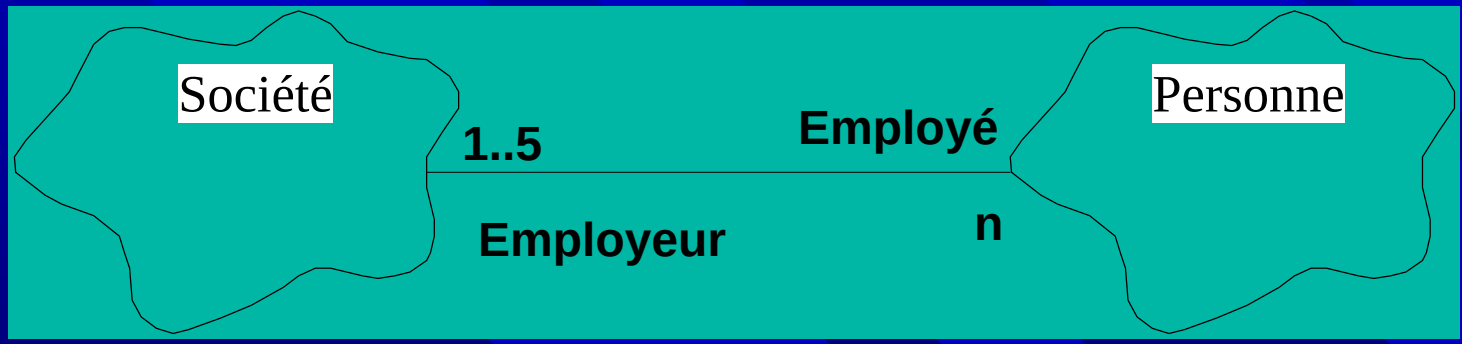
Association avec rôles



Cardinalité des relations

- La multiplicité des relations peut-être indiquée dans les diagrammes de classes
- Une information de cardinalité peut figurer à chaque extrémité des relations
- Valeurs conventionnelles de cardinalité
  - 1
  - N
  - 0 .. N
  - 1 .. N
  - M .. N

Association avec cardinalités

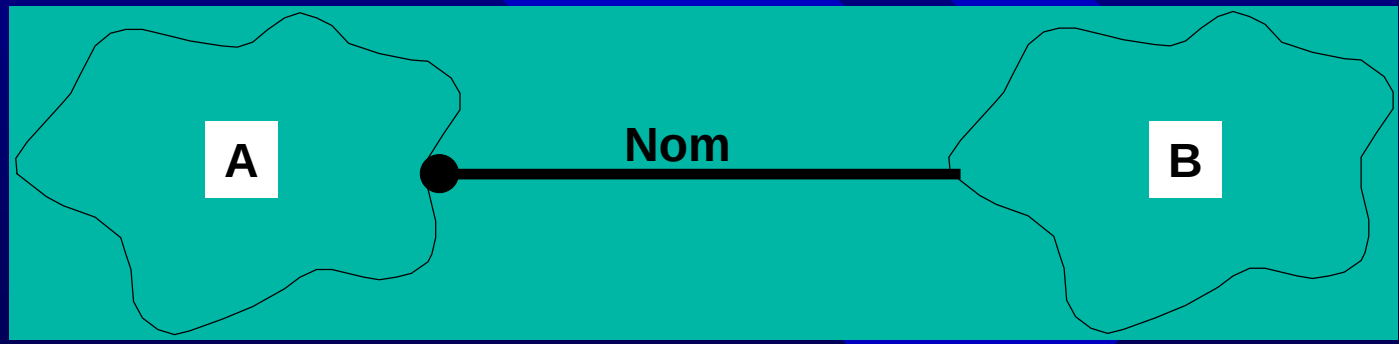


Cardinalité des relations (Suite)

- L’absence de cardinalité n’implique pas de valeurs par défaut
- La cardinalité peut également être exprimée au moyen de formules plus complexes
- Par défault il n’y a pas de corrélation entre les valeurs N dans un même diagramme

