# Étude de Faisabilité et Architecture Logicielle : Portabilité des Fonctionnalités de Simulation de Studio Photo (Type set.a.light 3D) vers des Architectures DCC Polyvalentes

## 1. Introduction

L'industrie de la création visuelle assiste depuis quelques années à une convergence accélérée entre les outils de prévisualisation spécialisés et les moteurs de création de contenu numérique (DCC) généralistes. La demande initiale, qui consiste à développer une extension logicielle (plugin) capable de reproduire les fonctionnalités critiques de *set.a.light 3D* au sein d'environnements plus robustes et polyvalents tels que Blender ou Unreal Engine 5 (UE5), s'inscrit précisément dans cette dynamique. *set.a.light 3D*, développé par Elixxier, s'est imposé comme un standard pour la planification de prises de vue grâce à son approche "bac à sable" intuitive, permettant aux photographes de simuler des configurations d'éclairage sans les contraintes logistiques du monde réel.1 Cependant, sa nature de "jardin clos" limite les possibilités d'exportation, de rendu cinématique avancé, ou d'intégration dans des pipelines de production complexes incluant de l'animation ou des effets visuels.

Le présent rapport technique se propose d'analyser la viabilité du portage de ces fonctionnalités vers Blender ou Unreal Engine 5. L'objectif n'est pas seulement de comparer deux logiciels, mais d'évaluer deux philosophies de développement distinctes : l'approche Open Source centrée sur le scripting Python et le Ray Tracing offline (Blender), face à l'approche temps réel industrielle basée sur le C++ et la rasterisation avancée (Unreal Engine). Cette analyse couvrira la déconstruction des fonctionnalités cibles, l'audit des écosystèmes existants, l'étude approfondie de la faisabilité technique — notamment concernant la génération complexe de plans d'éclairage 2D techniques — et proposera une feuille de route de développement structurée.

### 1.1 Contexte et Enjeux de la Prévisualisation Technique

La prévisualisation technique ne sert pas uniquement à produire de "belles images", mais à valider des hypothèses physiques. Dans un studio réel, la lumière obéit à des lois immuables : la loi en carré inverse, la réflexion de Fresnel, et la température de couleur. *set.a.light 3D* a réussi à encapsuler ces règles physiques dans une interface utilisateur (UI) qui masque la complexité mathématique sous-jacente.2 L'enjeu du portage est donc double :

1. **Fidélité Physique :** Le moteur hôte doit être capable de simuler la lumière avec une précision photométrique (Lux, Lumens, Kelvin) et non arbitraire.
2. **Abstraction Ergonomique :** Le plugin doit masquer la complexité inhérente aux DCC (noeuds de shaders, UV mapping, baking) pour offrir une expérience utilisateur (UX) fluide, centrée sur la terminologie photographique (F-stop, ISO, Vitesse d'obturation).

L'analyse comparative qui suit déterminera laquelle des deux plateformes offre le meilleur compromis entre la puissance native de son moteur de rendu et la flexibilité de son API pour construire cette couche d'abstraction nécessaire.

## 2. Déconstruction Analytique du Logiciel de Référence : set.a.light 3D

Avant d'envisager le développement d'un plugin, il est impératif d'isoler et de caractériser les fonctionnalités qui constituent l'ADN de *set.a.light 3D*. Ce logiciel n'est pas un simple moteur de rendu, c'est un outil de planification opérationnelle. L'analyse des documents techniques et des retours utilisateurs permet d'identifier cinq piliers fonctionnels majeurs qui devront être reproduits.

### 2.1 Le Moteur de Lumière "Orienté Objet" et Photométrique

Dans la plupart des logiciels 3D généralistes, une lumière est une entité abstraite (un point, une zone) définie par une puissance en Watts ou une couleur RGB. Dans *set.a.light 3D*, l'utilisateur ne manipule pas des "Area Lights", mais des équipements virtuels qui correspondent à du matériel réel.1

* **Encapsulation Matérielle :** Une source lumineuse est l'agrégation d'une tête de flash (strobe) et d'un modeleur (light shaper). Le logiciel gère la modification du flux lumineux en fonction du modeleur choisi (Softbox, Octabox, Bol beauté, Snoot). Par exemple, une Octabox de 120 cm ne se contente pas d'adoucir les ombres ; elle crée une signature lumineuse spécifique avec un "hotspot" central et une atténuation graduelle vers les bords, souvent simulée par des profils IES ou des textures de projection.3
* **Données Photométriques :** Les lumières sont calibrées. Changer la puissance de "1/1" à "1/128" affecte l'exposition de manière prédictible, permettant de tester des ratios d'éclairage (ex: Key Light à f/8, Fill Light à f/5.6).2

