のエージェント AI フレームワーク、プラットフォーム、プロトコル、ツール
AWS

AWS 規範ガイダンス

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS 規範ガイダンス: のエージェント AI フレームワーク、プラットフォーム、プロトコル、ツール AWS

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon の商標およびトレードドレスは Amazon 以外の製品およびサービスに使用することはできません。また、お客様に誤解を与える可能性がある形式で、または Amazon の信用を損なう形式で使用することもできません。Amazon が所有していないその他のすべての商標は Amazon との提携、関連、支援関係の有無にかかわらず、それら該当する所有者の資産です。

Table of Contents

のエージェント AI フレームワーク、プラットフォーム、プロトコル、ツール AWS

Aaron Sempf、Ansley Verzosa、Joshua Samuel、Amazon Web Services (AWS)

2026 年 1 月 (ドキュメント履歴)

エージェント AI は、AI、分散システム、ソフトウェアエンジニアリングの共通部分における強力なパラダイムです。AI モデルを使用し、ツールやリソースと統合する、自律型の非同期ソフトウェアエージェントで構成されるインテリジェントシステムのクラスです。エージェントは代理人として行動し、コンテキストや目標に対する理由を把握し、意思決定を行い、ユーザーやシステムに代わって意図的なアクションを実行できます。これらのエージェントは、分散環境内で独立して、多くの場合共同で動作し、インテリジェンス、メモリ、インテントが組み込まれた委任された目標を追求するように設計されています。

では AWS、エージェント AI を活用して複雑なワークフローを自動化し、意思決定プロセスを強化し、より応答性の高いシステムを作成できます。このガイドでは、効果的なエージェント AI ソリューションを構築するために必要な主要なコンポーネントについて説明します。

- フレームワークは、利点とユースケースのレビューなど、現在のエージェント AI フレームワークをプロファイリングします。これらのフレームワークが、パターン、プロトコル、ツール間の差別化されていない重労働をどのように軽減するかについて説明します。主要な選択基準を理解し、要件に適したフレームワークを選択します。
- プラットフォームは、エージェント AI プラットフォーム (マネージドエージェント、オープンソースオーケストレーション、ハイブリッド) の概要と、選択または設計に関する考慮事項を提供します。
- プロトコルは、エージェントとのやり取りに不可欠な標準化された通信プロトコルを探索します。Agent-to-agent プロトコルは、オープンソースの Model Context Protocol (MCP) や Agent2Agent (A2A) などの独自の実装とともに登場しています。一般的なプロトコルで、さまざまなプロトコルがシームレスにやり取りできるようにする方法について説明します。
- ツールには、プロトコルベースのツール (MCP など) 、フレームワークネイティブツール、メタツールに関する情報が用意されています。組織は、ワークフロー内のキーシステムと統合するツールキットを構築し、エンドユーザーとサーバーベースのエージェントワークフローの両方を実現できます。

対象者

このガイドは、最新のクラウドネイティブアプリケーション内で AI 駆動型ソフトウェアエージェントの能力を活用しようとしているアーキテクト、デベロッパー、テクノロジーリーダーを対象としています。

目的

このガイドは以下を行う際に役立ちます。

- さまざまなエージェント AI フレームワークを比較して、ユースケースに最適なものを選択します。
- 個々のエージェントを調整されたアダプティブシステムに変換する機能を提供するエージェント AI プラットフォームについて説明します。
- 持続可能なエージェント AI アーキテクチャを構築するためのオープンプロトコルの利点を理解します。
- エージェントシステムを構築するときに、適切なツール統合戦略を作成します。

このコンテンツシリーズについて

このガイドは、 でのエージェント AI に関するシリーズの一部です AWS。詳細およびこのシリーズの他のガイドについては、 AWS 「 規範ガイダンス」ウェブサイトの 「エージェント AI 」を参照してください。