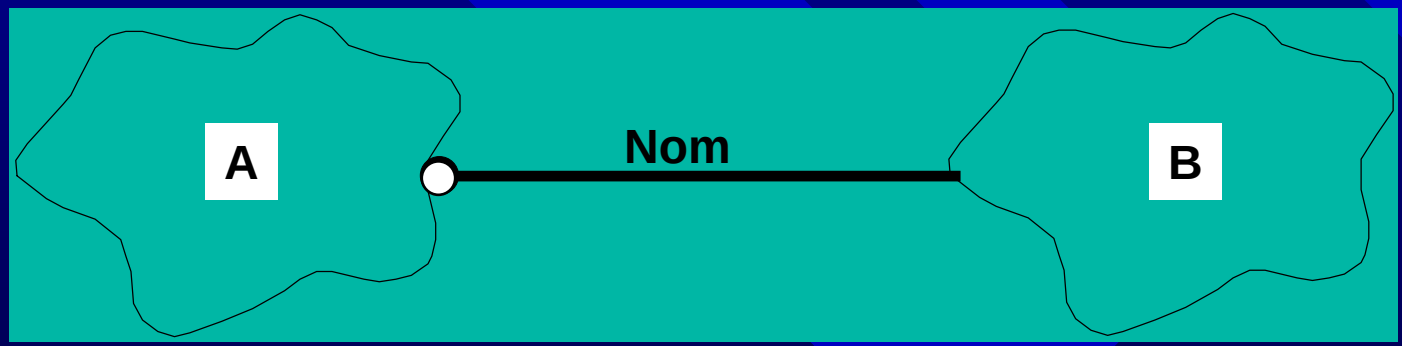
La relation d'aggrégation

- Egalement appelée relation *Has*
- Relation mono-directionnelle
- Exprime un couplage fort
- Exprime un groupement de composants ou de connaissances
- Dénote un des attributs de l'état des objets



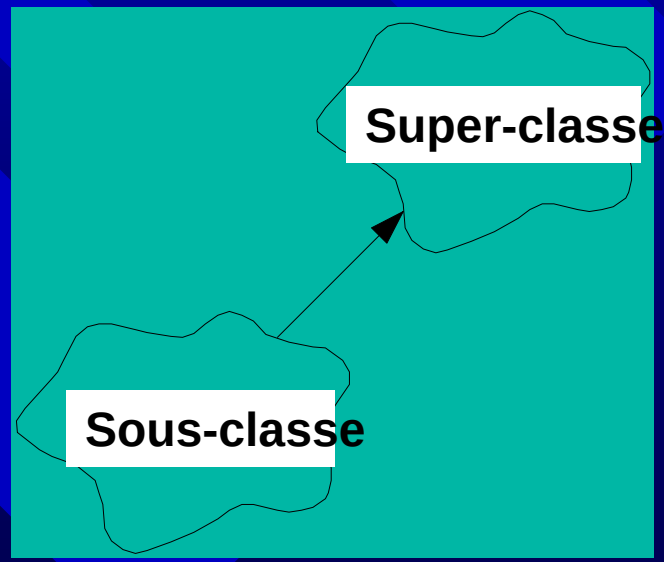
La relation d'utilisation

- Egalement appelée relation *Use*
- Relation mono-directionnelle
- Exprime un couplage faible
- Ne dénote aucun attribut de l'état des objets



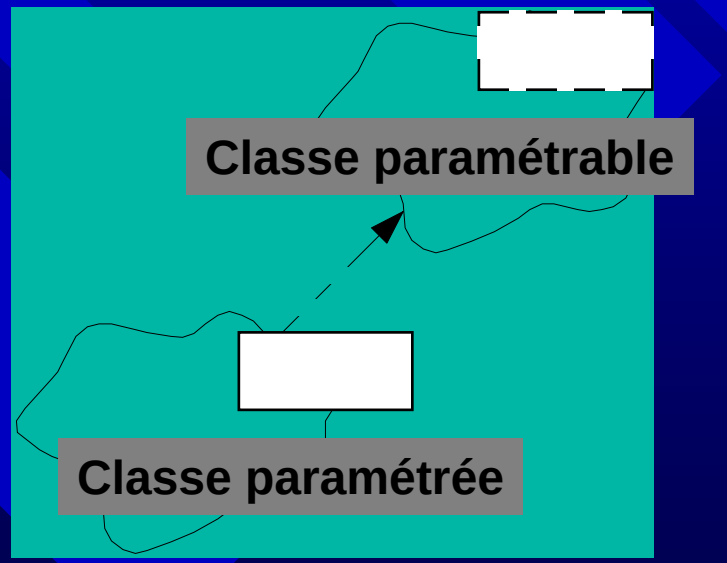
La relation d'héritage

- Exprime des relations de spécialisation ou de généralisation
- Factorisation de ce qui est commun à plusieurs classes
- Mécanismes d'abstraction sur des classes
- Héritage simple
- Héritage multiple



