ACVL - Jeu de Simulation Boursière
Document de Conception

Kévin Bonkoski Roxane Brenier Jérôme Lapostolet Chenhao Yan

23 avril 2015

Table des matières

Introduction

1 Architecture logique du logiciel

Pour la conception de notre application, nous avons choisi de suivre l'architecture Modèle-Vue-Contrôleur (MVC). Ceci présente plusieurs avantages. D'une part, cela force à bien séparer les différentes méthodes en fonction de leurs actions concrètes. Ensuite, cela permet un développement aisé, même à plusieurs, chacun pouvant s'occuper d'une des trois partie sans risque de conflit avec les autres jusqu'à l'assemblage final si les spécifications sont bien définies (d'où l'importance du travail d'analyse et de conception).

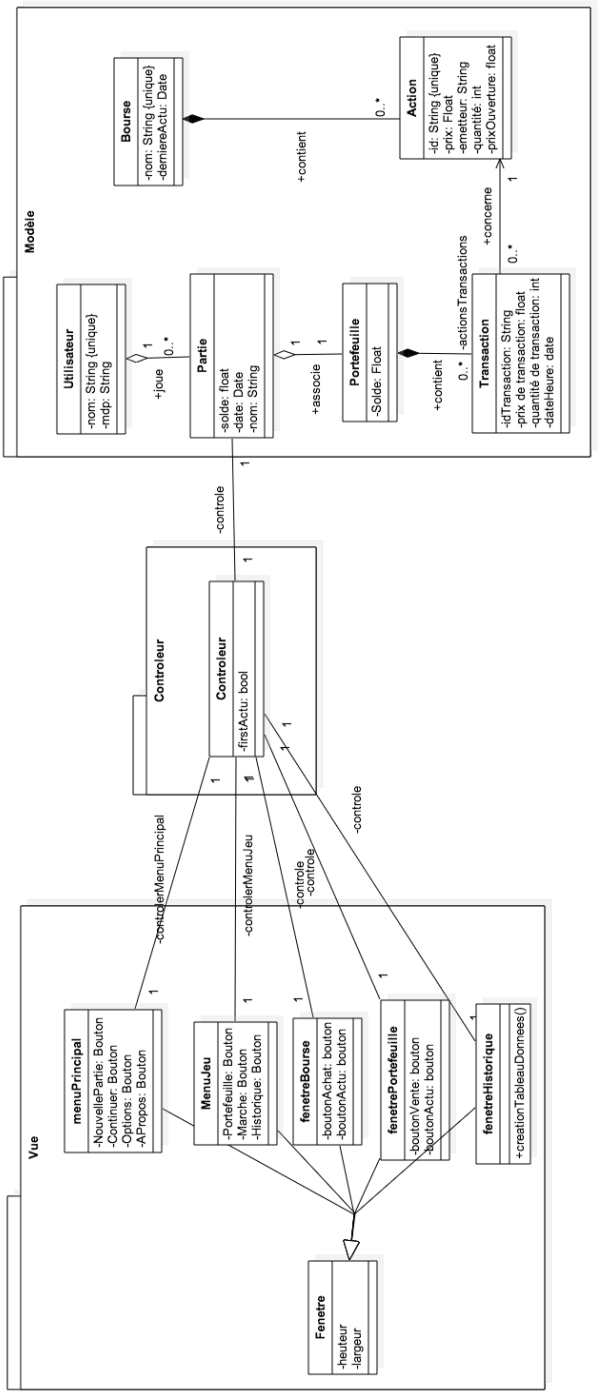

FIGURE 1 – Diagramme de la modèle MVC

Le modèle comporte globalement les classes vues dans le diagramme de classes d'analyse. Il est codé presque intégralement en C++ "classique" (il y a quelques exceptions afin de simplifier fortement le travail sur les dates par exemple) afin de permettre le choix d'un autre framework que *Qt* pour la vue.

