DOSSIER DE CONCEPTION : PROJET JEU

Mercredi 5 mars 2014

Groupe A

L3 info SPI

Sommaire

1 Présentation général **1**
1.1 Objectif du document 1
1.2 Objectif du projet 1
1.3 Documents de références 1

2 Modélisation du projet **2**
2.1 Les cas d'utilisations 2
2.2 Contraintes de conception 2
2.3 Le scénario du système 2
2.4 Schéma de navigation 2

3 Modélisation de la Sauvegarde des données **3**
3.1 Dictionnaire de données 3
3.2 Description des données sauvegardées et de leur stockage 3
3.3 Méchanismes de Sauvegarde des Données 4

4 Interface Homme-Machine **5**

5 Outils, normes et standards de programmation **6**
5.1 Languages utilisées 6
5.2 Envrionnement et outils de développement 6
5.3 Standards de programmation 6

1 Présentation général

1.1 Objectif du document

Ce document est rédigé dans le cadre du projet de troisième année de licence SPI option informatique de l'Université du Maine. Le Dossier de Conception vient en complément du Cahier des Charges. Il a pour objectif de monter la modélisation du système à partir des besoins et contraintes exprimés dans le Cahier des Charges. Il informe également sur les outils et les normes utilisés pour la conception du produit.

1.2 Objectif du projet

L'objectif de ce projet est de concevoir et de développer une application de Picross, ainsi que de rédiger les documents inhérant à la gestion et la finalisation d'un projet informatique. Les utilisateurs doivent pouvoir, bien évidemment, jouer, mais il doivent également pouvoir créer des grilles, modifier la taille des grilles, et charger les partie préalablement sauvegarder. Il doit également être possible de jouer sous son propre profil afin d'avoir accès aux options les plus poussées de l'application.

1.3 Documents de références

Afin de réaliser le présent Dossier de Conception nous nous sommes appuyés sur le Cahier des Charges ainsi que sur les exemples que nous avons récupérés grâce à un ancien élève de DUT informatique.

2 Modélisation du projet

2.1 Les cas d'utilisations

Partie de Kévin

2.2 Contraintes de conception

Partie de Kévin

2.3 Le scénario du système

Partie de Kévin

2.4 Schéma de navigation

Partie de Kévin

3 Modélisation de la Sauvegarde des données

3.1 Dictionnaire de données

Nom de l'entité	Type	Commentaires
Grille	Class Ruby Grille	Une collection de case, jointe à un nom et des paramètres ("ALEA20", 20/11/2014)
Profil	Chaîne de caractère	Le profil joueur, constitué uniquement du nom du profil ("joueur1")
Scores	Tableau	Un tableau à trois colonnes, comprenant le nom de la grille, le profil et le score
Parties Sauvegardées	Class Ruby Partie	Partie en cours, sauvegardée avec tous ses paramètres (profil = "", grille ="", temps= – :–, ...)

3.2 Description des données sauvegardées et de leur stockage

L'ensemble des données sauvegardées, présentées brièvement ci-dessus, est ici décrite plus en détail.

Grille : les grilles sont les supports du jeu(cf Cahier des Charges 3.1.1). Pour permettre à plusieurs utilisateurs de jouer sur une même grille, il est nécessaire de sauvegarder ces objets. Les grilles sont constituées d'un collection de case, d'un nom, attribué à la création et de divers paramètres, stockés sous formes d'entiers. Elles sont stockés dans un même fichier, les unes après les autres, à leur création, qu'elle soit manuelle ou aléatoire. Ce fichier comporte trois grilles par défaut.

Profil : Les profils sont les noms que choisissent les utilisateurs(cf Cahier des Charges 3.1.2, qui les définissent dans le jeu, et qui les lient aux grilles créées, aux parties jouées et aux scores. Un profil est constitué d'une simple chaîne de caractères, choisie par l'utilisateur à la création du profil. Cette chaîne est sérialisée à la création, dans un tableau contenant l'ensemble des profils créés, le tout stocké sur un même fichier spécifique aux profils. Ce fichier comporte un profil "par défaut".

