ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

02 Chapitre

Le paradigme objet

1I2AC1 : Génie logiciel et Conception orientée objet

Régis Clouard, ENSICAEN - GREYC

« N'importe quel programmeur peut écrire
du code que l'ordinateur comprend.
Les bons programmeurs écrivent du code
que les humains peuvent comprendre. »
Martin Fowler

Génie logiciel

- Une méthode pour organiser le travail
  - Agile : itérative et incrémentale
- Un paradigme
  - Conception orientée objet
- Un formalisme
  - UML

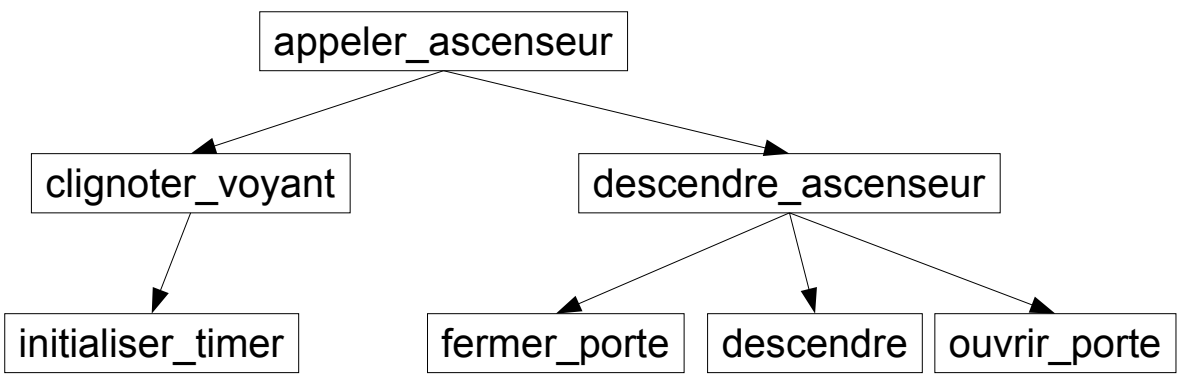
Plan du chapitre

1

Le paradigme objet

Paradigme procédural vs Paradigme objet

- Un changement de point de vue sur le problème.
- **Procédural**
  - Point de vue sur les opérations.
  - Les données sont inertes.
- **Objet**
  - Point de vue sur les données.
  - Les données sont animés.



Graphe d'appel

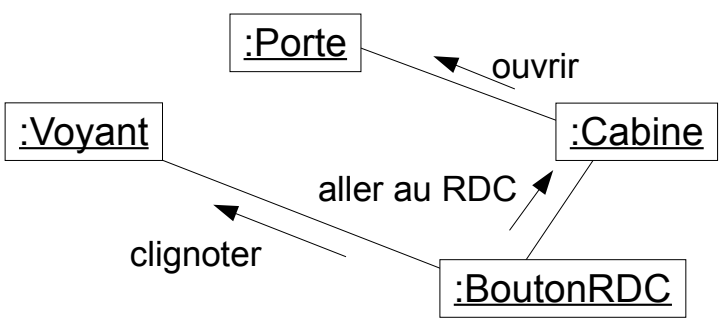


Diagramme de collaboration

Paradigme procédural vs Paradigme objet

Exemple

Comptage des étudiants présents en cours

Conception procédurale vs conception objet

- **Algorithmique**
  - Question à résoudre
    Que veut-on faire ?
  - Solution : la séquence d'appels de procédures

- **Modélisation**
  - Question à résoudre
    De quoi parle t-on ?
  - Solution : les objets avec les bons services

« Si je disposais d'un chapeau magique, quel type de données voudrais-je voir sortir du chapeau pour m'aider à résoudre le problème ? »

Exemple

Guichet automatique de billets
(GAB ou *ATM*)

Conception procédurale vs conception objet

- **Algorithmique**
  - Avantages
    - Proche de la machine
    - Calculs de complexité
  - Limites
    - Inaccessible aux clients
    - Inadaptée aux gros logiciels
    - Enchaînement figé
    - Maintenance et réutilisabilité compliquée

- **Modélisation**
  - Avantages
    - Adapter aux gros logiciels
    - Implémentation repoussée le plus tard possible
    - Maintenance et réutilisabilité facilitées
  - Limites
    - Vision fractionnée du logiciel
    - Calculs de complexité difficiles