### 2.2 La Caméra Virtuelle et le Triangle d'Exposition

La simulation de la caméra est le second pilier. Contrairement à une "caméra 3D" standard qui peut être parfaite, la caméra de *set.a.light* simule les contraintes physiques d'un boîtier DSLR ou Mirrorless.1

* **Interdépendance ISO/Ouverture/Vitesse :** L'image dans le viewport réagit en temps réel à ces trois paramètres. Si l'utilisateur règle sa caméra sur 100 ISO, f/16 et 1/200s sans éclairage suffisant, l'image doit être noire. Cette fonctionnalité est essentielle pour apprendre et valider l'exposition.
* **Optique et Profondeur de Champ (DoF) :** Le choix de la focale (ex: 85mm vs 24mm) influence la compression des plans et la distorsion du visage du modèle. La profondeur de champ doit être visualisable pour vérifier la netteté (focus peaking).1

### 2.3 Le Générateur de Plan Technique (Lighting Diagram)

C'est la fonctionnalité "signature" qui distingue *set.a.light 3D* d'un simple moteur de jeu. Le logiciel est capable de traduire la scène 3D complexe en un document 2D schématique et vectoriel, exportable en PDF ou JPG.1

* **Contenu du Diagramme :** Ce plan vue de dessus contient la position relative des lumières, des réflecteurs, du modèle et de la caméra. Il inclut des annotations automatiques : type de modeleur, hauteur du pied, inclinaison (tilt), puissance réglée, et parfois la référence de gelatine couleur utilisée.
* **Usage :** Ce document sert de feuille de route technique pour les assistants sur le plateau réel, permettant de reproduire le setup validé virtuellement. La capacité à générer ce document automatiquement, sans que l'utilisateur ait à dessiner, est une exigence critique du projet.

### 2.4 Ergonomie et Bibliothèque d'Assets

L'interface utilisateur (UI) est conçue pour minimiser la friction cognitive.

* **Drag & Drop :** L'ajout d'un élément dans la scène se fait par glisser-déposer depuis une bibliothèque visuelle. Il n'y a pas de curseur 3D à positionner manuellement via des coordonnées XYZ complexes pour un néophyte.2
* **Posing et Mannequins :** Le logiciel inclut des mannequins paramétriques dont on peut changer la pose, les vêtements, le maquillage et la coiffure, évitant à l'utilisateur d'avoir à apprendre le rigging ou le skinning.6

### 2.5 Analyse des Lacunes Actuelles (Gap Analysis)

Malgré ses forces, *set.a.light 3D* présente des limites qui justifient le besoin d'un plugin sur une autre plateforme :

* **Rendu "Plastique" :** Bien que réaliste, le rendu reste en deçà des standards de la photoréalité offerts par le Path Tracing moderne (Cycles) ou Lumen (UE5), notamment sur la peau (Subsurface Scattering) et les tissus.3
* **Manque de Flexibilité :** Impossible d'importer des décors complexes, de faire de l'animation cinématique avancée, ou de scripter des comportements personnalisés.
* **Fermeture :** Impossible d'exporter la scène vers d'autres formats 3D standards (USD, glTF) pour une reprise dans un pipeline VFX.

## 3. Analyse Approfondie de la Plateforme A : Blender

Blender, en tant que suite de création 3D open-source, présente une architecture modulaire extrêmement favorable au développement d'outils personnalisés. Son API Python complète permet d'accéder à quasiment toutes les strates du logiciel, de l'interface utilisateur au moteur de rendu.

### 3.1 Architecture Technique et Moteurs de Rendu

Le choix du moteur de rendu interne est crucial pour la précision de la simulation.

#### 3.1.1 Cycles : La Précision Photométrique (Ground Truth)

Cycles est un moteur de rendu par tracé de chemin (Path Tracer) physiquement basé. Il simule le comportement réel des photons, incluant les rebonds multiples (Global Illumination), la réfraction, et la dispersion sous-surfacique (SSS).8

* **Avantages pour la Simulation Studio :** Cycles gère nativement la décroissance de la lumière (Inverse Square Law) de manière physiquement exacte. La simulation de la translucidité des matériaux de softbox (le tissu diffuseur) est très précise, permettant de visualiser les "hotspots" réels.
* **Performance :** Bien que traditionnellement plus lent que le temps réel, l'introduction de cycles X et du débruitage (OptiX, OIDN) permet désormais une prévisualisation quasi-instantanée dans le viewport sur des GPU modernes, rendant l'expérience interactive viable.

