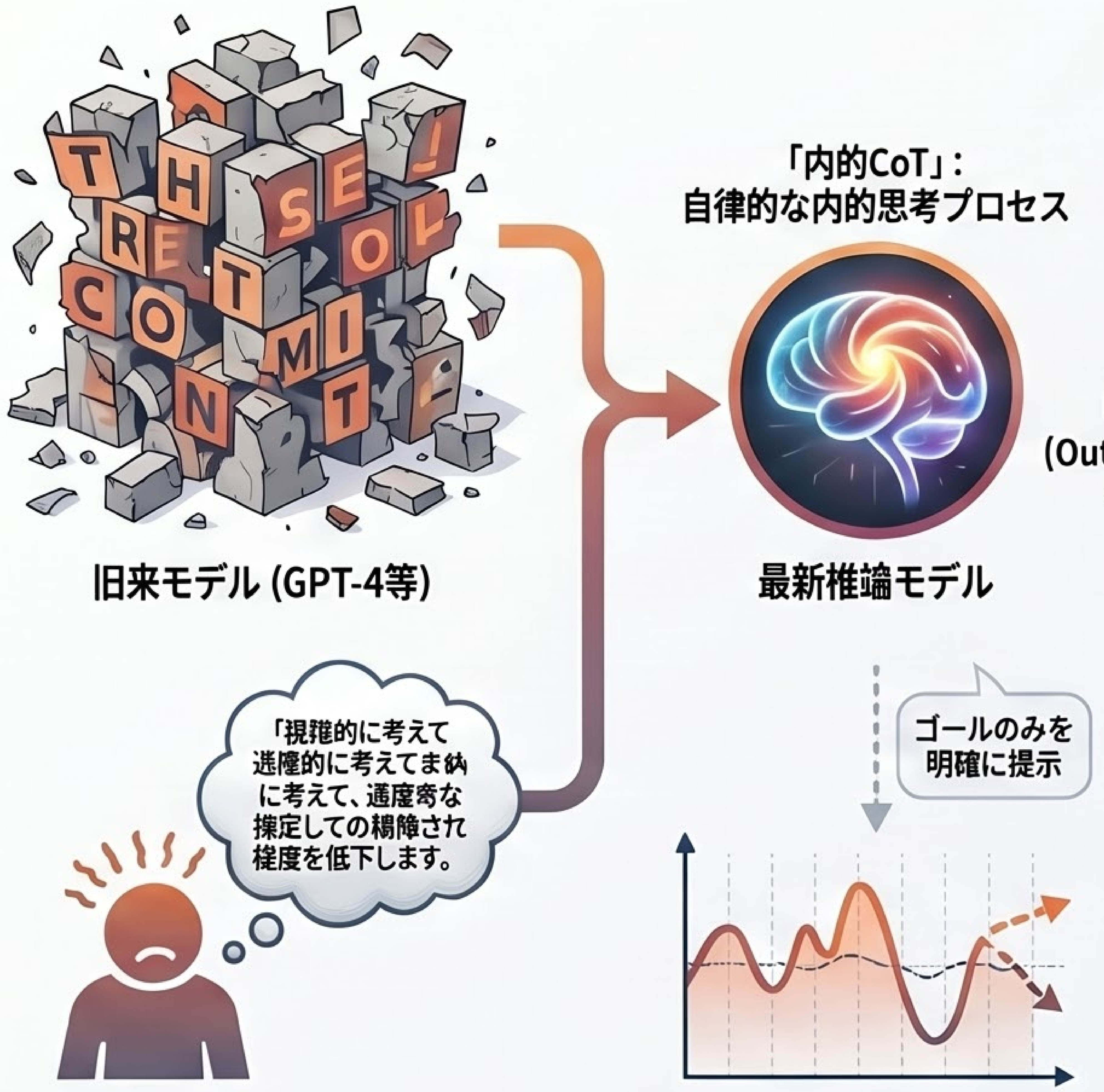
プロンプト・エンジニアリングの終焉：
AIエージェント時代の「陳腐化しない」管理術

