Rapport projet long de Technologie Objet

Tiffany Chalier - Rémy Dinissen - Nicolas Drege - Marin Franot -
Cédric Legoupil - Flavie Misarsky - François Testu

Avril 2022

Sommaire

1 Introduction

Au cours de ce projet long nous avons travaillé sur le développement d'un jeu vidéo d'aventure. Ce jeu se nomme "L'île 100 fin" et le but pour le joueur est de s'échapper d'une île déserte. Dans ce rapport nous allons vous expliquer nos différentes étapes de travail, notre avancement et nous présenterons aussi les outils qui nous aidés dans la réalisation de ce projet comme le diagramme UML.

2 Travail préliminaire sur le projet

Avant de commencer à faire le code de notre projet nous avons travaillé, à l'occasion du cours de Méthodes Agiles, sur les principaux objectifs de notre projet et les principales valeurs que nous voulions lui apporter.

2.1 Ce qui définit notre jeu

Pour : Les francophones
Qui souhaite : jouer
Notre produit est : Un jeu d'aventure scénarisé.
Qui : à pour objectif de survivre à une catastrophe et de s'échapper d'une île déserte.
À la différence des : jeux classiques du genre
Il permet de : vivre une multitude de mécaniques et d'histoires différentes.

2.2 Les valeurs de notre projet

Nous avons identifiés les pricipales valeurs de notre projet et nous les avons classés en trois catégories : Must, Should et Could pour déterminer lesquels étaient les plus importantes à réaliser.

2.2.1 Les Must

- Avoir plusieurs scénarios
- Intéragir avec des personnages et des objets
- Pouvoir se déplacer dans le jeu
- Avoir des menus ergonomiques
- Pouvoir sauvegarder sa progression

2.2.2 Les Should

- Avoir différents mini-jeux
- Avoir une temporalité dans le jeu
- Avoir un inventaire

- Avoir des beaux graphismes

2.2.3 Les Could

- Avoir un compteur des morts ud joueurs
- Avoir un compteurs des fins trouvés par le joueur

2.2.4 Graphique

Une fois les valeurs identifiées nous les avons placés sur un graphique récapitulatif qui permet de voir plus précisement si les valeurs répondent à un besoin et/ou sont satisfaisantes pour le joueur.

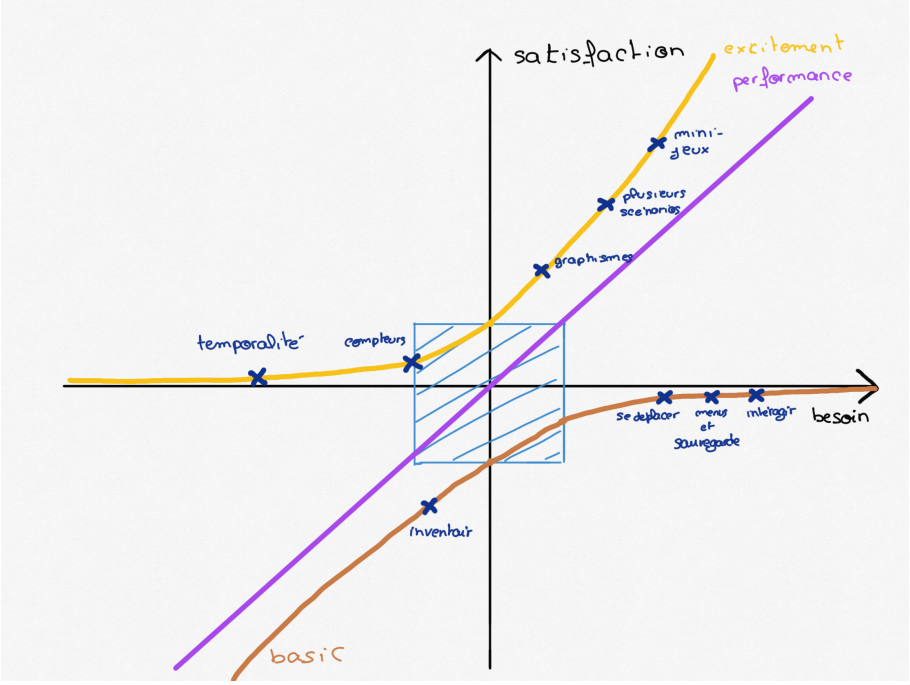

Figure 1: Valeurs du projet

2.2.5 But de nos valeurs

Pour terminer notre travail sur les valeurs du projet nous détaillons les buts de ces valeurs et ce qu'elles apporteraient.
- En tant que joueur je veux me déplacer afin d'avancer dans le scénario, de découvir de nouvelles zones de jeu, de rencontrer de nouveaux personnages et