#### 3.1.2 Eevee : L'Approche Temps Réel

Eevee (et sa nouvelle version Eevee Next) est un moteur de rasterisation qui utilise des astuces pour approximer l'éclairage global.8

* **Avantages :** Rendu immédiat, très faible latence, idéal pour le positionnement rapide des lumières sur des machines moins puissantes.
* **Limites :** La gestion des ombres douces et de la lumière indirecte nécessite souvent des "Irradiance Volumes" ou des réglages de biais d'ombre complexes, ce qui peut nuire à la précision requise pour une simulation technique rigoureuse. Cycles reste préférable pour la validation finale.

### 3.2 L'Écosystème Existant : Plugins et Addons

L'analyse des plugins existants démontre la vitalité de l'écosystème Blender et fournit des briques technologiques réutilisables ou inspirantes.

#### 3.2.1 Photographer 5 (L'étalon-or actuel)

L'addon *Photographer 5* développé par Chafouin est la preuve de concept la plus aboutie de la faisabilité d'une partie des fonctions demandées.9

* **Fonctions Couvertes :** Il implémente une caméra physique complète (ISO, Shutter, Aperture) qui contrôle l'exposition du rendu. Il ajoute des unités photométriques (Lumens, Candelas) aux lumières Blender et gère la température de couleur (Kelvin). Il propose également un "Light Mixer" pour gérer les intensités sans sélectionner les objets 3D.10
* **Limites par rapport à la demande :** *Photographer 5* est un outil pour artistes 3D, pas pour photographes de plateau. Il manque la bibliothèque visuelle de modeleurs "Brand Name" (Profoto, etc.) et surtout, il ne propose **aucune fonctionnalité de génération de plan d'éclairage 2D (Lighting Diagram)**, ce qui est le cœur de la demande utilisateur.

#### 3.2.2 Autres Outils Pertinents

* **Gaffer / Pro-Lighting Studio :** Ces addons se concentrent sur l'éclairage basé sur l'image (HDRI) et le placement rapide, mais manquent de la rigueur technique pour la simulation de studio physique précis.11
* **Gobos Light Textures :** Des bibliothèques existent pour projeter des textures, validant la possibilité d'utiliser des masques pour simuler des modeleurs complexes.5

### 3.3 Faisabilité Technique des Fonctionnalités Manquantes sur Blender

#### 3.3.1 Le Défi de l'Export de Plan 2D (Lighting Diagram)

C'est le point critique. Blender est un outil 3D, et générer un plan vectoriel 2D demande une approche spécifique.

* **Approche 1 : Freestyle / Grease Pencil Line Art.** Blender possède un moteur de rendu non-photoréaliste (NPR) capable de générer des lignes à partir de la géométrie 3D.12
  + *Analyse :* Bien que puissant, Freestyle génère des contours basés sur la géométrie réelle. Or, un plan technique utilise des **symboles** abstraits (un carré barré pour une softbox), pas le dessin du maillage 3D complexe. Cette approche nécessiterait de remplacer les objets 3D par des objets "Proxy" simplifiés au moment du rendu, ce qui est lourd à gérer.
* **Approche 2 : Génération Vectorielle via Python (Scripting).** C'est l'approche recommandée. L'API Python de Blender permet d'itérer sur tous les objets de la scène.
  + *Mécanisme :* Le script parcourt bpy.context.scene.objects. Pour chaque objet identifié comme une lumière studio, il extrait ses coordonnées X et Y, sa rotation Z, et ses métadonnées (type, puissance).
  + *Rendu :* Ces données sont envoyées à une librairie Python tierce intégrée à l'addon (comme fpdf2 ou cairo) pour dessiner des formes vectorielles (SVG) sur un canevas 2D. Cette méthode est indépendante du moteur de rendu, extrêmement rapide (quelques millisecondes), et produit des fichiers PDF professionnels parfaitement nets.7

#### 3.3.2 L'Interface Drag & Drop

Depuis la version 3.0, Blender intègre un **Asset Browser** natif supportant le Drag & Drop.15

* **Adaptation :** L'API permet de définir ce qui se passe lorsqu'un asset est lâché. On peut scripter un "Drop Handler" qui, lorsqu'on lâche une grille nid d'abeille sur une softbox, la parente automatiquement et ajuste les propriétés de diffusion de la lumière. Cette fonctionnalité est native et ne demande pas de hack complexe de l'UI.17

## 4. Analyse Approfondie de la Plateforme B : Unreal Engine 5

Unreal Engine 5 (UE5) représente l'état de l'art du rendu temps réel, utilisé massivement dans le jeu vidéo et la production virtuelle (The Mandalorian). Son architecture est fondamentalement différente de celle de Blender.

