Approximation d’ éclairage indirect en temps réel: Spécifications

Blaise Cardonne, Gauthier Bouyjou, Valentin Camus, Rihab Elrifai, Sylvain Durand
Encadrant / Client : François Desrichard

Axes de la présentation

1. Contexte
2. Description du projet et objectif global
3. Cahier des charges
    - Définition du besoin
    - Exigences fonctionnelles
    - Exigences non fonctionnelles
    - Contraintes
    - Tests et scénarios de validation
4. Vue générale du système
5. Modules du système et tests de validation
6. Planning prévisionnel et délais
7. Analyse des risques

Contexte

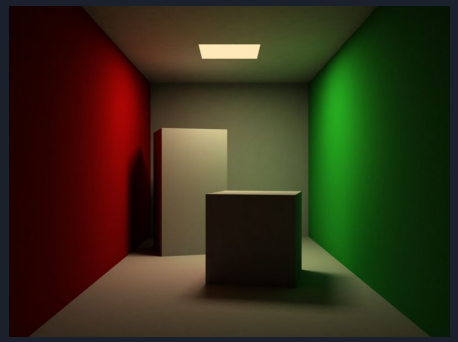
Parties prenantes :

- Client :
  - François Desrichard

- Équipe :
  - Blaise Cardonne
  - Gauthier Bouyjou
  - Valentin Camus
  - Rihab El Rifai
  - Sylvain Durand

Problématique générale :

Reproduire de l'éclairage globale de manière la plus réaliste possible avec une contrainte majeure qui est la production d'images en temps réel.



Contexte

État de l’art :

- Éclairage indirect
  - *Reflective shadow maps,* Carsten Dachsbacher et Marc Stamminger, 2005
  - *Imperfect shadow maps for efficient computation of indirect illumination,* Ritschel et al., 2008
- Rendu plus réaliste
  - *Virtual spherical gaussian lights for real-time glossy indirect illumination*, Yusuke Tokuyoshi, 2015

Description du projet et objectif global

Description du projet :

- Moteur de rendu minimaliste
  - Import de scène standardisée
  - Contribution des lumières de type BRDF (gestion des matériaux)
  - Éclairage direct avec ombrage
- Implémentation des articles de recherches

Objectif global :

- S’approcher de la solution que Square Enix propose dans son article.

Cahier des charges

Définitions du besoin :

- Besoin de visualiser des images hautements réalistes en un minimum de temps (nécessaire pour être compétitif dans le domaine du jeu vidéo).

Cahier des charges

Exigences fonctionnelles principales et optionnelles avec priorités de développement

FP = Fonction Principale FO = Fonction Optionnelle

Fonction	Description	Priorité
FP1	Chargement de scène 3d	Forte
FP2	Éclairage directe	Forte
FP3	Positionner caméra dans la scène pour comparaison avec une image de référence	Forte
FO1	Déplacement des sources de lumières	Faible
FP4	Choix de la méthode d’éclairage indirecte à utiliser	Forte
FP5	RSM (Reflective Shadow Maps)	Forte
FP6	GS (Gaussienne Sphérique)	Moyenne
FO2	ISM (Imperfect Shadow Maps)	Faible
FO3	Sauvegarde d’une frame de rendu sur disque	Faible

Cahier des charges

Exigences non fonctionnelles :

- Performance :
  - Système temps réel (> 60fps) sur une machine de la salle U3-112
- Documentation :
  - Manuel d'utilisation livré lors de la recette
- Prouvabilité du fonctionnement du logiciel :
  - Résultats des jeux de test

Cahier des charges

Contraintes :

- Logiciel
  - Compilable Linux, Mac, Window
  - Version OpenGL >= 4.3 (compute shader) ou >= 4.1 MacOS
  - Version de la librairie standard C++ >= 11
  - Nouveau dépot Github

