ULTIMAADE

Architecture Globale Version 1.1

Ultimaade Team ultimaade@gmail.com

Résumé : `Ultimaade` *est un logiciel libre et multiplateforme, de création d'histoires et de support multimédia.*

La principale fonction du logiciel est l'édition de ***romans visuels*** *(**visual novels** en anglais), type particulier de jeux vidéo très répandu au Japon et peu connu en occident.*

Ultimaade permettra également de faire des présentations/montages type vidéo ou texte, offrant ainsi aux enseignants par exemple, de créer du contenu pédagogique selon leurs domaines de compétence.

Dans un sens plus large le logiciel est destiné à tout type d'utilisateur, désirant créer une application multimédia autonome et multiplateforme : technicien ou non, enfant ou adulte.

Ce document décrit l'architecture et l'analyse fonctionnelle du logiciel.

Nous commencerons par une capture des besoins qui a été définie dans le document d'étude détaillé.

Nous entrerons ensuite dans la phase d'analyse et conception architecturale.

Ceci nous permettra de construire un pont entre la description purement fonctionnelle et générique des besoins et la réalisation.

Table des matières

Chapitre I

Contexte du projet

Suite à l'évolution rapide des technologies, EPITECH membre du groupe IONIS, a entrepris de former des futurs experts en technologies de l'information.

Dans cette optique l'école a mis sur pied une stratégie consistant pour les étudiants de chaque promotion, en la conception et la réalisation d'un projet innovant, et répondant à une problématique concrète de l'heure. Ce concept s'appelle EIP (Epitech Innovative Project).

Pour les étudiants, l'EIP réalisé représente le projet de fin d'étude, et conditionne le diplôme d'expert décerné par l'école.

C'est dans ce cadre que le groupe ULTIMAADE s'est réuni autour du projet qui fait l'objet du présent document d'architecture.

Chapitre II

Acteurs et Cas d'utilisation

II.1 Identification des acteurs

Nous définissons ici les acteurs qui vont interagir avec le système.

Nous entendons par acteur, un humain, une machine, ou un système, ne faisant pas partie de la solution à réaliser mais qui participe au fonctionnement général de la solution par une interaction.

Cependant dans notre cas nous avons typiquement des acteurs humains.

Ceux-ci pourront être des enfants/élèves, des enseignants, des professionnels de bandes dessinées, ou tout simplement un utilisateur désirant créer des animations, créer un CDROM multimédia...

- `les enfants` : peuvent créer des albums du style «tourne-page», à branchements multiples
- `les enseignants` : découverte géographique, scénarios de pédagogie embarqués, enseignement assisté par ordinateur, etc
- `les professionnels de bandes dessinées et animation` : éditer, importer leurs dessins, puis les animer afin de réaliser un film interactif à partir de plusieurs scènes.
- `tout autre type d'utilisateurs` : développer une application multimédia, créer des animations, présentations, créer un CDROM présentation, etc.

