Projet: StoneHearth

Equipe de développement : Adrien ARTS Chao LIN Ling-Chun SO Christopher TRUBLEREAU

SnowStorm Itd

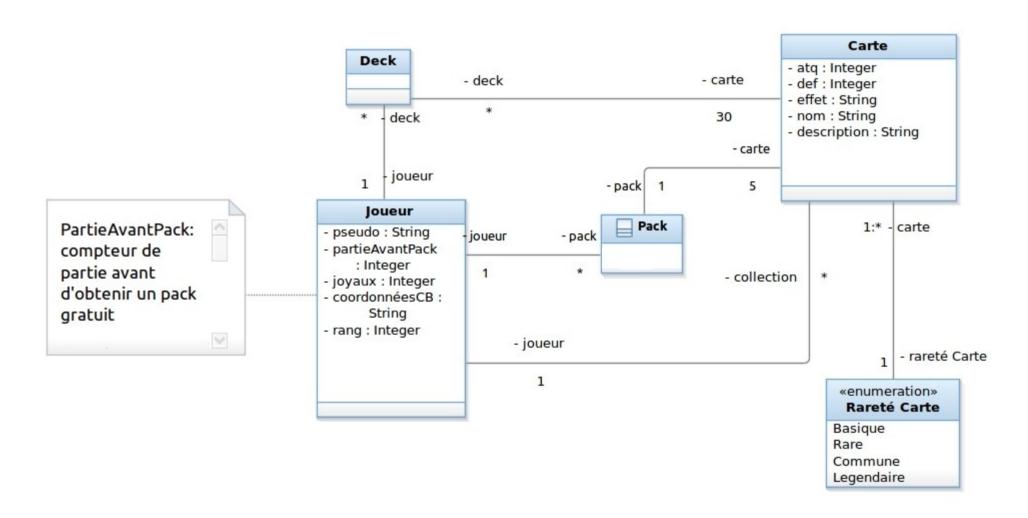


Sommaire

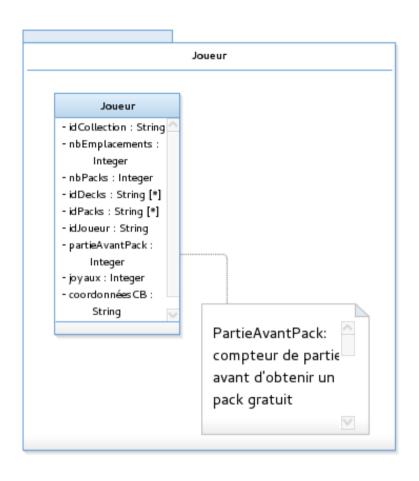
- Introduction
- Conception architecturale
 - Découpe structurelle
 - Interfaces et signatures
 - Diagramme de séquence inter-composants
- Tests d'intégration
- Factory
- Conception détaillée
- Conclusion