- Délais de rendu
  - Produit fini livrable avant le 25/02/2020

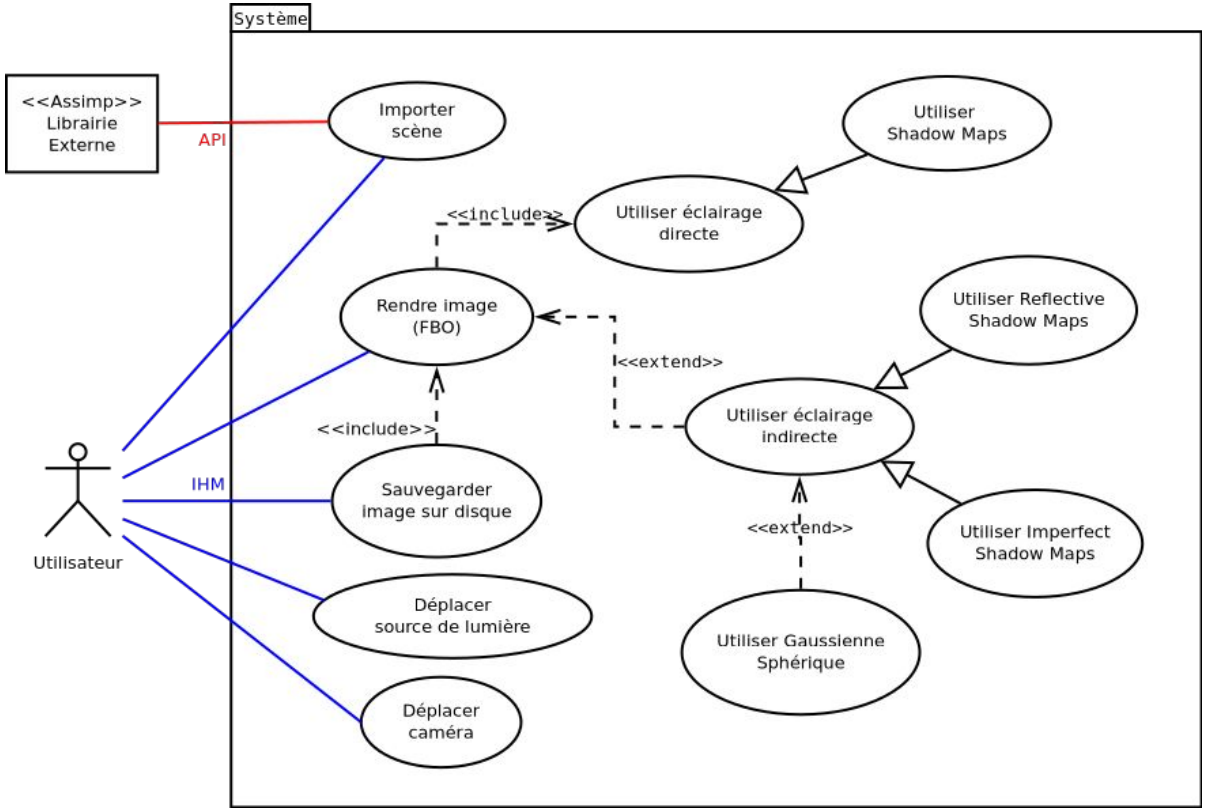
Cahier des charges

Tests et scénarios de validation des fonctionnalités :

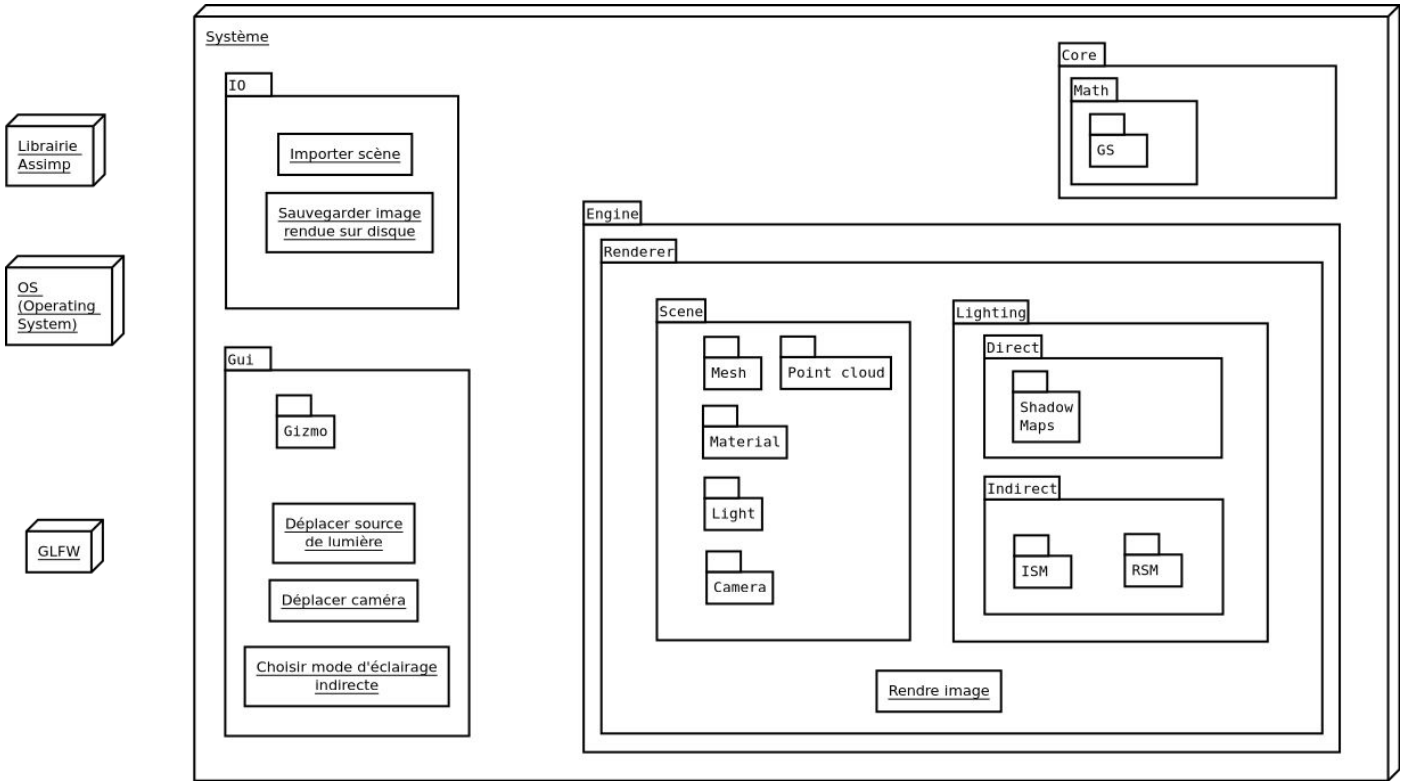
- Efficacité du rendu
  - Comparaison avec le logiciel PBRT

