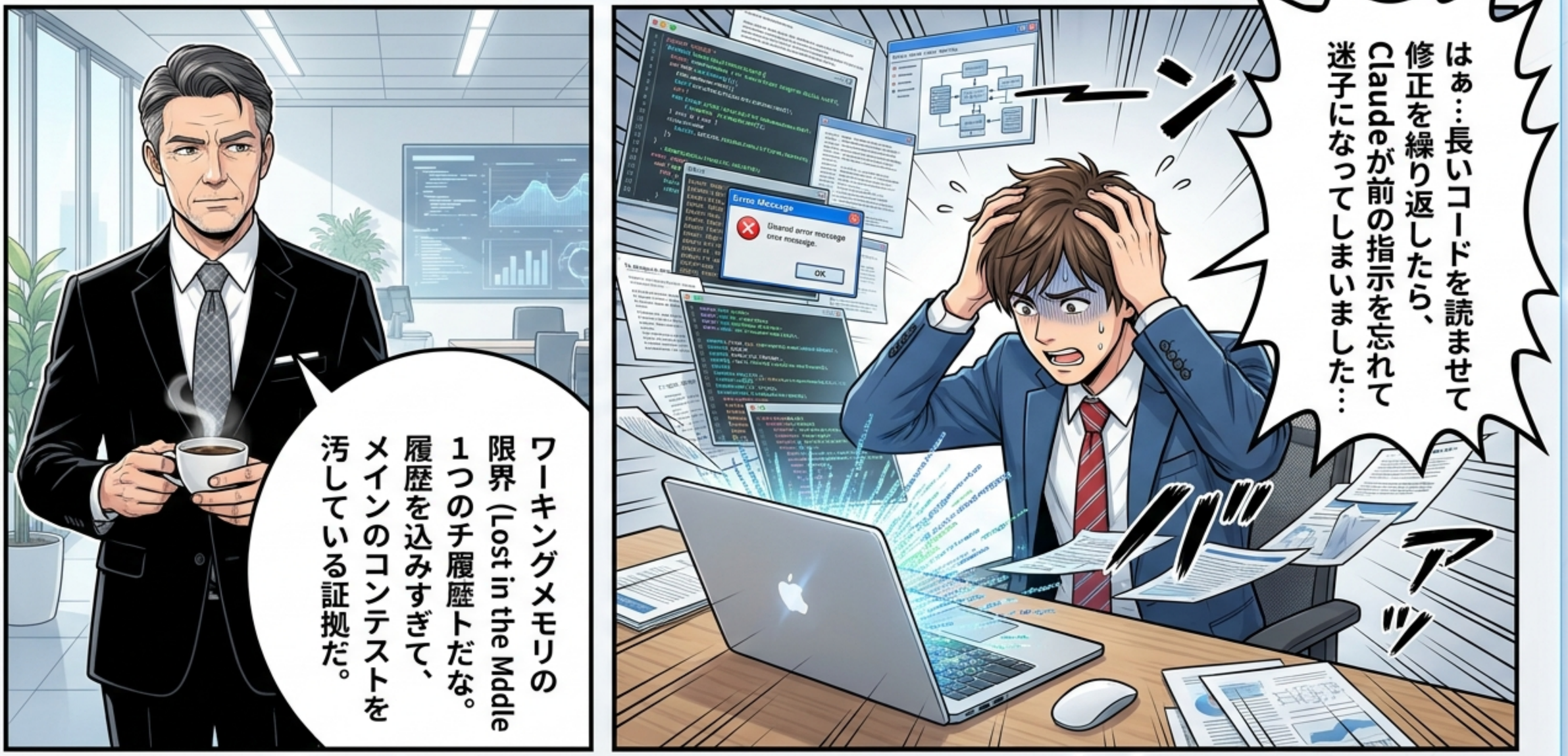
【完全ガイド】Claudeサブエージェント実践手順書

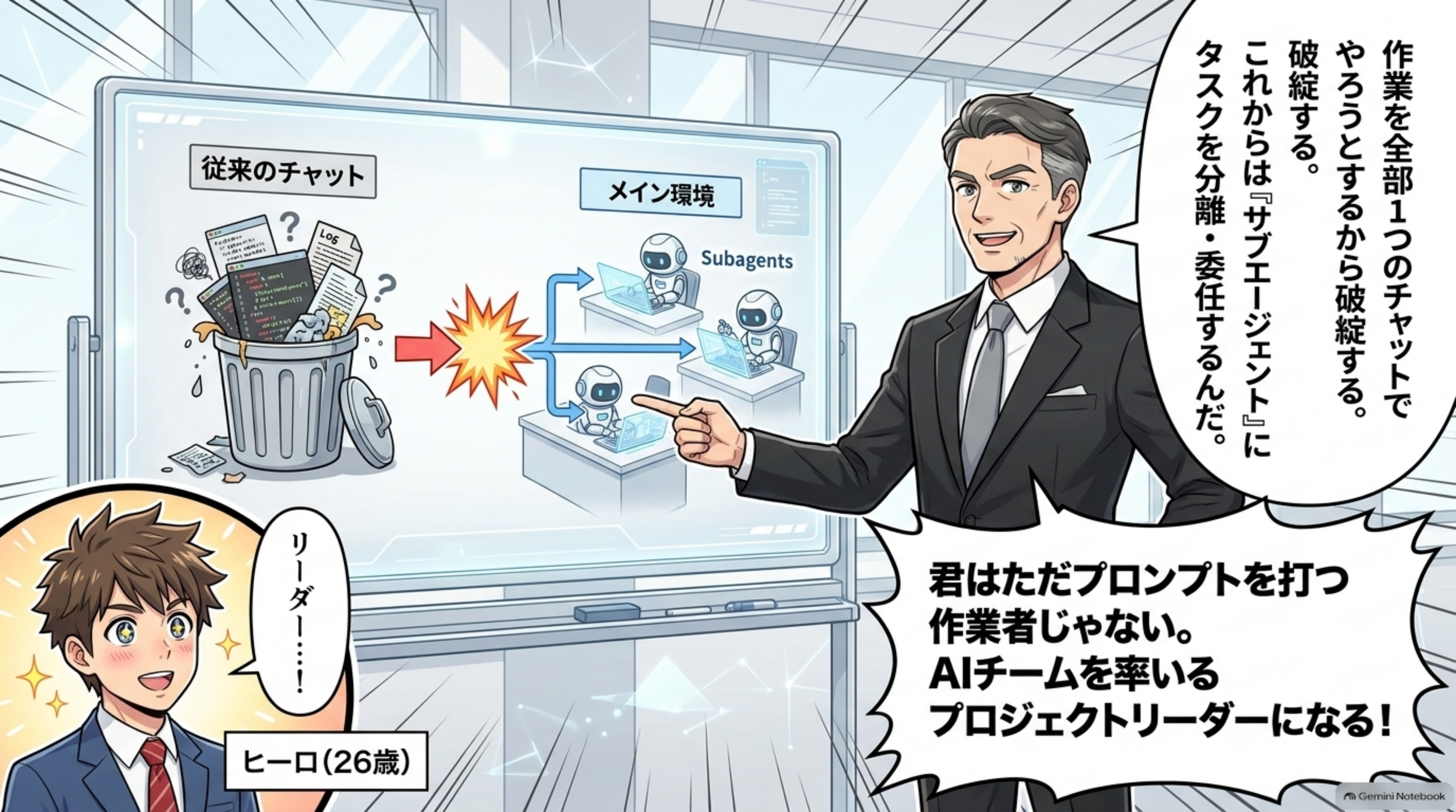
～AIをチームとして率いる新常識～



