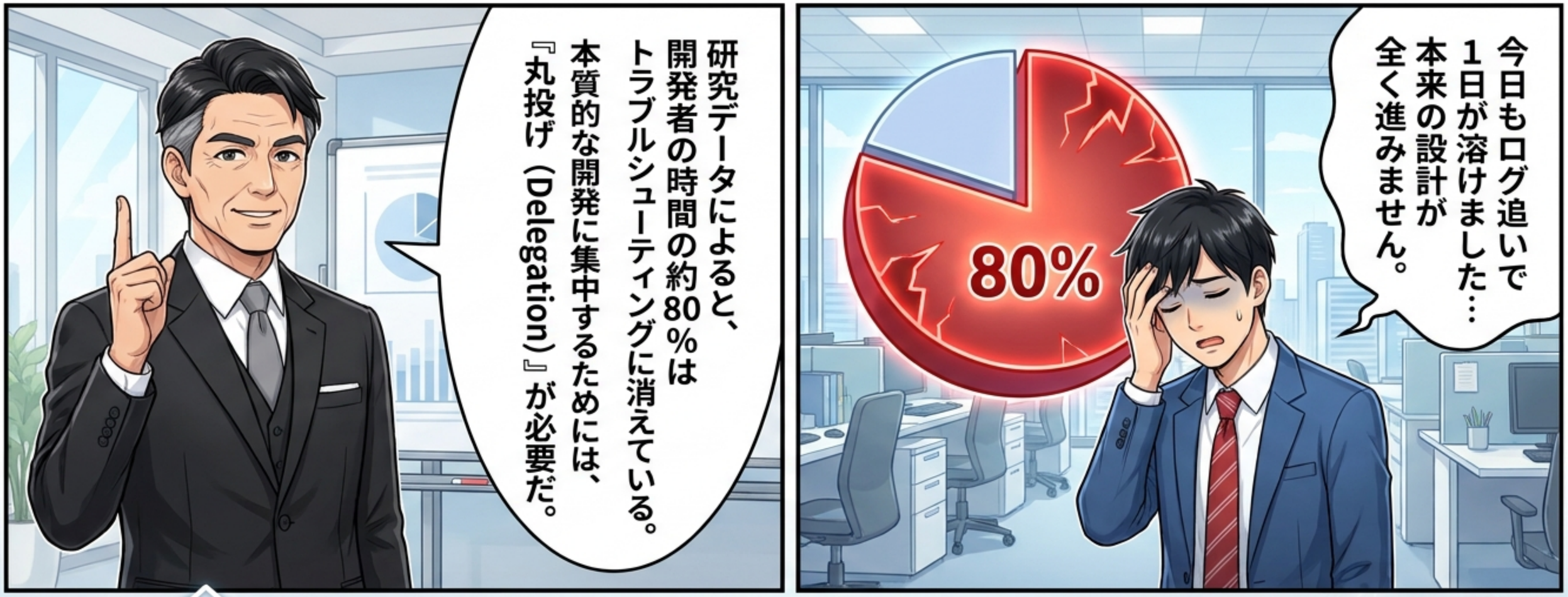
デバッグ地獄からの脱出！Claudeに任せる次世代トラブル解決術

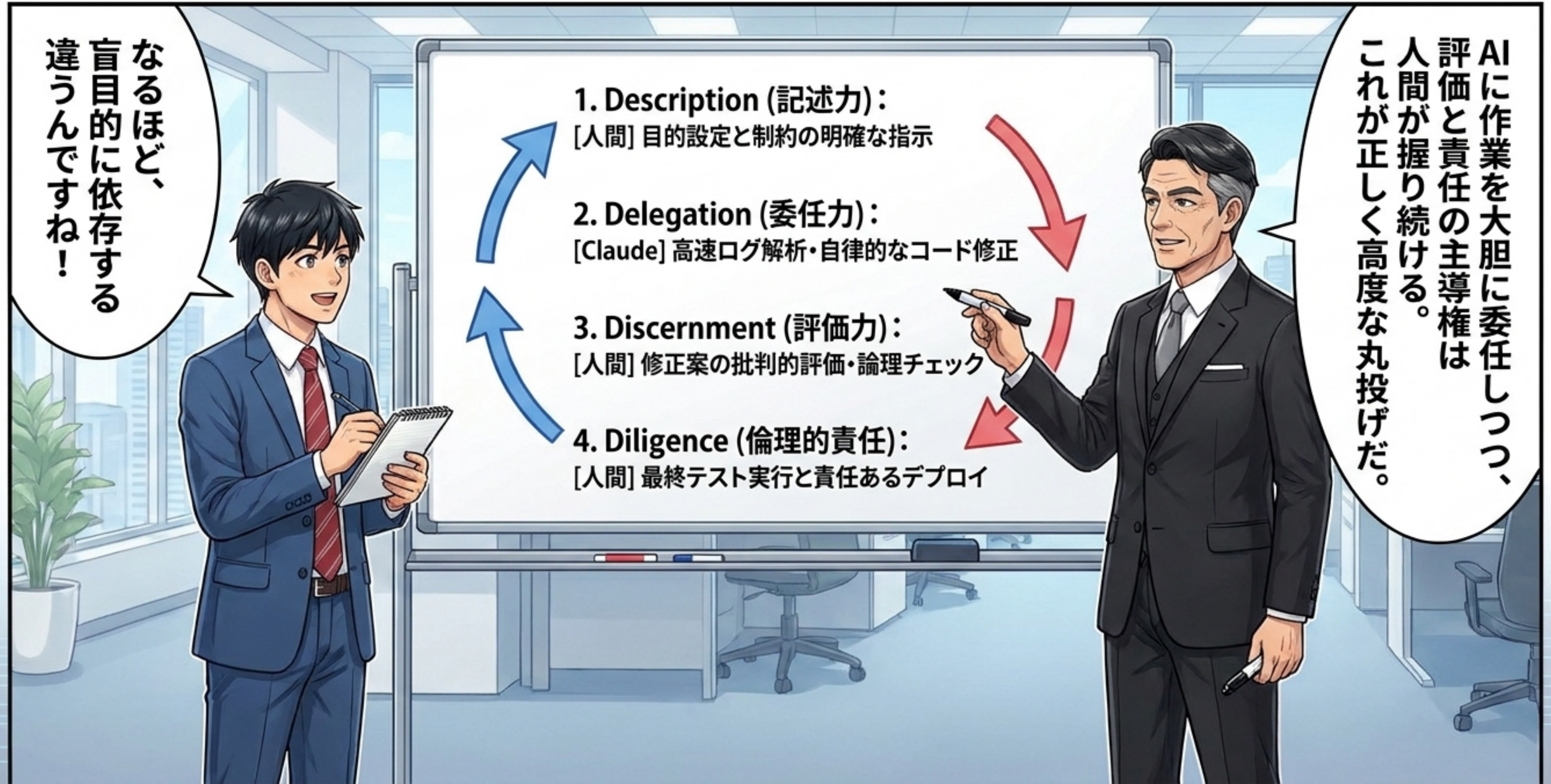


