

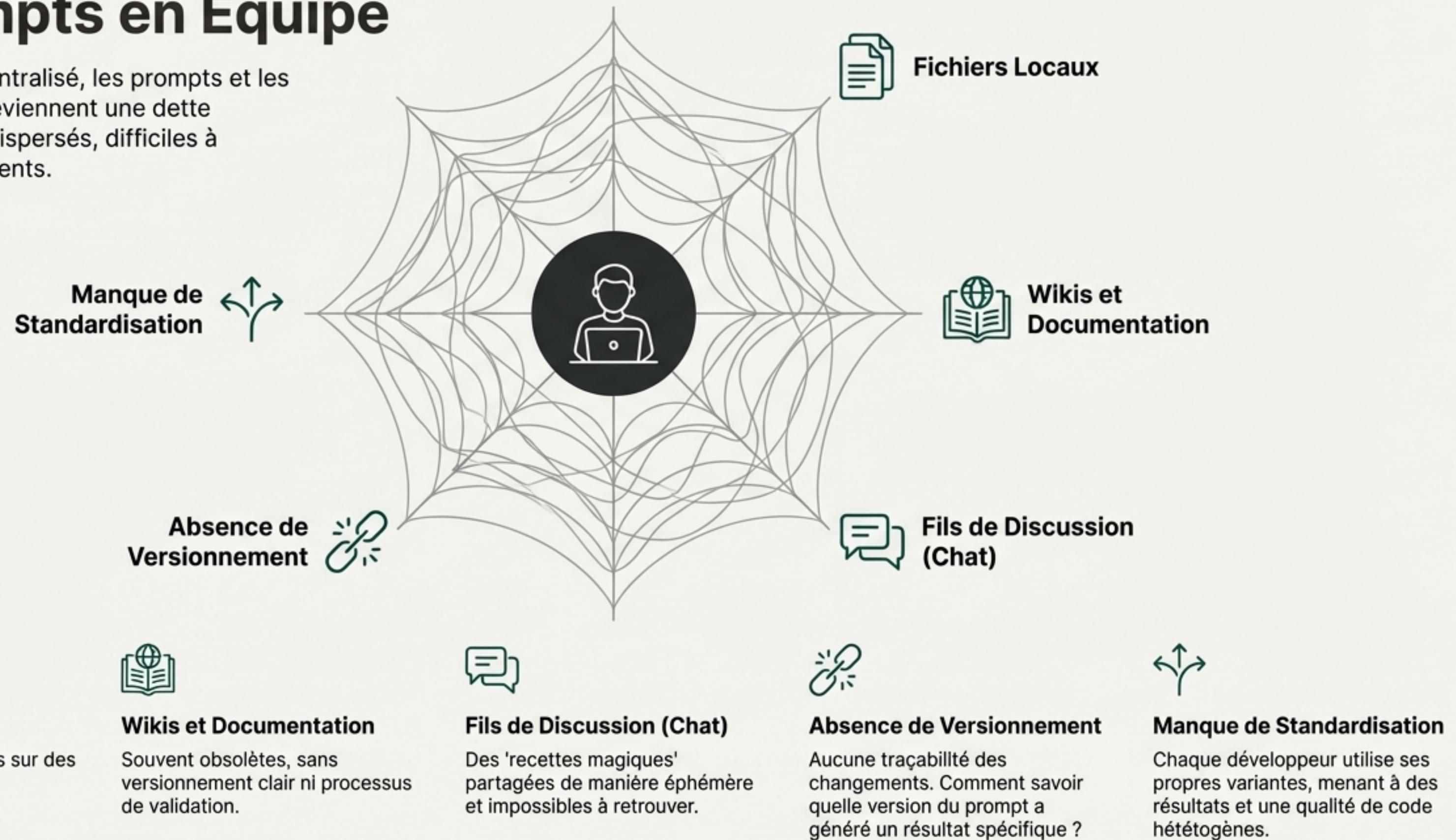
# **De l'Artisanat à l'Ingénierie : Standardiser les Prompts pour les Équipes DevOps**

Mettre en place une source de vérité versionnée, multi-niveaux et validée pour vos assistants de code IA.

