2025年 AIエンジニアリング 覇権の行方

Google Julesの進化と開発エージェント「三つ巴」戦略

2025年11月、AIソフトウェアエンジニアリング市場は三つの哲学がぶつかり合う新時代に突入した。本資料は、この「三つ巴」の現状を解き明かし、特に急速な進化を遂げたGoogle Julesを中心に、あなたのチームが採用すべき最適な戦略を提示する。

3つのAIエージェント、3つの個性

実用的なジュニアエンジニア

(The Junior Colleague)

GitHub PR中心のワークフローを自動化する、実用性とコストパフォーマンスのバランス型。

バランス　コスパ　安全性　CI/CD連携

完全自律型エンジニア

(The Autonomous Engineer)

人間の介入を最小限に、複雑なタスクを独力で長時間かけて解決する高性能・高価格な自律完結型。

自律性　完遂力　高価格　エンドツーエンド

AIペアプランナー

(The AI Pair Planner)

人間が策定したプランに基づき実装を支援する、協業とプランニングを重視したGitHubネイティブ環境。

協業　プランニング　GitHubネイティブ　思考整理

AI開発エージェント 3強比較マトリクス（2025年11月版）

特徴	Google Jules	Cognition Devin	GitHub Copilot Workspace
開発コンセプト	"The Junior Colleague" （実用的なジュニアエンジニア）	"The Autonomous Engineer" （完全自律型エンジニア）	"The AI Pair Planner" （協業プランナー）
搭載モデル	**Gemini 3 Pro** （2025/11より）	GPT-4o / o1（カスタム調整版） ※推定	GPT-4o, Claude 3.5 Sonnet など複数選択可
得意なタスク	バグ修正、テスト作成、マイグレーション、CI/CD連携	未知技術の調査〜実装、環境構築、E2Eの複雑な機能開発	仕様策定、設計ブラッシュアップ、複数ファイルのリファクタリング
実行環境	Google Cloud VM （セキュアなクリーン環境）	独自のサンドボックス環境 （Webブラウザ完結）	Codespaces / Web （GitHubエコシステム内）
ローカル連携	**Jules Tools (CLI)** **★最重要アップデート**	専用CLIあり （Webへのブリッジ）	Copilot CLI （ターミナル補完が主）
価格（目安）	**Free / Paid** （無料枠: 15タスク/日）	**高額** （月額 $500〜）	**安価** （Copilot Pro $20/月に包含）

Julesの急進化：単なるGitHub Botから開発パートナーへ

2025年8月: 正式版 (GA) リリース

Gemini 2.5 Pro搭載
Jules Critic (自己修正機能)

2025年9月: DX改善

Memory機能（リポジトリ毎の学習）
PRコメントへの自動修正対応

2025年10月: エコシステム拡張 (転換点)

10/2: Jules Tools (CLI) リリース
- ターミナルからの直接操作が可能に

10/3: Jules API 公開
- CI/CDやChatOpsとの連携を実現

2025年11月: パフォーマンスと柔軟性の向上

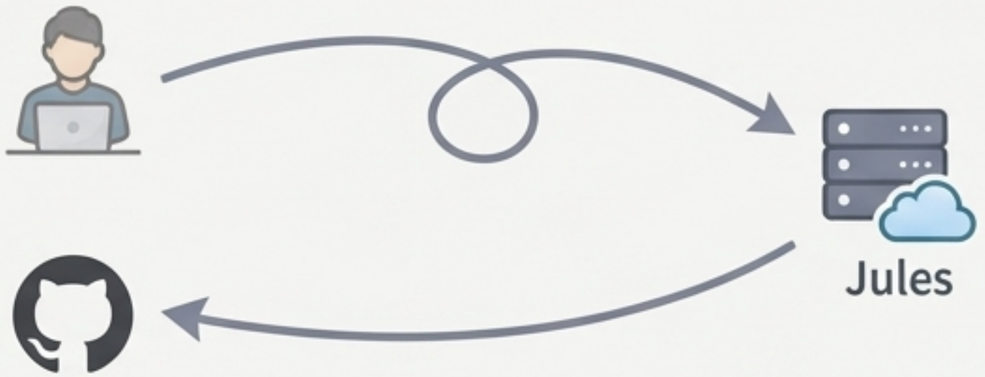
11/19: Gemini 3 Pro へモデル刷新
- 複雑なタスクの計画・実行精度が向上

