La Rochelle Université
Licence Informatique
UE Génie logiciel 2

Design pattern Abstract factory

Sujet d'Arnaud Revel

Version enseignant

Vous venez d'arriver dans votre nouvelle entreprise, et vous récupérez un programme qui avait été mis en place par votre prédécesseur. Ce programme permet, étant donné un chemin complet vers un fichier Windows, d'afficher juste le nom du fichier. Voici le code de la seule classe ce projet :

```
1  public class Main
2  {
3     public static void main_parse_filename(String path)
4     {
5        //index est l'endroit où se situe, dans la String path,
6        //la dernière apparition du caractère \
7        int index = path.lastIndexOf("\\") ;
8        //On construit une String qui ne contient que la partie
9        //située à droite du dernier caractère \
10       String r = path.substring(index+1);
11       System.out.println(r);
12    }
13 }
```

En passant en paramètre "C : \W*indows* \ *hello.dll*" à ce programme, il devrait renvoyer "hello.dll".

— Avec le temps, on souhaite faire évoluer les possibilités de ce programme pour permettre de donner un chemin vers un fichier Linux (de type "/user/share/hello.rc"). Pour ce faire, on va tout d'abord créer une interface ***FileNameParser***, qu'implémenteront deux sous classes ***ParseFileNameWindows*** et ***ParseFileNameLinux***. Ensuite, on créera une ***Factory*** qui permettra de créer des *FileNameParser* dans le main() ;

— On souhaite ajouter une autre fonctionnalité qui consiste à compter le nombre de dossiers qu'il faut parcourir depuis la racine pour trouver notre fichier. On va pour cela ajouter une interface ***CountFolders*** qu'implémenteront deux sous classes ***CountFoldersWindows*** et ***CountFoldersLinux***. On utilisera la même factory pour créer les deux types d'objets (un *FileNameParser* et un *CountFolders*) ;

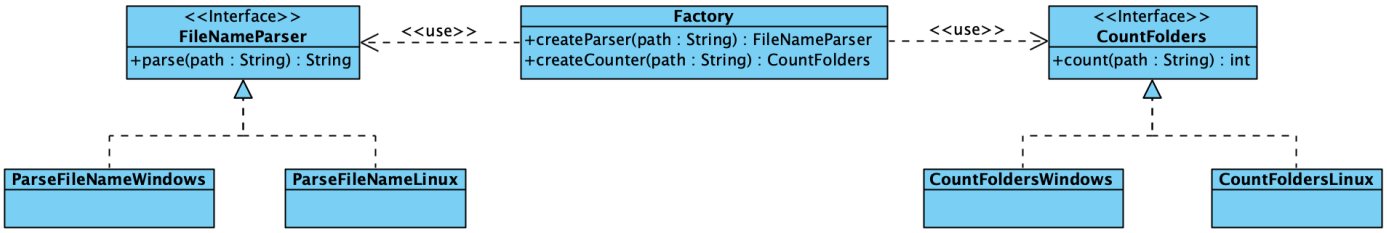

FIGURE 1 – Solution des étapes 1 et 2

— Le problème de cette implémentation est que, si on veut ajouter d'autres fonctionnalités, il faudra changer le code de la factory. On voudrait ne pas avoir à modifier notre factory et comprendre tout ce qui se passe dans la factory pour savoir où implémenter le code. On va donc faire de notre factory une interface et créer deux sous factory qui l'implémentent (***FactoryLinux*** et une ***FactoryWindows***). Dans la classe Main, il faudra cependant choisir explicitement quelle factory utiliser ;

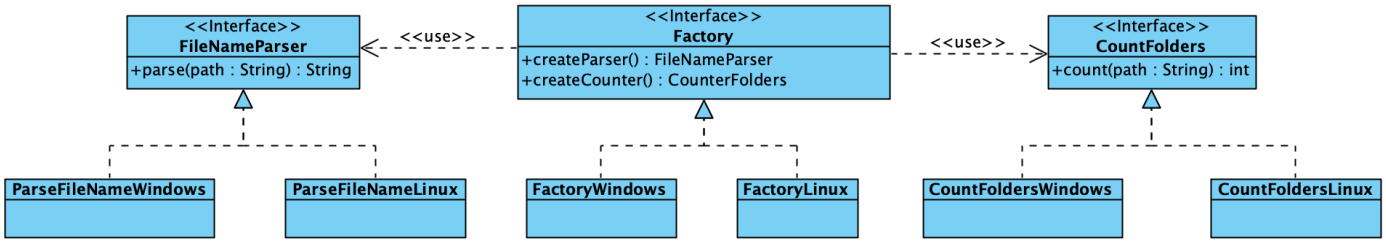

FIGURE 2 – Solution de l'étape 3

— Pour éviter que dans la classe Main on ait à choisir explicitement quelle factory instancier et pour éviter que le Main "ait connaissance" de toutes les factory on va créer une factory de factory ;