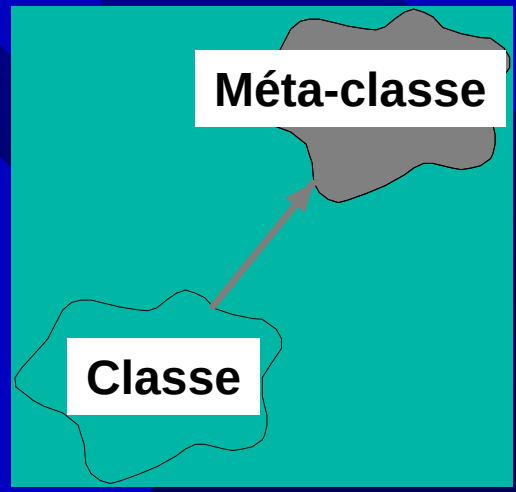
La relation d'instantiation de générique

- Utilisée uniquement en phase de conception
- Permet de construire une classe à partir d'un moule à classe
- Puissant mécanisme pour la réutilisation

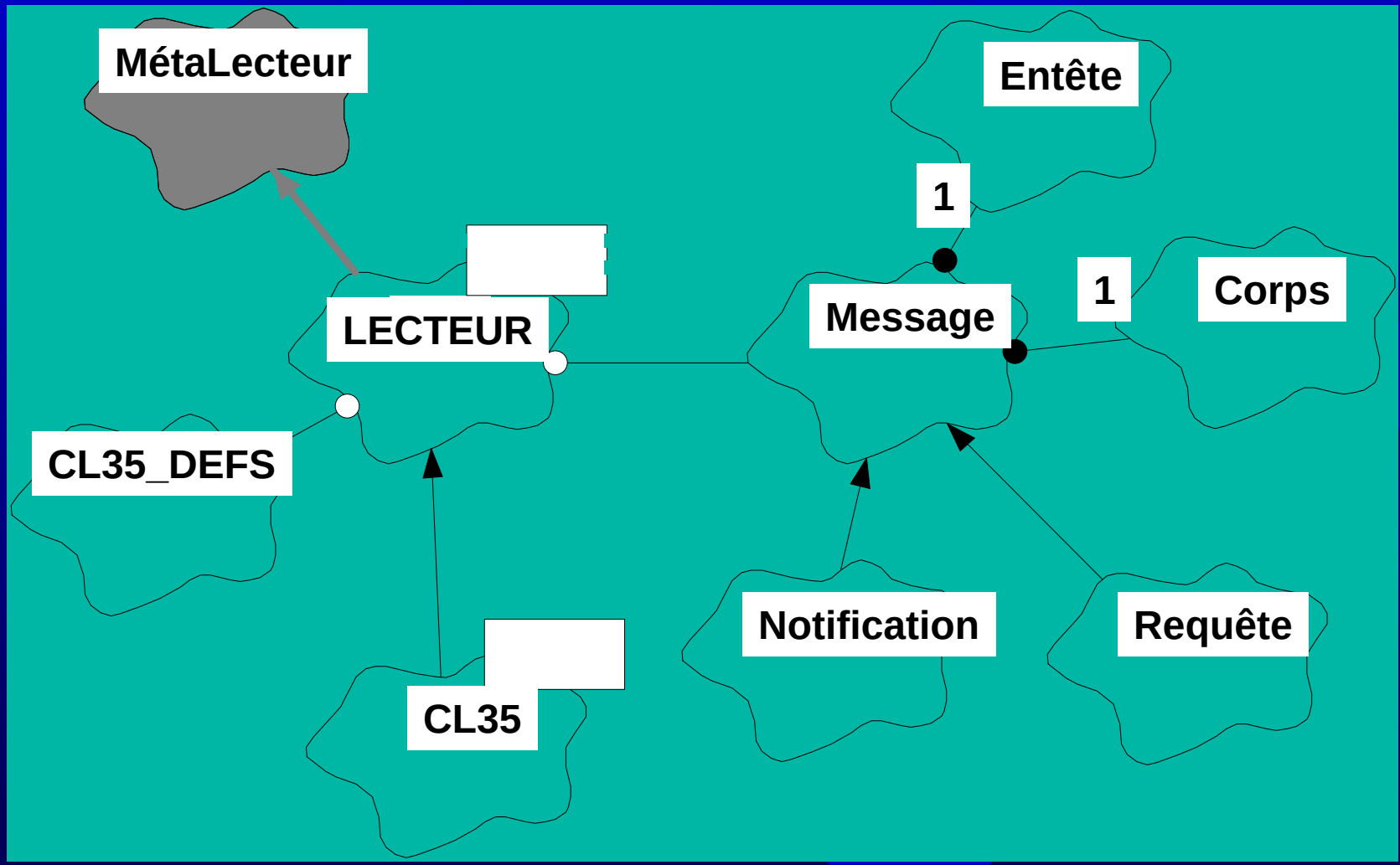


La relation de méta-classe

- Les classes elles-mêmes peuvent être vues commes des objets
- La méta-classe est la classe d'une classe
- La méta-classe contient l'état et le comportement qui appartiennent à la classe plutôt qu'aux objets
- Variables de classes et opérations de classe