11/20: Repolessセッション
- リポジトリ無しでタスク開始が可能に

ゲームチェンジャー「Jules Tools」が変えたこと

Before: 2025年9月まで

GitHub上の「非同期PR作成マシン」



開発者がWeb UIでタスクを依頼
→ Julesがバックグラウンドで処理
→ GitHubにPRが作成されるのを待つ

ローカル環境には介入できず、あくまでGitHub中心の非同期な関係。

After: 2025年10月以降

手元のターミナルで動く「常駐ペアプログラマー」

`jules run "..."`でローカルファイルを直接編集・テスト実行 → 開発者はリアルタイムで差分を確認・対話

- 🛠️ **CI/CDパイプラインへの直接介入**: テスト失敗時に自動修正
- ⏩ **ローカルでの迅速なイテレーション**: Web UIを介さず開発フローに密着
- 💬 **ChatOps連携**: Slack等から直接タスクを起動

今すぐ始める Jules Tools CLI

1. インストール

```
npm install -g @google-labs/jules
jules --version
```

2. 認証

```
jules auth login
```

ブラウザが開き、GoogleアカウントでCLIを認証します。

3. 初期化

```
cd /path/to/your-project
jules init
```

対話形式でテストコマンド等を設定し、`.julesignore`ファイルを生成します。

3. 初期化

```
cd /path/to/your-project
jules init
```

対話形式でテストコマンド等を設定し、`.julesignore`ファイルを生成します。

4. タスク実行

バグ修正

```
jules run "ログインAPIのnullポインタ例外を修正して"
```

Issue連携

```
jules fix --issue 123
```

事前確認

```
jules run "..." --dry-run
```

> **Pro Tip**
> `.julesignore`を`.gitignore`と同様に活用し、`node_modules`や`.env`ファイルを読み込ませないことで、精度向上とトークン節約が可能です。

Google AI開発ツールエコシステムの歩き方

Jules, Antigravity, Gemini CLI — それぞれの役割と最適な使い分け

ツール	主用途	実行場所	自律性	想定ユーザー
 **Jules**	コードベースへの変更を**タスク単位で自動実行**	Web UI + Jules Tools CLI	**高**：タスクを投げればPRまで完遂	既存プロダクトを持つソフトウェアエンジニア
 **Antigravity**	**IDE + エージェント**での開発全体管理	専用エディタ（VS Code系）	**高**：エージェントが計画〜検証まで主導	新規開発や一つのサービスに集中するエンジニア
 **Gemini CLI**	**ターミナル**からGeminiを叩いて開発・自動化	ローカルターミナル / Cloud Shell	**中〜高**：ReActループでツールを実行	CLI好きの開発者 / SRE / DevOps
 **Geminiアプリ**	日常・仕事全般の**汎用AIアシスタント**	ブラウザ / モバイルアプリ	**低**：人が都度指示する対話ベース	全てのナレッジワーカー（仕様策定・ドキュメント作成）

シナリオ別：Google AI開発ツールの連携ワークフロー

Scenario: 新機能「ユーザープロフィール編集」の実装

1. 要件整理 & 仕様策定 (Geminiアプリ)

チケットのドラフト、RFCの叩き台を作成。

「ユーザープロフィール編集機能の仕様案をマークダウンで書いて」

2. 初期実装 & 反復開発 (Antigravity or VS Code + Copilot)

IDE内でエージェントと対話しながら、UIコンポーネントやAPIエンドポイントの骨子を実装。

3. 面倒なタスクの委任 (Jules)

