FabNum — Evolutions envisagées

Document de brainstorming — capture les réflexions et décisions prises lors de la phase de conception.

Date : 2026-04-02 **Statut** : En cours de réflexion (quoi/pourquoi, pas encore le comment)

Architecture globale

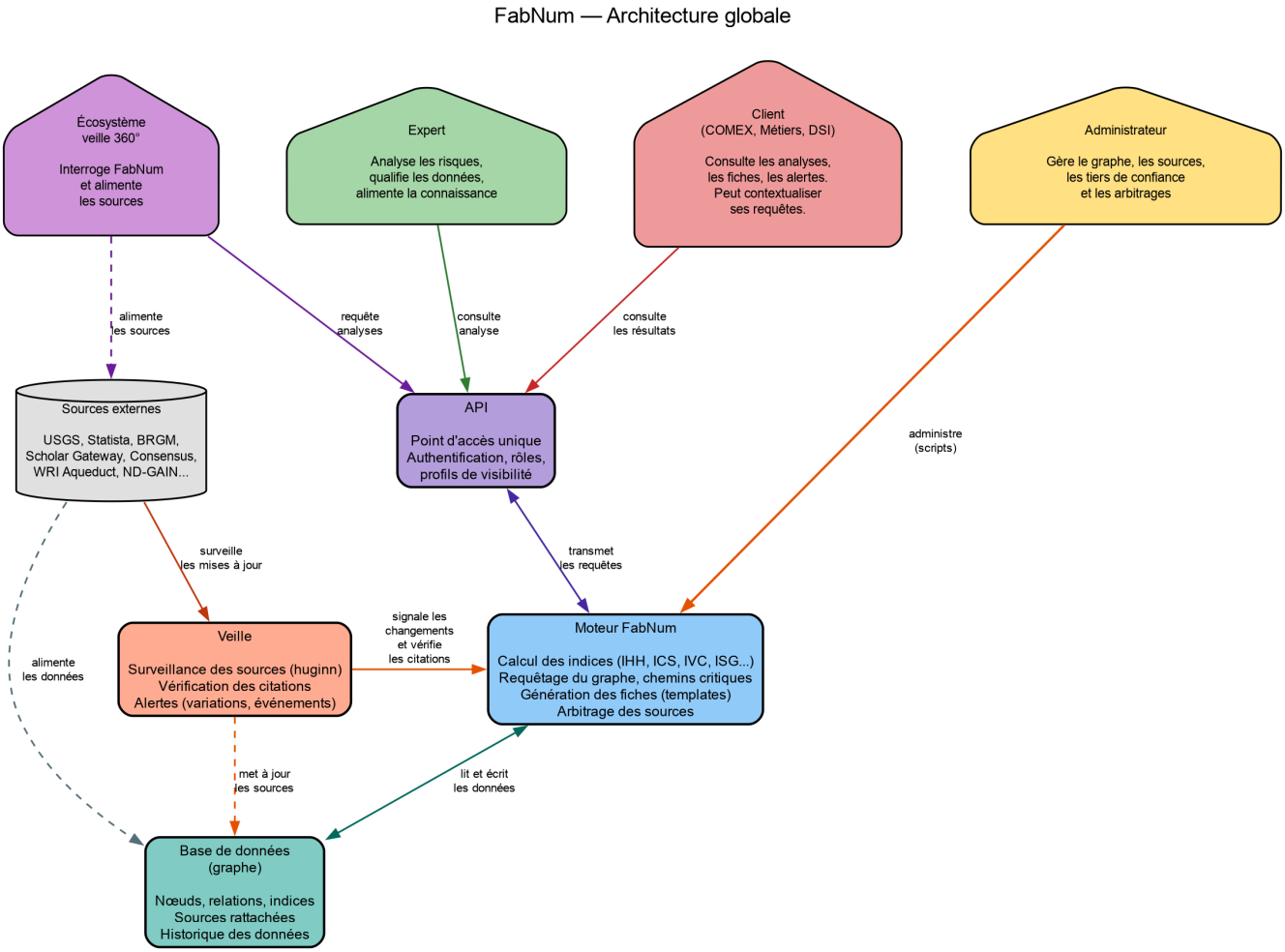

Figure 1: Architecture globale FabNum

Contraintes transversales

- **Budget : zéro en phase de démarrage** — tout est auto-hébergé et auto-financé. Briques technologiques open source et gratuites. Si le projet trouve son marché, cette contrainte sera réévaluée (ex : Neo4j Enterprise si justifié).
- **Vocation commerciale à terme** — FabNum est un outil de recherche aujourd'hui, un produit commercial demain si le marché est présent.
- **Big bang assumé** — peu d'utilisateurs actuels, deux environnements (prod/dev), possibilité de travailler en local. Pas de stratégie de migration progressive nécessaire.
- **Bus factor = 1** — développement assuré par Claude Code avec validation humaine. Qualité maintenue par outillage (ruff, SonarQube, skills quality-gate, linters IDE).
- **Équipe** — Stéphan : vision, arbitrage, validation des spécifications. Claude Code : conception technique, implémentation, agents spécialisés.

- **Intégration écosystème** — FabNum est un composant d'un écosystème plus large de veille stratégique 360° sur la polycrise. Il sera requêté par d'autres outils pour fournir des analyses et rapports.
- **Couplage** — FabNum se couple avec la méthodologie IRON et le livre blanc "Voyage vers la robustesse".
- **Personas cibles** — COMEX (vision stratégique, synthèses), Métiers (analyse opérationnelle), DSI (dépendances SI).
- **Open source** — la question de ce qui sera open source (moteur, graphe des essentiels) reste à trancher.
- **Choix technologiques structurants** — base de données, framework API, frontend expert : à définir rapidement en début d'implémentation.