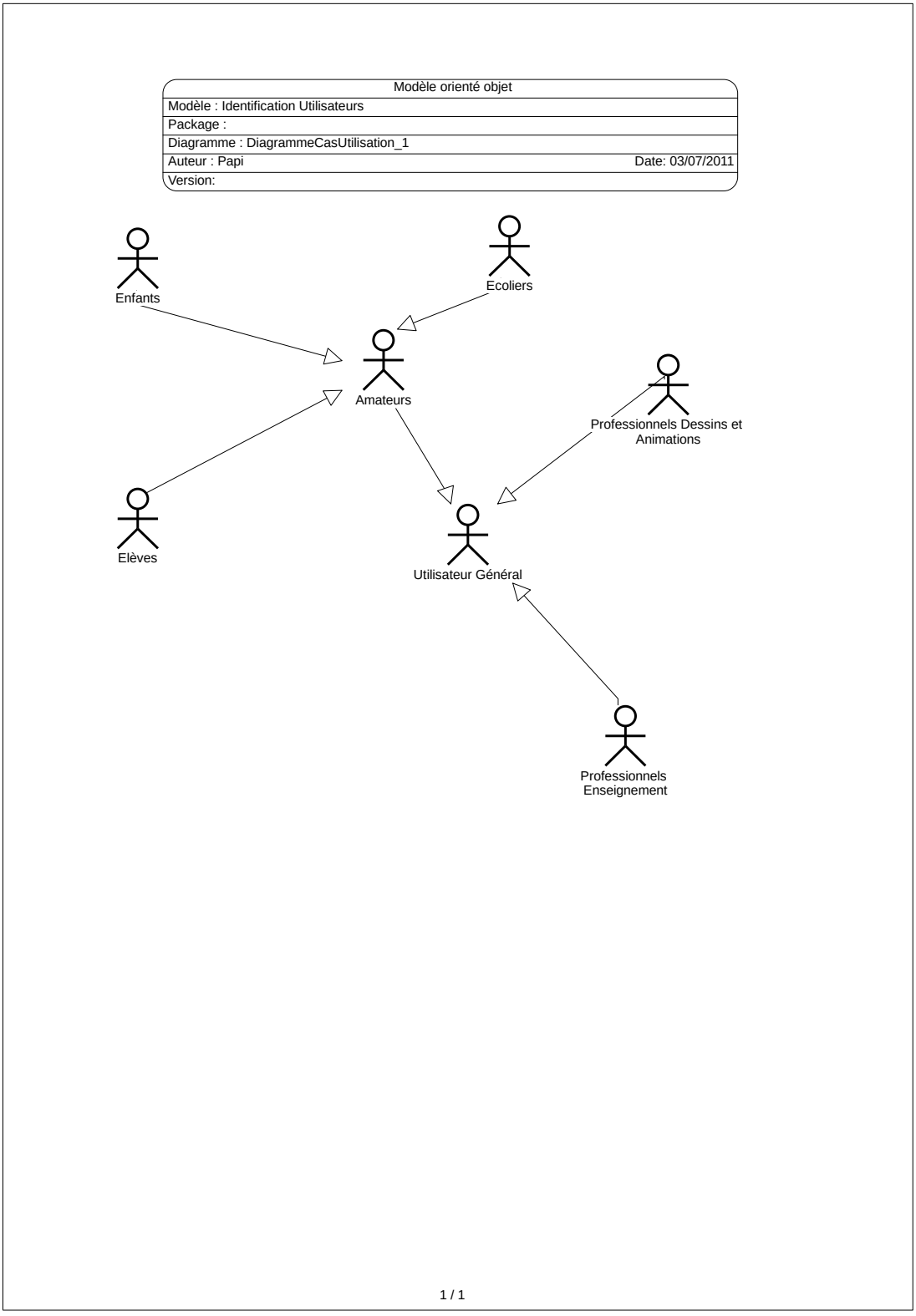

FIGURE II.1 – Utilisateurs Ultimaade

II.2 Identification des cas d'utilisation

Le logiciel regroupera principalement trois catégories d'utilisateurs, qui définiront les niveaux d'utilisation :

- les amateurs (enfants, néophytes)
- les professionnels de l'enseignement
- les professionnels du monde du jeu vidéo, dessins, animations,...