Vue générale du système

Diagramme de cas d'utilisation

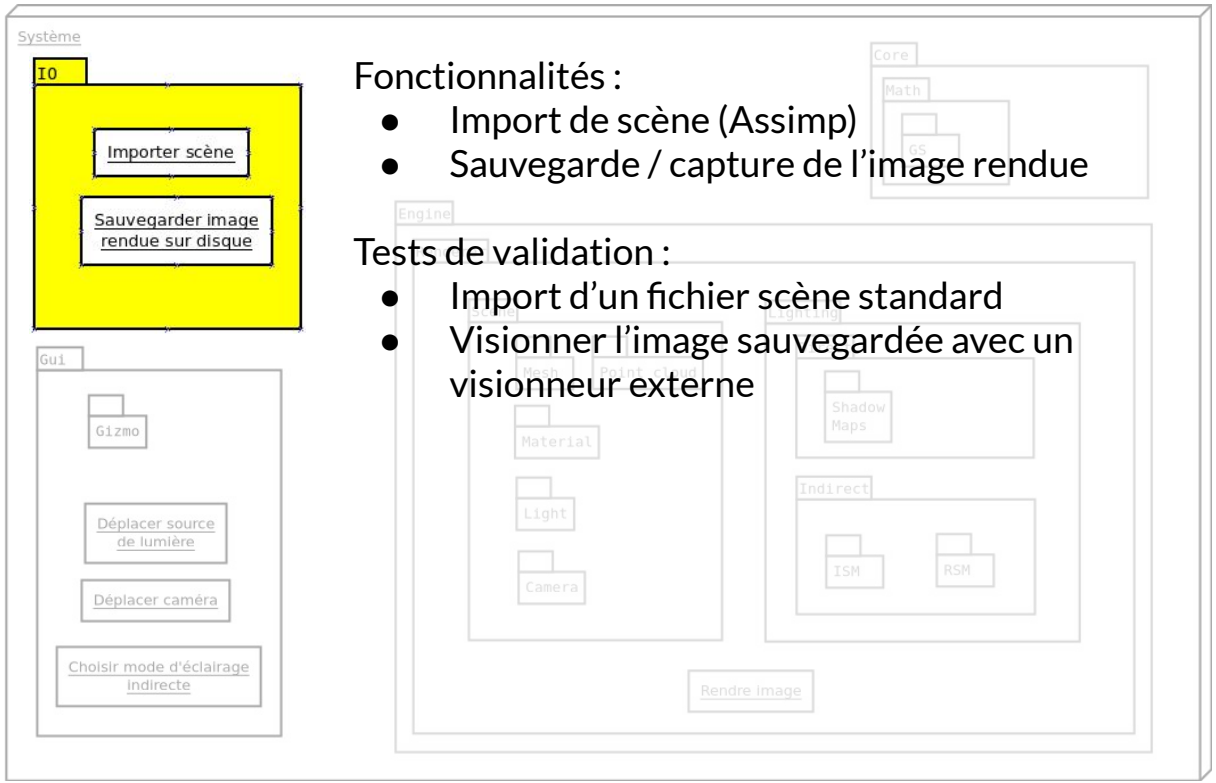


Modules du Système



Modules du Système

Tests de validation



Fonctionnalités :