Point 1 — Recentrage de FabNum : moteur + API + séparation des rôles

Quoi

- **Retrait des modules IA embarqués** (batch_ia/, pgpt/, IA/, page ia_nalyse) : FabNum n'est plus responsable de la génération de rapports IA.
- **Exposition d'une API REST** : les capacités du moteur (graphe, indices de criticité, chemins critiques, filtrage) deviennent des endpoints consommables par des services externes.
- **Séparation en trois rôles** :
  - **Admin** — gestion du graphe, des fiches, de la configuration
  - **Expert** — analyse de criticité, visualisations, plan d'action
  - **Service** — consommation de l'API par des services externes (ex : génération de rapports IA)
- Les rôles sont **assignables par utilisateur** et **cumulables** (un utilisateur peut être admin + expert).

Pourquoi

- FabNum a plus de valeur en tant que **moteur d'analyse + interface experte** qu'en tant qu'application monolithique.
- Séparer le moteur de ses usages permet à d'autres services de consommer les mêmes données/calculs sans couplage.
- Les modules IA actuels sont peu testés (~1500 lignes exclues du linting) et leur évolution ne doit pas être liée à celle du moteur.
- Tous les experts ne seront pas administrateurs à terme : la gestion des droits est nécessaire.

Distinction services internes / externes :

- **Services internes** (côté moteur, non exposés via l'API) — veille sur les sources, mise à jour des données, vérification des affirmations, calcul des indices. Peuvent utiliser de l'IA, du Python, huginn.
- **Services externes** (exposés via l'API) — requêtage, consultation des fiches, analyse. L'API expose les résultats du moteur, pas ses processus internes.

Décisions prises

- **Streamlit abandonné** — l'interface experte sera un frontend consommant l'API (un seul chemin d'accès aux données, pas deux).
- **Administration par scripts** — pas de frontend admin dédié, scripts + accès direct au moteur. Interface admin envisageable plus tard si le besoin se précise.
- **API-first** — architecture à un seul point d'accès pour tous les consommateurs (experts, services externes).
- Le système d'authentification actuel (connexion Gitea simple) devra évoluer pour supporter les rôles.

- Monitoring et observabilité de l'API : à traiter en implémentation, mentionné comme service d'administration.

Point 2 — Gestion rigoureuse des sources de données

Quoi

Les données (chiffres de production, parts de marché, réserves, projections) sont aujourd'hui enfouies dans le texte Markdown des fiches, sans existence structurée indépendante. Trois chantiers pour y remédier :

- **a. Identification et catégorisation des sources** — typer chaque source (rapport périodique, base de données, article académique, etc.), établir un lien explicite source ↔ donnée. Approche académique privilégiée (Scholar Gateway, Consensus).
- **b. Veille sur les sources** — service de surveillance pour détecter les mises à jour des sources (nouvelles éditions de rapports, apparition/disparition de sources). S'appuie sur un service huginn déjà actif pour un sujet connexe.
- **c. Contrôle strict citations/affirmations** — vérifier que chaque affirmation dans les fiches est effectivement contenue dans la ou les sources référencées. S'appuie sur le skill verify-citations existant (à faire évoluer).

Pourquoi

- Les données générées par IA n'ont pas de traçabilité : on ne peut pas vérifier, mettre à jour ou contester une donnée individuellement.
- Les sources sont listées en vrac sans lien vers les données qu'elles justifient (ex : `12-sources.md`).
- Les rapports (USGS, Statista, etc.) sont mis à jour annuellement : sans veille, les données deviennent obsolètes silencieusement.
- La crédibilité de l'outil dépend de la rigueur des sources — approche académique nécessaire.

Décisions prises

- Trois chantiers identifiés (sources, veille, contrôle) ; le modèle de données structuré et la génération des fiches seront traités dans des points séparés.
- Réutilisation de l'infrastructure huginn existante pour la veille.
- Évolution du skill verify-citations pour le contrôle des affirmations.

Point 3 — Modèle de données structuré

Quoi

Définir les spécifications de toutes les données nécessaires aux fiches et au moteur, organisées en deux niveaux :

- **Données essentielles** — celles qui alimentent les indices et le moteur d'analyse (productions, parts de marché, réserves, etc.)
- **Données informationnelles** — contexte, descriptions, projections — utiles aux fiches mais pas directement au calcul

Chaque donnée doit porter :

- Une **définition précise du périmètre mesuré** (ex : "extraction minière de lithium, hors raffinage, en tonnes de Li contenu")
- Des **liens explicites vers ses sources**, avec gestion de la multiplicité
- Les **dimensions de divergence** entre sources : valeur, temporalité, périmètre mesuré

Le modèle doit inclure un **système de tiers de confiance** pour les sources :

- Chaque source a un **tier** (niveau de fiabilité) assorti d'une **pondération** pour les arbitrages
- Le tier peut **varier selon le domaine** couvert (ex : une source tier 1 sur l'extraction peut être tier 2 sur le raffinage)

Des **scénarios d'arbitrage** doivent être spécifiés pour trancher quand les sources divergent :

- **Règle de base** : le tier 1 l'emporte par défaut.
- **Exception** : si N sources de tier inférieur convergent vers une autre valeur → alerte pour arbitrage humain.
- **Seuil de variation** : si une valeur évolue trop fortement entre deux mises à jour → alerte.
- **Principe directeur** : on travaille avec des estimations, pas des mesures exactes. La précision relative suffit.

