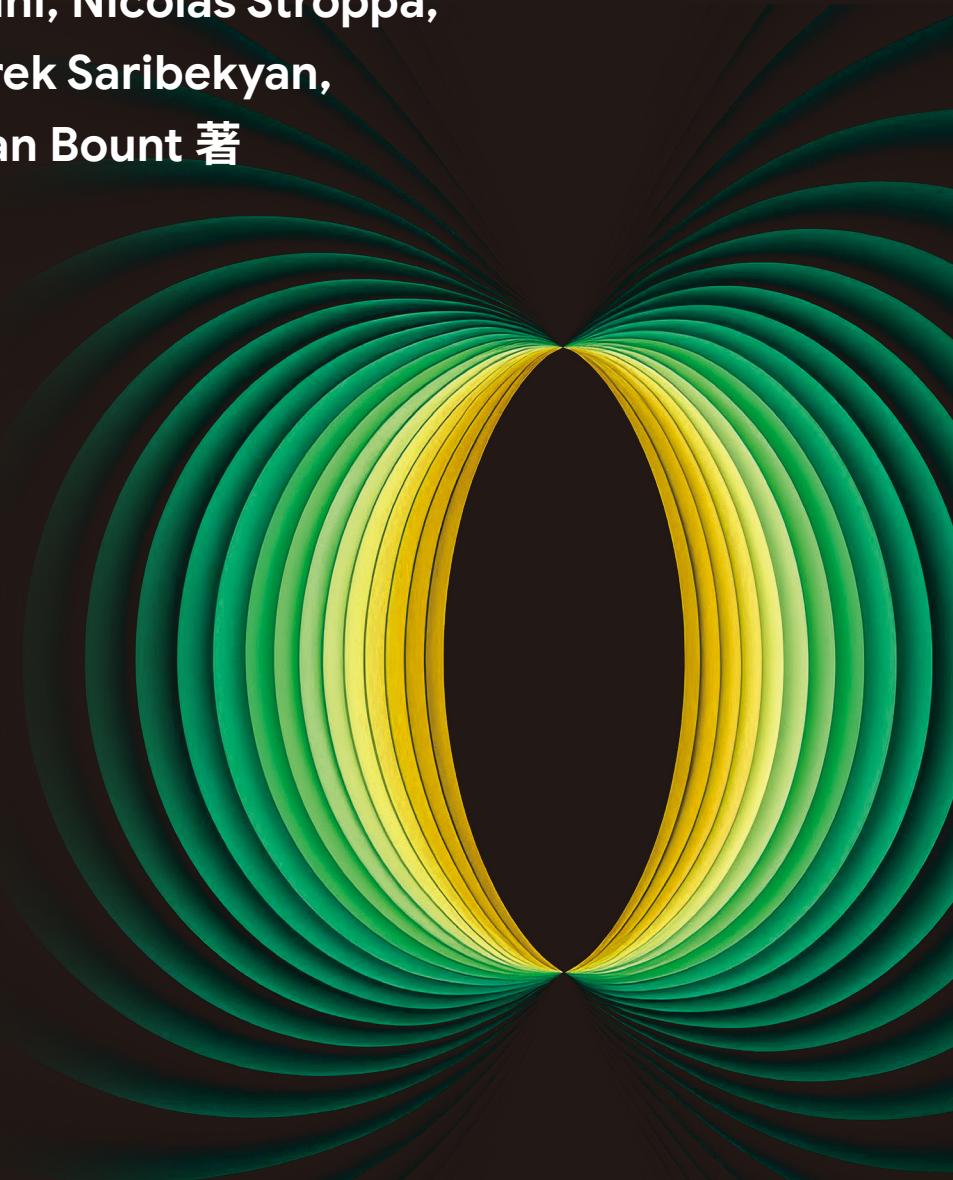


# Agents Companion

AI エージェントの構築と運用

**Antonio Gulli, Lavi Nigam,  
Julia Wiesinger, Vladimir Vuskovic,  
Irina Sigler, Ivan Nardini, Nicolas Stroppa,  
Sokratis Kartakis, Narek Saribekyan,  
Anant Nawalgaria, Alan Bount 著  
Kuma Arakawa 訳**

Google



## Acknowledgements

### Editors & curators

Anant Nawalgaria

### Content contributors

Steven Johnson

Hussain Chinoy

### Designer

Michael Lanning

## Table of contents

はじめに .....	P6
AgentOps .....	P8
エージェントの成功指標 .....	P11
エージェント評価 .....	P13
エージェントの能力の評価 .....	P13
トラジエクトリとツール使用の評価 .....	P16
最終応答の評価 .....	P18
ヒューマンインザループ評価 .....	P19
エージェント評価についてのさらなる考察 .....	P20
複数エージェントとその評価 .....	P21
マルチエージェントアーキテクチャの理解 .....	P22
マルチエージェントの設計パターンとそのビジネスインパクト .....	P23
エージェントの重要コンポーネント .....	P25
マルチエージェントシステムの課題 .....	P27
マルチエージェントの評価 .....	P28

エージェント型RAG:検索拡張生成における重要な進化	P29
エージェント型RAG とその重要性	P30
より良い検索、より良いRAG	P31
エンタープライズにおけるエージェント	P33
エージェントの管理者	P33
Google Agentspace	P35
NotebookLM Enterprise	P36
Google AgentSpace Enterprise	P37
エージェントからコントラクターへ	P39
契約	P39
契約ライフサイクル	P42
契約の実行	P42
契約交渉	P43
契約フィードバック	P43
サブコントラクト	P43
Google のCo-Scientist:マルチエージェントインテリジェンスにおけるケーススタディ	P44
車載AI:マルチエージェントアーキテクチャの実世界での使用例	P46
特化型エージェント	P46
対話型ナビゲーションエージェント	P47
対話型メディア検索エージェント	P48
メッセージ作成エージェント	P48
車両マニュアルエージェント	P49
一般知識エージェント	P49

使用されるパターン .....	P50
階層型パターン .....	P50
ダイアモンドパターン .....	P52
ピアツーピア .....	P54
協調型パターン .....	P55
応答ミキサーモード .....	P57
適応型ループパターン .....	P58
車載AIにおけるマルチエージェントアーキテクチャの利点 .....	P59
Agent Builder .....	P60
まとめ .....	P61
巻末注 .....	P64

# AI の未来はエージェント性にある

## はじめに

生成 AI エージェントは、従来型のスタンドアロンな言語モデルからの飛躍的な進歩を示すものであり、問題解決と対話に対する動的なアプローチを提供します。元の「Agents」に関するホワイトペーパーで定義されているように、エージェントとは、その環境を認識し、自由に使えるツールを用いて戦略的に行動することで、特定の目的を達成するように設計されたアプリケーションのことです。

エージェントの基本原則は、推論、論理、そして外部情報へのアクセスの統合にあり、これにより基盤となるモデルが本来持つ能力を超えたタスクの実行や意思決定が可能になります。これらのエージェントは自律的な運用能力を持ち、しばしば明示的な指示なしに、独立して目標を追求し、後続のアクションをプロアクティブに決定します。

エージェントのアーキテクチャは、その振る舞いと意思決定を駆動する 3 つの不可欠な要素で構成されています。

- ・**モデル**：エージェントのフレームワークにおいて、「モデル」という用語は、指示に基づいた推論および論理フレームワークを用い、中核的な意思決定ユニットとして機能する言語モデルを指します。モデルは、エージェントの特定の要件に応じて、汎用目的のものから、マルチモーダル、あるいはファインチューニングされたものまで様々です。
- ・**ツール**：ツールは、エージェントの内部能力と外部世界との間の隔たりを埋める上で極めて重要であり、外部のデータやサービスとの対話を促進します。これらのツールは、エージェントが実世界の情報にアクセスし処理する能力を与えます。ツールには、拡張機能、Function、およびデータストアが含まれます。拡張機能は API とエージェント間のギャップを埋め、エージェントが API をシームレスに実行できるようにします。Function は、特定のタスクを達成する自己完結型のコードモジュールです。データストアは、動的に最新の情報へのアクセスを提供し、モデルの応答が事実性と関連性に根差し続けることを保証します。
- ・**オーケストレーション層**：オーケストレーション層は、エージェントが情報を吸収し、内部推論を行い、その推論を活用して後続の行動や意思決定の判断材料とする方法を規定する、周期的なプロセスです。この層は、記憶、状態、推論、および計画の維持を担当します。プロンプトエンジニアリングフレームワークを採用して推論と計画を誘導し、環境とのより効果的な対話とタスクの完了を促進します。この層内では、ReAct、Chain-of-Thought、Tree-of-Thoughts などの推論技術を適用できます。

これらの基本的な概念を基に、この Agent Companion ホワイトペーパーは開発者向けに設計されており、より高度なトピックへの「102」ガイドとしての役割を果たします。本書では、エージェント評価手法を深く探求するとともに、複雑な実世界の問題を解決するエージェントの能力を強化するための、Google のエージェント製品の実践的な応用について解説します。

これらの理論的な概念を探求しつつ、それらが実世界の実装でどのように具現化されるかを検証します。特に、説得力のあるケーススタディとして、車載 AI に焦点を当てます。

自動車というドメインは、本番環境におけるマルチエージェントアーキテクチャの課題と可能性を象徴しています。現代の車両には、コネクティビティの有無にかかわらず機能し、安全性とユーザーエクスペリエンスの両立のためにオンデバイス処理とクラウド処理のバランスを取り、ナビゲーション、メディアコントロール、メッセージング、車両システムにまたがる特化された能力をシームレスに連携させる、対話型インターフェースが求められます。

この自動車というレンズを通して、階層型、協調型、そしてピアツーピアといった異なる連携パターンが、大きな制約のある環境下でいかにして堅牢で応答性の高いユーザーエクスペリエンスを生み出すために統合されるかを見ていきます。このケーススタディは、様々な業界のビジネスがそれぞれのドメインに合わせて適応できる、マルチエージェントシステムの実践的な応用例を示しています。

生成 AI を使って何かを構築したことがある人なら誰でも、アイデアからブルーフォブコンセプトに至るのは簡単である一方、高品質な結果を保証して本番環境に移行するのは非常に難しいことにすぐに気づきます。生成 AI エージェントもその例外ではありません。

品質と信頼性は、本番環境へのデプロイにおいて最もよく挙げられる懸念事項であり、「AgentOps」プロセスはエージェントの構築を最適化するためのソリューションです。

# AgentOps

この2年間で、生成AIの分野は大きな変化を遂げ、エンタープライズ顧客は関連ソリューションをいかに運用化するかに焦点を当てるようになりました。この結果、生成AIの運用化を説明する様々な用語、例えば「MLOps for GenAI」、「LLMOps」、「FMOps」、「GenAIOps」などが生まれました。

AgentOps (Agent and Operations) は、エージェントの効率的な運用化に焦点を当てた GenAIOps のサブカテゴリです。その主な追加コンポーネントには、内部および外部のツール管理、エージェントブレインプロンプト（目標、プロファイル、指示）とオーケストレーション、メモリ、そしてタスク分解が含まれます。

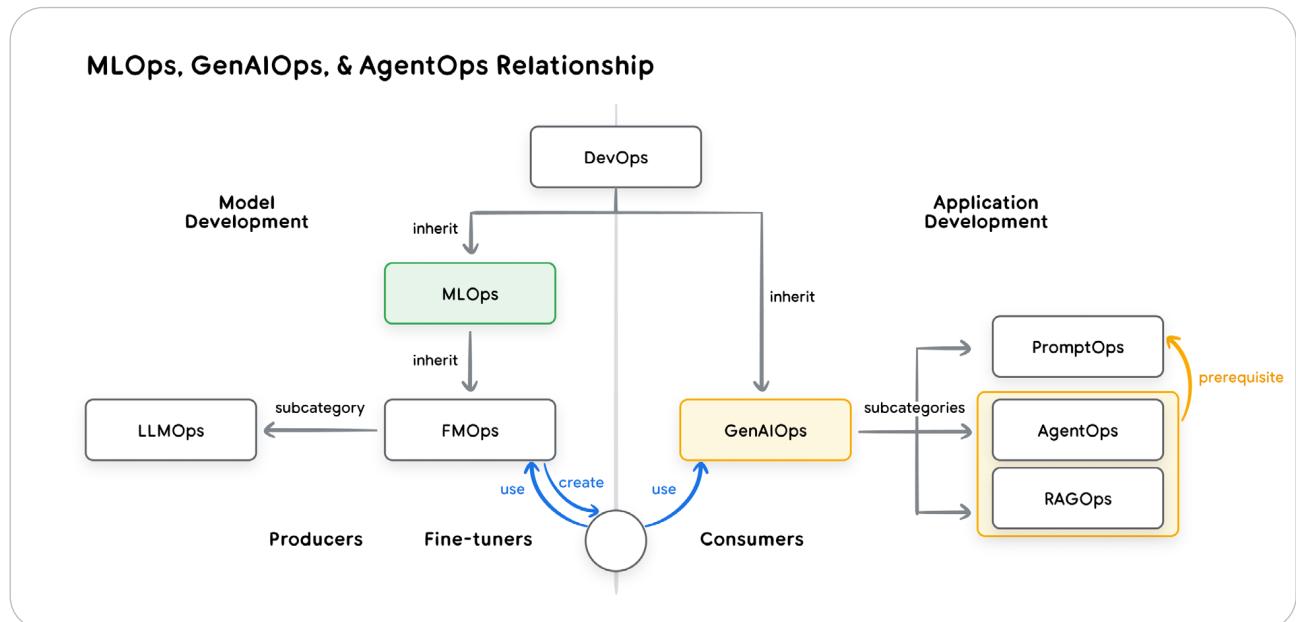


図1：DevOps、MLOps、およびAgentOps間の関係

これらの「Ops」はそれぞれ、バージョン管理、CI/CDによる自動デプロイ、テスト、ロギング、セキュリティ、そして（極めて重要な）メトリクスといった機能を必要とします。各システムはしばしばメトリクスに基づいて何らかの形の最適化を実装します。つまり、システムが何を行っているか・いないかを測定し、成果やビジネスメトリクスを測定し、より包括的なメトリクスを得るためにプロセスを自動化し、段階的に改善していくのです。この実践は、「A/B テスト」、「ML Ops」、「メトリクス駆動開発」などと呼ばれることがあります、それらは同じ一般的なアプローチから派生したものであり、私たちは AgentOps においてもこれらの原則に依拠します。

新しいプラクティスが古いものに取って代わるわけではない点に留意すべきです。DevOps と MLOps は依存関係にあるため、これらのベストプラクティスは AgentOps にとって依然として必要不可欠です。例えば、エージェントツールがエージェントオーケストレーションに基づいて API を呼び出す場合、それは非エージェント的なソフトウェアで呼び出す API と同じであることがよくあります。認証とシークレット管理、セキュリティ、プライバシー、例外処理、スロットリング、クオータ、そしてスケーラビリティは依然として極めて重要であり、エージェント設計に加えて慎重な API 設計が求められます。

これらの「Ops」用語を区別しやすくするために、それぞれを定義してみましょう。

- **DevOps (Development and Operations)**：人、プロセス、テクノロジーの要素を統合することにより、決定論的なソフトウェアアプリケーションを効率的に運用化する実践。DevOps は、以降のすべての用語の基盤として機能します。
- **MLOps (Machine Learning Operations)**：DevOps の能力を基に構築され、ML モデルの効率的な運用化に重点を置きます。主な違いは、ML モデルの出力が非決定論的であり、入力データに依存する点です（ガベージイン・ガベージアウト）。
- **FMOps (Foundation Model Operations)**：MLOps の能力を拡張し、事前訓練済み（スクラッチから訓練された）またはカスタマイズされた（ファインチューニングされた）基盤モデル（FM）の効率的な運用化に焦点を当てます。
- **PromptOps (Prompt and Operations)**：GenAIOps のサブカテゴリであり、プロンプトを効果的に運用化することに焦点を当てます。その主な追加機能には、プロンプトストレージ、リネージ、メタデータ管理（評価スコアを含む）、一元化されたプロンプトテンプレートレジストリ、およびプロンプトオプティマイザが含まれます。
- **RAGOps (RAG and Operations)**：GenAIOps のサブカテゴリであり、RAG ソリューションを効率的に運用化することを中心とします。その主要な追加機能には、オフラインでのデータ準備（クリーニング、チャンкиング、ベクトル化、類似性検索、ランキングを含む）による取得プロセスと、プロンプト拡張およびグラウンドティングによる生成プロセスが含まれます。
- **AgentOps (Agent and Operations)**：GenAIOps のサブカテゴリであり、エージェントの効率的な運用化に焦点を当てます。その主な追加コンポーネントには、内部および外部のツール管理、エージェントブレインプロンプト（目標、プロファイル、指示）とオーケストレーション、メモリ、そしてタスク分解が含まれます。

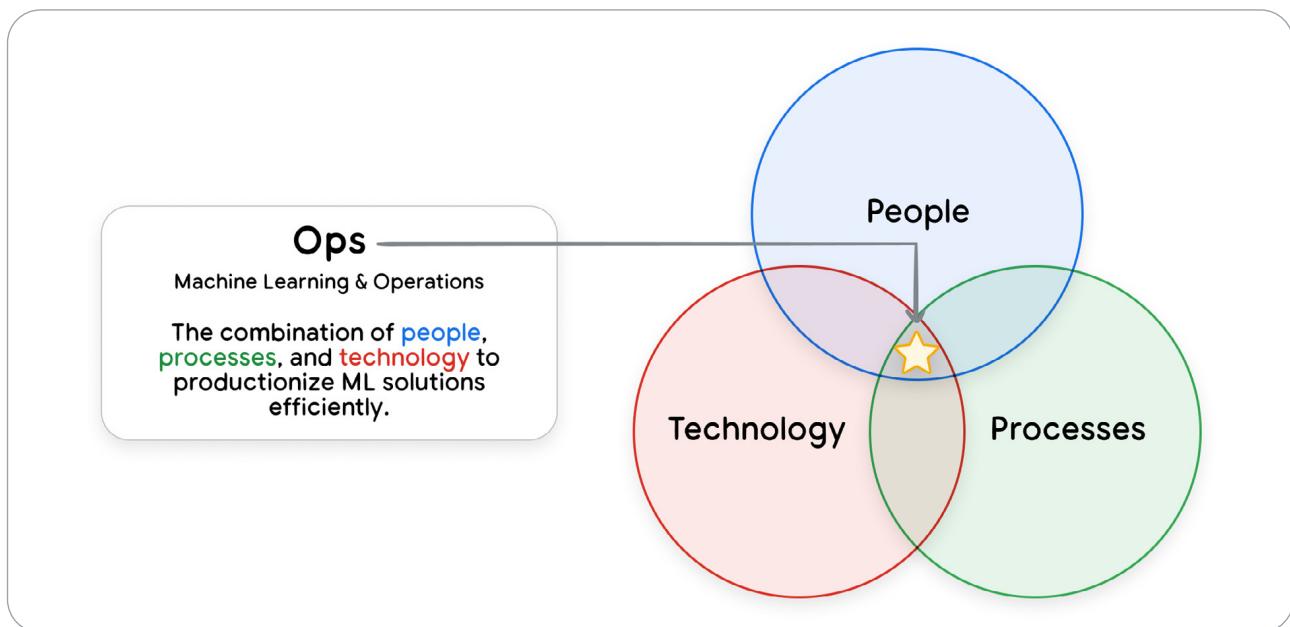


図2：各「Ops」は、テクノロジー、プロセス、そして人に関するものである<sup>14</sup>

これらの「Ops」はすべて、本質的には、機械学習ソリューションを本番環境に効率的にデプロイするために、人、プロセス、そしてテクノロジーが調和して連携するものです。Ops が単なるテクノロジーにとどまらないことを認識するのは極めて重要です。それは単に ML パイプラインを構築し提供することではありません。成功する Ops の実装は、顧客の運用モデル、既存の事業部門、そして全体的な組織構造を考慮し、より深く掘り下げます。この包括的なアプローチは、テクノロジーが顧客の特定のニーズに適合し、ビジネスにシームレスに統合されて価値を最大化することを保証します。

