# Analyse Architecturale de la Bibliothèque Mathsteps : Structures de Sortie, Sérialisation JSON et Taxonomie des Types de Changement

## 1. Introduction à la Pédagogie Computationnelle et aux Solveurs "Boîte Blanche"

L'évolution des technologies éducatives (EdTech) a marqué une transition fondamentale, passant de simples calculatrices numériques à des systèmes de tutorat intelligents capables d'expliciter le raisonnement mathématique. Dans ce contexte, la bibliothèque mathsteps, un projet open-source initialement développé par Socratic, incarne cette nouvelle génération de moteurs de résolution. Contrairement aux systèmes de calcul formel (Computer Algebra Systems - CAS) traditionnels tels que Mathematica ou SymPy, qui sont optimisés pour la vitesse d'exécution et la puissance symbolique brute — fonctionnant souvent comme des "boîtes noires" qui ingèrent une équation et recrachent une solution — mathsteps est conçu comme une "boîte blanche". Son architecture interne est délibérément structurée pour optimiser la pédagogie, c'est-à-dire la capacité à décomposer une transformation algébrique complexe en une série d'opérations atomiques, intelligibles pour un apprenant humain.1

Ce rapport propose une analyse technique exhaustive de l'architecture de mathsteps, en se concentrant spécifiquement sur ses mécanismes de sortie. Pour les développeurs cherchant à intégrer cette bibliothèque dans des applications modernes (telles que des applications mobiles sous Flutter ou des plateformes web réactives), il est impératif de maîtriser la structure des objets retournés, les stratégies de sérialisation JSON nécessaires pour le transfert de données, et la taxonomie précise des changeType qui constituent le vocabulaire logique du solveur. Nous examinerons comment la bibliothèque construit un arbre syntaxique abstrait (AST) pour représenter les expressions mathématiques, comment elle navigue récursivement dans cet arbre pour identifier des opportunités de simplification, et comment elle encapsule ces transformations dans des objets d'état (NodeStatus et EquationStatus) riches en métadonnées.1

En outre, nous établirons un parallèle entre l'automatisation algorithmique de mathsteps et les directives de création de contenu manuel utilisées par des leaders de l'industrie comme Photomath. Cette comparaison mettra en lumière la convergence entre les règles de validation algorithmique et les meilleures pratiques pédagogiques humaines, telles que la structure Entrée-Description-Sortie (IDO) et la granularité des étapes.3

## 2. Fondements Théoriques : L'Arbre Syntaxique Abstrait et la Récursivité Pédagogique

Pour comprendre le format de sortie de mathsteps, il est essentiel de disséquer la manière dont la bibliothèque perçoit et manipule les mathématiques. Le cœur du système ne traite pas les équations comme des chaînes de caractères textuelles, mais comme des structures de données hiérarchiques.

### 2.1 La Primauté de l'Arbre Syntaxique Abstrait (AST)

La bibliothèque mathsteps repose entièrement sur les capacités de parsing et de modélisation de mathjs, une bibliothèque mathématique extensive pour JavaScript.2 Lorsqu'une chaîne de caractères telle que 2x + 3x est ingérée par le système, elle est immédiatement convertie en un Arbre Syntaxique Abstrait (AST). Dans cet arbre, les feuilles représentent les opérandes (tels que les constantes 2, 3 ou le symbole x), tandis que les nœuds internes représentent les opérateurs (tels que l'addition + ou la multiplication \* implicite).

Cette représentation est cruciale car elle permet à mathsteps d'appliquer des transformations basées sur la structure plutôt que sur la syntaxe. Par exemple, pour simplifier 2x + 3x, l'algorithme ne cherche pas simplement des caractères adjacents ; il parcourt l'arbre pour identifier un motif spécifique : un nœud opérateur d'addition dont les deux enfants sont des nœuds opérateurs de multiplication, lesquels partagent un enfant symbole identique (x). C'est cette reconnaissance de motif structurel qui déclenche la règle de simplification appropriée.