Un **indice de fiabilité de la donnée** agrège plusieurs facteurs : granularité géographique disponible, tier de confiance de la source, ancienneté de la donnée, couverture du périmètre. Il permet à l'utilisateur de savoir à quel point il peut se fier à un résultat.

Historisation — chaque donnée, chaque valeur d'indice, chaque état du graphe est horodaté et conservé. Nécessaire pour la détection de tendances, les alertes de variation, et la rétro-analyse (backtesting des indices sur des crises passées : Spruce Pine, quotas chinois, etc.).

Les spécifications servent aussi de **cahier des charges pour la recherche de sources** : si une donnée essentielle n'a pas de source, on sait précisément ce qu'on cherche.

Pourquoi

- Sans définition précise du périmètre, l'arbitrage entre sources divergentes n'a pas de sens.
- La multiplicité des sources est la norme, pas l'exception — le modèle doit la gérer nativement.
- Les spécifications permettent d'identifier les trous (données sans source) et de piloter la recherche.

Contraintes de conception

- **Système ouvert et modulable** — de nouveaux indices sont prévus (eau, énergie, logistique). Le modèle ne doit pas câbler les 4 indices actuels (IHH, ICS, IVC, ISG) en dur.
- Le tier de confiance par domaine est une hypothèse à valider, mais le modèle doit pouvoir le supporter.

Décisions prises

- Deux niveaux de données : essentielles (moteur) et informationnelles (fiches).
- Tier de confiance par source, variable selon le domaine, avec pondération pour l'arbitrage.
- Scénarios d'arbitrage automatisables (tier 1 par défaut, convergence → alerte).
- Indice de fiabilité par donnée (agrégation multi-facteurs).
- Historisation native de toutes les données et indices.
- **Gouvernance** : un seul rôle admin couvre toute la gouvernance (tiers, arbitrage, sources). Pas de segmentation tant qu'il n'y a pas d'équipe pour le justifier.

Point 4 — Amorçage de la base avec le corpus existant

Quoi

Alimenter la base de données structurée (point 3) à partir des fiches actuelles, puis utiliser ce jeu de données réel pour **tester en grandeur réelle** les processus des points 2 et 3 (sources, veille, contrôle citations, arbitrage).

- **Extraction des données** depuis les fiches Markdown existantes vers le modèle structuré (tout ou partie du corpus, périmètre à décider).
- **Raccordement affirmations ↔ sources** via une table de correspondance externe (sans modifier les fiches), même si certains liens seront des hypothèses.
- **Test d'évolution des sources** — exercer le cycle complet : détection de mise à jour d'une source, impact sur les données, arbitrage, propagation.

Pourquoi

- Le corpus existant (~30+ minerais) constitue un terrain d'essai réel avec toutes ses imperfections (sources obsolètes, périmètres flous, données non vérifiées).
- Valider les processus sur des données imparfaites garantit qu'ils fonctionneront avec de meilleures sources.
- De nombreuses sources actuelles sont intéressantes et à conserver.

Décisions prises

- **Qualification progressive** — chaque donnée et chaque source est passée au crible des filtres qualité (tier de confiance, vérification citation, périmètre). Ce qui passe est conservé comme donnée de référence, ce qui ne passe pas est remplacé ou supprimé. Ce n'est pas "test puis jeter", c'est un processus de qualification.
- Le périmètre d'amorçage (tout le corpus ou un sous-ensemble) reste à décider.
- Les fiches existantes ne sont pas modifiées à cette étape.
- Cet amorçage sert aussi de validation des chantiers précédents.

Point 5 — Refonte des fiches : templates et génération

Quoi

Les fiches ne sont plus rédigées manuellement ni générées en bloc par IA. Elles deviennent un **rendu généré** à partir des données structurées (point 3) via un système de templates.

- **Templates par niveau du graphe** — chaque niveau (minerai, composant, opération, pays, etc.) a ses sections propres, adaptées à la nature des données.
- **Templates par usage** — plusieurs rendus possibles pour un même niveau :
  - **Expertise** — fiche complète avec toutes les sections, sources détaillées
  - **Synthèse** — données essentielles uniquement, vue condensée
  - **API/machine** — format structuré (JSON) pour les services externes
- La combinaison **niveau × usage** détermine le template appliqué.
- L'ordre et la composition des sections sont définis dans le template : modifier le template régénère les fiches en conséquence.
- Les sources sont intégrées nativement dans le rendu (rattachées aux données, pas listées en vrac).

Pourquoi

- Le format actuel (narration Markdown avec sections numérotées) mélange structure et contenu, rendant les fiches difficiles à maintenir et à faire évoluer.
- Les fiches actuelles ne reflètent pas le mécanisme de sources structurées (points 2 et 3).
- Pouvoir changer l'ordre des sections ou le format de rendu sans toucher aux données est essentiel pour un outil qui doit servir différents profils (admin, expert, service).

Décisions prises

- Génération des fiches depuis les données via templates (inversion du flux actuel).

- Deux axes de templates : par niveau du graphe et par usage.
- L'affichage est piloté par le modèle associé au contexte d'affichage.

