Projet : St  neHearth

SnowStorm ltd

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

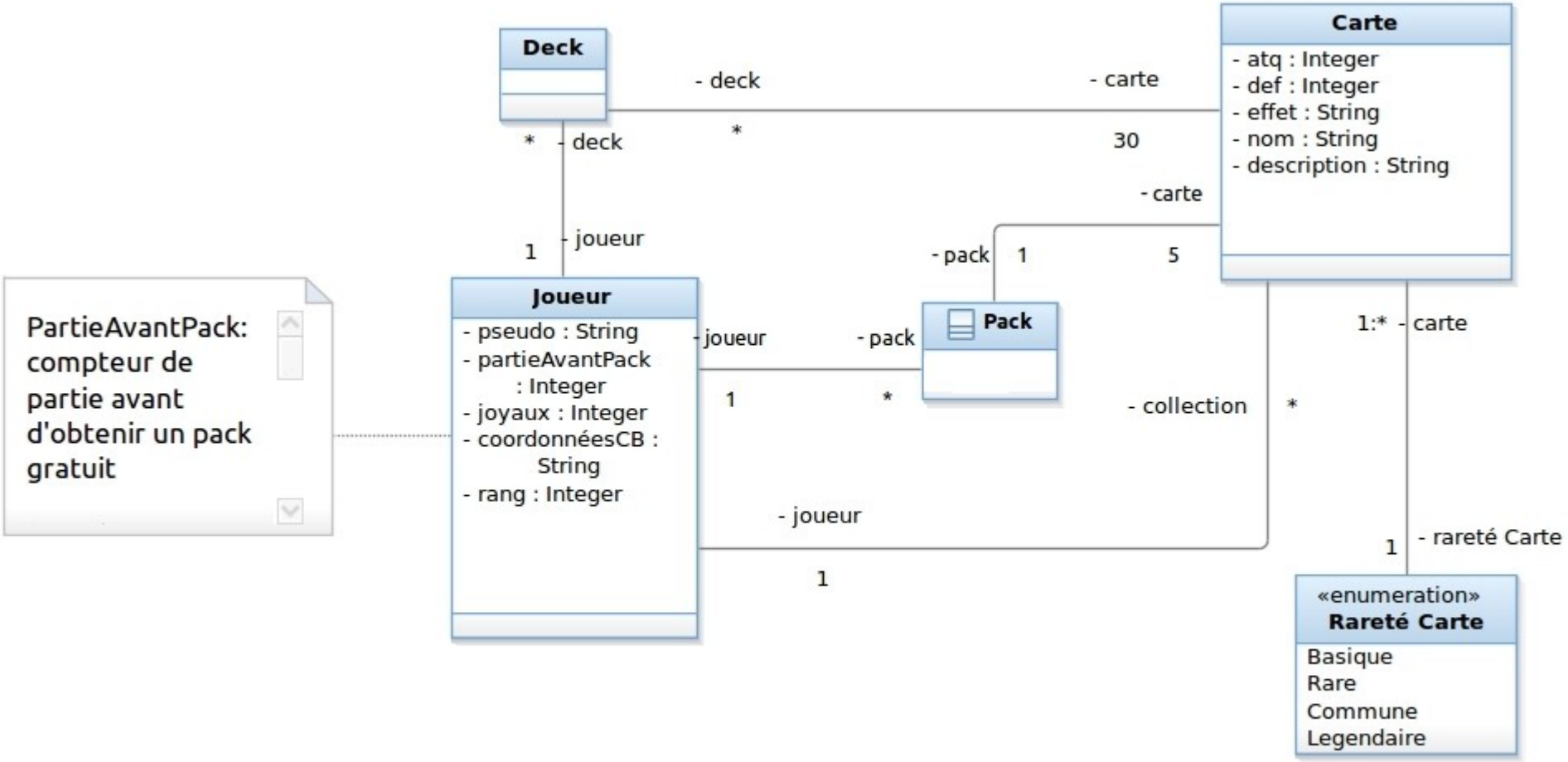
UPMC
SORBONNE UNIVERSITÉS

Sommaire

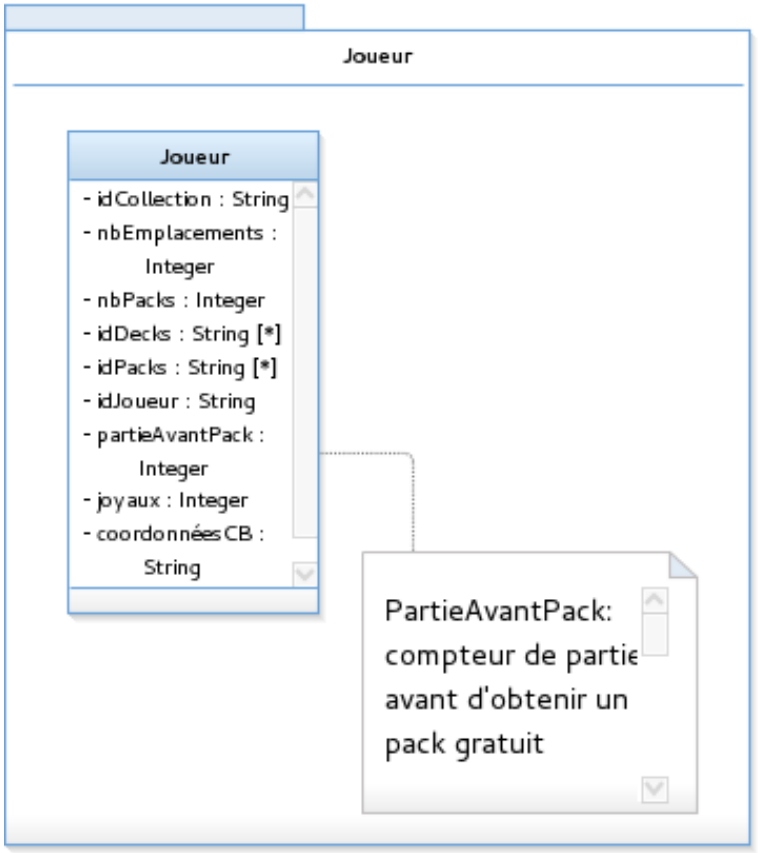
- Introduction
- Conception architecturale
  - Découpe structurelle
  - Interfaces et signatures
  - Diagramme de séquence inter-composants
- Tests d'intégration
- Factory
- Conception détaillée
- Conclusion

