claude-code-controller

Claude Codeの真の力を解き放つ

CLIをAPI化し、月額サブスクリプションで
「AIスワーム」を構築する技術ガイド

Based on ths work by
The Vibe Company (Stan Girard)

エージェント開発の「API税」問題

AIエージェントは未来ですが、公式SDKでの開発はコストが青天井です。

Official Agent SDK (API Route)

- Pay-per-token（従量課金）
- Cost Impact: ~$200/day

> "I was burning $200/day on agent API calls." — Stan Girard

Claude Code CLI (Subscription Route)

- Flat rate（定額制 - Max Plan）
- Cost Impact: ~$200/month
- Limit: Designed for human speed

疑問：なぜ、月額定額のCLIをプログラムから制御できないのか？

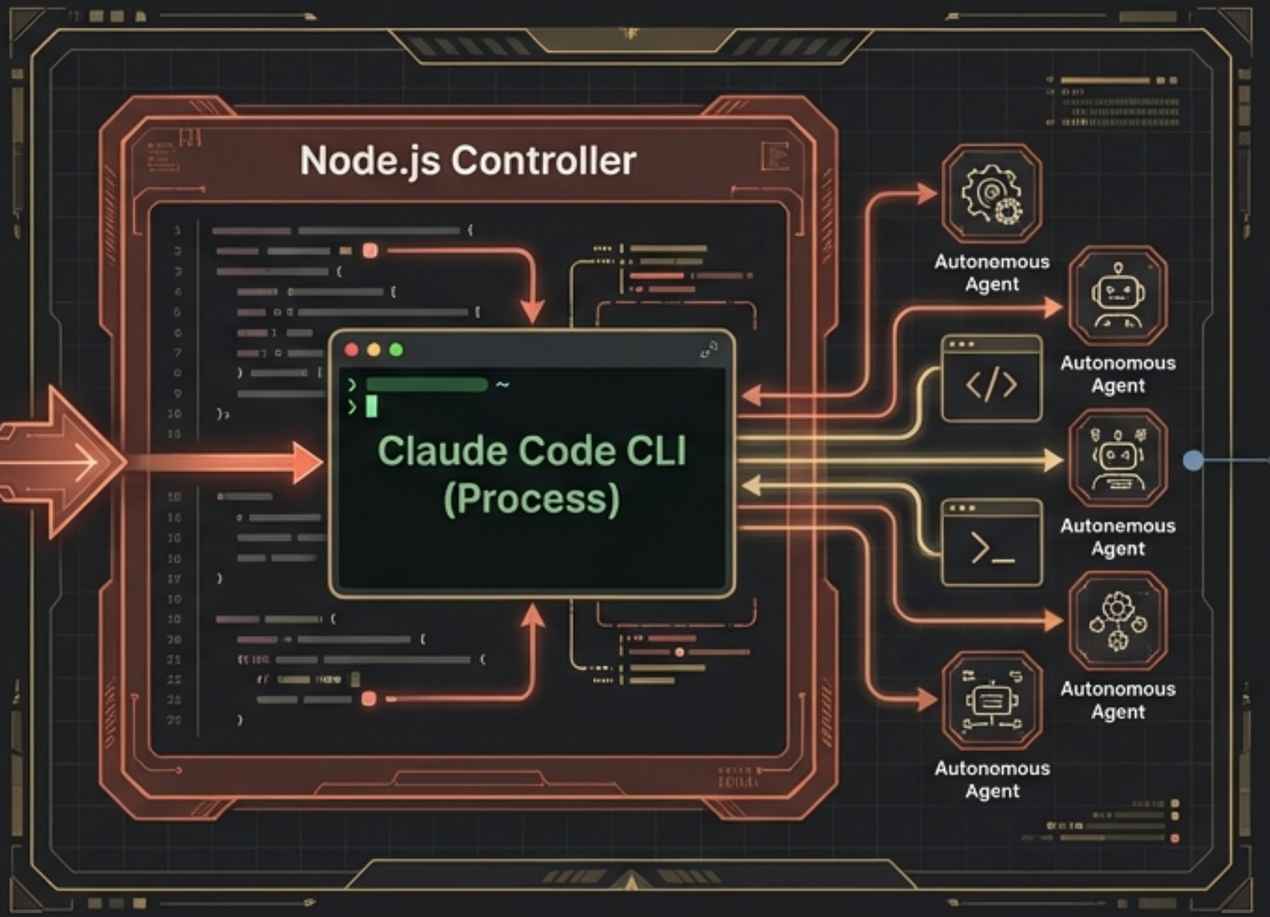
ブレイクスルー：CLIを「操り人形」にする

AIエージェントは未来ですが、公式SDKでの開発はコストが青天井です。