Point 6 — Stockage : du DOT vers une base de données

Quoi

Remplacer le fichier DOT (Graphviz) comme stockage primaire du graphe par une **base de données** capable de supporter :

- Les données structurées rattachées aux nœuds et arêtes (indices, sources, tiers de confiance, périmètres)
- Le requêtage (filtrage par indice, par minerai, par pays, etc.)
- Un schéma évolutif (ajout d'indices eau, énergie, logistique sans refonte)
- Les rôles d'accès (point 1)
- L'exposition via API (point 1)

Le format DOT peut rester comme **format d'import/export** mais n'est plus le stockage de référence.

Pourquoi

- Le DOT ne supporte pas le requêtage, la gestion de versions des données, ni les relations entre graphes.
- L'ajout d'un indice ou d'un attribut implique aujourd'hui de modifier le parsing du DOT.
- Les évolutions prévues (multi-graphes, sources structurées, API) dépassent les capacités du format fichier.

Contexte : architecture multi-graphes (à détailler dans un point futur)

Le graphe unique actuel (4 niveaux, chaîne minerai → produit final) sera complété par :

- **Chaînes connexes** — dépendances indirectes nécessaires à la fabrication mais non constitutives du produit (ex : quartz ultra-pur → creuset → wafer).
- **Chaînes de composants détaillés** — un composant (résistance, condensateur) a sa propre chaîne de valeur complète, connectée au graphe principal via un lien au nœud composant.

Cette architecture composable permet de gérer la complexité sans surcharger le graphe principal : on "zoome" sur un composant en suivant le lien vers sa chaîne dédiée. La base de données doit supporter nativement cette interconnexion de graphes.

Décisions prises

- Le DOT n'est plus le stockage primaire.
- Le choix technologique (base graphe type Neo4j Community, PostgreSQL+AGE, ArangoDB, ou autre) reste ouvert. Toutes les options sont open source.
- **Backup** : stratégie de sauvegarde à définir, mais simplifiée par la cadence d'évolution modérée du graphe.
- **Cadence de mise à jour** : mensuelle au démarrage (pour roder le workflow), puis trimestrielle une fois stabilisé. Alertes veille hors cadence pour les événements critiques.
- L'architecture multi-graphes (essentiel/connexe) est identifiée comme point futur à part entière.

Point 7 — Validation du modèle de niveaux

Quoi

Revoir et valider les niveaux de décomposition de la chaîne de valeur. Deux axes de réflexion :

- **Les étapes de la chaîne (axe vertical)** — les niveaux actuels (Extraction, Réserves, Traitement, Fabrication, Assemblage) ne sont pas universels. Certains minerais sautent des étapes (hélium : pas de raffinage), d'autres ne sont pas des minerais classiques (dérivés pétroliers). Piste privilégiée : des **intitulés génériques** (ex : "Transformation primaire" plutôt que "Raffinage") pour que tous les matériaux se positionnent aux mêmes niveaux, facilitant la comparaison et le requêtage.
- **La profondeur des dépendances (axe horizontal)** — jusqu'où remonte-t-on ? L'excavatrice qui extrait le minerai a sa propre chaîne de valeur. Critère d'arrêt : si une partie du graphe réel est identifiée comme pertinente, elle est intégrée en **chaîne connexe** (cf. point 6) sous forme de graphe minimum associé au bon niveau.

Méthode de validation

La validation doit être **argumentée**, par au moins une des approches suivantes :

- **Revue de littérature** — benchmark avec les modèles de chaînes d'approvisionnement existants, incluant les standards du domaine (SCOR model, ISO 28000, Open Supply Hub, Responsible Minerals Initiative).
- **Démonstration par les marges d'erreur** — montrer qu'avec les niveaux d'erreur cumulés le long de la chaîne, ajouter des niveaux n'améliorerait pas la précision des résultats.

Pourquoi

- Le modèle de niveaux conditionne toute la structure du graphe, les indices, les templates de fiches et l'API.
- Un modèle mal calibré (trop simple ou trop détaillé) compromet soit la pertinence soit l'utilisabilité.
- La crédibilité scientifique du projet nécessite des choix de modélisation justifiés.

Questions ouvertes

- Quels intitulés génériques pour les niveaux ? (à définir après revue de littérature)
- Le nombre de niveaux doit-il être fixe ou configurable par chaîne ?

