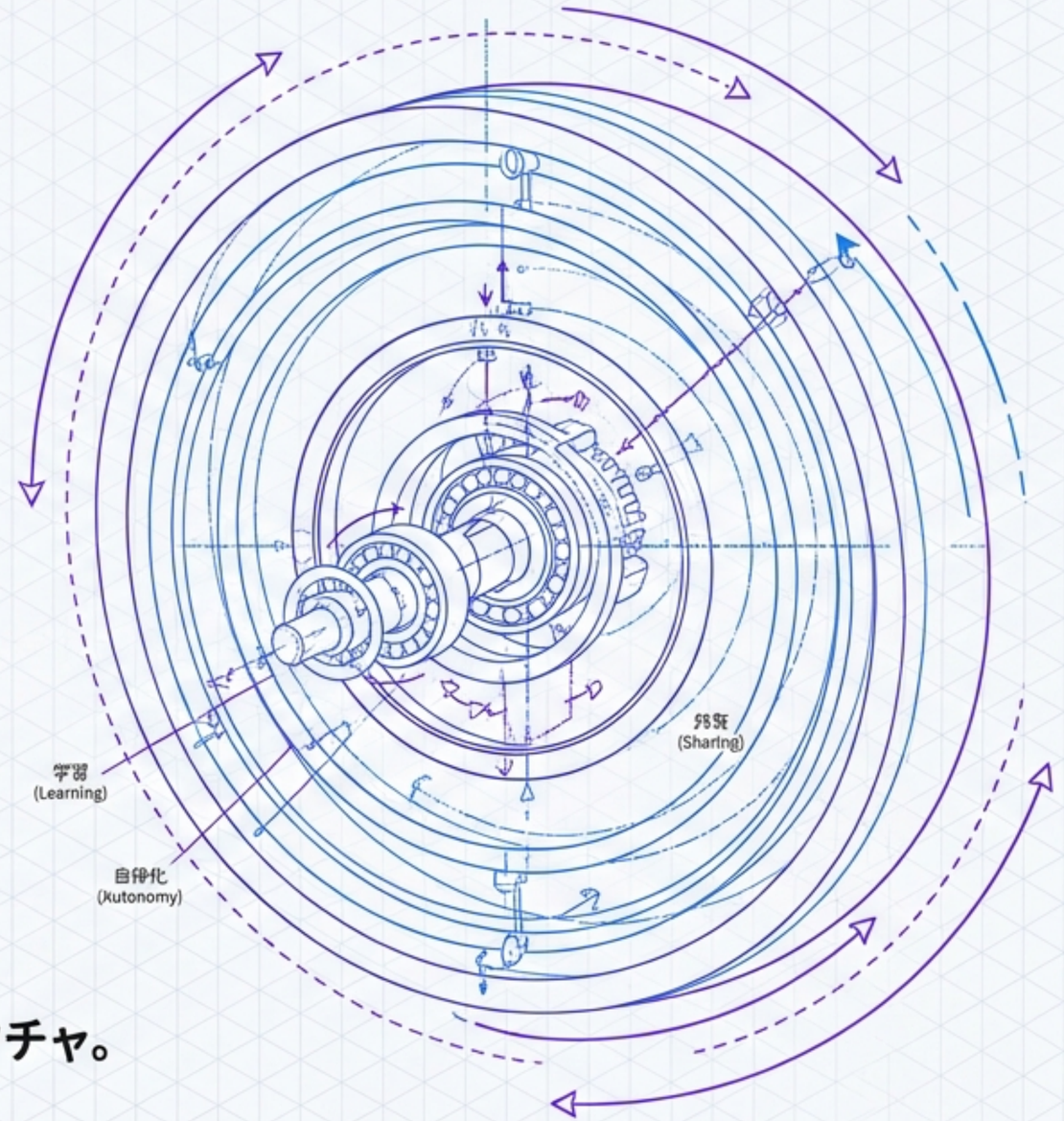
使い捨てのAIから、育てるAIへ

DeNAにおける自律的AI開発インフラの進化

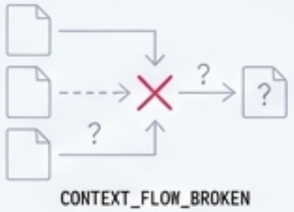


単なるコード生成ではない。
チームの「暗黙知」を学習し続けるインフラストラクチャ。

AI導入の「見えない天井」

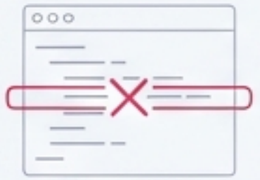
コンテキスト欠如

README以外の設計情報や設定が伝わらず、出力品質が安定しない