### 4.1 Architecture Technique et Moteurs de Rendu

#### 4.1.1 Lumen : L'Illumination Globale Dynamique

Lumen est le système de GI (Global Illumination) temps réel de UE5. Il calcule les rebonds de lumière diffuse et les réflexions spéculaires en temps réel, sans nécessiter de pré-calcul (baking) de "Lightmaps".3

* **Avantage Majeur :** La réactivité. Déplacer un réflecteur blanc dans une scène UE5 change instantanément l'éclairage indirect sur le visage du modèle avec une fidélité impressionnante. C'est l'expérience utilisateur la plus proche de la réalité physique.
* **Précision :** Bien que très performant, Lumen reste une approximation par rapport au Path Tracing pur, surtout pour des effets subtils comme la diffraction ou les caustiques complexes, bien que le moteur intègre aussi un Path Tracer hardware pour les rendus finaux.19

### 4.2 L'Écosystème Existant : Plugins et Concurrents

L'analyse de l'Unreal Marketplace révèle des solutions existantes qui valident la pertinence de la plateforme.

#### 4.2.1 MetaShoot (Le Concurrent Direct)

*MetaShoot* est un plugin pour UE5 qui transforme le moteur en studio photo virtuel.20

* **Fonctions :** Il propose une interface simplifiée ("One Click Render"), des presets d'éclairage, et des assets interactifs (trépieds, softboxes). Il utilise le Ray Tracing pour des réflexions réalistes.
* **Limites identifiées :** L'analyse des fonctionnalités 20 montre que *MetaShoot* se concentre sur le rendu d'images "Beauty" et la vidéo 360. Il ne semble pas proposer de **génération de plan technique 2D (Lighting Diagram)** automatisé. De plus, l'interface reste celle d'Unreal (Content Browser, Details Panel), qui peut être intimidante pour un photographe non-initié au développement de jeu.

#### 4.2.2 Cine Tracer

Bien que n'étant pas un plugin mais un logiciel autonome ("stand-alone") construit sur Unreal Engine, *Cine Tracer* est la référence en prévisualisation cinéma.22 Il intègre des caméras réelles et des lumières techniques. Son succès prouve que le moteur Unreal est capable de gérer ces calculs, mais le fait qu'il soit une application fermée souligne la difficulté de distribuer de tels outils sous forme de simple plugin modifiable.

### 4.3 Faisabilité Technique des Fonctionnalités Manquantes sur UE5

#### 4.3.1 Le Défi de l'Interface Utilisateur (Slate / UMG)

Développer une interface utilisateur personnalisée complexe (comme un planificateur studio) dans l'éditeur d'Unreal est notoirement difficile.

* **Editor Utility Widgets :** Ces widgets permettent de créer des panneaux personnalisés en utilisant le système UMG (Unreal Motion Graphics).23 C'est puissant, mais l'interaction avec le viewport 3D (ex: Drag & Drop depuis un widget vers la scène 3D) est limitée et nécessite souvent du code C++ pour contourner les limitations du système de script visuel (Blueprints).25
* **Complexité du Drag & Drop :** Implémenter un système où l'on glisse une lumière depuis une bibliothèque UI vers la scène pour qu'elle s'aligne automatiquement ("snap") sur le sol ou sur un rail demande une logique de Raycasting complexe à implémenter en C++ pour être fluide dans l'éditeur.26

#### 4.3.2 Le Problème du Plan 2D (SceneCapture2D)

Générer un plan technique vectoriel est le point faible majeur d'Unreal Engine. Le moteur est conçu pour pousser des pixels à 60 images par seconde, pas pour générer des fichiers PDF vectoriels.