Modèle de niveaux — Cas possibles

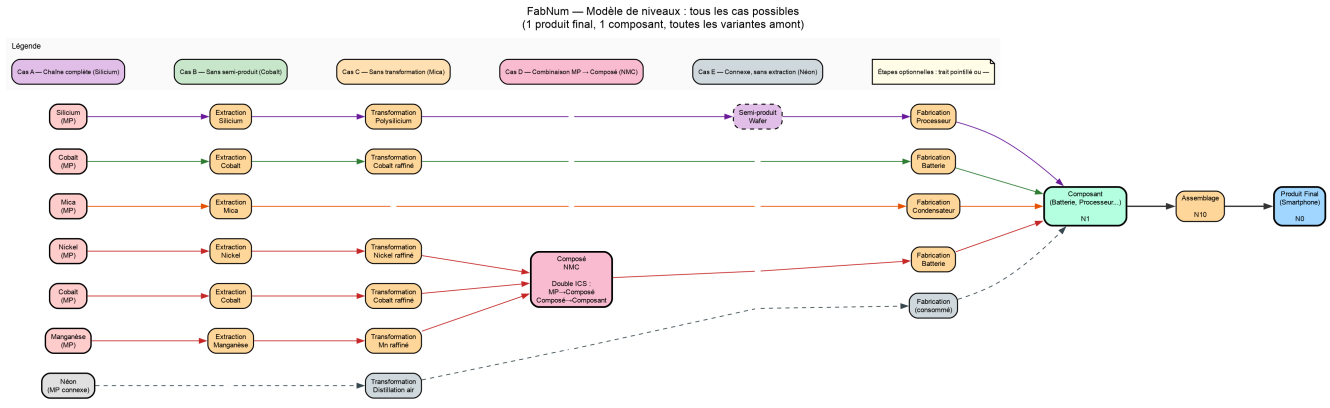

Figure 2: Modèle de niveaux — Cas possibles

Point 8 — Services, sécurisation et profils de visibilité

Quoi

Définir les services exposés par l'API, leur sécurisation et les niveaux de visibilité.

Familles de services identifiées :

- **Requêtage du graphe** — chemins critiques, filtrage, sous-graphes. Deux niveaux : expert (complet) et simplifié (clients).
- **Méta-indices dynamiques** — indices agrégés calculés sur un sous-graphe sélectionné (ex : IHH agrégé de la chaîne cobalt → smartphones). Les indices de base sont pré-calculés à chaque mise à jour du graphe et servis comme données.
- **Analyse** — vulnérabilités, comparaison de scénarios. Deux niveaux (expert / simplifié).
- **Administration** — CRUD graphe, gestion des sources, import/export (rôle admin uniquement).
- **Données structurées** — fiches (tous templates/usages), données brutes, sources associées.
- **Génération** — rapports, exports (PDF, JSON, DOT) — consommateurs externes.
- **Personnalisation client** — un client fournit son SI / sa nomenclature, on le projette sur le graphe FabNum pour obtenir :
  - Sa carte de risques propre (minerais critiques pour *son* SI)
  - De la veille/alertes ciblées sur *ses* dépendances
  - Des rapports contextualisés

Sécurisation :

- **Authentification** — utilisateur (Streamlit) vs service externe (clé API / token).
- **Autorisation par rôle** (admin, expert, service) — cf. point 1.
- **Profils de visibilité** — complémentaires aux rôles, définissent *ce qu'on peut voir* :
  - Chaîne principale seule vs chaîne + connexes
  - Tous les indices vs un sous-ensemble
  - Graphe complet vs sous-graphe filtré
  - Profils prédéfinis (ex : gratuit/standard/complet) ou sur mesure par client
- **Rate limiting** — protection du moteur contre les abus.
- **Monitoring / observabilité** — logging structuré, métriques, alerting (service d'administration).

Données client et RGPD :

- **Principe : moteur stateless côté client** — le client envoie son SI, le moteur croise avec le graphe, renvoie le résultat, rien ne persiste sur le serveur. Faisabilité à valider selon les traitements (veille ciblée → nécessite potentiellement du stockage).
- **Alternative : agent local chez le client** — un agent déployé côté client consomme l'API et stocke les résultats localement. Les données client ne quittent jamais leur périmètre.
- **Action manuelle** — export/import pour les cas simples.
- **RGPD** — à garder à l'esprit pour la phase commerciale.

Documentation utilisateur :

- Prématurée mais importante. À produire quand le système sera stabilisé (API, fiches, services).

Pourquoi

- L'API sans services définis et sécurisés est inutilisable.
- La personnalisation client transforme FabNum d'un outil générique en **plateforme de services à valeur ajoutée**.
- Les profils de visibilité permettent de proposer des niveaux de service différenciés et de protéger les données sensibles.

Décisions prises

- Les indices de base sont pré-calculés (pas de service de calcul à la volée), sauf les méta-indices dynamiques sur sélection.

- La personnalisation client (projection de SI sur le graphe) est identifiée comme service à forte valeur.
- Rôles (ce qu'on peut faire) et profils (ce qu'on peut voir) sont deux axes distincts.
- Moteur stateless côté données client (objectif), agent local client ou export manuel comme alternatives.

Point 9 — Multi-sectoriel et architecture des graphes

Quoi

FabNum ne se limite pas nécessairement au numérique. Le moteur pourrait servir d'autres secteurs (embarqué, automobile, énergie…) avec la même mécanique.

- Chaque **secteur** a son graphe principal (sa chaîne de valeur propre).
- Les secteurs peuvent devenir des **connexes les uns des autres** (l'embarqué est un connexe de l'automobile ; le numérique est un connexe de l'embarqué).
- Les **minerais sont partagés** entre les graphes — le cobalt apparaît dans le numérique, l'automobile et l'embarqué.
- Le moteur reste le même, seuls les graphes et leurs interconnexions changent.

Ce point englobe et généralise la réflexion essentiel/connexe mentionnée au point 6.

Pourquoi

- L'architecture multi-graphes (point 6) prend tout son sens à l'échelle multi-sectorielle.
- Les minerais critiques sont par nature transversaux — les analyser secteur par secteur sans voir les interdépendances est réducteur.
- Ça renforce le besoin d'un modèle de niveaux générique (point 7) et d'une base de données capable de gérer ces relations (point 6).

Décisions prises

- La vision multi-sectorielle est retenue comme horizon, pas comme priorité immédiate.
- L'architecture doit la rendre possible sans la requérir dès le départ.