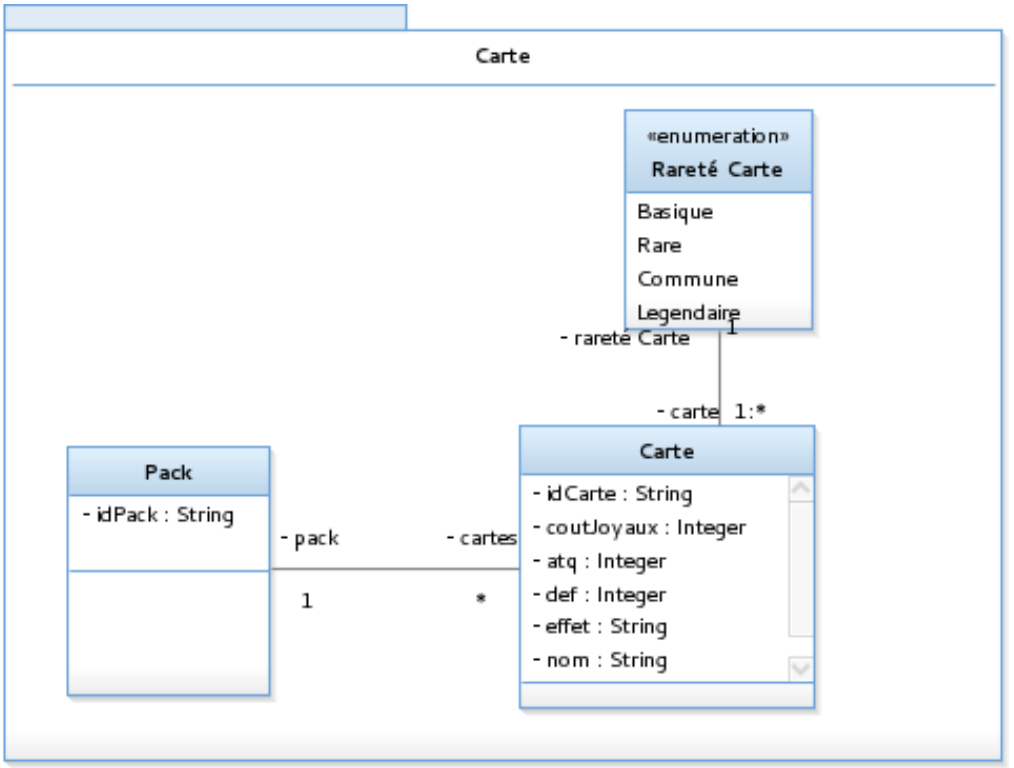
Conception architecturale

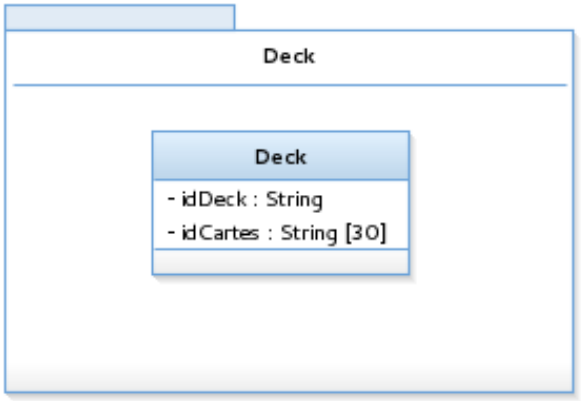