de trouver de nouveaux objets.
- En tant que joueur je veux avoir plusieurs scénarios afin d'augmenter le durée de vie du jeu.
- En tant que joueur je veux pouvoir sauvegarder afin de pouvoir reprendre la partie ultérierement.
- En tant que joueur je veux pouvoir intéragir avec des personnages afin d'obtenir des informations, d'obtenir des objets et de débloquer des quêtes.
- En tant que joueur je veux avoir un compteur de morts/fins afin de voir ma progression dans le jeu.
- En tant que joueur je veux avoir un inventaire afin d'utiliser et connître les objets.
- En tant que joueur je veux avoir de beaux graphismes afin d'avoir un jeu esthétique et agréable.
- En tant que joueur je veux avoir de nombreux mini-jeux afin e diversifier le gameplay.
- En tant que joueur je veux avoir une temporalité afin de donner du sens à l'enchainement des actions et quêtes.

3 Diagrammes

3.1 Diagramme d'utilisation

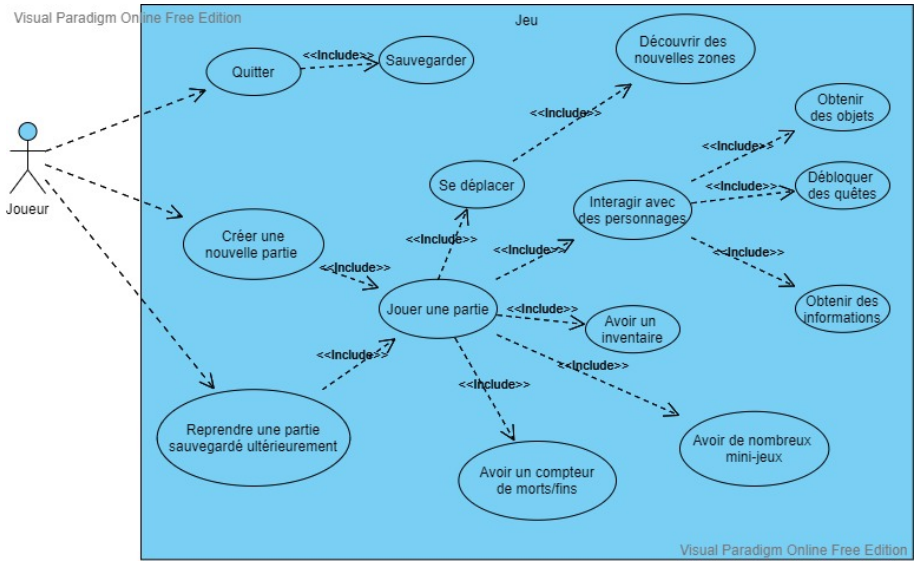