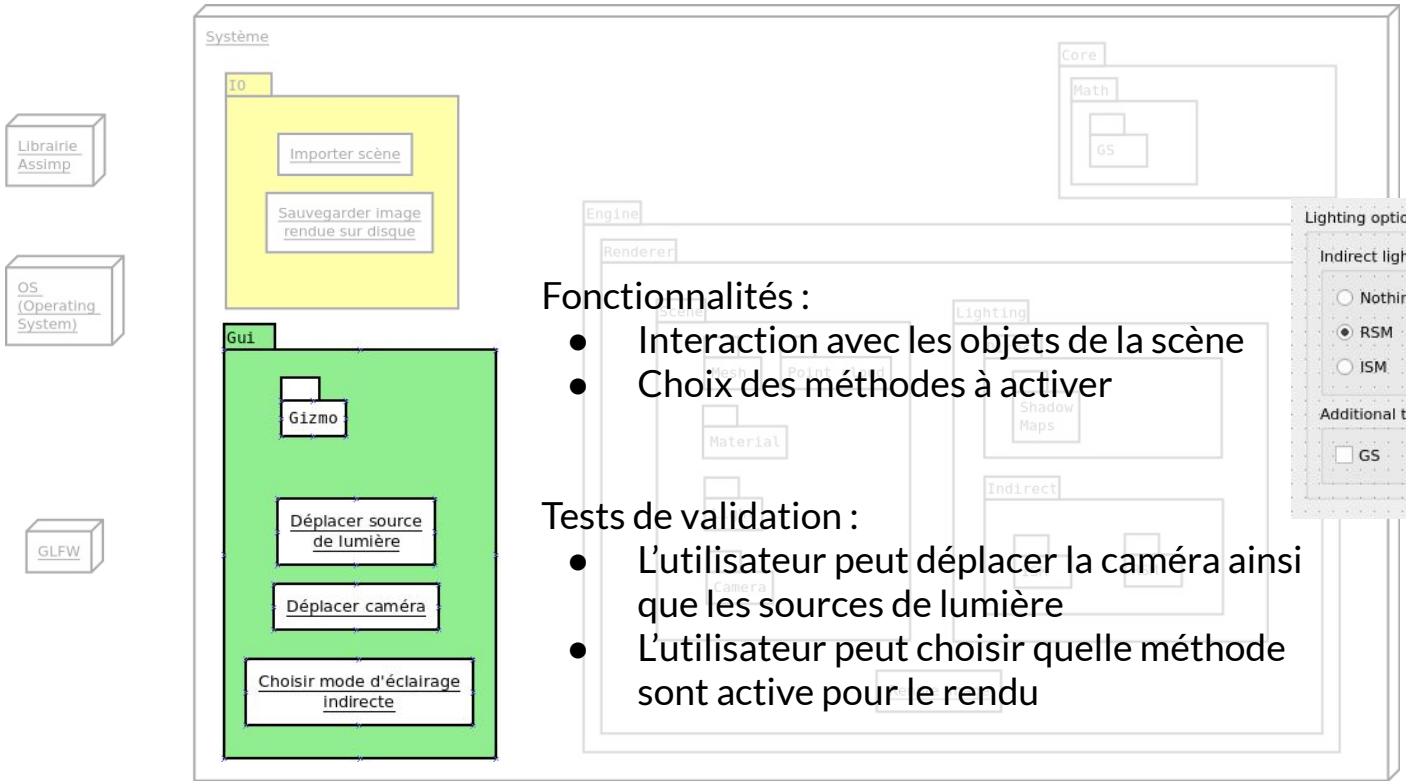
- Import de scène (Assimp)
- Sauvegarde / capture de l'image rendue

Tests de validation :

- Import d'un fichier scène standard
- Visionner l'image sauvegardée avec un visionneur externe

Modules du Système

Tests de validation



Fonctionnalités :

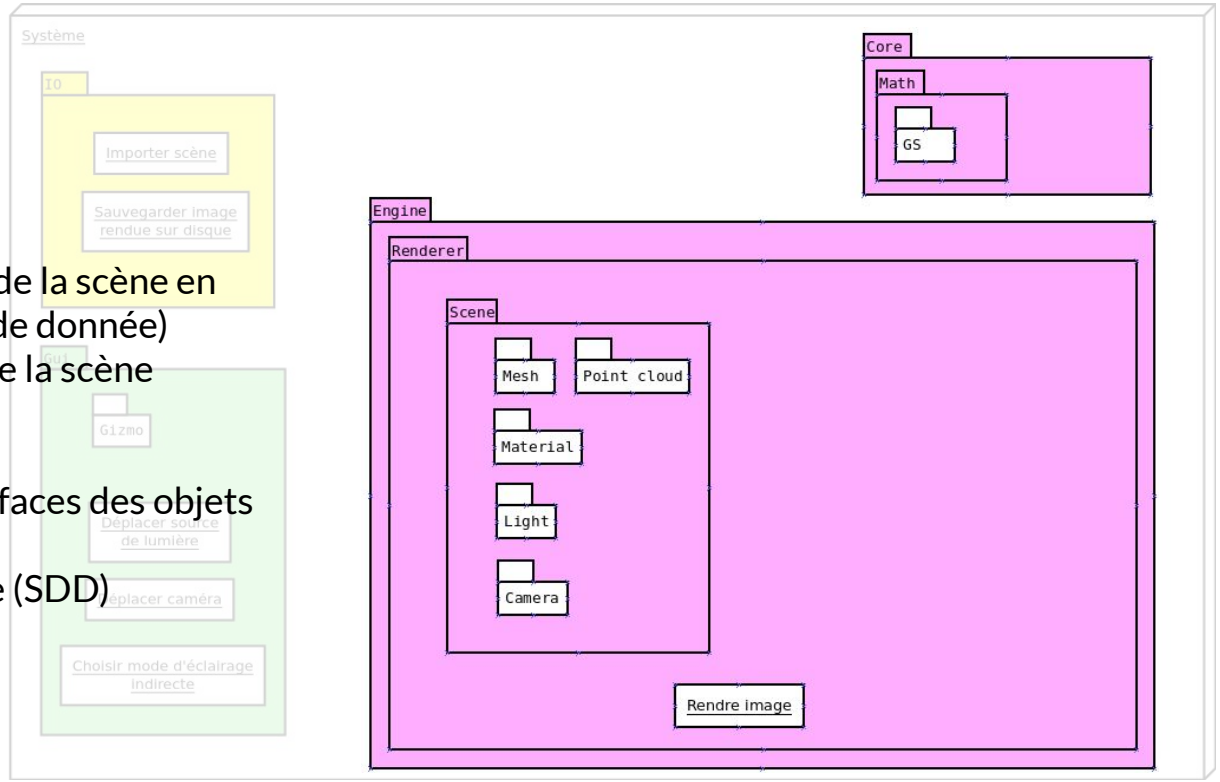
- Interaction avec les objets de la scène
- Choix des méthodes à activer