Rappel du diagramme de classe



Découpe structurelle







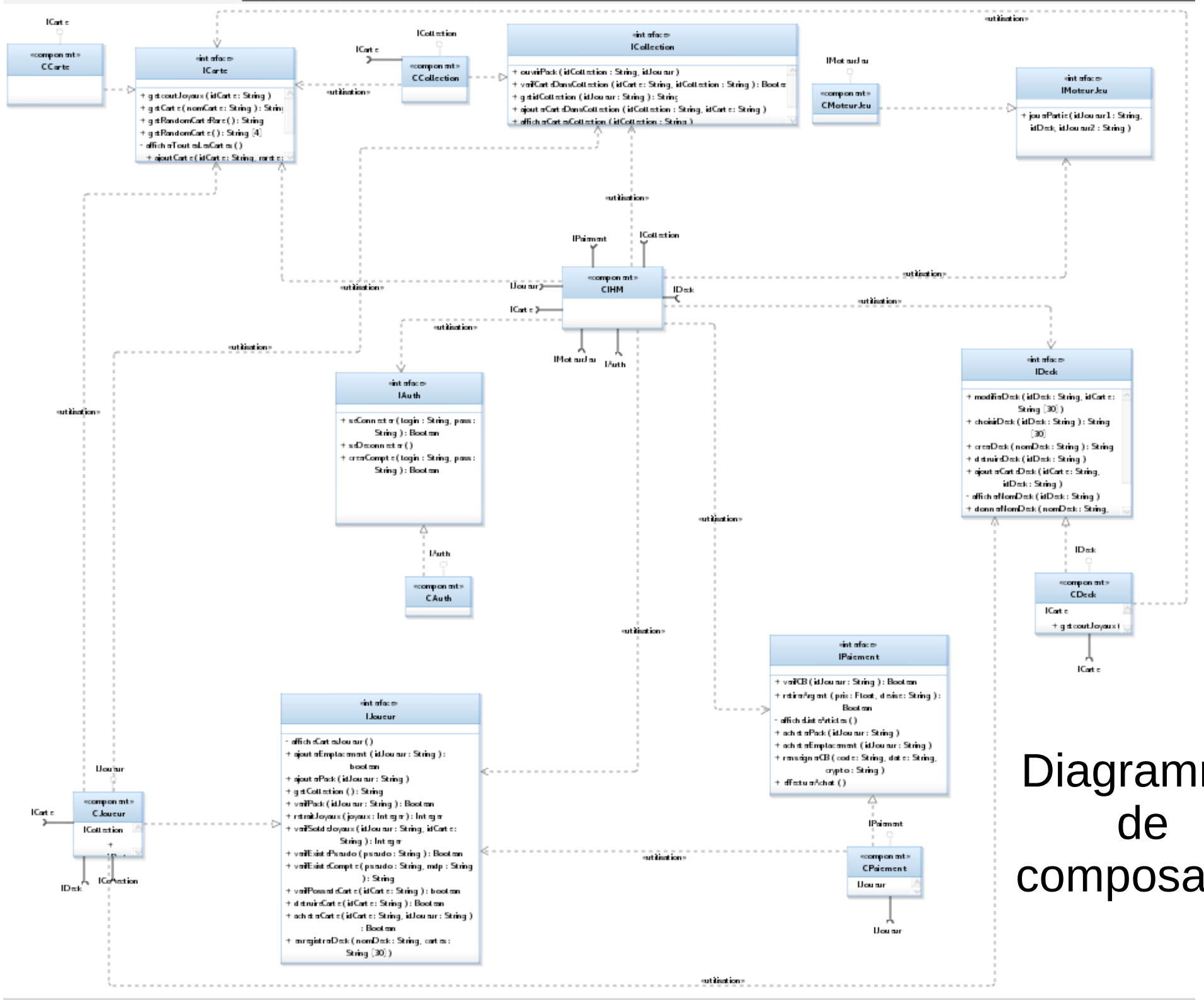


Diagramme de composants