Figure 2: Diagramme d'utilisation

3.2 Diagramme UML

3.3 Diagramme de classes

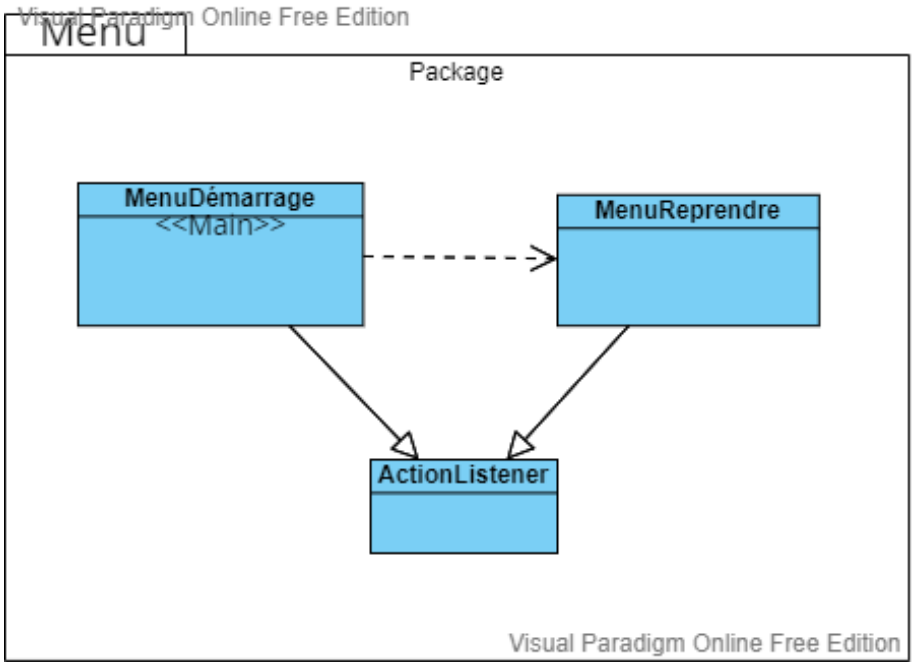

Figure 3: Diagramme de classe - Package Menu

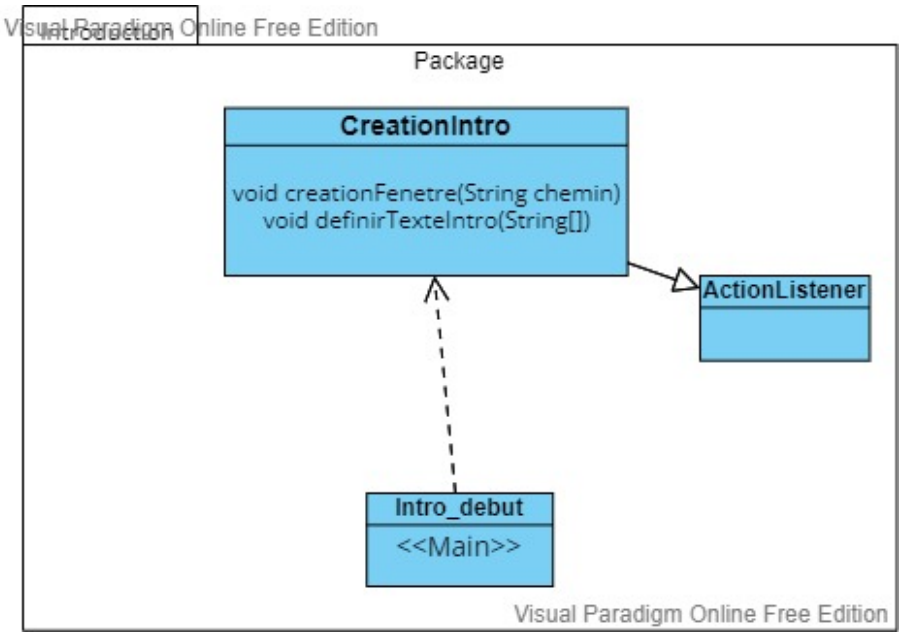

Figure 4: Diagramme de classe - Package Introduction

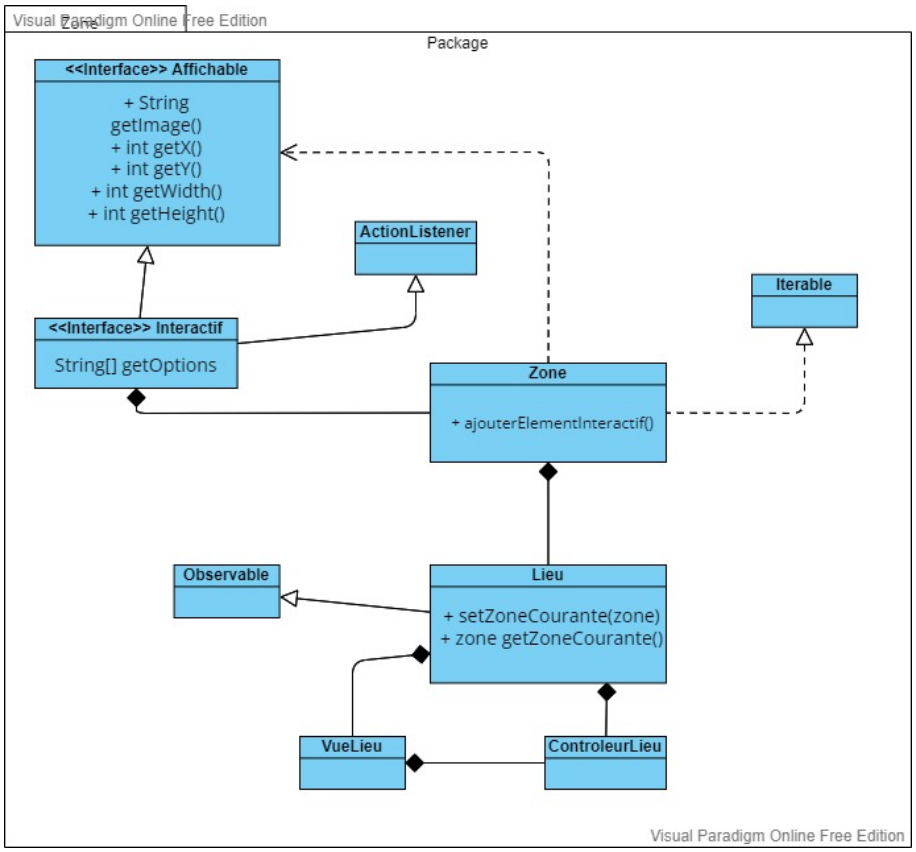

Figure 5: Diagramme de classe - Package Zone

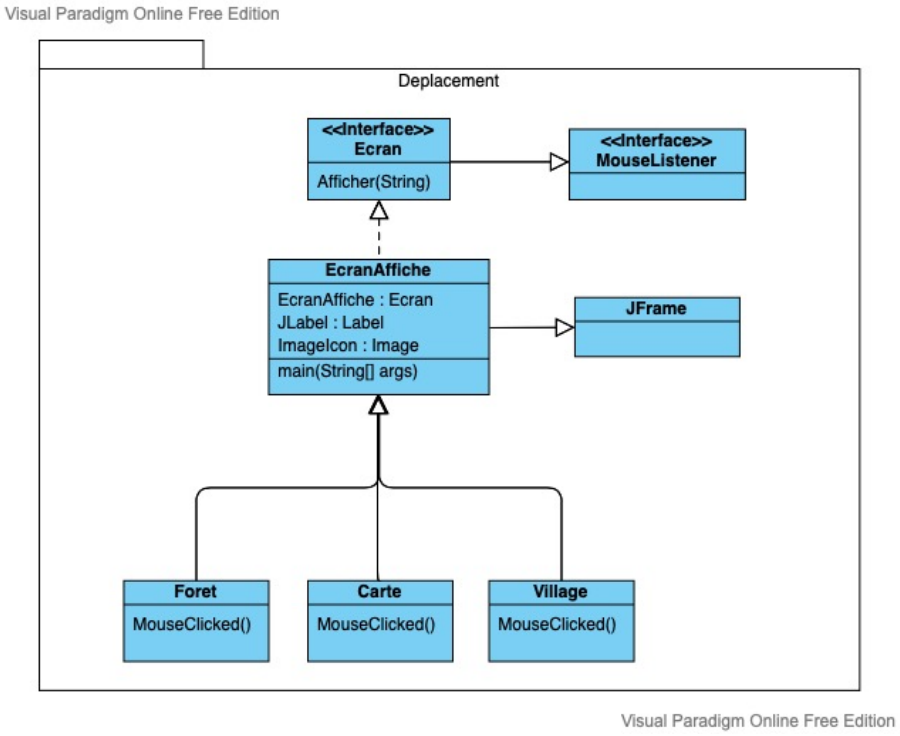

Figure 6: Diagramme de classe - Package Deplacement

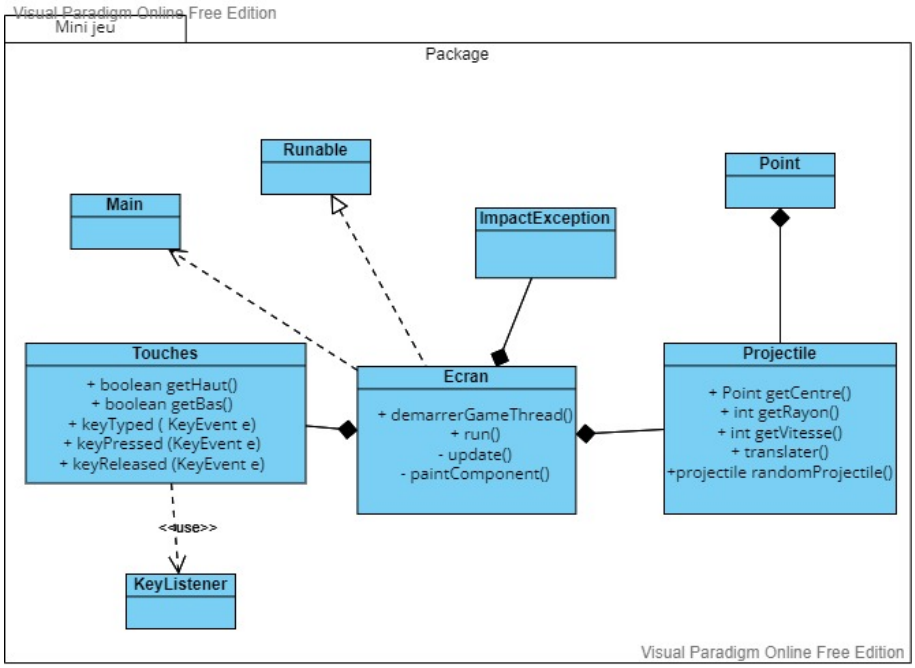

Figure 7: Diagramme de classe - Package Mini-jeu falaise

4 Avancement au cours des itération

4.1 Itération 1

Au cours de cette première itération nous nous sommes réparti le travail pour réaliser le maximum de Must possible. Chacun a travaillé sur une partie du projet, la répartition de ces parties à été faite au cours d'une réunion d'équipe où nous nous sommes tous acccordés sur ce qu'il y avait faire et qui le ferait. Cette répartition nous a permis d'avoir à la fin de l'itération