Université Caen Normandie

U.F.R des Sciences
L3 Informatique

Méthodes de conception

Contrôle continu

Réalisé par :
Akcel Arab
Belaid Azil
Massinissa Meghira
Mahdi Mobarek

Encadrant :
Charrier Christophe

Année universitaire 2025-2026

Table des matières

1 Introduction

Le Blackjack est l'un des jeux de cartes de casino les plus populaires au monde. Son attrait repose sur un subtil mélange de hasard et de stratégie. L'objectif est simple : battre le croupier en obtenant une valeur totale de cartes supérieure à la sienne, sans dépasser 21. Si un joueur dépasse ce seuil, il « saute » (bust) et perd immédiatement sa mise.

L'objectif de ce projet est de développer une application complète simulant une partie de Blackjack, tout en adoptant une architecture logicielle propre, robuste et facilement maintenable. Une grande importance a été portée à l'utilisation de patrons de conception (design patterns) pour assurer modularité, réutilisabilité et extensibilité. Enfin, l'application est entièrement automatisée via un système de build Apache Ant.

2 Développement et Architecture Logicielle

La conception repose sur une architecture claire et modulaire, guidée par les principes du génie logiciel. L'application est découpée en deux modules indépendants : **cartes** et **blackjack**. Cette séparation garantit une meilleure lisibilité et une grande réutilisabilité du code.

2.1 Architecture modulaire : `cartes` et `blackjack`

Le projet est structuré en deux modules distincts :

— **Le module cartes** : contient toutes les classes génériques relatives à la gestion d'un jeu de cartes.

— **Le module blackjack** : implémente la logique métier du jeu.

2.2 La bibliothèque cartes : le cœur fonctionnel

Ce module contient les éléments fondamentaux de tout jeu de cartes :

— **Carte**

— **Paquet**

— **Main**

2.3 Le module blackjack : moteur du jeu

Ce module implémente les règles du jeu via :

— **Joueur** (humain ou IA)
— **Croupier**
— **JeuBlackjack**

2.4 Patrons de conception utilisés

Factory Construction des paquets de cartes.

Strategy IA, décisions du croupier et du joueur.

Chain of Responsibility Vérification des mises.

Proxy Gestion des cartes visibles du croupier.

Observer + MVC Mise à jour automatique des vues Swing.