Anthropic APIのラッパーではありません。ローカルで動作する「本物のClaude Code CLI」をプログラムで制御する制御する技術です。

Input (TypeScript/REST)

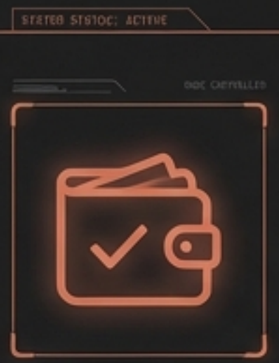
Not a wrapper for the Anthropic API. It's a technique to programmatically control the "real Claude Code CLI" running locally.



Pay for the subscription, run the swarm for free.

(サブスクリプションを支払えば、スワームは無料で動く)

なぜ、このアプローチがルールを変えるのか



Economic (コスト革命)

既存のMaxプランを使用。APIの従量課金や予期せぬ請求の恐怖から解放されます。

Day 0 Features (即時性)

Claude CLI本体がアップデートされれば、新しいモデルやツールが即座に使えます。SDKの対応待ち時間はゼロです。

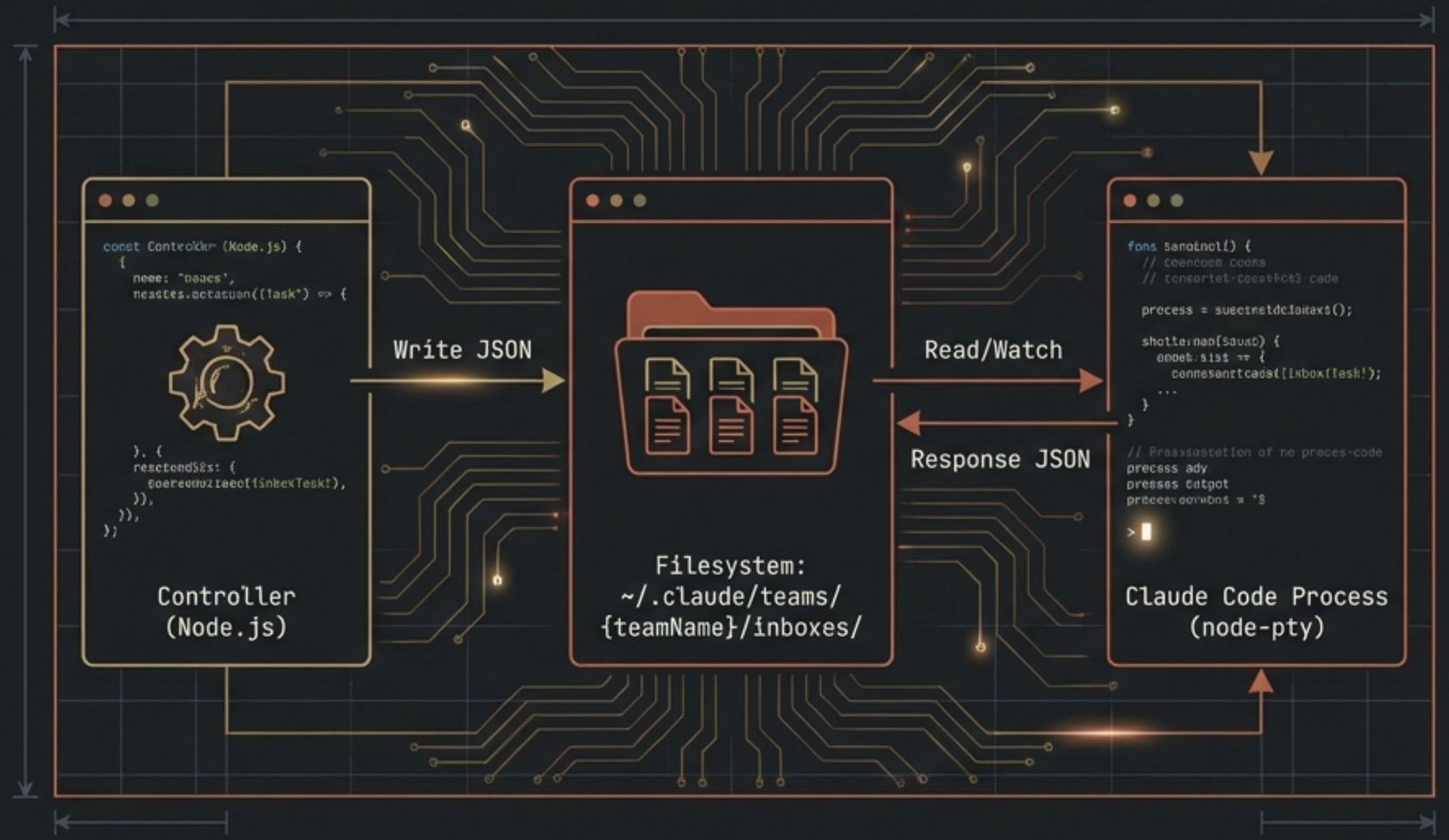
Real Capabilities (フル権限)