tous les éléments principaux de notre jeu pour qu'il fonctionne correctement. Notre objectif pour la prochaine itération est de relier tout nos éléments pour avoir un jeu fonctionnel.

4.1.1 Package Menu

Dans ce package, il y a les différents menus du jeu. Pour le moment, deux menus ont été créés, le menu de démarrage et le menu pour reprendre une partie en cours.

Ce package contient deux classes, une pour chaque menu. La plus importante est celle du menu de démarrage, elle contient le constructeur de ce menu et le main qui permet de lancer le menu. Ce premier menu affiche une image de fond ainsi que trois boutons qui permettent de démarrer le jeu, reprendre une partie ou quitter le jeu. Si l'on clique sur reprendre une partie, on est dirigé vers le menu de l'autre classe.

Cette deuxième classe permet d'afficher un fond et trois boutons qui permettent de sélectionner le profil que l'on veut reprendre. Cette classe ne possède pas de main, car elle est directement appelée par le menu de démarrage.

4.1.2 Package Introduction

Ce package va contenir les différents introductions que l'on peut trouver dans le jeu : l'introduction du début de jeu, mais également une petite introduction lorsque la partie en cours se termine à cause de la mort du personnage.

Il contient une classe CreationIntro qui comporte les différentes méthodes permettant la construction d'une fenêtre ayant pour fond une image d'illustration, et un carré de fond blanc dans lequel s'affiche les différents scripts. On y trouve aussi la construction d'un bouton servant à passer à la prochaine phrase du script, ainsi qu'un bouton servant à passer entièrement le script.

Cette classe permet d'afficher un script composé d'autant d'étape qu'on le souhaite.

L'objectif était d'automatiser le plus possible la création de cette écran d'introduction, afin que la création de nouvelles introductions soit le plus rapide et le plus facile possible.

