

Projet : StoneHearth

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

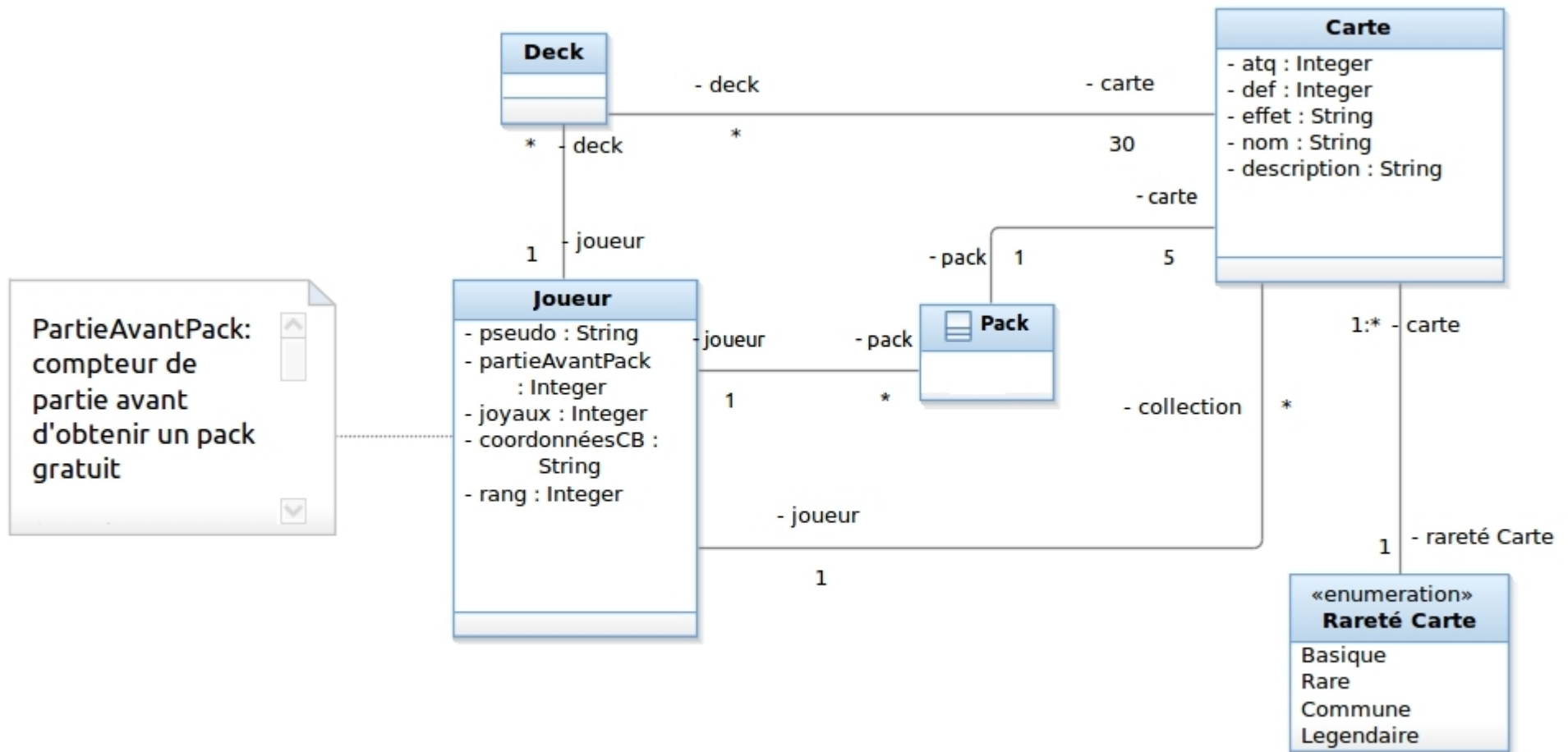
SnowStorm Ltd

Sommaire

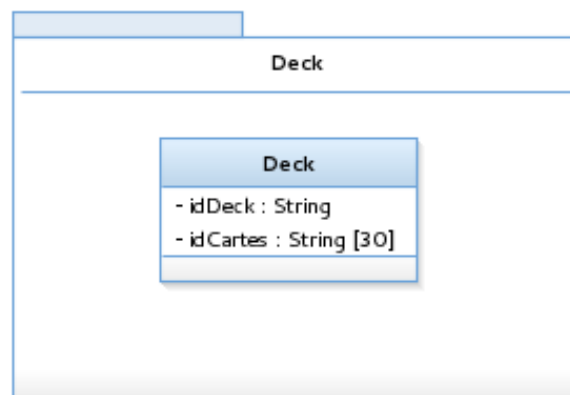
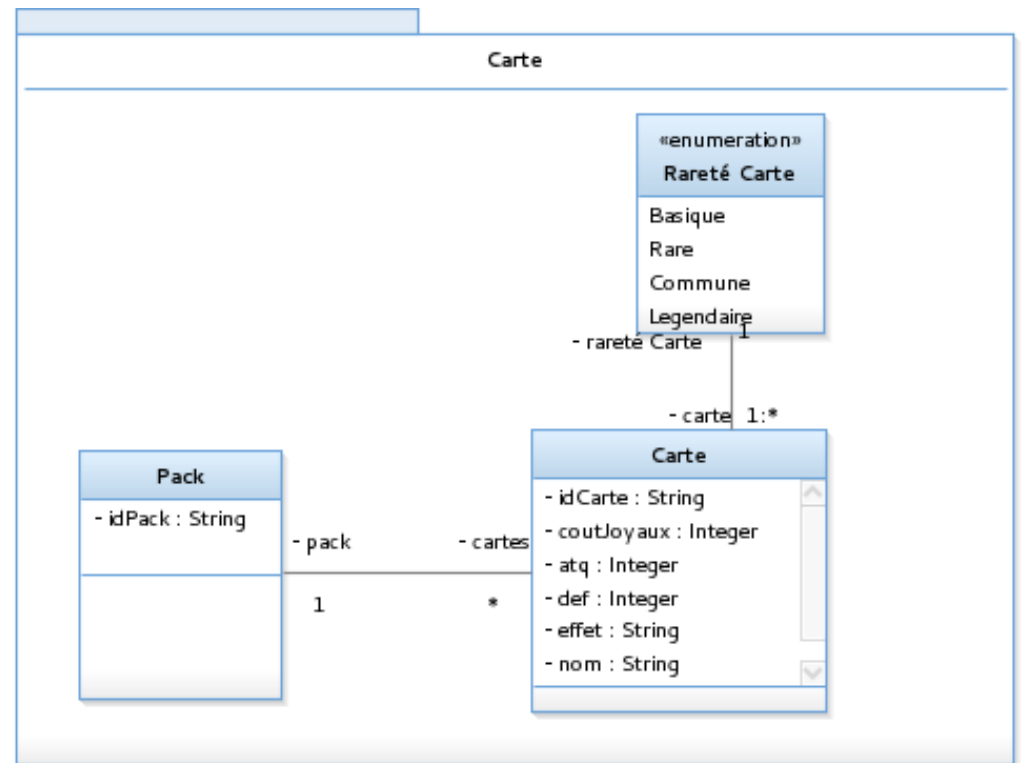