開発者の8割の時間を奪う「真犯人」



Claudeは数千行のログや複雑な依存関係を瞬時に解析する。
孤独な手作業を「AIとの協働」へシフトさせよう。

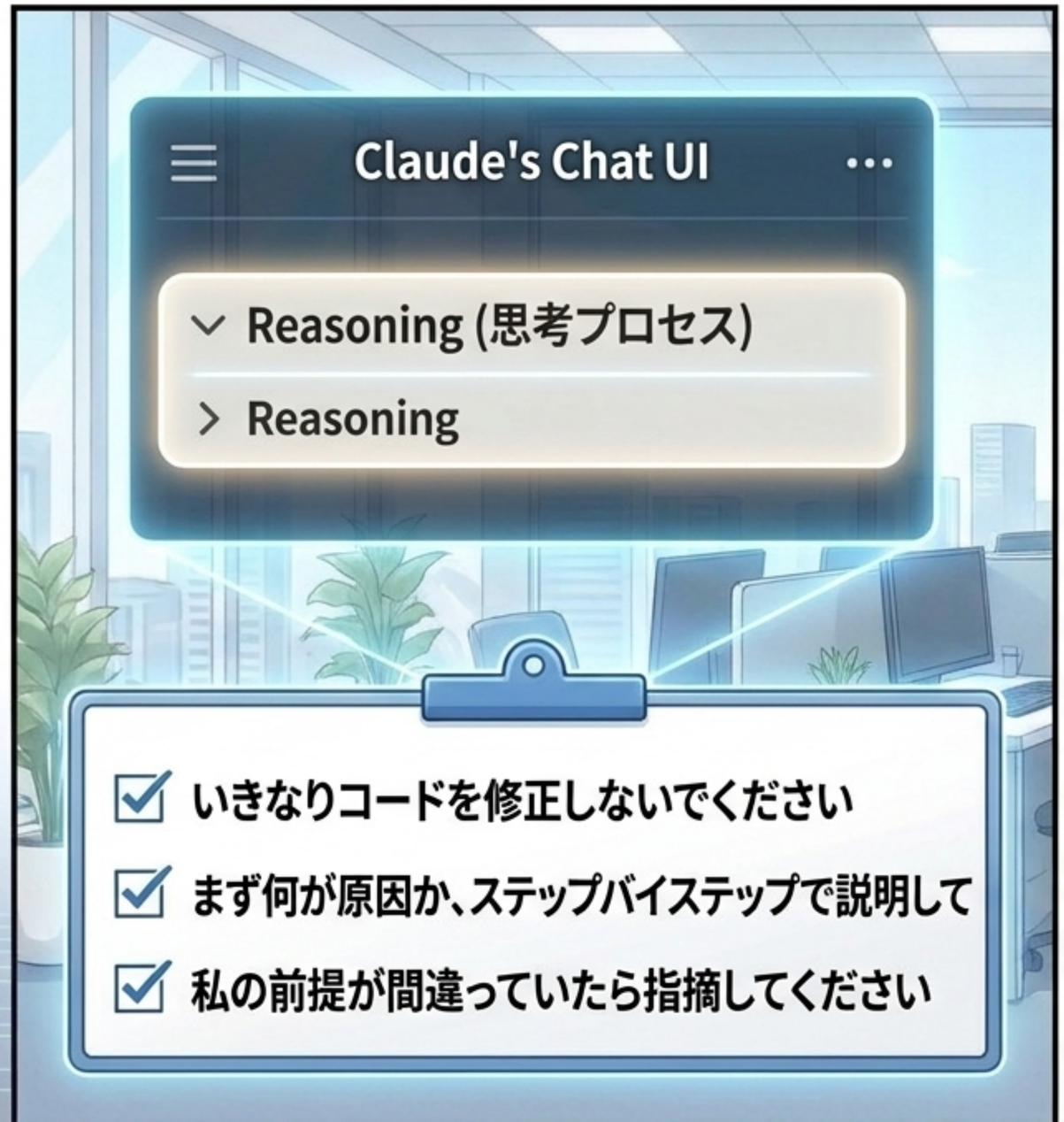
【図解】「正しい丸投げ」の4Dフレームワーク