Point 10 — Intégration des ressources transversales (énergie, eau)

Quoi

Énergie et eau ne sont pas des niveaux de la chaîne de valeur mais des **ressources transversales** consommées à chaque opération (extraction, transformation, fabrication, assemblage). Il faut :

- Définir **comment les modéliser** dans le graphe — attributs d'opération, nœuds de ressource connectés aux opérations, ou autre approche.
- Résoudre le **problème de granularité** — il faut pouvoir rattacher la consommation au bon niveau de chaque opération, avec une dimension géographique (disponibilité eau/énergie par pays).
- Travailler sur le **sourçage** de données aujourd'hui parcellaires — les consommations énergétiques et hydriques par opération et par pays sont mal documentées. Il faudra probablement travailler avec des fourchettes et estimations, en s'appuyant sur le système de tiers de confiance (point 3).

L'objectif est de pouvoir détecter des **croisements de risques** : pays instable (ISG) + stress hydrique + concentration d'une opération (IHH) = alerte.

Pourquoi

- L'énergie est un facteur de risque majeur et croissant (cf. détroit d'Ormuz et impact sur la chaîne numérique : Impact du détroit d'Ormuz sur la chaîne numérique).
- L'eau est un facteur de risque émergent, en particulier pour l'extraction et la transformation (lithium au Chili, semi-conducteurs à Taïwan).
- Sans ces données, le modèle est aveugle à des scénarios de rupture réalistes.

Difficultés identifiées

- Données très parcellaires — pas de base de données globale "consommation énergie/eau par opération minière par pays".
- Granularité hétérogène des sources (données pays vs données site vs données sectorielles).
- Les données devront être intégrées progressivement, au fur et à mesure de leur disponibilité.

Décisions prises

- Identifié comme chantier nécessaire mais difficile — la réflexion sur le sourçage et la modélisation est à mener.
- Le modèle de données (point 3) et la base de données (point 6) doivent être conçus pour accueillir ces ressources transversales, même si les données ne sont pas disponibles immédiatement.

Point 11 — Granularité géographique fine et risques climatiques

Quoi

Passer d'une géographie au niveau pays à une **localisation fine des opérations** (région, province, site) pour croiser avec des données de risques climatiques et environnementaux (stress hydrique, inondation, canicule, sécheresse).

- **Localisation fine des opérations** — identifier où se trouvent physiquement les sites de production, et idéalement la part de production de chaque site.
- **Croisement avec des données de risques** — stress hydrique (WRI Aqueduct, données ouvertes), risques d'inondation/canicule (ND-GAIN), événements historiques.
- **Alertes contextualisées** — un événement climatique dans une zone → alerte automatique sur les opérations concernées (lien avec la veille, point 2b).

Niveaux d'approximation envisagés :

Niveau	Granularité	Données nécessaires	Faisabilité
1	Pays	Risque climatique moyen pays	Disponible (ISG actuel)
2	Région/province	Données climatiques régionales	Faisable (WRI Aqueduct, ND-GAIN)
3	Site exact + part de production	Géolocalisation + répartition production	Difficile, partiel

Le niveau 3 ne sera pas atteignable pour tous les opérateurs, mais peut l'être pour les **nœuds les plus critiques** — ceux à IHH élevé où un incident aurait un impact systémique (exemples : Spruce Pine pour le quartz ultra-pur, Bayan Obo pour les terres rares, triangle du lithium).

Pourquoi

- Spruce Pine (2024) : une seule mine de quartz ultra-pur, un ouragan, et toute la chaîne des semi-conducteurs menacée.

- 40% des nouvelles fabs de semi-conducteurs sont construites dans des zones en stress hydrique.
- Le risque climatique au niveau pays est trop grossier — la Chine a des zones arides et des zones inondables, un risque moyen pays n'a pas de sens.
- L'utilité est certaine ; c'est la faisabilité du sourçage qui doit être évaluée.

Difficultés identifiées

- **Opacité** — les opérateurs (surtout chinois et petits producteurs) ne publient pas toujours la localisation de leurs sites.
- **Répartition de production** — même quand on connaît les sites d'un opérateur, la part de chaque site dans sa production totale est rarement publique.
- **Hétérogénéité des sources** — données de géolocalisation minière (S&P Global, USGS mineral facilities) vs données climatiques (WRI, ND-GAIN) vs rapports annuels d'entreprises — formats et granularités différents.

Décisions prises

- L'utilité est validée, la faisabilité est à évaluer.
- Approche par niveaux d'approximation — on ne vise pas le niveau 3 partout, mais en priorité sur les nœuds critiques (IHH élevé, impact systémique).
- Lien fort avec la veille (point 2b) pour les alertes événementielles.

Point 12 — Stocks et projection temporelle

Quoi

Intégrer les **niveaux de stock** comme donnée du graphe pour permettre des **scénarios de projection temporelle** : en cas de rupture d'approvisionnement, estimer le délai d'impact le long de la chaîne de valeur.

- **Stock comme attribut de nœud** — chaque nœud pertinent (opération, composant) porte une estimation de stock (en mois de consommation, tonnes, unités).
- **Moteur de projection** — simulation de rupture à un point du graphe et propagation temporelle : "la Chine coupe le germanium → les fibres optiques sont impactées dans X mois, les serveurs dans Y mois".
- **Scénarios what-if** — un service de calcul dynamique, pas juste une requête sur le graphe.

