Erode

Document d'architecture logicielle

Version 1.2

Historique des révisions

Date	Version	Description	Auteur
2017-01-30	1.0	Version initiale du document d'architecture	Mathieu O. Wilson, Alexandre Paquet, Odric Plamondon, Jonathan Sauvé
2017-02-17	1.1	Révision en vue de la remise officielle	Odric Plamondon, Alexandre Paquet
2017-04-14	1.2	Révision en vue de la remise finale	Alexandre Paquet, Mathieu O.Wilson

Table des matières

Document d'architecture logicielle

1. Introduction

Dans ce document, nous élaborerons sur les objectifs et contraintes architecturaux, sur la vue des cas d'utilisation, sur la vue logique du projet, sur la vue des processus, sur la vue de déploiement, ainsi que sur la taille et les performances qui pourraient avoir un impact sur l'architecture et le design du projet.

2. Objectifs et contraintes architecturaux

En ce concerne les objectifs et contraintes architecturaux de notre projet, ils sont nombreux. Tout d'abord, pour ce qui a trait aux objectifs, le fait que notre projet est pour les portes ouvertes de Polytechnique Montréal restreint la créativité que nous pouvons infuser dans le projet. En effet, étant donné que notre client est Polytechnique et que notre public cible sont les possibles futurs étudiants de Polytechnique, nous devons contraindre le thème du projet à des sujets, dans la mesure du possible, de tout âge. Suite à cela, le projet doit être réalisé en douze semaines, cela nous laisse peu de temps et donc, vient contraindre notre échéancier. Ensuite, comme contraintes architecturales, nous avons le fait que l'on développe sur Unity avec, comme plateforme cible, les machines roulant le système d'exploitation Windows, le tout se contrôlant avec le contrôleur de Xbox One. Cela a pour effet de restreindre la portabilité du jeu qu'à des plateformes roulant Windows. Finalement, étant donné le fait que l'équipe est composée de programmeurs, et d'aucun artiste, nous avons un budget de 200$ pour des "assets" servant à embellir notre jeu. Ce budget étant plutôt élevé, nous ne sommes pas beaucoup restreints dans le nombre d'assets que nous pouvons acheter ou dans la qualité de ceux-ci.

3. Vue des cas d’utilisation

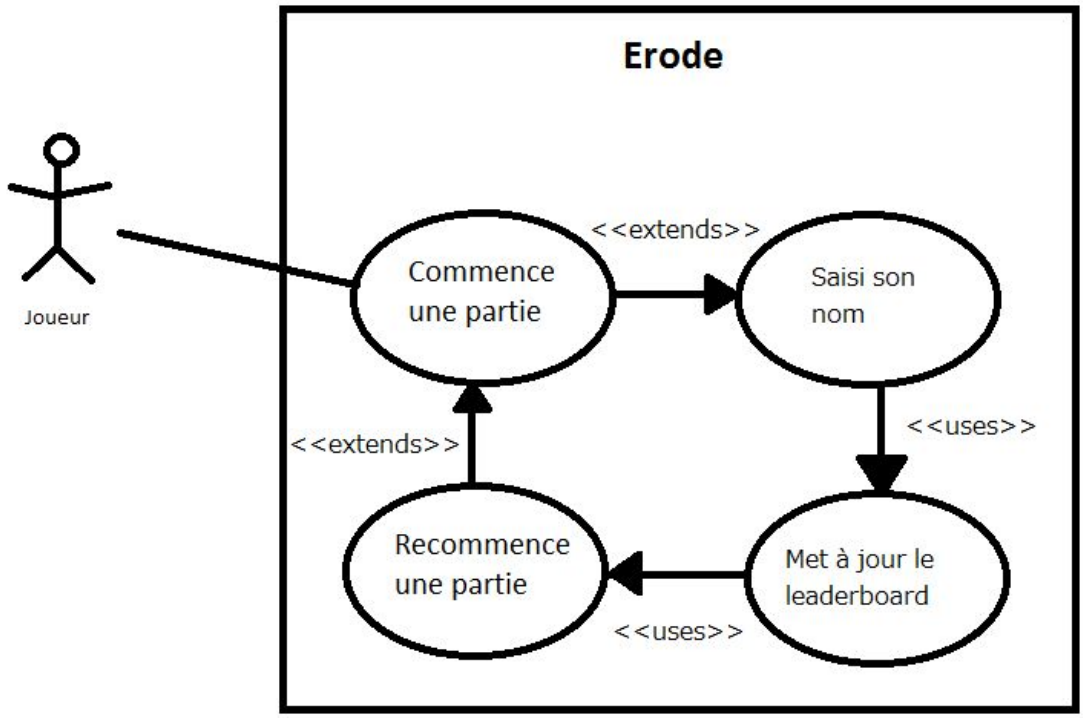
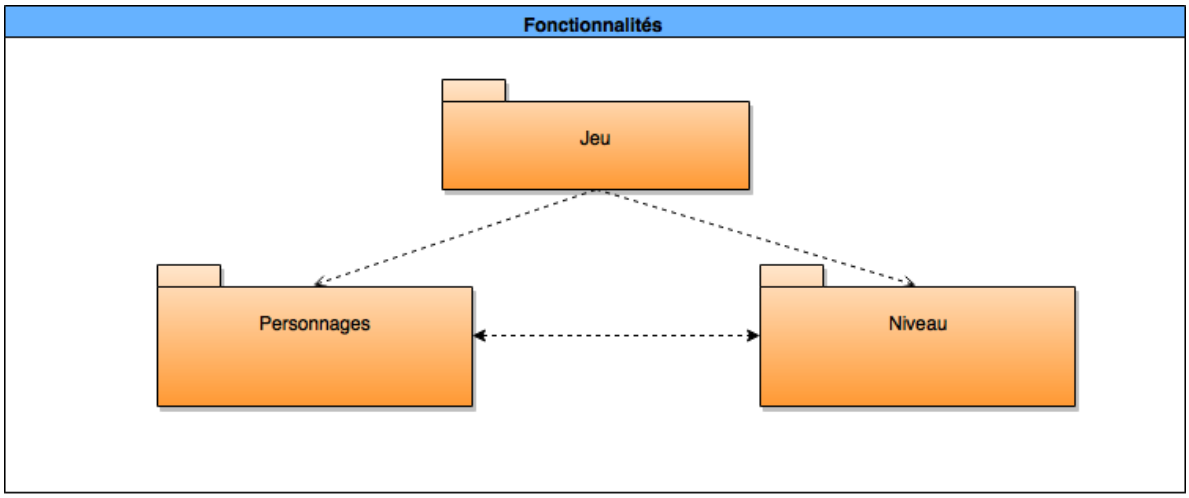