サンドボックス化された環境ではなく、実際のターミナル環境(Bash, Fllesystem, Git)へのフルアクセス権を持ちます。

Swarm Ready (スワーム)

複数のエージェントを並列起動し、オーケストレーションが可能です。

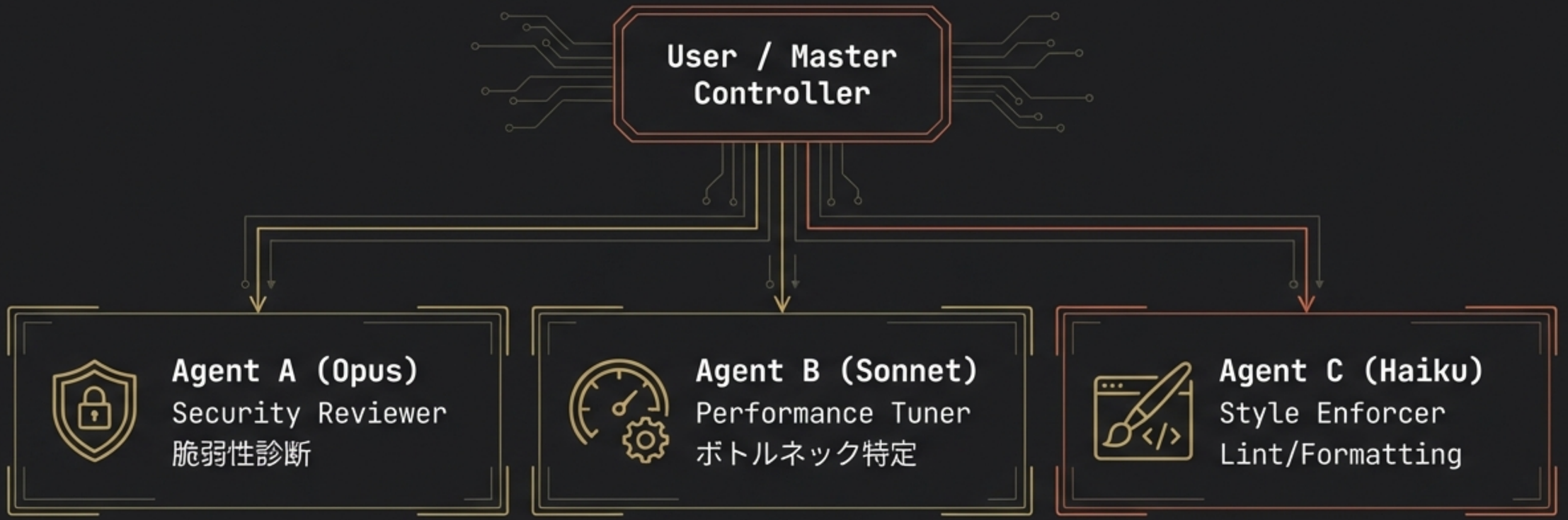
Under the Hood：内部プロトコルのリバースエンジニアリング



- **Protocol:** teams/inbox/tasks ファイルシステムプロトコルの解析に基づいています。
- **Communication:** コントローラーはJSONファイルを書き込み、node-ptyを経由してCLIプロセスと通信します。
- **Concurrency:** proper-lockfileを使用して、ファイル破損を防ぎつつ排他制御を行います。