次のセクションではエージェント評価について詳しく説明します。これは、AgentOps と自動化が有用なメトリクスを捉える上で重要な部分を占めます。そこに進む前に、思考実験から始めましょう。新しいエージェントのために、本番環境で A/B 実験を設定することを想像してみてください。処置群は新しいエージェントを使用し、対照群は使用しません。そのシナリオでは、処置群の方が優れていると判断するために、どのようなメトリクスを測定しますか？プロジェクトの投資収益率を決定するために、どのようなメトリクスを測定しますか？それは達成された目標ですか、それとも売上合計ですか、あるいはユーザージャーニーにおける一連の重要なステップですか？これらのメトリクスは、より詳細なエージェント評価メトリクスに加えて、理解され、計装化され、そして容易に分析可能でなければなりません。

## エージェントの成功指標

メトリクスは、エージェントの構築、モニタリング、そしてリビジョンの比較にとって極めて重要です。収益やユーザーインゲージメントのようなビジネスメトリクスは、おそらくエージェント自体のスコープ外ですが、これらがあなたのエージェントにとっての北極星指標であるべきです。

ほとんどのエージェントは目標を達成するように設計されているため、目標達成率は追跡すべき主要なメトリクスです。同様に、一つの目標は、いくつかの重要なタスクや重要なユーザーインタラクションに分解されることがあります。これらの重要なタスクとインタラクションは、それぞれ独立して計装化され、測定されるべきです。

したがって、エージェント自体の詳細に入る前に、私たちはすでに、ダッシュボードで容易に追跡できるはずのいくつかのメトリクスを特定しています。それは、各ビジネスメトリクス、目標、または重要なインターフェースは、試行回数、成功回数、各種の率など、馴染みのある形式で集計されるものです。加えて、あらゆるアプリケーションテレメトリシステムから取得できる、レイテンシ、エラーなどのメトリクスも、エージェントを追跡する上で非常に重要です。

これらのメトリクスはいずれもエージェントに特有のものではなく、どんなソフトウェアでも追跡可能ですが、エージェントの構築者にとってはさらに重要です。決定論的コードは指示されたことしか行いませんが、それに対してエージェントは、大量のデータで訓練された LLM に依拠することで、はるかに多くのことを実行できます。これらのハイレベルなメトリクスを計装化することは、オブザーバビリティの重要な部分です。これらはエージェントにとっての重要業績評価指標と考えることができます。集計レベルでのオブザーバビリティ、つまり、あなたのエージェントに対するより俯瞰的な視点を可能にします。

ヒューマンフィードバックもまた、追跡すべきより重要なメトリクスの一つです。エージェントやタスクの文脈における、単純な やユーザーフィードバックフォームは、エージェントがどこでうまく機能し、どこで改善が必要かを理解する上で大いに役立ちます。このフィードバックは、消費者向けシステムのエンドユーザーからだけでなく、エージェントをレビューする従業員、QA テスター、そしてプロセスやドメインのエキスパートからも得られます。

より詳細なオブザーバビリティもまた、エージェントの構築にとって非常に重要です。つまり、エージェントが何をしていて、なぜそれをしているのかを視覚化し理解できることです。エージェントは「トレース」を導入することでそれを計装化することができ、極めて重要なタスクやユーザーインタラクションだけでなく、エージェントのすべての内部動作をログに記録できます。概念的には、すべての内部ステップをメトリクスとして測定することも可能ですが、それが実際に行われることは稀です。その代わり、これらの詳細なトレースは、メトリクスや手動テストで問題が示されたときにエージェントをデバッグするために使用されます。詳細を掘り下げて、何が問題だったのかを確認できるのです。

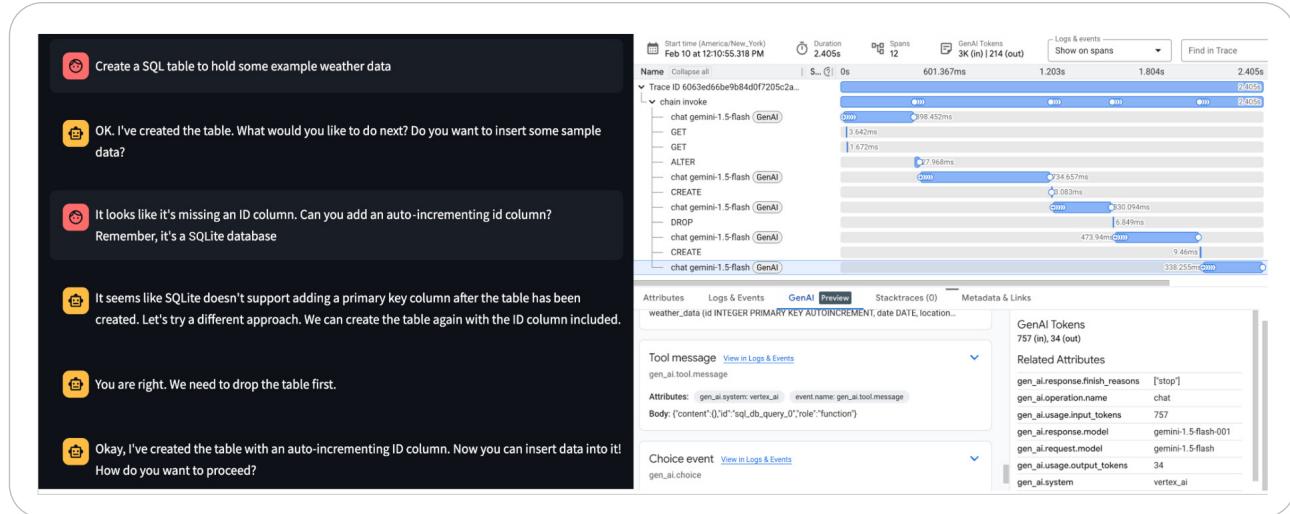


図 3：各ツールを備えたエージェントのトレースと LLM の OpenTelemetry スパンを示す Cloud Observability の例<sup>15</sup>

これまで、ビジネスメトリクス、目標、タスク、ヒューマンフィードバック、そしてトレースについて話してきました。これらはすべて、本番環境におけるエージェントの行動とその影響を理解するための方法です。エージェントを開発する際には、手動テストに加えて、自動テストが長期的にははるかに効率的であり、エージェントの振る舞いに対するより深い洞察を提供します。

# エージェント評価

フルーフォブコンセプトと本番環境に対応した AI エージェントとの間のギャップを埋めるためには、堅牢な自動評価フレームワークが不可欠です。主に最終的な出力に焦点が当てられる生成モデルの評価とは異なり、エージェント評価では意思決定プロセスに対するより深い理解が求められます。

エージェント評価は、本章で議論する 3 つのコンポーネントに分解できます。

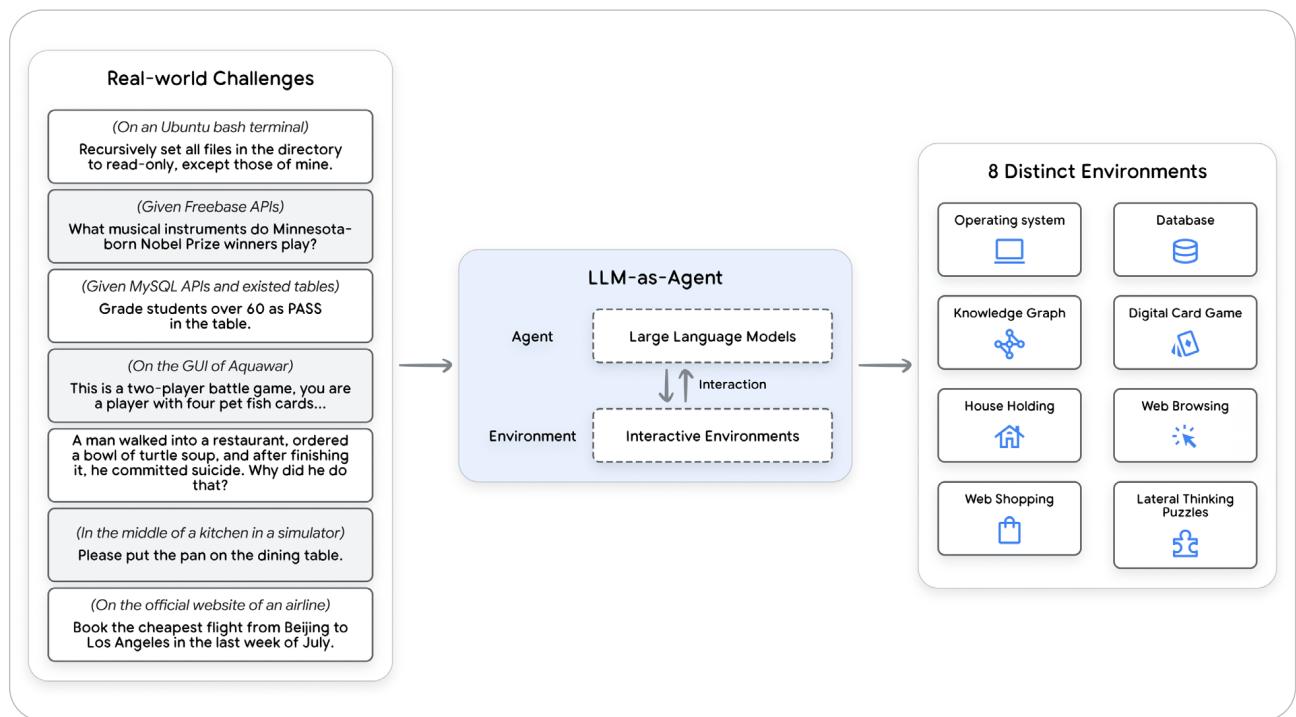
- 1. エージェントの能力の評価**：指示を理解し、論理的に推論する能力など、エージェントの核となる能力を評価する。
- 2. トラジェクトリとツール使用の評価**：エージェントが解決策に到達するまでにとるステップを分析する。これにはツールの選択、戦略、アプローチの効率性が含まれる。
- 3. 最終応答の評価**：エージェントの最終的な出力の品質、関連性、そして正確性を評価する。

## エージェントの能力の評価

あなた自身の特定のエージェントのユースケースを評価する前に、公開されているベンチマークや技術レポートを参照することで、ユースケースを構築する際に考慮すべき核となる能力や限界についての洞察を得ることができます。モデルのパフォーマンス、ハリシネーション、ツール呼び出し、計画といった、エージェントの最も基本的な能力のほとんどに対して、公開ベンチマークが存在します。

例えば、適切なツールを選択し使用する能力であるツール呼び出しは、Berkeley Function-Calling Leaderboard (BFCL)<sup>16</sup> や、一般的な誤りについても概説している T-bench<sup>17</sup> のようなベンチマークによって実証されています。もう一つの例として、PlanBench<sup>18</sup> は、いくつかのドメインと特定の能力にわたって、計画と推論を評価することを目的としています。

しかし、考慮すべき能力はツール呼び出しと計画だけではありません。エージェントは、その LLM や他の各コンポーネントから振る舞いを継承します。同様に、エージェントとユーザーのインタラクションは、従来の対話設計システムやワークフローシステムに歴史的背景を持ち、したがって、これらのシステムの有効性を判断するために使用されるメトリクスや測定基準のセットを継承することができます。

図 4 : AgentBench の概要図<sup>19</sup>

AgentBench<sup>19</sup> のような包括的なエージェントベンチマークは、いくつかのシナリオにわたるエンドツーエンドのパフォーマンスを捉えようとしています。この方法は、シナリオがあなたのエージェントのユースケースを反映している場合にはより現実的ですが、エージェントが実装していない能力をテストしている場合にはそうではありません。環境、ツール、指示、そしてユースケースの要件を、具体的であると同時に一般的でもあるような方法でシミュレートすることは困難です。企業や組織は、特化したユースケースのための公開ベンチマークを設置しています。例えば、Adyenのデータアナリストリーダーボード「DBAStep」<sup>20</sup>のようなものです。これらは、もしかしたらその評価アプローチとリーダーボードに掲載されているエージェントの両方を理解していれば、よりターゲットを絞った評価を提供してくれるかもしれません。

公開ベンチマークは、何が可能であるかの感覚を掴み、注意すべき落とし穴を特定するための、価値ある出発点です。ほとんどのベンチマーク評価には、一般的な失敗モードに関する議論が含まれており、それがあなた自身の、ユースケース固有の評価フレームワークを設定する上での指針となり得ます。

公開評価の先には、エージェントが意図した通りに動作することを保証するため、様々なシナリオにわたってその振る舞いを評価することが求められます。これは、エージェントとのインタラクションをシミュレートし、それがどのように応答するかを評価するということです。これには、最終応答の評価と、そこに至るまでの一連のステップ（トラジェクトリ）の評価が含まれます。これらが、私たちが最初に始めることを推奨する、最も一般的で実践的なアプローチです。これら以外にも、サブコンポーネントのより詳細な分析や、より広範なアプローチのために使用できる評価手法は数多く存在します。

ソフトウェアエンジニアは、これをコードの自動テストと比較するでしょう。自動テストへの投資は時間を節約し、構築しているソフトウェアに対する自信を与えてくれます。エージェントの場合、この自動化は時間と自信の両面で、より早く効果を発揮します。ソフトウェアテストの場合と比べても、エージェントが遭遇するであろうユースケースを正確に表現するためには、評価データセットをキュレーションすることが極めて重要になります。

## トラジェクトリとツール使用の評価

エージェントは通常、ユーザーに応答を返す前に、いくつかのアクションを実行します。例えば、用語の曖昧さを解消するためにユーザー入力をセッション履歴と比較したり、ポリシー・シード・キュメントを参照したり、ナレッジベースを検索したり、チケットを保存するためにAPIを呼び出したりします。それらのアクションはそれぞれが一つの経路上のステップであり、アクションの「トラジェクトリ」とも呼ばれます。あなたのエージェントが何かを行うたびに、その内部ではアクションのトラジェクトリが存在します。

エージェントがとると期待されるトラジェクトリと、エージェントが実際にたどったトラジェクトリを比較することは、アプリケーションをデバッグし、エラーや非効率性を特定し、最終的にパフォーマンスを改善したいと考える開発者にとって特に有用です。

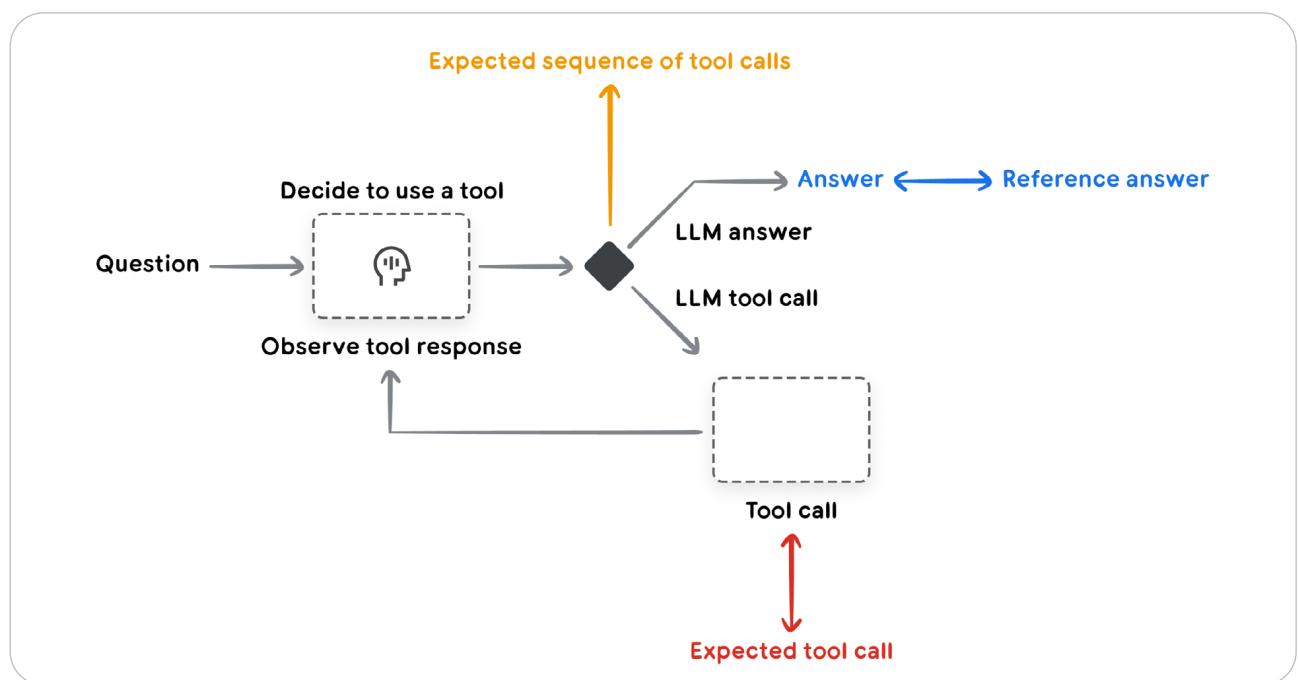


図 5 : LangSmith による最終評価とトラジェクトリのダイアグラム<sup>21</sup>

以下の 6 つの正解データに基づく自動トラジェクトリ評価は、エージェントのパフォーマンスを評価するための異なる視点を提供します。

- 1. 完全一致 :** AI エージェントに、理想的な解決策を完全に反映したアクションのシーケンス（「トラジェクトリ」）を生成させることを要求する。これは最も厳格なメトリクスであり、期待される経路からの逸脱を一切許さない。
- 2. 順序付き一致 :** このメトリクスは、ペナルティを科されない余分なアクションを許容しつつ、期待されるトラジェクトリを完了するエージェントの能力を評価する。成功は、追加アクションの柔軟性を持ちながら、核となるステップを順序通りに完了することによって定義される。

- 3. 順序不問一致：**順序付き一致と比較して、このメトリクスは順序を無視する。エージェントが必要なアクションをすべて含んでいるかを問うが、実行されたアクションの順序は考慮せず、余分なステップも許容する。
- 4. 適合率：**予測されたトラジェクトリ内のツール呼び出しのうち、正解トラジェクトリによれば、実際にどれだけが関連性があり正しいか。
- 5. 再現率：**正解トラジェクトリからの必須のツール呼び出しのうち、実際にどれだけが予測されたトラジェクトリで捉えられているか。
- 6. 単一ツール使用：**特定のアクションがエージェントのトラジェクトリ内に存在するかを理解する。このメトリクスは、エージェントが特定のツールを利用することをすでに学習したかどうかを理解するのに役立つ。

