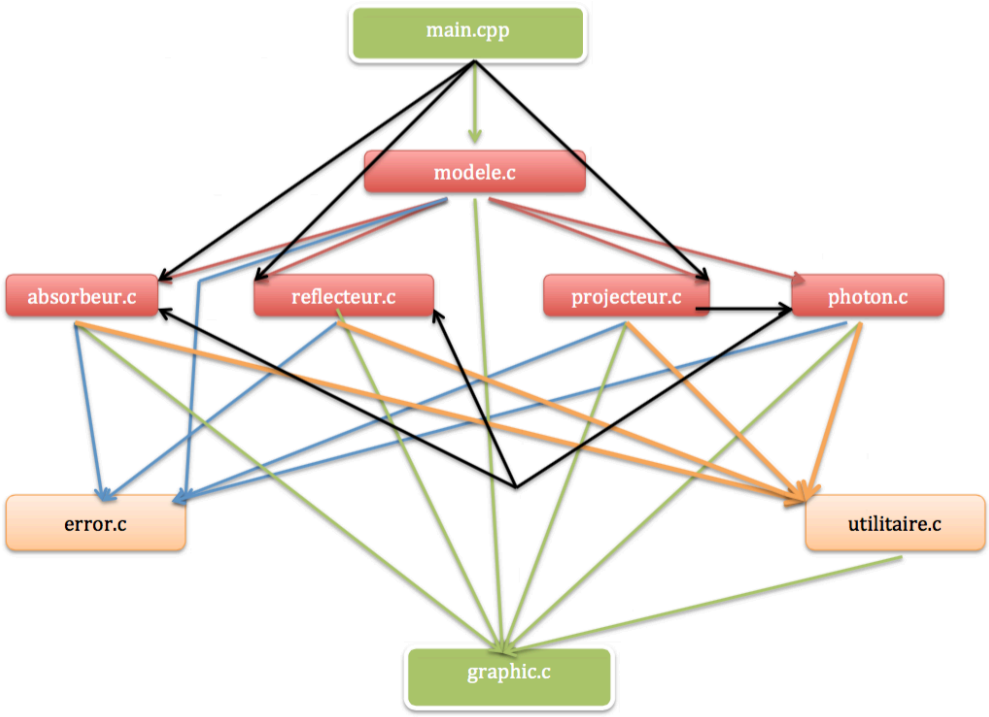
David Sanchez Del Rio EL 248127
Frank Dupont EL 264447

Rapport du rendu 3
Fiat Lux

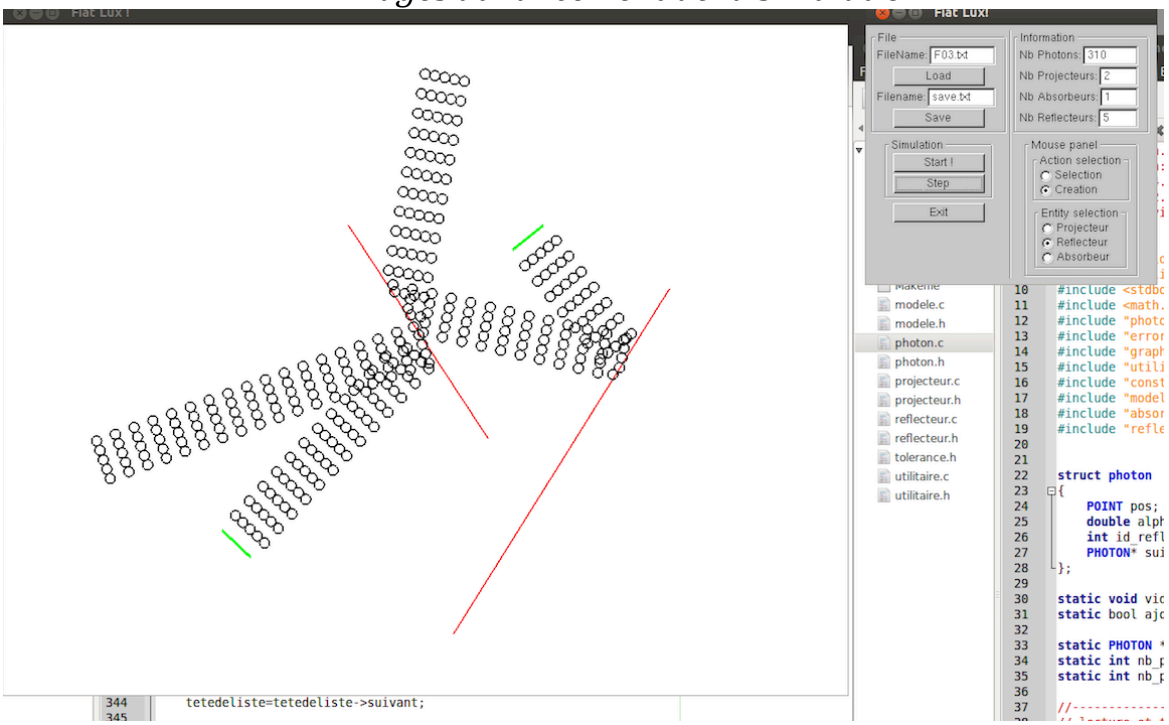
Dans le cadre du rendu final du projet, nous avons du changer quelque peu l'architecture logicielle du rendu 2 pour mieux pouvoir l'adapter à notre projet. En effet, nous avons du rajouter tout d'abord la dépendance entre photon.h et projecteur.h par soucis de simplicité lorsque des photons doivent être créés et projetés. Nous avons du également rajouter les dépendance des modules projecteur.h, reflecteur.h et absorbeur.h dans le module principal main. Nous avons pensé qu'il serait plus facile de créer, d'ajouter des entités ou de modifier leur nombre total si on travaillait directement dans leurs modules respectifs. Pour finir, après un changement de module pour notre fonction de réflexion, nous avons étés obligés de rajouter les modules absorbeur.c et reflecteur.c dans le module photon.c. En effet, nous avons pensé que pour mieux gérer les positions, les déplacements et les créations/destructions des photons au cours de la simulation il était préférable de tout faire dans ce module, même si cela impliquait de rajouter deux autres dépendances. L'architecture finale de notre code ressemble au dessin ci-dessous.