Interfaces et signatures

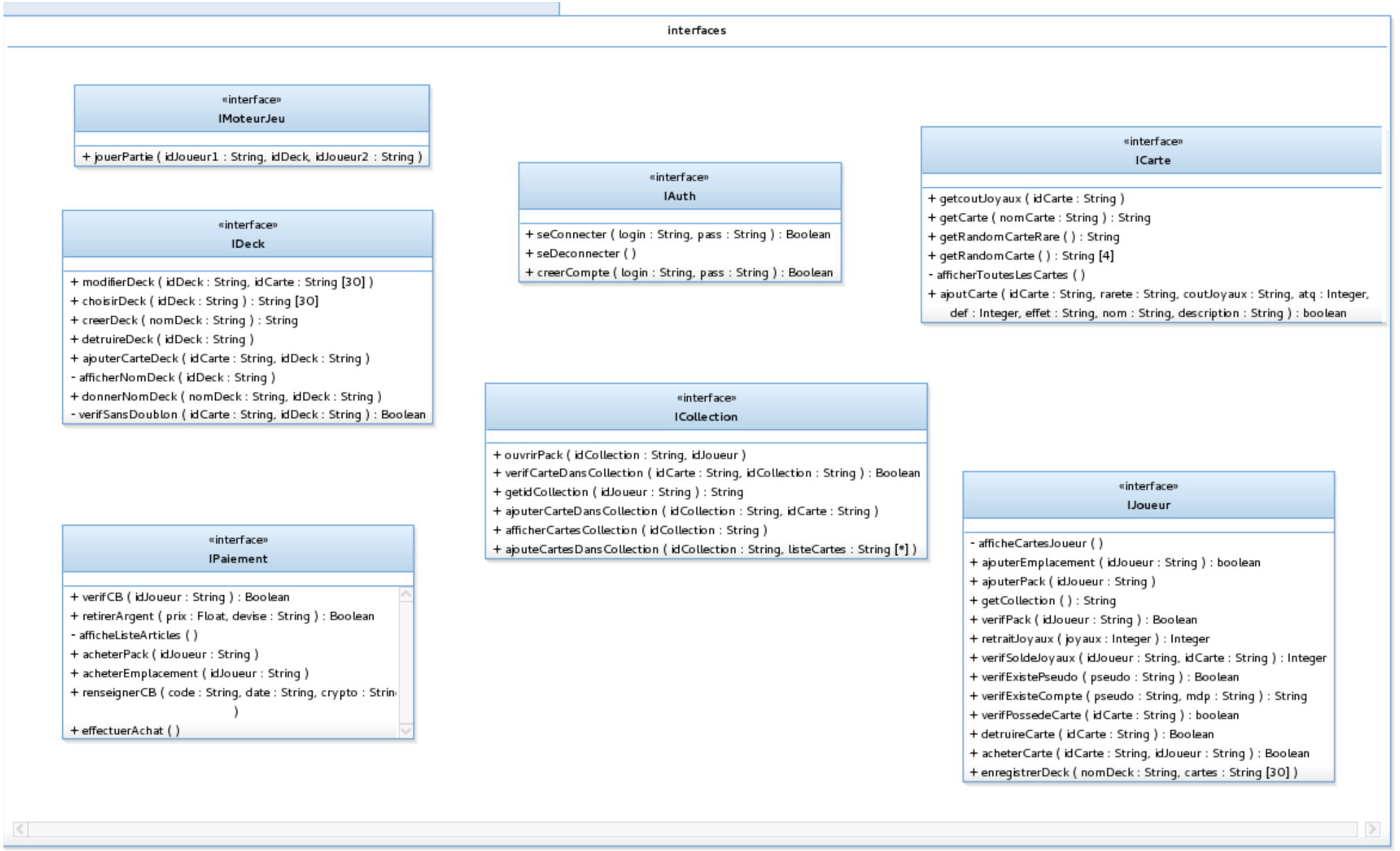
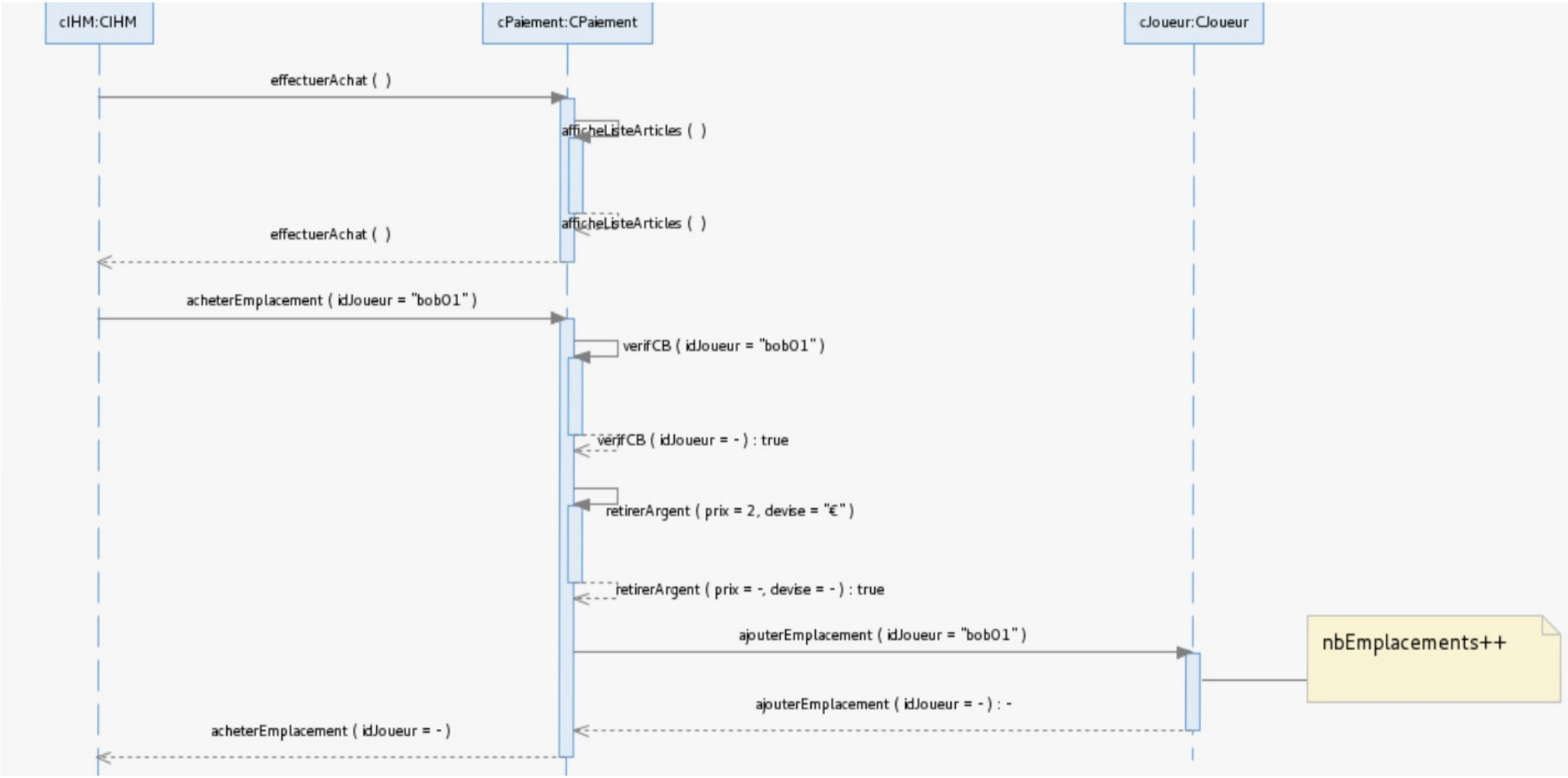