L'analyse des fichiers sources de la bibliothèque révèle que les objets manipulés sont des instances de classes définies par mathjs, telles que OperatorNode, ConstantNode, SymbolNode, et ParenthesisNode.4 Chaque étape de résolution produite par mathsteps n'est, en réalité, qu'un instantané de cet arbre à deux moments distincts : juste avant l'application d'une règle de transformation (oldNode) et juste après (newNode). Le format de sortie JSON doit donc être capable de capturer ou de représenter cette complexité structurelle.

### 2.2 Le Modèle de Résolution Récursif et Granulaire

La pédagogie mathématique est intrinsèquement récursive. La résolution d'une équation complexe implique souvent la résolution de plusieurs sous-problèmes imbriqués. Par exemple, pour résoudre une équation impliquant des fractions rationnelles, il faut d'abord trouver un dénominateur commun, ce qui implique de multiplier des termes, ce qui peut impliquer de distribuer des coefficients, et ainsi de suite.

L'architecture de sortie de mathsteps reflète fidèlement cette réalité cognitive. Au lieu de fournir une liste linéaire et plate d'étapes, la bibliothèque génère une structure arborescente d'objets d'état. Une étape de haut niveau, telle que "Simplifier le côté gauche de l'équation" (SIMPLIFY\_LEFT\_SIDE), agit comme un conteneur sémantique. Elle ne contient pas seulement le résultat final de la simplification, mais une propriété subSteps qui est elle-même un tableau d'étapes détaillant les micro-opérations effectuées pour réaliser cette simplification majeure.1

Cette granularité est non seulement une caractéristique technique mais une nécessité pédagogique. Elle permet aux interfaces utilisateur (UI) de mettre en œuvre des fonctionnalités d'affichage progressif ("progressive disclosure"), où l'utilisateur peut choisir de voir une vue d'ensemble ou de forer ("drill down") dans les détails d'un calcul spécifique. Cela s'aligne avec les principes de la théorie de la charge cognitive, évitant de submerger l'apprenant avec trop d'informations simultanées, une règle également stipulée dans les directives de création de contenu manuel de Photomath, qui avertissent contre l'encombrement ("crowding") des étapes.3

## 3. Modèles d'Objets Détaillés et Structure de Sortie

L'analyse de la documentation et des dépôts de code révèle que mathsteps utilise deux classes d'objets principales pour encapsuler l'état d'un problème mathématique à un instant *t* : NodeStatus pour les expressions algébriques et EquationStatus pour les équations complètes. La compréhension de ces objets est la clé de voûte pour interpréter et sérialiser la sortie de la bibliothèque.

### 3.1 L'Objet NodeStatus : L'Unité de Changement Expressionnelle

Lorsque la bibliothèque exécute une simplification d'expression via la fonction mathsteps.simplifyExpression(expression), elle retourne un tableau d'objets NodeStatus. Chaque objet représente une transformation discrète appliquée à l'AST.1

#### 3.1.1 Définition des Propriétés Techniques

Chaque instance de NodeStatus expose les propriétés suivantes, essentielles pour la reconstruction de l'interface utilisateur :

| **Propriété** | **Type de Donnée** | **Description Technique et Sémantique** |
| --- | --- | --- |
| **changeType** | String (Enum) | Un identifiant unique et immuable correspondant à la règle algébrique spécifique appliquée (ex: COLLECT\_LIKE\_TERMS). C'est la clé de localisation pour les descriptions textuelles. |
| **oldNode** | Object (MathNode) | L'état de l'arbre syntaxique abstrait (AST) de l'expression *avant* l'application de la règle. Il s'agit d'un objet complexe de mathjs. |
| **newNode** | Object (MathNode) | L'état de l'AST *après* l'application de la règle. |
| **subSteps** | Array<NodeStatus> | Une liste (pouvant être vide) d'objets NodeStatus enfants. Cette propriété matérialise la récursivité du solveur, contenant les étapes intermédiaires nécessaires pour passer de oldNode à newNode. |
| **changeGroup** | Boolean (Optionnel) | Un indicateur (flag) utilisé pour regrouper visuellement les termes affectés. Cela permet, par exemple, de colorer en bleu uniquement les termes  qui sont combinés, respectant ainsi les exigences de coordination des couleurs.1 |

#### 3.1.2 Sérialisation et Représentation JSON

Il est crucial de noter que oldNode et newNode ne sont pas des chaînes de caractères dans l'environnement d'exécution Node.js, mais des objets graphiques cycliques. Une tentative naïve de sérialisation via JSON.stringify() sur l'objet brut échouerait souvent ou produirait un payload excessivement lourd contenant des métadonnées internes inutiles pour le client (comme les pointeurs parents).