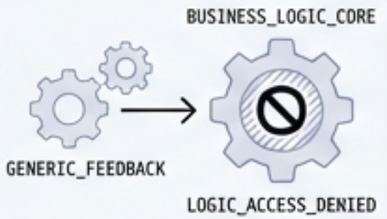
規約違反の多発

プロジェクト固有のルールを知らないため、修正の手間が発生



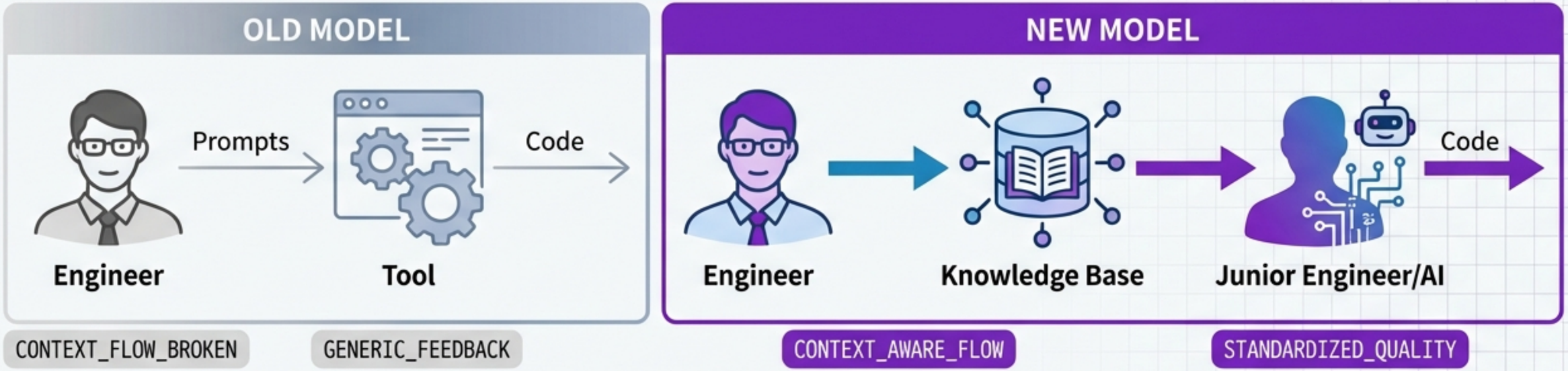
レビュー精度の限界

汎用的な指摘しかできず、ビジネスロジックに踏み込めない



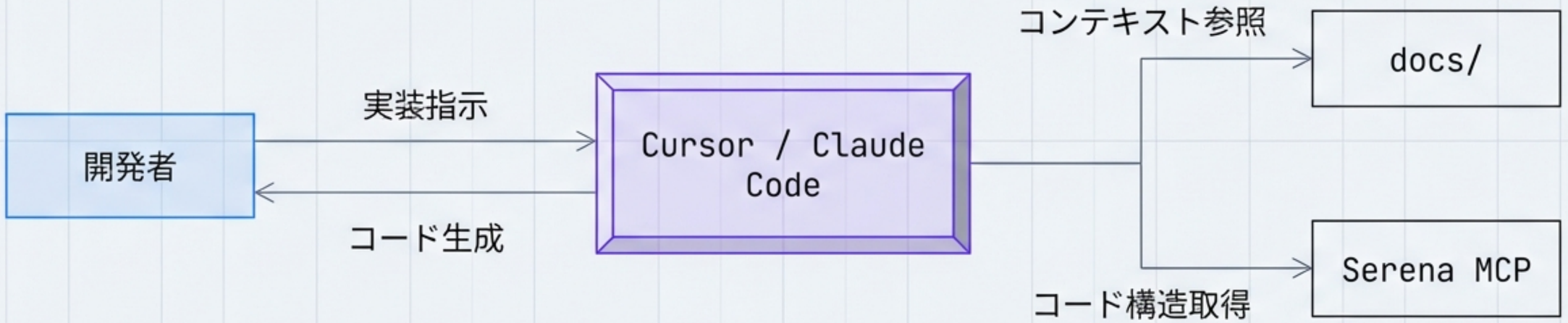
解決策：AIを「チームの一員」としてオンボーディングする

