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

Plan du chapitre

1

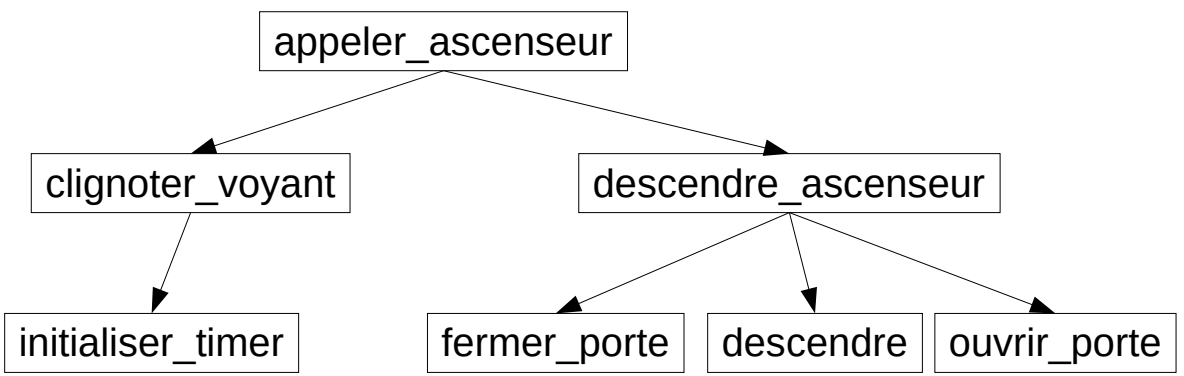
Le paradigme objet

Paradigmes de programmation

- Les 5 principaux paradigmes de programmation
  1) Assemblage : assembleur
  2) Procédural : **C** (*le solfège de l'informaticien*)
  3) Fonctionnel : **Scala**, Haskel, Clojure, Clos
  4) Objet : **Java** (1991), **C++** (1983), C#, **Javascript**, Objective C, D, ***PHP***, ***Python***, Rust, Ruby, ***Dart***
  5) Logique : Prolog
- Autres :
  - De script : **Shell**
  - Pile : Forth
  - Concurrence : Ocam, C//
  - etc

Paradigme procédural vs Paradigme objet

- Deux points de vue sur le problème → Points de vue duaux
- **Procédural**
  - Point de vue sur les opérations
  - Les données sont inertes
- **Objet**
  - Point de vue sur les données
  - Les données sont animées



Graphe d'appels

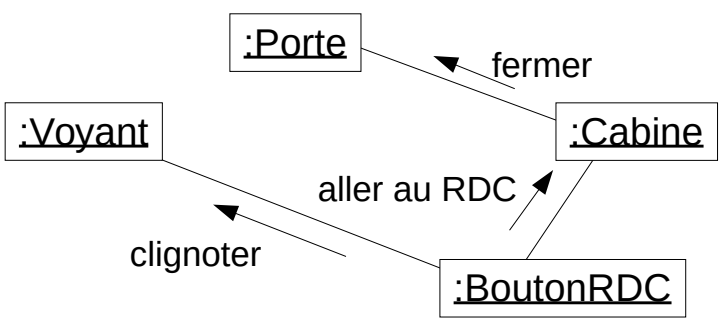


Diagramme de collaboration

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 ?**
  - Réponse : le graphe d'appels des procédures
- **Modélisation**
  - Question à résoudre : **De quoi parle t-on ?**
  - Réponse : la liste des 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 représentation machine
  - Limites
    - Inaccessible aux clients
    - Inadaptée à la complexité des gros logiciels
- **Modélisation**
  - Avantages
    - Adapter aux gros logiciels : approche cartésienne de la conception
    - Implémentation repoussée le plus tard possible
    - Accessible aux clients
  - Limite
    - Vision fractionnée du logiciel

Programmation procédurale vs 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 principe d’encapsulation
  2) Classe
  3) Associations
  4) Héritage
  5) Polymorphisme