Gemini Notebook



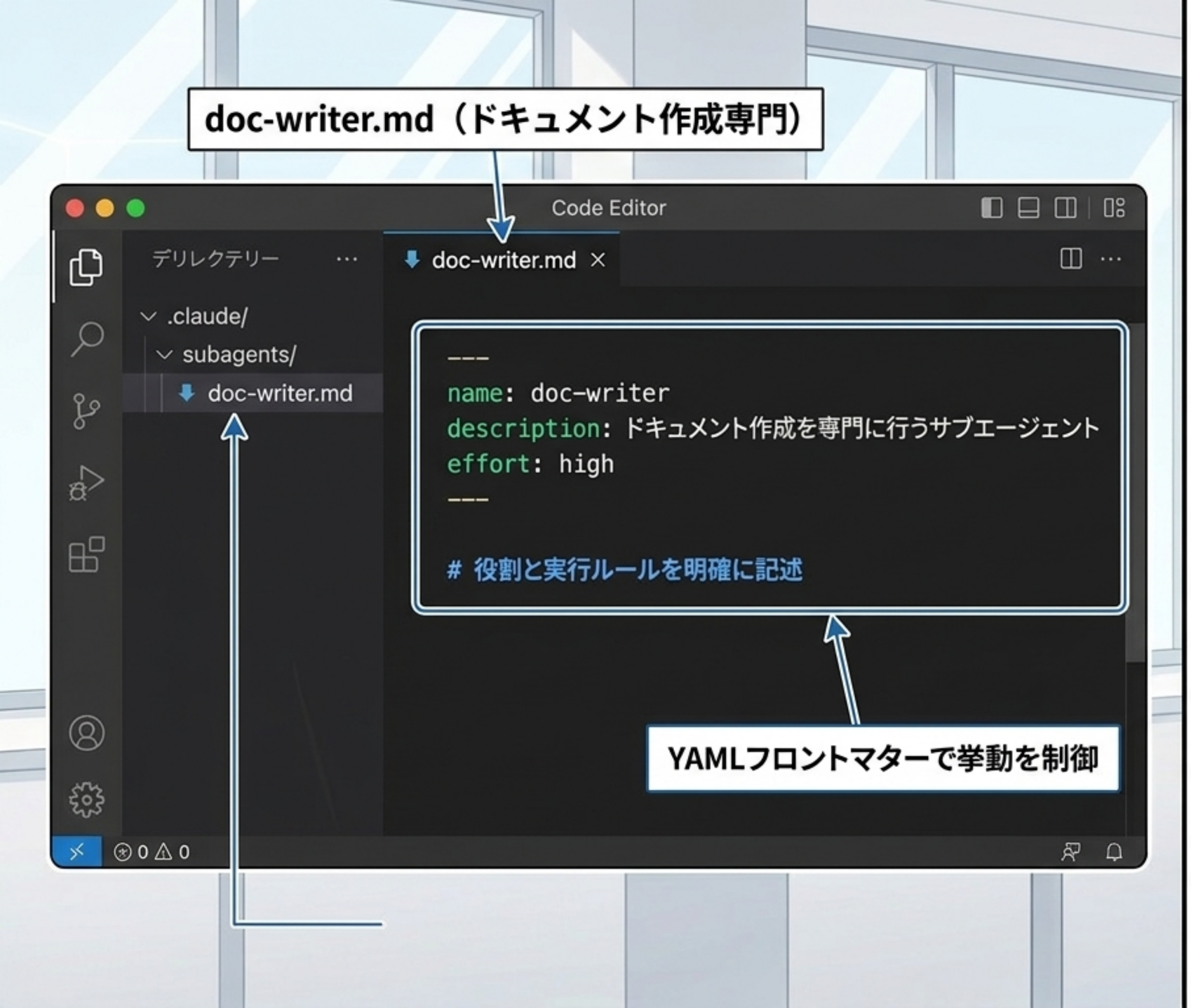
Gemini Notebook


従来のチャット
メイン環境
Subagents
LOG
作業を全部1つのチャットでやろうとするから破綻する。
破綻する。
これからは『サブエージェント』にタスクを分離・委任するんだ。
君はただプロンプトを打つ作業者じゃない。
AIチーム を率いるプロジェクトリーダーになる！
リーダー…！
ヒーロ（26歳）
Gemini Notebook