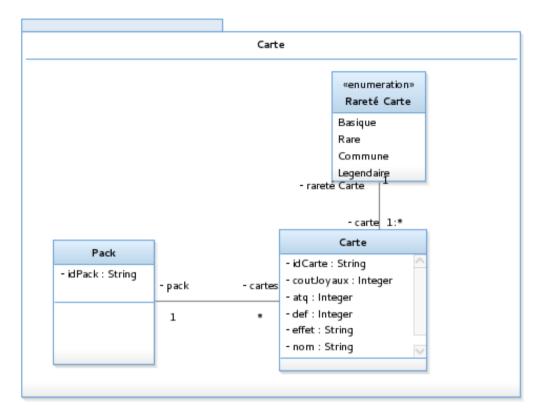
Conception architecturale

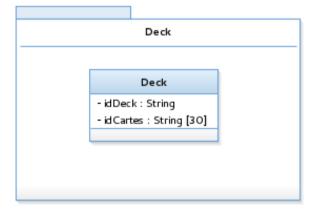
Rappel du diagramme de classe

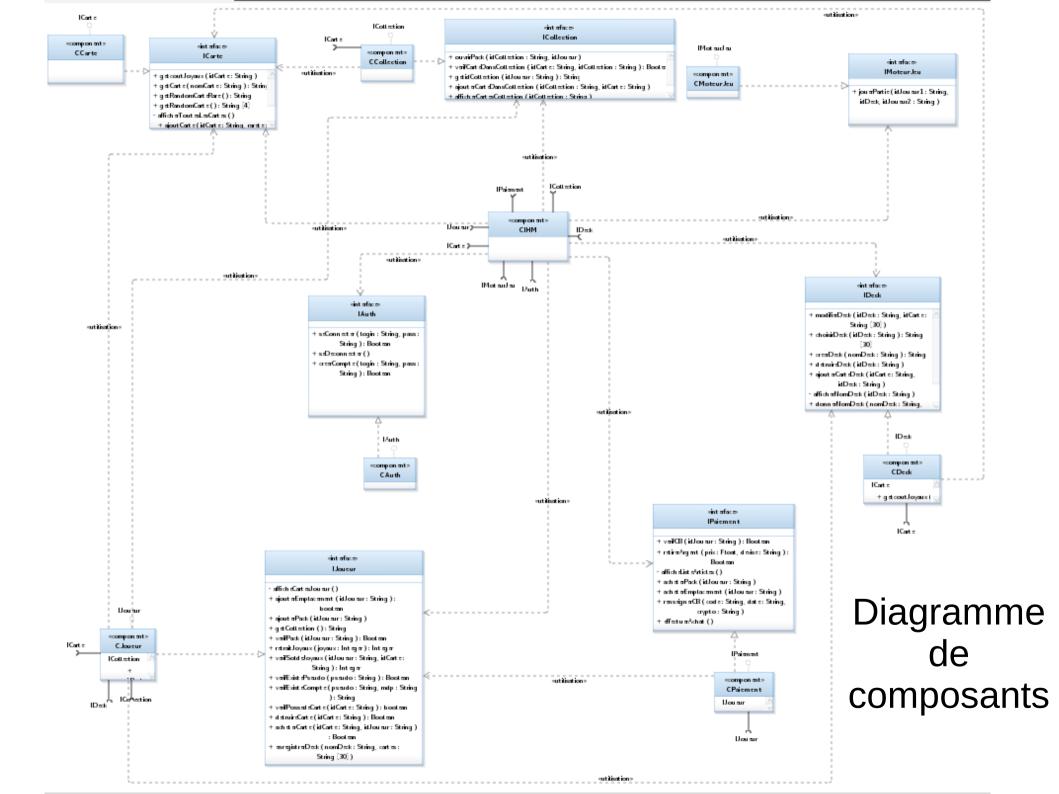


Découpe structurelle









Interfaces et signatures

interfaces

#interface** IMoteurJeu + jouerPartie (idJoueur1 : String , idDeck, idJoueur2 : String)

+ modifierDeck (idDeck : String , idCarte : String [30]) + choisirDeck (idDeck : String) : String [30] + creerDeck (nomDeck : String) : String + detruireDeck (idDeck : String) + ajouterCarteDeck (idCarte : String , idDeck : String) - afficherNomDeck (idDeck : String) + donnerNomDeck (nomDeck : String , idDeck : String)

verifSansDoublon (idCarte : String, idDeck : String) : Boolean

«interface»

```
#interface*
IPaiement

+ verifCB ( idJoueur : String ) : Boolean
+ retirerArgent ( prix : Float, devise : String ) : Boolean
- afficheListeArticles ( )
+ acheterPack ( idJoueur : String )
+ acheterEmplacement ( idJoueur : String )
+ renseignerCB ( code : String, date : String, crypto : String)
+ effectuerAchat ( )
```

«interface» IAuth + seConnecter (login : String, pæs : String) : Boolean

+ seDeconnecter ()

+ creerCompte (login : String, pass : String) : Boolean

«interface» ICarte

+ getcoutJoyaux (idCarte: String) + getCarte (nomCarte: String): String + getRandomCarteRare (): String