Pourquoi

- Passe FabNum de l'analyse **statique** (photo de la criticité à un instant T) à l'analyse **dynamique** (projection temporelle en cas de rupture).
- Valeur très forte pour les clients : "votre SI dépend du germanium via les fibres optiques, vous avez ~8 mois avant impact en cas de coupure chinoise".
- Les crises récentes (quotas chinois, Spruce Pine) montrent que le délai d'impact est une information critique pour la prise de décision.

Difficultés identifiées

- **Données extrêmement sensibles** — les niveaux de stock sont la donnée la plus confidentielle pour les entreprises. Quasiment jamais publiée.
- **Niveau de confiance faible** — on travaillera avec des estimations sectorielles, des moyennes, des fourchettes. L'indice de fiabilité (point 3) sera particulièrement bas sur ces données.
- **Variabilité rapide** — les stocks évoluent beaucoup plus vite que les parts de marché. La cadence trimestrielle pourrait ne pas suffire pour cette donnée.

Décisions prises

- Point identifié comme horizon — forte valeur, faisabilité difficile (sourçage).
- Le stock est un attribut de nœud comme les autres — pas de refonte du modèle de données.
- Le moteur de projection est un nouveau service de calcul à spécifier.

Ordre de réalisation logique

L'ordre des points dans ce document reflète l'ordre de la discussion, pas l'ordre de réalisation. L'analyse des dépendances donne :

Phase 1 — Fondations (parallélisable) :

- **Point 7** — Validation du modèle de niveaux (fondation théorique, prérequis au modèle de données)
- **Point 6 (spike)** — Spike technique sur 2-3 bases candidates (Neo4j Community, PostgreSQL+AGE, ArangoDB) avec un mini-graphe de test. En parallèle du point 7 car les critères de sélection (multi-graphes, schéma évolutif, requêtage, historisation, open source) sont indépendants du nombre exact de niveaux. Tests sur le serveur via Docker (16 Go RAM), installation native ensuite.

Phase 2 — Modèle et stockage :

1. **Point 3** — Modèle de données structuré (dépend des niveaux validés). Énergie et eau ne nécessitent pas de modélisation spéciale — ce sont des attributs d'opération comme les parts de marché, couverts nativement par le modèle extensible.
2. **Point 6 (implémentation)** — Mise en place du stockage retenu après le spike.

Phase 3 — Données et sources :

1. **Point 2** — Gestion des sources (peut démarrer en parallèle dès la phase 2 pour l'identification/catégorisation)
2. **Point 4** — Amorçage et qualification sur un échantillon représentatif : 3 minerais + 1 composant + 1 produit final (couvre tous les niveaux du graphe). Puis audit d'exhaustivité des minerais après l'amorçage.

Phase 4 — Fiches et API :

1. **Point 5** — Templates et génération des fiches
2. **Point 1** — API REST et rôles
3. **Point 8** — Services, sécurisation, profils (sous-spécification dédiée pour la personnalisation client)

Horizon (l'architecture doit les rendre possibles sans les implémenter d'emblée) :

- **Point 10** — Ressources transversales (énergie, eau)
- **Point 11** — Granularité géographique fine
- **Point 9** — Multi-sectoriel

MVP (jalon interne)

Définition : moteur fonctionnel avec données réelles dans une vraie base, requêtable, données sourcées avec indice de fiabilité. Pas de frontend, pas de produit fini — un jalon de validation interne.

Périmètre : phases 1 + 2 + 3 (validation niveaux → spike techno → modèle de données → base → amorçage 3 minerais + 1 composant + 1 produit final).

Critère de succès : on peut requêter la base et retrouver les mêmes analyses qu'aujourd'hui (ex : Sankey du germanium), avec des données sourcées et un indice de fiabilité.

Rétro-analyse : valider les indices sur des crises passées (Spruce Pine, quotas chinois) comme preuve de concept.

Tests : toute logique métier est testée (calculs d'indices, requêtage, arbitrage des sources, import/export). Pas de seuil de couverture arbitraire — on teste ce qui compte.

Planification temporelle

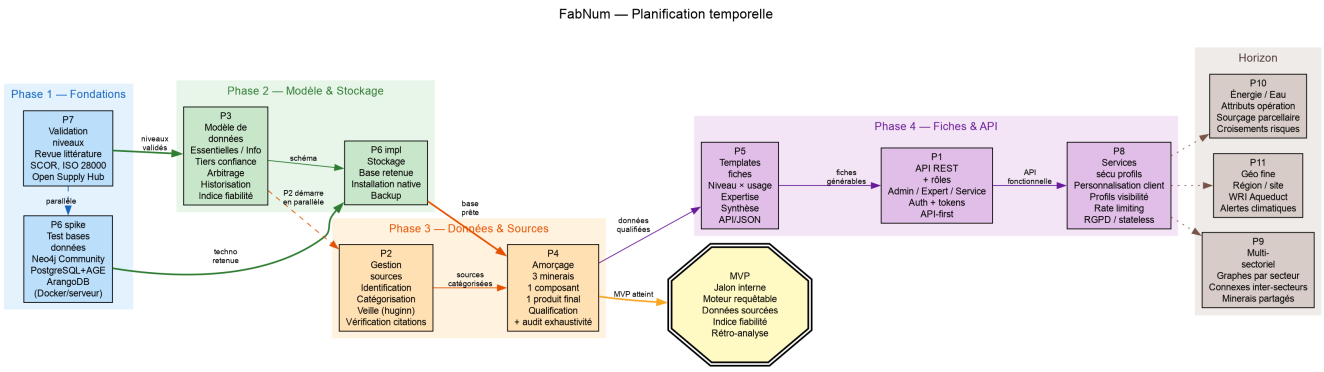

Figure 3: Planification temporelle

Diagramme d’Architecture Fonctionnelle (DAF)

