M1 ALMA
Université de Nantes
2011-2012

Projet de Vérification et Test : TP évalué

MARGUERITE Alain
RINCE Romain

Université de Nantes
2 rue de la Houssinière, BP92208, F-44322 Nantes cedex 03, FRANCE

Encadrant : Mottu Jean-Marie

Table des matières

1 Introduction et plan d'intégration

1.1 Présentation du problème

Ce TP évalué a pour objectif de corriger un logiciel de jeu d'échecs codé en java.

1.2 Plan d'intégration

Le logiciel utilisait le design pattern MVC. Il était demandé de se focaliser sur la partie métier uniquement. A l'origine nous dispostion du document suivant créer notre plan d'intégration :

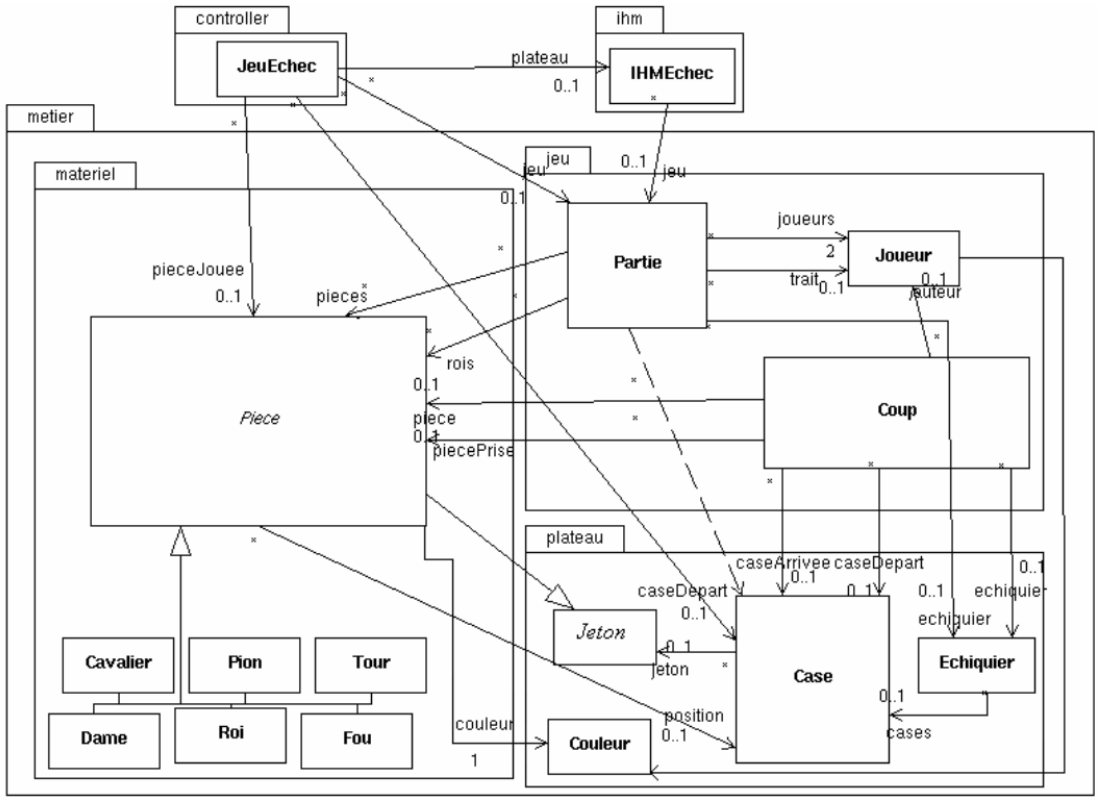

FIGURE 1.1 – Diagramme UML de la partie métier

Au terme de notre réflexion nous avons obtenu le graphe de dépendances suivant :