従来の手動デバッグ vs Claude対話型デバッグ

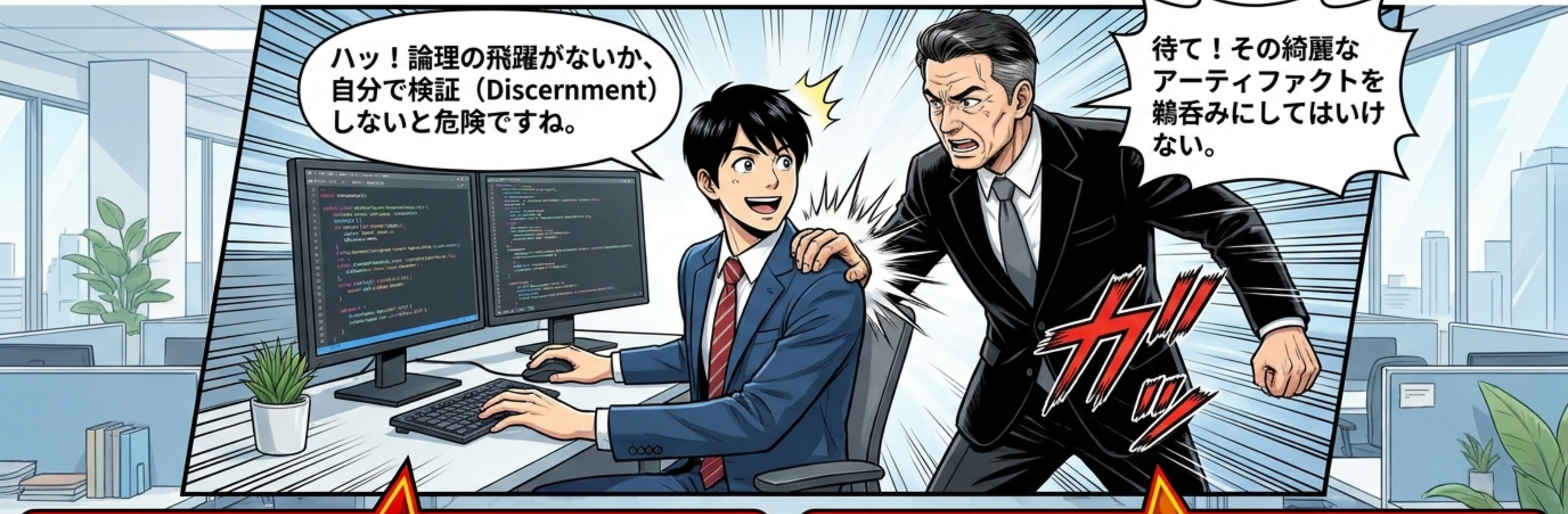


成功の秘訣は「推論」を引き出すこと





【警告】初心者が陥る2つの罠