Pour une API JSON efficace, ces nœuds doivent être transformés. La méthode recommandée est d'utiliser les fonctions .toString() (pour une représentation ASCII) ou .toTex() (pour une représentation LaTeX) fournies par mathjs avant la sérialisation finale.4

**Structure JSON Canonique pour NodeStatus (Post-Transformation) :**

JSON

{  
 "changeType": "ADD\_POLYNOMIAL\_TERMS",  
 "oldNode": "2x + 3x",  
 "newNode": "5x",  
 "subSteps":  
 },  
 {  
 "changeType": "SIMPLIFY\_ARITHMETIC",  
 "oldNode": "2 + 3",  
 "newNode": "5",  
 "subSteps":  
 }  
 ]  
}

Ce format JSON démontre clairement comment une opération de haut niveau (ADD\_POLYNOMIAL\_TERMS) encapsule la logique de regroupement (COLLECT\_LIKE\_TERMS) suivie de l'arithmétique simple (SIMPLIFY\_ARITHMETIC).

### 3.2 L'Objet EquationStatus : Gestion des Relations d'Égalité

La résolution d'équations introduit une dimension relationnelle (égalité ou inégalité) absente des simples expressions. Pour gérer cela, mathsteps introduit la classe EquationStatus, qui sert de conteneur pour les transformations affectant l'équilibre d'une équation.1

#### 3.2.1 Structure et Propriétés

L'objet EquationStatus partage des similitudes avec NodeStatus mais opère sur des objets Equation plutôt que des nœuds AST simples :

| **Propriété** | **Type de Donnée** | **Description Technique** |
| --- | --- | --- |
| **changeType** | String (Enum) | Identifiant de la règle de résolution d'équation (ex: SUBTRACT\_FROM\_BOTH\_SIDES). |
| **oldEquation** | Object (Equation) | L'objet représentant l'équation complète avant la transformation. |
| **newEquation** | Object (Equation) | L'objet représentant l'équation complète après la transformation. |
| **subSteps** | Array<EquationStatus> | Liste récursive des sous-opérations. |

#### 3.2.2 L'Objet Wrapper Equation

Une particularité technique de mathsteps est l'introduction d'une classe Equation qui n'existe pas nativement dans mathjs.7 Cette classe agit comme un wrapper contenant trois propriétés essentielles :

1. **leftNode** : L'AST représentant l'expression à gauche du comparateur.
2. **rightNode** : L'AST représentant l'expression à droite.
3. **comparator** : Une chaîne de caractères représentant l'opérateur de relation (=, <, <=, >, >=).

Cette distinction architecturale est vitale. Les développeurs ne peuvent pas invoquer des méthodes d'évaluation (evaluate()) directement sur un objet Equation. Ils doivent accéder aux propriétés leftNode ou rightNode pour effectuer toute analyse ou rendu. Lors de la sérialisation en JSON, il est d'usage de concaténer ces parties (ex: leftNode.toTex() + comparator + rightNode.toTex()) ou de conserver la structure objet pour permettre un formatage flexible côté client.

## 4. Taxonomie Exhaustive des changeType : Le Vocabulaire Algorithmique

La propriété changeType est sans doute la donnée la plus critique pour l'expérience utilisateur. Elle transforme une séquence de calculs bruts en une narration pédagogique. En analysant les fichiers sources du dépôt mathsteps (notamment lib/ChangeTypes.js et les divers modules de recherche dans lib/simplifyExpression et lib/checks 5), nous pouvons établir une taxonomie complète de ces énumérations.

Ces types de changements ne sont pas arbitraires ; ils correspondent strictement aux règles algébriques enseignées dans les programmes scolaires, reflétant une conception axée sur la didactique.