GitHubにpush後、Julesにタスクを依頼。

```
jules run "関連するユニットテストとE2Eテストを追加してカバレッジを90%以上にして"
```

```
jules run "既存のロギング規約に合わせて、この機能に監査ログを実装して"
```

4. 運用 & 調査 (Gemini CLI)

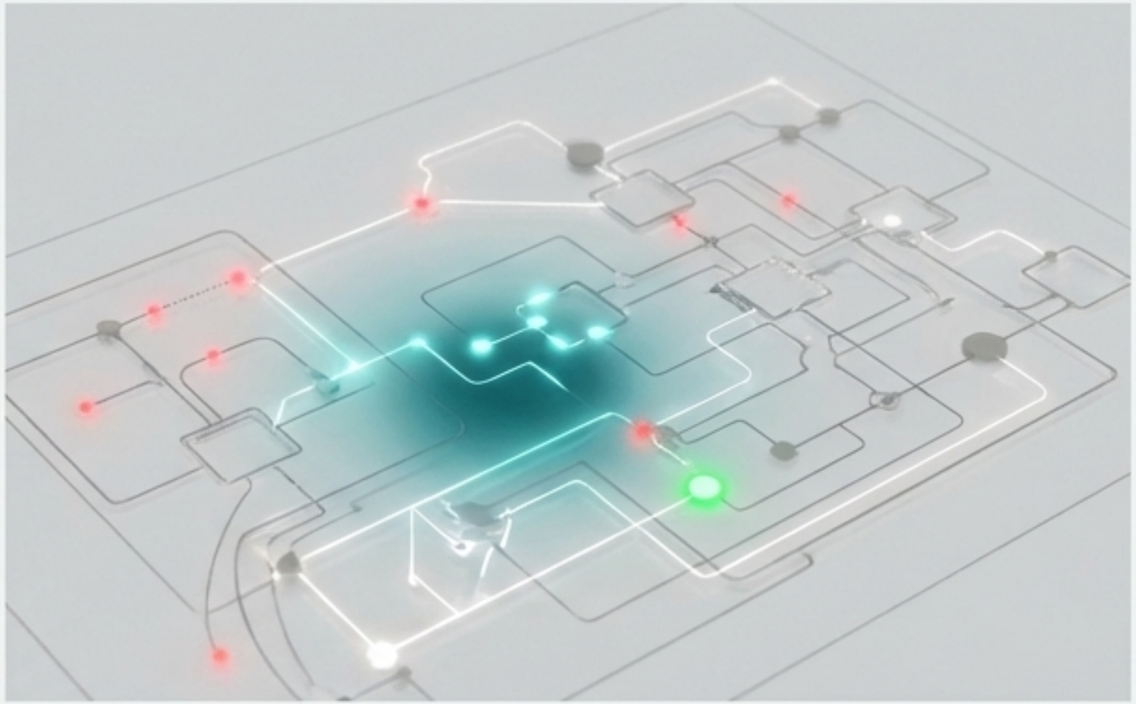
リリース後、Cloud Shellからコマンドを実行。

```
gemini run "過去1時間の 'profile-update' に関するエラーログを要約して"
```

"軽微な設定変更や調査スクリプトをその場で生成・実行。"

Julesの次章：開発ワークフローの完全自律化へ

現在開発中（リーク情報ベース）の機能は、Julesが「指示待ち」のエージェントから「能動的」なパートナーへと進化することを示唆している。



Proactive Mode（プロアクティブモード）

リポジトリを常時監視し、潜在的なバグ、パフォーマンスのボトルネック、非効率なコードを自律的に検知。

人間が指示する前に、Julesが自動で改善案をPRとして提案。「コードベースの健康診断と自動治療」を実現する。

Scheduled Tasks（スケジュール実行）

定期的なメンテナンス作業をスケジュール実行する機能。

ユーケース
- 毎週月曜朝に「全ライブラリの依存関係をアップデート」
- 毎晩「ドキュメントが古いコードを検出して更新」

2025年後半の賢いAIエンジニアリング戦略

「Julesを主軸に、Copilot WorkspaceとDevinを適材適所で組み合わせる」