La vue, totalement indépendante ne contient presque que des objets spécifiques à *Qt*. Elle contient des classes permettant d'afficher les fenêtres et leurs éléments.

Le contrôleur, unique pour l'application, effectue le lien entre les deux. Ainsi il transforme les types de bases en leur équivalent Qt lorsque nécessaire ou inversement afin de pouvoir effectuer sur le modèle les modifications impliquées par les actions de l'utilisateur. Il contient aussi les éléments permettant la mise à jour de la vue.

Il est à noté que du fait de la conception de Qt, une légère partie du rôle du contrôleur se fait ici dans la vue, puisque la réaction des boutons se gère directement depuis les classes correspondantes aux fenêtres. Mais nous avons fait en sorte que ces réactions se contentent la plupart du temps d'appeler des méthodes de contrôleur.

2 Description de l'incrément choisi

N'ayant pas le temps de développer intégralement le cahier des charges et ayant en tête de pouvoir démontrer en priorité les usages nous paraissant les plus pertinents, nous nous sommes limités à un incrément partiel. Ses possibilités sont détaillées dans le manuel utilisateur mais voici ses principales caractéristiques.

Nous ne gérons qu'une partie, d'un seul utilisateur et sans possibilité de sauvegarde. Ainsi à la fermeture toutes les données sont perdues. De plus cette partie concerne forcément le CAC40 et le solde initial du compte de l'utilisateur est fixé. Ceci permet néanmoins de pouvoir observer le comportement des fenêtres de bourse, portefeuille et historique. Ainsi l'utilisateur peut effectuer des achats et des ventes et constater les changements. Il peut également afficher son historique.

Concernant le fonctionnement de la bourse, elle est, dans cet incrément, active 24h sur 24. Une quantité d'actions disponibles est fixée aléatoirement une fois par jour et l'utilisateur ne peut alors pas en acheter plus sur la journée (pas plus qu'il ne peut acheter plus que son solde ne l'y autorise). Ce tirage se fait au démarrage de la partie puis à la première actualisation de l'utilisateur chaque jour (bien qu'ici ce sera difficile à tester du fait de l'absence de sauvegarde...). En effet les actualisations de l'affichage des prix et quantités ne se font que sur demande de l'utilisateur. Cependant, afin de simuler une variation continue, le temps passé depuis la dernière mise à jour est pris en compte dans le calcul du nouveau prix. Ainsi, plus un temps long aura passé, plus la variation sera potentiellement importante.

3 Conception détaillée

3.1 Diagramme de classes logicielles

Voici les diagrammes de classes logicielles des différents modules évoqués dans la description de l'architecture. On constate des changements par rapport au diagramme d'analyse bien sûr, du fait des contraintes de développement, mais la logique reste effectivement la même.

