Rapport pour le projet Arkanoid

Contexte du projet

Le but du projet est de réaliser un jeu vidéo appelé Arkanoid. Et ce en utilisant les principes vus lors des cours de programmation avancée. Le projet doit être réalisé en C++ avec les versions antérieurs de ce langage (version 11 et plus). Bien évidemment il fallait utiliser au maximum les design patterns pour avoir un ensemble clair et bien structuré.

Compilation

La compilation de ce projet a été automatisé par mes soins en utilisant l'outil CMake. Ce qui rend simple la compilation sur de multiples plateformes.

Structure du projet

Nous avons séparé le projet en plusieurs sous répertoires traitant chacun d'un cas particulier. Par exemple, les entités se situent tous dans le répertoire « Entity » tandis que les scènes (écrans) se situent dans le répertoire « Scene ».

Entity-Component-System (ECS)

J'ai décidé de partir sur le principe « nouveau » (ou pas vraiment) de l'ECS. Cela nous permettait de pouvoir concevoir les entités de nos choix sans pour autant complexifier le projet. Mais pourquoi sommes-nous partis sur ce principe, me diriez-vous, sachant que le projet en soit n'est pas fort compliqué. C'est justement pour un but pédagogique, car je voulais apprendre et comprendre ce principe c'est pourquoi en tant que défi nous sommes partis sur cette voie. J'ai eu affaire à de multiples soucis tout au long du projet, car ce principe était nouveau pour moi.

D'ailleurs une vidéo explique très bien pourquoi utiliser les compositions plutôt que les héritages, ce qui est un des principes de l'ECS.

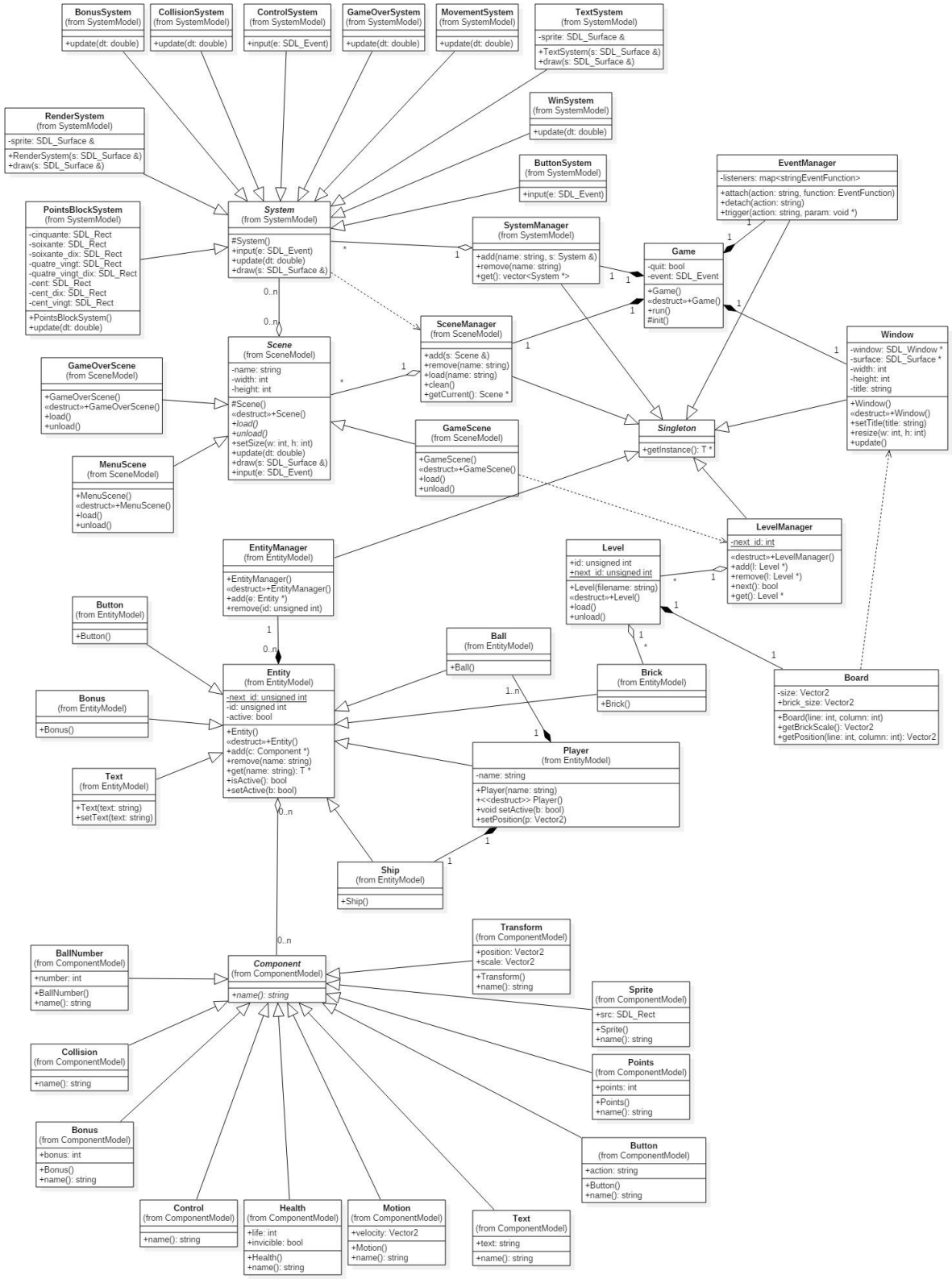
Lien de la vidéo : https://youtu.be/wfMtDGfHWpA

Design patterns

Nous avons essayé au maximum d'utiliser les designs patterns que nous avons réussis plus ou moins à comprendre. Par exemple, les singletons qui sont très simple à mettre en pratique et d'autres plus difficiles comme les observateurs, les médiateurs et encore d'autres.

UML

Nous avons conceptualisé le projet en utilisant un diagramme de classe permettant de représenter toutes les classes et leurs liens. Voici un extrait de l'UML, qui est également disponible sur le dépôt git.



Travail collaboratif

Pour permettre à chaque membre de notre équipe (2 personnes) de pouvoir travailler depuis chez soi, en ayant à disposition une plateforme adéquate, facile d'utilisation et répertoriant les changements

effectués au sein du projet, tout en ayant les derniers changements nous avons décidé d'utiliser l'outil de « versionning » git.

Difficultés

La principale difficulté fut les erreurs de segmentation et les autres erreurs qui surviennent à l'exécution. Donc il fallait constamment débugger le programme pour pouvoir corriger les soi-disant erreurs. Sinon on avait également eu des soucis de conceptualisation où on a eu de multiples idées et on devait donc les implémenter pour prendre la meilleure des possibles.

Tâches non réalisées

A l'heure actuelle, les bonus ne sont pas encore réalisés même si tout ce qu'il faut est déjà en place (entité et composants). De plus, le mode multijoueur n'est également pas présent et il n'y a pas de classement pour les scores. Il manque également la texture de fond pour le jeu. Et la collision n'est pas des plus performantes (sans réflexion).

Conclusion

Dans la globalité c'était un bon projet mais qui demandait beaucoup de temps pour avoir un projet final optimisé et avec toutes les options demandées. Cela m'a permis d'en apprendre beaucoup sur l'Entity-Component-System (ECS) et va rudement me servir pour les prochains projets personnels à venir.