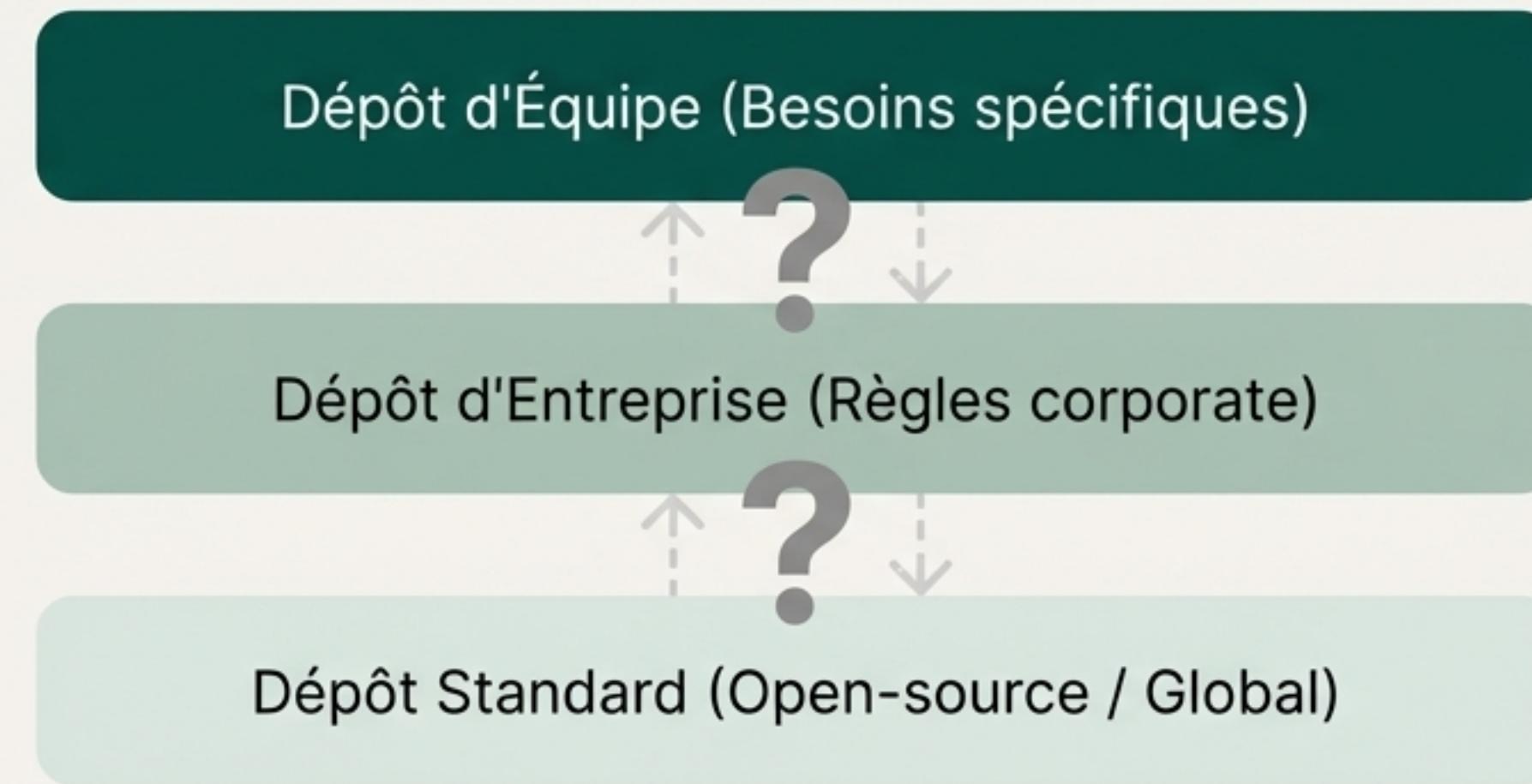
**PJ Prompt Unifier**

# Le Chaos Actuel : Gérer les Prompts en Équipe

Sans un système centralisé, les prompts et les règles de codage deviennent une dette technique. Ils sont dispersés, difficiles à maintenir et incohérents.



