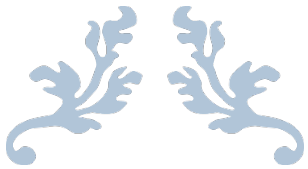

CAHIER DE CHARGES DU LOGICIEL JBCITY

Projet de Programmation Objets Concepts Avancés

PAR

HENOC CHRISTIAN KHOUILLA

LOUIS BERTIN NDJOMO

MOHAMED SALOU NABE

SOUHIRE KENAWI

20 NOVEMBRE 2013

Dans le but de nous préparer au monde de l'entreprise, le cours de Programmation Objet – Concepts Avancés nous exerce à la conception de système objet extensible. Pour cela, un projet de développement, qui s'étend sur toute la première période, nous est proposé.

Le but de ce cahier de charges est de pouvoir présenter le logiciel qui sera développé par nos soins. Dans un premier temps, nous présenterons les objectifs qui ont été fixés, puis une description du logiciel à réaliser, et enfin les modalités de réalisation de ce dernier.

I. OBJECTIFS

L'objectif fixé est de réaliser un jeu de gestion de ville dont le nom sera JBCity, à l'image du célèbre jeu qu'est SimCity. Ce dernier permet de simuler la gestion d'une ville en incarnant son maire.

Ce dernier est responsable de la mise en place des infrastructures qui permettront l'évolution de la ville, en fonction du budget qui lui est alloué. La particularité de ce jeu est qu'il n'a pas de fin, et chaque ville créée est unique.

II. DESCRIPTION DU LOGICIEL

JBCity est une simulation mettant en interaction, un individu considéré comme étant le maire et sa ville de tutelle. Le maire a pour objectif de construire et administrer sa ville en lui fournissant tous les outils et infrastructures nécessaire pour son évolution.

Les différentes actions ou stratégies du maire concourent à un résultat, nous mesurerons l'efficacité du maire de la ville sur ce résultat.

Les principes de jeu se basent sur ce que compose la ville, sur les actions du maire sur cette dernière. Il est aussi possible d'avoir un aperçu des différentes actions observables.

1. La ville

Celle-ci comporte des infrastructures qui sont les différents types de constructions, à savoir :

1. Infrastructure d'habitation (Maison, immeubles)
2. Infrastructure d'œuvre sociale (École, hôpitaux, centres d'accueil, etc.)
3. Infrastructure industrielle (Usine, central électrique, société de service, etc)
4. Infrastructure de transport (gare, aéroports, parking, etc.)

5. Infrastructure de voies de communication (routes, chemins de fer, lignes téléphoniques, lignes électriques)
6. Infrastructure Commerciale (Marché, supermarché, centre commercial, etc.)
7. Infrastructure de sécurité (postes de police, pompiers, etc.)
8. Infrastructure de loisir (Jardin public, stade, salles de spectacle)

Ces infrastructures ont un prix, et certains peuvent générer du revenu qui sera taxé afin d'augmenter le budget de la ville.

Comme dans la vie réelle, une ville dispose aussi d'une population. Celle-ci est intimement liée à la construction des infrastructures, qui peuvent augmenter ou diminuer le nombre d'habitants, les rendre plus joyeux ou tristes, malades ou en bonne santé. Et toutes ces actions se font automatiquement en fonction du temps et des saisons. Le taux de criminalité, l'indice de pollution influence également la population de la ville.

La ville peut être victime des catastrophes naturelles (tornade, foudre, tremblement de terre, etc.)

Les infrastructures de la ville et tout ce qu'elle contient sont représentées sur une carte qui sera affichée dans le jeu, et qui interagira avec le joueur.

2. Les actions du maire du ville

Le maire ou encore le joueur de l'application peut agir sur l'évolution de la ville. Il peut décider de la construction et la destruction des infrastructures. Il peut également faire la collecte des impôts.

3. Les résultats observables

Les résultats pouvant être observé lors de l'exécution

III. MODALITES DE REALISATION

En groupe de trois étudiants, nous repartirons notre travail en plusieurs phases de réalisation aux bouts desquelles des livrables seront fournis.

- Phase 1 : Cahier de charges
- Phase 2 : Modèle conceptuel et architecture
- Phase 3 : Squelette
- Phase 4 : Implémentation

Pendant chaque phase, les tâches sont réparties entre les membres de l'équipe.

Afin de pouvoir intégrer dans notre travail un certain nombre de modifications ou d'ajouts de fonctionnalités non prévues par le cahier de charge initial, sans que ça implique une remise en cause, nous allons nous assurer de la faible dépendance de nos modules.