

Retours de Claude et propositions d'intégration

Contexte

Analyse rédigée par Claude Code (Sonnet 4.6) le 2026-06-20, à partir d'une session de travail sur le PC Windows (eliob).

Logiciels utilisés par Julien : **QGIS**, **Inkscape**, **GIMP**, **Scribus**. Ces quatre outils forment un workflow cohérent : QGIS pour la donnée géospatiale, Inkscape et Scribus pour la mise en page cartographique et la PAO, GIMP pour le traitement d'image.

Note critique sur la page 227

La page 227 ("Quelles opportunités entre logiciels opensource et les IA") a vraisemblablement été générée par une IA en mode chatbot. Les liens GitHub vers

`modelcontextprotocol/servers/tree/main/src/servers/blender`, `.../qgis`, `.../freecad`, `.../postgis` sont **probablement hallucinés** — ces sous-dossiers n'existent pas dans le dépôt officiel MCP. À vérifier systématiquement avant d'agir sur ces références.

Analyse par logiciel

QGIS — Priorité 1 (le plus mature et le plus pertinent)

Ce qui existe réellement : Le projet `qgis-mcp` (disponible sur GitHub, maintenu par la communauté) expose QGIS comme un serveur MCP local. Il s'installe comme un plugin Python dans QGIS et ouvre un socket local sur lequel Claude peut envoyer des commandes PyQGIS.

Capacités concrètes :

- Charger des couches (Shapefile, GeoJSON, WMS, PostGIS)
- Lancer des algorithmes de traitement (buffer, intersection, statistiques zonales)
- Modifier la symbologie d'une couche
- Exporter une carte en PNG/PDF

Avantage spécifique à Julien : La base PostGIS `alteris_geo` sur le VPS Alteris (79.137.14.202:5432) est déjà accessible. Un workflow QGIS + MCP + PostGIS permettrait à Claude de : interroger la base en SQL spatial → charger le résultat dans QGIS → générer une carte thématique → exporter en PDF, le tout en langage naturel.

Contrainte : QGIS doit être ouvert sur la machine locale. Le MCP communique via socket local (pas de pilotage à distance).

Scribus — Priorité 2 (fort potentiel pour les rendus cartographiques)

Ce qui existe réellement : Scribus expose une API Python complète via son module `scribus` (accessible en mode headless : `scribus --python-script myscript.py`). Pas de serveur MCP publié, mais un **MCP custom est trivial à écrire** : un script Python qui reçoit des commandes MCP et les traduit en appels `scribus.*`.

Capacités concrètes via Python headless :

- Créer un document depuis un gabarit `.sla`
- Injecter du texte dans des cadres de texte nommés
- Placer des images
- Exporter en PDF

Cas d'usage concret pour Julien : Générer automatiquement une fiche de synthèse cartographique (titre, carte exportée depuis QGIS, texte de légende) en combinant la carte produite par QGIS MCP et un gabarit Scribus.

Contrainte : Scribus headless est instable sur certaines versions — à tester avec la version installée. La génération de gabarits `.sla` de référence est un prérequis.

Inkscape — Priorité 3 (manipulation SVG, pas contrôle UI)

Ce qui existe réellement : Pas de serveur MCP dédié pour contrôler l'interface Inkscape. En revanche, deux approches sont réalistes :

1. **Manipulation directe du SVG** — le format natif d'Inkscape est du XML/SVG standard. Claude peut générer ou modifier des fichiers SVG avec Python (`lxml`, `svgwrite`) sans lancer Inkscape.
2. **Inkscape CLI** — `inkscape --actions="verb1;verb2"` permet des opérations batch (export PNG/PDF, conversion de formats) pilotables depuis un MCP.

Cas d'usage concret pour Julien : Modifier programmatiquement une carte SVG exportée depuis QGIS (changer des couleurs, ajouter un texte, insérer un logo) avant intégration dans Scribus.

Contrainte : L'automatisation reste au niveau fichier, pas au niveau interactif. Pour un usage cartographique, QGIS couvre déjà la plupart des besoins de rendu.

GIMP — Priorité 4 (batch uniquement)

Ce qui existe réellement : GIMP peut être piloté en batch via `gimp --no-interface --batch='(script-fu-batch-list ...)'` ou via Python-Fu (`gimp --batch`). Un MCP custom est faisable.

Cas d'usage concret pour Julien : Traitement en masse d'images pour intégration dans des documents Scribus (redimensionnement, recadrage, conversion CMJN, compression). Moins critique si StirlingPDF couvre déjà les besoins de compression.

Contrainte : GIMP batch est lent au démarrage. Pour du traitement image simple, des alternatives Python (`Pillow`, `ImageMagick`) sont plus légères à wrapper en MCP.

Proposition de workflow intégré

PostGIS (Alteris) → QGIS MCP → carte exportée (PNG/PDF)

|



Inkscape CLI (ajustements SVG)

|



Scribus headless (mise en page finale)

|



Claude pilote l'ensemble de la chaîne : de la requête spatiale jusqu'au document final, sans intervention manuelle.

Ordre de priorité pour une mise en œuvre

Priorité	Outil	Action	Effort
1	QGIS	Installer le plugin <code>qgis-mcp</code> , tester avec une couche PostGIS	Faible
2	Scribus	Écrire un MCP custom Python headless, créer un gabarit <code>.sla</code> de référence	Moyen
3	Inkscape	MCP de manipulation SVG via <code>lxml</code> + CLI export	Faible
4	GIMP	MCP batch Python-Fu	Moyen (faible priorité)

Recommandation : commencer par QGIS. C'est le maillon central du workflow de Julien, le MCP existe déjà, et la base PostGIS Alteris est immédiatement exploitable.

Revision #2

Created 2026-06-20 09:08:06 UTC by Julien

Updated 2026-06-20 14:23:32 UTC by Julien