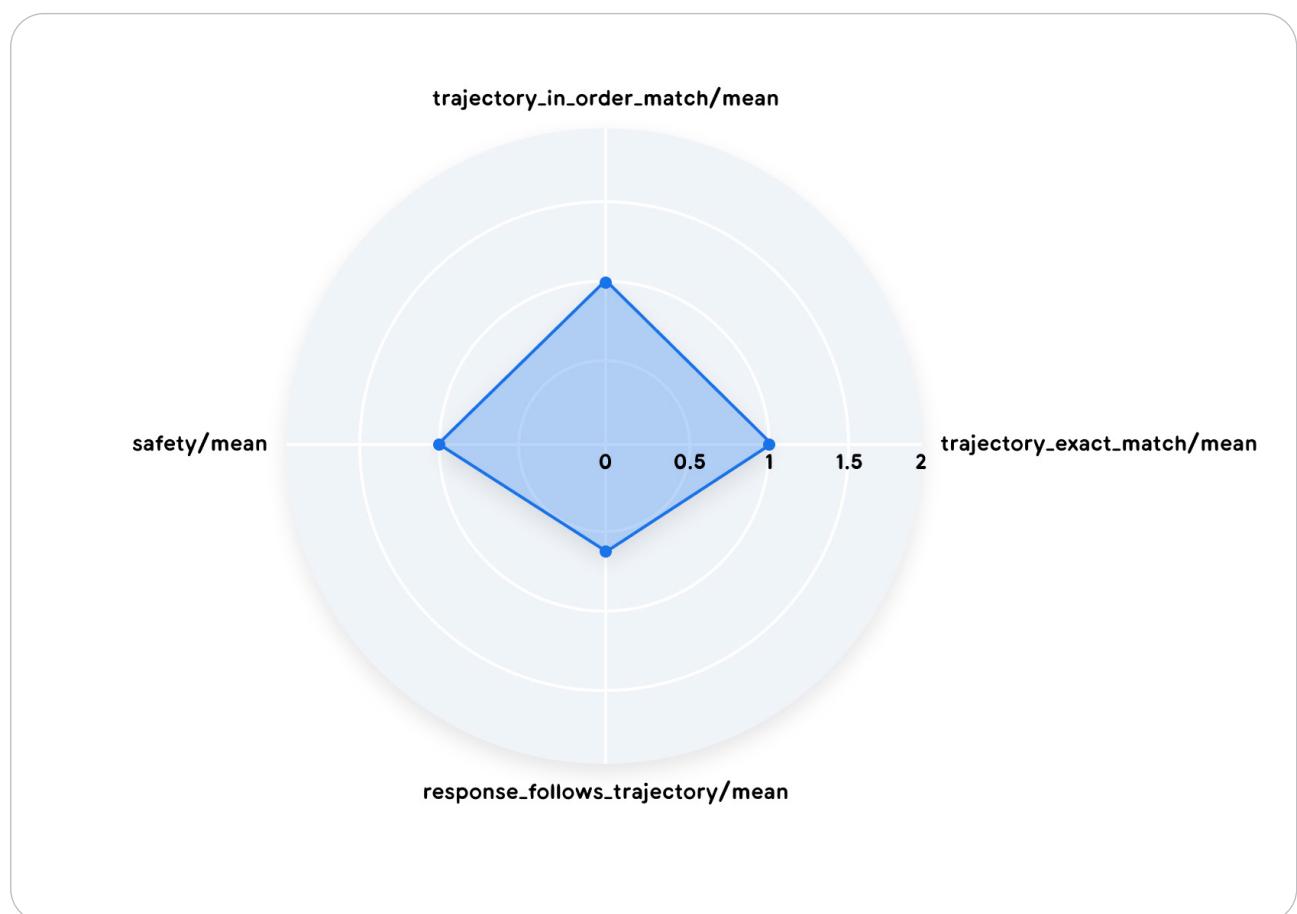


図 6：単一のトラジェクトリ評価をいくつかのメトリクスでプロットしたレーダーチャート<sup>24</sup>

これらのメトリクスは、あなたのエージェントのトラジェクトリを分析しデバッグするための異なる視点と考えることができます。各メトリクスは独自の視点を提供しますが、すべてがあらゆる状況に関連するわけではありません。例えば、理想的なトラジェクトリへの厳密な準拠を要求するユースケースもあれば、より創造的な逸脱を許容するユースケースもあります。この評価アプローチの明確な限界は、これが機能するためには正解トラジェクトリをあらかじめ用意しておく必要があるという点です。ここで議論されている正解データに基づく自動トラジェクトリ評価は、一般的なライブラリで広く利用されていますが、研究では、より効率的な評価のためにエージェント自動評価器の使用が進められています。例えば、「Agent as a Judge, 2024」などです。

## 最終応答の評価

最終応答評価は、単純な問い合わせに集約されます。つまり、「あなたのエージェントは目標を達成したか?」ということです。これを測定するために、特定のニーズに合わせてカスタマイズした成功基準を定義することができます。例えば、小売チャットボットが製品に関する質問に正確に答えているか、あるいは調査エージェントが適切なトーンとスタイルで調査結果を効果的に要約しているか、などを評価できます。

このプロセスを自動化するためには、自動評価器を使用できます。自動評価器とは、審査員として機能するLLMのことです。入力プロンプトと生成された応答が与えられると、それはユーザーが提供した一連の基準に照らして応答を評価することにより、人間による評価を模倣します。この評価が機能するためには、正解データが存在しないことを踏まえ、評価基準を非常に正確に定義する必要がある点を考慮することが極めて重要です。なぜなら、それがあなたの評価が注目する核となる部分だからです。様々なライブラリで定義済みの基準が多数見つかりますが、それらをたたき台として、あなた自身の「良い」の定義するために微調整してください。

## ヒューマンインザループ評価

エージェント開発とエージェント評価の分野は急速に進化しています。AI エージェントの評価には、明確な目的の定義、現実的な環境の設計、確率的な振る舞いの管理、そして特に社会的に影響の大きいアプリケーションにおける公平性の確保とバイアス緩和といった、重大な課題が存在します。したがって、以前に議論した（定義済みのメトリクスと自動評価器を含む）自動評価と並行して、ヒューマンインザループのアプローチを取り入れることが極めて重要です。ヒューマンインザループは、主観的な判断や創造的な問題解決を必要とするタスクにとって価値があり、また、あなたの自動評価アプローチが実際に機能し、あなたの好みに合致しているかを確認しキャリブレーションする役割も果たします。

主な利点には以下が含まれます。

- ・**主觀性**：人間は、創造性、常識、ニュアンスといった、定量化が困難な品質を評価できる。
- ・**文脈の理解**：人間の評価者は、エージェントのアクションが持つより広い文脈とその影響を考慮できる。
- ・**反復的な改善**：ヒューマンフィードバックは、エージェントの振る舞いと学習プロセスを洗練させるための貴重な洞察を提供する。
- ・**評価者の評価**：ヒューマンフィードバックは、あなたの自動評価器をキャリブレーションし、洗練させるためのシグナルを提供できる。

ヒューマンインザループ評価を実装するには、以下の手法を検討してください。

- ・**直接評価**：人間の専門家が、特定のタスクにおけるエージェントのパフォーマンスを直接評価またはスコア付ける。
- ・**比較評価**：専門家が、エージェントのパフォーマンスを他のエージェントや以前のイテレーションのものと比較する。
- ・**ユーザー調査**：参加者がエージェントと対話し、その振る舞い、ユーザビリティ、そして全体的な有効性についてフィードバックを提供する。

## エージェント評価についてのさらなる考察

このセクションでは、実践的な観点からエージェント評価を取り上げますが、これは氷山の一角にすぎません。あなたのエージェントのための評価データを見つけるのは難しいかもしれません。合成データや審査員としてのLLMを使用することはできますが、それでも評価は不完全である可能性があります。また、例えば「LLM-as-a-Judge」メトリクスは、エージェントの推論や中間的なアクションよりも最終的な成果を優先する可能性があり、重要な洞察を見逃す恐れがあります。加えて、エージェントシステムの評価は対話システムやワークフローシステムに歴史的背景を持つため、複数のインタラクションを通じてタスクのパフォーマンスを向上させる能力など、エージェントの能力を評価するための手法やメトリクスをいかに継承するかについては、探求すべきことが非常に多くあります。マルチモーダル生成の評価は、さらなる複雑さをもたらします。画像、音声、および動画の評価は、それぞれ独自の評価手法とメトリクスを必要とします。最後に、実世界の環境は動的で予測不可能であるため、さらなる課題をもたらします。これにより、制御された環境でエージェントを評価することが困難になります。

今後、これらの未解決の課題を解決するため、エージェント評価の分野は急速に進化しています。主要なトレンドには、エージェントの推論の理解を優先するプロセスベースの評価への移行、スケーラビリティ向上のためのAI支援の評価手法の増加、そして実世界のアプリケーションコンテキストへのより強い焦点が含まれます。新しい標準化されたベンチマークの開発も勢いを増しており、エージェント間の客観的な比較を促進しています。その一方で、説明可能性と解釈可能性への重点の高まりは、エージェントの振る舞いに対するより深い洞察を提供することを目的としています。

評価手法	👍長所	👎短所
人間による評価	ニュアンスのある振る舞いを捉え、人的要因を考慮できる	主観的、時間がかかる、高コスト、スケールさせにくい
LLM-as-a-Judge	スケーラブル、効率的、一貫性がある	中間ステップを見落とす可能性、LLMの能力による制限
自動メトリクス	客観的、スケーラブル、効率的	全ての能力を捉えきれない可能性、恣意的な調整がされやすい

表1：エージェントの自動評価における長所と短所の比較

評価手法を継続的に洗練させ続けることによってはじめて、来るべき未来においてAIエージェントが責任を持って、効果的に、そして倫理的に開発・デプロイされることを保証できる、という点はこの時点で明確でしょう。

# 複数エージェントとその評価

前章で見てきたエージェント評価は、自律的な AI エージェントの有効性、信頼性、そして適応性を評価するもので、現在では主要な焦点分野として浮上しています。AI システムは大きな進化を遂げ、複数の特化型エージェントが協調して複雑な目的を達成するマルチエージェントアーキテクチャへと移行しています。

マルチエージェントシステムは、複雑な問題を解決するために協働する、それぞれが特定の分野に特化したエキスパートのチームのようなものです。各エージェントは独立したエンティティであり、場合によっては異なる LLM を使用し、それぞれが独自の役割とコンテキストを持ちます。エージェントは共通の目標を達成するために、コミュニケーションをとり、協調します。このアプローチは、一つの LLM がタスクのあらゆる側面を処理する、従来のシングルエージェントシステムとは異なります。

マルチエージェントシステムは、シングルエージェントシステムに比べて、以下のような複数の利点を提供します。

- ・**精度の向上**：エージェントが互いの作業を相互検証できるため、より正確な結果につながります。
- ・**効率の向上**：エージェントが並列で作業できるため、タスクの完了が迅速化されます。
- ・**複雑なタスクのより良い処理**：大きなタスクを、より管理しやすい小さなサブタスクに分解し、各エージェントが特定の側面に集中できます。
- ・**スケーラビリティの向上**：特化された能力を持つエージェントを追加することで、システムを容易にスケールできます。
- ・**フォールトトレランスの向上**：あるエージェントが失敗しても、他のエージェントがその責務を引き継ぐことができます。
- ・**ハルシネーションとバイアスの低減**：複数のエージェントの視点を組み合わせることで、システムはハルシネーションとバイアスの影響を低減し、より信頼性が高く信用できる出力につながります。

## マルチエージェントアーキテクチャの理解

従来のモノリシックな AI システムとは異なり、マルチエージェントアーキテクチャは、問題を特化型エージェントが処理する個別のタスクに分解します。各エージェントは定義された役割を持って動作し、他者と動的に対話することで、意思決定、知識検索、そして実行を最適化します。これらのアーキテクチャは、より構造化された推論、分散型の問題解決、そしてスケーラブルなタスク自動化を可能にし、シングルエージェントのワークフローからのパラダイムシフトをもたらします。

その核として、マルチエージェントシステムは、モジュール性、協調、そして階層の原則を活用して、堅牢な AI エコシステムを構築します。これらのシステム内のエージェントは、例えばその機能に基づいて次のように分類できます。

- ・**プランナーエージェント**：高レベルの目的を構造化されたサブタスクに分解する責任を負う。
- ・**レトリーバーエージェント**：外部ソースから関連データを動的に取得することで、知識獲得を最適化する。
- ・**実行エージェント**：計算を実行し、応答を生成し、あるいは API と対話する。
- ・**評価エージェント**：応答を監視・検証し、一貫性と目的との整合性を保証する。

これらのコンポーネントを通じて、マルチエージェントアーキテクチャは、単純なプロンプトベースのインターラクションを超越し、適応的で、説明可能で、かつ効率的な AI 駆動のワークフローを可能にします。

## マルチエージェントの設計パターンとそのビジネスインパクト

効果的なマルチエージェントアーキテクチャを設計するために、特定のデザインパターンが登場しました。これらのパターンは、インタラクションプロトコル、委任メカニズム、そして役割分担を定義し、企業がAI駆動の自動化を構造化された方法で実装することを可能にします。

一般的なデザインパターンには、以下のようないわゆるがあります。

マルチエージェントシステムのタイプ	説明	例
逐次	各エージェントが次のエージェントに出力を渡す前に自身のタスクを完了するという、逐次的な方法でエージェントが動作する。	組立ライン。各作業員が特定の作業を行い、製品を次の作業員に渡す。
階層型	エージェントが階層構造で組織される。「マネージャー」エージェントがワークフローを調整し、「ワーカー」エージェントにタスクを委任する。	リーダーエージェントが戦略的な意思決定を行い、フォロワーエージェントがリーダーの指示に基づいてタスクを実行するシステム。
協調型	エージェントが協調して作業し、共通の目標を達成するために情報とリソースを共有する。	プロジェクトに取り組む研究者チーム。各メンバーが自身の専門知識と洞察を提供し貢献する。
競合型	エージェントが最良の結果を達成するため互いに競合することがある。	LLMが『Overcooked-AI』のようなゲームで個々のプレイヤーとして振る舞う。リソースを奪い合い、個々のパフォーマンスを最適化しながら、共有された目標を達成するためにアクションを協調させなければならない。

表2：マルチエージェントシステムのタイプ比較

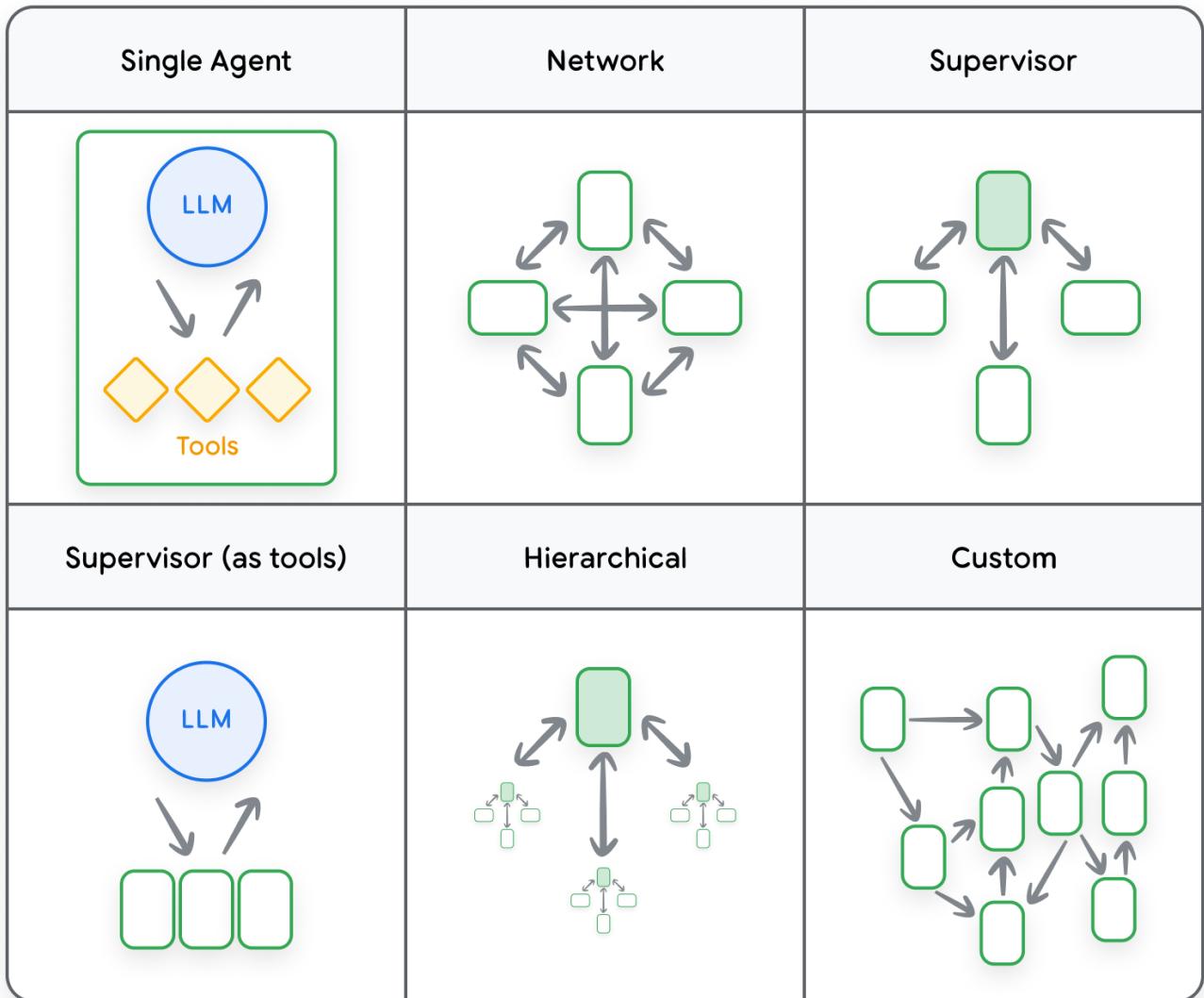


図 7：様々なマルチエージェントプロロー（LangGraph ドキュメンテーションより）<sup>23</sup>

デザインパターンの選択は、特定のアプリケーションと、エージェント間の望ましいインタラクションのレベルによって決まります。

企業はこれらのパターンから、運用上のボトルネックを削減し、知識検索を改善し、自動化の信頼性を高めることによって、利益を得ます。マルチエージェントシステムは、意思決定とワークフロー実行におけるアジリティを確保しつつ、企業が AI のデプロイをスケールさせることを可能にします。

## エージェントの重要なコンポーネント

LLM ベースの AI エージェントのアーキテクチャは、自律的な運用と知的なインタラクションに不可欠な、相互に関連するいくつかのコンポーネントで構成されています。

- ・**インタラクションラッパー**：このコンポーネントは、エージェントとその環境との間のインターフェースとして機能し、コミュニケーションを管理し、様々な入出力モダリティに適応します。
- ・**メモリ管理**：これには、当面のコンテキストのための短期ワーキングメモリ、キャッシュ、セッションが含まれます。また、エピソード、事例、スキル、または参照データとして、学習したパターンや経験を保存するための長期ストレージも含むことがあります。さらに、短期メモリのどの項目（例：ユーザーの好み）を長期メモリ（例：ユーザープロファイル）にコピーすべきか、またそれをエージェント、タスク、またはセッション間で共有できるかを決定するための「リフレクション」も含まれます。
- ・**認知機能**：これは多くの場合、Chain-of-Thought、ReAct、推論、思考、またはプランナーサブシステムによって支えられています。これにより、エージェントは複雑なタスクを論理的なステップに分解し、自己修正を行うことができます。場合によっては、不確かな場合に明確化のための質問をする、ユーザー意図の精緻化もこれに含まれます。
- ・**ツール統合**：このサブシステムにより、エージェントは外部ツールを利用し、自然言語処理の範囲を超えてその能力を拡張できます。動的ツールレジストリは、探索、登録、そして「Tool RAG」を可能にします。
- ・**フロー／ルーティング**：これは他のエージェントとの接続を管理し、マルチエージェントシステム内での動的なネイバーアクションと効率的なコミュニケーションを促進します。これは、タスクをバックグラウンドエージェントに委任したり、ユーザーインターラクションを別のエージェントにハンドオフしたり、あるいはエージェントをツールとして使用したりといった形で実装されることがあります。
- ・**フィードバックループ／強化学習**：これらは、インタラクションの結果を処理し、意思決定戦略を洗練させることによって、継続的な学習と適応を可能にします。生成 AI エージェントの場合、これが従来の強化学習訓練の形をとることは稀ですが、過去のパフォーマンスマトリックスを将来の意思決定に組み込むことは可能です。
- ・**エージェント間コミュニケーション**：エージェント間の効果的なコミュニケーションは、マルチエージェントシステムの成功にとって極めて重要です。エージェント間コミュニケーションプロトコルは、エージェント間の構造化された効率的なコミュニケーションを促進し、合意形成を可能にし、複雑な問題に協調して取り組むことを可能にします。