コンセプト：AIスワーム（群知能）の構築

「ペアプログラミング（1対1）」から「エンジニアリングマネジメント（1対多）」へのシフト



人間が待機している間に、3つの異なる視点からのレビューが同時に完了します。

claude-code-controller 機能概要

- **REST API** — curl や他言語からエージェントをHTTP経由で制御可能。
- **TypeScript SDK** — 型安全性を持ったプログラム制御（`ctrl.spawnAgent`, `agent.ask`）。
- **Web Dashboard** — リアルタイムのログ監視、承認フロー、プロセス管理UI。
- **Task Management** — タスクの作成、割り当て、依存関係の定義（`blockedBy`）。
- **Intervention** — 人間による承認（`Approval`）や緊急停止（`Kill`）の制御。

Quick Start：30秒で最初のエージェントを召喚する

Prerequisites: Claude Code CLI v2.1.34+ / Node.js or Bun

code-example.js

```
import { ClaudeCodeController } from 'claude-code-controller';

// 1. コントローラーの初期化
const ctrl = new ClaudeCodeController({ teamName: 'my-project' });
await ctrl.init();

// 2. エージェントの召喚（Sonnetモデル）
const agent = await ctrl.spawnAgent({
  name: 'coder',
  model: 'sonnet',
});

// 3. タスクの実行
const answer = await agent.ask(
  'package.jsonを読み、プロジェクト名を教えて。',
  { timeout: 60_000 }
);
console.log(answer);
```