Tests de validation :

- L'utilisateur peut déplacer la caméra ainsi que les sources de lumière
- L'utilisateur peut choisir quelle méthode sont active pour le rendu

Modules du Système

Tests de validation



Fonctionnalités :

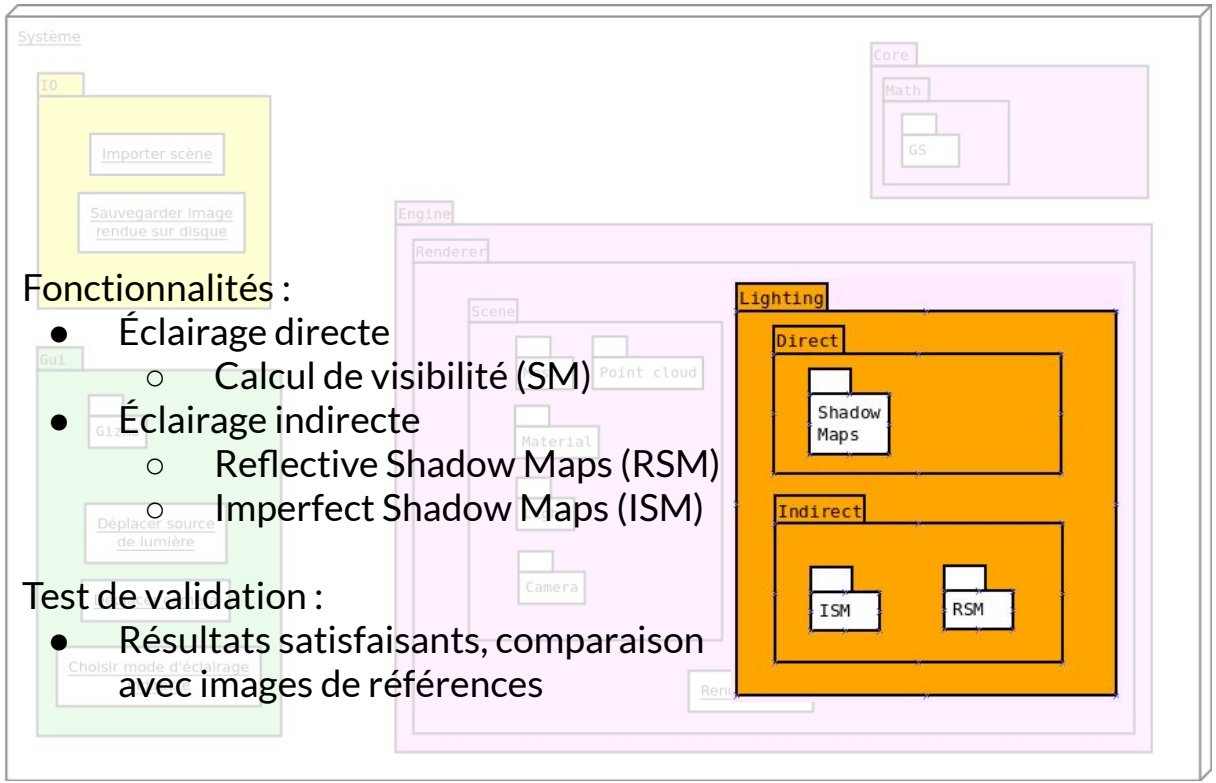
- Stockage des objets de la scène en mémoire (structure de donnée)
- Dessiner les objets de la scène

Test de validation :

- Visualisation des surfaces des objets correcte
- Pas de fuite mémoire (SDD)