### 4.1 Opérations Arithmétiques et Simplifications de Base

Ces types représentent les opérations de "nettoyage" et de calcul numérique direct. Ils apparaissent souvent comme des feuilles dans l'arbre des subSteps.

| **ChangeType (Identifiant JSON)** | **Contexte Mathématique** | **Exemple Illustratif** | **Intention Pédagogique** |
| --- | --- | --- | --- |
| SIMPLIFY\_ARITHMETIC | Calcul Numérique |  | Réduction d'opérandes constants. |
| NO\_CHANGE | Contrôle de Flux | N/A | Indique qu'aucune simplification n'est possible (état terminal). |
| SIMPLIFY\_DOUBLE\_UNARY\_MINUS | Notation |  | Élimination de la redondance des signes. |
| UNARY\_MINUS\_TO\_NEGATIVE\_ONE | Notation Explicite |  | Rend le coefficient implicite (-1) explicite pour faciliter les opérations ultérieures. |
| ADD\_COEFFICIENT\_OF\_ONE | Notation Explicite |  | Rend le coefficient implicite (1) explicite. |
| REMOVE\_ADDITION\_OF\_ZERO | Identité Additive |  | Application de l'élément neutre de l'addition. |
| REMOVE\_MULTIPLICATION\_BY\_ONE | Identité Multiplicative |  | Application de l'élément neutre de la multiplication. |
| REMOVE\_MULTIPLICATION\_BY\_NEGATIVE\_ONE | Notation |  | Retour à la notation standard après calcul. |
| REMOVE\_DIVISION\_BY\_ONE | Simplification Rationnelle |  | Élimination des dénominateurs unitaires. |
| REMOVE\_EXPONENT\_BY\_ONE | Exposants |  | Simplification de la notation exponentielle. |
| REMOVE\_EXPONENT\_BASE\_ONE | Exposants |  | Évaluation de la base unitaire. |
| REDUCE\_EXPONENT\_BY\_ZERO | Exposants |  | Application de la règle de l'exposant nul. |
| REDUCE\_ZERO\_DIVIDED\_BY\_ANYTHING | Zéro |  | Propriété du zéro en numérateur. |
| REDUCE\_MULTIPLICATION\_BY\_ZERO | Zéro |  | Propriété absorbante du zéro. |

### 4.2 Algèbre Polynomiale et Manipulation de Termes

Cette catégorie regroupe les opérations fondamentales de l'algèbre symbolique, cruciales pour la résolution d'équations linéaires et quadratiques.

| **ChangeType (Identifiant JSON)** | **Contexte Mathématique** | **Exemple Illustratif** | **Intention Pédagogique** |
| --- | --- | --- | --- |
| COLLECT\_LIKE\_TERMS | Regroupement |  | Identification visuelle des termes partageant la même variable. |
| ADD\_POLYNOMIAL\_TERMS | Addition |  | Exécution de l'addition sur les coefficients. |
| MULTIPLY\_POLYNOMIAL\_TERMS | Multiplication |  | Multiplication des coefficients et addition des exposants. |
| REARRANGE\_COEFF | Standardisation |  | Mise en forme canonique (coefficient devant la variable). |
| GROUP\_COEFFICIENTS | Factorisation locale |  | Étape intermédiaire montrant l'addition des coefficients. |

### 4.3 Opérations sur les Fractions

Les fractions constituant souvent une barrière cognitive majeure, mathsteps décompose ces opérations avec une granularité fine.

| **ChangeType (Identifiant JSON)** | **Contexte Mathématique** | **Exemple Illustratif** | **Intention Pédagogique** |
| --- | --- | --- | --- |
| ADD\_FRACTIONS | Addition Rationnelle |  | Macro-étape englobant le processus d'addition. |
| COMMON\_DENOMINATOR | Dénominateur Commun |  | Conversion au plus petit dénominateur commun (PPCM). |
| MULTIPLY\_DENOMINATORS | Multiplication |  | Calcul du nouveau dénominateur. |
| MULTIPLY\_NUMERATORS | Multiplication |  | Calcul du nouveau numérateur. |
| DIVIDE\_FRACTION\_FOR\_ADDITION | Séparation |  | Décomposition d'une fraction en somme de fractions. |
| SIMPLIFY\_FRACTION | Réduction |  | Simplification par le PGCD. |
| MULTIPLY\_FRACTIONS | Multiplication |  | Macro-étape pour la multiplication. |
| ADD\_CONSTANT\_AND\_FRACTION | Mixte |  | Conversion d'entier en fraction pour addition. |