⚠ 罠1: アーティファクト効果 (無批判な鵜呑み)

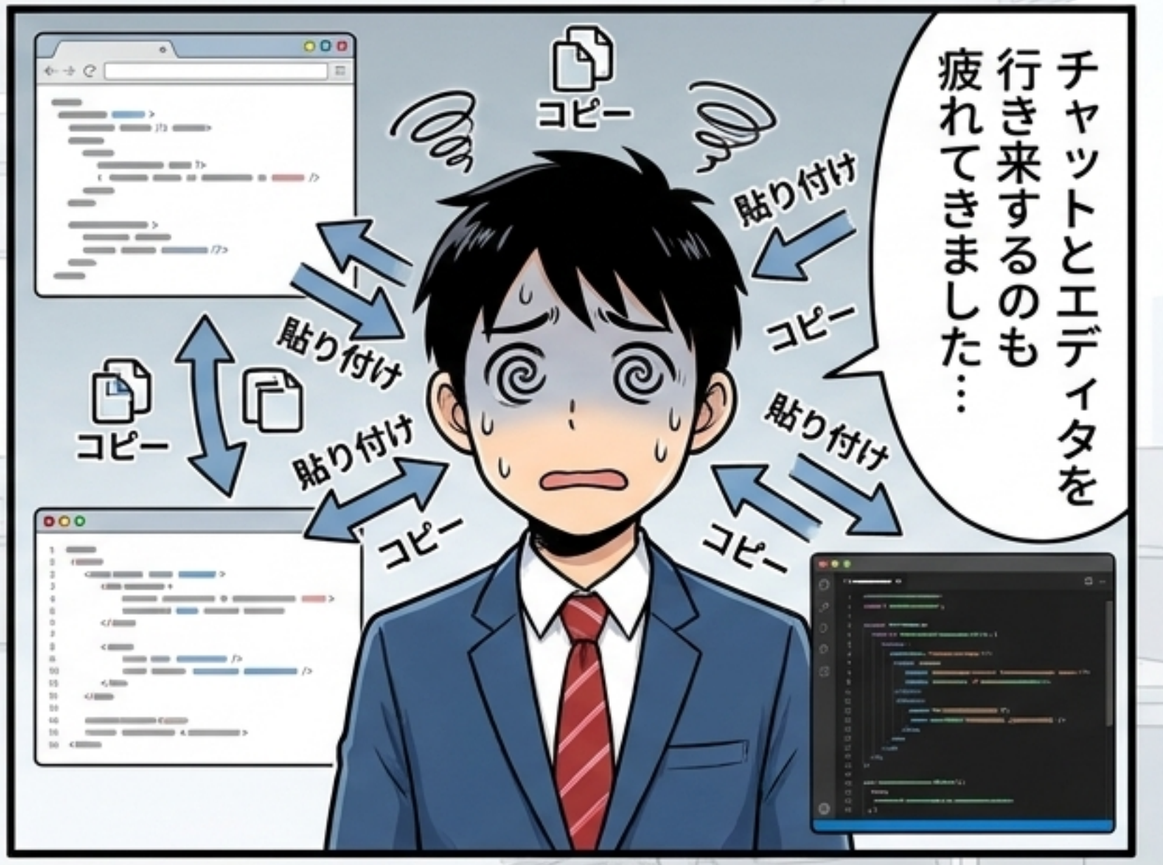
- 綺麗なコードを見ると批判的評価を怠り、二次バグを生む危険。（※データ実証：ファクトチェック行動が3.7ポイント低下）