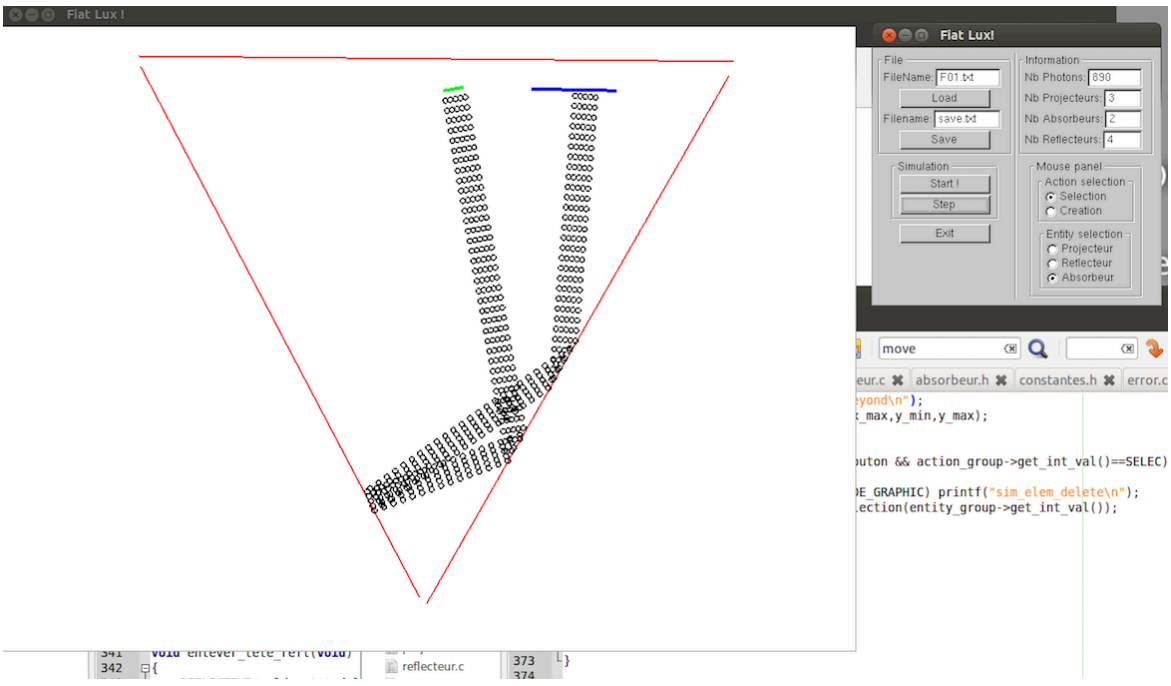


Architecture logicielle de notre projet : les flèches en noir sont celles que nous avons rajouté par rapport à Fig.11.b

Pour la gestion des données, nous avons pour la majorité des cas utilisé des listes chaînées pour stocker les entités différentes. Grâce aux fonctions d'ajout et de retrait de liste chaînée, nous avons pu stocker efficacement toutes les données présentes dans les fichier ainsi que les données qui se créent/se détruisent au fil de la simulation (points d'intersection, photons créés/détruits, création de reflecteurs etc…).

Images du lancement de la simulation :





Pour un « pas » de simulation nous avons approximativement un coût de calcul de

$$O(nbP*(nbR+nbA)+nbP)$$

Bien sûr, ce coût calcul est très approximatif, vu qu'il ne prend pas en compte tous les paramètres et varie très fortement en fonction du nombre de réflexions des photons à chaque étape. Ici, nous avons pris le cas de complexité qui est effectué pour une seule réflexion.

Responsabilité des modules :

	main.cpp	modele	photon	absorbeur	reflecteur	projecteur	graphic
Frank	1	0	1	0	0	0	1
David	0	1	0	1	1	1	0

Organisation du travail :

C'est la première fois que nous devions faire un travail aussi long en groupe. Nous avons essayé de séparer notre travail au maximum, pour pouvoir ensuite le mettre en commun et ne pas perdre trop de temps. Le but était donc de séparer les consignes en sous parties le plus indépendantes possible (l'exemple typique est la partie graphique et la partie vérification pour le rendu 2). Nous avons décider de commencer en général par les tâches les plus simples, qui sont souvent associées aux modules des entités (création, lecture, suppression…) avant de s'attaquer aux modules de plus haute importance qui permettent d'unifier tous ces modules pour que la simulation marche correctement.

Il n'y a pas vraiment eu de « plus gros bug » mais plutôt des bugs constants : dès qu'on en enlevait un d'autres apparaissaient. Si il y a bien une chose qu'on a appris avec le recul, c'est de ne pas sous estimer le temps que met la recherche de bugs et essayer de finir le projet plus en avance la prochaine fois.

Pour conclure, nous pouvons affirmer sans aucun doute que la complexité et la longueur du travail demandé ne sont pas à négliger. Mais néanmoins c'est un projet intéressant et efficace pour pouvoir apprendre à bien programmer. Les ressources informatiques et les assistants à dispositions rendent la tâche plus simple, même si elle reste complexe de toutes façons.