- ・リモートエージェントコミュニケーション：組織内でのエージェント間コミュニケーションは、エージェントがメッセージ、タスク、知識を共有するために不可欠です。マルチエージェントシステムにリモートエージェントが含まれるようになると、コミュニケーションプロトコルはさらに重要になります。非同期タスクとセッションは永続的である必要があり、エンドユーザーがセッション中でない間も通知によって更新されなければなりません。エージェント間の交渉では、ユーザーをセッションに参加させることや、サポートされているUX能力を考慮する必要があります。
- ・エージェント&ツールレジストリ（メッシュ）：少数のツールやエージェントを超えてスケールするにつれて、ツールやエージェントの「メッシュ」から、探索、登録、管理、選択、および利用を行うための堅牢なシステムが必要になります。極めて重要なのは、ツールとエージェントのオントロジーと説明、それらの能力と要件、そしてパフォーマンスマトリクスです。あなたのエージェントは計画を立て、そのようなシステムからどのツールまたはどのエージェントを使用するかを選択し、それらの選択はシステム内のデータによって情報提供されます。

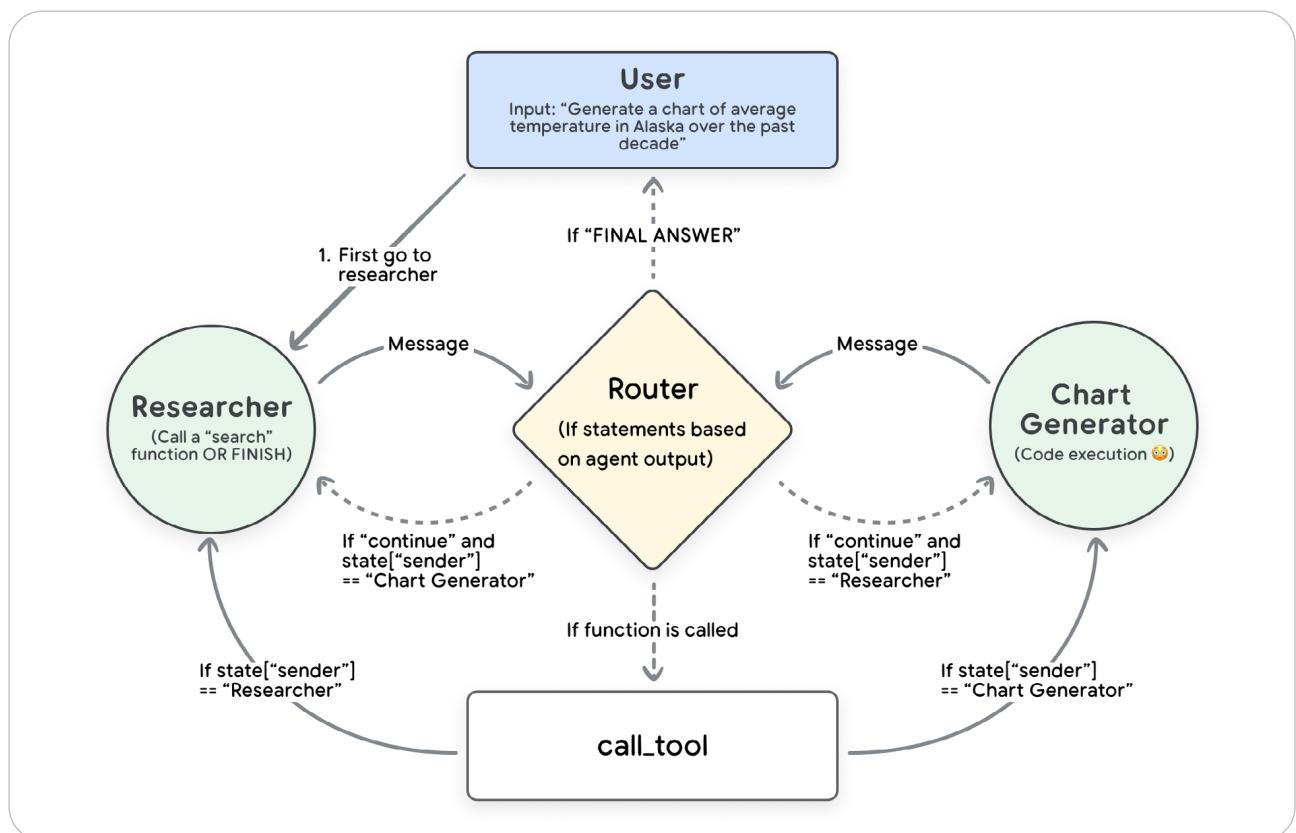


図 8：ユーザーと複数の自己協調エージェントとのインタラクションのプロセス<sup>24</sup>

これらのアーキテクチャ要素は、マルチエージェントシステム内における LLM エージェントの、自律的な運用と協調能力のための堅牢な基盤を提供します。

## マルチエージェントシステムの課題

マルチエージェントシステムは数多くの利点を提供する一方で、いくつかの課題にも直面しています。

- ・**タスクコミュニケーション**：こんにちのほとんどのエージェントフレームワークは、構造化された非同期タスクではなく、メッセージでコミュニケーションを行います。
- ・**タスク割り当て**：複雑なタスクを異なるエージェント間で効率的に分割することは困難な場合があり、フィードバックループの実装はしばしば開発者に委ねられます。
- ・**推論の協調**：エージェントに効果的に議論させ、共に推論させるためには、洗練された協調メカニズムが必要となります。
- ・**コンテキストの管理**：エージェント間のすべての情報、タスク、会話を追跡し続けることは、非常に困難な作業になります。
- ・**時間とコスト**：マルチエージェントのインタラクションは、計算コストが高く、時間がかかることがあります。これは、より高い実行時価格とより多くのユーザーインテンシティにつながります。
- ・**複雑性**：マイクロサービスアーキテクチャが各マイクロサービスにより高い柔軟性と単純さを許容するのと同じように、システム全体としては通常、より複雑になります。

これらの課題に対処することは、堅牢で効率的なマルチエージェントシステムを開発する上で極めて重要です。

## マルチエージェントの評価

幸いなことに、マルチエージェントシステムの評価は、シングルエージェントシステムの評価の明確な延長線上にあります。エージェント成功メトリクスは変わりません。北極星となるビジネスメトリクス、目標や重要なタスクの成功メトリクス、そしてレイテンシやエラーといったアプリケーションテレメトリメトリクスなどがそれに含まれます。マルチエージェントシステムをトレースで計装化することは、複雑なインタラクションの間に何が起きているかをデバッグし、理解する助けとなります。

「エージェント評価」のセクションでは、エージェントの自動評価における2つの最良のアプローチとして「トラジエクトリの評価」と「最終応答の評価」を議論しましたが、これはマルチエージェントシステムの場合も同様です。マルチエージェントシステムの場合、アクションのトラジエクトリには、複数の、あるいはすべてのエージェントが含まれることがあります。たとえ複数のエージェントが1つのタスクで協調するとしても、最終的には単一の最終回答がユーザーに返され、それは個別に評価することができます。

マルチエージェントシステムは、おそらくより多くのステップを持つため、各ステップでドリルダウンして評価することができます。各エージェントを個別に評価することも、システム全体として評価することも可能です。トラジエクトリ評価は、まさにこれを実行するためのスケーラブルなアプローチです。

マルチエージェントシステムに特有の、問うべきいくつかの質問があります

- ・**協力と協調**: エージェントは、共通の目標を達成するために、どれくらいうまく協働し、アクションを協調させているか?
- ・**計画とタスク割り当て**: 我々は正しい計画を立て、それに従ったか? 子エージェントは主計画から逸脱したり、袋小路に迷い込んだりしなかったか?
- ・**エージェントの利用**: エージェントは、適切なエージェントを選択し、そのエージェントをツールとして使用するか、バックグラウンドタスクを委任するか、あるいはユーザーを転送するかを、どれくらい効果的に選択しているか?
- ・**スケーラビリティ**: エージェントを追加するにつれて、システムの品質は向上するか? レイテンシは低下するか? 我々はより効率的になっているか、それとも非効率になっているか?

これらのタイプの質問は、開発者がマルチエージェントシステムの改善領域を特定するための指針となり得ます。あなたはこれらの質問に、シングルエージェントシステムで使用するのと同じツールを使って答えることになりますが、分析の複雑性は増します。

# エージェント型 RAG： 検索拡張生成における重要な進化

マルチエージェントアーキテクチャにおける重要な進歩は、エージェント型検索拡張生成です。従来の RAG パイプラインは、ベクトルデータベースから知識を検索し、それを統合のために LLM に入力するという静的なアプローチに依存しています。しかし、このアプローチは、曖昧な、マルチステップの、あるいは多角的なクエリを扱う際にはしばしば失敗します。

エージェント型 RAG は、反復的な推論に基づいて検索を能動的に精緻化する、自律的な検索エージェントを導入します。これらのエージェントは、以下の方法で検索を強化します。

- ・**コンテキストを考慮したクエリ拡張**：単一の検索パスに依存する代わりに、エージェントは複数のクエリの精緻化を生成し、より関連性が高く包括的な結果を取得する。
- ・**マルチステップ推論**：エージェントは複雑なクエリをより小さな論理的ステップに分解し、情報を逐次的に取得して構造化された応答を構築する。
- ・**適応的なソース選択**：単一のベクトルデータベースからデータを取得する代わりに、検索エージェントはコンテキストに基づいて最適な知識ソースを動的に選択する。
- ・**検証と修正**：評価エージェントは、取得された知識を最終応答に統合する前に、ハルシネーションや矛盾がないか相互検証する。

このアプローチは、応答の精度、説明可能性、そして適応性を大幅に向上させ、法務調査、科学的発見、そしてビジネスインテリジェンスのような分野で複雑な知識検索タスクを扱う企業にとって、極めて重要なイノベーションとなっています。

## エージェント型 RAG とその重要性

エージェント型 RAG（検索拡張生成）は、RAG の強みと AI エージェントの自律性を組み合わせた、先進的なアプローチです。従来の RAG システムは、LLM の応答を強化するために、外部の知識ソースから関連情報を取得します。エージェント型 RAG は、インテリジェントエージェントを用いて取得プロセスをオーケストレーションし、取得された情報を評価し、それを最適に利用する方法について意思決定を行うことで、これをさらに一步進めます。

エージェント型 RAG は、従来の RAG に比べて、以下のような複数の利点を提供します。

- ・**精度の向上**：エージェントは、取得された情報の品質を評価し、どのソースを信頼するかについて意思決定できるため、より正確で信頼性の高い応答につながる。
- ・**文脈理解の強化**：エージェントは、ユーザーのクエリと取得された情報のコンテキストを考慮して、より関連性が高く意味のある応答を生成できる。
- ・**適応性の向上**：エージェントは、変化する情報のニーズに適応し、最新で最も関連性の高い情報を提供するために検索戦略を動的に調整できる。この適応性は、ヘルスケア、金融、法務調査など、情報が絶えず進化する複雑なドメインにおいて極めて重要である。

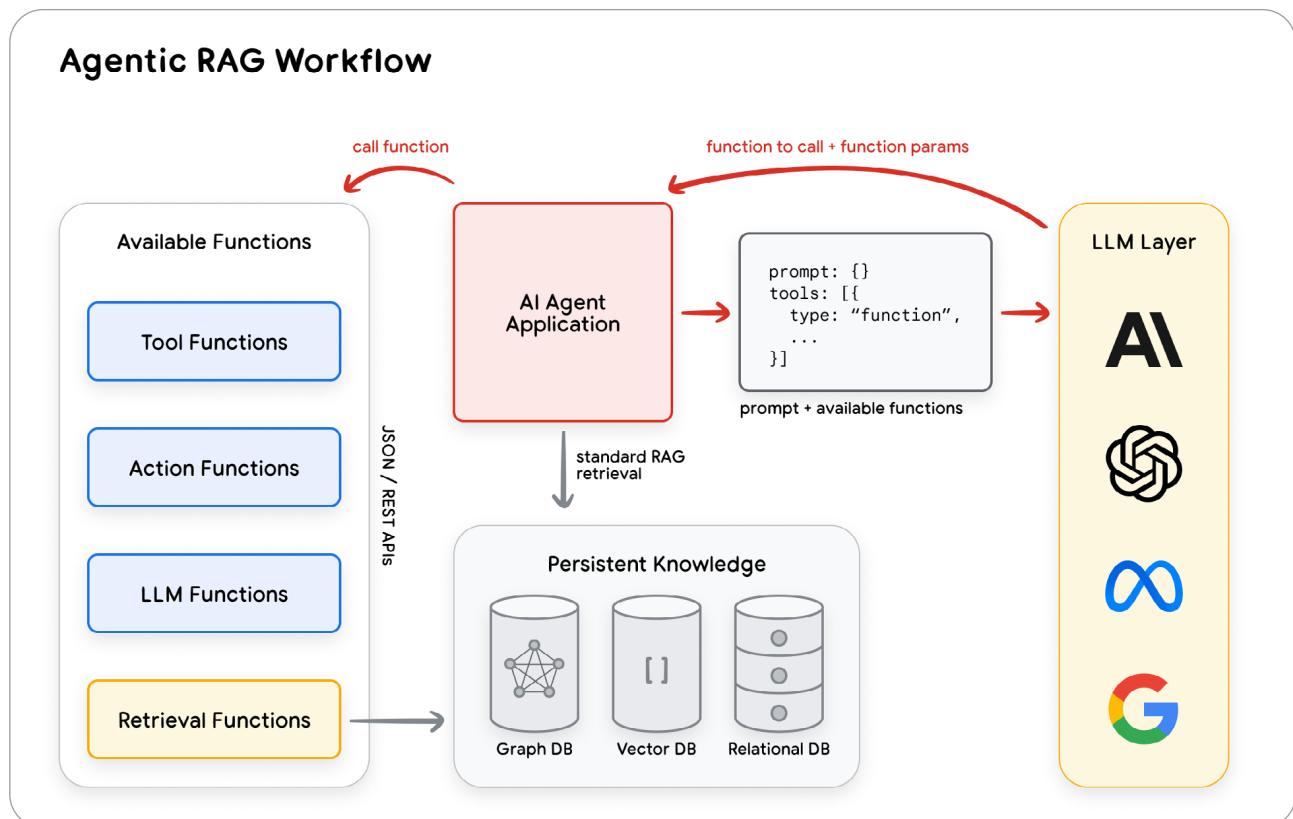


図 9：エージェント型 RAG のダイアグラム ([Vectorize.io](https://vectorize.io) より)<sup>25</sup>

エージェント型 RAG は、ヘルスケア、金融、法務調査など、情報が絶えず進化する複雑なドメインにおいて特に価値があります。例えば、ヘルスケア分野では、エージェント型 RAG は複雑な医療データベース、研究論文、そして患者記録をナビゲートするのに役立ち、医師に包括的で正確な情報を提供します。

## より良い検索、より良い RAG

ほとんどすべての RAG のアプローチは、関連データをインデックスし取得するために検索エンジンを必要とします。エージェントの導入は、クエリの精緻化、フィルタリング、ランキング、そして最終回答の洗練を可能にします。エージェント型 RAG のエージェントは、情報を取得するために複数の検索を実行しています。

既存の RAG 実装を最適化しようとしている開発者にとって、エージェントを導入する前に、まず検索結果（再現率で測定）を改善することが通常、最も価値があります。検索パフォーマンスを向上させるための主要なテクニックには、以下のものがあります。

- ・ソースドキュメントを解析し、チャunk化する：Vertex AI Layout Parser は、複雑なドキュメントレイアウト、埋め込みテーブル、チャートのような埋め込み画像を処理でき、セマンティックチャunk カーを使用してチャunkを見出しの階層を持つトピック上に維持します。
- ・チャunkにメタデータを追加する：シノニム、キーワード、著者、日付、タグ、カテゴリによって、検索でブースト、ベリー、フィルターをかけることが可能になります。これらにより、ユーザー やエージェントは検索結果をより詳細に制御できます。
- ・埋め込みモデルをファインチューニングするか、埋め込み空間を変更する検索アダプターを追加する：これらにより、検索可能なベクトルのインデックスが、汎用埋め込みモデルよりもあなたのドメインをより良く表現できるようになります。
- ・より高速なベクトルデータベースは検索品質を向上させ得る：埋め込みを検索するには、速度と精度の間でトレードオフを行う必要があります。超高速な Vertex AI Vector Search にアップグレードすることで、レイテンシと品質の両方を改善できます。
- ・ランカーを使用する：ベクトル検索は高速ですが近似的であり、数十から数百の結果を返すはずです。これらは、上位いくつかの結果が最も関連性が高い、または最良の回答であることを保証するために、より高度なシステムによって再ランキングされる必要があります。
- ・グラウンディングのチェックを実装する：グラウンディングされた生成の保護手段として、各フレーズが取得されたチャunk によって実際に引用可能であることを保証できます。

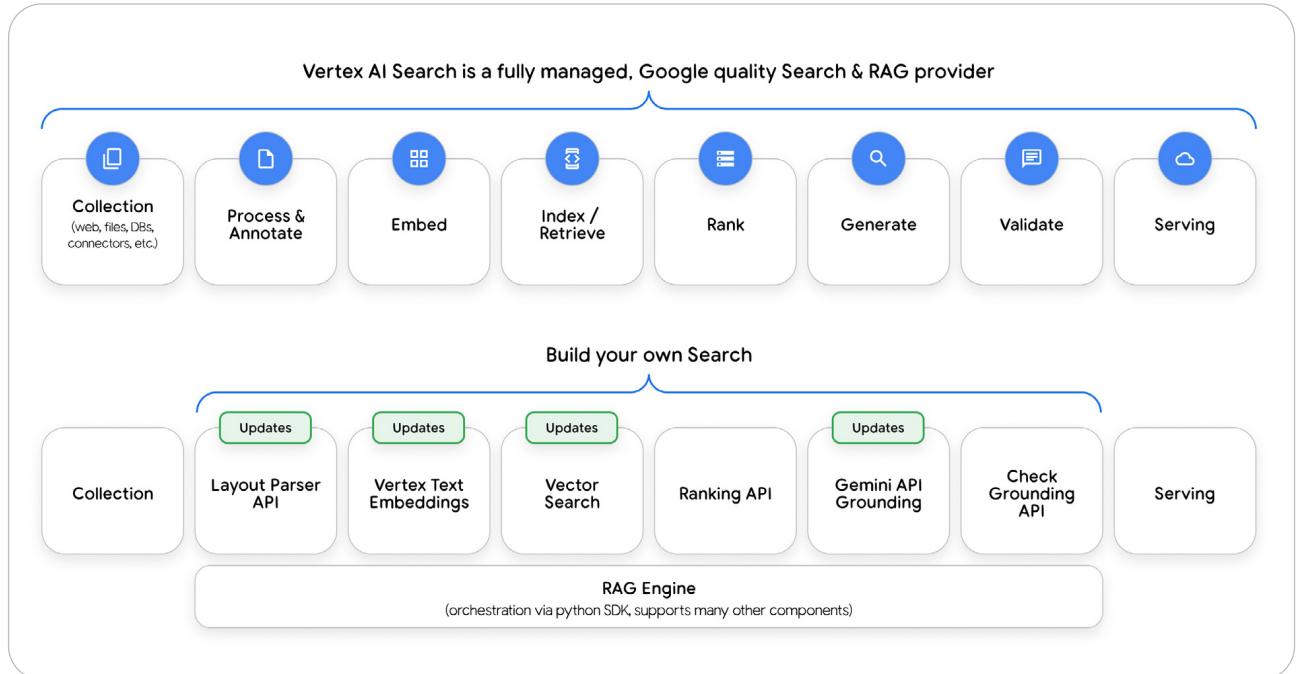


図 10：Vertex AI Search<sup>26</sup>、検索ビルダー API<sup>27</sup>、RAG エンジン<sup>28</sup>といった、一般的な RAG および検索コンポーネントのダイアグラム

