ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

02
Chapitre

Un paradigme : conception orientée 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

Le génie logiciel aujourd'hui

- Rappel : le génie logiciel est défini par 3 composants :
  - Une méthode pour organiser le travail
    - Méthode agile (SCRUM, Extreme Programming)
  - Un paradigme
    - Conception orientée objet (Java)
  - Un formalisme
    - UML (Unified Modeling Language)

Plan du chapitre

1

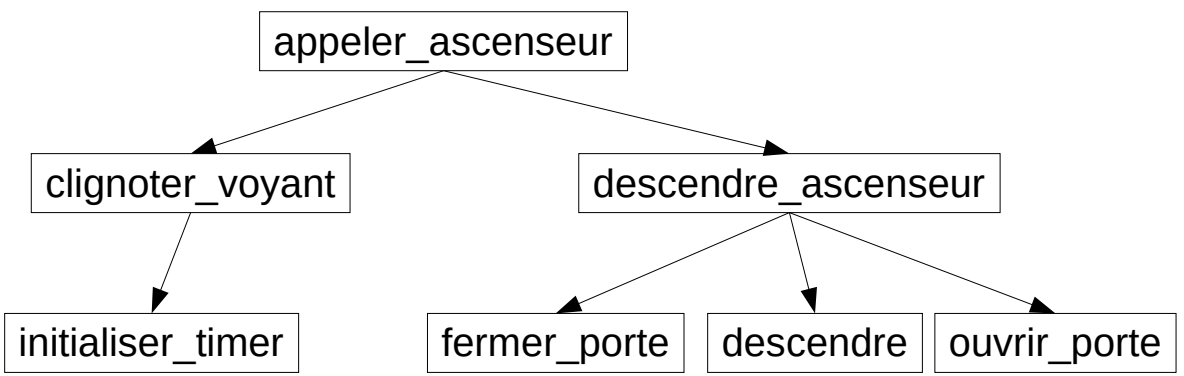
Le paradigme objet

Paradigmes de programmation

- Donnez quelques paradigmes de programmation

Paradigme procédural vs Paradigme objet

- Deux points de vue sur le problème → Duaux
- **Procédural**
  - Point de vue sur les opérations
  - Les données sont inertes
- **Objet**
  - Point de vue sur les données
  - Les données sont animés



Graphe d'appels

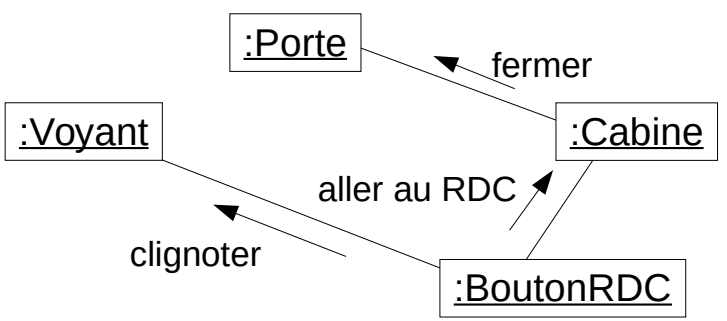


Diagramme de collaboration

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

Conception orientée objet (COO)

Conception orientée objet :

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

Conception orientée objet

Exemple

Guichet automatique de billets
(GAB ou *ATM*)

Conception procédurale vs Conception objet

- **Algorithmique**
  - Avantages
    - Proche de la machine
  - Limites
    - Inaccessible aux clients
    - Inadaptée aux gros logiciels : complexité
- **Modélisation**
  - Avantages
    - Adapter aux gros logiciels : approche cartésienne de la conception
    - Implémentation repoussée le plus tard possible
    - Accessible aux clients
  - Limites
    - Vision fractionnée du logiciel

Programmation procédurale / Programmation orientée objet

- La différence ne concerne que quelques mots clés
  - 6 mots clés suffisent pour passer du C au Java :
    `class, extends, implements, interface, new, public`
  - Mais ce sont deux paradigmes différents
- 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 et encapsulation
  2) Classe
  3) Association
  4) Héritage
  5) Polymorphisme

Plan du chapitre

1
Le paradigme objet

2
Les objets (encapsulation)

Notion d'objet

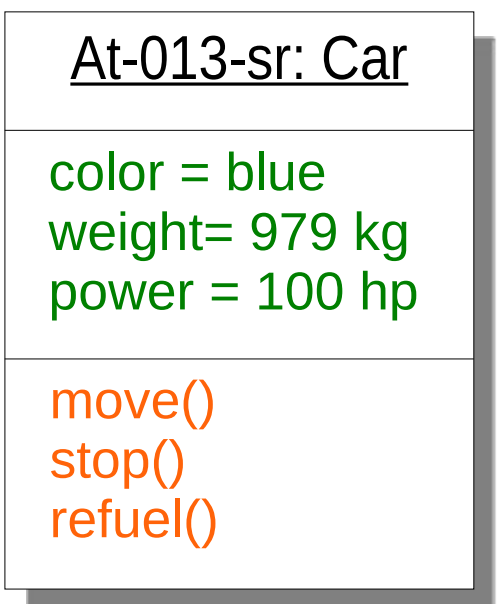
- Physiquement
  - **Objet = structure en C incluant des pointeurs sur des procédures**
- Par exemple une voiture

```
typedef struct s_car {
    int color;
    int weight;
    int power;

    void (*move)();
    void (*stop)();
    void (*refuel)(int);
} Car;

Car at_01_sr;
at_01_sr.weight = 979;
at_01_sr.refuel(10);
```


```
Car at_01_sr;
at_01_sr.weight = 979;
at_01_sr.refuel(10);
```