Langages de programmation

Langages de programmation objet

- Foisonnement de langages de programmation orientés objet.
  - Simula (Ole-Johan Dahl & Kristen Nygaard, 1963)
  - SmallTalk (Alan Kay, 1972), Eiffel (Bertrand Meyer ,1986)
  - **C++** (Bjarne Stroustrup, 1983), Objective C, D
  - **Java** (Sun MicroSystem, 1991), C# – basés sur UML
  - **Python**, Perl, Clos
  - Ruby, **Scala**

Programmation procédurale / Programmation orientée objet

- Différence
  - Ne concerne que quelques mots clés
    - P. ex C ↔ C++
  - Mais ce sont deux paradigmes différentes
- Conséquence :
  - Le paradigme objet ne s'apprend pas par le langage

Concepts de la conception objet

- La conception orientée objet s'appuie sur 5 concepts :
  1) Objet
  2) Classe
  3) Association
  4) Héritage
  5) Polymorphisme

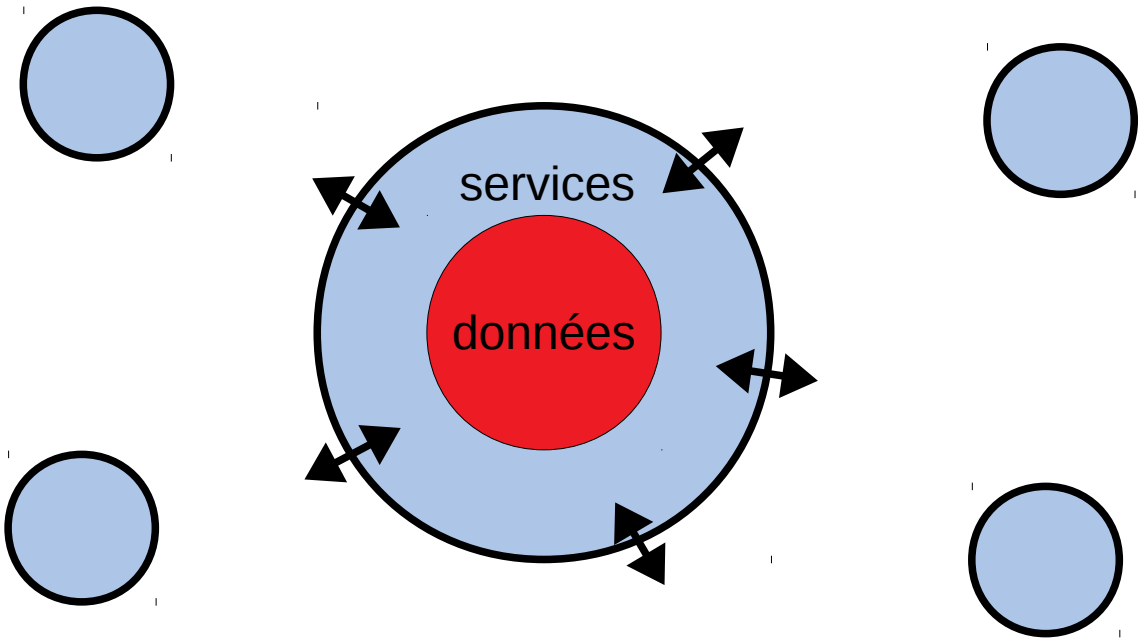
Plan du chapitre

1
Le paradigme objet

2
Les objets

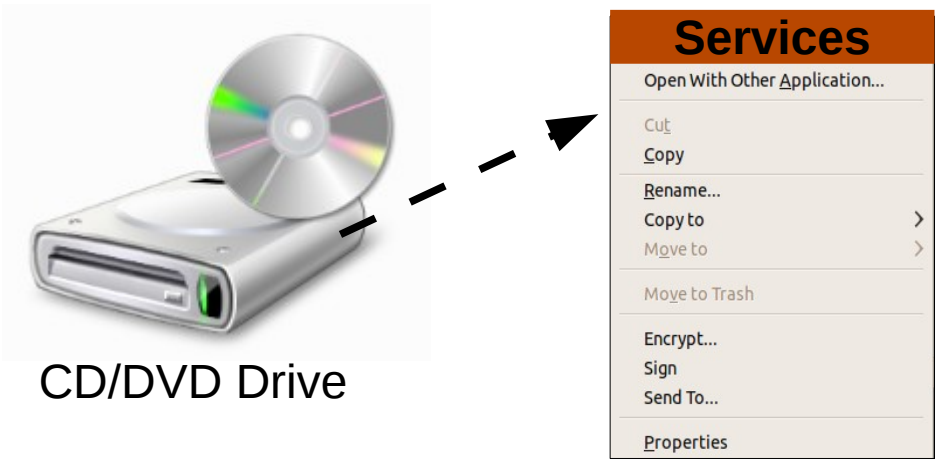
Encapsulation

- Objet : Les données d'un objet sont cachées et protégées de l'extérieur



Encapsulation

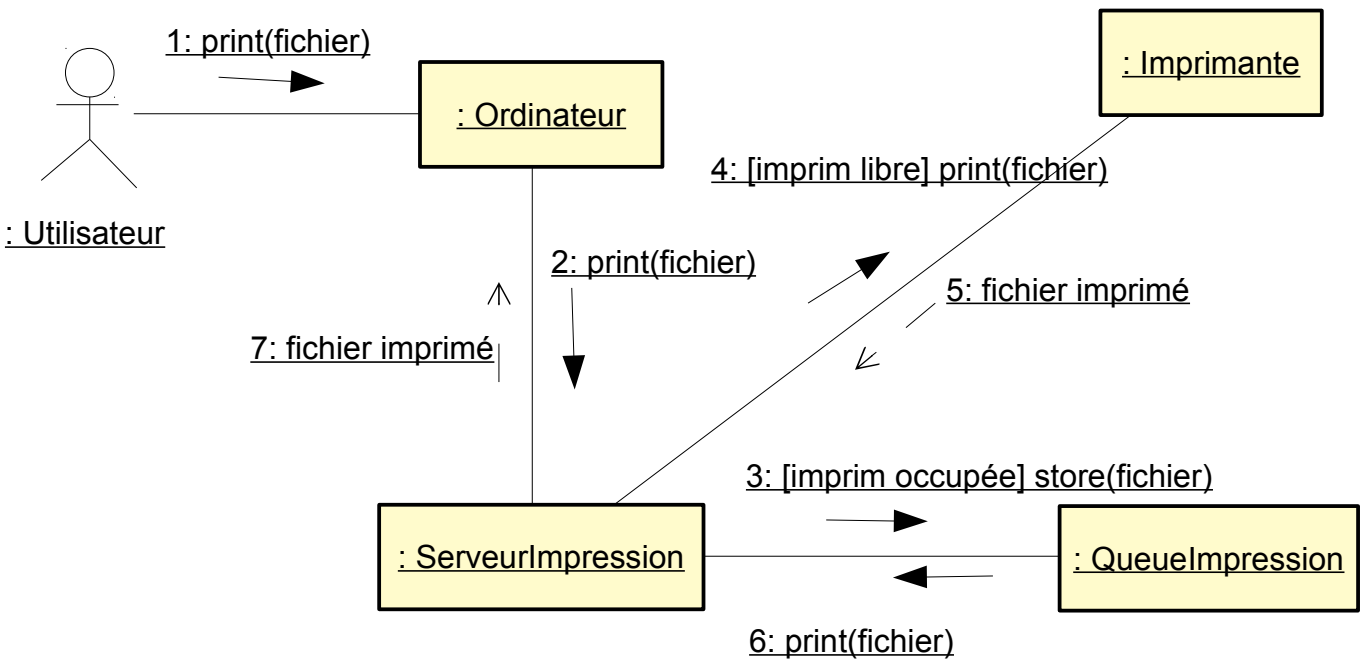
- objet = **fournisseur de services**



CD/DVD Drive

Communication entre objets

- Un objet ne doit pas être omniscient mais au contraire spécialisé
  - Sinon cela revient à faire de la conception procédurale
- Il doit donc faire appel aux services d'autres objets