### 4.4 Résolution d'Équations

Ces types sont spécifiques à la fonction solveEquation et décrivent les mouvements stratégiques pour isoler la variable.

| **ChangeType (Identifiant JSON)** | **Contexte Mathématique** | **Exemple Illustratif** | **Intention Pédagogique** |
| --- | --- | --- | --- |
| SIMPLIFY\_LEFT\_SIDE | Stratégie Globale |  | Indique une focalisation temporaire sur le membre de gauche. |
| SIMPLIFY\_RIGHT\_SIDE | Stratégie Globale |  | Indique une focalisation temporaire sur le membre de droite. |
| ADD\_TO\_BOTH\_SIDES | Isolation (Inverse) |  | Opération inverse de la soustraction. |
| SUBTRACT\_FROM\_BOTH\_SIDES | Isolation (Inverse) |  | Opération inverse de l'addition. |
| MULTIPLY\_BOTH\_SIDES | Isolation (Inverse) |  | Opération inverse de la division. |
| DIVIDE\_FROM\_BOTH\_SIDES | Isolation (Inverse) |  | Opération inverse de la multiplication. |
| SWAP\_SIDES | Formatage |  | Convention de placer la variable isolée à gauche. |
| STATEMENT\_IS\_TRUE | Validation |  | Détection d'une identité (infinité de solutions). |
| STATEMENT\_IS\_FALSE | Validation |  | Détection d'une contradiction (aucune solution). |

### 4.5 Distribution et Factorisation

Ces types gèrent la modification de la structure des expressions par expansion ou regroupement.

| **ChangeType (Identifiant JSON)** | **Contexte Mathématique** | **Exemple Illustratif** | **Intention Pédagogique** |
| --- | --- | --- | --- |
| DISTRIBUTE | Expansion |  | Application de la distributivité. |
| DISTRIBUTE\_NEGATIVE\_ONE | Expansion |  | Distribution spécifique d'un signe négatif. |
| FACTOR\_POLYNOMIAL | Factorisation |  | Factorisation de polynômes (quadratiques, etc.). |
| FACTOR\_SYMBOL | Factorisation |  | Mise en facteur commun d'une variable. |
| FACTOR\_DIFFERENCE\_OF\_SQUARES | Factorisation Remarquable |  | Identité remarquable . |
| FACTOR\_PERFECT\_SQUARE | Factorisation Remarquable |  | Identité remarquable . |

Cette taxonomie révèle une couverture étendue de l'algèbre de niveau collège et début lycée (Pre-Algebra et Algebra I). L'absence de types liés au calcul différentiel ou intégral dans les fichiers sources analysés confirme que le domaine d'application actuel de la bibliothèque est limité aux mathématiques élémentaires et intermédiaires.2

## 5. Stratégies d'Implémentation JSON et Schémas de Données

Pour un environnement de production, en particulier dans une architecture client-serveur (comme une application mobile Flutter communiquant avec un backend Node.js), la structure des objets en mémoire doit être convertie en un format de transfert robuste.

### 5.1 Schéma JSON Implicite et Recommandé

Bien que mathsteps ne fournisse pas de schéma JSON formel (comme JSON Schema), l'analyse de ses sorties permet de définir un contrat d'interface strict. Voici le schéma JSON recommandé pour standardiser les réponses de l'API :

JSON

{  
 "$schema": "http://json-schema.org/draft-07/schema#",  
 "title": "MathSteps Response",  
 "type": "object",  
 "properties": {  
 "steps": {  
 "type": "array",  
 "items": {  
 "$ref": "#/definitions/Step"  
 }  
 }  
 },  
 "definitions": {  
 "Step": {  
 "type": "object",  
 "properties": {  
 "stepId": {  
 "type": "integer",  
 "description": "Index unique pour l'itération UI."  
 },  
 "changeType": {  
 "type": "string",  
 "enum":,  
 "description": "Clé d'énumération identifiant la règle algébrique."  
 },  
 "oldExpression": {  
 "type": "string",  
 "description": "Représentation LaTeX ou ASCII de l'état N."  
 },  
 "newExpression": {  
 "type": "string",  
 "description": "Représentation LaTeX ou ASCII de l'état N+1."  
 },  
 "subSteps": {  
 "type": "array",  
 "items": {  
 "$ref": "#/definitions/Step"  
 },  
 "description": "Liste récursive des sous-étapes."  
 },  
 "changeGroup": {  
 "type": "boolean",  
 "description": "Indicateur pour le regroupement visuel."  
 }  
 },  
 "required":  
 }  
 }  
}