フレームワーク

のエージェント AI の基礎 AWSは、自律的で目標指向の動作を可能にするコアパターンとワークフローを調べます。これらのパターンを実装する中心にあるのは、フレームワークの選択です。フレームワークは、事前記述されたコードのソフトウェア基盤であり、構造化された環境と、本番環境対応の自律型 AI エージェントの構築に必要な構築と管理、ツール、オーケストレーション機能のための共通の機能を提供します。

効果的なエージェント AI フレームワークは、raw large language model (LLM) インタラクションを推論、コラボレーション、アクションが可能な調整されたインテリジェントシステムに変換するいくつかの重要な機能を提供します。

- エージェントオーケストレーションは、単一または複数のエージェント間の情報の流れと意思決定を調整し、人間の介入なしに複雑な目標を達成します。
- ツール統合により、エージェントは外部システム、APIs、データソースとやり取りして、言語処理を超えて機能を拡張できます。詳細については、 Strands Agentsドキュメントの「ツールの概要」を参照してください。
- メモリ管理は、永続的またはセッションベースの状態を提供し、インタラクション全体でコンテキストを維持します。これは、長時間実行されるタスクやアダプティブタスクに不可欠です。より高度なフレームワークには、概要とユーザー設定を保存する長期メモリが組み込まれているため、パーソナライズされたコンテキストに応じたエージェントエクスペリエンスが可能になります。詳細については、 LangChain ブログの「エージェントフレームワークについて考える方法」を参照してください。
- ワークフロー定義は、高度な自律推論を可能にするチェーン、ルーティング、並列化、リフレクションループなどの構造化パターンをサポートします。
- デプロイとモニタリングにより、自律システムのオブザーバビリティを備えた開発から本番稼働への移行が容易になります。詳細については、Amazon Bedrock AgentCore の一般提供に関するお知らせを参照してください。

これらの機能は、フレームワーク環境全体でさまざまなアプローチと重点を置いて実装され、それぞれがさまざまな自律型エージェントのユースケースと組織コンテキストに異なる利点を提供します。

このセクションでは、エージェント AI ソリューションを構築するための主要なフレームワークをプロファイリングして比較し、その長所、制限、自律運用の理想的なユースケースに焦点を当てます。

- ストランドエージェント

- LangChain と LangGraph
- CrewAI
- AutoGen
- ???
- エージェント AI フレームワークの比較

> **Note**
>
> このセクションでは、AI の機関を特にサポートするフレームワークについて説明し、機関のないフロントエンドインターフェイスや生成 AI については説明しません。

Strands Agents

Strands Agents は、オープンソースAWS ブログで説明されているように AWS、 によって最初にリリースされたオープンソース SDK です。 Strands Agentsは、モデルファーストのアプローチで自律 AI エージェントを構築するように設計されています。 とシームレスに連携するように設計された柔軟で拡張可能なフレームワークを提供し AWS のサービス 、サードパーティーのコンポーネントとの統合にオープンなままです。Strands エージェントは、完全自律型ソリューションの構築に最適です。

の主な機能 Strands Agents

Strands Agents には、次の主要な機能が含まれています。

- モデルファースト設計 – 基盤モデルがエージェントインテリジェンスの中核であるという概念に基づいて構築され、高度な自律型推論を可能にします。詳細については、 Strands Agentsドキュメントの「エージェントループ」を参照してください。
- マルチエージェントコラボレーションパターン – 分散エージェントネットワーク全体でスケーラブルなコラボレーションとガバナンスを可能にする Swarm、Graph、Workflow パターンなどの組み込み調整モデル。詳細については、「 Strands Agents ドキュメント」の「Multi-agent Patterns」を参照してください。
- MCP 統合 – Model Context Protocol (MCP) のネイティブサポートにより、LLMs、一貫した自律オペレーションが可能になります。