* **La Solution "Pixel" (SceneCapture2D) :** La méthode standard consiste à placer une caméra orthographique au-dessus de la scène qui rend vers une texture (RenderTarget).27
  + *Limitations :* Cela produit une image raster (pixels), pas un plan vectoriel. Si l'on veut remplacer les objets 3D réalistes par des icônes symboliques, il faut utiliser des astuces de rendu complexes : assigner des tags "OwnerOnlySee" à des Sprites 2D qui ne sont visibles que par la caméra du plan, tout en cachant les meshs 3D.29 C'est une solution lourde ("hacky") qui manque de la netteté et de la précision d'un export vectoriel.
* **Absence de Librairie Vectorielle Native :** Contrairement à Python sur Blender qui a accès à tout l'écosystème PyPI (reportlab, fpdf), Unreal n'a pas de librairie native facile d'accès pour écrire du SVG ou du PDF depuis des Blueprints. Cela nécessiterait l'intégration de librairies C++ tierces, augmentant considérablement la complexité de développement et de maintenance.

## 5. Matrice de Décision et Recommandation Stratégique

L'analyse comparative met en lumière une divergence claire entre la puissance brute et la flexibilité de développement.

| **Critère d'Évaluation** | **Blender (Python / Cycles)** | **Unreal Engine 5 (C++ / Lumen)** |
| --- | --- | --- |
| **Précision Photométrique** | **Excellente** (Cycles est Ground Truth). | Très Bonne (Lumen/Path Tracer), mais focus temps réel. |
| **Qualité du Rendu Temps Réel** | Bonne (Eevee Next progresse), mais inférieure à UE5. | **Excellente** (Lumen est le leader du marché). |
| **Facilité de Développement UI** | **Haute** (Python API + Addons existants). | Moyenne/Faible (Slate/UMG complexe, C++ souvent requis). |
| **Génération Plan 2D (Diagram)** | **Très Faisable** (Scripting Python vers SVG/PDF). | Difficile (Solution pixelisée via Render Targets, pas de vectoriel natif). |
| **Distribution & Installation** | **Simple** (Fichier Zip léger, logiciel portable). | Complexe (Launcher Epic, compilation de shaders, poids > 50Go). |
| **Cible Utilisateur** | Indépendants, PME, Photographes technophiles. | Studios de Prod Virtuelle, Cinéma, Équipes dédiées. |
| **Modèle Économique** | Vente d'Addon (GPL), marché actif (BlenderMarket). | Marketplace Epic, Royalties possibles au-delà d'1M$ CA. |

### 5.1 Recommandation : Blender comme Plateforme Cible

Au regard de la demande spécifique — **"avoir les fonctions de set.a.light 3D"** — et en particulier la fonction critique de génération de plans d'éclairage (diagrammes), **Blender apparaît comme la plateforme la plus adaptée et la plus pragmatique.**

**Justification Stratégique :**

1. **L'avantage "Python" pour le Plan 2D :** La capacité de Blender à manipuler des données abstraites via Python permet de générer des plans techniques vectoriels (PDF) de qualité professionnelle, une fonctionnalité centrale de *set.a.light 3D* qui serait extrêmement pénible à reproduire sous UE5.
2. **Barrière à l'Entrée :** Le public cible (photographes) est plus susceptible d'installer Blender (200 Mo) sur un laptop standard que de télécharger l'écosystème Unreal (50 Go+) nécessitant une carte graphique de dernière génération pour profiter de Lumen.
3. **Rentabilité du Développement :** Le temps de développement pour un MVP (Minimum Viable Product) sous Blender est estimé à 30-40% inférieur à celui sous Unreal, grâce à l'agilité de Python et à l'absence de temps de compilation.

## 6. Plan de Développement Détaillé (Roadmap pour Blender)

Ce plan de développement structure la création du plugin en quatre phases distinctes, visant un produit commercialisable en 6 à 8 mois.

### Phase 1 : Le Noyau Physique et Photométrique (Mois 1-2)

Cette phase vise à transformer Blender en simulateur physique fiable.

* **1.1 Conversion des Unités (Le Moteur Photométrique) :**
  + Blender utilise les Watts (Radiométrie) pour la puissance des lumières. Il faut implémenter une conversion vers les unités photométriques (Lumens, Lux).
  + *Implémentation :* Créer un script Python qui convertit les entrées utilisateur (ex: "Flash 500Ws") en valeurs compatibles Cycles, en tenant compte de l'efficacité lumineuse des sources (Luminous Efficacy). Pour un flash xénon, on considérera une efficacité d'environ 35-50 lumens/watt, contre 80-100 lm/W pour des LEDs modernes.10
* **1.2 Shader Universel de Lumière :**
  + Développer un "Node Group" maître appliqué à toutes les lumières du plugin.
  + *Paramètres exposés :* Température (Kelvin) via un noeud Blackbody, Tint (Vert/Magenta), Puissance (EV ou % de puissance).
  + *Texture IES :* Intégrer un slot pour charger dynamiquement des profils IES réels, garantissant que la distribution spatiale de la lumière est correcte.