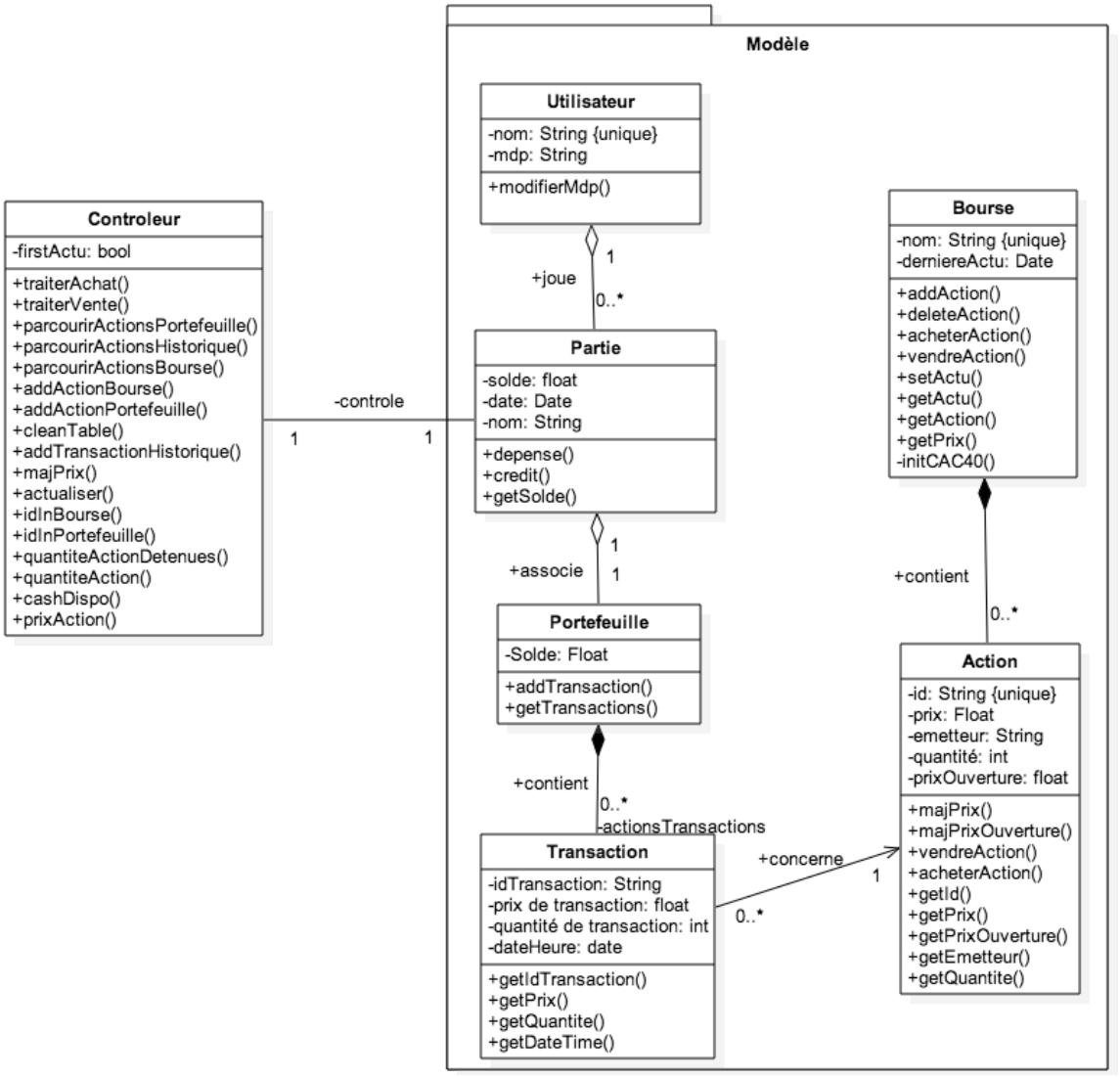

FIGURE 2 – Diagramme de classes logicielles : les classes de modèle et le controleur