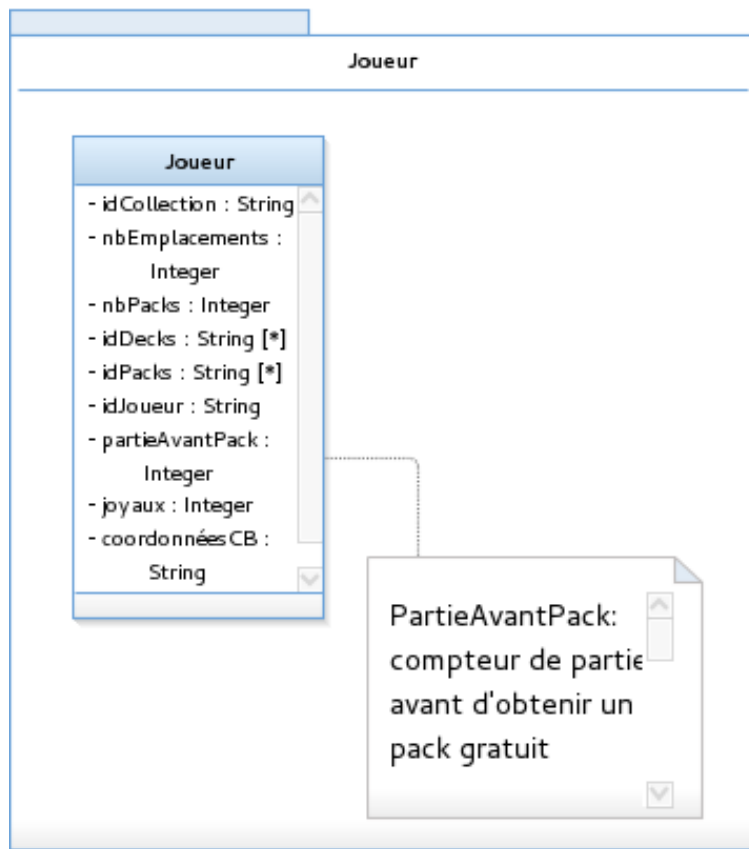
- Introduction
- Conception architecturale
 - Découpe structurelle
 - Interfaces et signatures
 - Diagramme de séquence inter-composants
- Tests d'intégration
- Factory
- Conception détaillée
- Conclusion