Diagramme de séquence inter-composants

Effectuer Achat



Gérer Deck

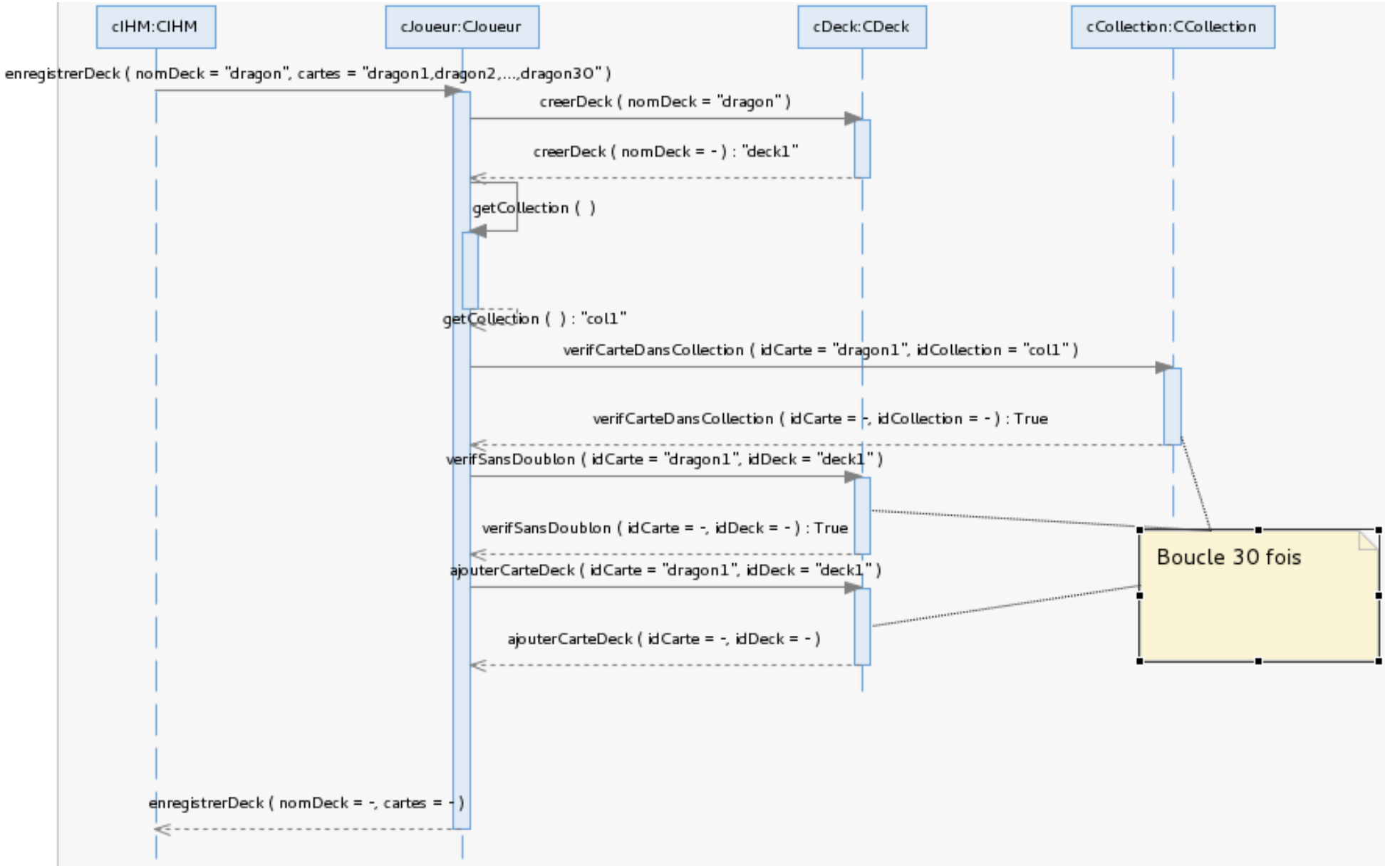