Plan du chapitre

1
Le paradigme objet

2
Les objets

3
Les classes

Classe

- Représente un concept du domaine.
- Génératrice d’objets

Exemple
Voiture

Plan du chapitre

1
Le paradigme objet

2
Les objets

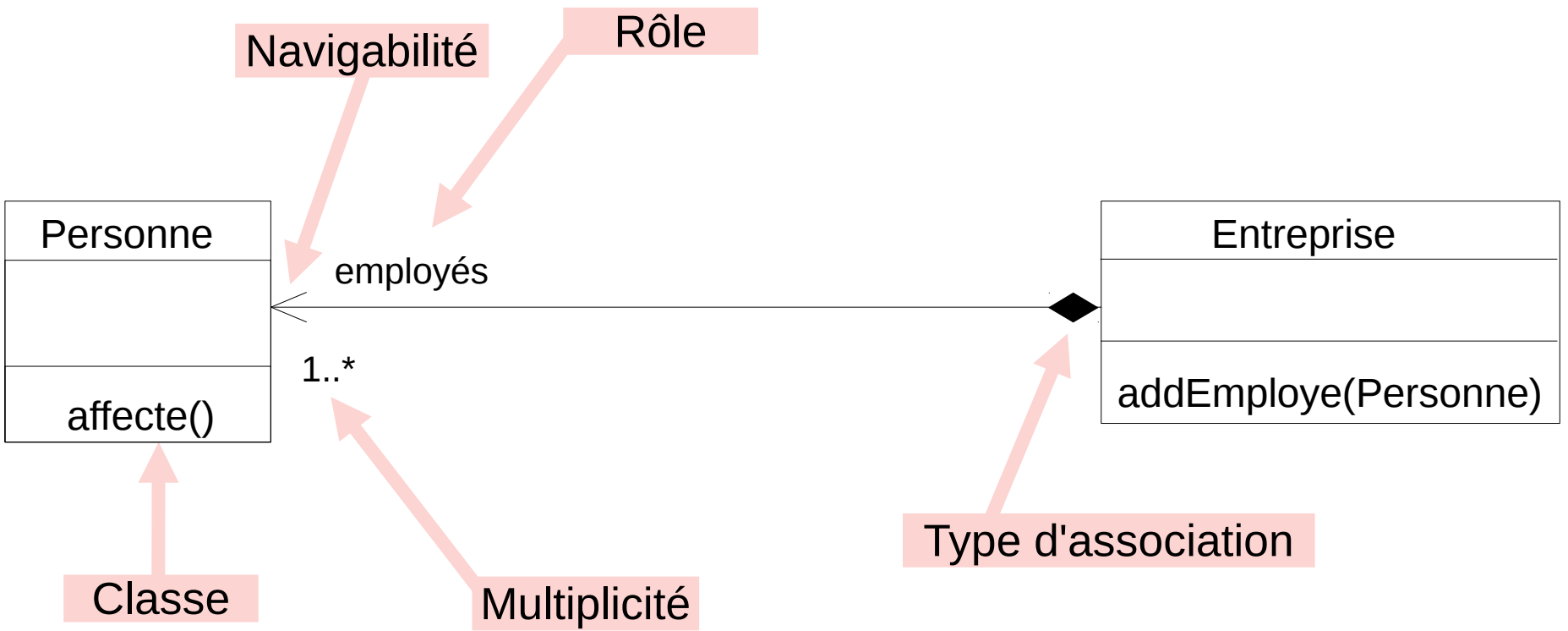
3
Les classes

4
Associations entre classes

Association

- Définition
- Navigabilité
- Rôle
- Multiplicité
- Types
  - Standard
  - Agrégation
  - Composition
  - Dépendance

Résumé de la notation



Plan du chapitre

1
Le paradigme objet

2
Les objets

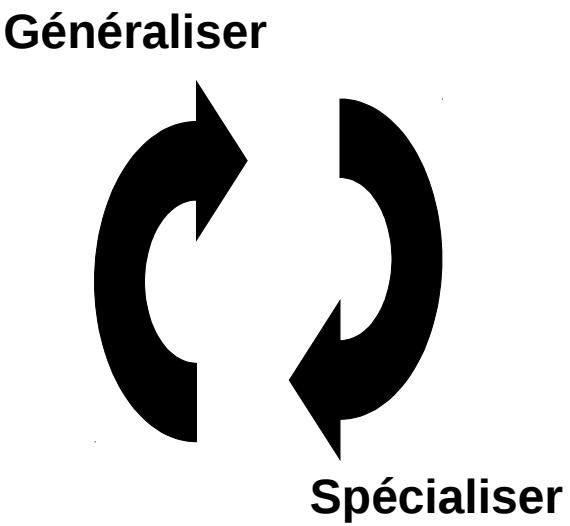
3
Les classes

4
Associations entre classes

5
Héritage et polymorphisme

Généralisation et spécialisation

- Deux points de vue menant au mécanisme d'**héritage**.



Héritage

- Définition
- Transtypage
- Visibilité

Héritage

- Portée des noms
- Polymorphisme
- Interface & classes abstraites
- Héritage multiple

Que retenir de ce chapitre ?

- Le paradigme objet définit plusieurs concepts importants :
  - Classe et objet.
  - Encapsulation.
  - Relations.
    - Association.
      - Standard, Agrégation, Composition, Dépendance.
    - Héritage et polymorphisme.
  - Classe et Type.

Que retenir de ce chapitre ?

- Dans l'utilisation de ces concepts, le développeur doit respecter deux principes fondamentaux :
  - Restreindre le plus possible la visibilité des membres pour respecter le principe d'encapsulation et la sécurité.
    - Ainsi les attributs sont TOUJOURS privés.
  - Exprimer le plus de choses statiquement pour être vérifiables par le compilateur.