Conception architecturale

Rappel du diagramme de classe



Découpe structurelle



Interfaces et signatures

interfaces

«interface»
IMoteurJeu

+ jouerPartie (idJoueur1 : String, idDeck, idJoueur2 : String)

«interface»
IDeck

+ 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)
- verifSansDoubleton (idCarte : String, idDeck : String) : Boolean

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

«interface»
ICollection

+ ouvrirPack (idCollection : String, idJoueur)
+ verifCarteDansCollection (idCarte : String, idCollection : String) : Boolean
+ getIdCollection (idJoueur : String) : String
+ ajouterCarteDansCollection (idCollection : String, idCarte : String)
+ afficherCartesCollection (idCollection : String)
+ ajouteCartesDansCollection (idCollection : String, listeCartes : String [*])

«interface»
ICarte

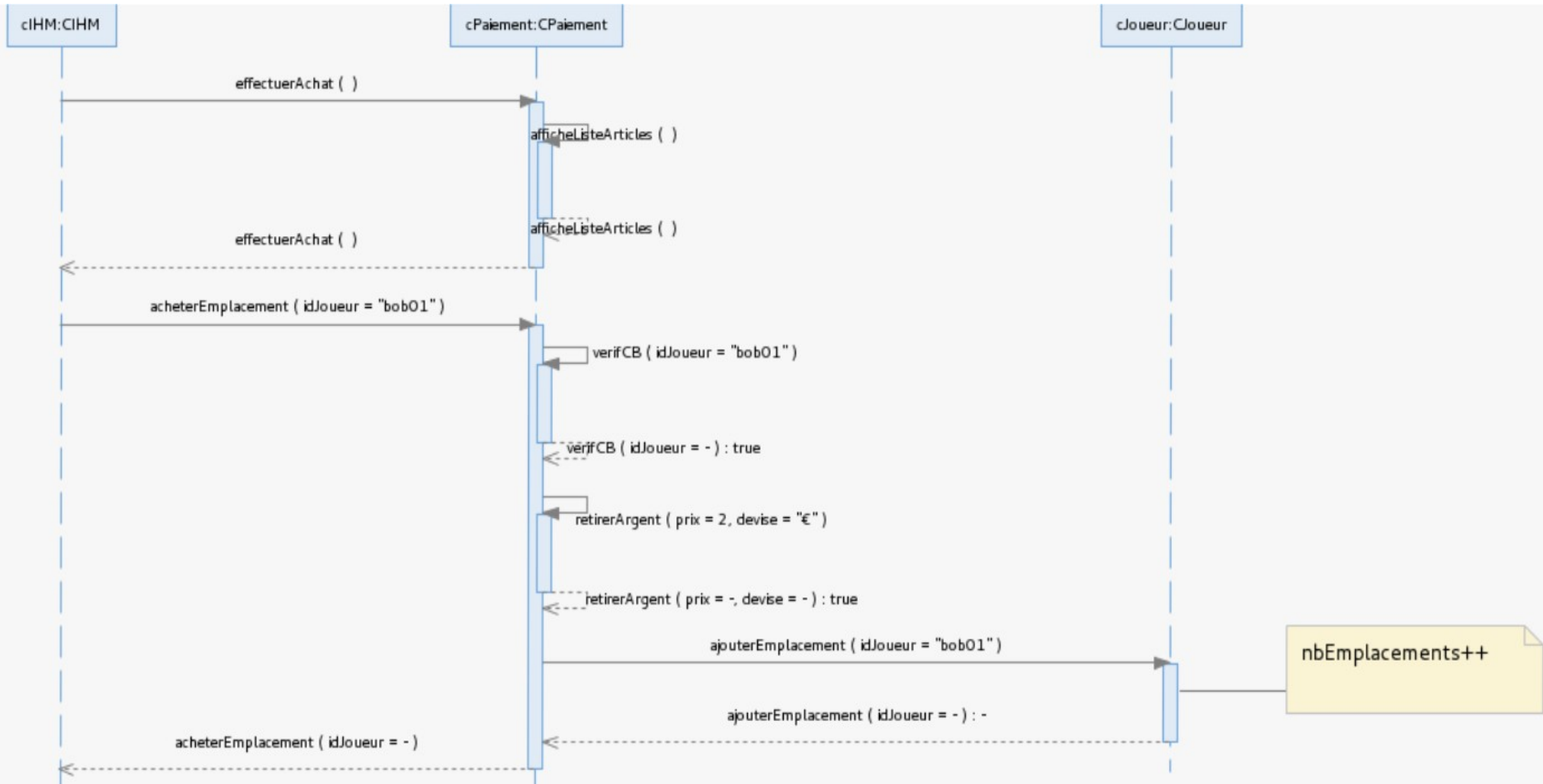
+ getcoutJouaux (idCarte : String)
+ getCarte (nomCarte : String) : String
+ getRandomCarteRare () : String
+ getRandomCarte () : String [4]
- afficherToutesLesCartes ()
+ ajoutCarte (idCarte : String, rarete : String, coutJouaux : String, atq : Integer, def : Integer, effet : String, nom : String, description : String) : boolean