### 5.2 Logique de Sérialisation (Code Node.js)

Le défi principal réside dans la conversion des objets mathjs en chaînes de caractères avant la sérialisation JSON, car JSON.stringify ne gère pas nativement les structures cycliques de l'AST de mathjs.

Voici un algorithme de sérialisation robuste pour un backend Node.js :

JavaScript

const mathsteps = require('mathsteps');  
  
// Fonction utilitaire pour sérialiser un nœud AST  
function serializeNode(node) {  
 // Utilisation de.toTex() pour une compatibilité maximale avec les rendus UI (ex: KaTeX/MathJax)  
 // Alternative:.toString() pour du texte ASCII brut.  
 return node? node.toTex() : "";   
}  
  
// Fonction récursive pour sérialiser une étape  
function serializeStep(step) {  
 const serialized = {  
 changeType: step.changeType,  
 // Gestion conditionnelle selon si c'est un NodeStatus ou EquationStatus  
 oldExpression: step.oldNode? serializeNode(step.oldNode) : (step.oldEquation? step.oldEquation.ascii() : ""),  
 newExpression: step.newNode? serializeNode(step.newNode) : (step.newEquation? step.newEquation.ascii() : ""),  
 subSteps:  
 };  
  
 if (step.subSteps && step.subSteps.length > 0) {  
 serialized.subSteps = step.subSteps.map(serializeStep);  
 }  
  
 return serialized;  
}  
  
// Exemple d'exécution  
const steps = mathsteps.simplifyExpression('2x + 3x');  
const jsonOutput = steps.map(serializeStep);  
  
console.log(JSON.stringify(jsonOutput, null, 2));

Cette approche garantit que le JSON produit est "propre", exempt de métadonnées internes de bibliothèque, et prêt à être consommé par n'importe quel client frontend.

## 6. Stratégies d'Intégration Mobile et Offline (Focus Flutter)

L'intégration de mathsteps dans un environnement mobile comme Flutter présente un défi architectural majeur : mathsteps est une bibliothèque JavaScript/Node.js, tandis que Flutter utilise le langage Dart. De plus, les exigences modernes imposent souvent un fonctionnement hors ligne (offline-first).

L'analyse des snippets de recherche suggère plusieurs voies d'intégration, allant de l'utilisation de ponts JavaScript à l'intégration de modèles d'IA pour la reconnaissance d'équations.

### 6.1 Exécution du Moteur JS dans Flutter

Puisque mathsteps ne peut pas être transpilé directement en Dart, la solution la plus viable pour une exécution locale (offline) est d'incorporer un moteur JavaScript au sein de l'application Flutter.

* **Solution Technique :** Utiliser le package flutter\_js ou flutter\_qjs.8 Ces packages permettent de lancer une instance de moteur JS (QuickJS ou JavaScriptCore) directement sur l'appareil mobile.
* **Flux de Données :**
  1. L'application Flutter charge le bundle mathsteps.js (qui doit être "bundlé" avec ses dépendances via Webpack ou Rollup pour inclure mathjs).
  2. Flutter envoie l'équation sous forme de chaîne (ex: "2x + 3x = 10") au moteur JS via un canal de message.
  3. Le moteur JS exécute mathsteps.solveEquation(), sérialise le résultat en JSON (selon la logique définie en section 5.2), et renvoie la chaîne JSON à Dart.
  4. Dart désérialise le JSON en objets Dart natifs pour l'affichage.