La figure ci-dessous représente les diagrammes des cas d'utilisation du logiciel

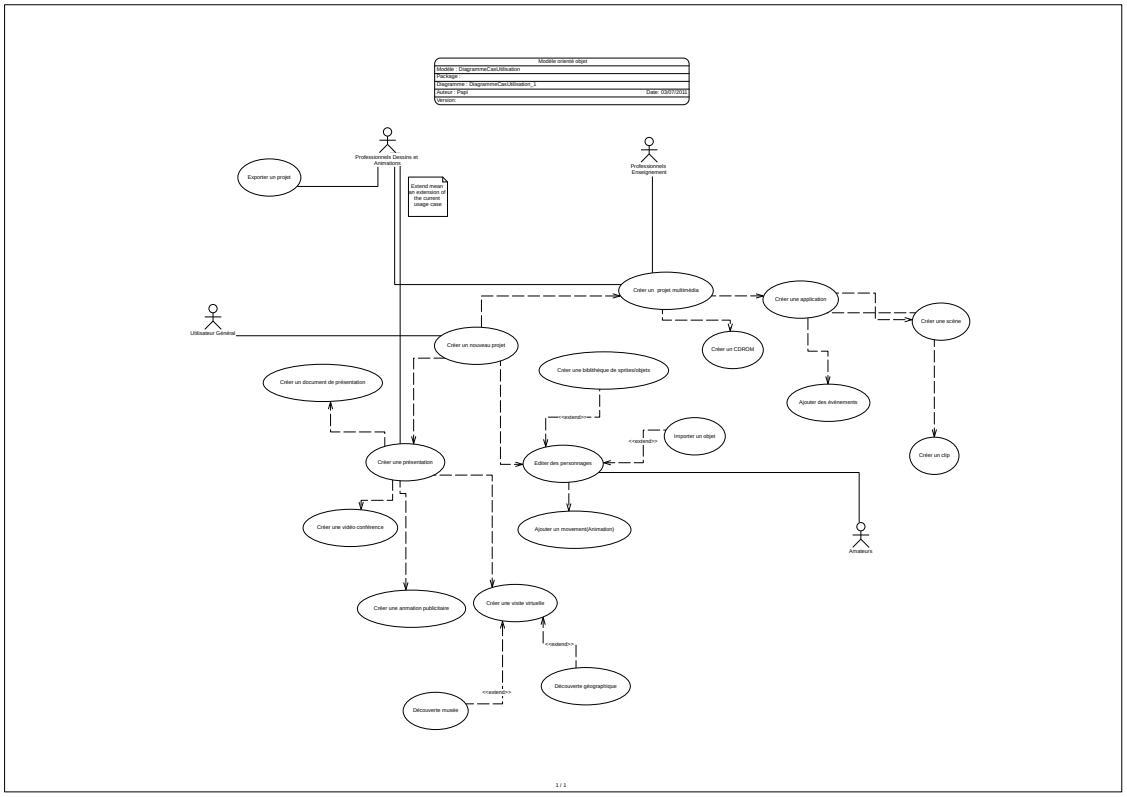

FIGURE II.2 – Utilisation Ultimaade

Chapitre III

Classes candidates et Fonctionnement

III.1 Technologies et principaux éléments

Le développement du logiciel fera intervenir les technologies et tiendra compte des considérations suivantes :

- Développement en C++/QT
- Sprites et Scènes doivent être vus de la même façon. À savoir un enchènement, assemblage d'éléments
- S'inspirer des Docks Mac pour l'ergonomie de la GUI
- Implémenter un système de cache des objets pour les mises en scènes
- Jouer sur les canaux des images (principalement pour png), dans le module de dessin
- Les objets :
  - objet = desciptif XML(xdf) + Ressources binaires (voir la représentation de la structure des objects ultimaade ci-dessous)
  - pas de sérialisation des objets
  - une scène est une timeline qui gère des sprites/objets
  - un sprite/objet est une timeline qui gère des images
  - tous les objets se calculent et se redessinent. Ainsi, le comportement des objets est connu des seuls plugins qui les utilisent.

III.2 Classes candidate

Par souci de clarté, nous avons mis dans un document à part les diagrammes des classes, accessibles via ce lien

La description technique de ces classes rentre la documentation technique du logiciel.

III.3 Fonctionnement

La création d'un jeu ou d'un programme avec Ultimaade passe par différents éditeurs, utilisant différents objets en mémoire.

III.3.1 Les éditeurs

- l'éditeur d'histoires(storyboard)
  Il s'agit en fait du module de montage.
  Un projet sous Ultimaade sera constitué d'une ou plusieurs scènes, qui correspondent aux différents « écrans ».
  Ainsi, pour un jeu, une histoire, la première scène pourrait être employée pour le menu principal, la deuxième pour le premier niveau, la troisième pour le tableau des records, etc.
  L'éditeur d'histoires permet d'ajouter ou de supprimer les différentes scènes d'un jeu ou d'une histoire, et d'en modifier les propriétés.
  Dans l'architecture du logiciel, le module de montage représenté par cet éditeur, utilisera essentiellement des scènes. Celles-ci auront été créées dans le module de scène.

- l'éditeur de scènes
  Chaque scène est constituée d'objets qui ont chacun leurs fonctionnalités.
  Par exemple, le héros d'un jeu et les ennemis seront représentés par des objets « Actifs », les décors seront représentés à l'aide d'objets de type « Décors », tandis que les objets « Texte » permettront d'afficher les dialogues.
  L'éditeur de scènes permettra d'ajouter ces objets et d'en modifier les propriétés.
  De façon globale, l'éditeur de scène défini par le module de scène, est au centre de la création des projets.
  Il utilise les objets issus de l'utilisation de tous les autres modules ci-dessous

- l'éditeur d'évènements
  L'éditeur d'évènements permettra de « programmer » les différents objets de la scène.
  Il permettra d'ajouter les actions que peuvent effectuer un objet.
  Cet éditeur permettra une approche visuelle de la programmation.