旧来のプロンプトが「有害」になる理由



「外的CoT」：段階的に考えて

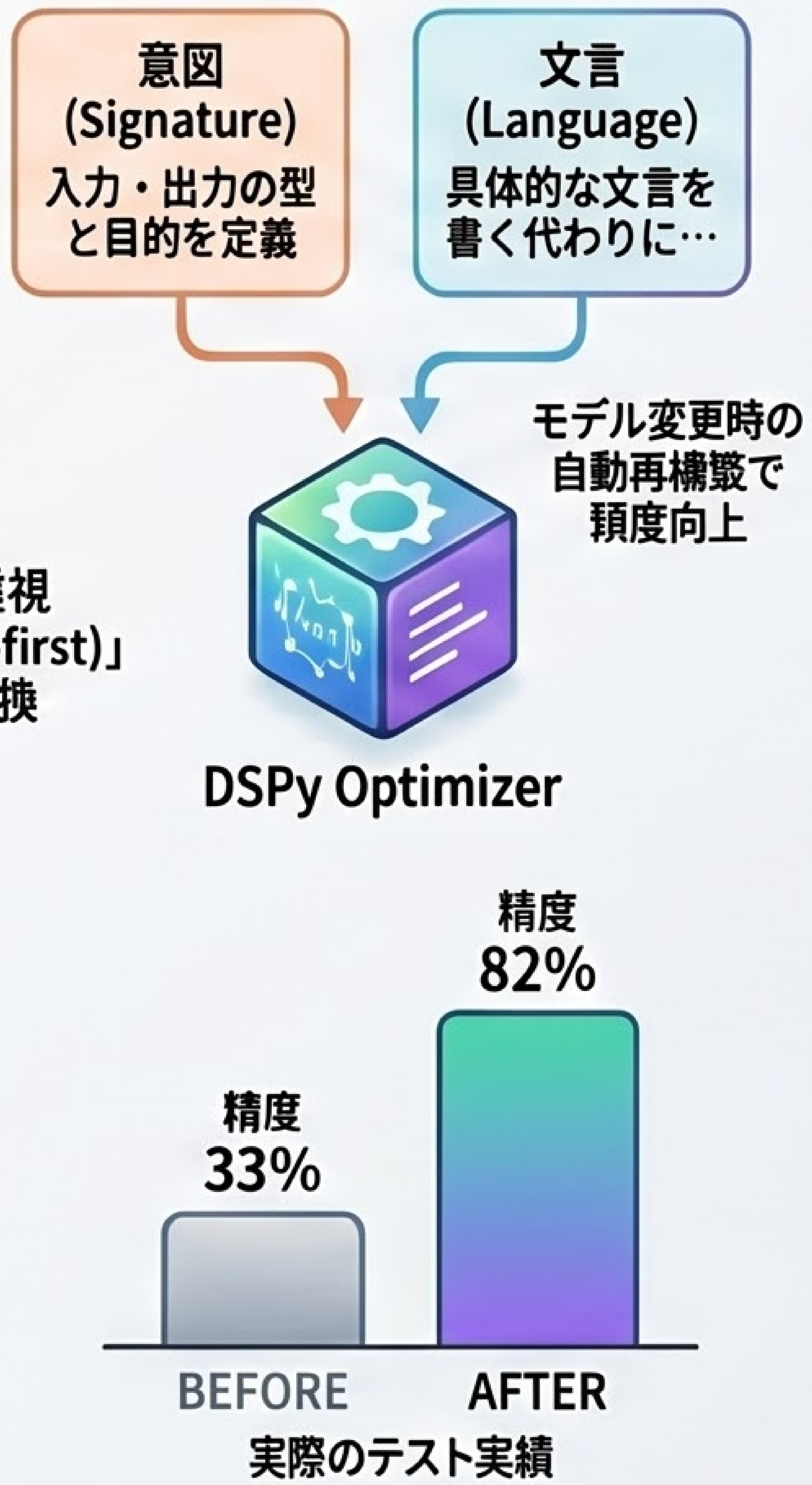
過度な指定は楯餡を剘限し、
稞度を疮下させます。

サイレントな「モデルドリフト」の脅威

一度完成したプロンプトに依存する
設計は本質的に脆弱です。

「結果重視
(Outcome-first)」
への転換

DSPyによる「プロンプトの自動コンパイル」



「職人芸」から「アルゴリズム」へ

データに基づいた自動コンパイルへと
開発プロセスが進化します。

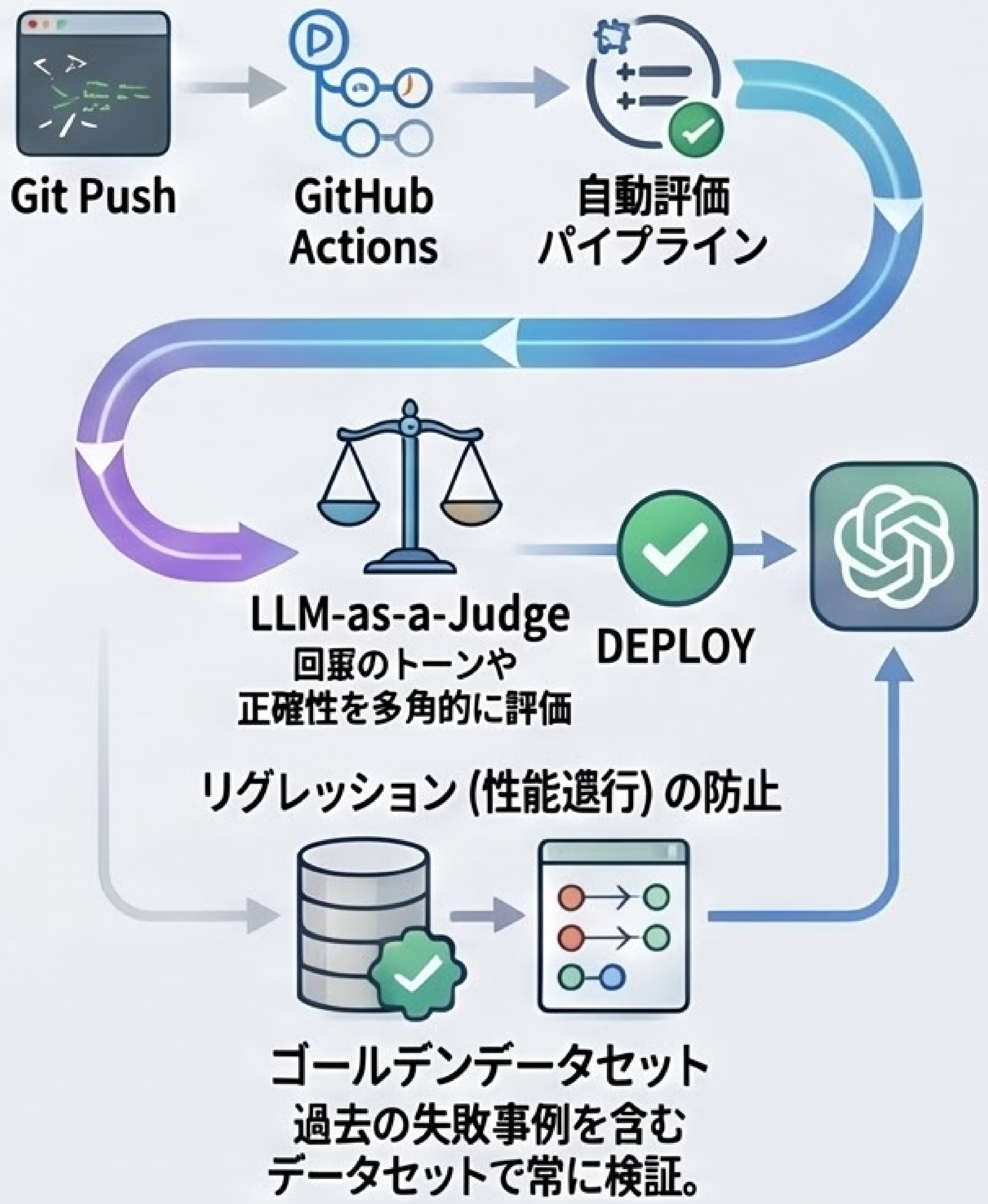
モジュール化と標準化による堅牢な設計

MCP (Model Context Protocol) の導入
LLMと外郊ツール閪の通信を標準化
し、ツール接続を抽象化。

MCP (Model Context Protocol) の導入
LLMと外部ツール期の通信を標準化し、
ツール接続を抽象化。

必要な時だけ詳機な指示をロードする
「段階的開示」アーキテクチャ。

評価駆動開発（EDD）とCI/CDパイプライン



評価項目	従來のLLMテスト	評価驅動開発 (EDD)
パス基準	完全一致などの バイナリ判定	複数次元の 品質スコアリング
失敗時の対応	人間による 手動レビュー	前後比較(DIff)による 自動特定
適応性	低い (手動更新が必要)	高い (継続的・動的な評価)