Un autre problème encore…

- On effectue une opération de maintenance évolutive d'une application en production
  - Site Web de publication d'annonces de vente de véhicules d'occasion
    - Trois catégories d'annonces selon le prix du véhicule : <5K€, entre 5K€ et 10K€, >10K€
      - Publier l'annonce : publier()
      - Facturer (à l'annonceur) selon la catégorie : facturer()
  - On veut permettre l'ajout d'options (payantes)
    - Mettre un fond jaune
    - Faire apparaître le véhicule en haut de liste
    - Ajouter une ou plusieurs photos
    - …
  - La (ou les) classe(s) Annonce existe déjà…
    - Et ne doit pas être modifiée !

Le modèle « Décorateur » (1/6)

- Décorateur
  - Pattern structurel de niveau objet
- Alias
  - Enveloppe
- Intention
  - Permet de remplacer un objet de base par un autre objet (avec conformité de type) tout en lui ajoutant des compétences supplémentaires (de manière dynamique)
  - Donne une alternative souple à l'héritage (via la délégation)

Le modèle « Décorateur » (2/6)

- Motivation
  - Le décorateur est un objet qui offre l'interface de l'objet décoré mais qui enveloppe ce dernier et lui ajoute une fonctionnalité
    - L'objet décoré est un délégué du décorateur
  - On peut imbriquer récursivement les décorateurs
  - Par exemple, si on veut agrémenter une fenêtre de texte (qui gère l'affichage et les évènements) d'une barre de défilement, d'un encadrement particulier et/ou d'un compteur de caractères…
    - À chaque « agrément » (barre, cadre, compteur…) correspondra un décorateur de type « fenêtre »
    - Les décorateurs seront composés par délégation pour fabriquer par exemple une fenêtre de texte avec compteur et barre de défilement…

Le modèle « Décorateur » (3/6)

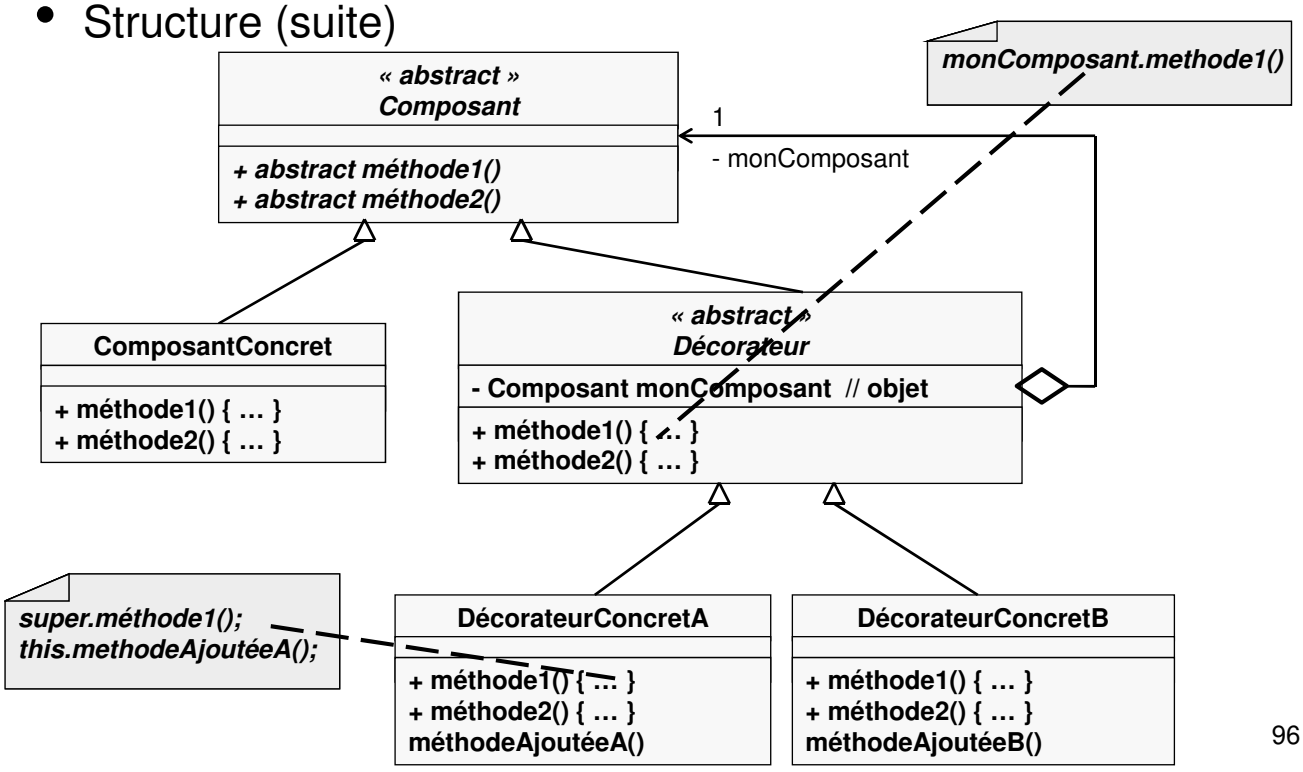
- Indication d'utilisation
  - Pour pouvoir « ajouter » des opérations à des objets sans avoir à modifier les classes existantes
    - Au moment de la configuration (déploiement)
  - Quand l'héritage n'est pas souhaitable, pas possible ou limité

Le modèle « Décorateur » (4/6)

- Participants
  - *Composant* : classe abstraite (ou interface) qui spécifie l'interface des objets qui peuvent être décorés
  - *Composant concret* : classe qui définit un objet à décorer
  - *Décorateur* : classe abstraite qui implante l'interface de *Composant* et qui gère une référence à un *Composant*
  - *Décorateur concret* : ajoute une responsabilité au composant et redéfinit les méthodes de l'interface
- Collaboration
  - Le décorateur transmet les requêtes à l'objet décoré et peut y ajouter des opérations complémentaires

Le modèle « Décorateur » (5/6)

- Structure (suite)



Le modèle « Décorateur » (6/6)

- Structure
  - Diagramme d'objets (exemple)