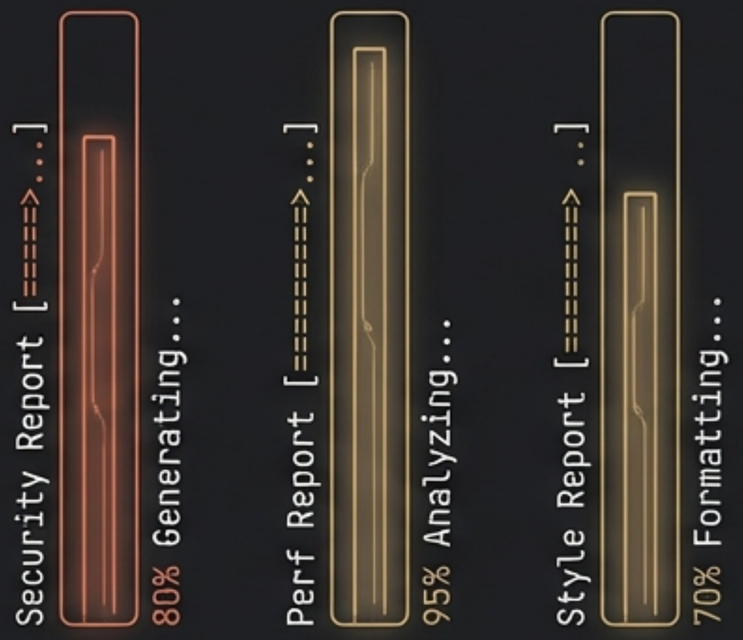
Code in Action：並列コードレビューの実装

1つのプルリクエストに対して、3つの専門エージェントを同時に走らせる。

JS code-example.js

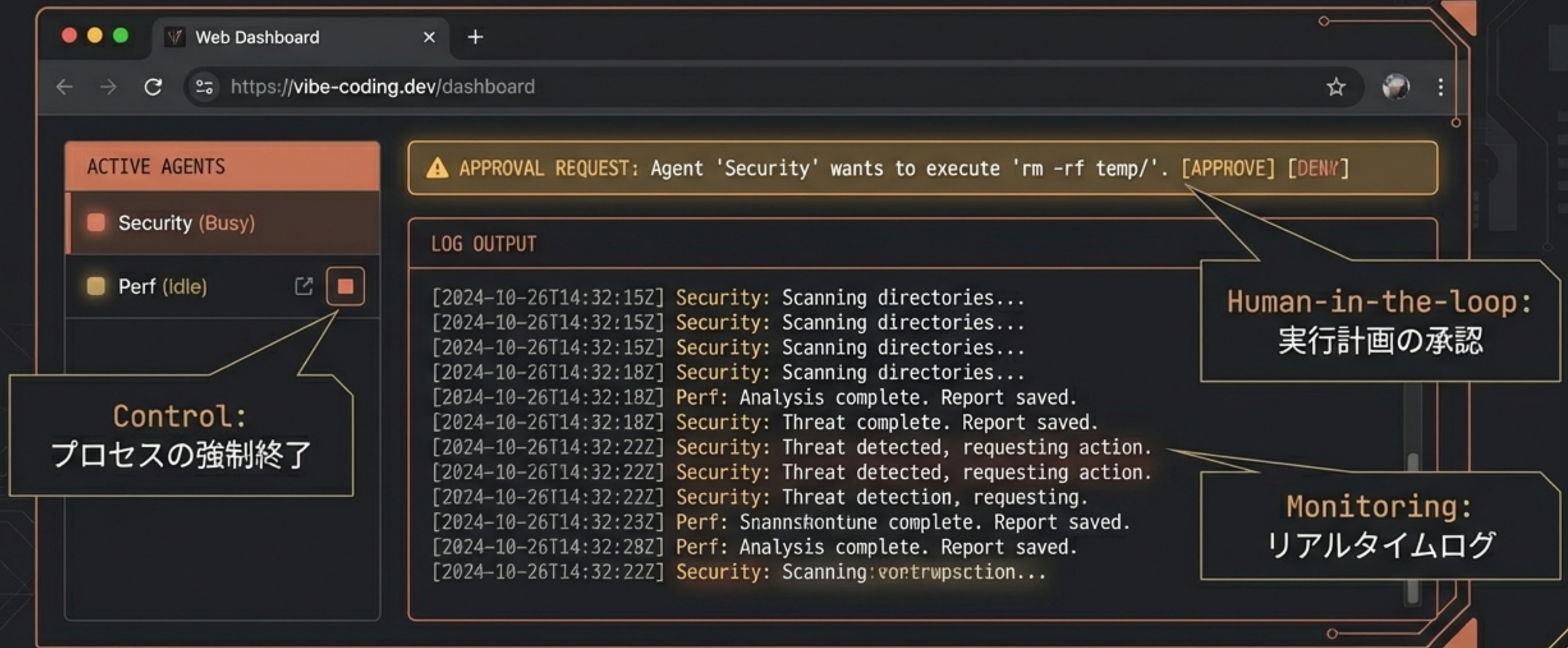
```
// 3体のエージェントを並列起動
const [security, perf, style] = await Promise.all([
  ctrl.spawnAgent({ name: 'security', model: 'opus' }),
  ctrl.spawnAgent({ name: 'perf', model: 'sonnet' }),
  ctrl.spawnAgent({ name: 'style', model: 'haiku' }),
]);

// レビュータスクを同時配信
const reviews = await Promise.all([
  security.ask('src/ のセキュリティ脆弱性をレビューして'),
  perf.ask('src/ のパフォーマンス課題を特定して'),
  style.ask('コードスタイルと命名規則をチェックして'),
]);
```

TERMINAL OUTPUT



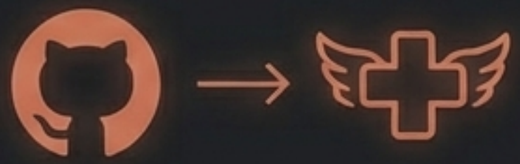
Web Dashboardと「Vibe Coding」

PCの前から離れ、外出先やスマホから自宅の強力なマシン上のエージェントを指揮する



単なるコーディングを超えて：自動化のユースケース

Scenario A: CI/CD Healer

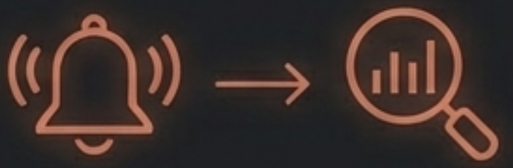


GitHub Actionsでテスト失敗 -> Webhook -> エージェントが起動し修正PRを作成。

Scenario B: The Night Shift (夜間バッチ)