+ getRandomCarteRare (): String + getRandomCarte (): String [4]

- afficherToutesLesCartes ()

+ ajoutCarte (idCarte : String, rarete : String, coutJoyaux : String, atq : Integer, def : Integer, effet : String, nom : String, description : String) : boolean

«interface» ICollection

+ ouvrirPack (idCollection : String, idJoueur)

+ verifCarteDansCollection (idCarte : String, idCollection : String) : Boolean

+ getidCollection (idJoueur : String) : String

 $+ \ \mathsf{ajouterCarteDansCollection} \ (\ \mathsf{idCollection} : \mathsf{String}, \ \mathsf{idCarte} : \mathsf{String} \)$

+ afficherCartesCollection (idCollection: String)

+ ajouteCartesDansCollection (idCollection : String, listeCartes : String [*])

«interface»

- afficheCartesJoueur ()

+ ajouterEmplacement (idJoueur : String) : boolean

+ ajouterPack (idJoueur : String)

+ getCollection () : String

+ verifPack (idJoueur : String) : Boolean

+ retraitJoyaux (joyaux : Integer) : Integer

+ verifSoldeJoyaux (idJoueur : String, idCarte : String) : Integer

+ verifExistePseudo (pseudo : String) : Boolean

+ verifExisteCompte (pseudo : String, mdp : String) : String

+ verifPossedeCarte (idCarte : String) : boolean

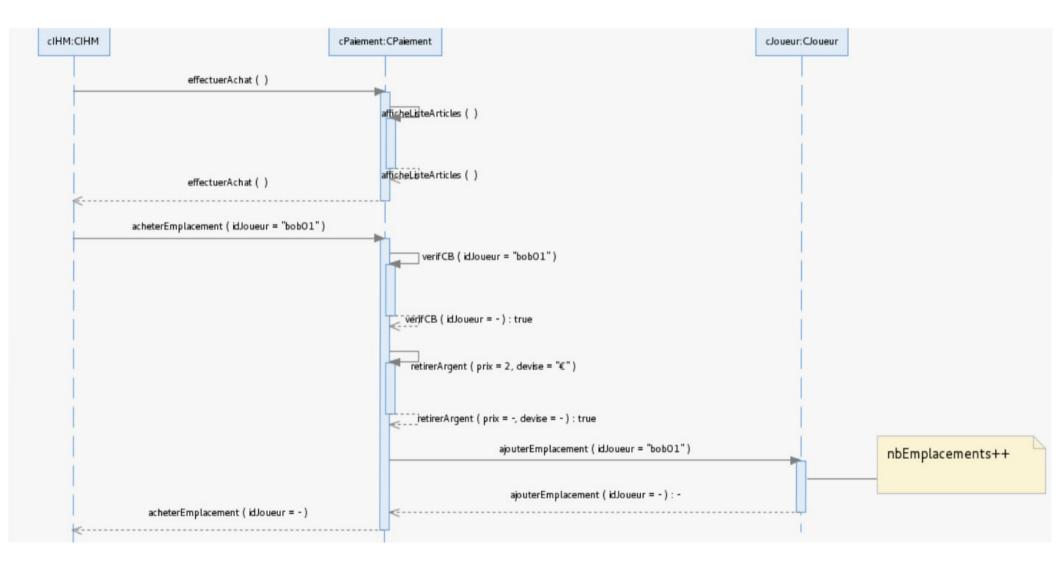
+ detruireCarte (idCarte : String) : Boolean

+ acheterCarte (idCarte : String, idJoueur : String) : Boolean

+ enregistrerDeck (nomDeck : String, cartes : String [30])