4.1.3 Package Interaction

4.1.4 Package Mini-jeu

4.1.5 Package Deplacement

4.1.6 Package Zone

Ce paquetage permet l'affichage et la gestion des zones et de leurs éléments interactifs.

Il utilise une interface Affichable qui correspond à toute classe devant être affichée sous la forme d'une image et une interface Interactif qui correspond à tout élément affichable qui a aussi des options d'interactions.

L'interface Interactif hérite donc d'Affichable et d'ActionListener (pour pouvoir réaliser les options lors d'un choix dans le controleur).

Une classe Zone réalise Affichable et contient une liste d'éléments interactifs.

Une classe Lieu correspond au modèle de la zone courante (la zone à afficher dans laquelle se trouve le personnage). Elle possède 2 méthodes (setZoneCourante et getZoneCourante) permettant le changement ou l'obtention de la zone courante. Cette classe est Observable pour permettre la modification du controleur et de la vue lors d'un changement de zone.

La classe VueLieu permet l'affichage de la zone et la classe ControleurLieu permet l'affichage des éléments interactifs ainsi que la création des menus sur ces éléments.

4.2 Iteration 2

Comme pour la première itération, nous nous sommes répartis le travail lors d'une réunion de groupe. Nous avions pour objectifs d'améliorer les classes réalisées pour l'itération précédente, de relier nos différentes classes afin de pouvoir commencer à créer les scénarios du jeu, et de commencer la création d'autres mini-jeux qui servent dans les différents scénarios.
Nous avons également fait une réunion avant la fin de l'itération pour partager ce que chacun avait fait, et pour relier tous ensemble nos classes.

L'objectif pour la prochaine itération est d'avoir au moins 2 scénarios fonctionnels.

4.2.1 Package Affichable

Il contient une interface Affichable qui correspond à toute classe devant être affichée sous la forme d'une image et une interface Interactif qui correspond à tout élément affichable qui a aussi des options d'interactions.

L'interface Affichable spécifie des méthodes pour avoir la position de l'image (getX, getY), obtenir l'image (getImage) sous la forme d'une ImageIcon, obtenir le nom de l'élément (getNom).

L'interface Interactif hérite d'Affichable et d'ActionListener (pour pouvoir réaliser les options lors d'un choix dans le controleur) et spécifie la méthode pour obtenir la liste des choix possibles sous la forme d'une liste de chaine de caractères.

4.2.2 Package mini-jeu Bataille Navale

Ce package sert à la réalisation d'un mini-jeu de type bataille navale. Nous nous sommes inspirés du tp sur le morpion, et avons réalisé une interface Modele, une classe Modele simple qui réalise cette interface, et une classe utilisant Swing pour afficher le jeu.

4.2.3 Package Introduction

Ce package va contenir les différents introductions que l'on peut trouver dans le jeu : l'introduction du début de jeu, mais également une petite introduction lorsque la partie en cours se termine à cause de la mort du personnage.

Il contenait initaliement une classe CreationIntro qui comportait les différentes méthodes permettant la construction d'une fenêtre ayant pour fond une image d'illustration, et un carré de fond blanc dans lequel s'affiche les différents scripts. On y trouvait aussi la construction d'un bouton servant à passer à la prochaine phrase du script, ainsi qu'un bouton servant à passer entièrement le script.

Il est maintenant composé d'une classe Introduction, qui utilise la la classe IU pour former la fenetre avec une l'image de l'introduction (utilise le système d'affichage des zones du paquetage Zone), une barre noir (utlise le système d'affichage de la barre utilisateur du paquetage IU) comportant le texte de l'introduction et un bouton continuer pour passer au texte suivant. A la fin celui-ci change la zone courante du joueur pour commencer le scénario.

4.2.4 Package Barre d'outils

Ce paquetage contient les éléments qui seront dans la barre utilisateur qui est dans le paquage IU. Pour le moment il n'est composé que des éléments qui permettent de créer l'inventaire du joueur. Pour réaliser cet inventaire nous utilisons une interface modèle et une classe modèle qui réalise cette interface. Enfin nous avons aussi une classe qui permet l'affiche de l'inventaire grâce à Swing.

4.2.5 Package IU

Ce paquetage représente l'interface de l'utilisateur. Il forme la fenetre du joueur composé d'une vue sur la Zone (classe ZoneVue du paquetage Zone) et d'une barre utilisateur (classe BarreUtilisateur).