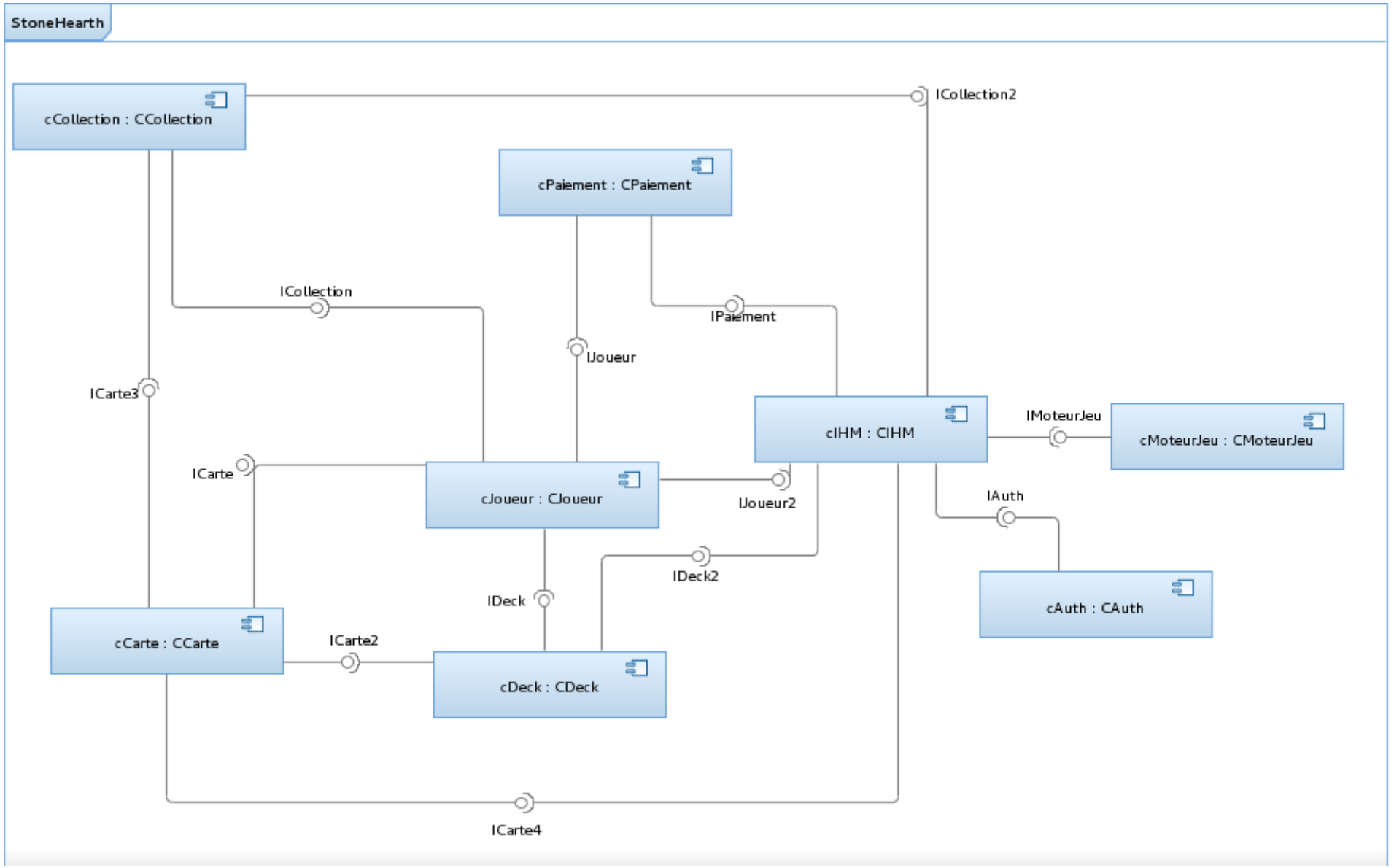
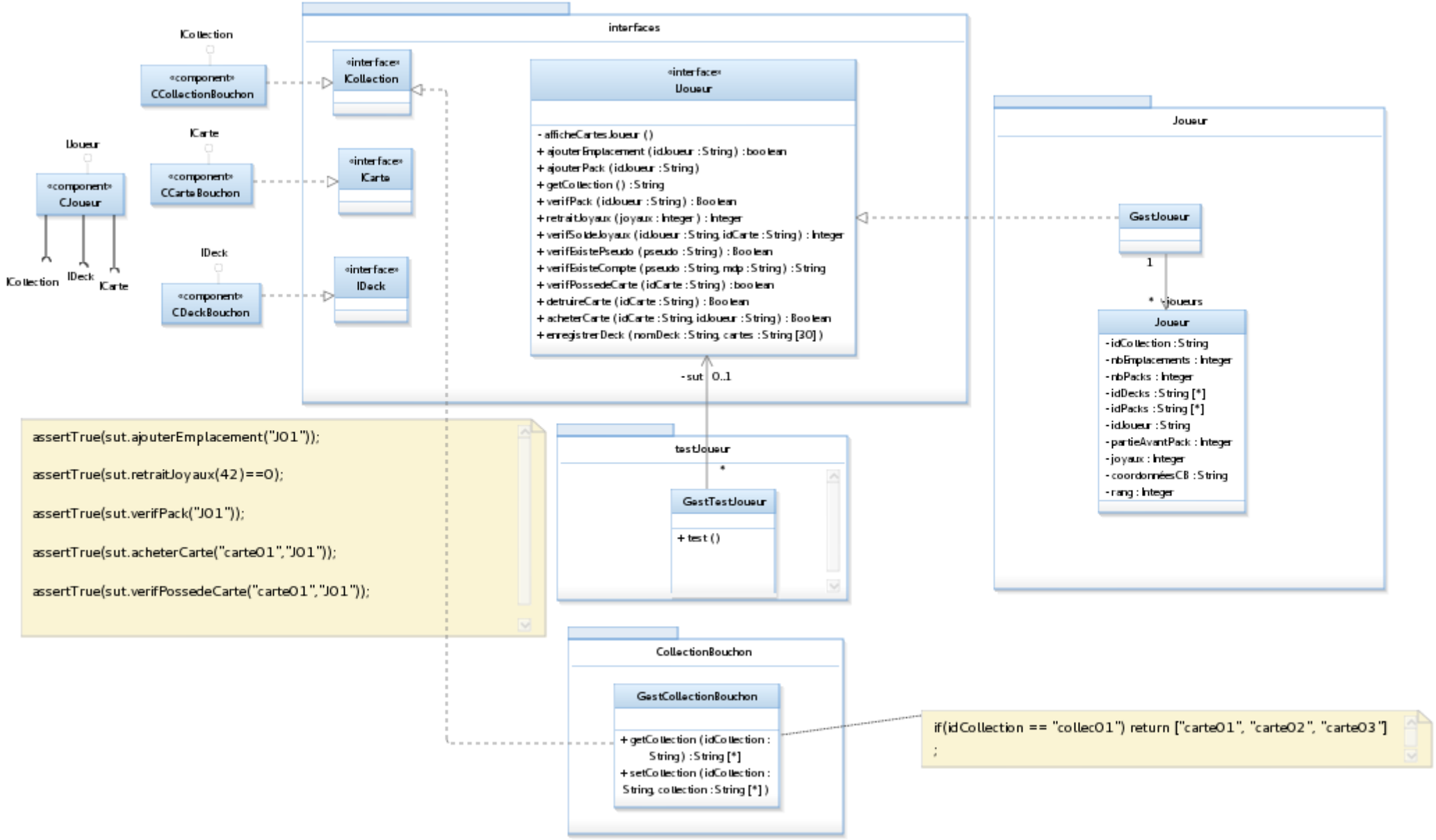