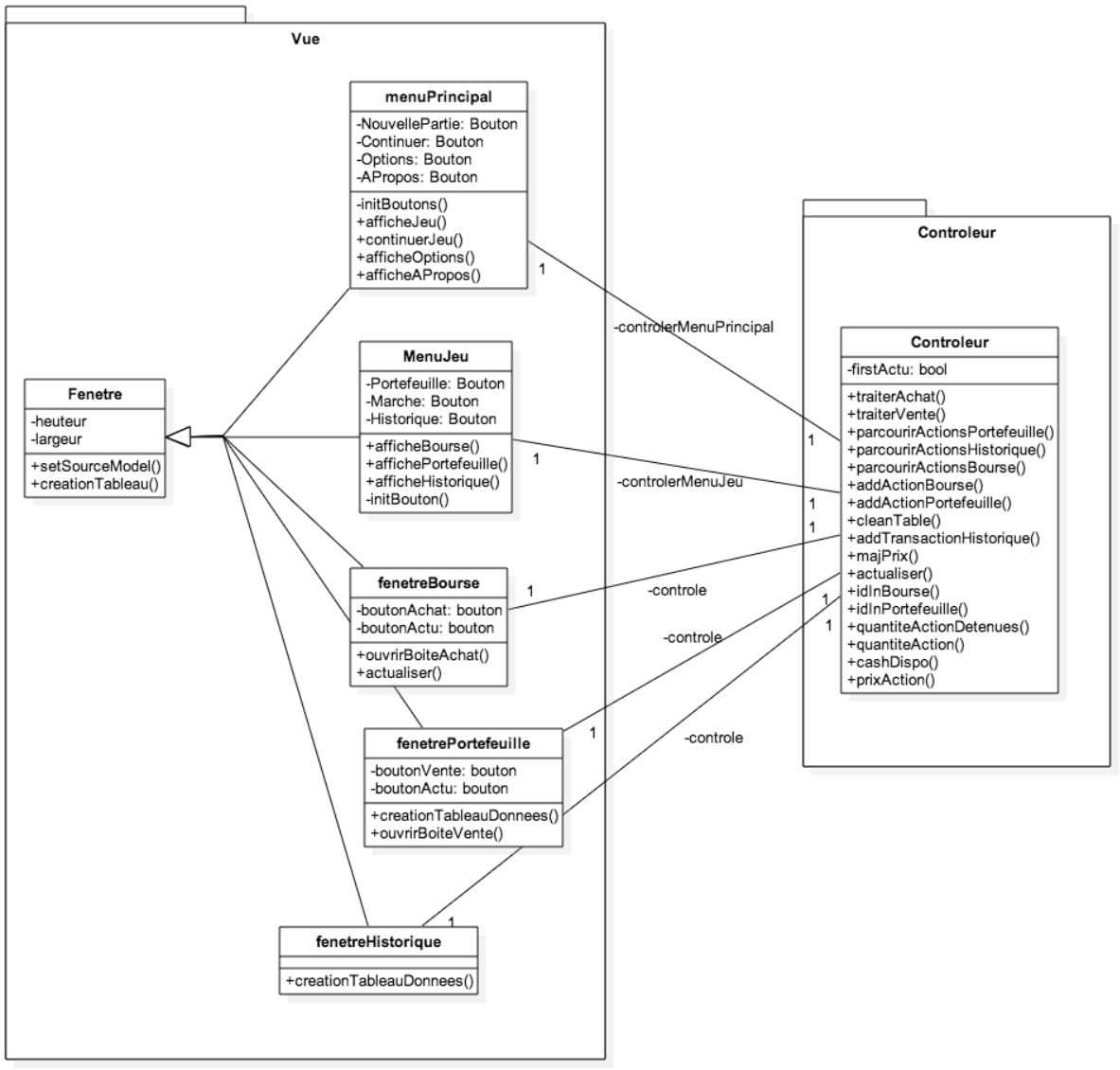

FIGURE 3 – Diagramme de classes logicielles : les classes de vue et le controleur

3.2 Diagrammes de séquence logiciels

Les diagrammes de séquences logiciels des principales actions sont montrés ici.

Pour l'achat/vente, nous ne présentons que le diagramme correspondant à l'achat, celui pour la vente étant très similaire.