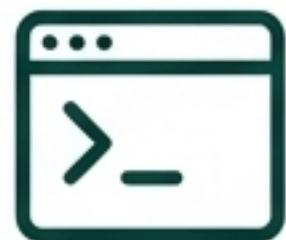
# Le Défi de la Gouvernance à l'Échelle



**Comment concilier des standards globaux, des règles d'entreprise et des besoins spécifiques à une équipe ?**

# La Solution : L'Écosystème Git-Ops de Prompt Unifier

Une approche Git-Ops pour la gestion de prompts, transformant le chaos en un workflow structuré, collaboratif et fiable. L'écosystème repose sur deux piliers : un outil CLI et un standard de dépôt.



## **`prompt-unifier` - Le Moteur d'Orchestration**

Un outil CLI en Python pour synchroniser, valider et déployer vos prompts et règles depuis des sources multiples.

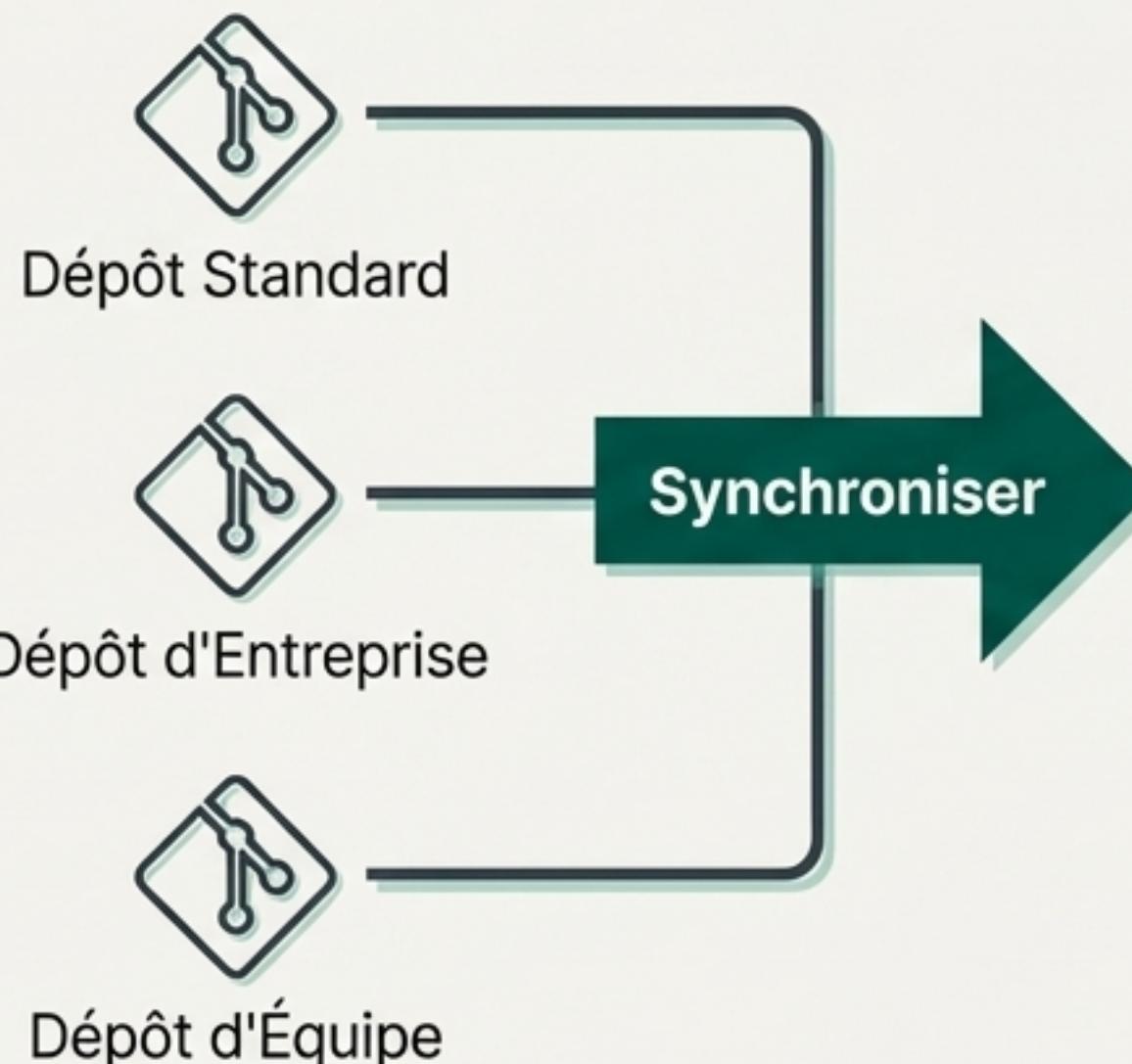


## **`prompt-unifier-data` - La Source de Vérité**

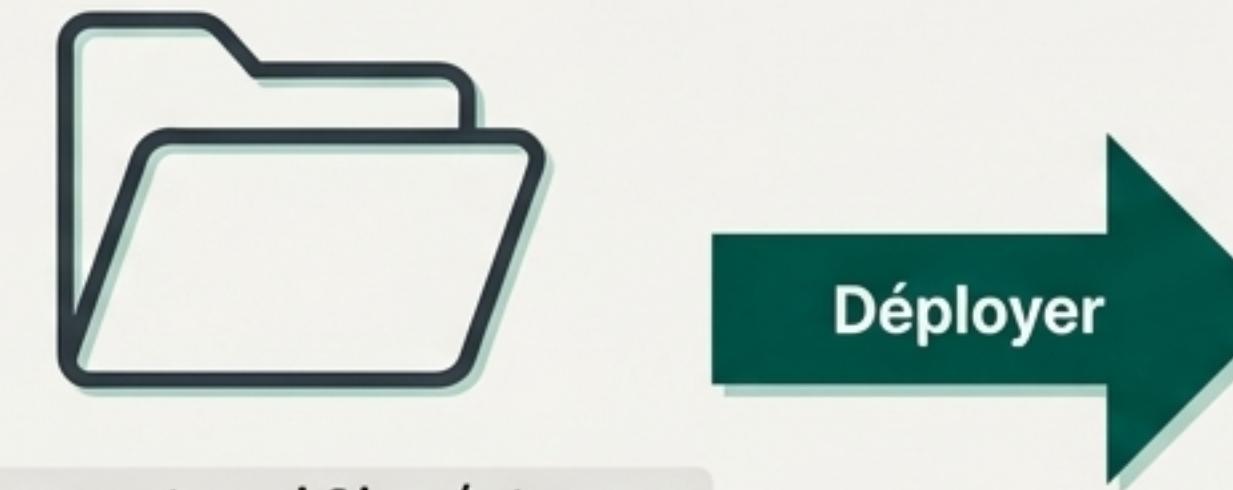
Un dépôt Git structuré servant de collection centralisée et versionnée pour tous les prompts et règles.