「育てるほど楽になる」

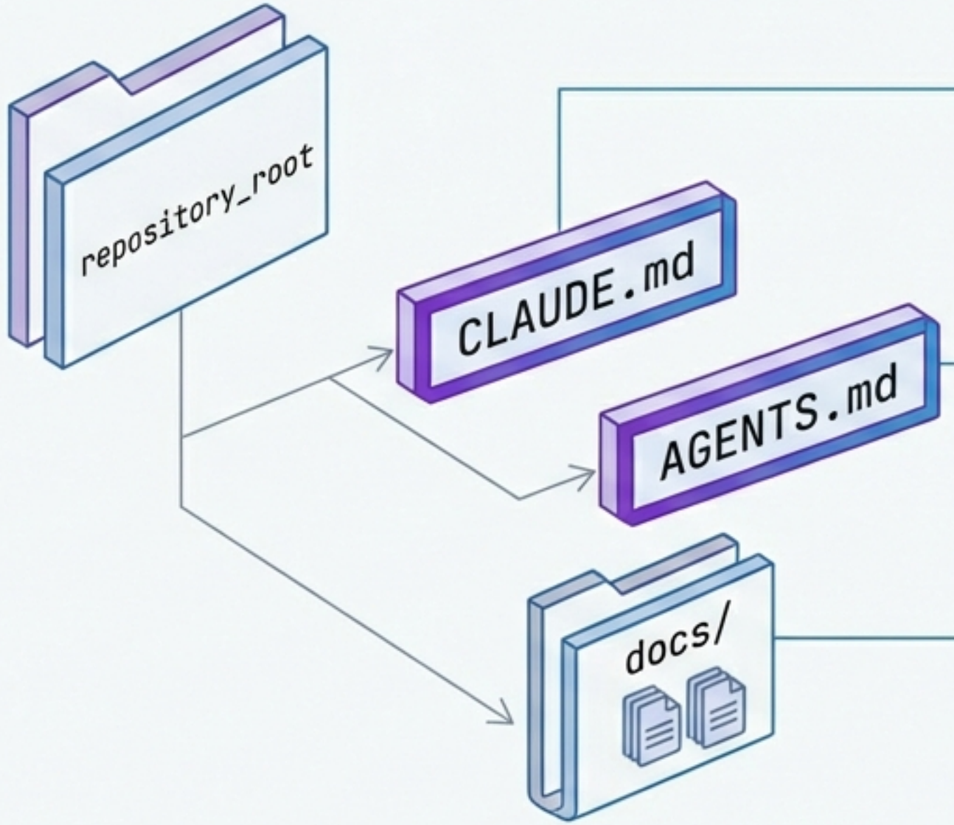


- 人間のレビュー量を減らす
- 成果物の品質を標準化する

全体アーキテクチャ：知識循環ループ



リポジトリ自体を「脳」にする



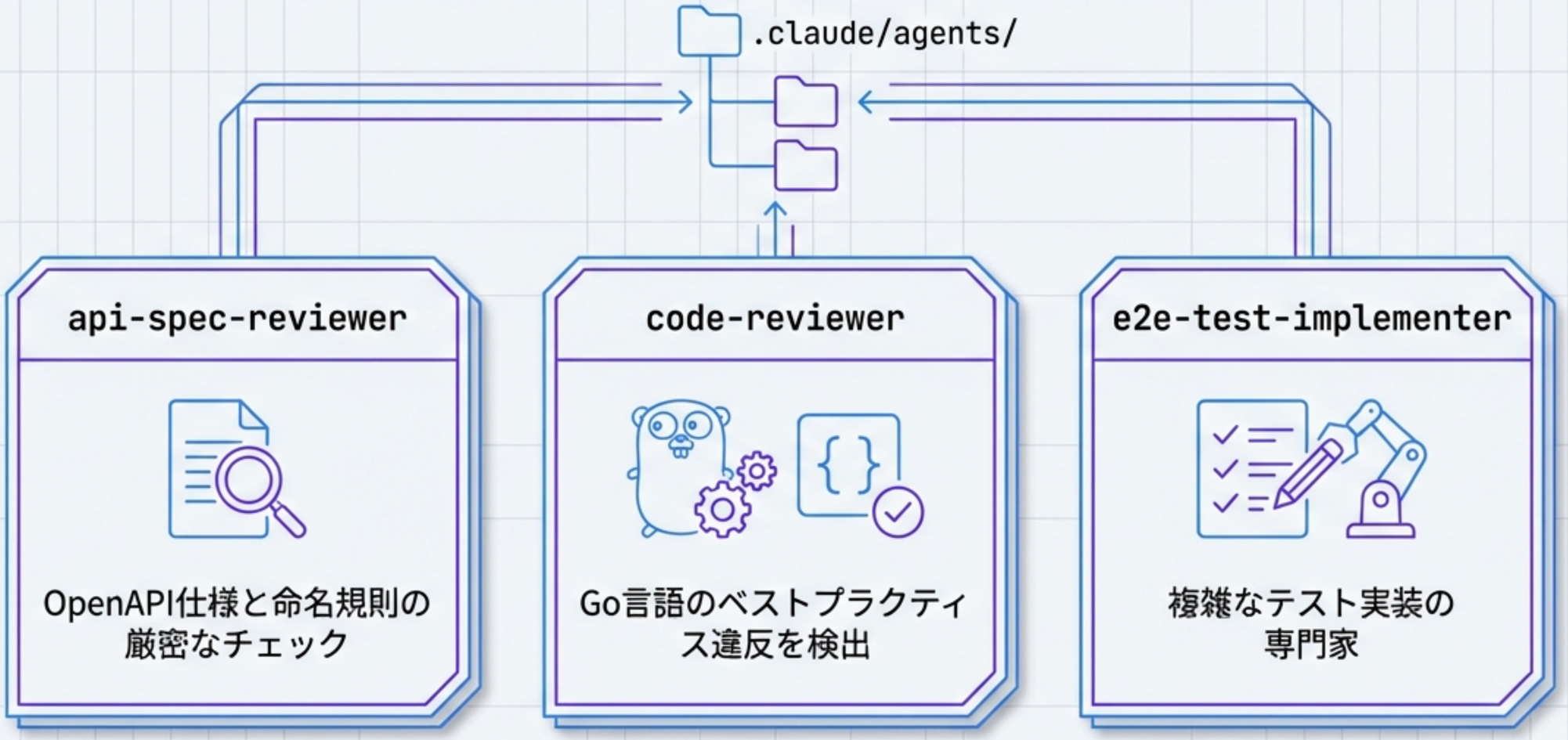
プロンプトメモリ (Prompt Memory)
プロジェクト概要や手順を定義

共通コンテキスト (Common Context)
技術スタックや用語の共有知識