評価項目	従来の一体型チャット	サブエージェント活用
コンテキスト管理	セッションが肥大化・圧迫	独立環境でメインは常にクリーン（コンテキスト衛生）
タスクの分離	設計・調査・コードが混在	役割ごとに専用エージェントを定義
作業の進行	1ターンずつ待つ直列処理	バックグラウンド処理・並行実行が可能
成果物の管理	ログの山に成果物が埋もれる	まとまった最終成果物のみを返す



Gemini Notebook


doc-writer.md（ドキュメント作成専門）
Code Editor
デリレクテリー
doc-writer.md
.claude/
subagents/
doc-writer.md
---
name: doc-writer
description: ドキュメント作成を専門に行うサブエージェント
effort: high

役割と実行ルールを明確に記述
YAMLフロントマターで挙動を制御


まずは
.claude/subagents/ に
にMarkdownファイルを作る。
ここがエージェントの控室だ。
Gemini Notebook

Terminal

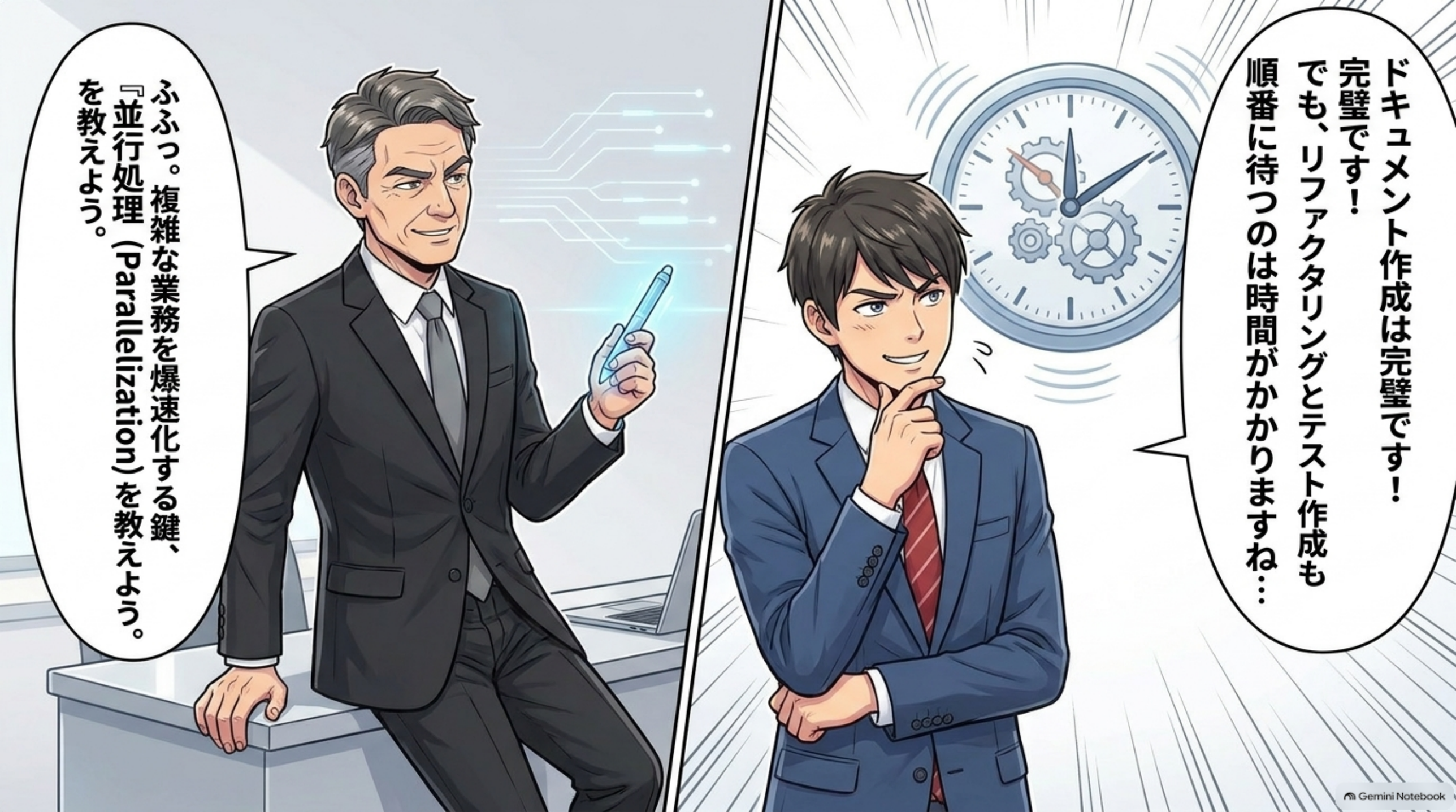
```
> claude "doc-writer サブエージェントを使用して
src/utils.py の仕様書を作成してください"
```