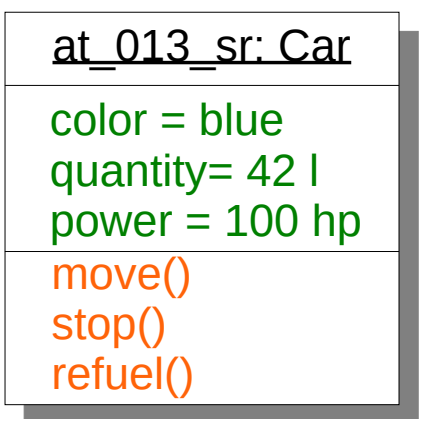
Plan du chapitre

1
Le paradigme objet

2
Les objets et le principe d’encapsulation

Notion d'objet

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

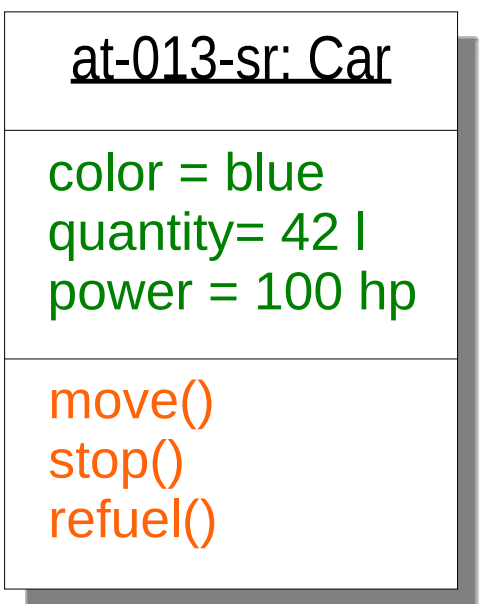



```
typedef struct s_car {
    int color;
    int quantity;
    int power;

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

```
Car at_01_sr;

at_01_sr.power = 110;
at_01_sr.refuel(10);
```

Notion d'objet

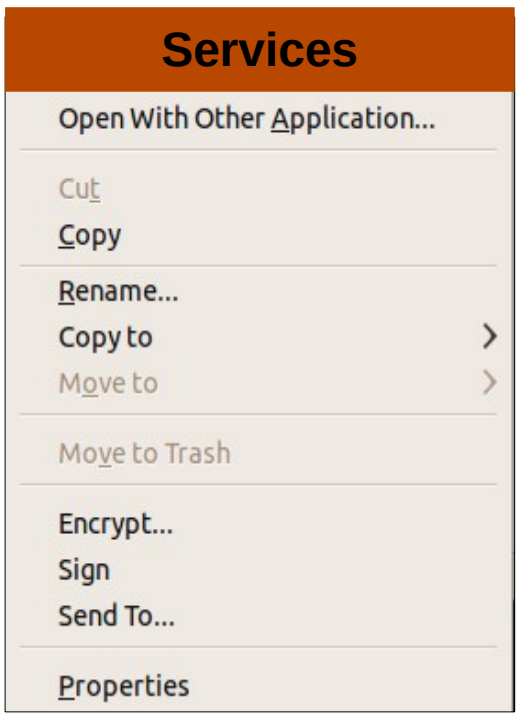
- Conceptuellement
  - **Objet = propriétés + services**
  - **Propriété (Attribut) : donnée** membre
    - Possédant une valeur
    - Pouvant évoluer au cours du temps
  - **Service (Méthode) : procédure** membre attachée à l'objet
    - Utilisant potentiellement les données membres pour fonctionner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



Principe d'encapsulation

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

CD/DVD Drive



Principe d’encapsulation

- Exemple : parcours d’une chaîne de caractères pour lui appliquer un traitement
- En C
- En Java

Principe d’encapsulation

- Éprouvez la différence essentielle entre les deux types d’instruction

```
1)at_01_sr.color = "blue"; color = at_01_sr.color;