Figure 1 - Diagramme de cas d’utilisation

L’interface du jeu sera très simple et et facile à naviguer pour le joueur. Au menu principal, le joueur ne peut que commencer la partie. En situation de jeu, il est possible pour le joueur d’appuyer sur le bouton pause lorsqu’il en ressent le besoin. À la fin de la situation de jeu, le classement apparaît et le joueur peut saisir son nom si son score est dans le classement, sinon il peut choisir de recommencer la partie.

4. Vue logique

Dans l'engin de jeu Unity 3D, il n'y a pas vraiment de paquetage. Tout le code est inclus dans le même projet. Les "classes" ajoutées sont en fait des scripts qui sont ajoutés comme composantes à des objets de l'arbre de jeu, et qui sont actualisés avec les objets auxquels ils sont associés. Ceci dit, au lieu de présenter des paquetages, la section présentera les différents groupes de fonctionnalités. Ces groupes de fonctionnalités représentent les différents éléments sur lesquelles les scripts vont agir, en particulier les personnages, le niveau et la boucle principale de jeu.



Jeu	
Description:	Ce groupe de fonctionnalités contient les scripts qui relatifs aux changements de contextes, ainsi que l'interface usager de ces contextes..
Classes incluses:	Ce groupe de fonctionnalités contient les classes suivantes GameManager: Contrôle les changements de contexte et l'apparition des éléments d'interface usager. CameraController: Cette classe contrôle la caméra, qui doit suivre le personnage du joueur afin de toujours le garder en vue. La caméra doit aussi réagir à certain événements du jeu avec des effets de "glisse" ou en se secouant.
Relations:	Ce groupe de fonctionnalités est en relation avec le groupe Joueur.
Sous-groupes:	Personnages, Niveaux

Personnages	
Description:	Ce groupe de fonctionnalités contient les scripts qui contrôlent un personnage. Il peut s'agir d'un personnage contrôlé par un joueur ou bien une intelligence artificielle.
Classes incluses:	Ce groupe de fonctionnalités inclut les classes suivantes CharacterController: Cette classe contrôle les différentes actions qui peuvent être effectuées sur le personnage comme les collisions, les attaques, la perte de points de vie et les effets visuels, entre autres. CharacterStateMachine: Cette classe contrôle les "inputs" et les actions qui sont disponibles au personnage selon son état. AnimationStateBehaviour: Les scripts de comportement d'animation s'occupent du transfert d'information entre les animations et la logique d'un personnage.
Relations:	Ce groupe de fonctionnalités est en relation avec le groupe Niveau.