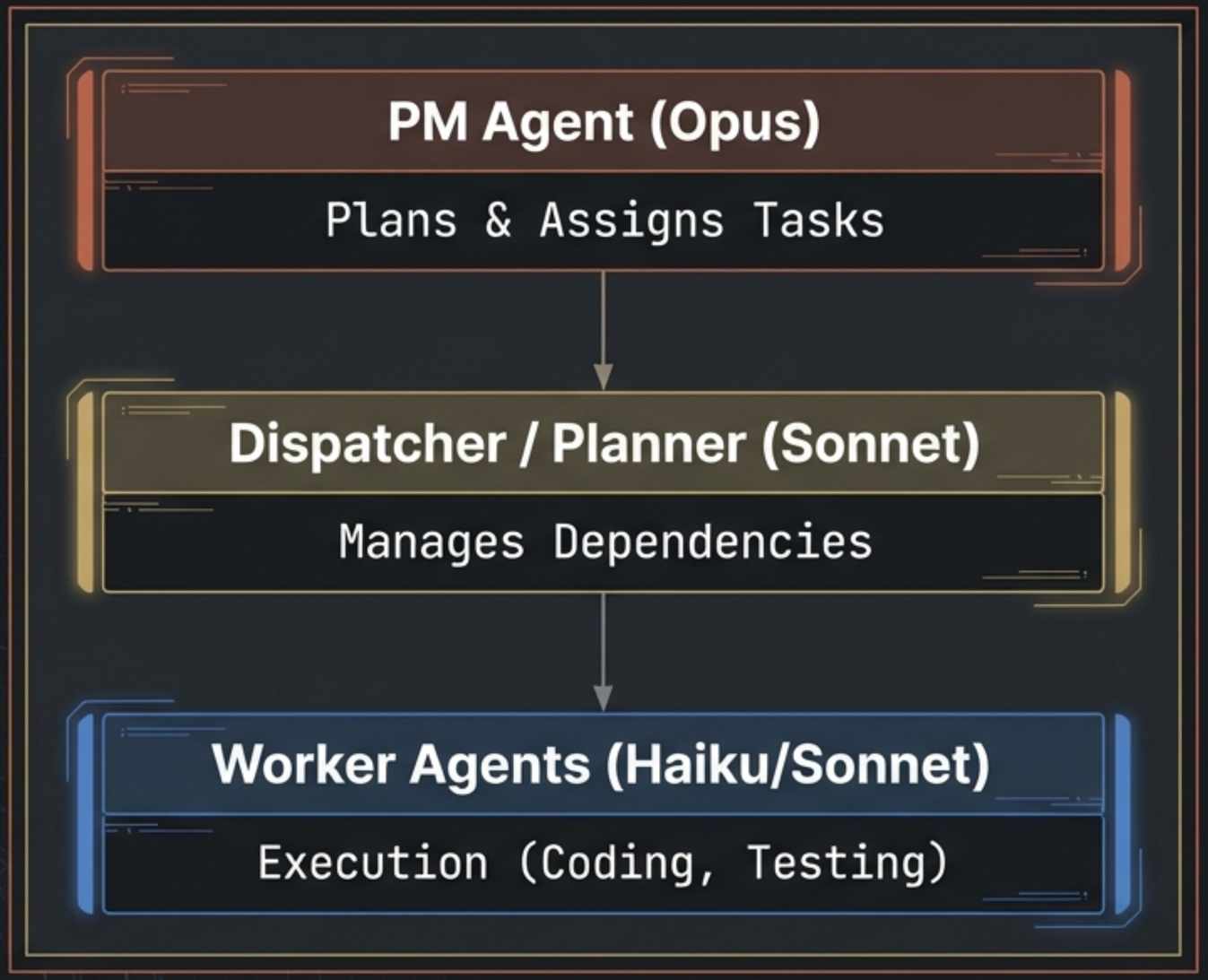
Cronジョブで毎晩起動 -> 依存関係の更新、レガシーコードのリファクタリング -> 朝。 朝にはPRが完成。