Modules du Système

Tests de validation



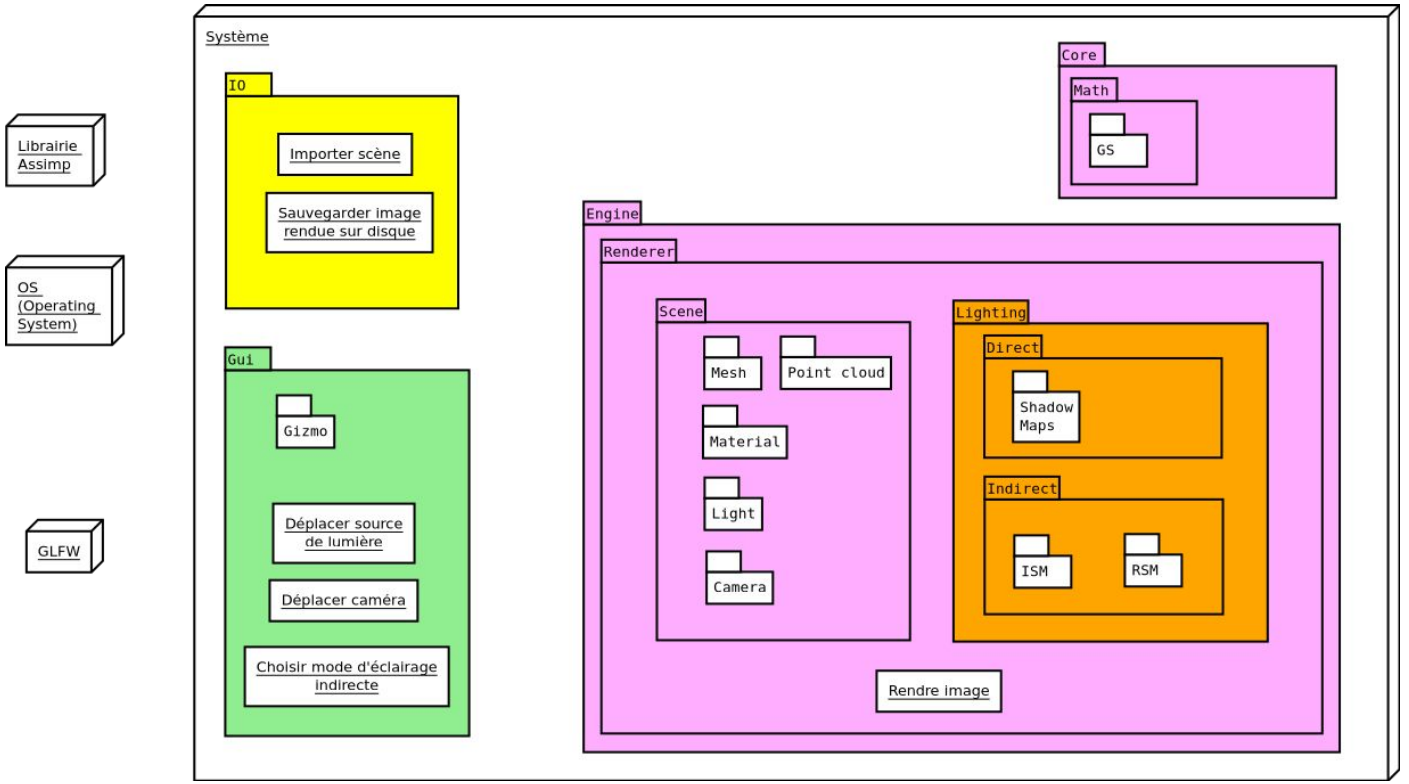
Fonctionnalités :

- Éclairage directe
  - Calcul de visibilité (SM)
- Éclairage indirecte
  - Reflective Shadow Maps (RSM)
  - Imperfect Shadow Maps (ISM)

Test de validation :

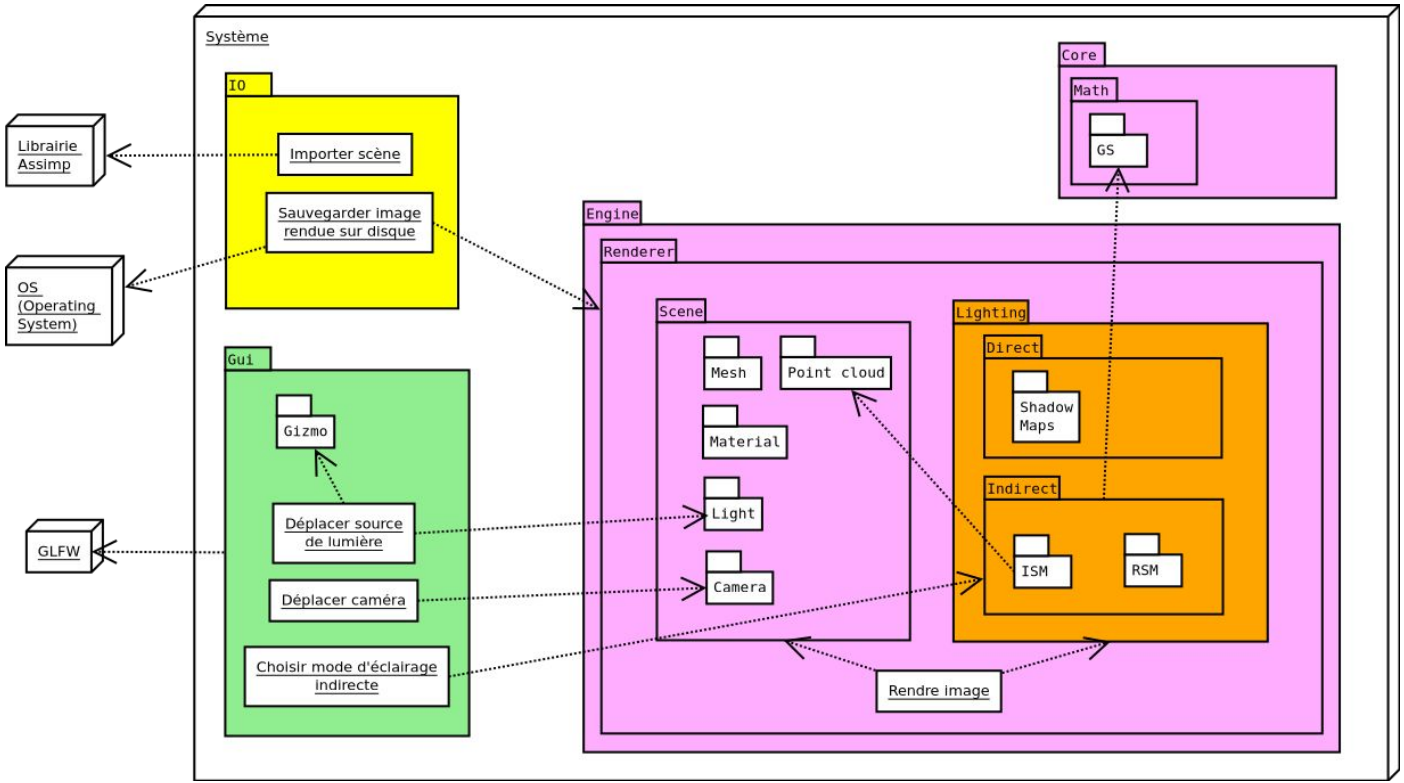
- Résultats satisfaisants, comparaison avec images de références