### 6.2 Rendu Mathématique avec LaTeX

Une fois les étapes reçues (contenant des chaînes LaTeX comme \frac{1}{2}), Flutter doit les afficher. Le rapport identifie flutter\_tex comme la solution standard pour le rendu LaTeX hors ligne.9

* **Architecture de Rendu :** flutter\_tex utilise une WebView locale optimisée injectant MathJax ou KaTeX. Cela garantit que les expressions mathématiques complexes sont rendues avec une fidélité typographique parfaite.
* **Configuration Offline :** Il est impératif de configurer les permissions Android (usesCleartextTraffic) et iOS pour permettre le chargement des ressources HTML/JS locales sans connexion internet.11

### 6.3 Reconnaissance Optique de Caractères (OCR) et IA

Pour une expérience utilisateur complète (similaire à Photomath), l'application doit pouvoir scanner des équations. Les recherches indiquent l'utilisation de modèles TensorFlow Lite (TFLite) ou de bibliothèques spécialisées.

* **Google ML Kit Text Recognition :** Solution robuste pour le texte, mais souvent limitée pour les structures mathématiques complexes (fractions verticales, intégrales).12 Il fonctionne hors ligne.
* **Modèles Spécialisés (Im2Latex) :** Pour une reconnaissance mathématique précise, des modèles de type "Image-to-LaTeX" sont nécessaires. Ces modèles, basés sur des architectures CNN-LSTM ou Transformer (Vision Transformers), peuvent être convertis au format .tflite pour s'exécuter sur mobile.14
* **Intégration TFLite :** Le package tflite\_flutter permet de charger ces modèles .tflite et d'exécuter l'inférence localement. Le flux consiste à : Capturer l'image -> Prétraiter (redimensionnement, normalisation) -> Inférence TFLite -> Décodage des tokens de sortie en chaîne LaTeX -> Envoi à mathsteps pour résolution.14

## 7. Analyse Comparative : Mathsteps vs Standards Industriels (Photomath)

Il est instructif de confronter l'approche algorithmique de mathsteps aux standards de qualité éditoriale humaine, tels que définis dans les directives internes de Photomath.3

| **Critère** | **Approche Mathsteps (Algorithmique)** | **Approche Photomath (Humaine)** | **Convergence / Divergence** |
| --- | --- | --- | --- |
| **Structure de l'Étape** | Objet NodeStatus avec oldNode (Entrée) et newNode (Sortie). | Modèle IDO (Input-Description-Output) strict. | **Convergence forte.** L'architecture de mathsteps automatise le modèle IDO préconisé par les experts humains. |
| **Granularité** | Utilise subSteps pour décomposer les opérations (ex: dénominateur commun avant addition). | Exige "une seule déclaration logique par étape" pour éviter l'encombrement cognitif. | **Convergence.** La récursivité de mathsteps empêche naturellement les sauts logiques trop importants ("crowding"). |
| **Description Textuelle** | Fournit un changeType (énumération technique, ex: ADD\_POLYNOMIAL\_TERMS). | Exige des phrases complètes, impératives, en anglais correct, sans pronoms personnels ("I", "we"). | **Divergence.** mathsteps fournit la logique brute, mais la couche de présentation (le texte) doit être construite par le développeur frontend pour correspondre au ton "neutre et impératif" exigé. |
| **Coordination des Couleurs** | Utilise changeGroup pour marquer les nœuds affectés. | Exige une coloration stricte des éléments correspondants (ex: bleu pour les variables combinées). | **Convergence Technique.** mathsteps fournit les métadonnées (changeGroup) nécessaires pour implémenter la coloration dynamique exigée par les standards visuels. |
| **Contenu Visuel** | N/A (Bibliothèque purement symbolique/algébrique). | Exige des graphiques (GeoGebra) pour la géométrie et les fonctions. | **Divergence Majeure.** mathsteps ne génère pas de graphiques. Une application complète devra coupler mathsteps avec une bibliothèque de traçage comme fl\_chart ou une intégration GeoGebra. |

Cette comparaison valide la pertinence de mathsteps comme moteur de base : son architecture respecte intrinsèquement les principes de découpage et de flux logique qui font la qualité des solutions manuelles, à condition que la couche de présentation (texte et visuel) soit traitée avec soin.