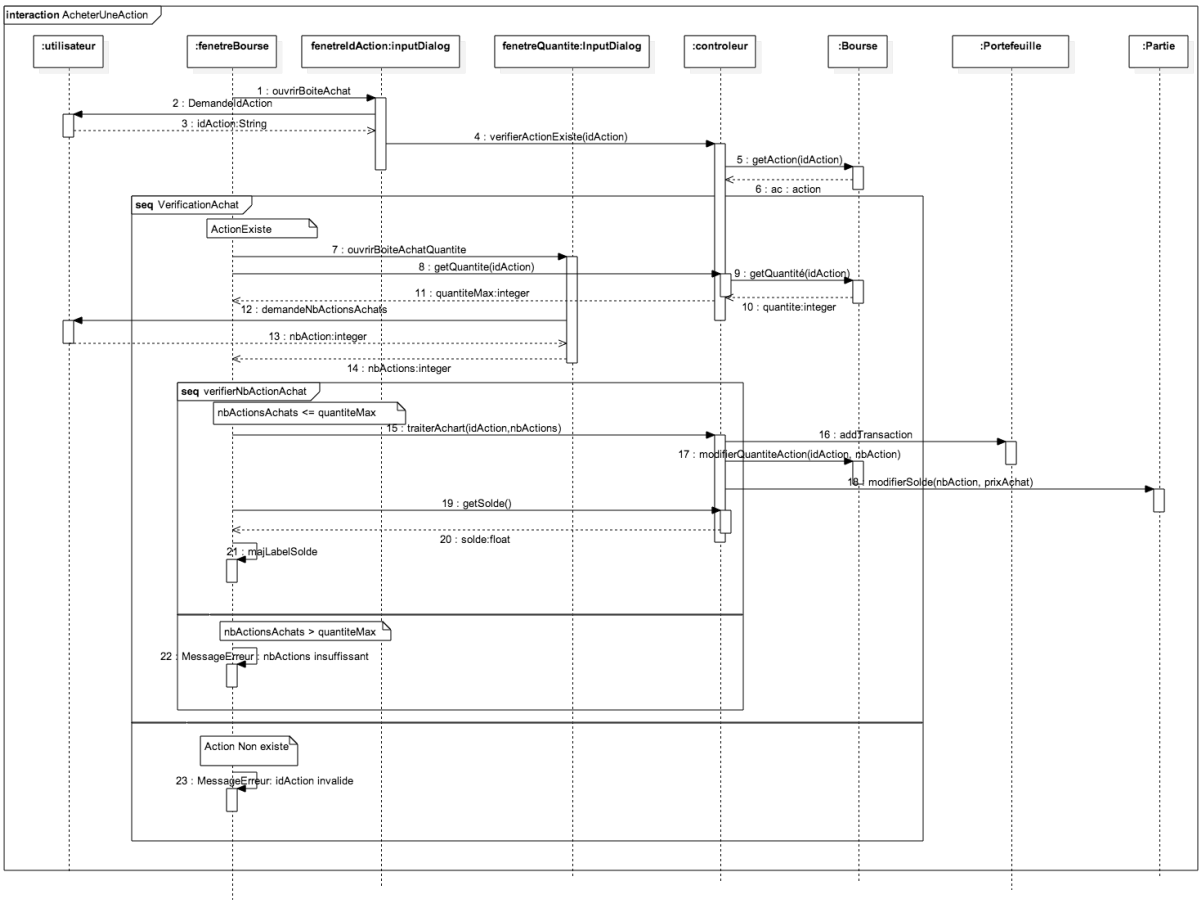

FIGURE 4 – Diagramme de séquence logiciel : achat d'une action

L'actualisation une fois demandée par l'utilisateur dans la fenêtre bourse. Le fonctionnement est très similaire pour elle demandée en fenêtre portefeuille. De plus des actualisations sont faites au lancement des fenêtres mais il n'y a alors pas d'appel à *cleanTable(model)*.