Notion d'objet

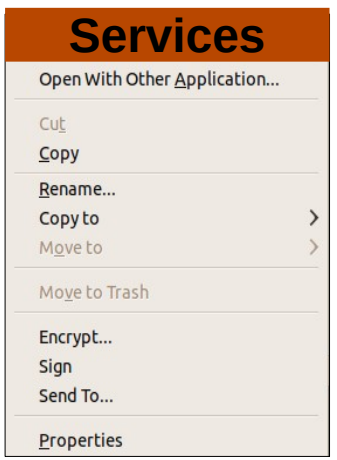
- Conceptuellement
  - **Objet = propriétés + services**
  - **Propriété (Attribut) : donnée** membre
    - Possédant une valeur
    - Évoluant au cours du temps
  - **Service (Méthode) : procédure** attachée à l'objet
    - Utilisant potentiellement les données membres pour fonctionner
    - Déclenchée par appel explicite à partir de l'objet



Encapsulation

- objet = **fournisseur de services**
  ≠ structure de données

CD/DVD Drive



Encapsulation

- Exemple : chaîne de caractères
- En C
  - Il faut connaître la représentation physique pour l'utiliser
- En Java
  - Nul besoin de connaître la représentation physique. Il suffit d'utiliser les services

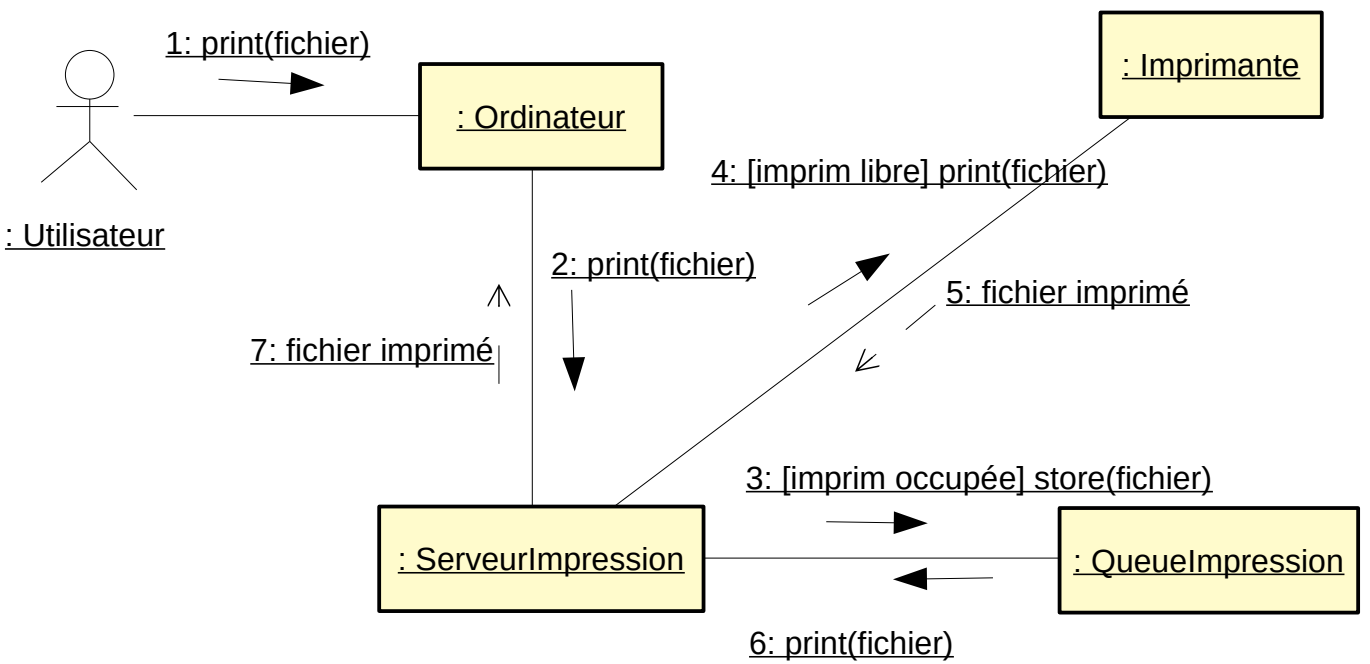
Encapsulation

- Les attributs ne sont pas une préoccupation de la conception mais de programmation
- Un attribut n'existe que parce qu'un service en a besoin

- **Ne me parlez plus d'attributs (sauf à ma demande)**

Association

- 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 qu'il connaît par la liste de ses associations



Plan du chapitre

1
Le paradigme objet

2
Les objets

3
Les classes

Classe

- Représente un concept du domaine
- Génératrice d’objets
- Nom
- Casse
- Représentation UML

Les attributs

- Nom
- Casse
- Représentation UML

Les méthodes

- Nom
- Casse
- Représentation UML

Implémentation en Java

Encapsulation

- Quelle différence entre :
  - `Car at_01_sr;`
  - `at_01_sr.color = "blue";`
  - `at_01_sr.setColor("blue");`

→ Voilà pourquoi il ne faut pas parler d'attribut au moment de la conception. Les attributs ne sont qu'une préoccupation de programmation