知識ベース (Knowledge Base)
コーディング規約、APIガイドライン

更新すれば、AIの出力品質が即座に向上する仕組み

特化型サブエージェントによる分業

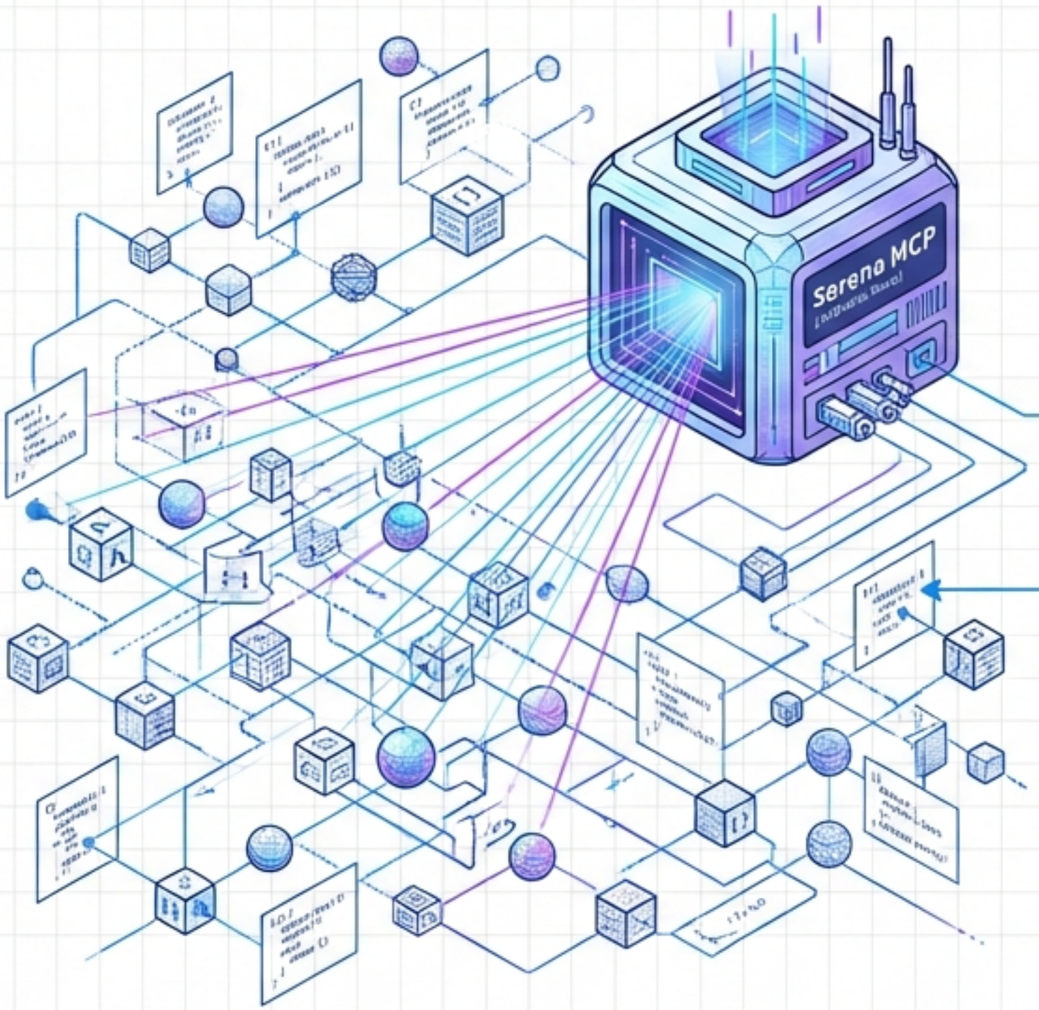


プロンプトをバージョン管理し、全員が同じ「賢いAI」を使える

大規模コードベースを「見る」目：Serena MCP

The Problem

モノレポ規模では単純検索が機能しない



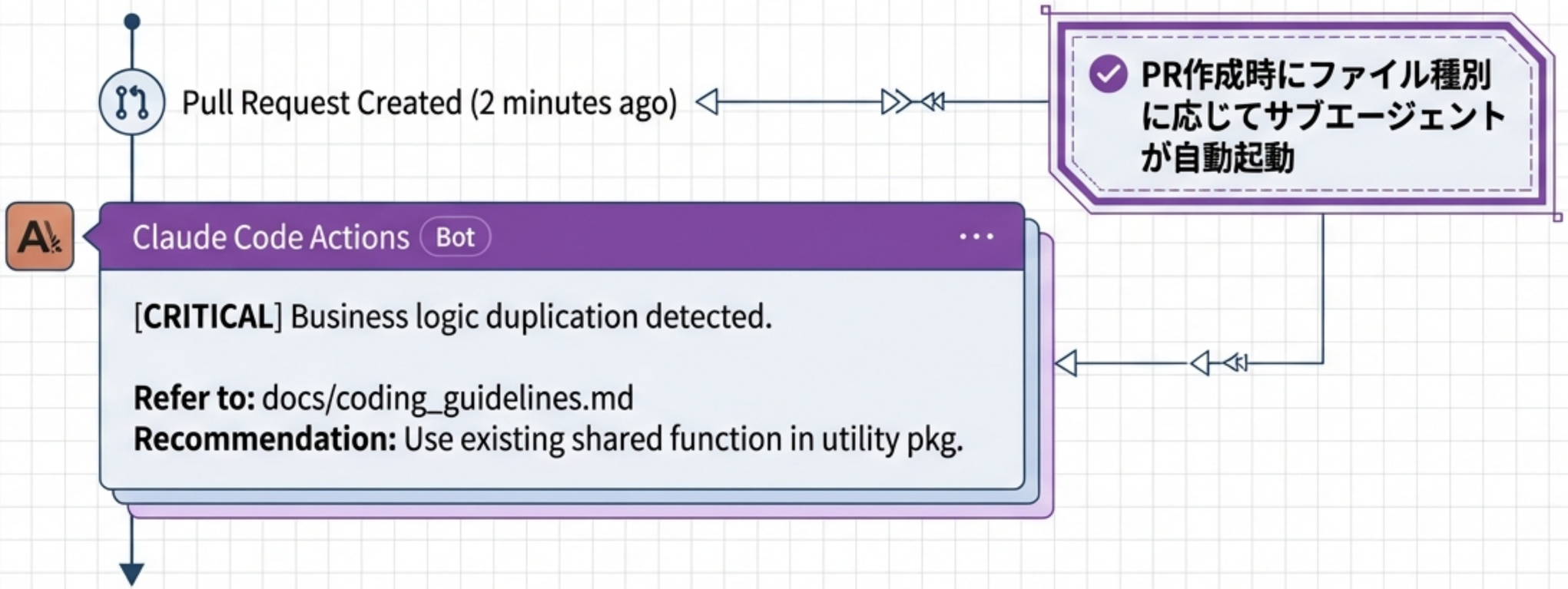