Scores : Les scores sont des entiers, dépendant du temps réalisé pour finir la Grille. La donnée à sauvegardée "Scores" est un tableau à trois colonne, comprenant le score lui même, le nom de la grille sur lequel il a été réalisé, et le nom du profil l'ayant réalisé. Ce tableau est stocké sur un fichier propre.

Parties Sauvegardées : Elles correspondent à la sérialisation d'une instance de la classe Partie, contenant un certin nombre de paramètres et d'informations, comme la grille utilisée, l'utilisateur, le temps écoulé, etc. Ces paramètres sont conservés sur un fichier lors de la sérialisation. L'utilisateur peut ainsi quitter une partie, fermer le programme, et reprendre cette partie plus tard. Chaque Partie Sauvegardée est conservée sur un fichier propre.

3.3 Méchanismes de Sauvegarde des Données

Cette partie détaille l'ensemble des méchanismes de sauvegarde des données précédemment décrites, c'est à dire quand et comment ces données seront sauvegardées.

- Les Grilles sont sauvegardées après leurs initialisation. Elles sont générées de deux manières : soit aléatoirement, sur demande d'un utilisateur, soit manuellement dans un éditeur. À chaque fois, après avoir été initialisées, elles sont ajoutées à une collection, qui permet différents tris (par taille, par créateur). Cette collection est sérialisée via le module YAML et sauvegardée dans un fichier propre, à chaque fois qu'une grille est ajoutée. Cette collection est chargée depuis le fichier au lancement du jeu via le module YAML.

- Les Profils sont sauvegardés à leur création. À chaque fois qu'un utilisateur crée un nouveau profil, celui ci est ajouté à une collection de Profil. Cette collection est sérialisée et sauvegardée dans un fichier propre via le module YAML. Elle est chargée au lancement du jeu depuis ce fichier via le module YAML.

- Les Scores sont sauvegardés à chaque fin de parties. Ils sont stockés dans un tableau, comme indiqué ci-dessus. À chauqe fois qu'un nouveau score est réalisé, il est ajouté au tableau, et le tableau est sérialisé et sauvegardé dans un fichier propre via le module YAML. Le tableau est chargé depuis ce fichier à chaque lancement du jeu via le module YAML.

- Les Parties Sauvegardées sont sauvegardées sur demande de l'utilisateur en cours de parties. Elles sont ajoutées à une collection de Parties Sauvegardées, permettant de trier ces Parties, notemment en fonction du Profil joueur, pour proposer à ce dernier de continuer la(les) dernière(s) Partie(s) Sauvegardée(s). À chaque fois qu'une Partie Sauvegardée est ajoutée, la collection est sérialisée et sauvegardée sur un fichier propre via le module YAML. Cette collection est chargée depuis le fichier à chaque lancement du jeu.

4 Interface Homme-Machine

Partie d'Anice

5 Outils, normes et standards de programmation

5.1 Languages utilisées

Ci dessous nous référençons la liste des différents langages utilisés dans le projet :

- Ruby : Langage de programmation orienté objet, utilisé en version 2.0.0 sur nos machines personnelles, avec une compatibilitée sur les versions antétrieurs,

5.2 Envrionnement et outils de développement

Le développement sera effectué sur des ordinateurs suffisamment puissants auxquelles nous avons accès à l'Université du Maine, ainsi que sur nos machines personnelles.

- Version utilisé de Ruby : 2.0,
- GTK : Bibliothèque graphique (The **G**IMP **T**ool**k**it) utilisé avec Ruby.

5.3 Standards de programmation

Durant le développement, nous respecterons les conventions de codages ci contre :

- Les commentaires doivent être rédigés en français et permettent d'expliquer de façon claire le code.
- Le code doit être rédigé de la manière la plus lisible possible (indenté et explicite).
- La documentation du code sera faites à l'aide de Ruby doc.