Vertex AI Search<sup>26</sup> は、あなたのデータに対して Google 品質の検索を提供する強力な検索エンジンであり、あらゆる RAG またはエージェント型 RAG の実装で使用できます。上記の各コンポーネントは、一切の開発時間を必要とせず、Vertex AI Search 内で自動的に利用可能です。独自の検索エンジンを構築したい開発者のために、上記の各コンポーネントはスタンドアロン API<sup>27</sup> として公開されており、RAG エンジン<sup>28</sup> は LlamaIndex のような Python ベースのインターフェースでパイプライン全体を容易にオーケストレーションできます。

# エンタープライズにおけるエージェント

## エージェントの管理者

2025年はエージェントの年であり、これは特に企業において顕著になるでしょう。企業は、従業員の特定タスク遂行を支援したり、バックグラウンドで自律的に動作して自動化を行ったりするエージェントを開発・使用することになります。ビジネスアナリストは、業界のトレンドを容易に発見し、AIが生成した洞察に支えられた、説得力のあるデータ駆動型のプレゼンテーションを作成できます。人事チームは、確定拠出年金の選択のような複雑なタスクであっても、合理化されたオンボーディングによって従業員エクスペリエンスに革命を起こすことができます。ソフトウェアエンジニアは、プロアクティブにバグを特定・解決し、より高い効率で構築とイテレーションを行い、デブロイサイクルを加速させることができます。マーケターは、より深いパフォーマンス分析を可能にし、コンテンツ推薦を最適化し、キャンペーンを容易にファインチューニングして、より良い結果を達成できます。

2つのタイプのエージェントが登場しています：

1. 「アシスタント」：ユーザーと対話し、タスクを受け取り、それを実行し、ユーザーのもとに戻ってくるエージェント。Gems や GPTs のようなフレームワークによって普及した対話型エージェントは、通常このカテゴリに属します。アシスタントは、様々なタスクを支援できる汎用的なものもあれば、特定のドメインやタスクに特化したものもあります。例としては、会議のスケジュール調整、データ分析、コード作成、マーケティングのプレスリリース作成、営業担当者の販売機会の支援、あるいはユーザーの要求に応じて特定のトピックについて深い調査を行うエージェントなどが挙げられます。これらのエージェントは、要求された情報を返したり要求されたタスクを迅速に実行したりする同期的なものもあれば、（深い調査を行うエージェントのように）戻ってくるまでにより長い時間動作するものもあります。
2. 「オートメーションエージェント」：バックグラウンドで動作し、イベントをリッスンし、システムやデータの変更を監視し、そしてスマートな意思決定を行い行動するエージェント。アクションには、バックエンドシステムへの働きかけ、観察を検証するためのテストの実行、問題の修正、適切な従業員への通知などが含まれます。これは、まさに未来の自動化の中核と見なすことができます。過去には自動化のロジックを規定するために特別なコードを書く必要がありました。今や私たちは AI エージェントのスマートで汎用的な意思決定能力に頼り始めることができます。

単にタスクを実行させるためにエージェントを呼び出してその出力を待つのではなく、ナレッジワーカーはますますエージェントの管理者になっていくでしょう。彼らは複数のエージェントにタスクを割り当てて管理し、いずれかのエージェントが助けを必要としているか、あるいは先に進むために承認が必要かを確認し、特定のエージェントの出力を使って新しいタスクを開始し、長時間実行タスクの実行をモニタリングしてレビューし、正しい方向に誘導することになります。この種のバーチャルチームマネジメントを可能にする新しいユーザーインターフェースが登場し、タスクを実行し、他のエージェントを呼び出し（あるいは作成さえし）ながら動作するマルチエージェントシステムのオーケストレーション、モニタリング、そして管理を可能にするでしょう。

Google Agentspace は、この種の体験を提供し、以下のことを可能にすることを目指しています。

- ・ノーコード／ローコードインターフェース、またはフルコードフレームワークを使用して新しいエージェントを作成し、特化型アシスタントとオートメーションエージェントの両方を実装する。
- ・企業の管理者がエージェントアクセスを設定・管理する。
- ・適切な時に、適切なエージェントを呼び出す。
- ・チームマネジメントに適した UI で、複数のエージェントをモニタリング、管理、そしてオーケストレーションする。

## Google Agentspace

Google Agentspace<sup>29</sup> は、関連情報へのアクセスを促進し、複雑なエージェント型ワークフローを自動化することによって、企業の生産性を向上させるために設計された、一連の AI 駆動のツールを提供します。それは、Gemini の高度な推論能力、Google の検索インフラストラクチャの力、そして物理的な場所に問わらずエンタープライズデータへの安全なアクセスを活用します。

Aagentspace は、パーソナライゼーション、自動回答生成、文脈理解、そして包括的な情報検索といった領域でしばしば不十分となる、従来の知識管理システムに固有の限界に直接的に対処します。それは、ばらばらのコンテンツソースを統合し、グラウンドィングされたパーソナライズされた応答を生成し、運用ワークフローを合理化することによって、従業員に効率的な情報アクセスを提供します。主な機能には、多種多様なデータ形式をインジェストし、Software-as-a-Service プラットフォームからデータを同期し、アクセス制御された検索結果と AI が生成した回答を提供し、そして AI アシスタンスとインテリジェントエージェントを一貫性のあるワークフローに統合する能力が含まれます。

Agentspace Enterprise のアーキテクチャは、いくつかの中核となる原則に基づいています。これらのうち最も重要なのは組み込みの信頼であり、これはシングルサインオン認証、統合された権限モデル、ユーザーレベルのアクセス制御といった機能を通じて、セキュリティ、説明可能性、そしてガバナンスを優先します。Google の高度なインテリジェンスが活用され、機械学習を通じてユーザーの振る舞いとコンテンツのパターンを識別し、その結果として意味理解、ナレッジグラフ、そして LLM を介して関連性の非常に高い結果が提供されます。オンデマンドおよび自動化されたデータ更新によって多様なエンタープライズシステムに接続することで、ユニバーサルな接続性が実現され、それによって情報のサイロ化が解消されます。検索機能、推薦、LLM、ナレッジグラフに対するきめ細かい制御を通じて、エンタープライズレベルのカスタマイズとユーザーレベルのパーソナライゼーションが促進され、個々のユーザーの役割、権限、検索履歴に基づいたパーソナライズされた体験が提供されます。リアルタイムフィードバックと適応メカニズムは、機械学習とユーザー入力による結果の継続的な洗練を可能にします。ブレンド型検索拡張生成 (RAG) は、カスタマイズ可能なデータブレンディングを可能にし、エンタープライズデータにグラウンドィングされた生成 AI アプリケーションを強化します。最後に、スケーラビリティは重要な設計上の考慮事項であり、システムは地理的地域、言語、そしてピーク時の使用需要にわたる成長に対応できるように設計されています。

セキュリティは Agentspace の基本原則です。Google Cloud のセキュアバイデザインのインフラストラクチャ上に構築されており、AI エージェントをデプロイするための堅牢な環境を提供します。これにより、ロールベースのアクセス制御 (RBAC)、Virtual Private Cloud (VPC) Service Controls、そして Identity and Access Management (IAM) との統合といった、きめ細かい IT 制御が提供されます。これらのセキュリティ対策は、機密情報の保護、データの保護、そして規制コンプライアンスを保証し、ユーザーに組織全体で安心して AI エージェントをデプロイできるという安心感を与えます。

## NotebookLM Enterprise

NotebookLM<sup>29</sup> は、複雑な情報を理解し統合するプロセスを合理化するために設計された、リサーチおよび学習ツールです。ユーザーは、ドキュメント、メモ、その他の関連ファイルを含む様々なソース資料をアップロードでき、人工知能の力を活用してより深い理解を促進することが可能になります。多角的なトピックを調査していると想像してみてください。NotebookLM を使えば、散在するすべてのリソースを、単一の整理されたワークスペースに一元化できます。本質的に、NotebookLM は専門のリサーチアシスタントとして機能し、リサーチプロセスを加速させ、ユーザーが單なる情報収集から真の理解へと移行することを可能にします。

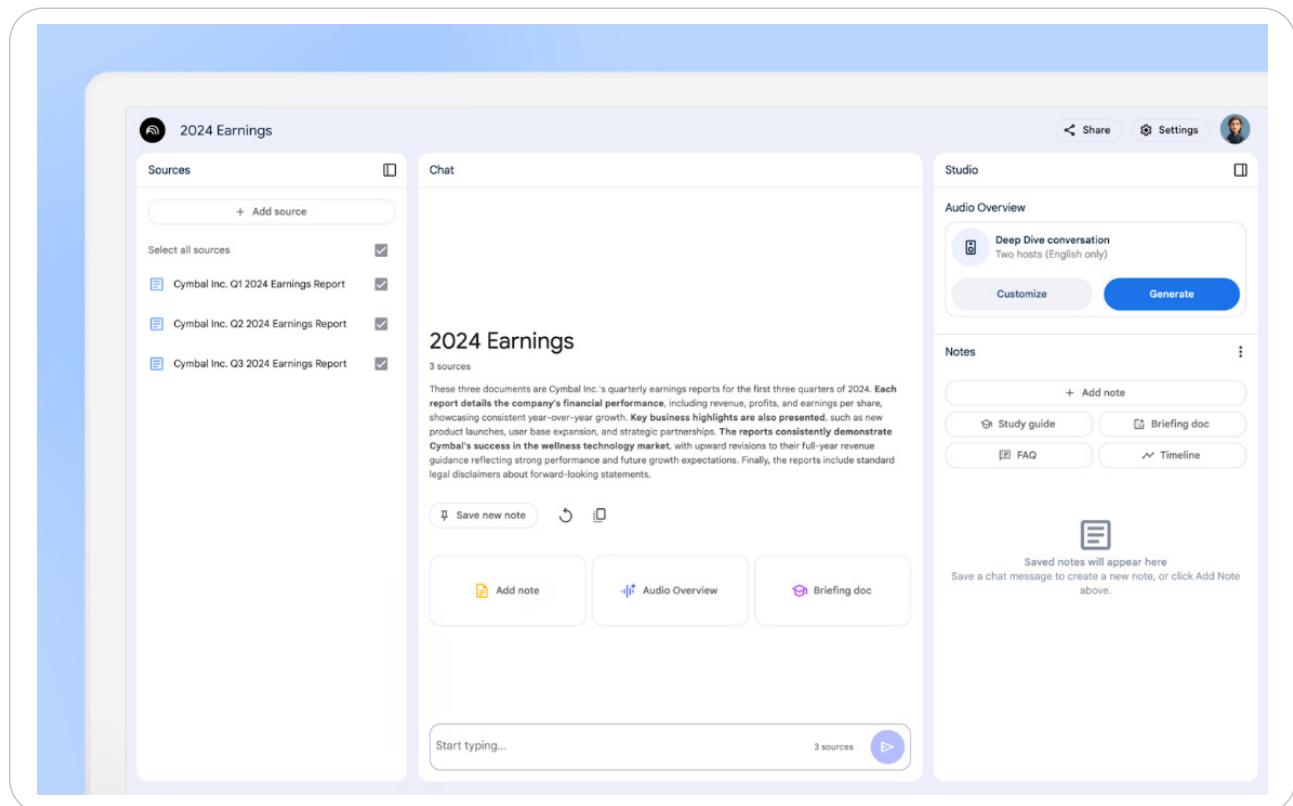


図 11：いくつかの模擬収益レポートを設定された NotebookLM Enterprise<sup>30</sup>

NotebookLM Plus はこの基盤の上に構築されており、強化された機能と能力を持つプレミアムティアを提供します。ソースのアップロード、質問、要約の生成といったコア機能はそのままに、NotebookLM Plus は通常、ソース資料用のストレージ増量といった機能を追加し、より大規模で複雑なプロジェクトでの作業を可能にします。また、よりニュアンスのある要約オプション、強化された質問応答能力、あるいは異なるソース間のつながりや関係性をより効果的に特定する能力など、より高度な AI を活用した分析ツールが含まれることもあります。

NotebookLM Plus の基盤の上に構築された NotebookLM Enterprise<sup>30</sup> は、これらの強力な能力をエンタープライズ環境にもたらし、従業員がデータと対話し、そこから洞察を引き出す方法を合理化します。このエンタープライズグレードのバージョンは、ユーザーが情報をアップロードして統合するだけでなく、隠れたパターンを発見し、革新的な方法でデータに関わることを可能にします。その代表的な例が AI が生成する音声要約機能です。これは、ユーザーが自らのリサーチ内容を「聴く」ことを可能にすることで、理解を深め、効率的な知識の吸収を促進します。

技術的には、NotebookLM は、コンシューマー版とエンタープライズ版の両方で、アップロードされたドキュメントを処理し、主要な概念を抽出し、要約を生成するために LLM を採用しています。音声要約機能は、明瞭さと自然さを保証するために高度なプロセッサ制御を取り入れた、テキスト読み上げ (TTS) 技術を使用しています。重要な点として、NotebookLM Enterprise は、エンタープライズグレードのセキュリティとプライバシー機能を組み込んでおり、企業の機密データが最大限の注意を払って扱われ、組織のポリシーに従って保護されることを保証します。

## Google AgentSpace Enterprise

Google Agentspace は、従業員に、エンタープライズ情報の決定的な情報源として機能する、統一された企業ブランドのマルチモーダル検索エージェントを提供します。Google の広範な検索能力を活用し、Agentspace は比類のない対話型アシスタンスを提供します。従業員は、複雑なクエリへの回答、プロアクティブな推薦、そしてあらゆるソースからの情報への統一されたアクセスを得ることができます。この機能は、ドキュメントや E メールのような非構造化データと、テーブルのような構造化データの両方に及びます。統合された翻訳機能により、情報の元の言語に関わらず、包括的な理解が保証されます。Confluence、Google Drive、Jira、Microsoft SharePoint、ServiceNow など、広く使用されているサードパーティ製アプリケーション向けの構築済みコネクタは、従業員が関連するデータソースにシームレスにアクセスしてクエリを実行できるようにし、より情報に基づいた意思決定を促進します。

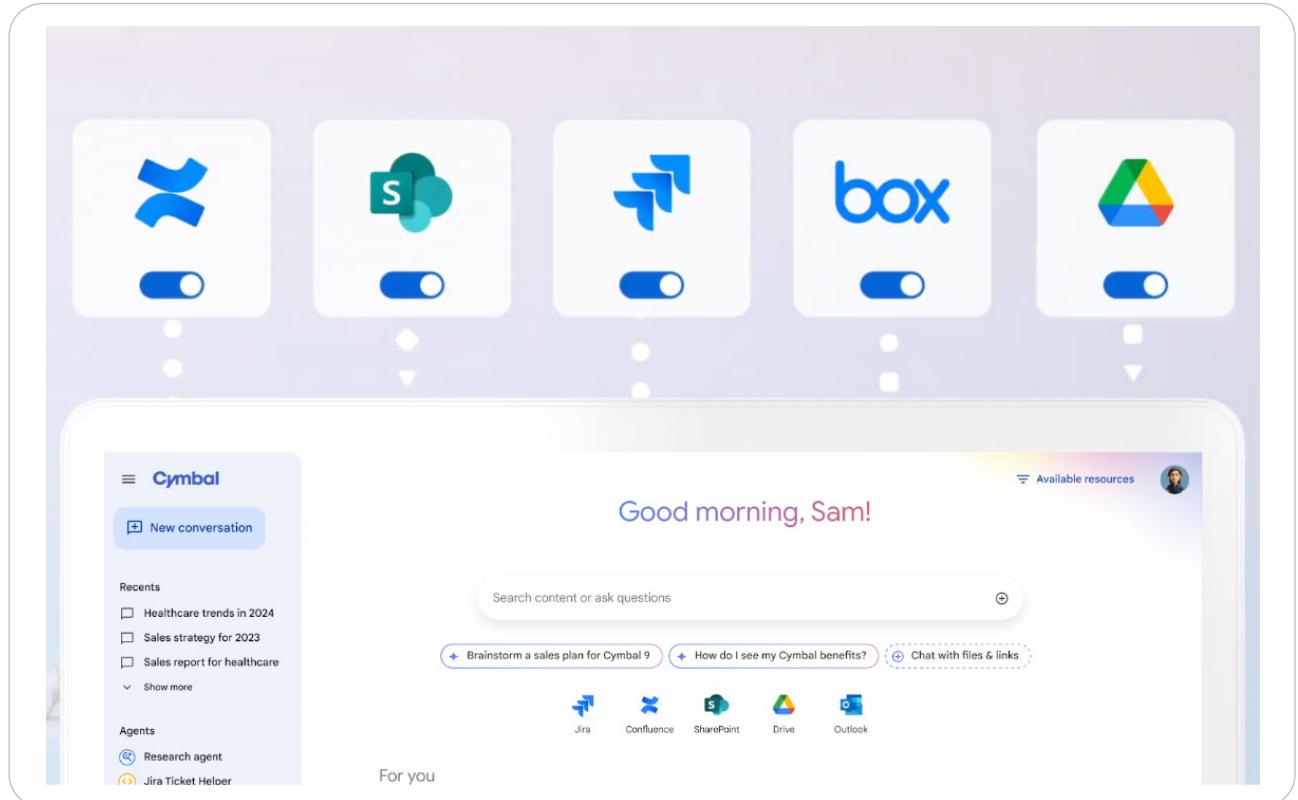


図 12 : Cymbal 銀行向けにカスタマイズされ、いくつかのデータストアに接続された Agentspace<sup>29</sup>

エージェントは、単なる情報検索をはるかに超えて、実世界でアクションを起こし、非同期タスクやワークフローを管理し、従業員の業務遂行を支援することができます。エージェントのギャラリーは、組織固有のニーズに基づいて設定することができ、深い調査、アイデア生成と価値に基づく精緻化、クリエイティブアセット生成、データ分析などを支援します。

Agentspace Enterprise Plus は、特定のビジネス機能に合わせたカスタム AI エージェントの作成を促進します。このプラットフォームは、コンテキストを認識する生成 AI エージェントの開発とデプロイを可能にし、マーケティング、財務、法務、エンジニアリングなど、部門を越えた従業員が、より効果的なリサーチを行い、迅速にコンテンツを生成し、マルチステップのワークフローを含む反復的なタスクを自動化できるようにします。一元化されたインターフェースが、これらの特化型エージェントの探索とアクセスを合理化し、スケーラブルな AI 導入を促進します。カスタムエージェントは、内部および外部のシステムやデータに接続し、企業のドメインとポリシーに合わせて調整され、場合によっては独自のビジネスデータで訓練された機械学習モデルを利用することさえあります。このプラットフォームは、エージェントの開発、デプロイ、そしてライフサイクル管理のための、構築者向けツールを提供します。

# エージェントからコントラクターへ

今日の様々なツールやプラットフォームで AI エージェントを定義するための共通インターフェースは非常にシンプルで、通常、目標、いくつかのテキストによる指示、エージェントが使用できるツール、そして一連の事例の指定が含まれます。これはプロトタイプのデモを作成するには十分かもしれません、定義が不十分になる可能性があり、AI エージェントがプロトタイプから本番環境へ移行するのに苦労する主要な理由の一つかもしれません。

私たちは、エージェントインターフェースを進化させ、AI エージェントを用いて複雑なタスクを解決すること、特に重要度の高い文脈での解決を目的とした「契約遵守エージェント」へと進化させることを提案します。

## 契約

コントラクターの重要なアイデアは、リクエスターとエージェント間の契約を規定し標準化することで、以下のことを可能にすることです。

1. 私たちが実世界のシナリオで、何らかの作業を依頼する企業とサービスについて合意する際に行うのと同様に、成果を可能な限り正確に定義する。これにより、エージェント（コントラクター）は望ましい成果に対して検証を行い、望ましい目的が達成されるまでイテレーションを行うことができる。
2. 定義における曖昧さを避け、目標の共通理解におけるギャップを埋めるために、タスクを明確化・精緻化するだけでなく、交渉することも可能にする。
3. より大きな契約を解決するために必要な新しいサブコントラクトを、コントラクターが標準的な方法で生成するためのルールを定義する（下記のサブコントラクトに関するセクションを参照）。