The Solution

LSP (Language Server Protocol) Based Structure Server

- **シンボル（関数・クラス）の抽出**
- **「この関数はどこ？」に即答**
- **正確なコンテキスト把握により、 ハルシネーションを低減 🧠✔️**

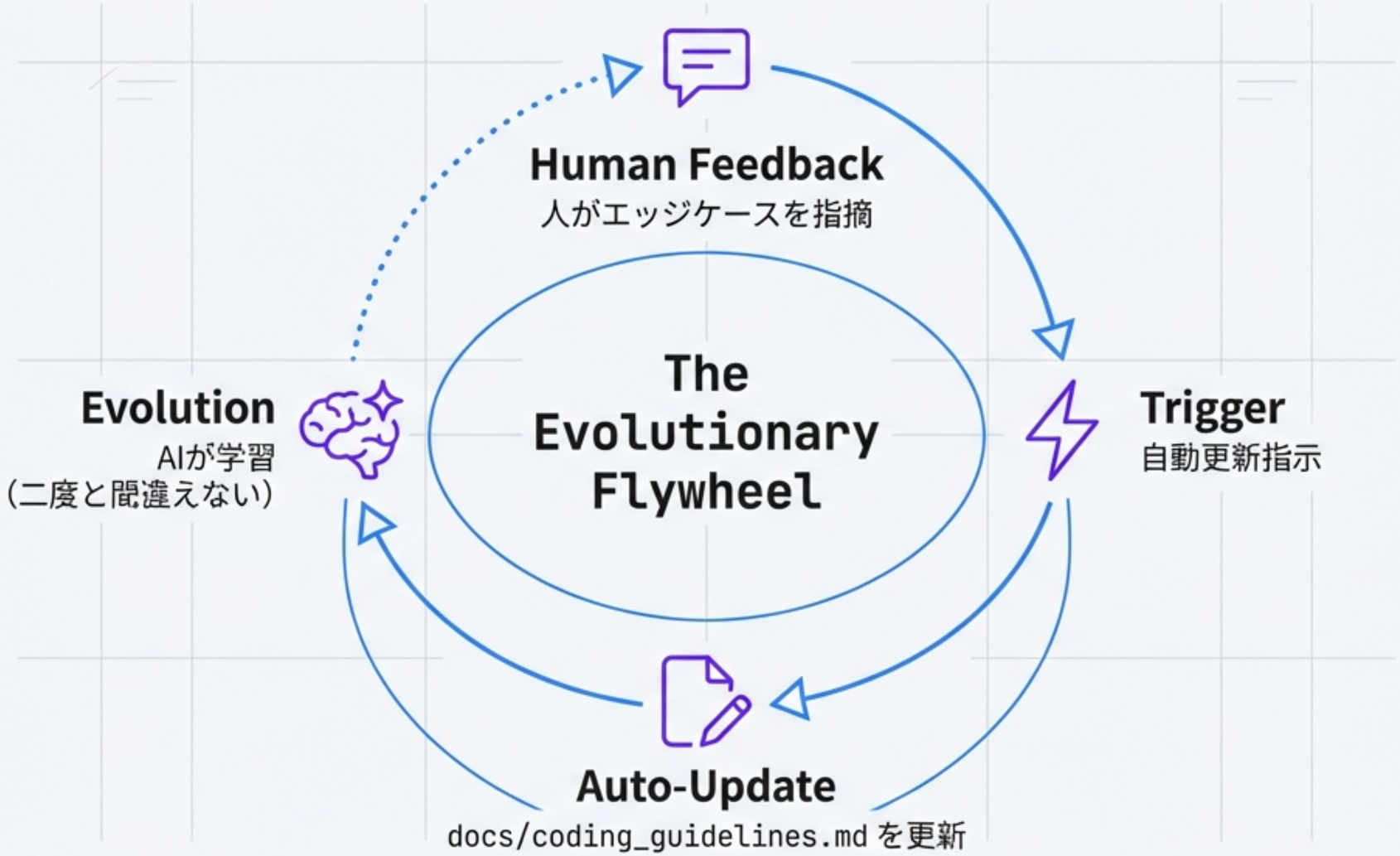
Workflow Phase 1: 実装 (Implementation)

Workflow Phase 2: 自動レビュー (Automated Review)

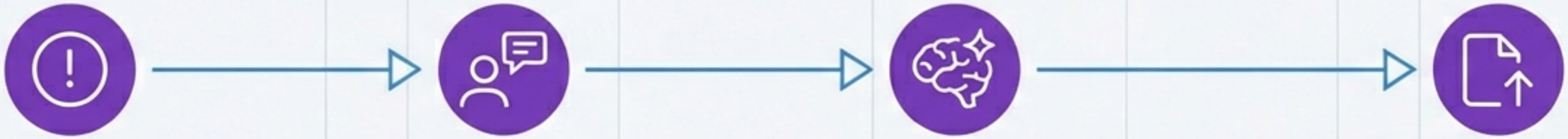


人間は「本質的な設計議論」に集中できる

Workflow Phase 3: ナレッジ循環 (Knowledge Circulation)



実例：一度の指摘を、永遠の資産へ



Issue Found

API spec missing for edge case.

Human Action

Reviewer comments on PR.