Diagramme de séquence inter-composants

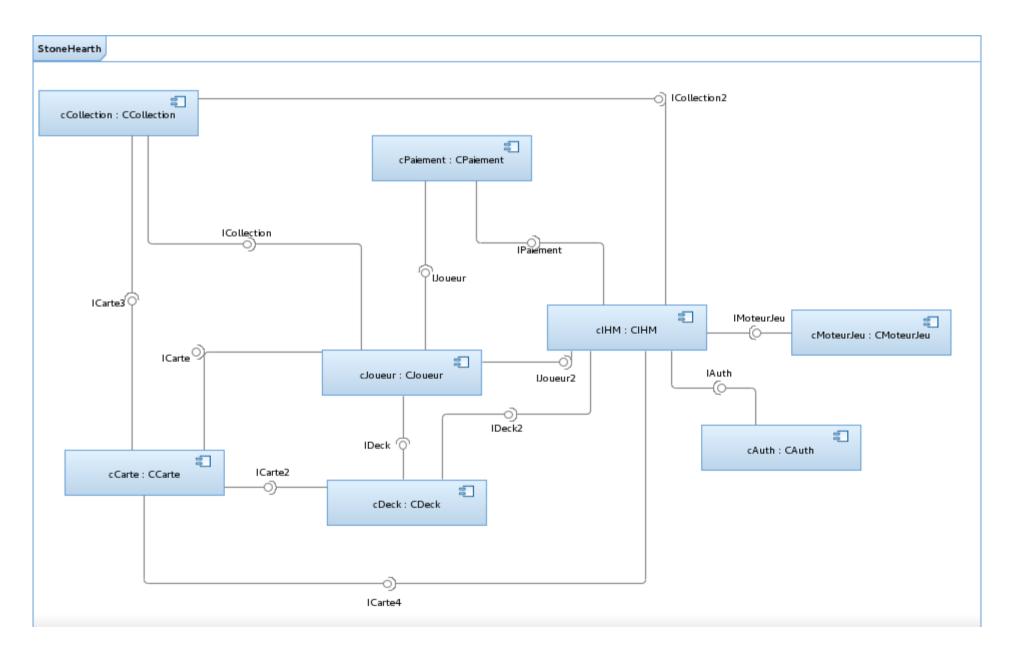
Effectuer Achat



Gérer Deck

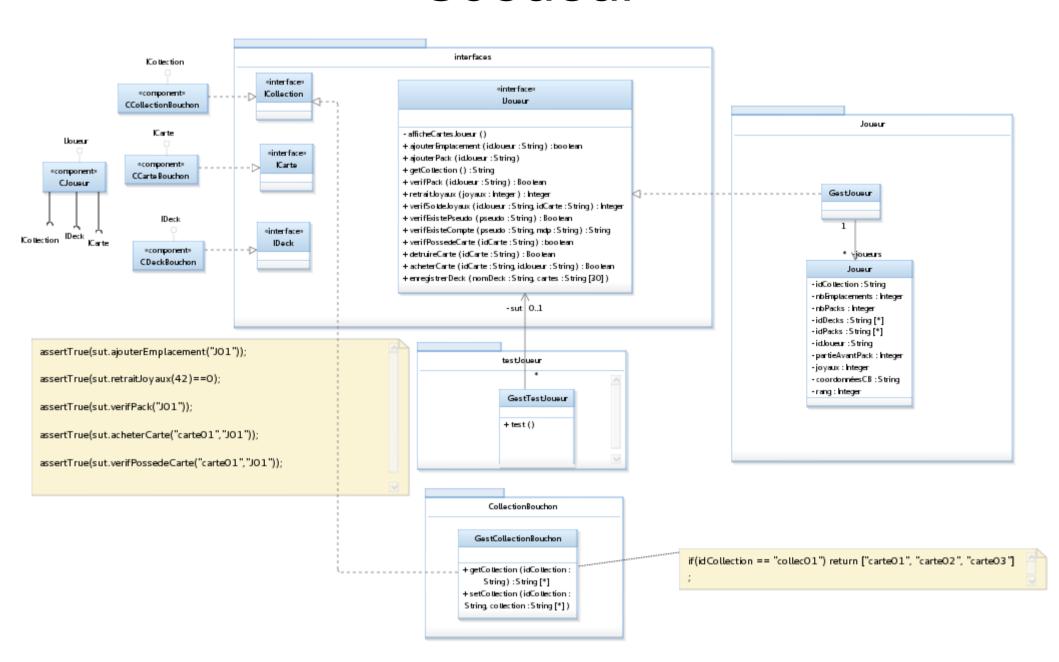
```
cIHM:CIHM
                                                    cJoueur:Cloueur
                                                                                                          cDeck:CDeck
                                                                                                                                              cCollection:CCollection
enregistrerDeck ( nomDeck = "dragon", cartes = "dragon1,dragon2,...,dragon30" )
                                                                      creerDeck ( nomDeck = "dragon" )
                                                                     creerDeck ( nomDeck = - ) : "deck1"
                                                             getCollection ( )
                                                         getCollection ( ): "col1"
                                                                         verifCarteDansCollection (idCarte = "dragon1", idCollection = "col1")
                                                                             verifCarteDansCollection ( idCarte = -, idCollection = - ) : True
                                                          verifSansDoublon (idCarte = "dragon1", idDeck = "deck1")
                                                              verifSansDoublon (idCarte = -, idDeck = -) : True
                                                                                                                                                       Boucle 30 fois
                                                           ajouterCarteDeck ( idCarte = "dragon1", idDeck = "deck1" )
                                                                  ajouterCarteDeck ( idCarte = -, idDeck = - )
                   enregistrerDeck ( nomDeck = -, cartes = -)
```