⚠ 罠2: ハルシネーション

- 最新ライブラリの仕様などで、AIがもっともらしい嘘（存在しない関数など）を作り出す現象。

コピペの往復をなくす「Claude Code」





- Claude Codeとは：ターミナル上で動作する自律型コーディングエージェント。
- 革命的な機能：人間に代わってファイルを直接読み込み、検索し、テストを実行し、コード修正からGitコミットまでを自動で完結。


【図解】自律型エージェントループ
1. Explore (探索) :
プロジェクト内の
ファイル検索・解読
2. Plan (計画) :
原因特定・
修正方針の立案
3. Code (実装) :
ソースコードの直接編集
4. Commit (検証・確定) :
テスト自動実行・差分確認
テストが通るまで、
自分で泥臭くコードを
直して回し続けてくれる
るんですか！？
ああ。OpenBSDの27
年間誰も気づかなかった
バグすら発見・修正した
実績がある。

エフォートレベル（集中力ダイヤル）の使い分け

low: 単純修正向け。思考・ツール呼び出しを最小限にし爆速・低コスト。

medium (デフォルト): 標準デバッグ。コストと精度のベストバランス。

high / xhigh : 複雑なバグ。探索を増やし、自律的な二重チェック (自己検証) を実行。

max : 超難関調査。許容上限のリソースを投入し粘り強く探索 (※セッション限定)。



【メタファー】コンテキストウィンドウの圧迫に注意

課題（迷子現象）：

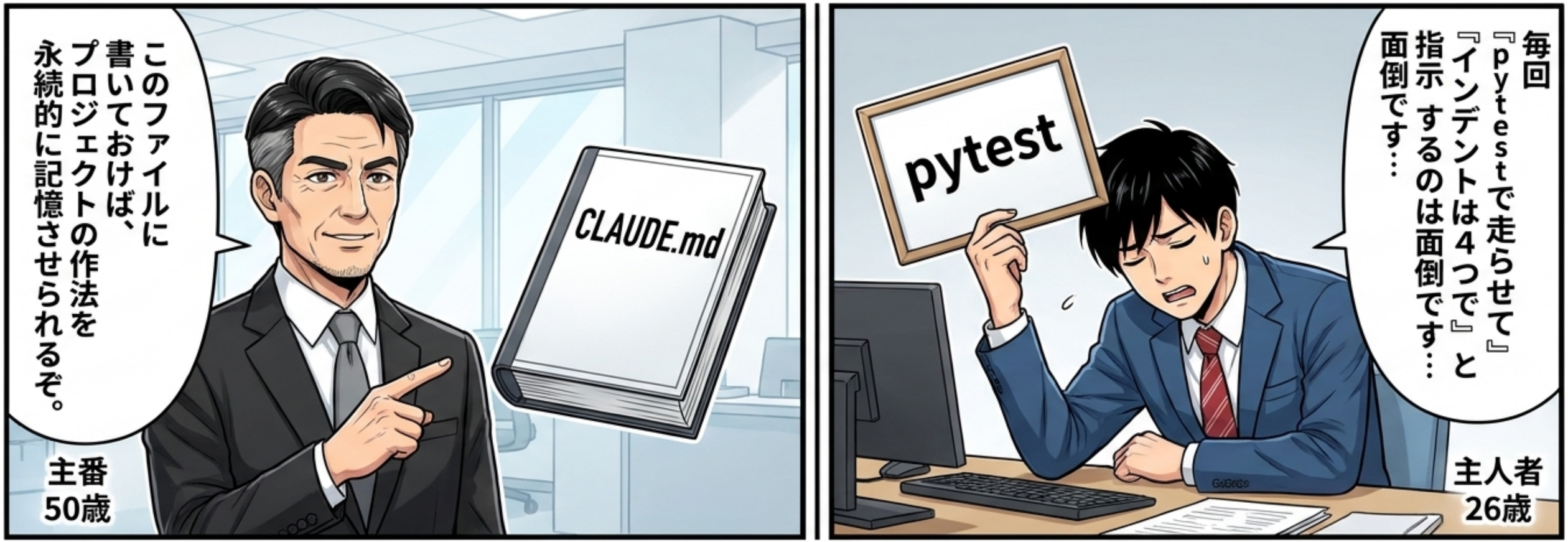
セッションが長引き大量のファイルを読み込むと、AIの「作業デスク（コンテキスト）」が埋まり、古い指示を忘れてしまう。