System Reaction

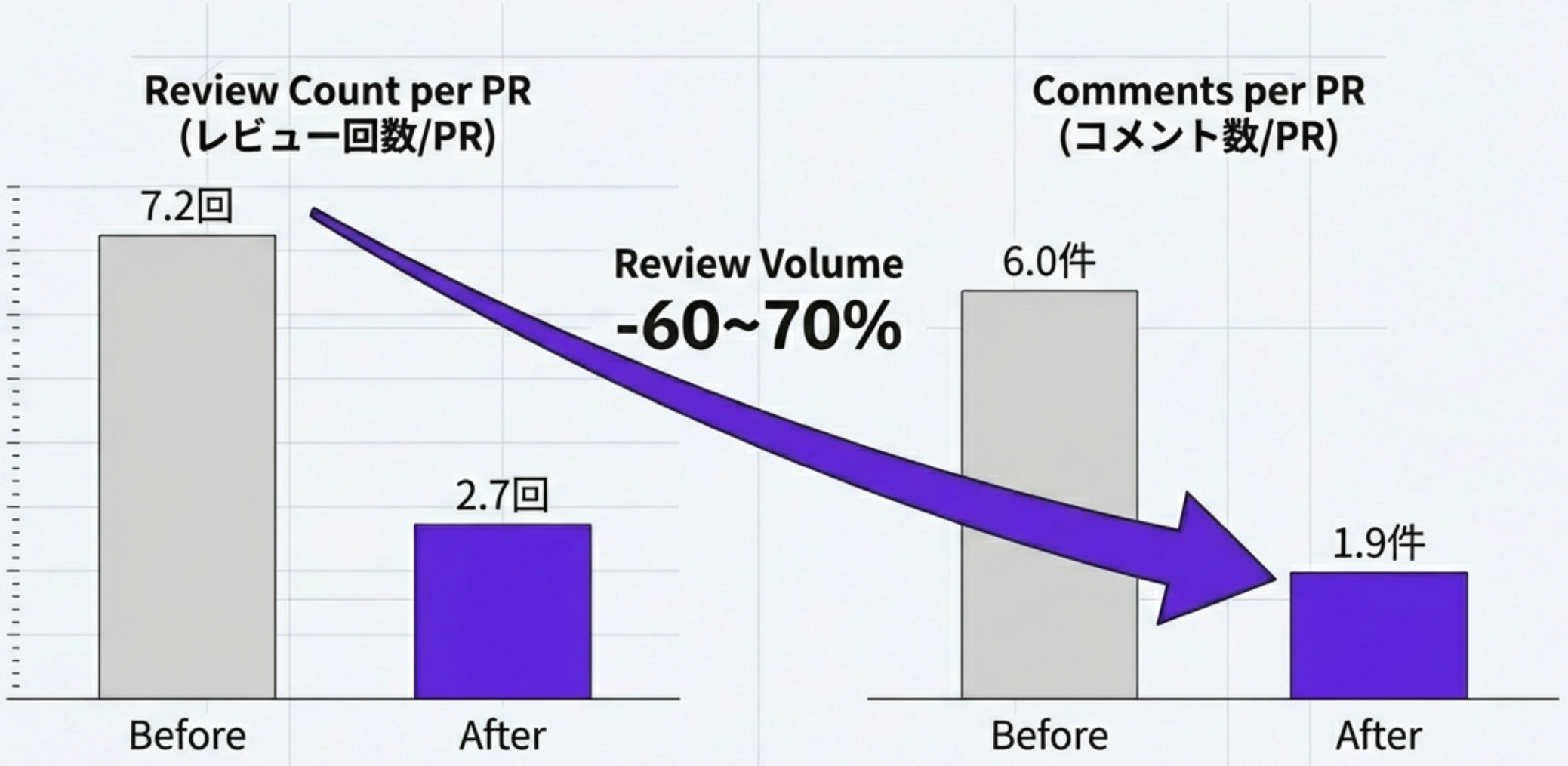
AI generates NEW PR to update `docs/api_development_guideline.md`

Outcome

New members & AI inherit this knowledge instantly.