3 Diagrammes UML

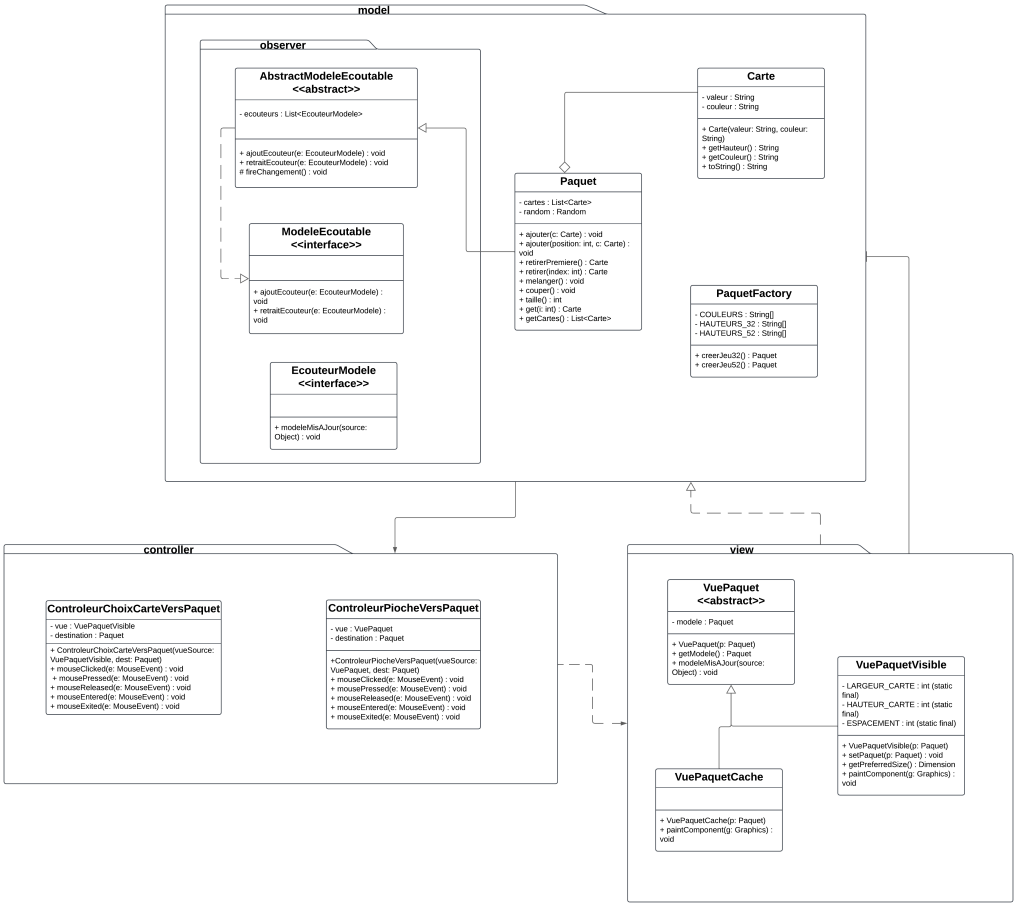

FIGURE 1 – Diagramme UML du package `cartes`

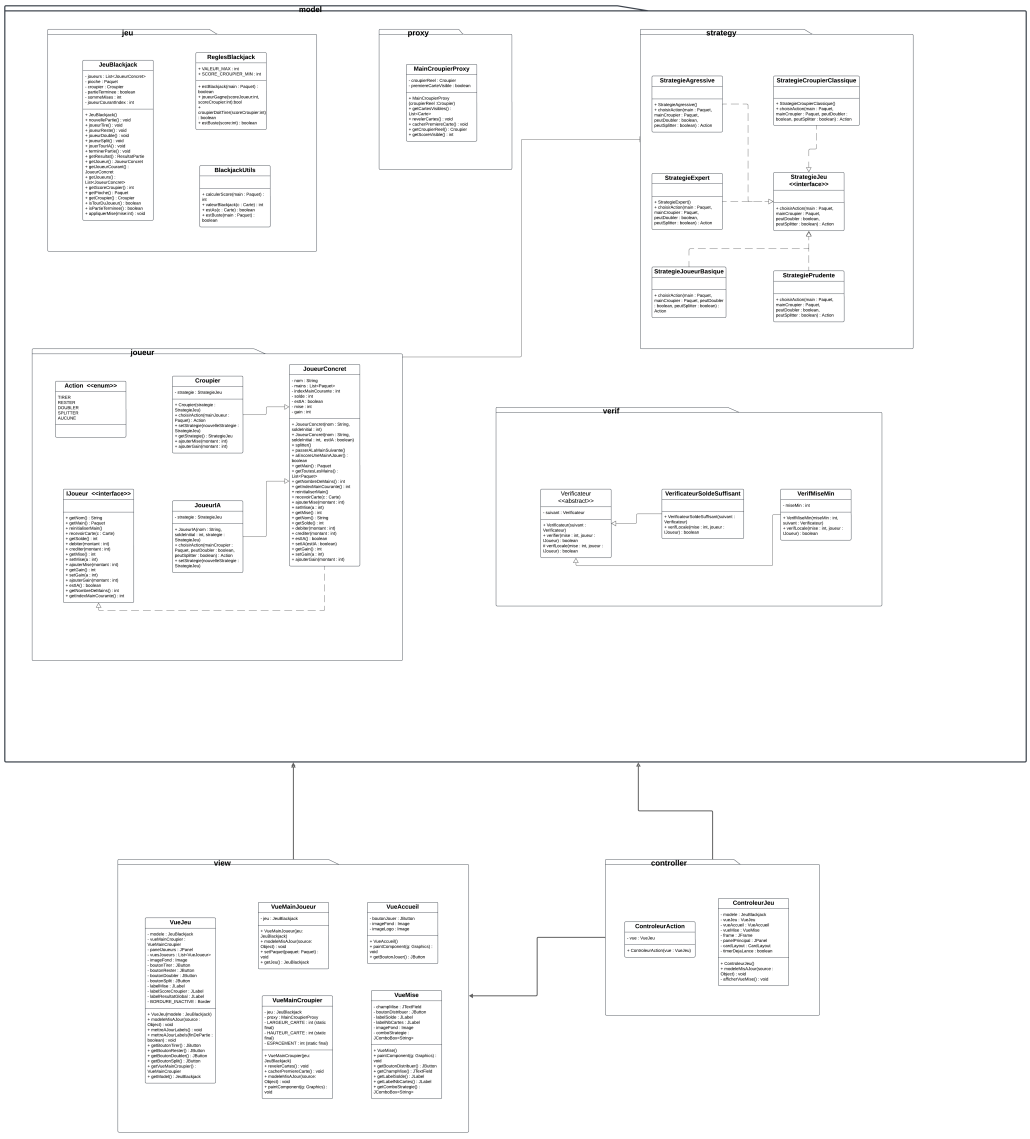

FIGURE 2 – Diagramme UML du package `blackjack`

4 Illustration de l'interface du jeu

Afin d'offrir une expérience utilisateur confortable, une interface graphique a été développée avec Swing.

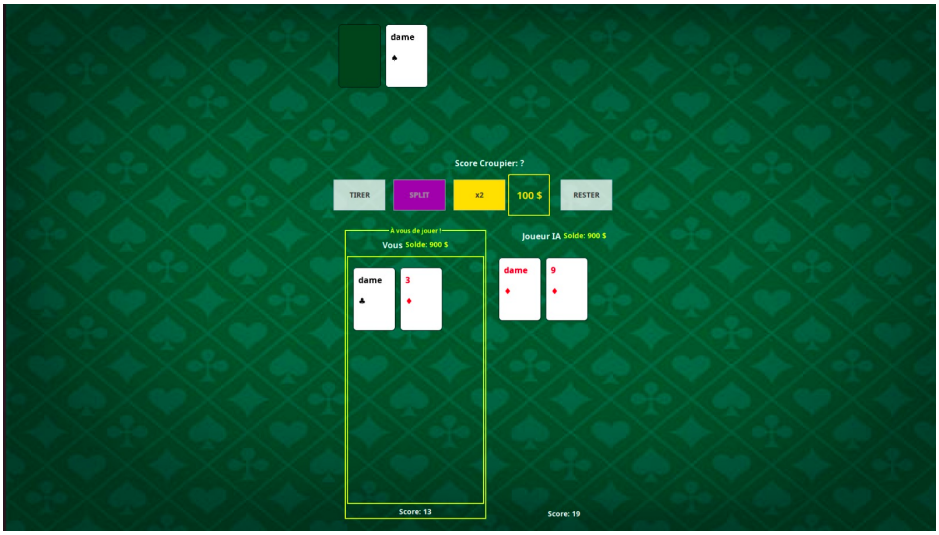

FIGURE 3 – Interface graphique du jeu de Blackjack

5 Utilisation

L'application a été pensée pour être simple d'utilisation et agréable à prendre en main. Afin d'améliorer le confort de l'utilisateur et de lui laisser le temps d'assimiler les résultats, **une attente de quelques secondes est systématiquement appliquée entre la fin d'une manche et le début de la suivante**. Cela permet d'observer clairement le résultat avant de poursuivre le jeu.

L'exécution du projet repose sur **Apache Ant**, ce qui rend la compilation et le lancement extrêmement simples.

5.1 Compiler le projet

```
ant compile
```

5.2 Exécuter le jeu

```
ant run
```

Ces commandes automatisent entièrement le cycle de build sans intervention manuelle.

6 Conclusion

Ce projet a permis d'appliquer des concepts avancés de conception logicielle, notamment les design patterns, l'architecture MVC et la modularité. L'application obtenue est robuste, évolutive et constitue une base solide pour des extensions futures telles que :

— IA plus avancée,
— Mode multijoueur,
— Animations plus immersives,
— Historique des parties.