- l'éditeur d'objets graphiques (module de dessin)

- l'éditeur d'objets animés (module d'animation)

III.3.2 Les objets

Différents objets seront utilisés par les différents modules, comme le montre les diagrammes des classes. Chaque objet peut afficher des images ou animations dans la scène.

De plus, l'insertion d'un objet dans la scènes permet d'avoir accès aux conditions, actions ou expressions qui y sont liées.

Exemples d'objets avec leurs fonctionnalités de base.

- Actifs : permettent d'afficher une ou plusieurs animations 2D, un objet statique sur la scène, de stocker plusieurs variables et chaînes. Ces objets pourront être employés en guise de personnage, d'ennemi, de bonus, etc.
- Décors : images statiques. Ce sont les seuls objets qui ne donnent accès à aucune condition, action ou expression. Mais il est possible de tester s'il y a collision entre les objets de type « Décors » et les autres objets. Par exemple, des plates-formes statiques dans un jeu de seront représentées par des Décors.
- Compteurs : permettent de stocker une variable et de l'afficher de diverses manières (nombres, barres...)
- Textes : permettent d'afficher du texte et de stocker plusieurs chaînes de caractères.
- Tableaux : permettent de stocker un tableau de nombres ou de chaînes multi dimmensionnel, et de le sauvegarder sur le disque dur. ça pourrait être souvent employé pour les sauvegardes dans les jeux.
- INIs : permettent de stocker diverses informations dans un fichier INI. La plupart des langages de programmation est munie d'outils de parsing pour ce type de fichier. Ces objets seront plus utitilisés pour les paramètres de l'application que pour la création des objets.

La description de l'objet fait partir de la structure qui le représente en mémoire.

Le schéma ci-dessus décrit les relations entre les différents éditeurs(modules), et leurs utilisations des objets :

III.3.3 Anatomie d'un roman visuel

Un roman visuel (visual novel) sera constitué d'une part, de composants visibles, il s'agit de ce que le lecteur du roman voit. Ce sont les dessins/sprites et animations.

De l'autre côté il y a les composants qui sont sous le capot, c'est à dire toute la partie qui anime et fait défiler les images. Il s'agit là des composant invisibles, et c'est à ceux là que nous nous intéressons dans cette partie.

Car c'est de ce côté là que les jeux vidéo et les visual novels (qui sont la vocation même d'ultimaade) ont le plus de points communs.

Puisqu'en réalité ils partagent la même technologie.

L schéma ci-dessus illustre et décrit la structure minimale d'un visual novel type créé avec `Ultimaade`

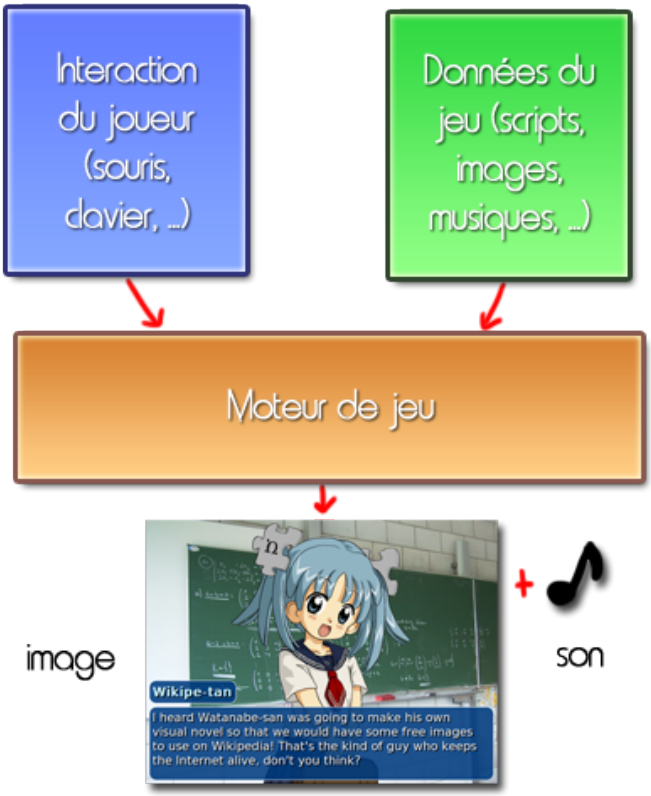

FIGURE III.1 – Anatomie d'un roman visuel

Parmi les différents types de ressources voici celles que l'on retrouvera principalement :