契約、初期定義		
フィールド	説明	必須
タスク／プロジェクト記述	コンタクターに達成を期待する内容を詳細に記述する。可能な限り具体的かつ曖昧さのないようにするべきである。	はい
成果物と仕様	コンタクターのタスクから期待される成果と成果物を正確に記述する。これには、何をもって成果物として許容可能な成果とするかを明確にする仕様の一覧や、成果物が期待を満たしているかを検証する方法の詳細が含まれる。	はい
スコープ	コンタクターが完了する責任を負うタスクのスコープを明確にし、タスクのあらゆる側面について詳細に記述する。スコープ外のものを明確にするためにも使用される。	いいえ
想定コスト	タスク完了にかかるコストの期待値を示す。これは通常、タスクの複雑性と、どのツールが使用されるかを組み合わせた関数となる。	はい
想定期間	タスク完了にかかる期間の期待値を示す。	はい
入力ソース	タスクを完了するのに有用であると見なされ、使用できる入力ソースを規定する。	いいえ
報告とフィードバック	フィードバックループがどのようなものであるべきかを規定する。例えば、進捗に関する更新をどのくらいの頻度で期待するか、フィードバックを提供するためにどのメカニズム／サーフェス（Eメール、APIなど）が使用されるかなど。	はい

表 3：契約に関するデータモデルの例

契約のイテレーション：フィードバックと交渉		
フィールド	説明	必須
定義の不十分さ	定義が不十分であるか、タスクの開始者からの明確化が必要な側面をハイライトする。	いいえ
コスト交渉	タスクを完了するにはコストが高すぎると考えられる場合。	いいえ
リスク	契約の履行における潜在的なリスクをハイライトする。	いいえ
必要な追加インプット	契約を履行するのに有用だと思われる追加のデータや情報の種類を表明する。	いいえ

表 4：コントラクター間のメッセージングのイテレーションに関するデータモデルの例

## 契約ライフサイクル

以下の図は、契約を定義し、交渉し、実行するライフサイクルを示しています。

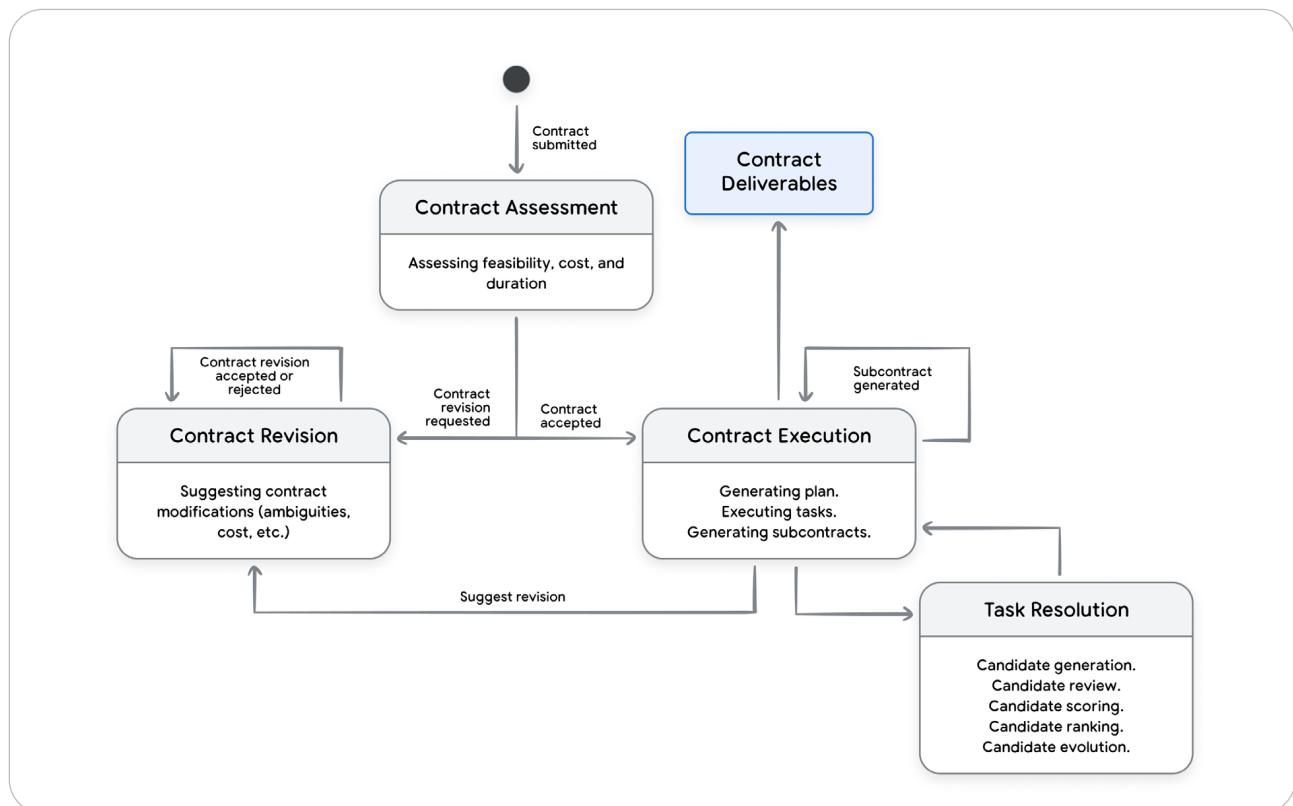


図 13 : Agentspace における、コントラクターとしてのエージェントのライフサイクル

## 契約の実行

これには、コントラクターランタイムが、定義された仕様に従って契約を履行し、契約の背後にあるタスクを解決できる能力が必要となります。レイテンシよりも品質と完全性を優先することで、例えば、異なるソリューションを生成し、それらをレビュー、スコアリングし、進化させることによって、LLM の能力を最大限に活用できます。この方法論の一例が、後述の Co-Scientist 研究で示されています。エンジンは、提供された期待値に基づいて結果と成果物をイテレーションし自己検証することができ、バリデーターが満たされるまで改善と自己修正を行うことができます。一連の客観的な基準に照らしてソリューションを具体的に検証する能力は、AI の文脈において非常にうまく機能することが証明されており、例えば AlphaCode のような成功した AI システムの核となっていました。

## 契約交渉

「エンタープライズの世界における多くのタスクは、（レイテンシとコストの面で）制約の少ない方法で LLM の力が活用されることで、大きな利益を得られる」というのが、特にオートメーションエージェントのコンテキストにおける契約の背後にある、中心的な仮説の一つです。

ますます複雑なタスクに取り組むことができ、顧客がコントラクターの結果を信頼し頼ることができるようにすることが、企業にとっての真の価値を保証します。そうは言っても、タスクが適切に優先順位付けされ、リソースが公正に割り当てられることを確実にするためには、相対的な優先順位という概念を持つ必要があります。そこで私たちは、コスト（通常は顧客または契約の開始者ごとに相対的）という概念を導入します。これは、契約の開始者とコントラクターの間で議論・交渉することができ、その契約が、契約の開始者が開始した他の契約と比較して適切なリソースを受け取れることを確実にするためのものです。コントラクターは、例えば仕様や成果物といった、契約の他の側面についても交渉することができます（フィードバックに関する下位のセクションも参照）。

## 契約フィードバック

契約は、フィードバックを提供し、特に曖昧さを解決するための手段を提供します。タスクがますます複雑になるにつれて、タスクの仕様に関連する曖昧さやその他の問題ができるだけ早期に提起できることが極めて重要になります。コントラクターは、契約を受け取った直後（初期契約評価）と、その後は契約で事前定義された頻度で、契約に関するフィードバックを提供できます。

このフィードバックには、明確化の要求や、タスクの定義の不十分さや仕様の誤りに関する他のタイプのフィードバック（不整合、矛盾した仕様、明確化など）が含まれます。

## サブコントラクト

契約の定義と仕様に直接含まれるわけではありませんが、サブコントラクトを生成することによってタスクをサブタスクに分解する能力は、コントラクターのエンジンを駆動するための中核となる概念です。

タスクが直接取り組むには複雑すぎると見なされた場合、コントラクターはそのタスクをより小さく、より簡単なタスクに分解することを決定でき、それらは解決のために実行キューに追加されます。これは、前述の契約の形式化によってのみ可能になります。この形式化により、コントラクターは他の契約を、均一かつ標準化された方法で生成、処理、そして操作することが可能になります。

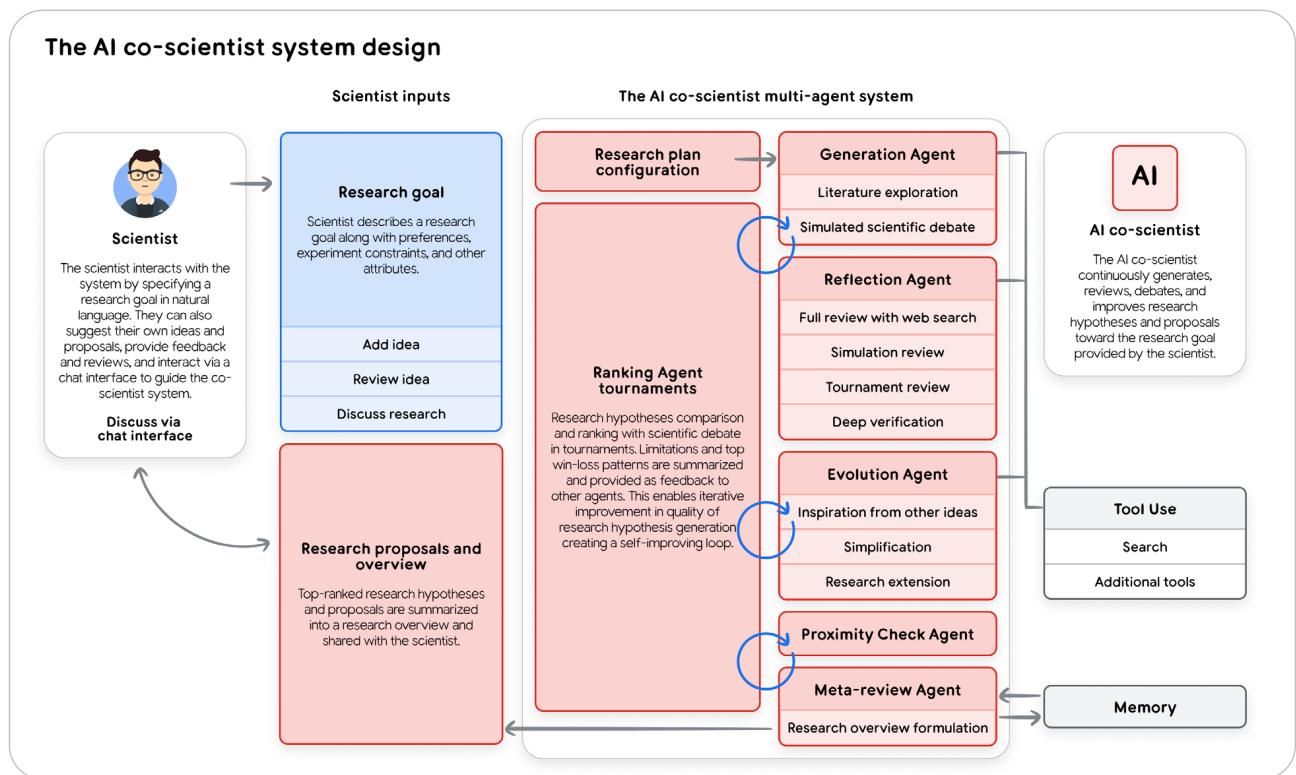
# Google の Co-Scientist：マルチエージェント インテリジェンスにおけるケーススタディ

Google の AI 共同科学者は、科学研究に応用されたマルチエージェント LLM システムの代表的な例です。このシステムは、それぞれが独自の役割と専門知識を持つ特化型エージェントのチームを活用し、科学的発見のペースを加速させます。これらのエージェントは協調して仮説を生成、評価、そして精緻化し、科学的探究の反復的なプロセスを模倣します。

共同科学者システムは、科学的手法から着想を得た、「生成、議論、そして進化」というアプローチを採用しています。このアプローチには、多様な仮説を生成し、その可能性を批判的に評価し、継続的なフィードバックと分析を通じてそれらを精緻化することが含まれます。このシステムは、研究プロセスの特定の側面にそれぞれ特化した、異なる LLM の強みを活用し、より包括的で堅牢な成果を達成します。

例えば、肝線維症の治療に関する研究において、共同科学者は既存の薬を特定しただけでなく、新たなメカニズムと有望な薬剤候補も提案し、新たな洞察を生成するその潜在能力を実証しました。その主要なコンポーネントには、以下のよう�습니다。

- ・**データ処理エージェント**：大量の実験データを集約し、構造化する。
- ・**仮説生成エージェント**：既存の研究と新たな知見に基づいて、可能性のある説明を提案する。
- ・**検証エージェント**：研究者に提示する前に、シミュレーションを実行し、結果を検証する。
- ・**協調エージェント**：異なる研究チーム間で知見を伝え、学際的な協力を強化する。

図 14：AI 共同科学者システムのプロセス図<sup>31</sup>

このアプローチは、マルチエージェントアーキテクチャが、いかにして静的な AI インタラクションを超越し、その代わりに、研究、エンタープライズオートメーション、そして知識管理における高度なタスクを処理できる、動的で進化するインテリジェンスを育むかを例証しています。

# 車載 AI：マルチエージェントアーキテクチャの実世界での使用例

マルチエージェントの概念が実践でどのように具現化されるかを説明するために、車載対話型 AI 向けに設計された包括的なマルチエージェントシステムを検証してみましょう。そこでは、いくつかの特化型エージェントが協調し、直感的でシームレスな車内エクスペリエンスを提供します。

まず、特化型エージェントのいくつかの例を見ていき、続いて、それらを組み合わせてマルチエージェントシステムを形成するための、いくつかの一般的なパターンを見ていきます。

## 特化型エージェント

まず、特化型エージェントのいくつかの例を見ていきましょう。

## 対話型ナビゲーションエージェント

対話型ナビゲーションエージェントは、ユーザーが場所を見つけ、場所を提案し、Google プレイスや Google マップのような API を使用してナビゲートするのを支援することに特化しています。このエージェントは、「近くの寿司屋を見つけて」のようなナビゲーションのインテントを理解し、関連する結果を得るために Places API にクエリし、ユーザーの好みと履歴データに基づいてこれらの結果を再ランキングし、そして車の車載システムにナビゲーションコマンドを送信します。

### 対話例

**ユーザー：**「昼食休憩をとりたいのですが。ミュンヘンへのルート沿いに、どこか良いレストランはありますか？」

**ナビゲーションエージェント：**「A9号線沿いのルート上で、評価の高いレストランを3軒見つけました。25キロ先、インゴルシュタット出口にある『Gasthaus Alte Mühle』は星4.5で、レビューでは伝統的なバイエルン料理とビアガーデンが賞賛されています。45キロ先のサービスエリアにある『La Strada』は星4.1で、手早く済ませられるイタリア料理の評価が高いです。あるいは、70キロ先まで待てるなら、アレルスハウゼン近郊の『Zeitlos』が星4.7です。こちらは少し高級な、ファームトゥテーブルの料理です。どちらに最もご興味がありますか？」

**ユーザー：**「バイエルン料理のお店がいいですね。そこに便利な駐車場はありますか？高速道路からはどれくらい離れていますか？」

**ナビゲーションエージェント：**「『Gasthaus Alte Mühle』には約40台収容の専用駐車場があります。インゴルシュタット北のA9号線出口からはわずか1.2キロ、車で約2分です。シュヴァイネブルーテンと自家製ダンプリングで知られています。正午頃は混み合いますが、現在のところ中程度の混雑状況です。こちらを目的地に設定し、案内を開始しますか？今向かえば、12時15分頃に到着します。」

## 対話型メディア検索エージェント

対話型メディア検索エージェントは、ユーザーが音楽、オーディオブック、ポッドキャストを見つけて再生するのを支援することに特化しています。このエージェントは、ローカルデータベースやストリーミングサービスからメディアを取得し、必要に応じて追加のコンテンツのためにウェブ検索を使用し、気分、天気、時間帯といったコンテキスト要因に基づいてメディアを提案し、「ドライブ用にリラックスできるプレイリストを再生して」のような音声コマンドを処理します。また、ユーザーが「ジョン・コルトレーンのようなアーティストは誰?」といった質問をした際には、類似のアーティストやコンテンツを特定します。

### 対話例

**ユーザー：**「この長い高速道路のドライブで眠くならないように、何かアップテンポな曲をかけて。」

**対話型メディア検索エージェント：**「元気が出るように、アップテンポなドライビングプレイリストを再生します。最初はクイーンの『Don't Stop Me Now』、続いてダフト・パンクとストロマ工の曲を再生します。あなたの好みに合わせて、ヨーロッパの人気のラジオヒット曲もいくつか追加しました。周囲の状況を認識しやすいように音量を調整しました。曲をスキップしたい場合は『次』と言ってください。」

## メッセージ作成エージェント

メッセージ作成エージェントは、ユーザーが運転中にメッセージや E メールの下書き作成、要約、そして送信を行うのを支援することに特化しています。このエージェントは、「アレックスに遅れると伝えて」のような音声コマンドを解析し、自然言語モデルを使用して文脈に適したメッセージを作成し、ユーザーによる修正オプション付きの下書きレビューを提供し、そして SMS、WhatsApp、E メールを含む様々なメッセージングアプリと統合します。

### 対話例

**ユーザー：**「E40 号線がひどい渋滞だ。チームに、朝のスタンドアップミーティングに 20 分遅れると伝えてください。」

**メッセージ作成エージェント：**「仕事のチーム宛にメッセージを作成しました。『E40 号線で渋滞にはまっています。朝のスタンドアップミーティングに約 20 分遅れます。先に始めてください。』このメッセージを『仕事チーム』のチャットグループに送信しますか、それとも先に編集しますか？」

## 車両マニュアルエージェント

車両マニュアルエージェントは、検索拡張生成（RAG）システムを活用して、車関連の質問に回答することに特化しています。このエージェントは、ユーザーのクエリに基づいて車両マニュアルから関連セクションを取得し、LLMを使用してこの情報を要約・文脈化し、適切な場合にはユーザーを詳細なドキュメントや解説動画にリンクします。

### 対話例

ユーザー：「このフォルクスワーゲンのレーンキープアシスト機能をオフにするにはどうすればいいですか？」

車両マニュアルエージェント：「お使いのフォルクスワーゲンでレーンキープアシストをオフにするには、ステアリングホイールの左側にある、車と車線のアイコンが付いたボタンを押してください。無効になると、ドライバーディスプレイに確認表示が出ます。また、インフォテインメントシステムの「設定」>「運転支援」>「レーンキープ」から永久に無効にすることもできます。代わりに設定メニューの手順を案内しましょうか？」

## 一般知識エージェント

一般知識エージェントは、世界、歴史、科学、文化、その他の一般的なトピックに関する、事実に基づいた質問に答えることに特化しています。このエージェントは、広範なナレッジベースにアクセスして事実に基づいた質問に答え、人物に関する情報を提供し、概念や現象について文脈に沿った説明を行い、ハルシネーションを避けながら事実情報にグラウンドィングし続け、そしてコンテキスト認識を維持することでフォローアップの質問に応答します。

### 対話例

ユーザー：「ザルツブルクについての面白い事実を教えてください。」

一般知識エージェント：「オーストリアのザルツブルクは、歴史上最も有名な作曲家の一人であるモーツアルトの生誕地です。そのよく知られた事実に加え、ザルツブルクの旧市街はユネスコ世界遺産であり、愛されるミュージカル『サウンド・オブ・ミュージック』の一部は1964年にそこで撮影されました。世界で最も権威のある音楽と演劇の祭典の一つであるザルツブルク音楽祭は、1920年から毎年そこで開催されています。」