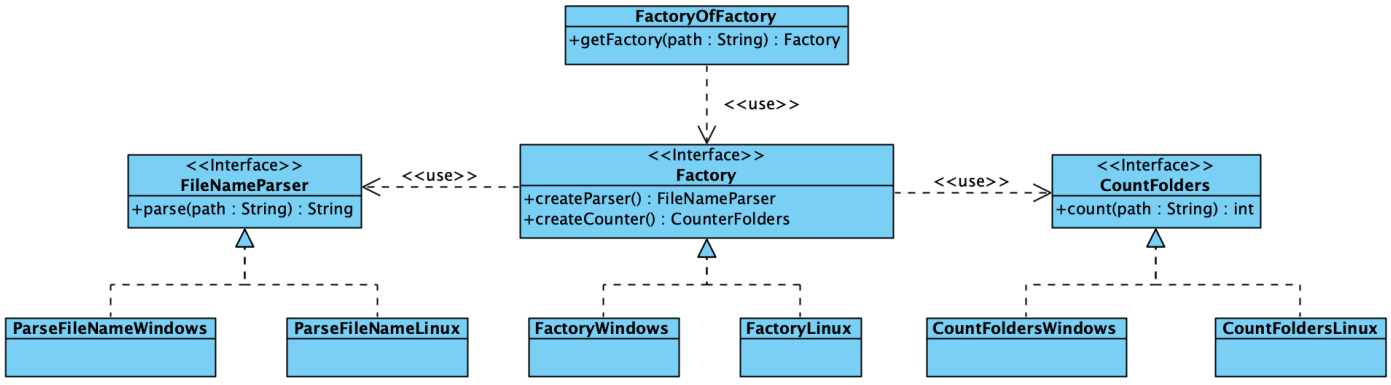

FIGURE 3 – Solution de l'étape 4

— On veut ajouter le support des chemins natifs Mac (la forme est alors " :user :share :hello.rc "). Faîtes les modifications nécessaires ;

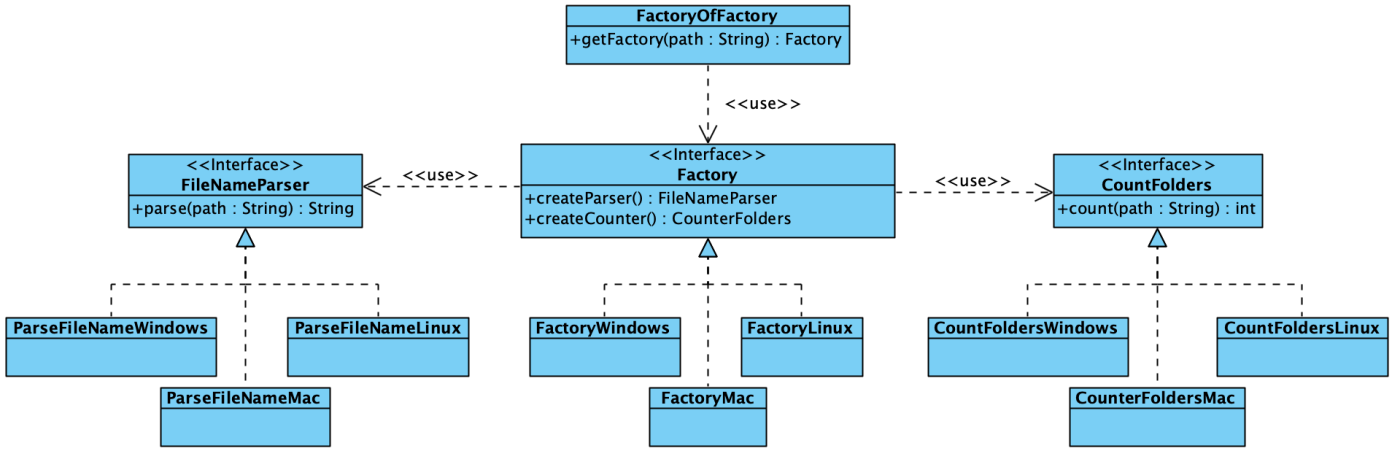

FIGURE 4 – Solution de l'étape 5

— On souhaite ajouter la fonctionnalité consistant à donner le nom du répertoire source qui contient notre fichier. Faîtes les modifications nécessaires.

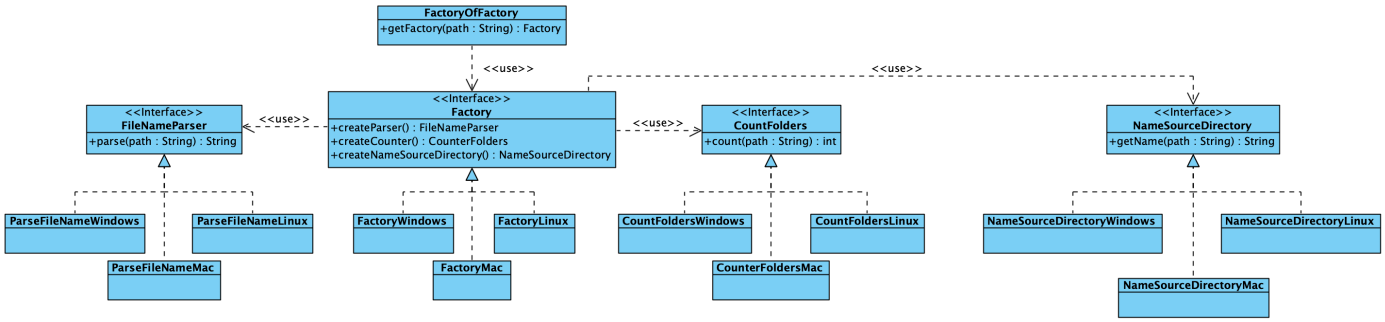

FIGURE 5 – Solution de l'étape 6