# Un Workflow Unifié pour des Sources Multiples

## Phase 1: Dépôts Git Distants



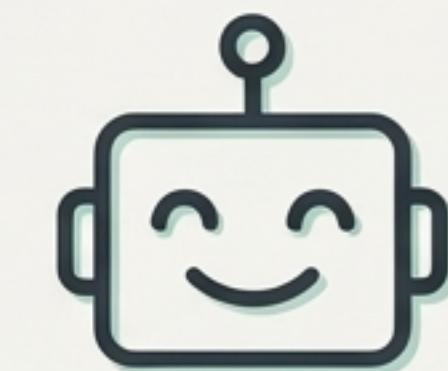
## Phase 2: Stockage Local Centralisé



/.prompt-unifier/storage

Prompts et règles fusionnés.  
Stratégie "le dernier gagne" :  
l'équipe surcharge l'entreprise,  
qui surcharge le standard.

## Phase 3: Utilisation Active



Prompts et règles prêts à  
l'emploi dans vos outils  
(Continue, Kilo Code, etc.).

# Zoom n°1 : Anatomie d'un Dépôt de Données

Une structure claire et prévisible, consistante à travers tous les dépôts sources (standard, entreprise, équipe).

```
└─ prompts/ // Modèles pour instruire les IA
    └─ airflow/
    └─ kubernetes/
    └─ python/
└─ rules/ // Standards et bonnes pratiques
    └─ global/
    └─ airflow/
    └─ python/
```

Contient les templates utilisés pour instruire les modèles IA pour des tâches spécifiques (génération, refactoring, analyse, etc.).