### Phase 2 : Bibliothèque d'Assets et UX "Studio" (Mois 3-4)

Cette phase se concentre sur l'ergonomie (cacher la complexité).

* **2.1 Modélisation des Équipements (Digital Twin) :**
  + Créer une bibliothèque d'assets 3D optimisés (Low Poly pour la fluidité) représentant les standards du marché : Têtes Profoto/Broncolor/Godox, pieds C-Stands, girafes.
  + *Les Modeleurs (Modifiers) :* Modéliser les softboxes non pas comme de simples cubes, mais avec des propriétés de matériaux translucides pour Cycles (Transmission + Subsurface Scattering) afin de simuler la diffusion réelle de la toile.
* **2.2 Intégration Asset Browser & Drag-and-Drop :**
  + Utiliser l'API bpy.types.FileHandler ou les hooks de l'Asset Browser pour détecter le glisser-déposer.
  + *Logique d'Attachement (Snapping) :* Développer un opérateur modal. Lorsqu'un modeleur (ex: Octabox) est dragué près d'une tête de flash, il doit s'y "snapper" magnétiquement et devenir son enfant hiérarchique (Parenting), héritant de son orientation.
* **2.3 Interface de Contrôle (Le "Light Mixer") :**
  + Créer un panneau dans la barre latérale (N-Panel) de la vue 3D.
  + Ce panneau doit lister toutes les lumières de la scène sous forme de tableau de bord, permettant de "Solo", "Mute" ou changer la puissance de chaque source sans avoir à naviguer dans la scène 3D pour cliquer sur l'objet.31

### Phase 3 : Le Module "Diagramme Technique 2D" (Mois 5-6)

C'est la fonctionnalité critique qui nécessite le plus de développement Python pur.

* **3.1 Bibliothèque de Symboles Vectoriels :**
  + Dessiner en SVG une collection d'icônes 2D standardisées (vue de dessus) correspondant à chaque asset 3D (ex: un cercle avec des rayons pour une Octabox).
  + Stocker ces données soit sous forme de fichiers SVG externes, soit encodées en chaînes de caractères dans le code Python pour un plugin monolithique.
* **3.2 Algorithme d'Extraction et de Mapping :**
  + Écrire un script qui parcourt la scène (scene.objects).
  + *Projection :* Récupérer les coordonnées mondiales (X, Y) et la rotation (Z) de chaque objet pertinent. Ignorer l'axe Z (hauteur) pour le dessin, mais le stocker pour l'annotation textuelle.
  + *Mapping d'échelle :* Convertir les unités Blender (mètres) en unités de page (millimètres sur un A4), en définissant une échelle (ex: 1:50 ou 1:100).
* **3.3 Génération du Document (Backend PDF) :**
  + Utiliser une librairie comme reportlab ou écrire un générateur XML SVG simple.
  + Placer les symboles aux coordonnées calculées.
  + Ajouter des flèches directionnelles indiquant l'orientation des lumières.
  + Générer automatiquement une légende (Gear List) en bas de page listant tout le matériel présent dans la scène.

### Phase 4 : La Caméra Virtuelle et l'Exposition (Mois 7)

* **4.1 Script d'Exposition Automatique :**
  + Lier les propriétés d'exposition du Color Management de Blender (bpy.context.scene.view\_settings.exposure) aux variables custom de la caméra virtuelle (ISO, Aperture, Shutter).
  + *Calcul :* Le script doit recalculer l'exposition du viewport à chaque changement de paramètre pour simuler l'assombrissement/éclaircissement de l'image.
* **4.2 Simulation des Objectifs :**
  + Implémenter des presets de focales fixes (35mm, 50mm, 85mm, 105mm Macro) qui ajustent le champ de vision (FoV) de la caméra Blender.
  + Ajouter une option "Sensor Size" (Full Frame vs APS-C) pour appliquer le facteur de recadrage (Crop Factor) correct.

## 7. Étude de Faisabilité Technique et Points de Vigilance

### 7.1 Le Défi des Performances (Cycles vs Eevee)

Le réalisme de *set.a.light 3D* est bon, mais Cycles est *meilleur*. Le risque est la performance.