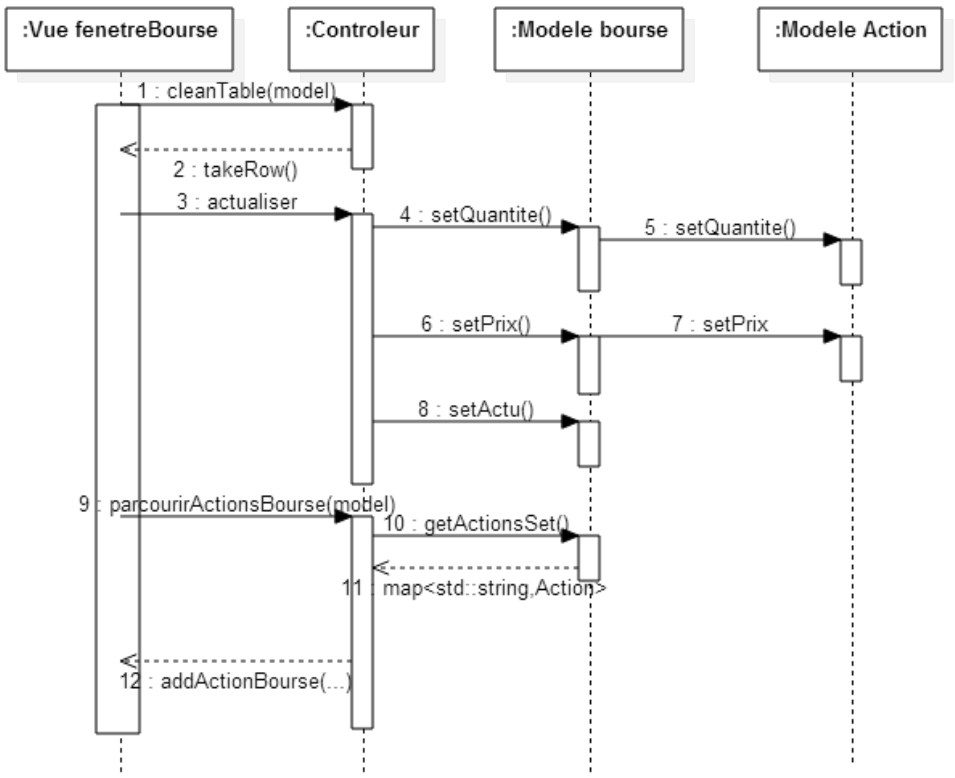

FIGURE 5 – Diagramme de séquence logiciel : actualisation de la bourse

L'affichage des fenêtres est à peu près tout le temps le même, nous ne détaillerons donc que celui concernant l'affichage de la fenêtre de bourse. La fin du diagramme correspondrait à celui de l'actualisation sans l'appel à *cleanTable(model)*.

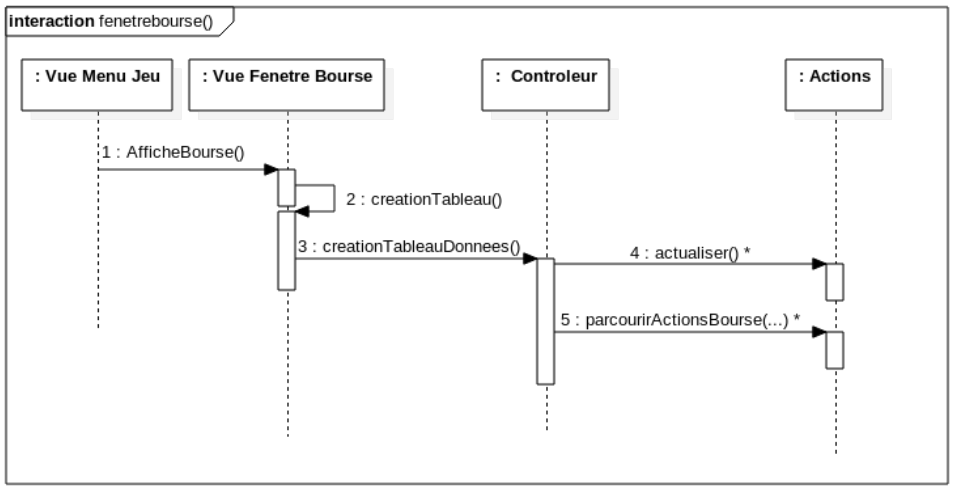

FIGURE 6 – Diagramme de séquence logiciel : consultation de la bourse

Table des figures