Exemple de diagramme de classes



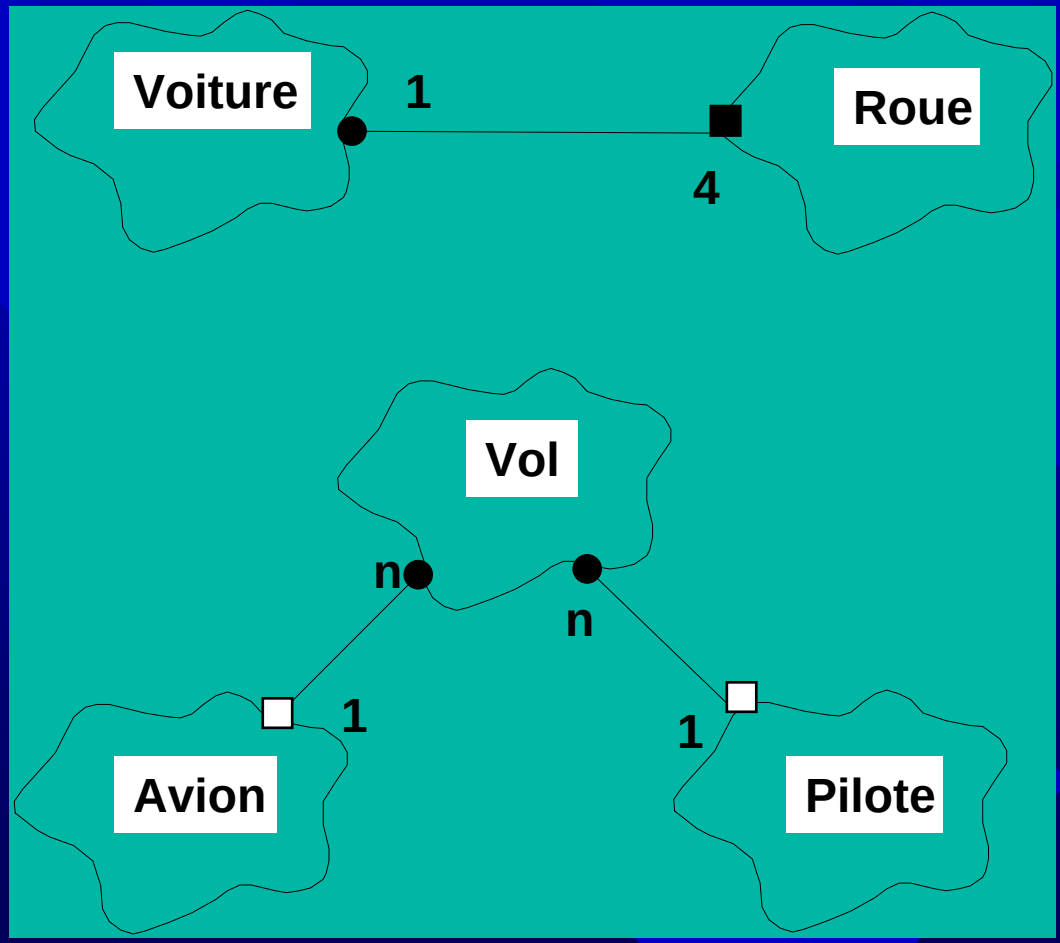
Règles de subordination : interprétation générale

- L'existence des objets d'une classe est subordonnée à l'existence des objets d'une autre classe
- Notion de propriété ou de non-propriété qui vient complémenter la notion d'aggrégation
- La classe propriétaire doit être informée de la disparition de la classe possédée
- Ne pas confondre avec la persistance
- Information indépendante des cardinalités inverses

Règles de subordination : interprétation restreinte

- La contenance par valeur ou par référence
- Cas particulier de l'interprétation générale
- Utile pour la génération automatique de code C++ par exemple
- Information dépendante des cardinalités inverses

Exemples de subordination



Règles de visibilité entre classes

- Précision du niveau d'encapsulation
- Par défaut les attributs sont visibles
- Quatre niveaux de visibilité
  - Public
  - Protégé
  - Privé
  - Réalisation