Modules du Système



Modules du Système

Dépendance entre modules



Planning prévisionnel et délais

- 4 itérations (2 semaines chacunes)
  - itération 1 : Dossier de conception (5 jours-homme)
  - itération 2 : 1er phase de développement (45 jours-homme)
  - itération 3 : 2nd phase de développement (51 jours-homme)
  - itération 4 : Recette, Manuel d'utilisation, site web, résultat des jeux de test (25 jours-homme)

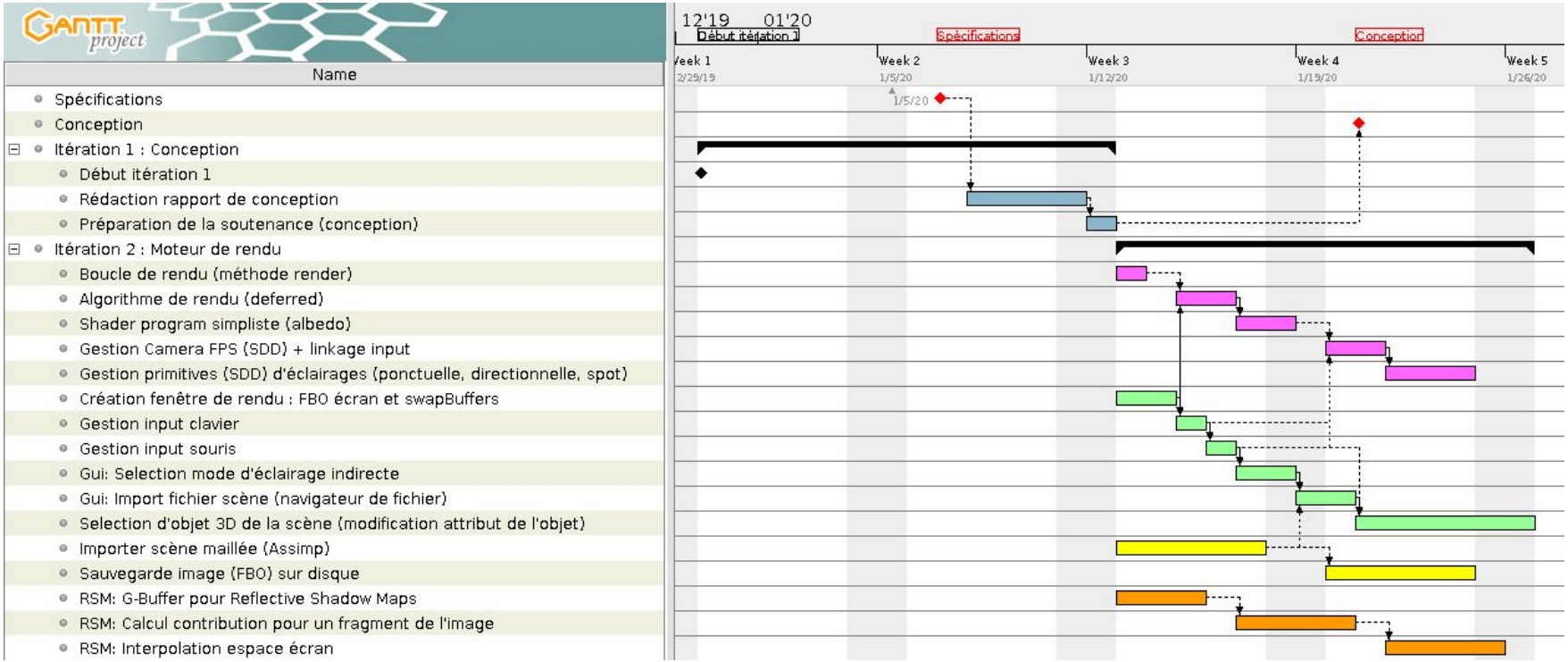
- Total : 126 jours-hommes

Module	Core Engine	Gui	IO	Lighting
Couleur				

Équipe

Planning prévisionnel et délais

Itération 1 et 2

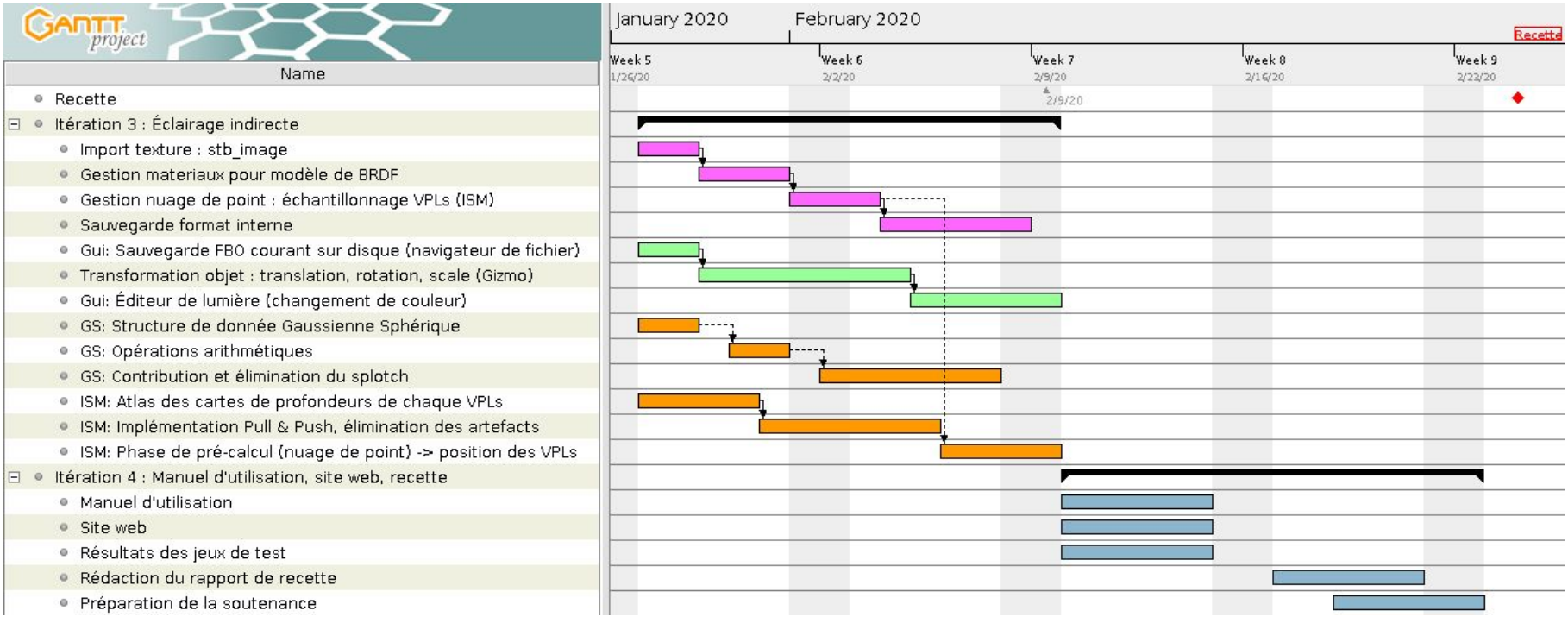


Module	Core Engine	Gui	IO	Lighting
Couleur				

Équipe

Planning prévisionnel et délais

Itération 3 et 4



Analyse des risques

Principaux risques : risques impactant

Risque	Probabilité d'apparition	Prévention	Solution
Indisponibilité / abandon d'un membre de l'équipe au cours du projet	20%	Faire en sorte que sur le planning initial des tâches soient faites en binôme	Les tâches en binôme deviendront des tâches monômes
Perte de temps sur le moteur alors que le coeur du sujet du chef d'oeuvre reste l'éclairage indirecte	40%	L'objectif est de faire une base moteur minimaliste afin de mieux se concentrer sur la partie de recherche	Faire une très bonne estimation des prévisions (tâches) et une conception détaillé précise évitant les questions lors des différents sprints