対策コマンド：

- /compact：会話履歴を自動要約・軽量化
- /clear：デスクを完全に掃除して新しく開始

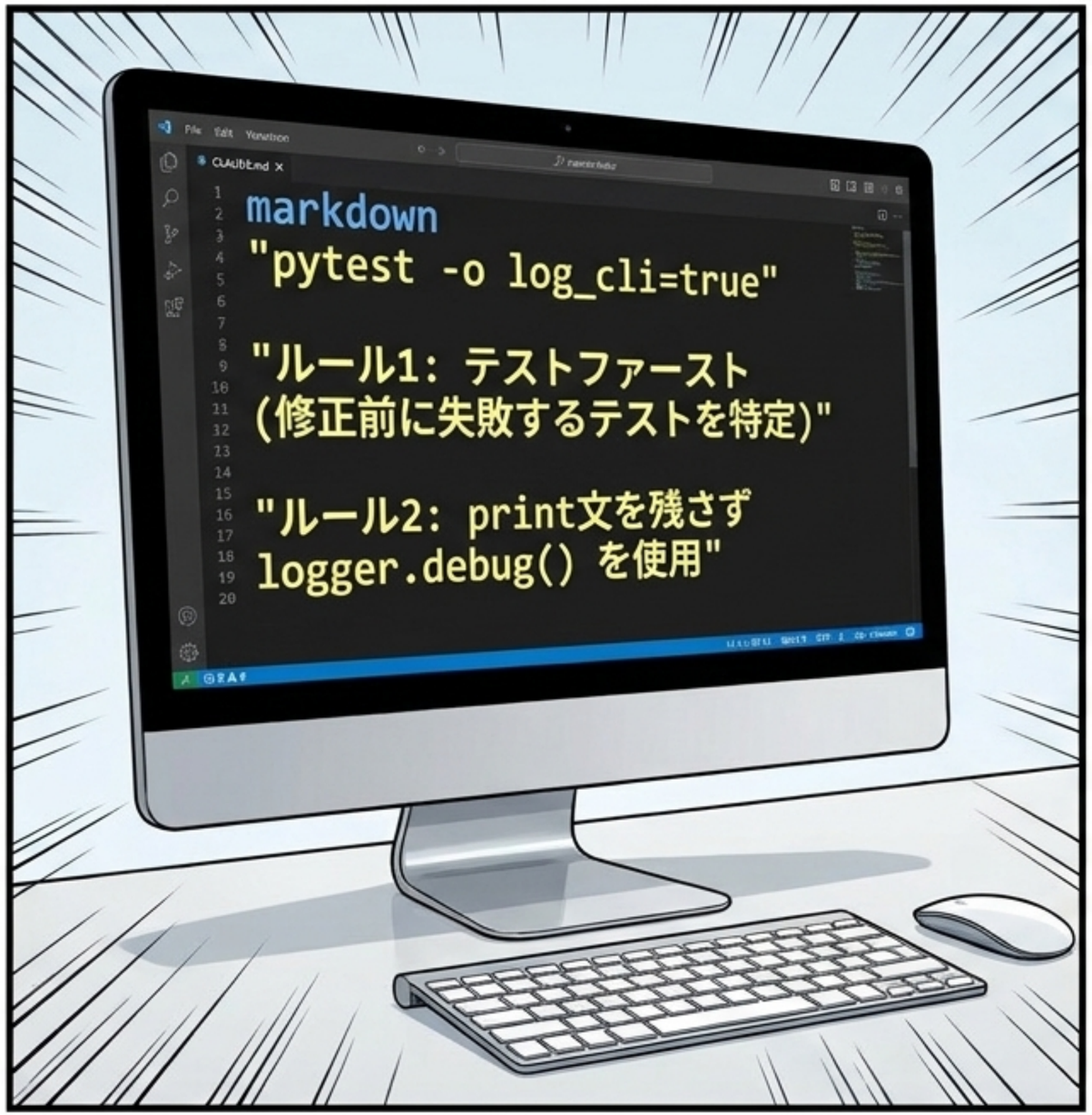


プロジェクトメモリ「CLAUDE.md」