«interface»
IJoueur

- afficheCartesJoueur ()
+ ajouterEmplacement (idJoueur : String) : boolean
+ ajouterPack (idJoueur : String)
+ getCollection () : String
+ verifPack (idJoueur : String) : Boolean
+ retraitJouaux (jouaux : Integer) : Integer
+ verifSoldJouaux (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

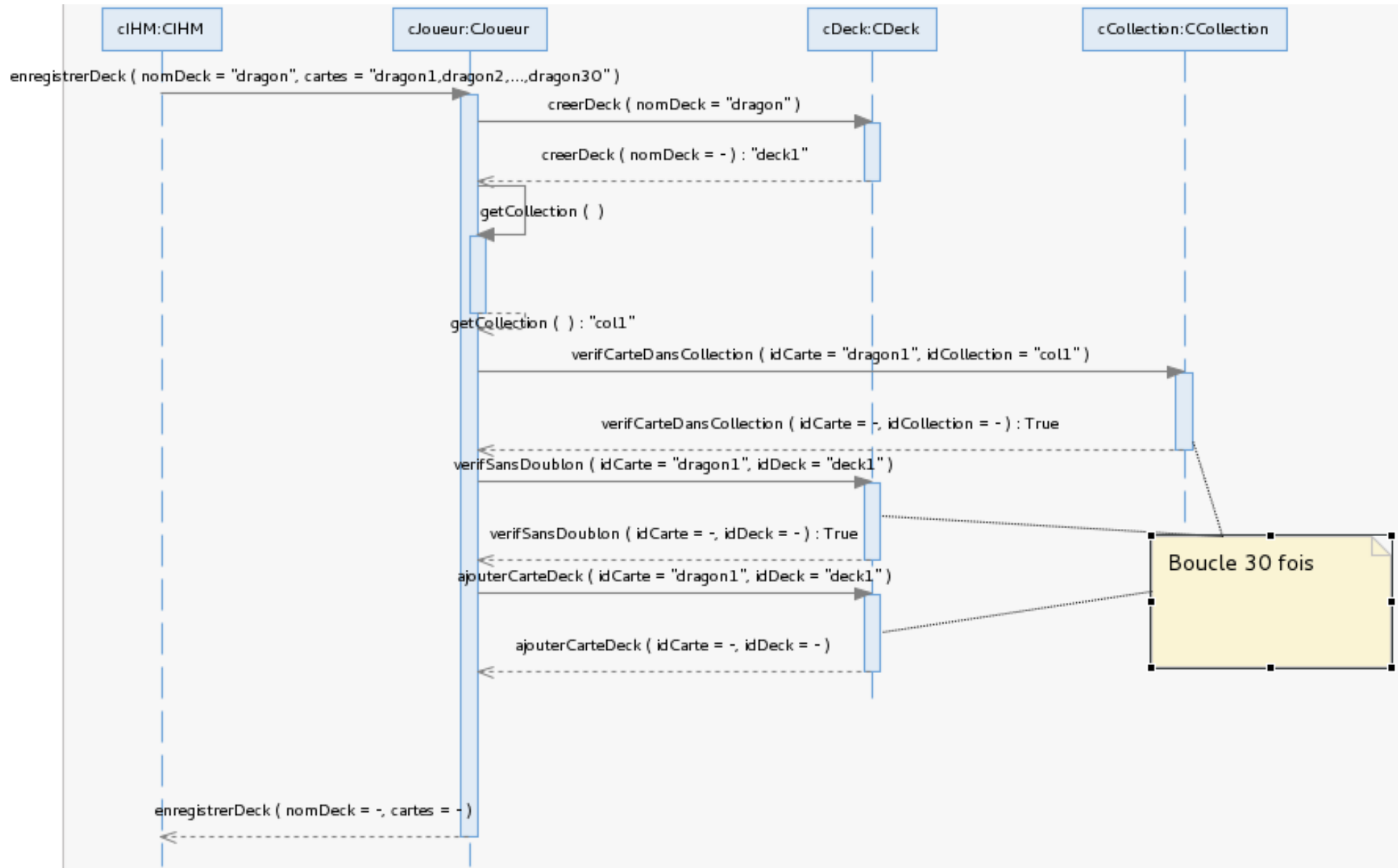
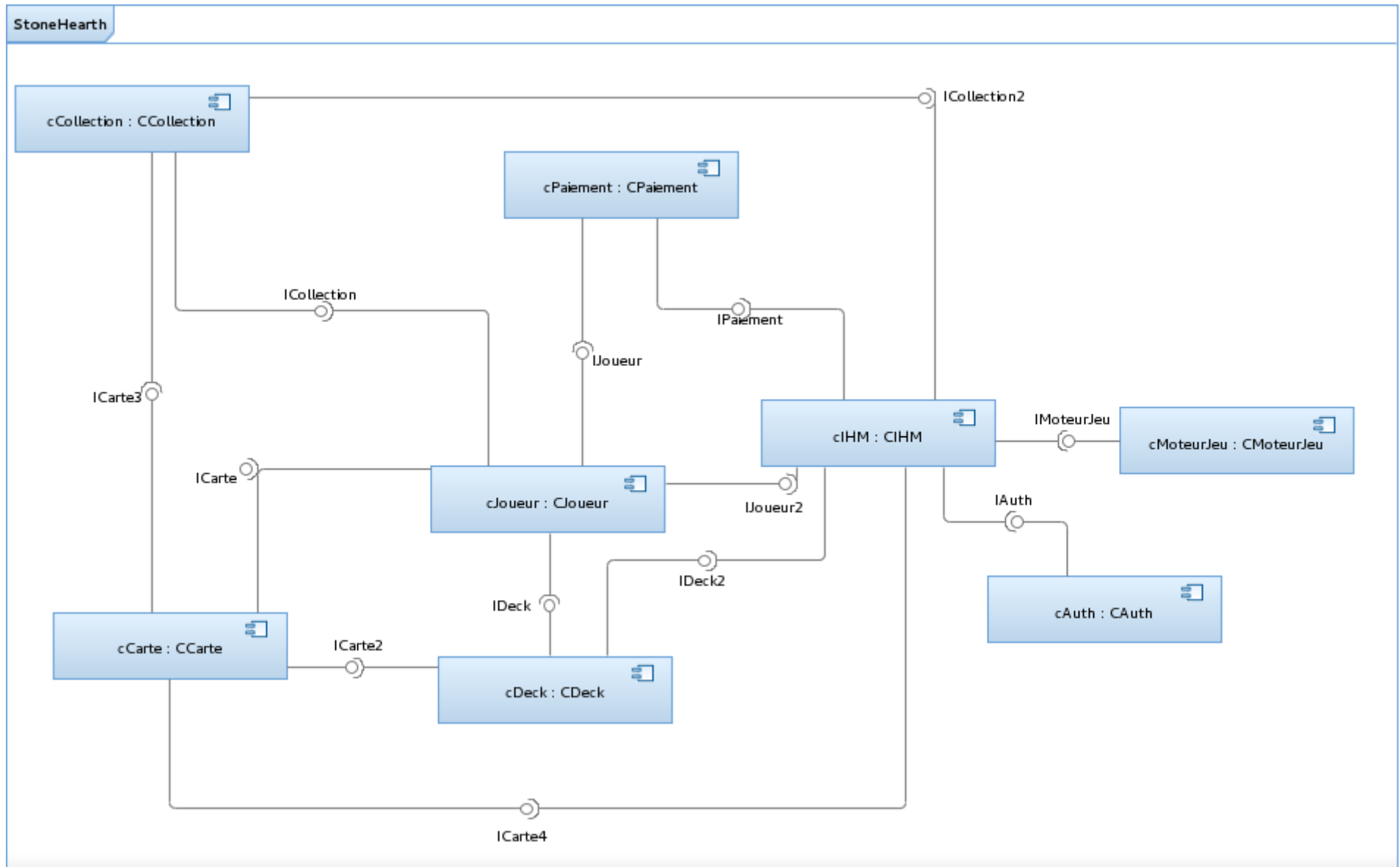
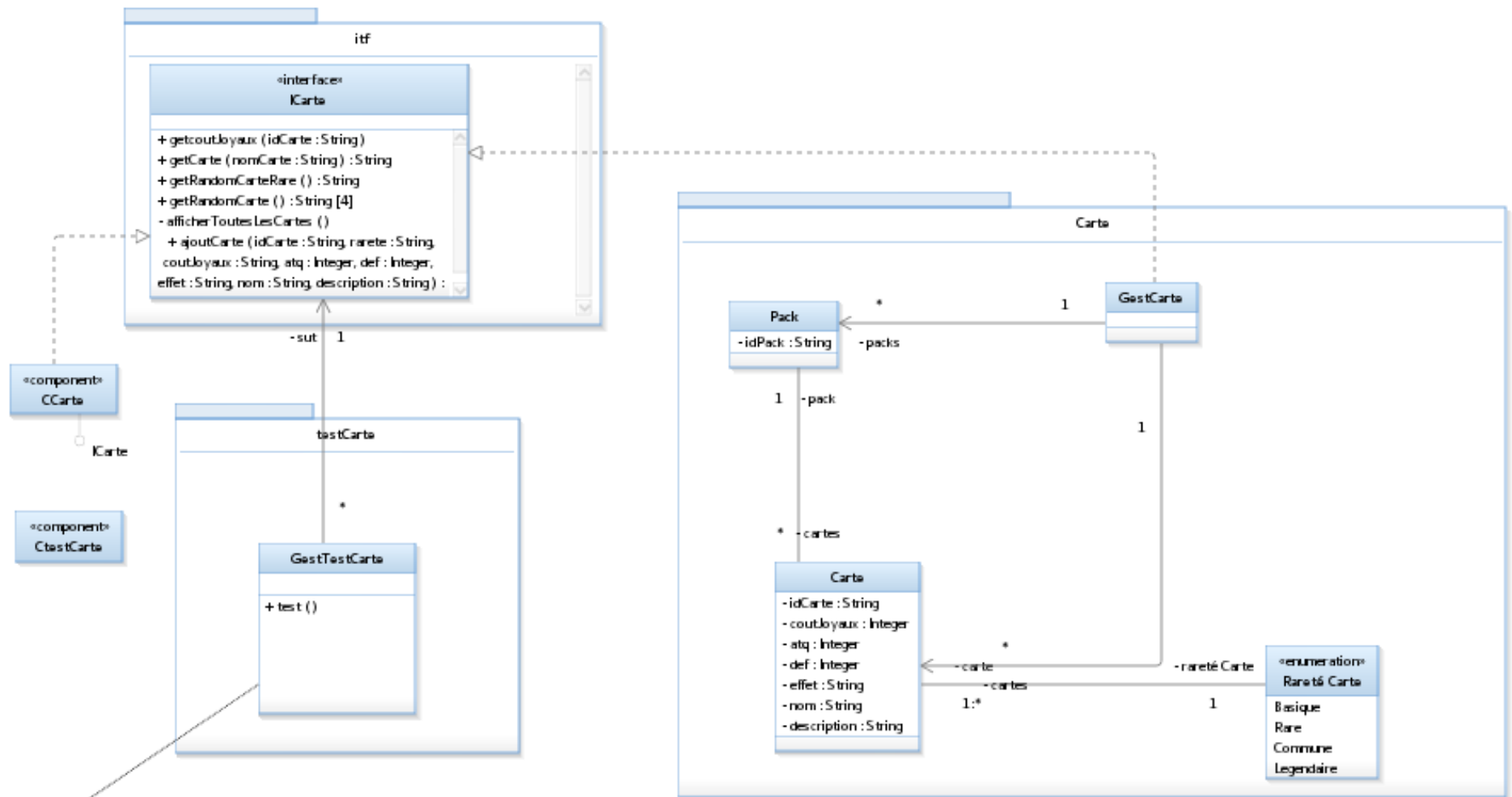