Dépassement des délais	60%	Effectuer une planification temporelle assez large avec marge au sein même d'une itération	Abandonner les fonctionnalités optionnelles, impossibilité d'ajouter des marges car développement serré sur 2 itérations
Sprint inefficace si on doit réviser des contrôles pour d'autres UEs	40%	Connaître les dates de contrôles à l'avance	Revoir la prévision afin de caler les itérations pour que les sprints de développement ne se trouve pas dans une phase de révision d'un CT

Analyse des risques

Principaux risques : risques mineurs

Risque	Probabilité d'apparition	Prévention	Solution
Performance décevantes (framerate)	70%	Utiliser des techniques d'optimisations	Se référer aux implémentations existantes et utiliser des outils de mesures de performances et de "profiling"
Non compatibilité entre OS	80%	Il faut que tous les membres de l'équipe puissent compiler sur son OS avant de commencer à développer	Faire un CMakeLists.txt multi OS
Rendu réaliste non performant (éclairage indirect)	50%	Bien comprendre les articles en relisant la partie méthode et algorithmes et en communicant entre membre de l'équipe si besoin	Se référer aux implémentations existantes, chercher plus de détails dans l'état de l'art, demander de l'aide à nos professeurs d'IG
Manque de communication dans le groupe	10%	Utiliser un bon outil de communication dès le départ	Utiliser un autre logiciel / outil de communication

Merci de votre attention.

Des questions ?