## 使用されるパターン

これらのエージェントを組み合わせるために使われる、いくつかの一般的な方法を見ていきましょう。

### 階層型パターン

中央のオーケストレーターエージェントがクエリを分類し、それらを特化型エージェントにルーティングします。例えば、ユーザーが「近くの寿司屋を見つけて」と尋ねると、オーケストレーターはこれをナビゲーションリクエストとして検出し、それをナビゲーションエージェントにルーティングします。ナビゲーションエージェントは、場所の検索とマップ API とのインタラクションを処理します。

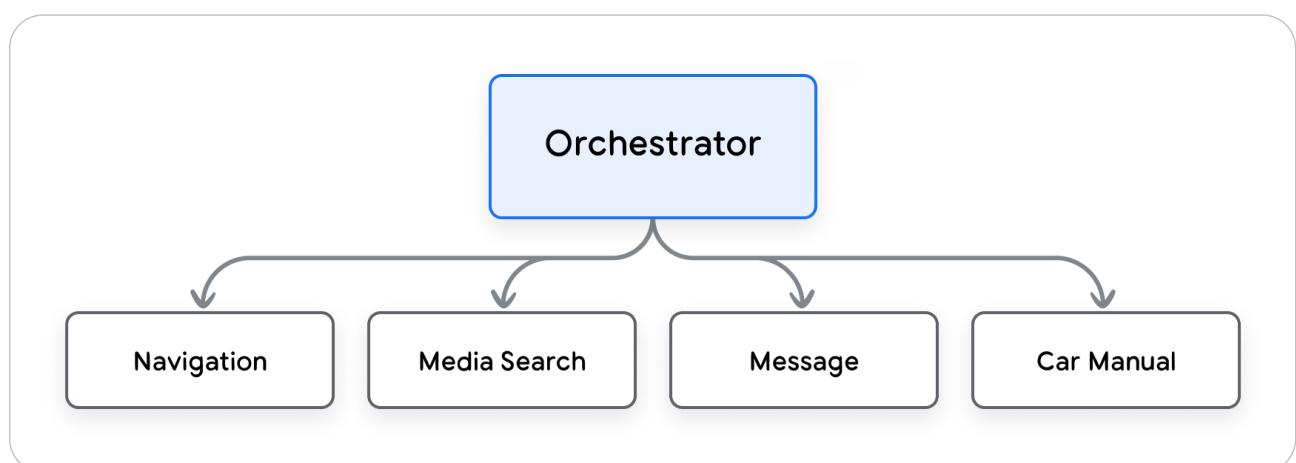


図 15：階層型パターン

オーケストレーターエージェントは、クエリを最も適切な特化型エージェントにルーティングし、対話全体のフローを管理することに特化しています。このエージェントは、ユーザーのクエリを分析してドメインとインテントを判断し、リクエストを最も適した特化型エージェントに送り、複数ターンにわたって対話のコンテキストを維持し、特化型エージェントが満足のいく応答を提供できない場合にはフォールバック戦略を管理します。

## 対話例

### 1ターン目

ユーザー：「この先に、トイレが綺麗なサービスエリアはありますか？」

オーケストレーターエージェント：[ 内部でナビゲーションエージェントにルーティング ]

ナビゲーションエージェント：「E19号線を20キロ先にサービスエリアがあります。『Aire de Waregem』です。昨年リノベーションされ、施設と清潔さで高評価を得ています。ベーカリーの『Paul』とクイックサービスのレストランもあります。」

### 2ターン目

ユーザー：「今晚、目的地は雨が降りますか？」

オーケストレーターエージェント：[ 天気のクエリと認識し、天気情報エージェントにルーティング ]

天気情報エージェント：「目的地のブリュッセルでは、予報によると今晚19時頃から小雨が降る可能性が60%です。気温は14°C前後になるでしょう。屋外に出る予定なら、傘を持って行くことをお勧めします。」

## ダイアモンドパターン

この階層型パターンのバリエーションでは、特化型エージェントからの応答は、ユーザーに届く前に中央のモデレーションエージェントを通過します。例えば、ナビゲーションエージェントが近くのレストランに関する事実に基づいた応答を生成し、それが次にリフレーズエージェントを通過します。リフレーズエージェントは、テキスト読み上げ合成の前に、ユーザー好みに基づいてトーンとスタイルを調整します。

リフレーズエージェントは、ユーザー好みとコンテキストに合わせて、応答のトーン、スタイル、そして表現方法を適応させることに特化しています。このエージェントは、技術情報を会話的な言葉遣いに変換し、運転状況や緊急性に基づいて応答の長さを調整し、ユーザー好みに合わせてコミュニケーションスタイルをパーソナライズし、運転中でも理解しやすいように複雑な情報をフォーマットします。

### 対話例

**[リフレーズ前の車両マニュアルエージェントからの生データ応答]**：「タイヤ空気圧監視システムのアラートが発生した場合、車両は1本または複数のタイヤが推奨空気圧である2.2バールを下回ったことを検知しました。該当タイヤの空気圧は約1.8バールであり、これは推奨レベルを18.75%下回っています。これは直ちに安全上の危険を構成するものではありませんが、空気圧が不足したタイヤでの長時間の運転は、燃費の低下、タイヤ寿命の短縮、そしてハンドリング特性の低下を招く可能性があります。できるだけ早くサービスステーションにお向かいになり、タイヤを適切な仕様まで再充填することが推奨されます。」

**リフレーズエージェント：**[運転というコンテキストに適応させ、情報密度と緊急度を調整]

**ユーザーへの最終応答：**「右前のタイヤの空気が少し減っていますね。1.8バールまで下がっています。運転を続けても安全ですが、次のガソリンスタンドで空気を入れた方がいいでしょう。タイヤの空気圧が低いと、燃費に影響したり、タイヤの摩耗が早まったりすることがあります。もしよろしければ、近くで空気圧を調整できる場所を探しましょうか？」

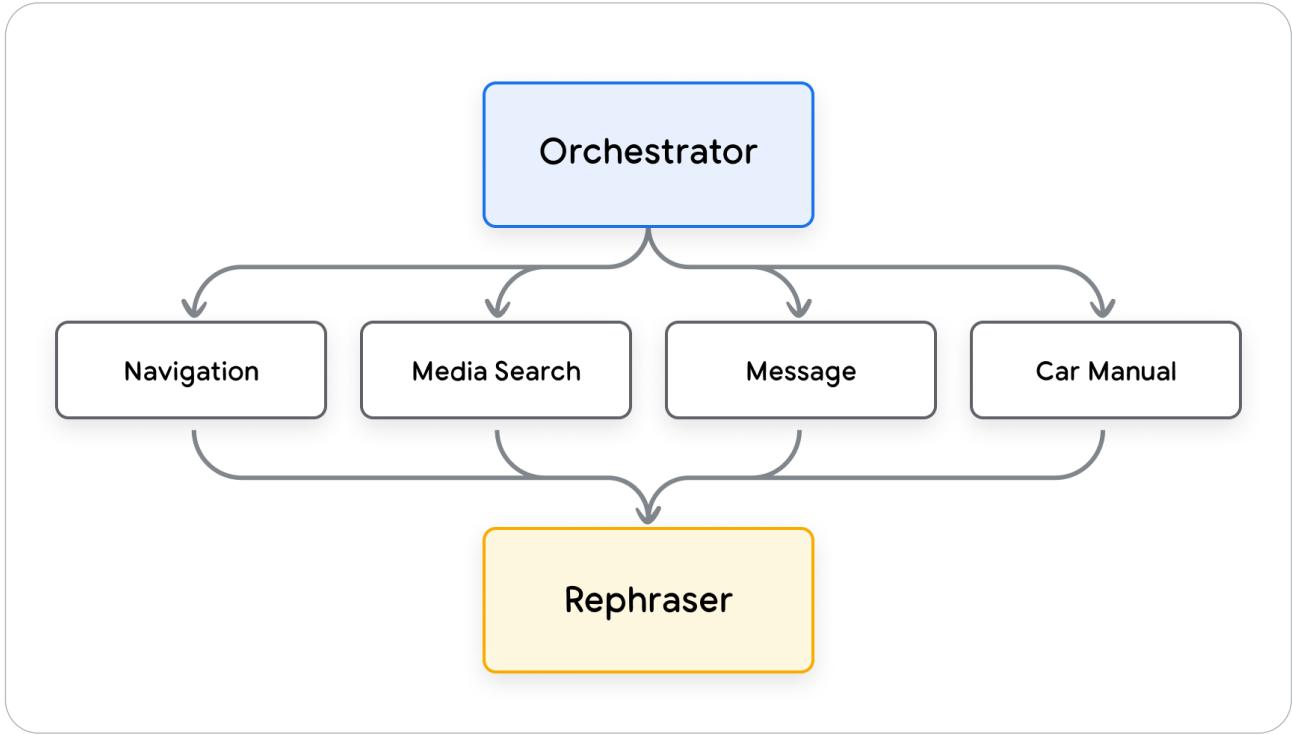


図 16：ダイアモンドパターン

## 変換例

**初期応答（事実ベース&ニュートラル）：**「近くに評価の高い寿司レストランを見つけました。『桜寿司』で、場所はメインストリート 123 番地です。評価は星 4.7 で、夜 10 時まで営業しています。ナビゲーションを開始しますか？」

**リフレーズ後（遊び心のあるスタイル）：**「お寿司が無性に食べたい？お任せあれ！メインストリート 123 番地の『桜寿司』へ向かいましょう。新鮮なネタが待ってますよ！評価は星 4.7、営業は夜 10 時まで。さあ、旨味の冒険へ出かけますか？」

## ピアツーピア

エージェントは、オーケストレーションがルーティングミスを犯したことを検知すると、クエリを相互にハンドオフすることができます。これにより、初期の誤分類から回復できる、より回復力のあるシステムが生まれます。

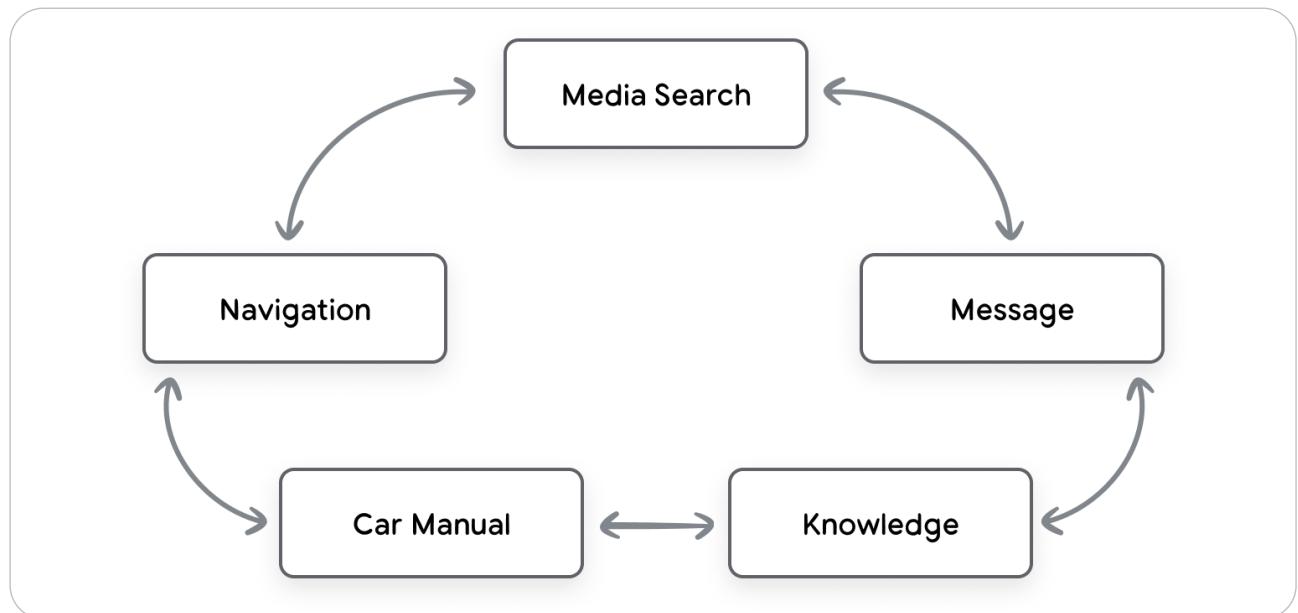


図 17：ピアツーピア

### シナリオ例

1. ユーザーが尋ねる：「近くで寿司を食べられる場所を見つけて」
2. オーケストレーターはこれを正しく対話型ナビゲーションエージェントにルーティングし、ナビゲーションエージェントは近くの寿司レストランに関する情報を提供する。
3. ユーザーがフォローアップする：「ニューヨークのセントラルパークはどのくらいの広さ？」
4. オーケストレーターは、（直前のナビゲーション関連の対話に基づき）これを最初は再び対話型ナビゲーションエージェントにルーティングするかもしれない。
5. しかし、対話型ナビゲーションエージェントはこれをナビゲーションリクエストではなく一般知識に関する質問だと認識し、一般知識エージェントにハンドオフする。一般知識エージェントは、セントラルパークの広さに関する事実情報を提供できる。

## 中央集権的なオーケストレーションと比較した、ピアツーピアのハンドオフの利点：

1. 誤分類への回復力：たとえ中央のオーケストレーターがルーティングでエラーを犯しても、特化型エージェントはクエリが自身のドメイン外であると認識し、適切にリダイレクトできる。
2. ルーティングにおけるドメイン専門知識：特化型エージェントは、しばしば自身のドメインの境界について、より深く理解している。メディア検索エージェントは、どのような種類の音楽関連クエリを扱えるかについて、汎用オーケストレーターよりも正確に把握している。
3. オーケストレーションの複雑性の低減：中央のオーケストレーターは初期のルーティングで完璧な精度を必要としないため、その意思決定ロジックの複雑性が低減される。

## 協調型パターン

協調型パターンでは、複数のエージェントが同じタスクの相補的な側面に取り組み、応答ミキサーワークフローが異なるエージェントの応答から要素を組み合わせて包括的な回答を作成します。このアプローチは、異なるエージェントがそれぞれの特化された専門知識に基づいて、完全なソリューションに対して重要な要素を提供し貢献することを認識したものです。このパターンは、特に次のような場合に価値があります。

1. クエリの異なる側面が、異なるタイプの専門知識（例：技術仕様、実践的なアドバイス、概念的な説明）を必要とする場合
2. 単一のエージェントでは、ユーザーのニーズに完全に応えるための十分な情報を持っていない場合
3. ユーザーが同じ質問に対して複数の視点から利益を得られる場合
4. 異なる特化型エージェントが、別々のナレッジベースや推論能力にアクセスできる場合

応答が競合する競合型アプローチとは異なり、協調型パターンでは、異なるエージェントからの応答は冗長ではなく相補的であると仮定します。応答ミキサーワークフローは、各ソースから最も価値のある情報を特定し、それを統合して、各スペシャリストの独自の強みを活用した一貫性のある回答を作成します。

例えば、ハイドロプレーニング現象への対処法について尋ねられると、車両マニュアルエージェントは車両固有の安全システム情報を提供し、運転のヒントエージェントは実践的な運転テクニックを提供し、そして一般知識エージェントはその現象の背後にある物理学を説明します。これらが一体となることで、単一のエージェントが単独で提供できるよりも、より完全で有用な応答が作成されます。

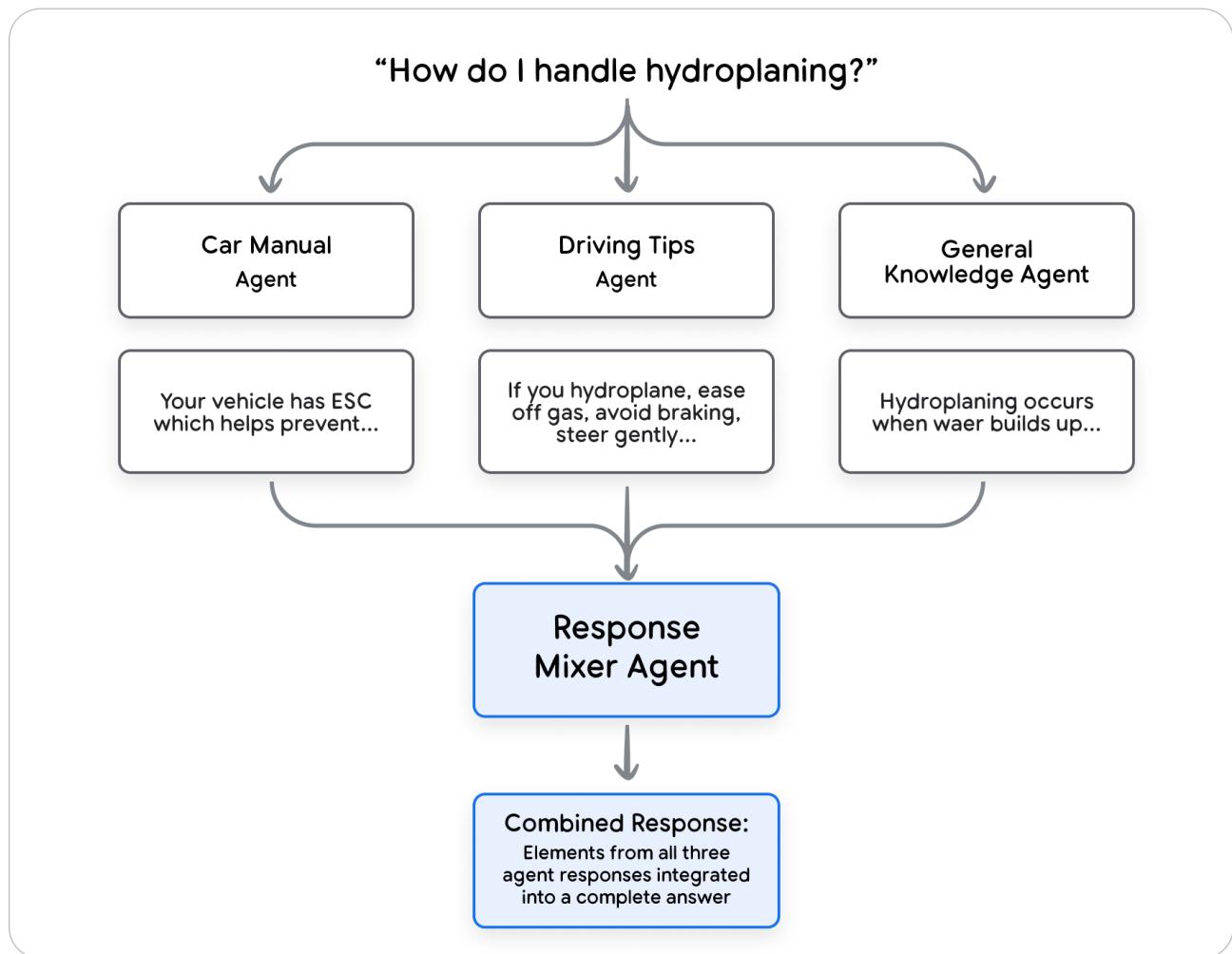


図 18：協調型パターン

## 応答ミキサーワージェント

応答ミキサーワージェントは、複数のエージェントから最良の応答を選び出し、必要に応じてそれらをマージすることに特化しています。このエージェントは、回答をその精度と関連性に基づいて評価し、不正確な情報を削除し、異なる応答の有用な部分を組み合わせて、ユーザーにとって最も役立つ回答を作成します。

### 対話例

ユーザー：「ハイドロプレーニング現象にはどう対処すればいいですか？」

[舞台裏では、複数のエージェントが応答を生成：]

**車両マニュアルエージェントの応答：**「あなたのアウディには、トラクションコントロールを維持することでハイドロプレーニング現象を防ぐのに役立つ、エレクトロニック・スタビリティ・コントロール（ESC）とアンチロック・ブレーキ・システム（ABS）が装備されています。」 [信頼度スコア：71%]