* *Solution :* Utiliser Eevee pour la phase de mise en place (positionnement, cadrage) pour garantir 60 FPS. Utiliser un bouton "Shoot" qui déclenche un rendu Cycles progressif dans le viewport pour valider la lumière fine (ombres douces, diffusion). Avec les GPU modernes et le Denoising, cette transition est quasi transparente.

### 7.2 L'Ergonomie "Non-Blender"

Blender est complexe (raccourcis clavier, interface dense). Le succès du plugin dépendra de sa capacité à **masquer** cette complexité.

* *Solution :* Utiliser des "Workspaces" personnalisés qui ferment les panneaux inutiles (Timeline, Outliner complexe) et ne montrent que la vue 3D et le panneau de contrôle du plugin. L'utilisateur ne doit jamais avoir à toucher au Shader Editor nodal pour régler une température de couleur.

### 7.3 La Gestion des Mises à Jour Blender

Blender évolue vite (3.6 -> 4.0 -> 4.1). L'API Python subit parfois des changements de rupture (Breaking Changes).

* *Stratégie :* Adopter une architecture de code modulaire et suivre les versions LTS (Long Term Support) de Blender pour garantir la stabilité du plugin sur le long terme pour les clients professionnels.

## 8. Conclusion

Le projet de porter les fonctionnalités de *set.a.light 3D* vers une plateforme généraliste est non seulement techniquement réalisable, mais répond à un réel besoin du marché : celui d'avoir un outil de planification studio qui bénéficie de la puissance de rendu "Next Gen".

L'analyse comparative désigne **Blender** comme la plateforme la plus viable. Son architecture ouverte, la puissance de son API Python pour la génération de documents techniques (plans 2D), et la qualité photométrique de son moteur Cycles offrent le terreau idéal pour ce développement. Bien qu'Unreal Engine 5 soit supérieur en rendu temps réel pur, sa rigidité en matière d'interface éditeur et de génération vectorielle en fait un choix plus risqué et coûteux pour ce cas d'usage spécifique orienté "Planification et Diagramme".

Le plan de développement proposé permet une montée en charge progressive, sécurisant d'abord le cœur physique avant de développer les couches d'abstraction ergonomique qui feront le succès du produit final.