## 8. Conclusion

La bibliothèque mathsteps offre une infrastructure robuste et pédagogiquement saine pour la résolution programmatique de problèmes mathématiques. Son format de sortie, basé sur une hiérarchie d'objets NodeStatus et EquationStatus sérialisables en JSON, fournit toutes les métadonnées nécessaires pour construire des interfaces éducatives riches. La taxonomie des changeType agit comme une carte détaillée de la logique algébrique, permettant une localisation précise des explications.

Pour les développeurs, le succès de l'intégration repose sur trois piliers :

1. **La Sérialisation Intelligente :** Convertir les objets AST complexes en formats portables (LaTeX) avant le transfert JSON.
2. **L'Architecture Hybride :** Utiliser des moteurs JS embarqués pour l'exécution offline sur mobile (Flutter), couplés à des moteurs de rendu LaTeX performants.
3. **L'Enrichissement Sémantique :** Mapper les changeType techniques vers des phrases naturelles et pédagogiques, en respectant les standards de neutralité et de clarté.

Bien que limitée par son approche basée sur des règles (pattern-matching) et son domaine restreint à l'algèbre élémentaire, mathsteps représente une brique technologique essentielle pour démocratiser l'accès à un tutorat mathématique détaillé et interactif, sans les coûts associés à la création manuelle de contenu ou les risques d'hallucination des grands modèles de langage (LLM).

### Annexe : Tableau de Référence Complet des ChangeTypes

Ce tableau agrège les énumérations changeType identifiées à travers l'analyse du code source et de la documentation, classées par domaine mathématique.

| **Domaine** | **ChangeType (Enum String)** | **Description de la Transformation** |
| --- | --- | --- |
| **Arithmétique** | SIMPLIFY\_ARITHMETIC | Évaluation directe (ex: ). |
|  | SIMPLIFY\_DOUBLE\_UNARY\_MINUS | Résolution de signes (ex: ). |
|  | UNARY\_MINUS\_TO\_NEGATIVE\_ONE | Conversion syntaxique (ex: ). |
| **Identités** | REMOVE\_ADDITION\_OF\_ZERO | Simplification additive (ex: ). |
|  | REMOVE\_MULTIPLICATION\_BY\_ONE | Simplification multiplicative (ex: ). |
|  | REMOVE\_DIVISION\_BY\_ONE | Simplification de division (ex: ). |
|  | REDUCE\_EXPONENT\_BY\_ZERO | Règle de puissance (ex: ). |
| **Polynômes** | COLLECT\_LIKE\_TERMS | Identification des termes semblables (ex: ). |
|  | ADD\_POLYNOMIAL\_TERMS | Addition des coefficients (ex: ). |
|  | MULTIPLY\_POLYNOMIAL\_TERMS | Multiplication de termes (ex: ). |
|  | REARRANGE\_COEFF | Mise en forme (ex: ). |
| **Distribution** | DISTRIBUTE | Développement (ex: ). |
|  | DISTRIBUTE\_NEGATIVE\_ONE | Développement de négation (ex: ). |
| **Fractions** | ADD\_FRACTIONS | Étape parente pour l'addition de fractions. |
|  | COMMON\_DENOMINATOR | Recherche et application du PPCM. |
|  | SIMPLIFY\_FRACTION | Réduction par le PGCD. |
|  | MULTIPLY\_FRACTIONS | Étape parente pour la multiplication. |
|  | DIVIDE\_FRACTION\_FOR\_ADDITION | Séparation (ex: ). |
| **Équations** | SIMPLIFY\_LEFT\_SIDE | Conteneur pour les opérations sur le membre de gauche. |
|  | SIMPLIFY\_RIGHT\_SIDE | Conteneur pour les opérations sur le membre de droite. |
|  | ADD\_TO\_BOTH\_SIDES | Opération d'isolation par addition. |
|  | SUBTRACT\_FROM\_BOTH\_SIDES | Opération d'isolation par soustraction. |
|  | MULTIPLY\_BOTH\_SIDES | Opération d'isolation par multiplication. |
|  | DIVIDE\_FROM\_BOTH\_SIDES | Opération d'isolation par division. |
|  | SWAP\_SIDES | Réorganisation visuelle (ex: ). |
|  | STATEMENT\_IS\_TRUE | Identification d'identité (). |
|  | STATEMENT\_IS\_FALSE | Identification de contradiction (). |