cIHM:CIHM
cJoueur:CJoueur
cDeck:CDeck
cCollection:CCollection
enregistrerDeck (nomDeck = "dragon", cartes = "dragon1,dragon2,...,dragon30")
creerDeck (nomDeck = "dragon")
creerDeck (nomDeck = -) : "deck1"
getCollection ( )
getCollection ( ) : "col1"
verifCarteDansCollection (idCarte = "dragon1", idCollection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

Diagramme de structure interne

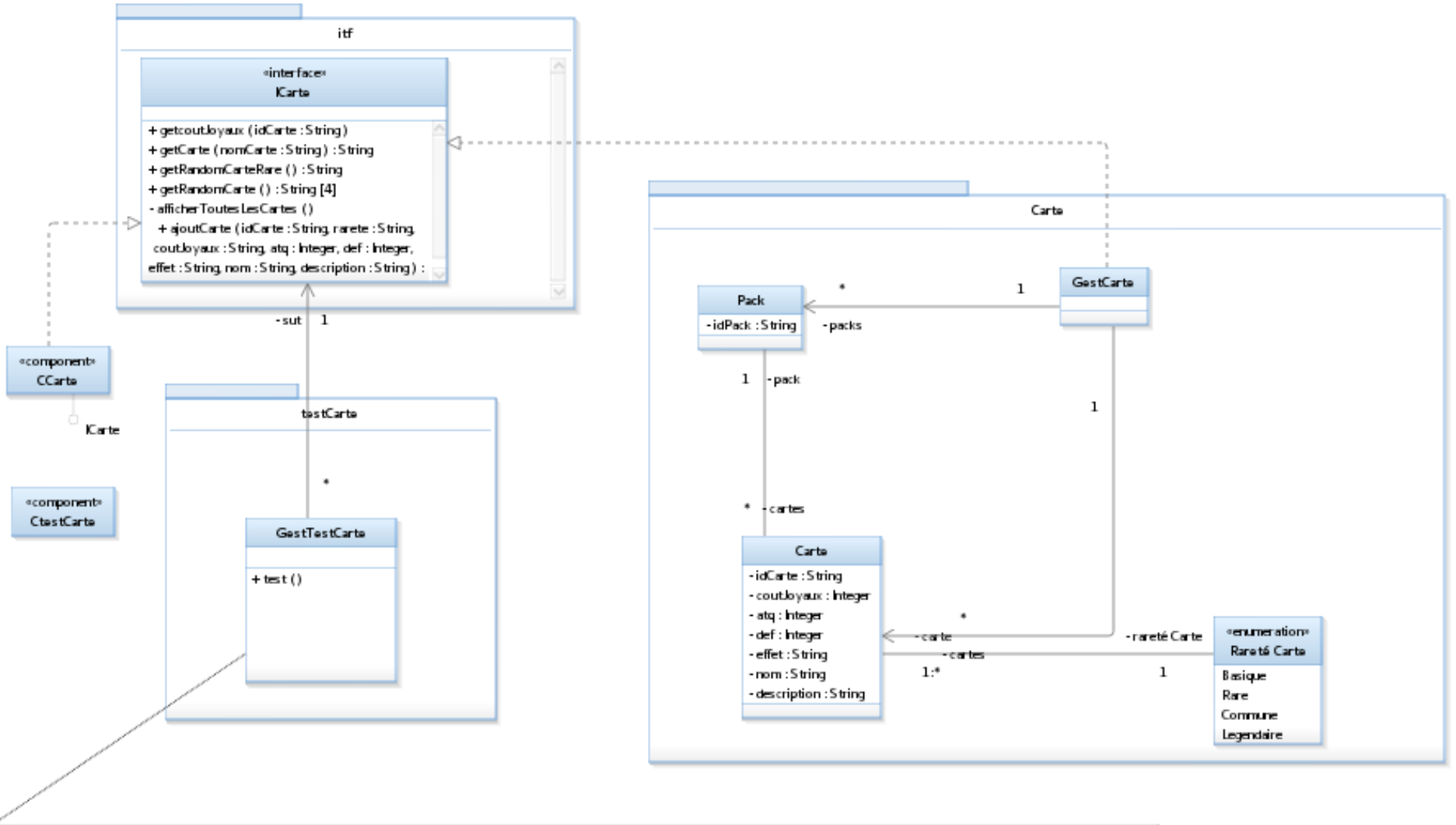


Tests d'intégration

CJoueur


ICollection
«component»
CCollectionBouchon
IJoueur
«component»
CJoueur
ICarte
«component»
CCarteBouchon
IDeck
«component»
CDeckBouchon
ICollection
IDeck
ICarte
interfaces
«interface»
ICollection
«interface»
ICarte
«interface»
IDeck
«interface»
IJoueur
- 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])
- sut
0..1
Joueur
GestJoueur
1
* joueurs
Joueur
- idCollection : String
- nbEmplacements : Integer
- nbPacks : Integer
- idDecks : String [*]
- idPacks : String [*]
- idJoueur : String
- partieAvantPack : Integer
- joyaux : Integer
- coordonnéesCB : String
- rang : Integer
testJoueur
*
GestTestJoueur
+ test ()
assertTrue(sut.ajouterEmplacement("J01"));
assertTrue(sut.retraitJoyaux(42)==0);
assertTrue(sut.verifPack("J01"));
assertTrue(sut.acheterCarte("carte01","J01"));
assertTrue(sut.verifPossedeCarte("carte01","J01"));
CollectionBouchon
GestCollectionBouchon
+ getCollection (idCollection : String) : String [*]
+ setCollection (idCollection : String, collection : String [*])
if(idCollection == "collec01") return ["carte01", "carte02", "carte03"]
;