Niveau	
Description:	Ce groupe de fonctionnalités contient les scripts qui contrôlent les éléments présent dans le niveau. Il s'agit de la plateforme, des obstacles et des power-ups.
Classes incluses:	Ce groupe de fonctionnalités contient les classes suivantes Grid: Cette classe contrôle la création de la grille de jeu. Elle gère et optimise les opération sur les tuiles. Tile: Cette classe contient les informations d'une tuile comme ses points de vie et sa visibilité. PowerUp: Cette classe contrôle les bonus octroyés par les power-up ainsi que leurs durée de vie. ObstacleController: Cette classe contrôle le comportement des obstacles, comme leurs collisions et leurs déplacements. Spawner: Cette classe contrôle l'apparition des différents types d'obstacle, de power-up et d'ennemis sur la grille de jeu.
Relations:	Ce groupe de fonctionnalités est en relation avec le groupe Joueur.

5. Vue des processus

Dans le cas d'un jeu vidéo développé avec Unity, il y a très peu de composantes qui interagissent entre elles. En effet, toutes les classes (aussi appelées scripts) dérivent de « MonoBehaviour », il n'y a donc pas différentes composantes à proprement parler. De plus, chaque élément se trouvant directement dans l'éditeur de Unity est essentiellement un « GameObject ». Bien qu'il y ait peu de processus à analyser, il y a d'autres diagrammes intéressants à présenter dans le cadre de notre jeu vidéo. Par exemple, nous avons les différentes boucles d'animation liées aux « inputs » du joueur, ainsi que celles des différents IA.