- AWS のサービス 統合 – Amazon Bedrock、 など AWS のサービス へのシームレスな接続により AWS Lambda AWS Step Functions、包括的な自律型ワークフローを実現します。詳細については、AWS 「週次ラウンドアップ (ブログ)」を参照してください。AWS
- 基盤モデルの選択 – Amazon Bedrock の Anthropic Claude、Amazon Nova (Premier、Pro、Lite、Micro) など、さまざまな基盤モデルをサポートして、さまざまな自律推論機能を最適化します。詳細については、 Strands Agentsドキュメントの「Amazon Bedrock」を参照してください。
- LLM API 統合 – Amazon Bedrock、OpenAI など、本番デプロイ用のさまざまな LLM サービスインターフェイスとの柔軟な統合。詳細については、 Strands Agentsドキュメントの「Amazon Bedrock の基本的な使用方法」を参照してください。
- マルチモーダル機能 – テキスト、音声、画像処理を含む複数のモダリティをサポートし、包括的な自律型エージェントインタラクションを実現します。詳細については、 Strands Agentsドキュメントの「Amazon Bedrock マルチモーダルサポート」を参照してください。
- ツールエコシステム – AWS のサービス 対話のための豊富なツールセットと、自律機能を拡張するカスタムツールの拡張性。詳細については、 Strands Agentsドキュメントの「ツールの概要」を参照してください。

Strands Agents を使用する場合

Strands Agents は、次のような自律型エージェントのシナリオに特に適しています。

- 自動ワークフロー AWS のサービス のために とのネイティブ統合を必要とする AWS インフラストラクチャ上に構築される組織
- 本番稼働用自律システムのエンタープライズグレードのセキュリティ、スケーラビリティ、コンプライアンス機能を必要とするチーム
- 特殊な自律タスクのために異なるプロバイダー間でモデル選択に柔軟性を必要とするプロジェクト
- エンドツーエンドの自律プロセスのために既存の AWS ワークフローやリソースと緊密に統合する必要があるユースケース

の実装アプローチ Strands Agents

Strands Agents は、クイックスタートガイドで説明されているように、ビジネスステークホルダーに簡単な実装アプローチを提供します。このフレームワークにより、組織は次のことが可能になります。

- 特定のビジネス要件に基づいて、Amazon Bedrock の Amazon Nova (Premier、Pro、Lite、Micro) などの基盤モデルを選択します。
- エンタープライズシステムとデータソースに接続するカスタムツールを定義します。
- テキスト、画像、音声など、複数のモダリティを処理します。
- ビジネスクエリに自律的に応答し、タスクを実行できるエージェントをデプロイします。

この実装アプローチにより、ビジネスチームは AI モデル開発に関する深い技術的専門知識を必要とせずに、自律型エージェントを迅速に開発およびデプロイできます。

の実例 Strands Agents

AWS Transform for .NET Strands Agentsは、AWS Transform 「 for .NET」で説明されているように、 を使用してアプリケーションのモダナイゼーション機能を強化します。これは、.NET アプリケーションを大規模にモダナイズするための最初のエージェント AI サービスです (AWS ブログ)。この本稼働サービスは、複数の特殊な自律型エージェントを採用しています。エージェントは連携して、レガシー .NET アプリケーションを分析し、モダナイゼーション戦略を計画し、人間の介入なしにクラウドネイティブアーキテクチャへのコード変換を実行します。 AWS Transform for .NET は、エンタープライズ自律システムの Strands Agentsの本番稼働準備状況を示しています。

LangChain および LangGraph

LangChain は、エージェント AI エコシステムで最も確立されたフレームワークの 1 つです。 は、LangChain ブログで説明されているように、その機能LangGraphを拡張して、複雑でステートフルなエージェントワークフローをサポートします。これらを組み合わせることで、独立した運用のための豊富なオーケストレーション機能を備えた高度な自律型 AI エージェントを構築するための包括的なソリューションを提供します。

LangChain および の主な機能 LangGraph

LangChain および には、次の主要な機能LangGraphが含まれています。

- コンポーネントエコシステム – さまざまな自律型エージェント機能用の構築済みコンポーネントの広範なライブラリ。特殊なエージェントの迅速な開発を可能にします。詳細については、LangChainドキュメントの「クイックスタート」を参照してください。
- 基盤モデルの選択 – Anthropic Claude、Amazon Bedrock の Amazon Nova モデル (Premier、Pro、Lite、Micro) など、さまざまな推論機能のためのさまざまな基盤モデルをサポートします。詳細については、 LangChainドキュメントの「入力と出力」を参照してください。