Diagramme de structure interne



Tests d'intégration

CCarte

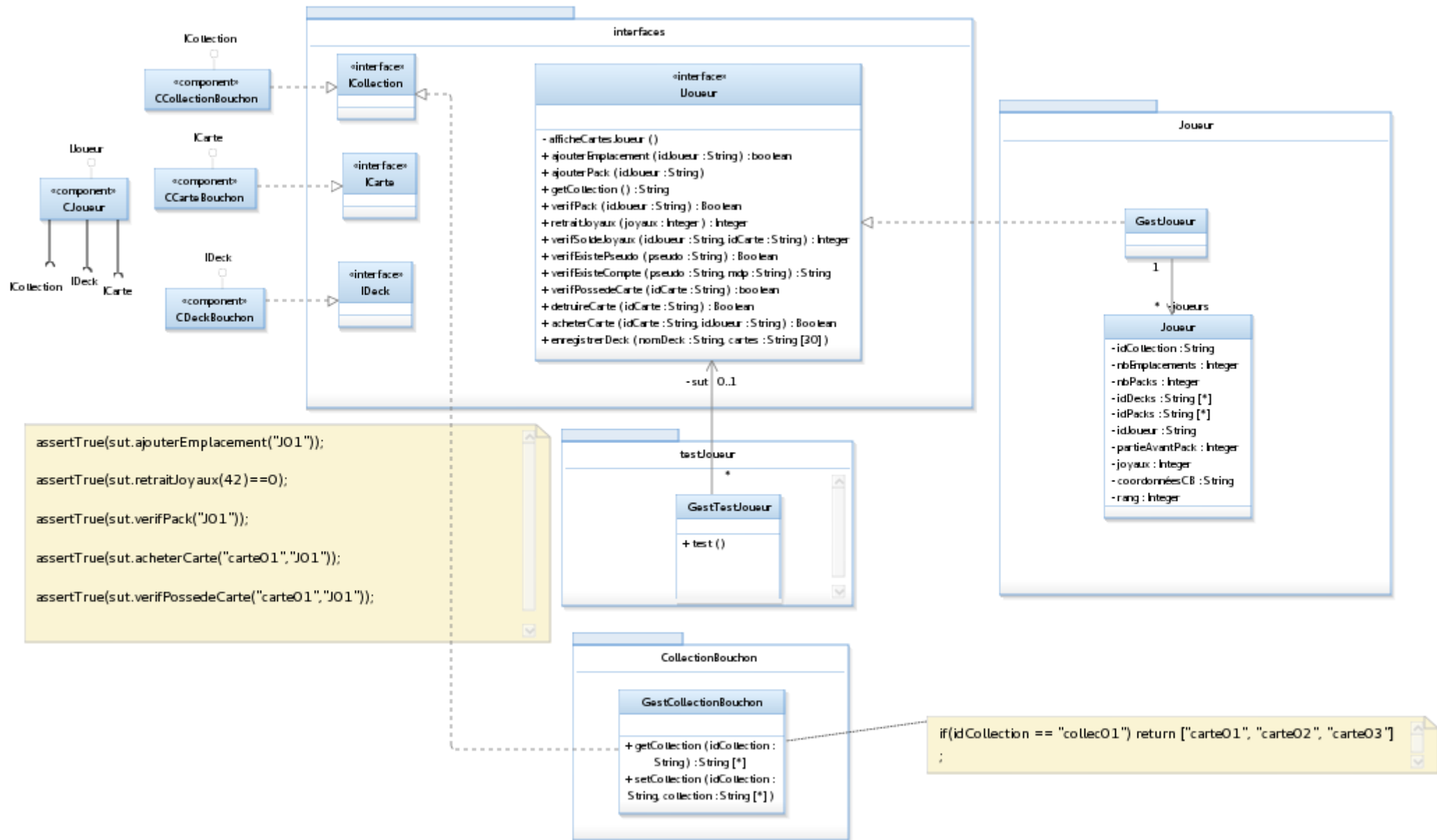


```
assertTrue(sut.ajoutCarte("carte01",42,5,10,"Augmente l'atq des cartes de type dragon","Dragon roi", "Il fait peur","LEGENDAIRE") == true);
assertTrue(sut.ajoutCarte("carte02",4,5,10,"incapacite un adversaire pendant un tour","Dragon du lait", "Il fait rire","COMMUNE") == true);
assertTrue(sut.ajoutCarte("carte03",52,5,10,"degats des sorts +1, vous piochez une carte","Dragon azur", "Il est regretté","RARE") == true);

assertTrue(sut.ajoutCarte("carte04",22,5,10, "reduit le coût des cartes de type dragon de 1","Dragon reine", "elle fait peur","LEGENDAIRE") == true);
;

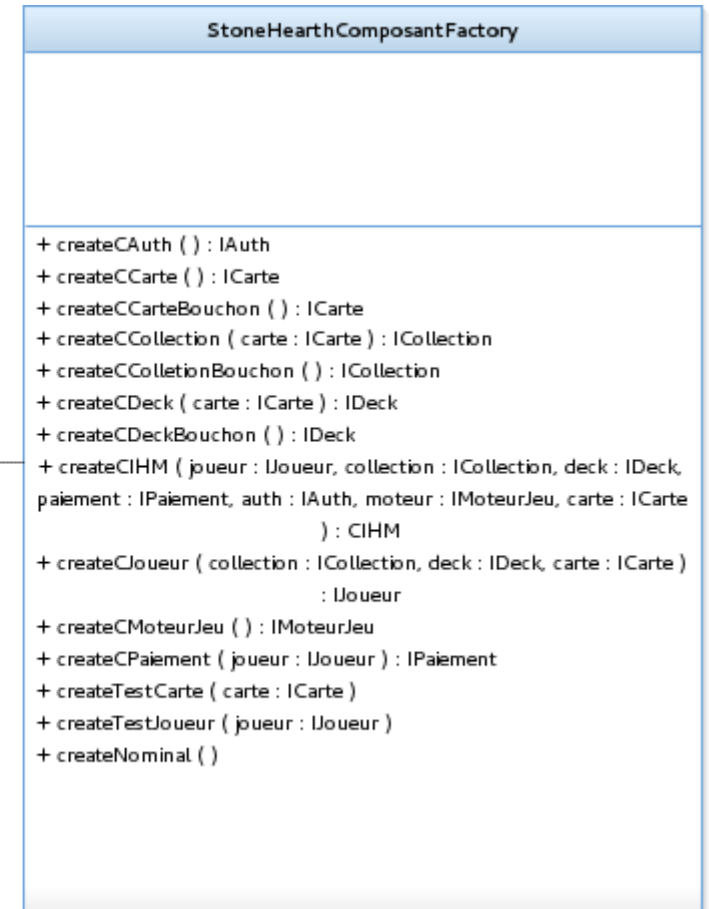
assertTrue(sut.getRandomCarteRare() == "carte03");
```