注意しろ！
動かない時は配置パスのミスか、
YAML の文法エラーが多い。

指示出しの際は、『コード変更はしない』等、
境界条件（制約プロンプト）を明確にな！

Gemini Notebook


ドキュメント作成は完璧です！
完璧です！
でも、リファクタリングとテスト作成も
順番に待つのは時間がかかりますね…
ふふっ。複雑な業務を爆速化する鍵、
『並行処理（Parallelization）』を教えよう。
を教えよう。
Gemini Notebook

並列処理 (Parallel Processing)

互いに独立したタスクは、専用のエージェントに同時に処理させる。これで作業時間は大幅に大幅に短縮される。

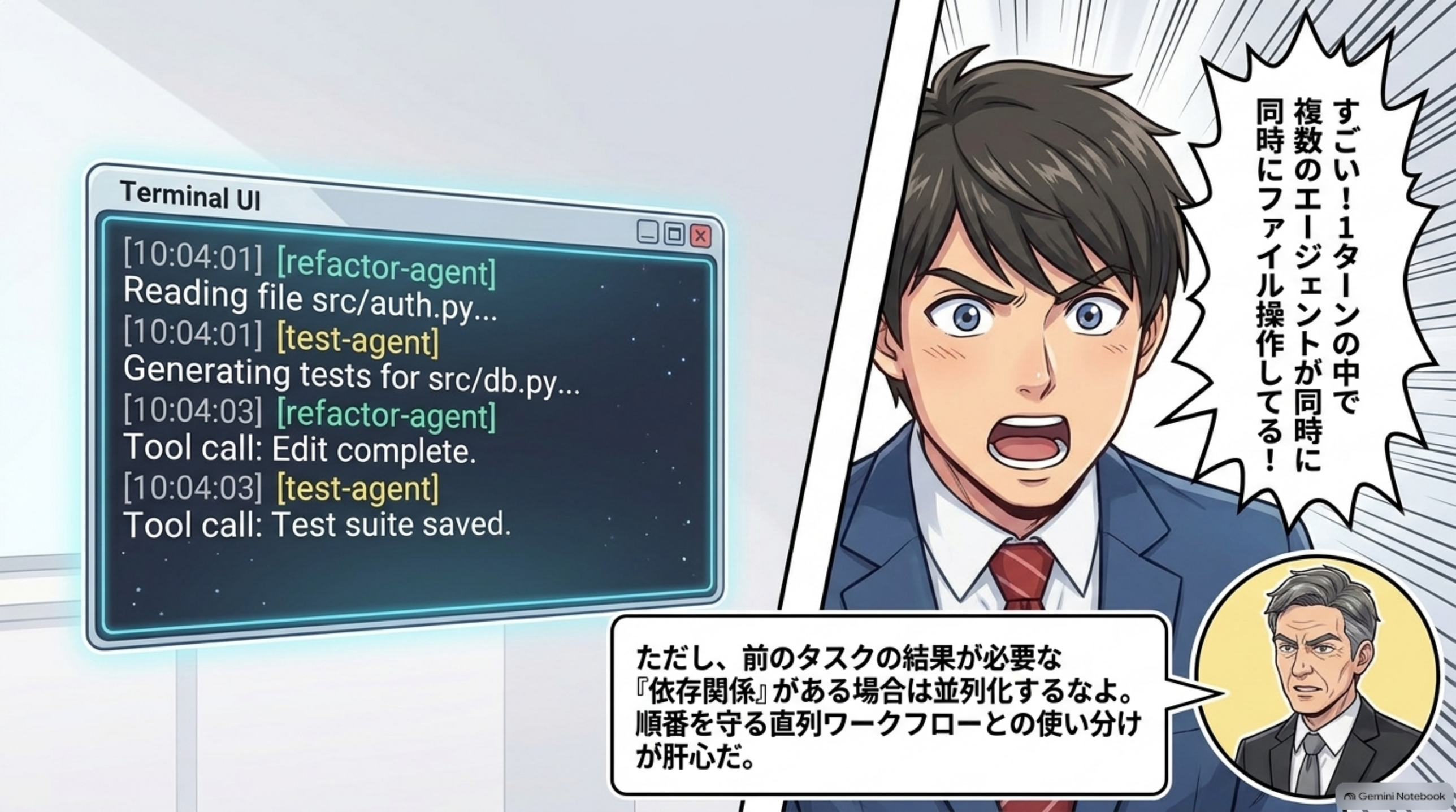
直列 - Serial



並列 - Parallel



Gemini Notebook


Terminal UI
[10:04:01] [refactor-agent]
Reading file src/auth.py...
[10:04:01] [test-agent]
Generating tests for src/db.py...
[10:04:03] [refactor-agent]
Tool call: Edit complete.
[10:04:03] [test-agent]
Tool call: Test suite saved.
すごい！1ターンの中で
複数のエージェントが同時に
同時にファイル操作してる！
ただし、前のタスクの結果が必要な
『依存関係』がある場合は並列化するなよ。
順番を守る直列ワークフローとの使い分け
が肝心だ。
Gemini Notebook


早くて便利ですが、
簡単な作業に無駄に
無駄に考え込んで
時間がかかる時が
があります…
そこで
『エフォート
(Effort)レベル』の
調整だ。
これはAIの『知能』ではなく、
『どれだけ懸命に働くか』を
制御する重要なダイヤルだ。
Gemini Notebook

成果物が不十分なとき、