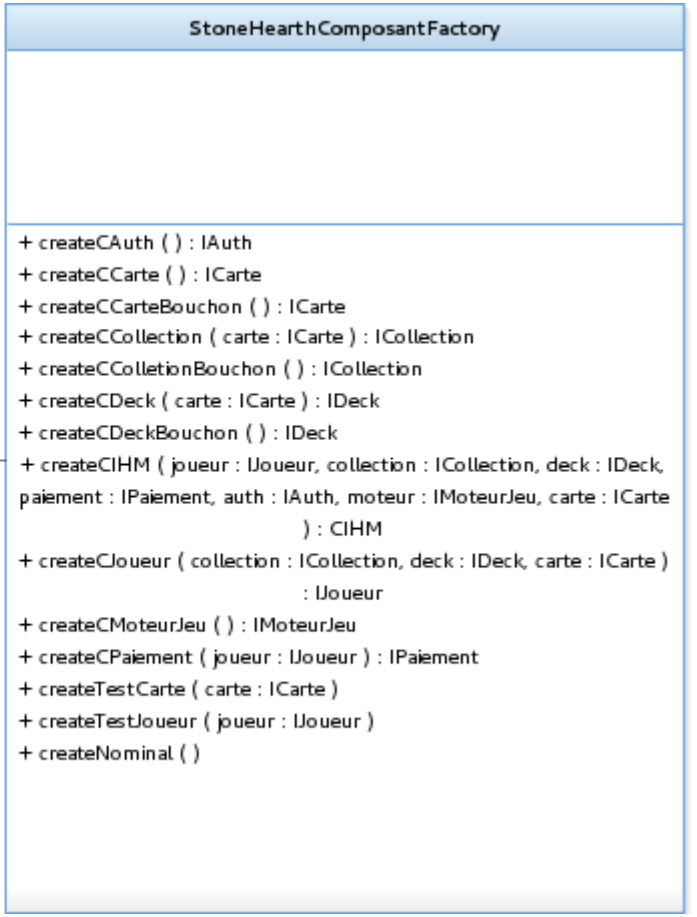
CCarte




```
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");
```

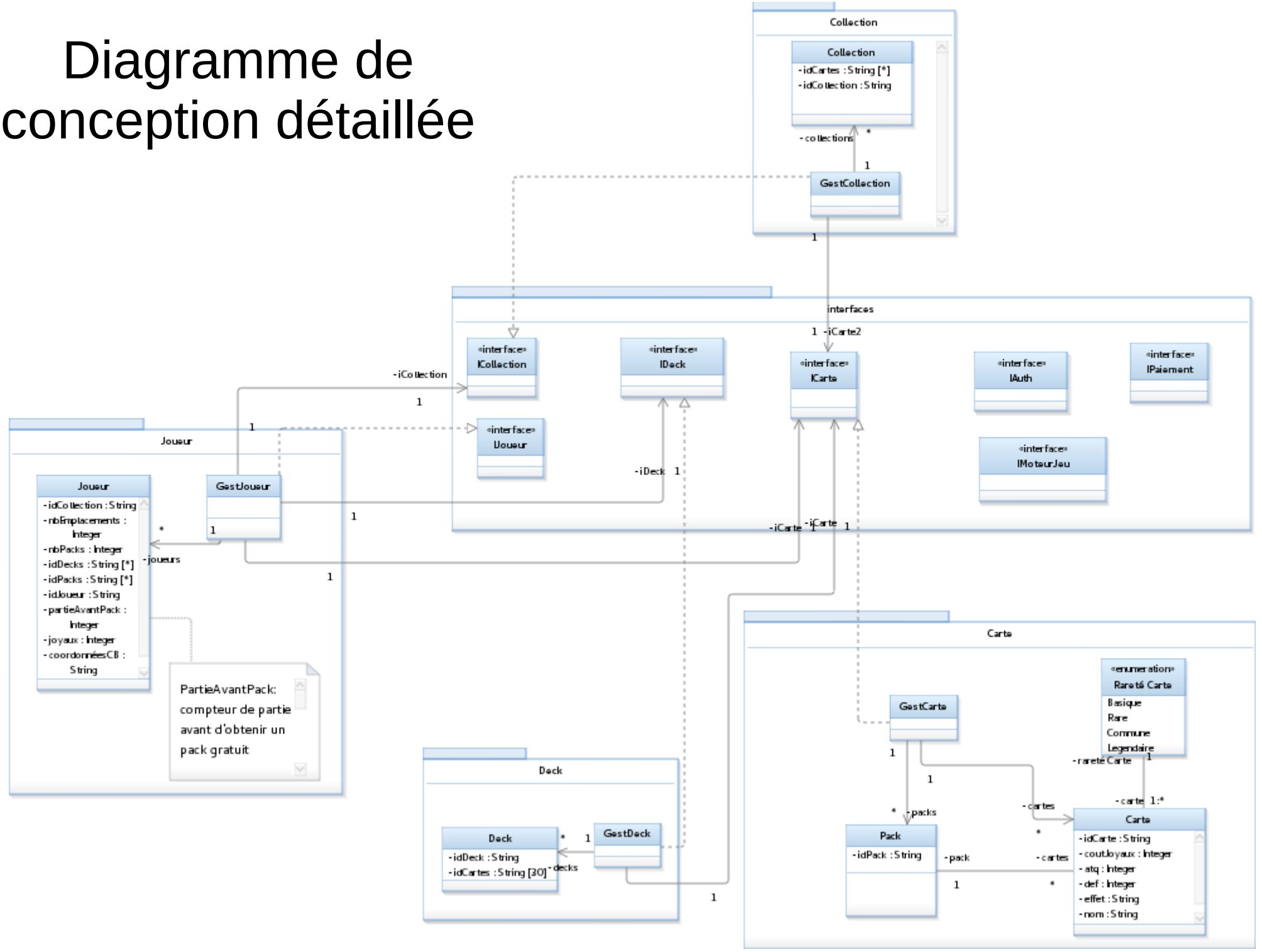
Factory

```
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