2)at_01_sr.setColor("blue"); at_01_sr.getColor();
```

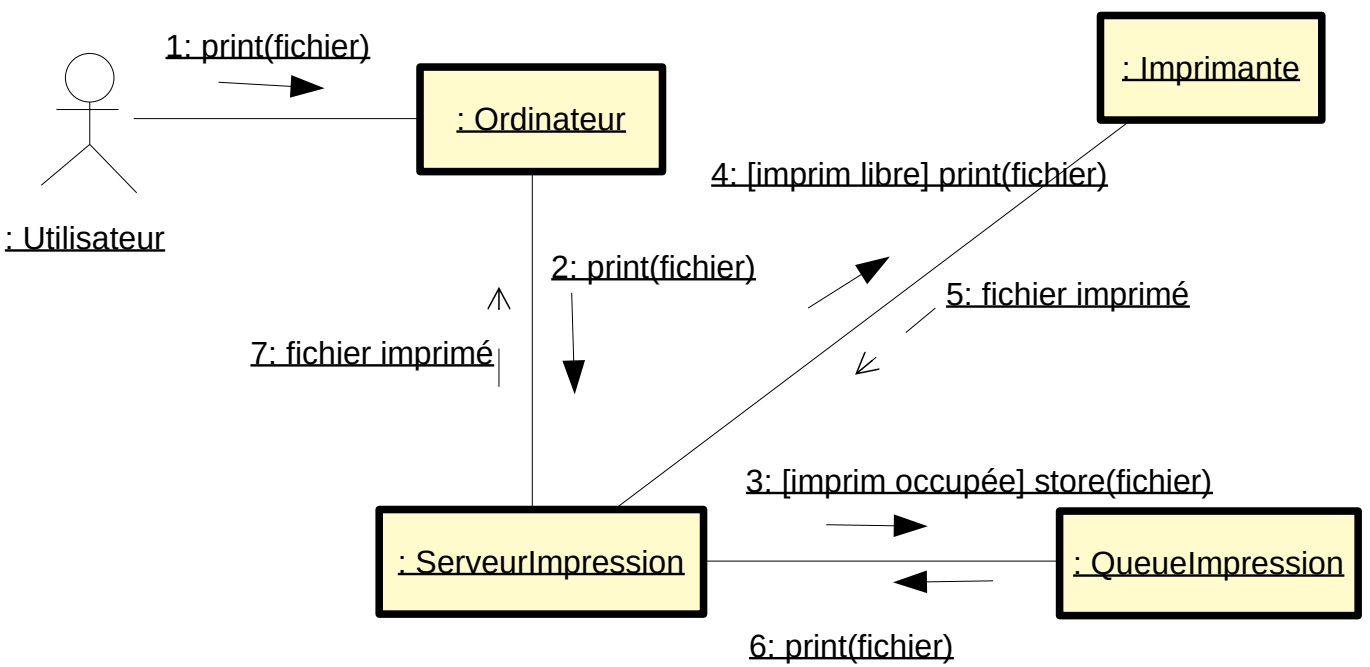
Principe d’encapsulation

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

En conception, 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

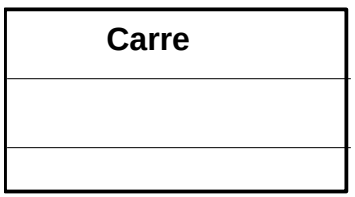
1
Le paradigme objet

2
Les objets et le principe d’encapsulation)

3
Les classes

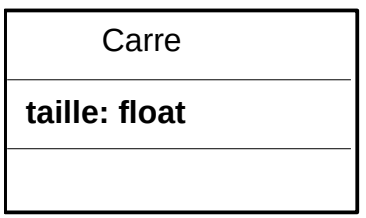
Classe

- Intention
  - Représente un concept du domaine
  - Génératrice d'objets
- Nom : substantif **au singulier**
- Casse : PascalCase
- Représentation UML



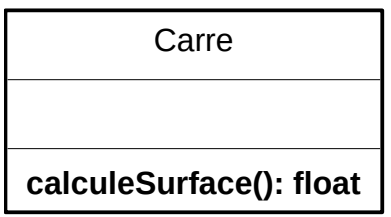
Attribut / Propriété

- Intention
  - Stocke une propriété de l'objet
- Nom : substantif
- Type : **primitifs** ou assimilés
- Casse : camelCase
- Représentation UML



Méthode / Service

- Intention
  - Propose un service
- Nom : verbe
- Casse : camelCase
- Représentation UML



Implémentation en Java

Exemple

Cas de Voiture