Scenario C: Incident Response



PagerDutyアラート -> ログ分析エージェントが起動 -> 原因の一次切り分けと修正案の提示。

AIマネジメント：Opusによる「PMエージェント」



Workflow Overview:

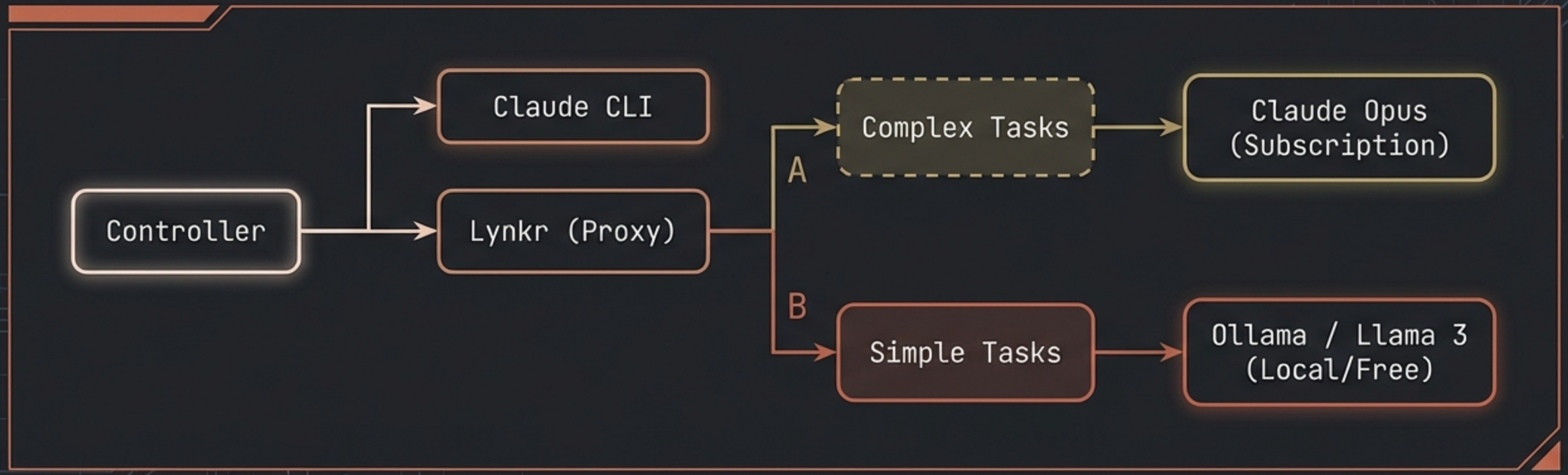
1. PM: 要件定義書(PRD)を読み込み、タスクを分解

2. Dispatcher: 依存関係に基づき、適切なエージェントに割り当て

3. Review: 完了したタスクをPMが検証

ハイブリッド運用：Local LLMとの融合

単純作業はローカルで、複雑な設計はClaudeで



プロキシツール（Lynkr）を介して、タスクの難易度に応じてモデルを使い分けることで、サブスクリプションの制限リスクをさらに低減します。

リスクと考慮事項（Fine Print）

Stability

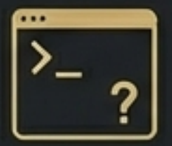
非公式の内部プロトコルに依存しているため、CLIのアップデートで破損する可能性があります（バージョン固定を推奨）。

Account Safety

極端な並列実行（DDoS的な使用）はアカウント制限のリスクがあります。
常識的な範囲での利用を。

Security

エージェントに Bash や Write 権限を与える際は、必ずガードレール（承認フロー）を設定してください。



「AIを使う」から「AIを率いる」時代へ

あなたの月額$200のサブスクリプションは、もはや単なるチャットボットではありません。
それは24時間365日稼働する、専門エンジニアのチーム です。

Repository: The-Vibe-Company/claude-code-controller
Star the project on GitHub.

'Vibe Coding'を始めましょう。ただし、利用は計画的に。