Contient les guides, standards et bonnes pratiques. Ces fichiers servent de contexte enrichi pour les IA, assurant que les résultats respectent les normes de l'équipe.

# Zoom n°2 : Anatomie d'une Règle : Contexte et Standards

```
rules/airflow/01-airflow-dag-standards.md
```

```
---
```

```
title: "Airflow DAG Standards"
```

```
description: "Core standards for designing and structuring Airflow DAGs..."
```

```
tags: [airflow, dag, standards]
```

```
version: 1.0.0
```

```
category: "standards"
```

```
applies_to: ["dags/**/*.py"]
```

```
---
```

```
# Airflow DAG Standards
```

```
## 1. Core DAG Principles
```

```
### Idempotency and Determinism
```

```
- **Idempotency**: Every task must be idempotent...
```

## Métadonnées Structurées

**(YAML Frontmatter):** Permet de catégoriser, versionner et filtrer les règles. Le champ `applies\_to` lie la règle à des fichiers spécifiques.

## Contenu Lisible (Markdown):

Facile à écrire et à maintenir pour les humains.

**Contexte pour l'IA:** Peut être injecté dans un prompt pour guider la génération de code selon les standards définis.

# Zoom n°3 : Anatomie d'un Prompt : Le Framework S.C.A.F.F.

Les prompts sont des modèles pour instruire une IA. Ils suivent le framework S.C.A.F.F. pour garantir clarté, contexte et précision.

## S.C.A.F.F.

### Situation:

Quel est le contexte de départ ?

### Challenge:

Quelle est la tâche précise à accomplir ?

### Audience:

À qui s'adresse le résultat ?

### Format:

Quelle est la structure de sortie attendue ?

### Foundations:

Quelles sont les règles ou principes fondateurs à respecter ?

prompts/python/01-python-refactoring.md

```
---  
title: "Python Refactoring Assistant"  
description: "Refactor Python code to improve readability..."  
---
```

You are an expert Python developer...

#### ### Situation

The user provides a piece of Python code...

# Le Moteur : `prompt-unifier`, le CLI au Cœur du Système



Versionnement Git



Gestion Centralisée



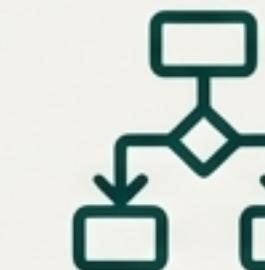
Validation Intégrée



Déploiement Facile



Support Multi-Dépôts



Organisation Structurée

Harmonisez les standards tout en préservant l'autonomie des équipes.

# Le CLI en Action (1/3) : Synchronisation Multi-Niveaux

## >\_ init ⚙

Crée un fichier de configuration .prompt-unifier/config.yaml et prépare le répertoire de stockage local.

```
prompt-unifier init
```

## ⇄ sync ⇄

Clone ou met à jour les prompts depuis plusieurs dépôts. La stratégie 'le dernier gagne' est idéale pour surcharger les standards.

```
# Synchroniser plusieurs dépôts (le dernier gagne)
prompt-unifier sync \
    # 1. Base de prompts standards (open-source)
    --repo https://gitlab.com/waewoo/prompt-unifier-data.git \
    # 2. Standards de l'entreprise (surcharge la base)
    --repo https://gitlab.com/my-company/company-prompts.git \
    # 3. Prompts de l'équipe (surcharge l'entreprise)
    --repo https://gitlab.com/my-team/team-prompts.git
```

# Le CLI en Action (2/3) : Vérification et Exploration

Validez la syntaxe et la conformité de vos fichiers \*après\* leur fusion, et explorez facilement le contenu disponible.