**一般知識エージェントの応答：**「ハイドロプレーニング現象は、タイヤと路面の間に水が溜まることで、タイヤが路面との接触を失ったときに発生します。」 [信頼度スコア：65%]

**安全のヒントエージェントの応答：**「ハイドロプレーニング現象が発生したら、アクセルを緩め、急なブレーキやステアリング操作を避け、トラクションを回復するまで行きたい方向に穏やかにハンドルを切ってください。」 [信頼度スコア：94%]

**応答ミキサーワージェント：**（最も関連性の高い部分を選び出し、一つの完全な回答にマージ）

**ユーザーへの最終応答：**「ハイドロプレーニング現象（タイヤと路面の間に水が溜まるでタイヤが路面との接触を失う現象）が発生したら、アクセルを緩め、急なブレーキやステアリング操作を避け、トラクションを回復するまで行きたい方向に穏やかにハンドルを切ってください。あなたのアウディのエレクトロニック・スタビリティ・コントロールと ABS システムがコントロールの維持を助けてくれますが、適切な運転技術は依然として不可欠です。」

この競合型アプローチは、ルーティングミスによって重要な安全情報が見逃されることなく、また、たとえそのクエリに複数の妥当な解釈が存在する場合でも、ユーザーが可能な限り最も役立つ応答を受け取れることを保証します。

## 適応型ループパターン

対話型ナビゲーションエージェントは、最初のクエリで満足のいく成果が得られない場合に、ループパターンを実装して検索結果を反復的に改善することができます。

### 対話例

1. ユーザーが尋ねる：「ヴィーガンメニューのある、良いイタリアンレストランを見つけて」
2. ナビゲーションエージェントが、これらのパラメータで Places API にクエリする。
3. エージェントが結果を評価し、すべての基準を満たすレストランが見つからないと判断する。
4. エージェントは、「結果なし」と返す代わりに、自動的にクエリを再構成する。
  - ・第1ループ：「ベジタリアンオプションのあるイタリアンレストラン」を検索する
  - ・第2ループ：「イタリアンレストラン」に範囲を広げ、次に植物由来のオプションに言及しているものをフィルターする
  - ・第3ループ：それでも成功しない場合、「ヴィーガンレストラン」を検索し、イタリア料理の影響を受けたものをフィルターする
5. エージェントは、これらの段階的な検索から最も一致するものを提示し、それらが元のリクエストとどのように関連するかを説明する。

このループ動作は、可用性とコンテキストに適応する、より堅牢な検索能力を可能にし、完全一致が見つからない場合でも有用な結果を提供します。

## 車載 AI におけるマルチエージェントアーキテクチャの利点

マルチエージェントシステムは、複雑なタスクを特化された役割に分解することで、車載 AI に強力な利点をもたらします。これは、一人のジェネラリストに頼るのではなく、エキスパートのチームを編成するようなものだと考えてください。

このアプローチでは、各エージェントは自身が最も得意とすることに特化します。ナビゲーションエージェントは、場所の検索とルートの計画にのみ集中します。メディア検索エージェントは、音楽とポッドキャストのエキスパートになります。車両マニュアルエージェントは、車両の機能とトラブルシューティングに特化します。特定のドメインに集中することで、各エージェントは自身の領域でより深い能力を開発します。

この特化によって、システム全体がより効率的になります。エージェントがより狭い範囲のタスクを扱うことで、そのパフォーマンスを最適化することがより簡単になります。その結果、より高品質な応答が、より迅速に、そしてより低い計算コストで提供されます。パフォーマンスが向上するのは、システムが各タスクに適切なリソースを割り当てることができるからです。単純なリクエストは最小限の処理能力を使用し、一方で複雑な質問は必要な時にのみ、より強力なリソースを活用します。

車内では速度が重要であり、マルチエージェントシステムは肝心なところでその真価を発揮します。エアコン操作や窓の開閉といったクリティカルな機能は、即座の応答のために、高速なオンデバイスエージェントで実行されます。一方で、レストランの推薦を探すといった緊急性の低いタスクは、より広範な知識を持つクラウドベースのエージェントを使用できます。この分離によって、システムが他にいかなる処理を行っていても、必要不可欠な車両コントロールの応答性が維持されることが保証されます。

この設計は、自然な回復力も生み出します。インターネットのコネクティビティが途切れても、オンデバイスエージェントで動作する必要不可欠な機能は動き続けます。一時的にレストランの推薦は利用できなくなるかもしれません、エアコン操作や基本的なメディアの再生は、変わらず完璧に機能します。

# Agent Builder

**Vertex AI Agent Builder** は、開発者向けの製品とサービスの集合体です。私たちは、あなたがエージェントを構築し接続するための、包括的なプラットフォームをまとめ上げました。Google Cloud の卓越したエンジニアリングとセキュリティ、Google DeepMind の AI 研究、そして AgentOps のベストプラクティスが結集しています。Google Cloud は、このプラットフォーム上で自社のエージェントを構築しており、今やあなたもそれが可能です。2025 年には、エージェント開発者を対象とした、さらに多くのエキサイティングな発表にご期待ください。

**Vertex AI Agent Engine** は、人気のあるオープンソースのエージェントライブラリとの、Google のエンジニアリングが管理する統合に依拠することで、開発を合理化します。これは、マネージドのオートスケーリングランタイムと、エージェントが必要とする多くのサービス（例：セッション、事例、トレース、評価）を提供します。これは、あらゆるフレームワークであなたが作成したエージェントを安全にデプロイするための、非常に低労力で高価値な方法です。

**Vertex AI Eval Service** は、このホワイトペーパーで議論されたすべての評価ツール、そしてそれ以上のものを提供します。Vertex AI Eval Service を介することで、LLM、RAG、そしてエージェント評価は安定かつスケーラブルになり、モニタリングや実験のオファリングとの便利な統合も備えています。

エージェントツールの大きなポートフォリオ、そして今後注目すべきさらなるツール：

- Vertex AI Search<sup>3</sup> または RAG エンジン<sup>28</sup> による検索。
- Gen AI Toolbox for Databases<sup>32</sup> を介した、DB からの検索に基づかない取得。
- 完全な ACL をサポートする数百の API とのアプリケーション統合<sup>33</sup>。
- Apigee Hub<sup>34</sup> を使用して、あらゆる API をマネージドでエンタープライズ対応のツールに変換。

そしてもちろん、エージェントにとって最高の LLM。これには Vertex AI Model Garden<sup>35</sup> へのアクセスや、エージェントの時代を牽引する Gemini モデルファミリー<sup>36</sup> へのアクセスも含まれます。

# まとめ

このホワイトペーパー（以前の『Agents』に関するホワイトペーパーのコンパニオン版）では、急速に進化する生成AIエージェントのランドスケープを、その基本的なアーキテクチャから、高度な評価手法、そしてマルチエージェントシステムの変革の可能性に至るまで探求してきました。

はい、承知いたしました。「Summary」の章にある文章を、これまでの翻訳と用語を統一し、プロの技術系書籍翻訳者として翻訳します。

## 開発者向けの主要なポイント

**1. AgentOpsは不可欠である：**成功するエージェントの構築は、初期のフルーフオブコンセプトをはるかに超えます。DevOpsやMLOpsのベストプラクティスを統合しつつ、ツール管理、オーケストレーション、メモリ、タスク分解といったエージェント固有の要素にも焦点を当てる、AgentOpsの原則を取り入れてください。

**2. メトリクスが改善を駆動する：**「北極星」として、ビジネスレベルのKPI（目標達成率、ユーザーエンゲージメント、収益など）から始めてください。次に、エージェントを計装化し、重要なタスク、ユーザーインタラクション、エージェントのアクション（トレース）に関連する、きめ細かいメトリクスを追跡します。ヒューマンフィードバック（👍、ユーザーアンケート）は非常に価値があります。

**3. 自動評価が鍵である：**手動テストのみに頼らないでください。エージェントの能力、トラジェクトリ（実行されたステップ）、そして最終応答を評価する、自動評価フレームワークを実装します。トラジェクトリ評価には完全一致、順序付き一致、適合率／再現率のようなテクニックを、最終応答の品質には自動評価器（審査員としてのLLM）を活用します。

**4. ヒューマンインザループは極めて重要である：**自動メトリクスは強力ですが、人間による評価は、特に創造性、常識、ニュアンスといった主観的な側面において、不可欠なコンテキストを提供します。自動評価手法をキャリブレーションし検証するために、ヒューマンフィードバックを使用してください。ドメイン知識をアウトソースしてはいけません。

**5. マルチエージェントシステムは利点を提供する：**複雑なタスクには、マルチエージェントアーキテクチャを検討してください。それらは、精度、効率、スケーラビリティ、そしてフォールトトレランスを向上させることができます。様々な設計パターン（逐次、階層型、協調型、競合型）を理解し、自身のアプリケーションに適したものを選択します。

**6. エージェント型 RAG は関連性を向上させる：**検索クエリを能動的に精緻化し、取得した情報を評価し、進化する知識に適応するエージェントを組み込むことで、従来の RAG を超えてください。これは、より正確で文脈に即した応答につながります。

**7. 検索の最適化は RAG の基礎である：**複雑なエージェント型 RAG に飛び込む前に、基盤となる検索エンジンを最適化してください。セマンティックチャンキング、メタデータエンリッチメント、埋め込みモデルのファインチューニング、そしてランカーの使用といったテクニックは、検索品質を大幅に向上させることができます。

**8. エージェントとツールのレジストリは重要である：**使用するエージェントやツールの数が増えるにつれて、その能力、オントロジー、そしてパフォーマンスを管理するためのレジストリが不可欠になります。

**9. セキュリティは最重要である：**エージェントをデプロイする際、特に企業内では、セキュリティを最優先してください。RBAC、VPC Service Controls、IAM との統合のような組み込みのセキュリティ機能を提供する Google Agentspace のようなプラットフォームを活用します。

**10. 開発者サイクルの効率的な利用：**生成 AI エージェントの業界は急速に進化しているため、伝統的な「ビルドかバイカ」の設計上の選択は、常に念頭に置くべき最重要事項であり続けます。すべてをスクラッチから構築する代わりに、プラットフォームや製品を代替案として検討してください。これは、急速に変化する業界の激しい変化をある程度緩衝し、自身のデータ、ドメイン、そしてユーザーに集中することを可能にします。

**11. 企業におけるエージェント：**エージェントは、私たちの生産性を大幅に向上させ、自動化を達成する方法を変革しています。ナレッジワーカーは、ますますエージェントのフリート（集団）を管理するようになり、斬新な UX が登場するでしょう。Google Agentspace は、企業のデータとワークフローの上に、エンタープライズサーチ、AI、そして AI エージェントを配置することを可能にする強力なツールです。

## エージェントの研究開発における今後の方向性

AI エージェントの分野は急速な進化を遂げています。現在進行中の研究開発の主要な領域には、以下が含まれます。

- ・高度な評価手法：推論に焦点を当てたプロセスベースの評価、AI 支援の評価、そして標準化されたベンチマークを含む、より堅牢でスケーラブルな評価手法の開発。

- ・マルチエージェントの協調：マルチエージェントシステム内での協調およびコミュニケーションのメカニズムを改善し、より効果的な協調、タスク処理、そして推論を可能にすること。
- ・実世界への適応：動的で予測不可能な実世界の環境に適応し、学習できるエージェントの創出。車載 AI のような本番システムは、エージェントが、クリティカルな機能のためのオンデバイスパフォーマンスと、複雑なタスクのためのクラウドベースの能力との間で、いかにバランスを取らなければならないかを示しており、それはしばしば変化するコネクティビティの状況に適応することを伴います。
- ・説明可能性と解釈可能性：エージェントの振る舞いをより透明で理解可能なものにし、開発者やユーザーがその意思決定プロセスに対するより深い洞察を得られるようにすること。
- ・長期記憶と学習：エージェントが長期間にわたって情報を保持・活用し、継続的な学習と適応を可能にする、より高度なメモリメカニズムの開発。
- ・エージェント間コミュニケーションプロトコル：エージェントが、特に不透明なリモートシステム間で、タスク、知識、メッセージをどのように共有するかを、より良く定義すること。
- ・エージェントからコントラクターへ：エージェントが次のレベルの信頼性と実用性に到達するためには、タスクの定義をステップアップさせ、明確な成果物、検証メカニズム、そして曖昧さを交渉する能力を持つ「契約」へと変えていく必要があります。これは、私たちが他の企業に業務を委託する方法と似ています。

## Call to Action

AI の未来はエージェント型です。私たちは、開発者の皆さんにこれらの概念を受け入れ、次世代のインテリジェントアプリケーションの構築を始めることを奨励します。このホワイトペーパーで議論されたツールとテクニックを試してみてください。開発プロセスを加速させるために、Google Agentspace、NotebookLM Enterprise、Vertex Eval Service、Cloud Observability、そして Vertex AI Search といった、利用可能なリソースを探求してください。提供されているコード例、チュートリアル、そしてドキュメントに飛び込み、実践的な経験を積んでください。構築し、評価し、イテレーションを行い、そして成長し続けるエージェント開発者のコミュニティに貢献してください。可能性は無限であり、今こそ構築の時です！具体的には、参考文献にあるコードと Colab ノートブックから始めてみてください。

## 卷末注

1. Shafran, I., Cao, Y. et al., 2022, ‘ReAct: Synergizing Reasoning and Acting in Language Models’ . Available at: <https://arxiv.org/abs/2210.03629>.
2. Wei, J., Wang, X. et al., 2023, ‘Chain-of-Thought Prompting Elicits Reasoning in Large Language Models’ . Available at: <https://arxiv.org/pdf/2201.11903.pdf>.
3. Wang, X. et al., 2022, ‘Self-Consistency Improves Chain of Thought Reasoning in Language Models’ . Available at: <https://arxiv.org/abs/2203.11171>.
4. Diao, S. et al., 2023, ‘Active Prompting with Chain-of-Thought for Large Language Models’ . Available at: <https://arxiv.org/pdf/2302.12246.pdf>.
5. Zhang, H. et al., 2023, ‘Multimodal Chain-of-Thought Reasoning in Language Models’ . Available at: <https://arxiv.org/abs/2302.00923>.
6. Yao, S. et al., 2023, ‘Tree of Thoughts: Deliberate Problem Solving with Large Language Models’ . Available at: <https://arxiv.org/abs/2305.10601>.
7. Long, X., 2023, ‘Large Language Model Guided Tree-of-Thought’ . Available at: <https://arxiv.org/abs/2305.08291>.
8. Google. ‘Google Gemini Application’ . Available at: <http://gemini.google.com>.
9. Swagger. ‘OpenAPI Specification’ . Available at: <https://swagger.io/specification/>.
10. Xie, M., 2022, ‘How does in-context learning work? A framework for understanding the differences from traditional supervised learning’ . Available at: <https://ai.stanford.edu/blog/understanding-incontext/>.
11. Google Research. ‘ScaNN (Scalable Nearest Neighbors)’ . Available at: <https://github.com/google-research/google-research/tree/master/scann>.
12. LangChain. ‘LangChain’ . Available at: <https://python.langchain.com/v0.2/docs/introduction/>.
13. Sokratis Kartakis, 2024, ‘GenAI in Production: MLOps or GenAIOps?’ . Available at: <https://medium.com/google-cloud/genai-in-production-mlops-or-genaiops-25691c9becd0>.
14. Sokratis Kartakis, 2024 ‘Gen AI Ops, Operationalize Generative AI, A practical Guide’ . Available at: <https://medium.com/google-cloud/genaiops-operationalize-generative-ai-a-practical-guide-d5bedaa59d78>.

15. Cloud Trace overview. Available at: <https://cloud.google.com/trace/docs/overview>.
16. Berkeley Function-Calling Leaderboard (BFCL). Available at:  
[https://gorilla.cs.berkeley.edu/blogs/8\\_berkeley\\_function\\_calling\\_leaderboard.html](https://gorilla.cs.berkeley.edu/blogs/8_berkeley_function_calling_leaderboard.html).
17. Karthik Narasimhan, et al. 2024, ‘τ -bench’ . Available at <https://arxiv.org/abs/2406.12045>.
18. Karthik Valmeekam, et al., 2023, ‘PlanBench’ . Available at: <https://arxiv.org/abs/2206.10498>.
19. Xiao Liu, et al., 2023, ‘AgentBench’ . Available at: <https://arxiv.org/abs/2308.03688>.
20. Martin Iglesias, et al., 2025, ‘DABStep’ Available at: <https://huggingface.co/spaces/adyen/DABstep>.
21. LangSmith platform for agent observability.  
Available at: <https://docs.smith.langchain.com/evaluation/concepts#agents>.
22. Mingchen Zhuge, et al., 2024, ‘Agent-as-a-Judge: Evaluate Agents with Agents’ .  
Available at: <https://arxiv.org/abs/2410.10934>.
23. Multi-agent documentation from LangGraph.  
Available at: [https://langchain-ai.github.io/langgraph/concepts/multi\\_agent/](https://langchain-ai.github.io/langgraph/concepts/multi_agent/).
24. LangChain blog 2024, ‘Multi-agent workflows’ .  
Available at: <https://blog.langchain.dev/langgraph-multi-agent-workflows/>.
25. Vectorize blog 2024, ‘How I finally got agentic RAG to work right’ .  
Available at: <https://vectorize.io/how-i-finally-got-agentic-rag-to-work-right/>.
26. Vertex AI Search, product documentation. Available at: <https://cloud.google.com/enterprise-search>.
27. Vertex AI Search Builder APIs, product documentation.  
Available at: <https://cloud.google.com/generative-ai-app-builder/docs/builder-apis>.
28. Vertex AI RAG Engine, product documentation.  
Available at: <https://cloud.google.com/vertex-ai/generative-ai/docs/rag-overview>.
29. Agentspace product documentation.  
Available at: <https://cloud.google.com/agentspace/agentspace-enterprise/docs/overview>.
30. NotebookLM Enterprise product documentation.  
Available at: <https://cloud.google.com/agentspace/notebooklm-enterprise/docs/overview>.

31. Juraj Gottweis, et al., 2025, ‘Accelerating scientific breakthroughs with an AI co-scientist’ . Available at: <https://research.google/blog/accelerating-scientific-breakthroughs-with-an-ai-co-scientist/>.
32. Hamsa Buvaraghan, et al. 2025, ‘Announcing public beta of Gen AI Toolbox for Databases’ . Available at: <https://cloud.google.com/blog/products/ai-machine-learning/announcing-gen-ai-toolbox-for-databases-get-started-today?e=48754805>.
33. Google Cloud Integration Connectors, product documentation.  
Available at: <https://cloud.google.com/integration-connectors/docs>.
34. Apigee API Hub, product documentation.  
Available at: <https://cloud.google.com/apigee/docs/apihub/what-is-api-hub>.
35. Vertex AI Model Garden, product documentation.  
Available at: <https://cloud.google.com/model-garden>.
36. Gemini family of LLMs, product documentation.  
Available at: <https://cloud.google.com/vertex-ai/generative-ai/docs/learn/models#gemini-models>.
37. Get Started Evaluating Agents with the Vertex Eval Service. Available at:  
<https://cloud.google.com/vertex-ai/generative-ai/docs/models/evaluation-agents>.
38. Irina Sigler, Ivan Nardini. Jan 2025 ‘Introducing Agent Evaluation in Vertex AI’ . Available at: <https://cloud.google.com/blog/products/ai-machine-learning/introducing-agent-evaluation-in-vertex-ai-gen-ai-evaluation-service?e=48754805>.
39. Review sample agent evaluation notebooks for LangGraph, CrewAI, and LangChain.  
Available at: <https://github.com/GoogleCloudPlatform/generative-ai/blob/main/gemini/evaluation/>.
40. Review many sample agents, primarily beginner and intermediate level.  
Available at: <https://github.com/GoogleCloudPlatform/generative-ai/>.
41. Review many sample agents, intermediate and advanced levels.  
Available at: <https://github.com/GoogleCloudPlatform/applied-ai-engineering-samples>.