Software Engineering
Solver de Sudoku

François Flandin
Anthony Vasta

Novembre 2024

1 Introduction

Dans ce rapport, nous allons présenter les différents choix de conception logiciels pour notre projet de solver de sudoku.

2 La Grille

2.1 Les Cellules

Avant de parler de la manière dont la grille est construite, nous devons présenter d'abord ce qui la compose, les cellules.
Chaque cellule possède deux champs, la valeur, 0 si vide, 0 <n <10 sinon, ainsi que ses valeurs possibles enregistrées dans un objet de type *HashSet*, car on n'a besoin de stocker qu'un seul exemplaire de chaque nombre. Les valeurs possibles seront utilisées dans les *Deduction Rule*.

2.2 La Conception

La grille est conçue comme un tableau de 81 elements de classe Cellule, parce que c'était la version initiale qui était décrite dans l'énoncé, de plus, on rajoute 3 tableaux de tableaux, qui définissent les lignes/colonnes/carres de la grille, afin de faciliter la navigation dans la grille.

3 Les Deduction Rules

3.1 La conception

Chaque *Deduction Rule* implémente l'interface *Deduction Rule*, ce qui nous permet d'être beaucoup plus modulaire dans notre approche, par exemple dans la partie du *StateDR*. De plus chaque *Deduction Rule* implémente le Design Pattern **singleton**, car il n'est pas nécessaire d'avoir 150 objets de type *DR1*.

3.2 Déduction Rule 1

Si il reste strictement une case libre dans une ligne/colonne/carré, remplir la case avec le chiffre manquant.

3.3 Déduction Rule 2

On utilise la liste de valeurs possibles attribuées à chacune des cases, lorsqu'il ne reste plus qu'une valeur dans cette liste, on l'attribue à la case. C'était la première *Deduction Rule* implémentée et n'a pas posé tant de problèmes.

3.4 Déduction Rule 3

Cette *Deduction Rule* était la plus compliquée a concevoir, par manque d'idée et par problèmes de conception, nous avons essayé une première version, qui était réellement compliquée a mettre en place, on s'est tournés vers une autre solution:

On regarde toutes les valeurs possibles dans une ligne/colonne/carré, si une case dans sa section (ligne/colonne/carré) est la seule à avoir la valeur possible on lui attribue cette valeur.

3.5 Le StateDR

StateDR implémente le Design Pattern state, qui nous permet facilement de transitionner entre les différentes *Deduction Rule*.

4 Les Problèmes Sans Solutions

Voici la liste des problèmes dont nous avons aucune solution.

- Vérification de la validité de la grille.
- Création du .jar
- Si la *DR3* tourne sans surveillance, elle peut générer quelques faux inputs.

5 Repartition du Travail

Pour la répartition, nous avons conçu les bases de l'architecture ensemble, puis nous avons développé les différentes parties ensemble grâce a la fonctionnalité de code partage sur IntelliJ.