## ✓ `validate`

Vérifie la syntaxe du frontmatter YAML, la présence des champs requis et la structure des fichiers sur l'ensemble du contenu fusionné.

```
# Valider l'intégralité du stockage
# synchronisé
prompt-unifier validate

# Valider uniquement les prompts avec une
# sortie détaillée
prompt-unifier -v validate --type prompts
```

## 📋 `list` et `status` ↗

Affichez une table de tous les prompts/règles disponibles. Vérifiez l'état de synchronisation de vos dépôts.

```
# Lister tous les contenus triés par date
prompt-unifier list --sort date

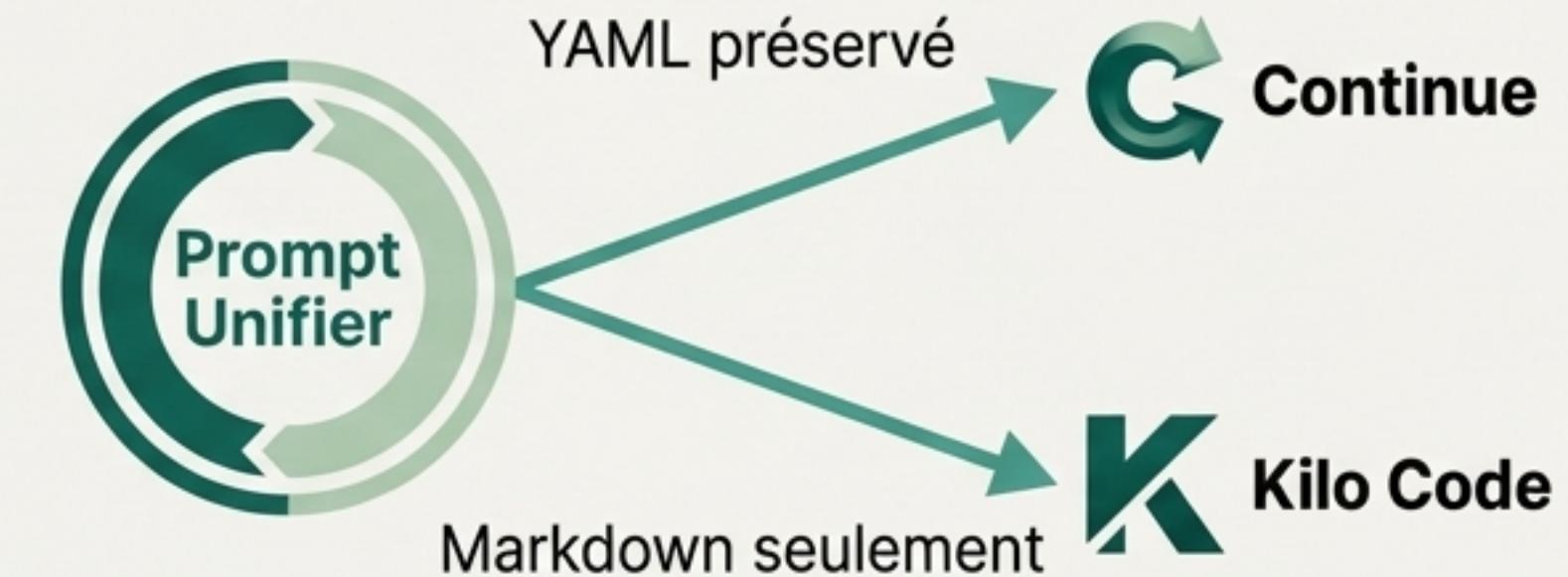
# Vérifier si de nouveaux commits sont
# disponibles
prompt-unifier status
```

# Le CLI en Action (3/3) : Déploiement vers Vos Outils

Déployez vos prompts et règles validés vers vos assistants de code IA. Le système gère les spécificités de chaque outil via des "handlers".

## Handlers Supportés

- **Continue:** Support complet, préserve le frontmatter YAML. Destination : `./.continue/`
- **Kilo Code:** Convertit en Markdown pur (sans frontmatter). Destination : `./.kilocode/`



## Exemples de Déploiement Avancé

```
# Déployer uniquement les prompts tagués "python"  
prompt-unifier deploy --tags python
```

```
# Déployer vers Kilo Code et nettoyer les anciens fichiers  
prompt-unifier deploy --handlers kilocode --clean
```

```
# Simuler un déploiement pour voir les changements  
prompt-unifier deploy --dry-run
```

# La Qualité à l'Échelle : Intégration Continue (CI/CD)

Le dépôt `prompt-unifier-data` inclut une pipeline GitLab CI qui garantit que chaque contribution est validée automatiquement, empêchant les erreurs d'atteindre la branche principale.