Diagramme de structure interne

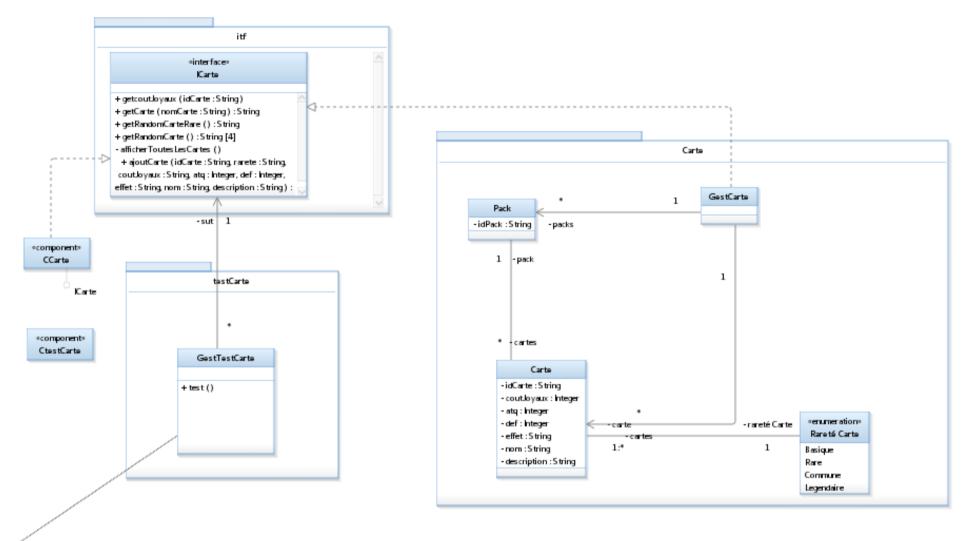


Tests d'intégration

CJoueur



CCarte



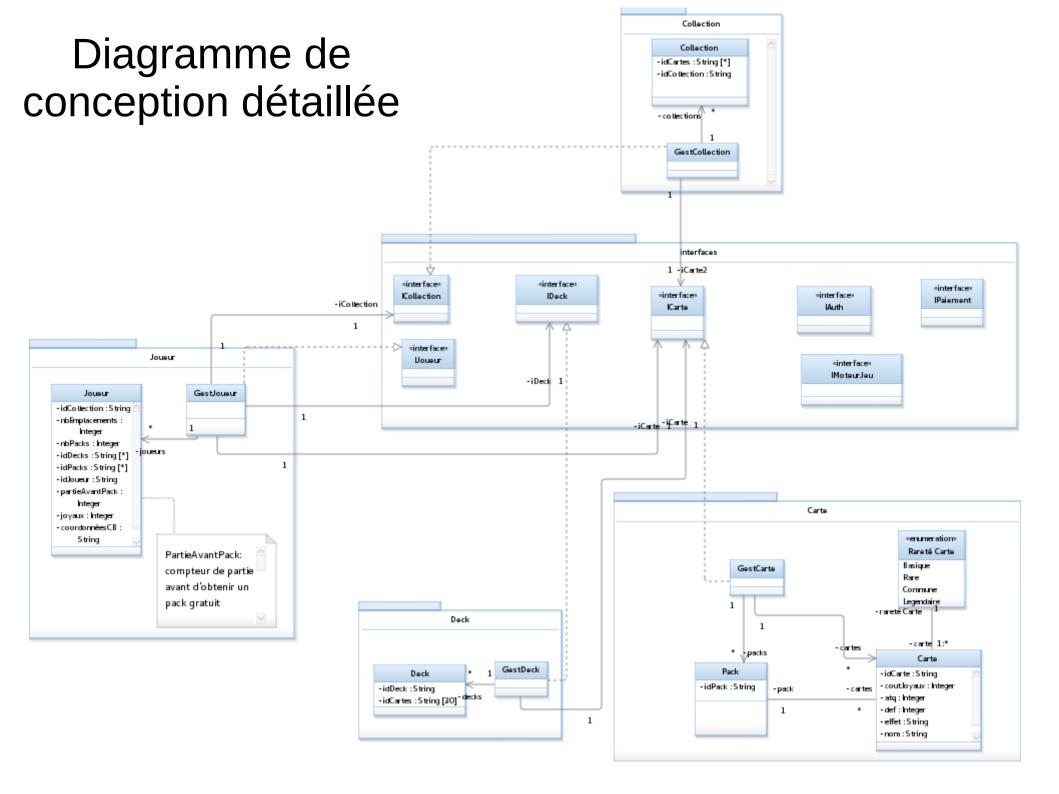
```
assertTrue(sut.ajoutCarte("carteO1",42,5,10,"Augmente l'atq des cartes de type dragon", "Dragon roi", "Il fait peur", "LEGENDAIRE") == true);
assertTrue(sut.ajoutCarte("carteO2",4,5,10,"incapacite un adversaire pendant un tour", "Dragon du lait", "Il fait rire", "COMMUNE") == true);
assertTrue(sut.ajoutCarte("carteO3",52,5,10,"degats des sorts +1, vous piochez une carte", "Dragon azur", "Il est regretté", "RARE") == true);
assertTrue(sut.ajoutCarte("carteO4",22,5,10, "reduit le coût des cartes de type dragon de 1", "Dragon reine", "elle fait peur", "LEGENDAIRE") == true);
assertTrue(sut.getRandomCarteRare()=="carteO3");
```

Factory

```
createNominal(){
      IMoteurJeu mj = createMoteurJeu();
      IPaiement p = createPaiement();
      IAuth auth = createCAuth();
      ICarte carte = createCCarte();
      ICollection coll = createCCollection(carte);
      IDeck deck = createCDeck(carte);
      IJoueur joueur = createCJoueur(coll,deck,carte);
      IHM ihm = createCIHM(joueur,coll,deck,p,auth,mj,carte);
```

StoneHearthComposantFactory + createCAuth (): IAuth + createCCarte () : ICarte + createCCarteBouchon (): ICarte + createCCollection (carte : ICarte) : ICollection + createCColletionBouchon (): ICollection + createCDeck (carte : ICarte) : IDeck + createCDeckBouchon () : IDeck + createCIHM (joueur : Doueur, collection : ICollection, deck : IDeck, paiement : IPaiement, auth : IAuth, moteur : IMoteurJeu, carte : ICarte): CIHM + createCloueur (collection : ICollection, deck : IDeck, carte : ICarte) : Doueur + createCMoteurJeu () : IMoteurJeu + createCPaiement (joueur : Doueur) : IPaiement + createTestCarte (carte : ICarte) + createTestJoueur (joueur : Joueur) + createNominal ()

Conception détaillée



Conclusion