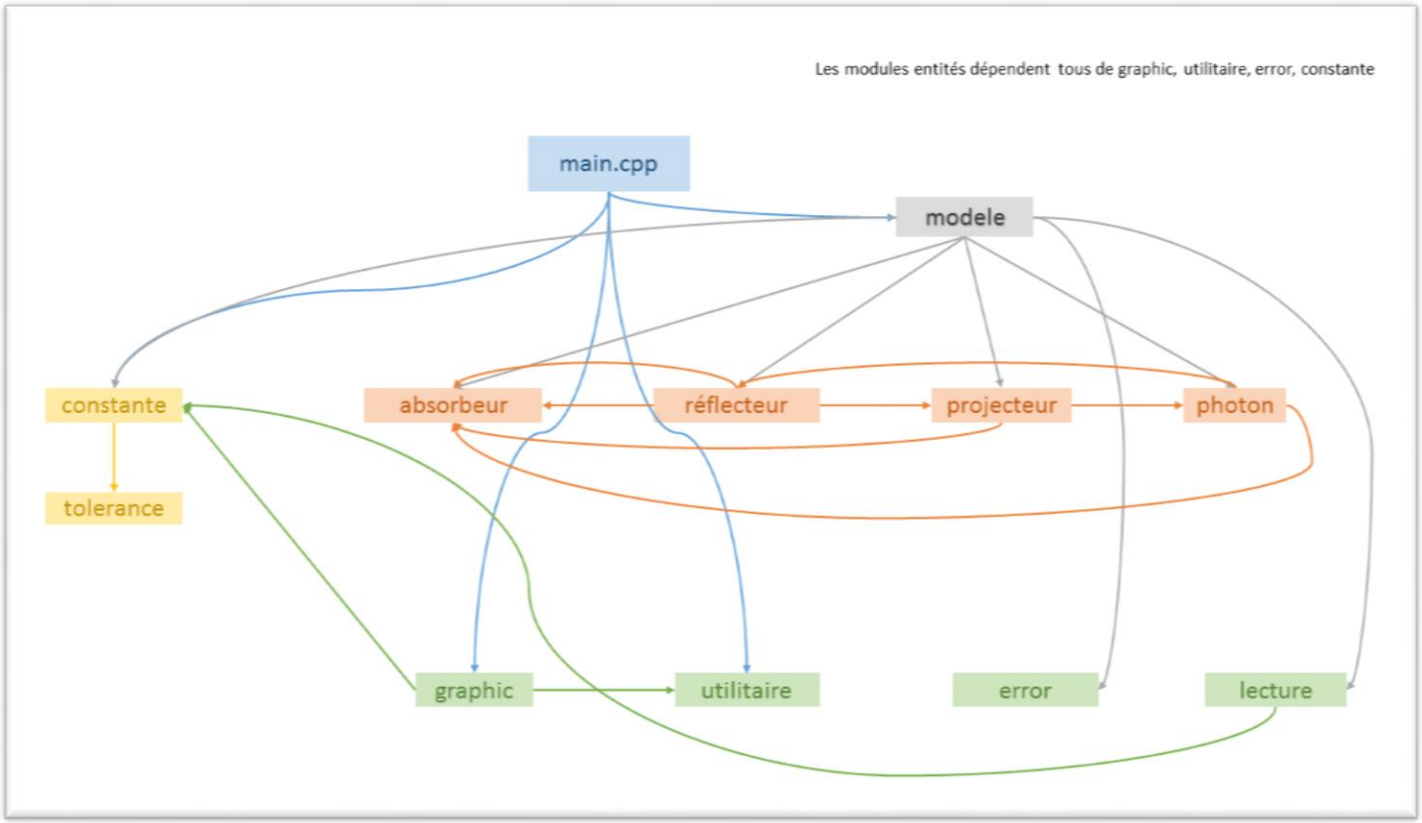
RAPPORT FINAL

Architecture logicielle

Nous avons ajouté des fonctions principalement dans modèle, ce sont les fonctions qui servent à sélectionner une entité ainsi qu'une fonction pour créer une entité. Différentes fonctions ont été rajoutées dans le fichier main afin de différencier le cas Graphic et le cas Final. Nous avons bien sûr aussi compléter les fonctions stub du rendu 2. En lien avec les modifications du module modèle, nous avons ajouté ou complété différentes fonctions dans les modules entités afin de pouvoir sélectionner, créer, et supprimer les dites entités. Entre les modules de ces dernières, nous avons rajouté des dépendances (cf figure ci-dessous) afin de pouvoir faire les vérifications nécessaires lors de la création d'une entité pendant l'exécution du programme.