CJoueur



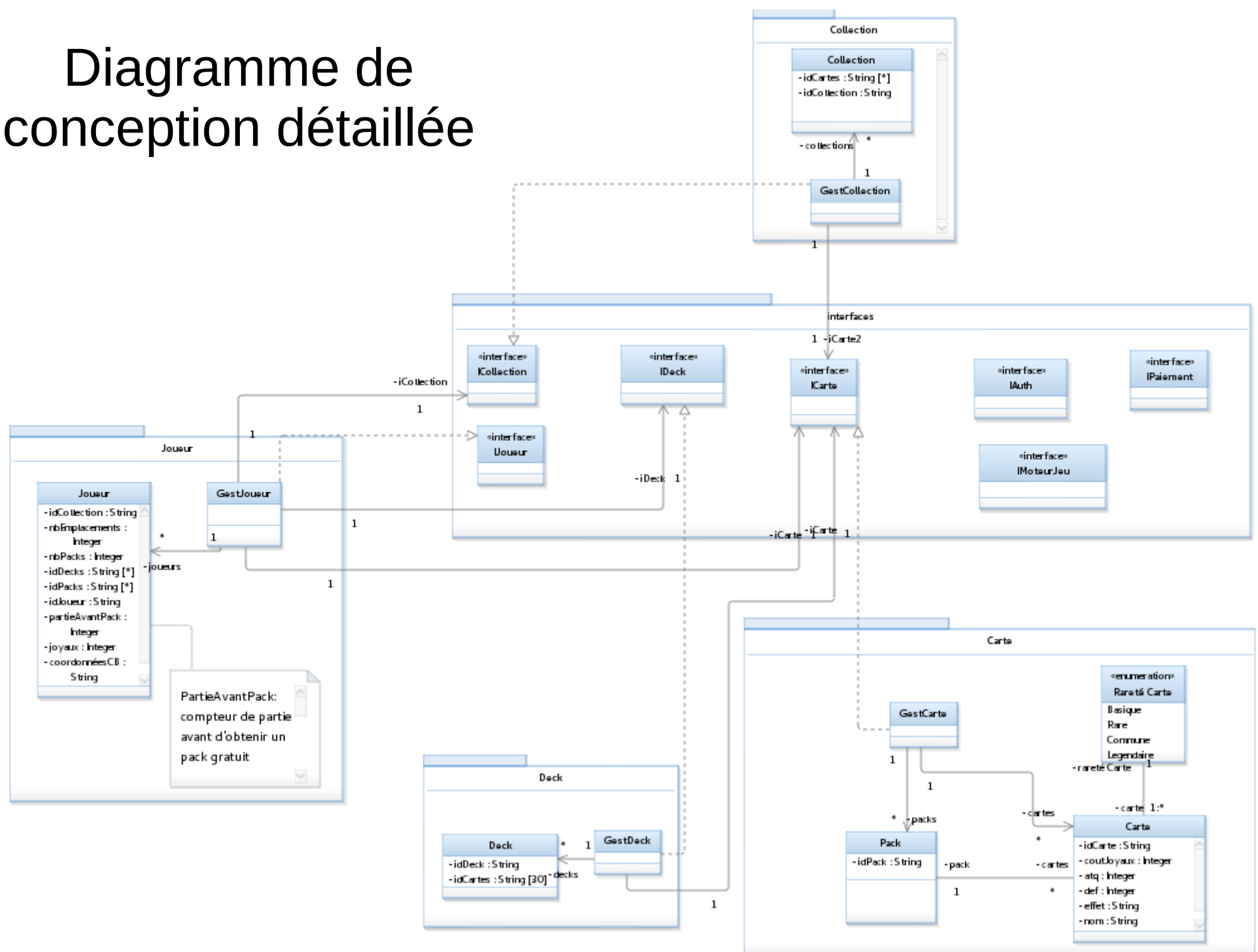
Factory

```
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);  
  
}
```



Conception détaillée

Diagramme de conception détaillée



Conclusion