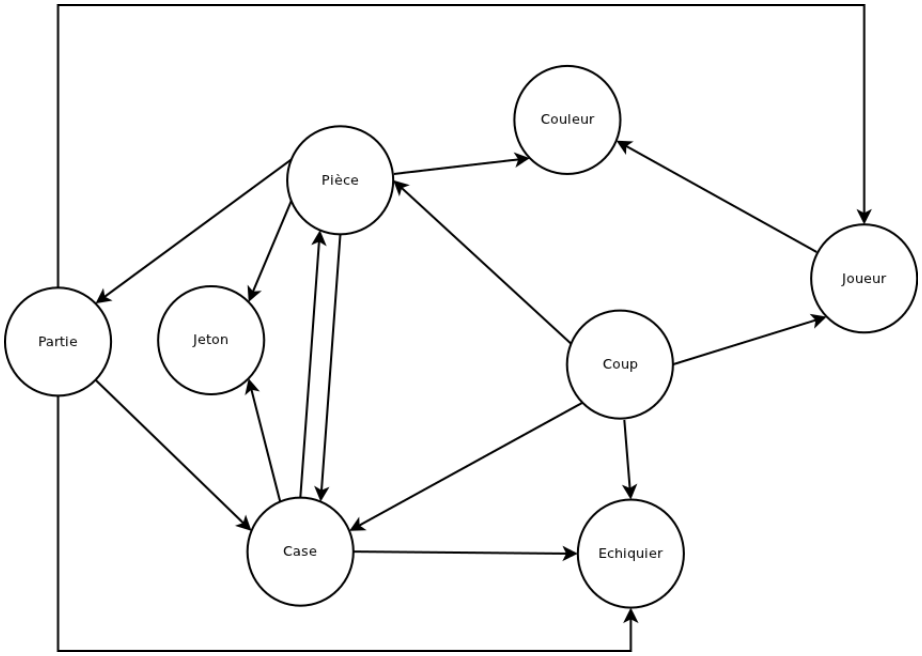

FIGURE 1.2 – Graphe de dépendance

Le plan d'intégration en utilisant un stub de la classe de Case découle du graph de dépendances à savoir :

- Couleur (Ne dépend de rien)
- Joueur (Dépend de couleur)

Puis :

- Pièce(Avec StubCase)
- Echiquier(Avec StubCase)
- Coups(Avec StubCase)
- Partie(Avec StubCase)
- Case
- Pièce
- Echiquier
- Coup
- Partie

2 Rapport de Tests

2.1 Dame

1. Erreur peutBouger : absence d'un test dans le if : la dame peut aussi se déplacer sur une ligne. Trouvée grâce au testPeutBougerLigne().

2.2 Fou

1. Erreur peutBouger : condition en trop dans le if :le fou peut se déplacer en ligne. Trouvée grâce au testPeutBougerLigne().

2.3 Pion

1. Erreur getType() : retourne chaîne vide au lieu de P. Trouvée grâce au testGetType().

2.4 Echiquier

1. Erreur intervalleLibre : elementAt(1) ou lieu de elementAt(i). Trouvée grâce au testIntervallePasLibre().
2. Erreur getIntervalle : nom mal choisi. On pourrait croire à une inversion entre le if et le premier elsif. Trouvée grâce au testIntervallePasLibre().
3. Erreur getLigne : Indice de départ mal positionné. Trouvée grâce au testIntervallePasLibre().

2.5 Coup

1. Erreur estValide() : test sur la pièce allant être prise au lieu de celle se déplaçant. Trouvée grâce au test testEstValide().

2. Erreur annuler() : la pièce prise pouvait être nulle. Trouvée grâce au test testAnnuler().

2.6 Partie

1. Erreur private Joueur joueurs[] : joueurs non initialisés. Trouvée grâce à testPartie().
2. Erreur echec(Joueur joueur_p) : au niveau du nombre de pieces à parcourir. Trouvée grâce à testJouerEchec().
3. Erreur changerTrait() : erreur de changement de trait entre les deux joueurs. Touvée grâce à testJouer().

2.7 Case

1. Erreur Case(char colonne_p, int ligne_p) testConstEqCol() : Inversion des parametres du constructeur pour l'initialisation des attributs. Touvée grâce à testConstEqCol().
2. Erreur isOccupee() : Renvoie estLibre au lieu de isOccupee. Trouvé grâce à testEffectuer() (TestCoup).
3. Erreur caseVoisine() : la fonction condière que les case en diagonales (de distance inférieur ou égale à 1 au lieu de 0) ne sont pas voisines. Cependant une pièce telle que le roi utilise caseVoisine pour connaitre le déplacement autorisé.Trouvé grâce à testCaseVoisineFalse().
4. Nous avons été intrigué par le fait que a méthode estAudessus n'est jamais utilisée.