Architecture logicielle finale



Gestion des données

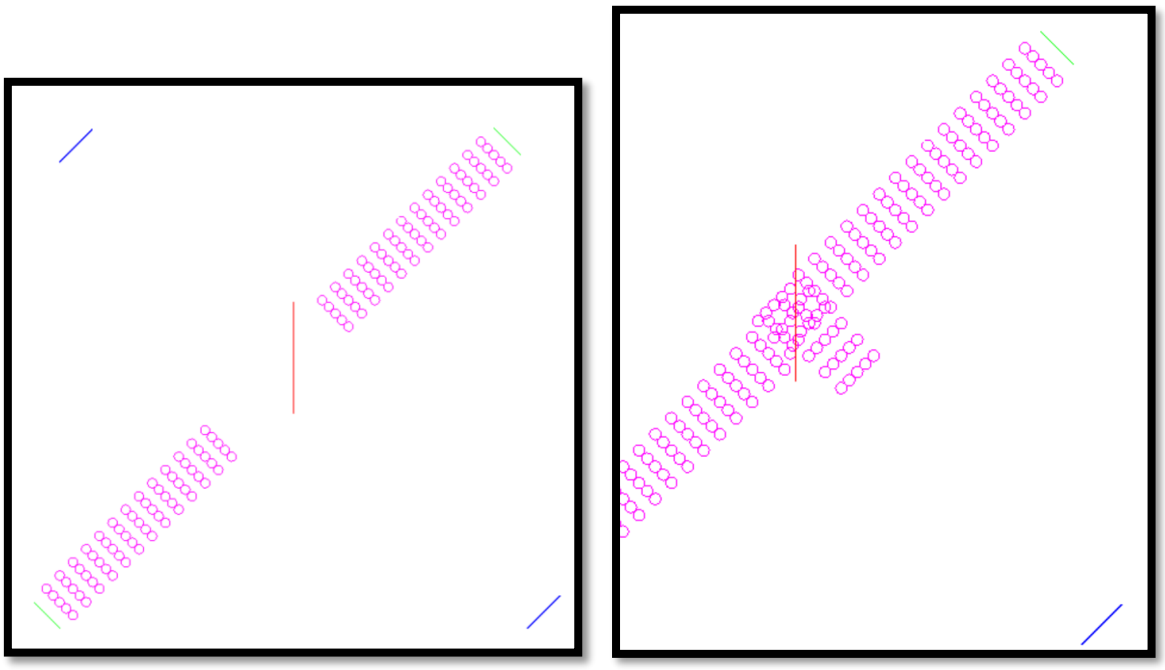
Au niveau des structures, nous avons minimisé le cout mémoire en arrangeant les différents champs stockés en fonction de leur taille mémoire et d'une architecture 32 bits. Afin d'éviter une explosion de l'utilisation de la mémoire, nous avons utilisé des listes chainées pour chaque entités. Chaque élément d'une entité possède son propre identifiant numérique. Par rapport à un tableau cela évite d'avoir des emplacements de mémoire occupés pour rien. L'ajout d'un élément se fait en tête de liste car c'est le cout calcul le plus faible.

Coût calcul et coût mémoire

Le coût mémoire est linéaire et ne dépend que du nombre de projecteurs (O(nbProj)). En effet, au cours d'une mise à jour, la mémoire occupée par les absorbeurs, projecteurs et réflecteurs est constante contrairement à celle des photons qui sont créés. Puisqu'ils sont créés au nombre de 5 par projecteurs, alors le coût calcul serait de 5*nbProj mais le terme dominant est nbProj.

Le coût calcul est de O(nbProj+nbPhot*(nbRefl+nbAbso)). La première partie concerne la création des photons alors que la deuxième partie s'occupe du déplacement des photons.

Illustrations



Comme on peut le voir sur les images ci-dessus, les photons sont réfléchis sur le reflecteurs.

Méthodologie et conclusion

Nous nous sommes répartis le travail à faire d'après la donnée et nous nous voyions régulièrement pour faire le point et pour s'aider mutuellement. Nous nous ne sommes pas répartis le travail par module mais plutôt par tâches à faire. La charge de travail d'Alix était moins conséquente que celle de Sven, bien que le temps consacré par chacun fut quasiment le même, Sven ayant plus de facilité. Alix s'est occupée de la lecture des fichiers, de la vérification des entités et d'une partie GLUI. Sven s'est occupé des réflexions, des créations

d'entités et de certaines actions utilisateurs. Le reste a été fait conjointement On a commencé par les modules des entités, puis le module modèle avant de toucher au graphic et au main. On a fait des fonctions vides sur les parties pas encore codées afin de pouvoir tester la partie en cours de travail. Puis, nous avons parfois travaillé en créant un programme appart comme pour la fonction vérification, ce qui nous a permis de la tester avec différents vecteurs que nous choisissions. Avec le recul, nous aurions peut-être pu réduire les dépendances entre nos modules au détriment du coût calcul. Le bug le plus fréquent et le plus difficile était la plupart du temps lié aux maths du projet (vérification des intersections, réflexion…), résolu en faisant des printfs et en réfléchissant à une autre alternative que celle proposée dans la donnée ou dans le rendu public.

Pour conclure, le fait des cours sur la première moitié du semestre est positif. Néanmoins il serait agréable d'avoir des machines virtuelles qui plantent moins souvent, ou la possibilité d'utiliser des terminals virtuels. Les rendus intermédiaires publics ne nous ont pas été utiles, néanmoins c'est une bonne idée.