- les images, il s'agit des background, des sprites, et de toutes les images composant l'interface utilisateur (comme une boîte de dialogue par exemple).
  Il s'agira de fichiers BMP, JPEG, PNG.
- les sons, comme le bruit d'une porte qui s'ouvre ou une réplique doublée par un acteur. Il pourrait s'agir de fichiers WAV.
- les musiques.

  Attention c'est différent des sons !

  La différence est en plutôt technique, puisqu'une musique est juste un son qui est plutôt long (plusieurs minutes) et qui doit donc être géré différemment par le moteur de visual/jeu. Les formats de fichier sont là aussi bien connus : MP3, OGG, etc.
- les scripts. Il s'agit du script du roman visuel. Comme le script d'un film : c'est à dire le déroulement des scènes, la narration et les dialogues. Il s'agit de formats textuels. Ulltimaade stockera les scripts sous le format xml.

Les choix réalisés par le joueur ou lecteur du visual, et leurs conséquences sont exprimés via des arbres de dialogues (`dialog tree`).

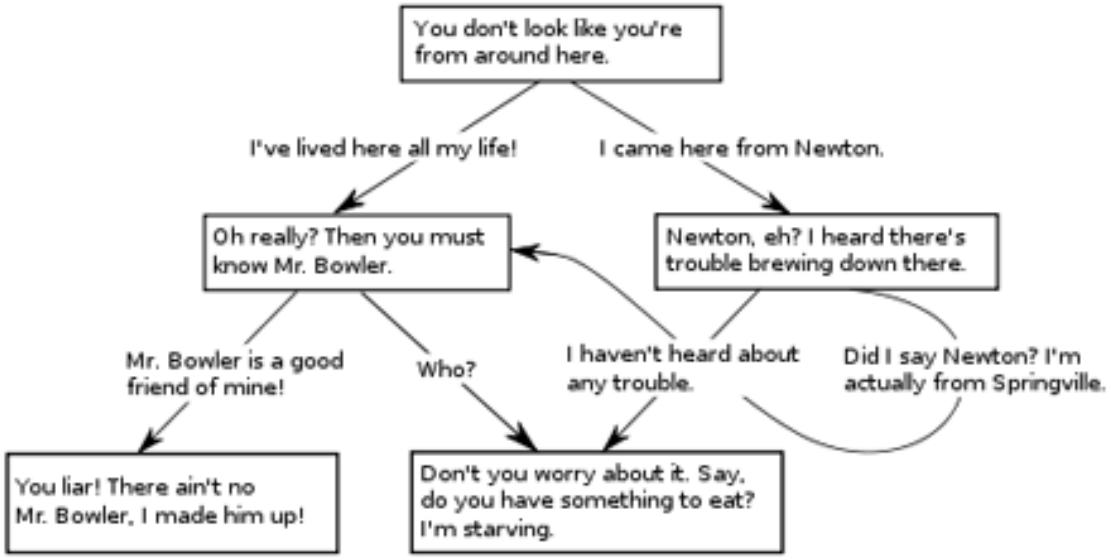

FIGURE III.2 – `Arbre de dialogue d'un roman visuel`

III.3.4 Structure des données représentant les objets en mémoire

Cette partie vise à décrire l'organisation d'un projet réalisé avec le logiciel Ultimaade.

La hiérarchie des entités représentant le projet suit la logique de dépendance des modules du logiciel. Ci-dessous le modèle conceptuel de données correspondant.

Le modèle physique qui en découle sera implémenté par un système de gestion de base de données locale. Il ne s'agira en rien d'une base données relationnelle classique, et accessible en réseau telle que MySQL, PostGreSQL, ou autre.

Les données seront stockées dans des fichiers XML en relation les uns avec les autres, ou dans une base SQLITE3 (un simple fichier local). Le choix sera fait en fonction des contraintes de développement.