「使い捨て」ではなく「育てる」運用

導入効果：圧倒的なノイズ低減



チーム開発の質的変化

品質の標準化

経験の浅いメンバーでも、
ガイドライン準拠のコードを
最初から書ける

コンテキストの一元管理

AI設定＝ドキュメント。
最新の仕様が常にリポジトリ
にある

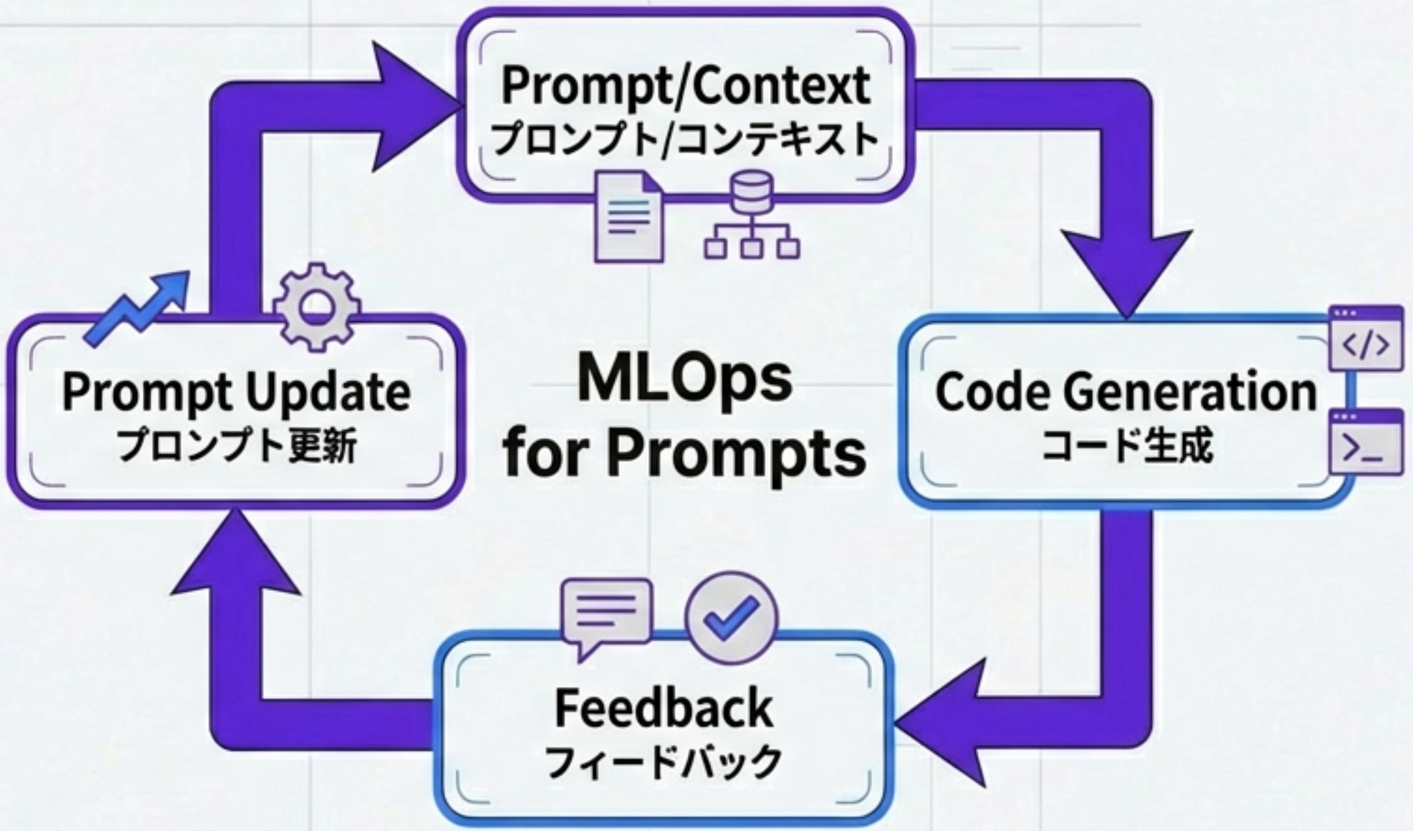
クリエイティブへの回帰

機械的な指摘はAIに任せ、
人間はアーキテクチャとUXに
注力

プロンプトエンジニアリングの資産化

MLOps for Code Generation

モデルの再学習ではなく、
コンテキスト（Docs）と
プロンプト（Agents）を
管理・改善する。



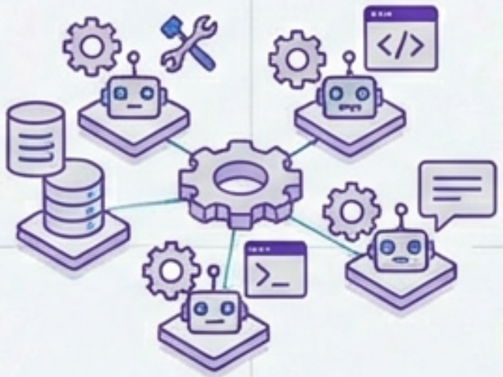
組織の知的資産としてAIを育成する

Conclusion

AIは「使う」ものではなく、チームで「育てる」もの。

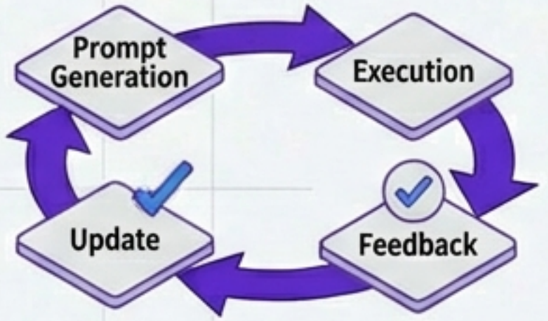
Context:
ナレッジのリポジトリ管理

コンテキストをドキュメントとして
一元管理し、常に最新の仕様を共有。



Specialization:
サブエージェントによる分業

専門的なタスクをサブエージェントに
分担させ、効率的な分業体制を構築。



Circulation:
フィードバックループの自動化

継続的なフィードバックと改善の
サイクルを自動化し、AIを育成。

Start raising your infrastructure today.