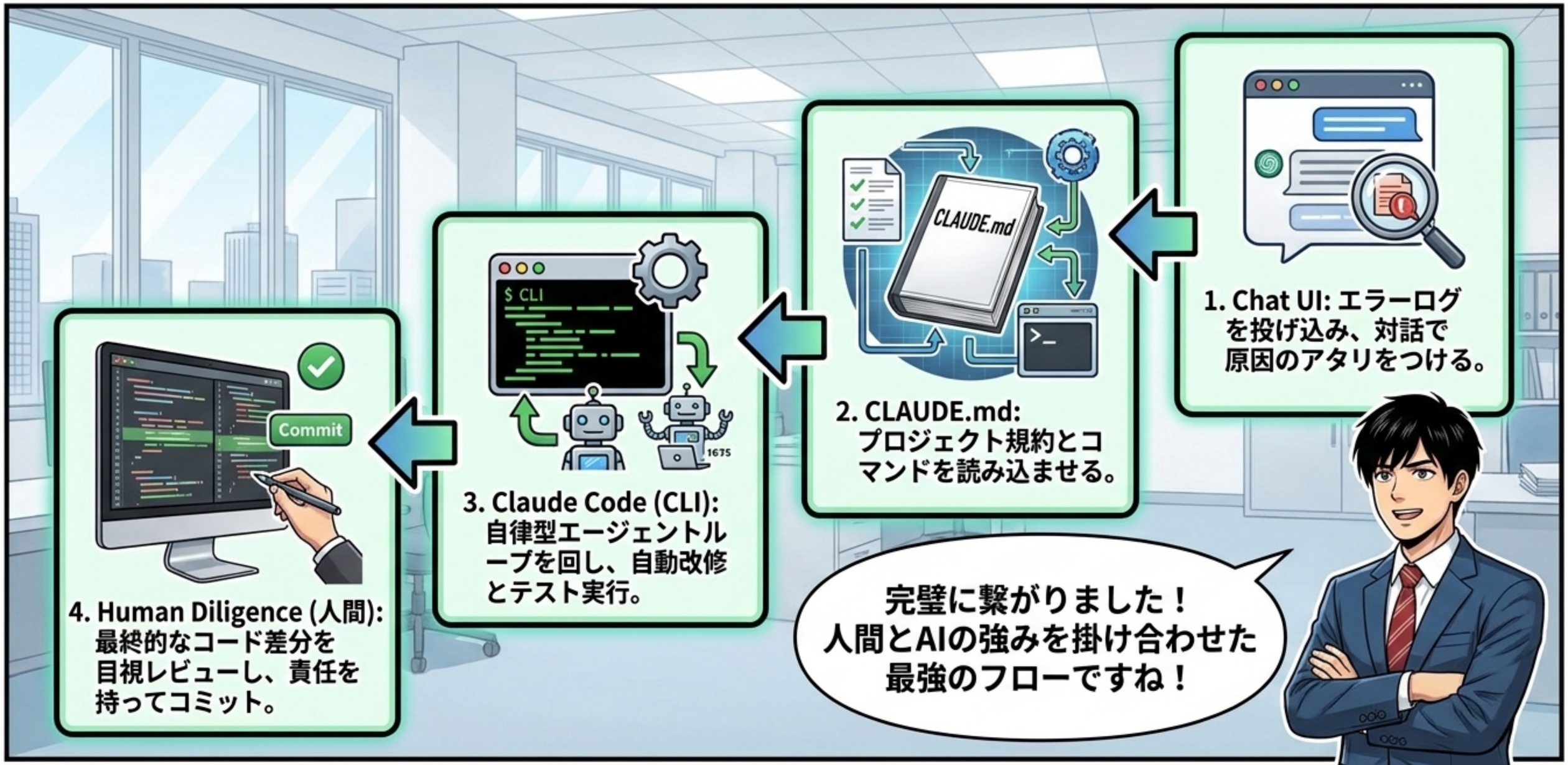
- 機能：プロジェクトルートに配置する専用の永続メモリファイル。
- 効果：起動時にClaudeが自動読み込み。テストコマンドやデバッグ規約を記憶し、新メンバーでも熟練エンジニアと同じ精度でAIを動かせる。