BarreUtilisateur est une barre noir contenant tous les boutons de l'utilisateur (bouton pour l'inventaire, pour la carte) qui permettent d'afficher les éléments correspondants (la fenetre pour la carte correspond à un Jdialog pour apparaitre par dessus la fenetre principal).

Cette barre a le joueur pour attribut et possède une zone de texte pour les messages à afficher au joueur. Elle réalise donc la classe PropertyActionListener et se met à jour dès que la méthode setMessage de la classe Joueur est utilisée. Elle possedera aussi une zone de texte pour le compteur de morts dans les prochaines versions.

4.2.6 Zone

Il s'agit maintenant d'une interface Zone qui hérite de Affichable, et d'une classe ZoneSimple qui implémente cette interface. Une zone correspond donc à un nom, une image et un ensemble d'éléments interactifs.

La classe Lieu a été retirée par rapport à la première itération et la vue et le controleur sur la Zone prennent maintenant le joueur en attribut qui contient les méthodes que contenait Lieu (setZoneCourante et getZoneCourante). La vue se met donc à jour (en tant que listener sur joueur) dès que la méthode setZoneCourante est utilisée.

4.2.7 Joueur

Le paquetage Joueur contient l'interface Joueur qui spécifie les méthodes en rapport avec le joueur: obtenir la zone courante du joueur (getZoneCourante), obtenir l'inventaire du joueur (getInventaire), obtenir la carte du joueur (getCarte), obtenir le message à afficher au joueur (getMessage), changer la zone courante (setZoneCourante), changer le message (setMessage) et des méthodes d'ajout de Listener sur joueur.

JoueurSimple est une réalisation de cette interface qui s'initialise avec une zone de départ en paramètre.

4.2.8 Carte

Anciennement, sous le nom Deplacement, ce package permet d'afficher la carte, les zones accesssibles sur la carte et de déplacer le joueur lorsque celui-ci choisit une zone. De la même manière que le package Dialogue, la vue (l'affichage de la carte), le modèle (la carte) et le controleur (menu lorsqu'on clique sur une zone pour pouvoir s'y rendre) sont séparés pour permettre tout changement facilement.

Une carte est un élément affichable avec des méthodes pour obtenir l'ensemble des lieux accessibles et l'ensemble de celles non accessibles.

Un lieu est un élément intéractif (interface Interactif), qui a pour attribut le joueur (classe Joueur) et la zone (classe Zone) 'à laquelle il correspond pour pouvoir modifier la zone courante du joueur lorsque l'option "Se rendre à" est sélectionné. Il a aussi pour attribut l'image à afficher sur la carte ainsi que le nom du lieu.

4.2.9 Package Dialogue

Précédemment intitulé Interaction, celui ci permet de construire et d'afficher des boites de dialogues avec les différents personnages. Il a été modifié par rapport au package Interaction de sorte de créer autant de dialogues et de choix à l'aide d'un arbre de dialogue. Il a été aussi modifié pour séparer la vue, le modèle et le controleur de sorte que chacun puisse être facilement modifiable si nécesssaire.

Un dialogue (interface Dialogue) est un élément affichable (utilise l'interface affichable. On peut obtenir l'image pour le dialogue, le nom (pour le nom du personnage), une liste de chaine caractères (pour les différents choix actuels) et la phrase actuelle du personnage.

Son implémentation DialogueSimple utilise la classe ArbreDialogue qui correspond à un arbre dont chaque cellule a une phrase et une liste de choix. Cette classe contient un pointeur sur la cellule courante pour savoir la phrase actuelle ainsi que les choix possibles de cete étape dialogue.

Faire un choix utilise la méthode setChoix(entier) de la classe DialogueSimple qui descend dans l'arbre de dialogue selon le choix et envoie un signal aux Listeners (la vue).

La vue est faite à partir de swing et consiste en un label contenant l'image et un panel composé de labels avec des textes numérotés pour les choix et un label avec un texte pour la phrase du personnage.

La controleur correspond à un panel avec un zone de saisie de texte qui ne peut prendre que les chiffres correspondant aux différents choix.