#### Sources des citations

1. set.a.light 3D V3 - elixxier Software, consulté le décembre 11, 2025, <https://www.elixxier.com/en/set-a-light-3d/>
2. set.a.light 3D V2.0 | User Manual - elixxier Software, consulté le décembre 11, 2025, <https://www.elixxier.com/wp-content/uploads/2022/09/set_a_light_3d_v2_5_manual_v1.07_en.pdf>
3. 3D Lighting: Techniques, Applications, and Career Paths in Digital Art, consulté le décembre 11, 2025, <https://cada-edu.com/guides/3d-lighting>
4. Experiences with Set.a Light 3D?? : r/photography - Reddit, consulté le décembre 11, 2025, <https://www.reddit.com/r/photography/comments/1bpirav/experiences_with_seta_light_3d/>
5. Blender Addon for Lighting & Rendering l Gobos Light Textures - YouTube, consulté le décembre 11, 2025, <https://www.youtube.com/watch?v=VwJuWQTnYII>
6. set.a.light 3D Review - YouTube, consulté le décembre 11, 2025, <https://www.youtube.com/watch?v=Eb42ie5n_x8>
7. Lighting: From Concept to Render – Blog - Blender Studio, consulté le décembre 11, 2025, <https://studio.blender.org/blog/wing-it-from-concept-to-render/>
8. Unreal Engine vs Blender: which one is better to choose? | - iRender, consulté le décembre 11, 2025, <https://irendering.net/unreal-engine-vs-blender-which-one-is-better-to-choose/>
9. Photographer - Camera Exposure, White Balance, Autofocus, Physical Lights, Bokeh, Render Queue - Released Scripts and Themes - Blender Artists, consulté le décembre 11, 2025, <https://blenderartists.org/t/photographer-camera-exposure-white-balance-autofocus-physical-lights-bokeh-render-queue/1101721>
10. Photographer 5 - Blender Add-on - chafouin, consulté le décembre 11, 2025, <https://chafouin.gumroad.com/l/photographer5>
11. The Best Addons For Lighting In Blender - Gachoki Studios, consulté le décembre 11, 2025, <https://gachoki.com/addons-for-lighting-in-blender/>
12. Freestyle Lines tools make for interesting technical illustrations : r/blender - Reddit, consulté le décembre 11, 2025, <https://www.reddit.com/r/blender/comments/1oz0ajd/freestyle_lines_tools_make_for_interesting/>
13. Freestyle data and US Patent Application Drawings - Blender Artists, consulté le décembre 11, 2025, <https://blenderartists.org/t/freestyle-data-and-us-patent-application-drawings/1517330>
14. Tutorial: Rendering 2D Icons as 3D Objects in Blender - YouTube, consulté le décembre 11, 2025, <https://www.youtube.com/watch?v=zthvZvw-yJE>
15. Asset Browser - Blender 5.0 Manual, consulté le décembre 11, 2025, <https://docs.blender.org/manual/en/latest/editors/asset_browser.html>
16. Create Your Own Asset Libraries in Blender! (Tutorial) - YouTube, consulté le décembre 11, 2025, <https://www.youtube.com/watch?v=jpz13q52YWM>
17. Get if a drag and drop was done from the asset browser in python?, consulté le décembre 11, 2025, <https://blender.stackexchange.com/questions/289152/get-if-a-drag-and-drop-was-done-from-the-asset-browser-in-python>
18. Python API for Drag and Drop Events? - Blender Stack Exchange, consulté le décembre 11, 2025, <https://blender.stackexchange.com/questions/317234/python-api-for-drag-and-drop-events>
19. Designing Visuals, Rendering, and Graphics with Unreal Engine - Epic Games Developers, consulté le décembre 11, 2025, <https://dev.epicgames.com/documentation/en-us/unreal-engine/designing-visuals-rendering-and-graphics-with-unreal-engine>
20. MetaShoot - Becoming Atlux - atlux.ai in Code Plugins - UE ..., consulté le décembre 11, 2025, <https://www.unrealengine.com/marketplace/en-US/product/metashoot>
21. MetaShoo for Unreal Engine offers simplified lighting and rendering - CGPress, consulté le décembre 11, 2025, <https://cgpress.org/archives/metashoo-for-unreal-engine-offers-simplified-lighting-and-rendering.html>
22. Used 3D software to plan every shot for a short film—here's how the pre-viz compares to the final frames : r/cinematography - Reddit, consulté le décembre 11, 2025, <https://www.reddit.com/r/cinematography/comments/1gvr8gn/used_3d_software_to_plan_every_shot_for_a_short/>
23. Creating Drag and Drop UI in Unreal Engine - Epic Games Developers, consulté le décembre 11, 2025, <https://dev.epicgames.com/documentation/en-us/unreal-engine/creating-drag-and-drop-ui-in-unreal-engine>
24. Making Editor Utility Widgets for our game in Unreal Engine 4 - YouTube, consulté le décembre 11, 2025, <https://www.youtube.com/watch?v=u8c4Q2ve3AQ>
25. Drag and drop elements in scrollbox in widget editor utility - Unreal Engine Forums, consulté le décembre 11, 2025, <https://forums.unrealengine.com/t/drag-and-drop-elements-in-scrollbox-in-widget-editor-utility/2455675>
26. Request 5 part 1 : How To Drag and Drop 3d Actor from widget to local world, consulté le décembre 11, 2025, <https://dev.epicgames.com/community/learning/tutorials/jV8/request-5-part-1-how-to-drag-and-drop-3d-actor-from-widget-to-local-world>
27. Creating an outlined minimap from SceneCapture : r/unrealengine - Reddit, consulté le décembre 11, 2025, <https://www.reddit.com/r/unrealengine/comments/19dlvnh/creating_an_outlined_minimap_from_scenecapture/>
28. Creating a Minimap, and using Hidden and Show Scene Capture Flags, with Unreal Engine 5.2 - YouTube, consulté le décembre 11, 2025, <https://www.youtube.com/watch?v=4clmzyF05W4>
29. Show Specific Actors In Scene Capture Component 2d - Rendering - Unreal Engine Forum, consulté le décembre 11, 2025, <https://forums.unrealengine.com/t/show-specific-actors-in-scene-capture-component-2d/622274>
30. Rendering Specific Actors with Render Targets and Scene Captures - Unreal Engine, consulté le décembre 11, 2025, <https://www.youtube.com/watch?v=xzLLPPR-VE0>
31. Blender Light Manager - Light list - Released Scripts and Themes, consulté le décembre 11, 2025, <https://blenderartists.org/t/blender-light-manager-light-list/1272197>