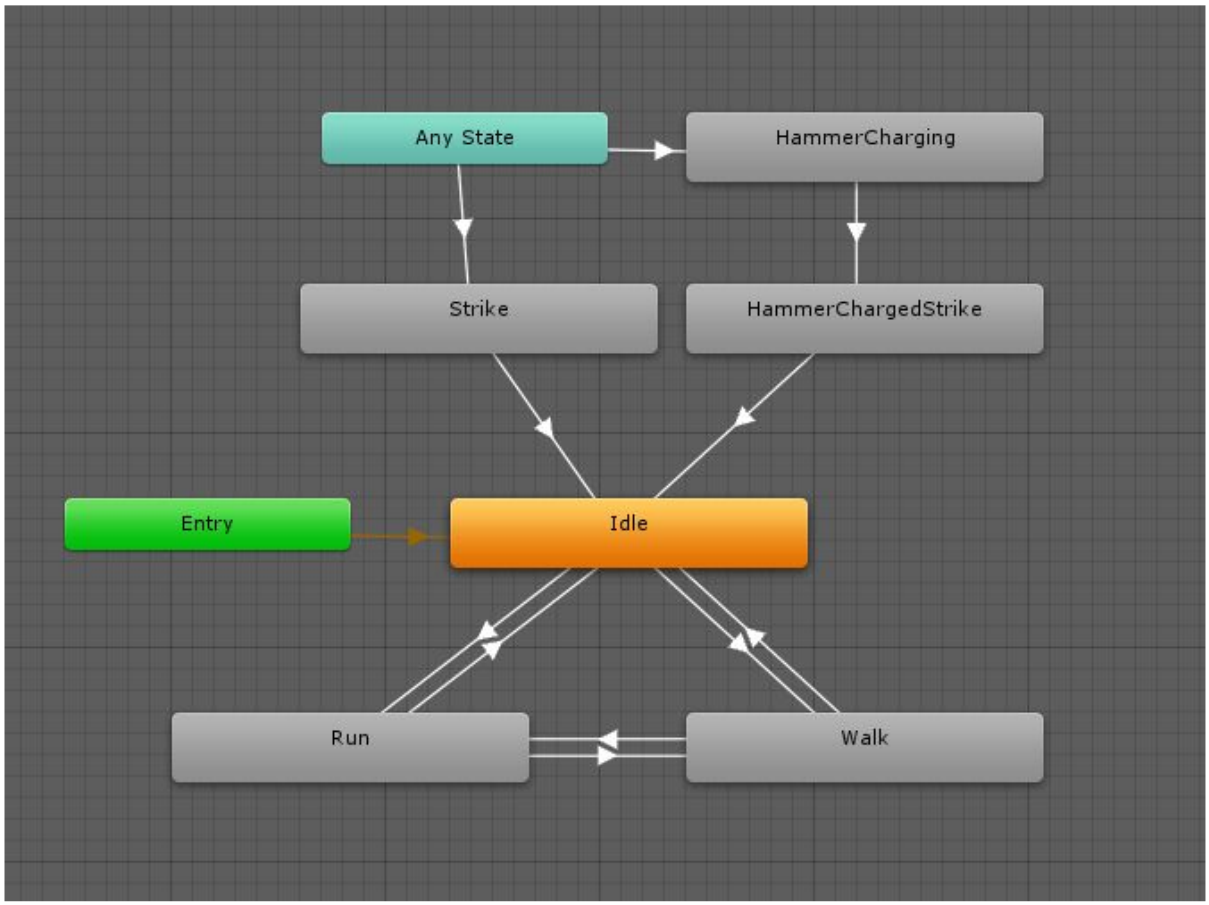


Figure 2 - Machine à états des animations du personnage

Dans le diagramme ci-haut, nous pouvons observer la machine à états qui régit les différentes animations possibles pour le personnage, et comment nous passons d'une à l'autre. Par exemple, la boucle entre l'état « Idle », « Run » et « Walk » gère les mouvements de marche d'Horatio. Les flèches représentent un « trigger », qui réagit quand une certaine action est effectuée avec la manette. Dans notre cas, la marche est gérée avec le joystick gauche. Si l'amplitude du mouvement du joystick est petite, c'est l'animation « Walk » qui est activée. Lorsqu'elle est près d'un certain seuil prédéfini, Unity effectue un mélange entre l'animation « Walk » & « Run ». Quand l'amplitude dépasse le seuil, seul l'animation « Run » est jouée. Il est possible pour le personnage de retourner à tout moment

à l'animation « Idle » lorsque le joystick est relâché. Les autres boucles fonctionnent environ de la même façon. La case « Entry » veut simplement dire que l'animation de départ est « Idle ». Finalement, la dernière particularité est « Any State ». Cette case veut dire que peu importe dans quel état notre personnage se trouve, il peut se rendre aux cases ciblées par « Any State ». C'est un moyen de limiter le nombre de flèches dans le diagramme et de le simplifier du même coup.

6.Vue de déploiement

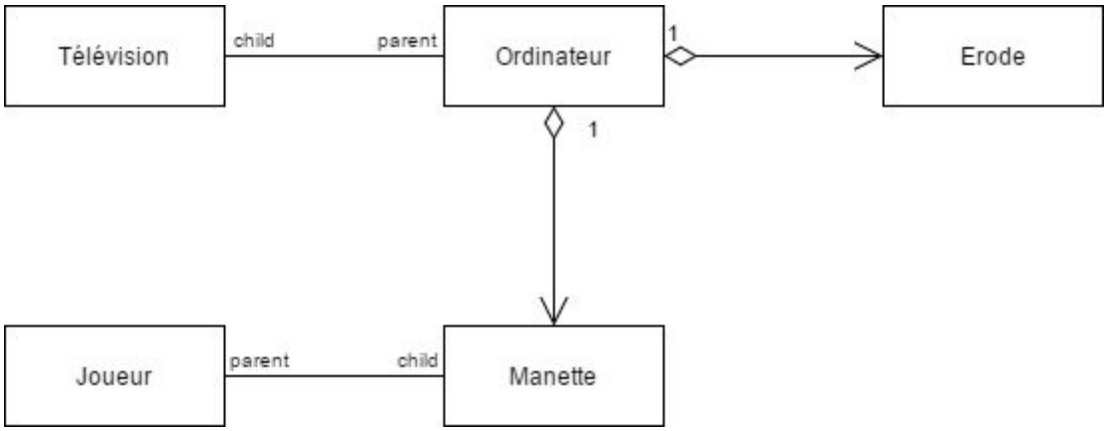


Figure 3 - Configuration de matériel physique

Lorsque viendra le temps de déployer le jeu vidéo durant les portes ouvertes de Polytechnique, il faudra configurer l'ensemble du système pour permettre aux joueurs de s'amuser. C'est ce que montre la figure X ci-haut. Comme on peut l'observer, le jeu Erode sera installé sur un ordinateur sous Windows, qui lui sera connecté à un téléviseur. Une fois le jeu lancé, le joueur aura la possibilité d'y jouer avec une manette de Xbox One connecté à ladite télévision.

7.Taille et performance

Le jeu étant installé sur un ordinateur quelconque, il n'y a pas vraiment de limite sur la quantité d'espace que le jeu peut occuper sur le disque dur. Au maximum, le jeu pourrait atteindre 2 giga-octets, donc n'importe quel ordinateur fera l'affaire. Le jeu ne devrait jamais nécessiter plus de 2 gigaoctets de mémoire vive non plus.

Afin que l'expérience du joueur soit parfaite, le jeu devrait toujours s'exécuter à au moins 60 images par secondes.