効果的なCLAUDE.mdの書き方

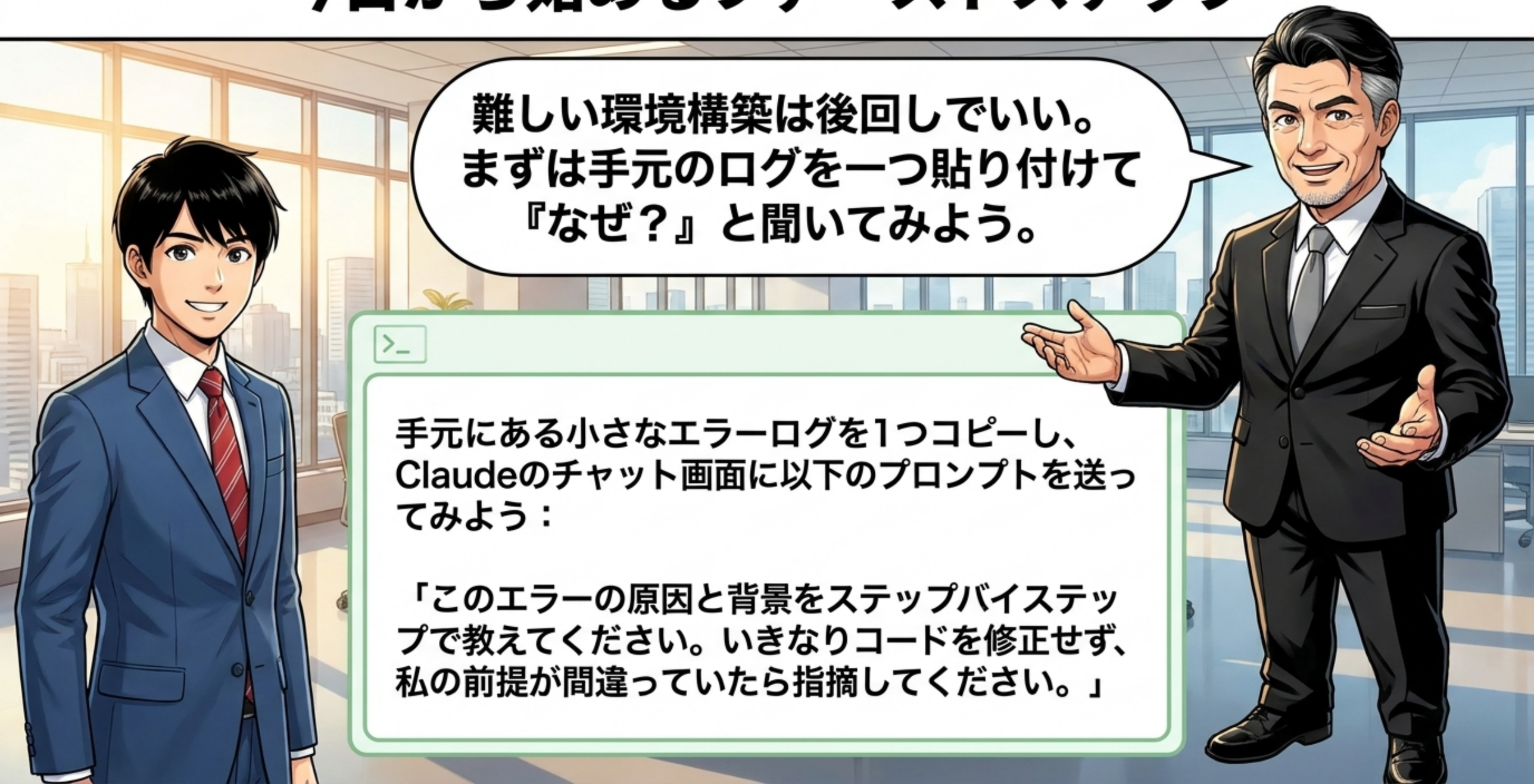




【統合】次世代のデバッグ・黄金ワークフロー



今日から始めるファーストステップ



クリエイティブな開発へ飛び出そう！