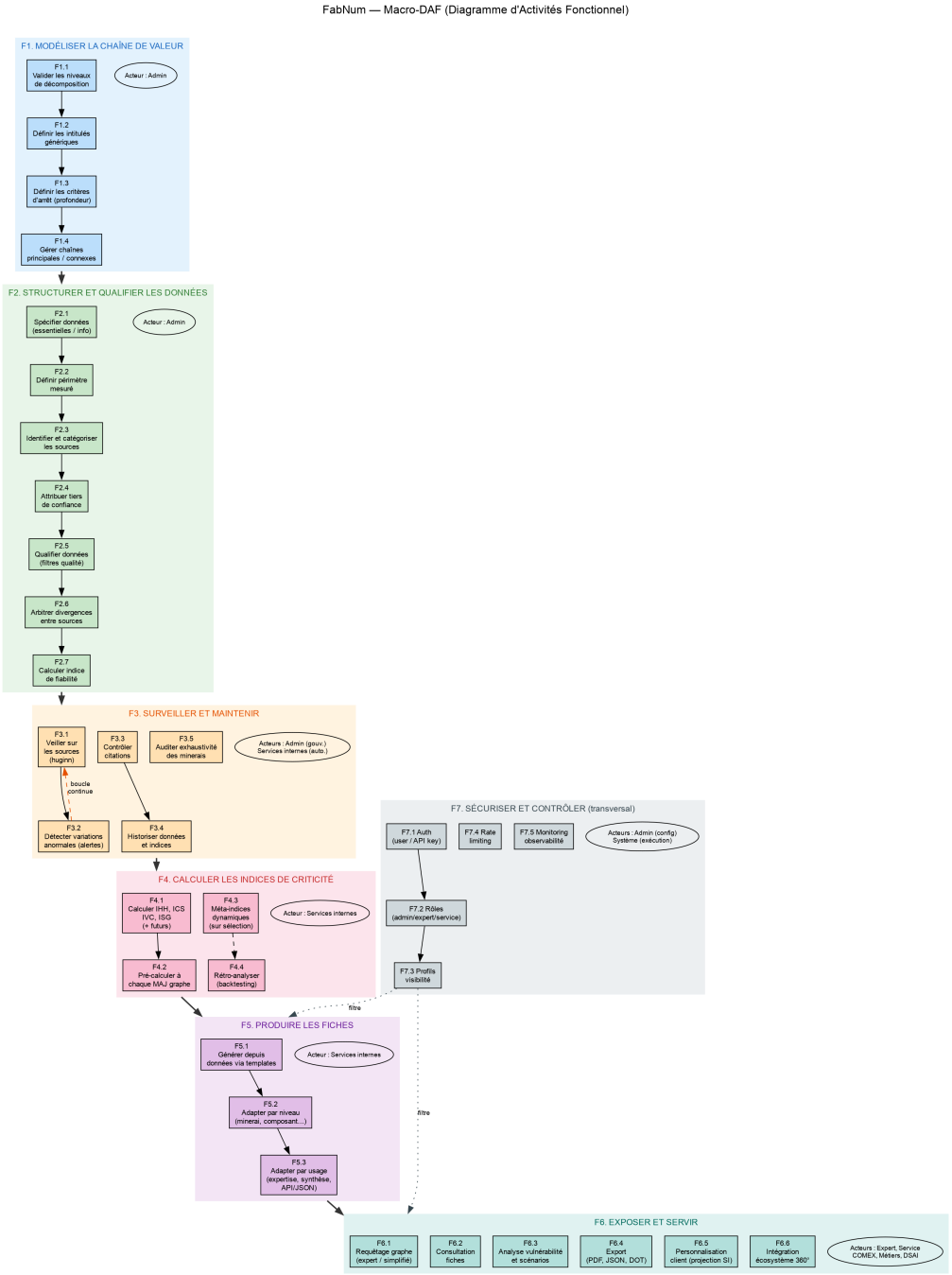

Figure 4: Macro-DAF

Diagramme d’Architecture Technique (DAT)

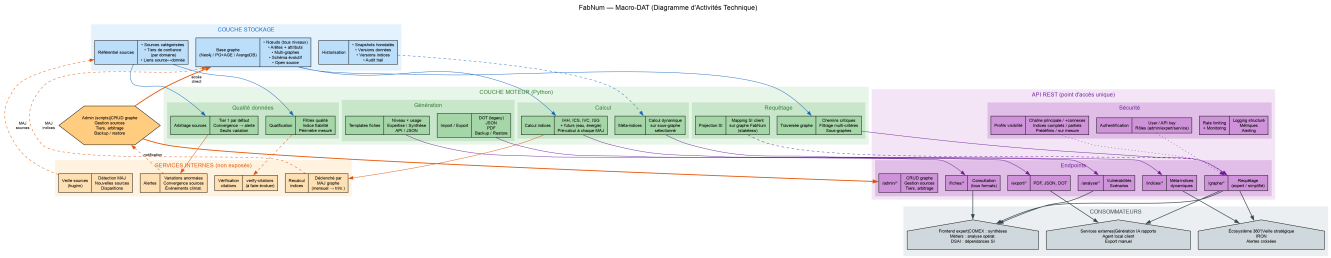

Figure 5: Macro-DAT