- **Automatisation de la Qualité:**

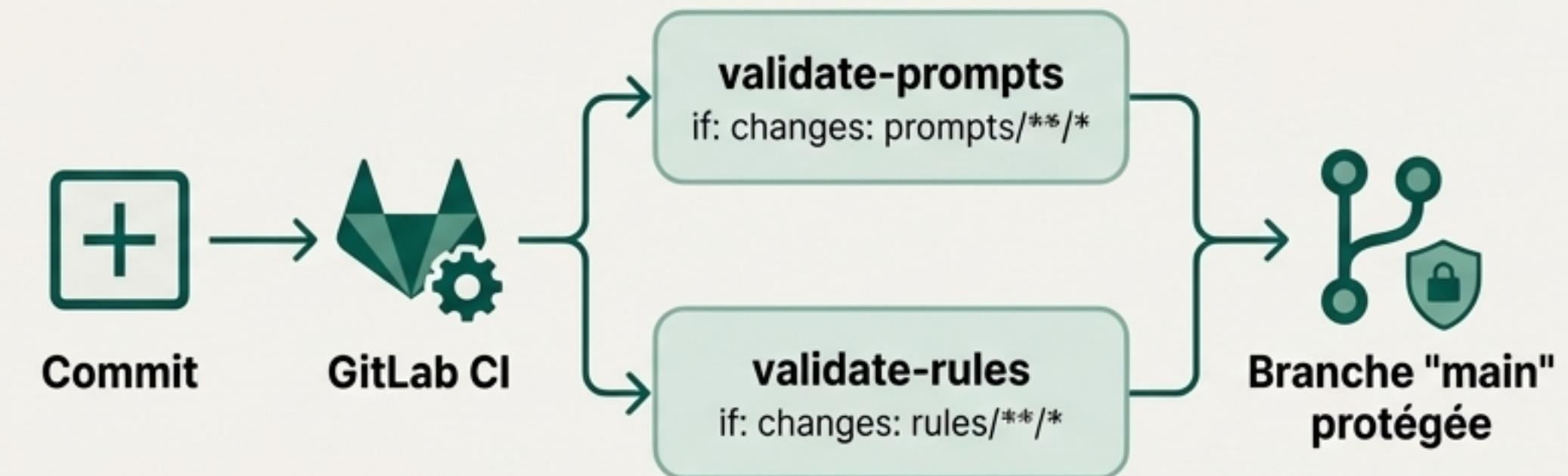
Aucune validation manuelle requise.

- **Feedback Rapide:** Les

développeurs savent immédiatement si leurs changements sont conformes.

- **Confiance et Fiabilité:** Seuls des

prompts et règles valides sont fusionnés, garantissant la stabilité de la source de vérité.



# Passez à l'Action : Adoptez une Approche Structurée

## Étapes de Démarrage Rapide

### 1. \*\*Installer le CLI\*\*

```
pip install prompt-unifier
```

### 2. \*\*Initialiser dans votre projet\*\*

```
prompt-unifier init
```

### 3. \*\*Utiliser le dépôt d'exemple comme base\*\*

Forkez le dépôt pour créer votre propre bibliothèque d'entreprise ou d'équipe.

<https://gitlab.com/waewoo/prompt-unifier-data>

## Contribution

Ce projet est open source (Licence MIT). Vos contributions sont les bienvenues.

Avant de soumettre, assurez-vous que vos changements passent la validation locale :

```
make validate
```

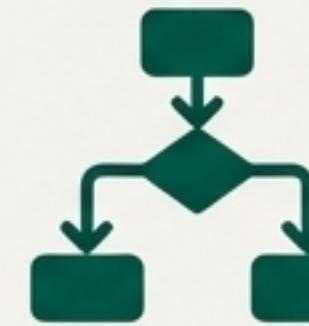
# La Transformation : de l'Artisanat à l'Ingénierie des Prompts

## Le Chaos des Prompts



Incohérent, Opaque, Fragile, Isolé.

## L'Ingénierie des Prompts



Standardisé, Fiable, Collaboratif, Scalable.

L'écosystème Prompt Unifier fournit la structure et les outils nécessaires pour faire de l'assistance par IA une discipline d'ingénierie fiable au sein de votre équipe.