すぐにモデル（知能）を上げるのは素人だ。



【ステップ1】
まずエフォート（頑張り）
を上げて再実行。

【ステップ2】
それでもダメな時だけ
モデル（知能）を変更しろ。

Gemini Notebook

エフォートレベル	読み込み・思考	返答スピード	自己検証 (二重チェック)
low	必要最低限	素早い	ほとんど行わない
medium / auto	標準的 (バランス重視)	標準的	適度に行う
high / max	広範囲・深層思考	複数ターン試行	厳密な再検証 (非常に多い)



なるほど！定型文なら low でサクッと、
厳密なセキュリティチェックなら high で
徹底的に自己検証させるんですね！

Gemini Notebook

IMPLEMENTATION

```
claude --effort low
```

CLI起動時のフラグ

```
/effort high
```

セッション内コマンド



環境変数から
モデルデフォルトまで、
設定には優先順位
(Which setting wins)
があることを覚えておけ。

PRIORITY HIERARCHY



Gemini Notebook


分離、並列、そしてリソース制御…。
僕はもう、ただプロンプトを
打つだけの作業者じゃない。
AIチームのリーダーなんですね！
その通りだ。
下準備はチームに任せ、
創造的な意思決定という、
お前本来の仕事に集中しろ。




```
mkdir -p .claude/subagents
echo -e "---
  name: quick-reviewer
  description: 簡易レビューを行う
  effort: low
  ---
  改善点を3点以内で簡潔に指摘してください。" > .claude/subagents/quick-reviewer.md
claude "quick-reviewer を使って主要ファイルをレビューして"
```

小さな1歩が、業務を飛躍させるスタートラインになる。

Gemini Notebook