- LLM API 統合 – Amazon Bedrock、 など、柔軟なデプロイOpenAIのための複数の大規模言語モデル (LLM) サービスプロバイダー向けの標準化されたインターフェイス。詳細については、LangChainドキュメントのLLMs」を参照してください。
- マルチモーダル処理 – テキスト、画像、オーディオ処理の組み込みサポートにより、リッチなマルチモーダル自律型エージェントインタラクションが可能になります。詳細については、LangChainドキュメントの「マルチモダリティ」を参照してください。
- グラフベースのワークフロー – 複雑な自律型エージェントの動作をステートマシンとして定義LangGraphし、高度な決定ロジックをサポートできます。詳細については、LangGraph「プラットフォーム GA の発表」を参照してください。
- メモリ抽象化 – 短期および長期のメモリ管理のための複数のオプション。これは、時間の経過とともにコンテキストを維持する自律型エージェントに不可欠です。詳細については、 LangChainドキュメントの「チャットボットにメモリを追加する方法」を参照してください。
- ツール統合 – さまざまなサービスと APIs、自律型エージェント機能を拡張します。詳細については、 LangChainドキュメントの「ツール」を参照してください。
- LangGraph プラットフォーム – 長時間稼働の自律型エージェントをサポートする、本番環境向けのマネージドデプロイおよびモニタリングソリューション。詳細については、LangGraph「プラットフォーム GA の発表」を参照してください。

LangChain と を使用するタイミング LangGraph

LangChain と LangGraphは、特に次のような自律型エージェントシナリオに適しています。

- 自律的な意思決定に高度なオーケストレーションを必要とする複雑な複数ステップの推論ワークフロー
- さまざまな自律機能のための構築済みのコンポーネントと統合の大規模なエコシステムへのアクセスを必要とするプロジェクト
- 自動システムの構築を希望する既存の Pythonベースの機械学習 (ML) インフラストラクチャと専門知識を持つチーム
- 長時間実行される自律型エージェントセッション全体で複雑な状態管理を必要とするユースケース

LangChain と の実装アプローチ LangGraph

LangChain と は、 LangGraphドキュメントに詳述されているように、ビジネスステークホルダーに構造化された実装アプローチLangGraphを提供します。このフレームワークにより、組織は次のことが可能になります。

- ビジネスプロセスを表す高度なワークフローグラフを定義します。
- 決定ポイントと条件付きロジックを使用して、複数ステップの推論パターンを作成します。
- マルチモーダル処理機能を統合して、さまざまなデータ型を処理します。
- 組み込みのレビューおよび検証メカニズムを使用して、品質管理を実装します。

このグラフベースのアプローチにより、ビジネスチームは複雑な意思決定プロセスを自律型ワークフローとしてモデル化できます。チームは、推論プロセスの各ステップと、決定パスを監査する能力を明確に把握できます。

と の実際の例 LangChain LangGraph

Vodafone は、LangChainエンタープライズのケーススタディで説明されているように、 LangChain (および LangGraph) を使用して自動エージェントを実装し、データエンジニアリングとオペレーションのワークフローを強化しています。パフォーマンスメトリクスを自律的にモニタリングし、ドキュメントシステムから情報を取得し、自然言語インタラクションを通じて実用的なインサイトを提示する内部 AI アシスタントを構築しました。

このVodafone実装では、LangChainモジュラードキュメントローダー、ベクトル統合、および複数の LLMs (OpenAI、LLaMA#3、および Gemini) のサポートを使用して、これらのパイプラインを迅速にプロトタイプ化およびベンチマークします。次に、モジュール式のサブエージェントLangGraphをデプロイして、マルチエージェントオーケストレーションを構造化するために使用されます。これらのエージェントは、収集、処理、要約、推論タスクを実行します。 APIs を通じてこれらのエージェントをクラウドシステムLangGraphに統合しました。

CrewAI

CrewAI は、特に GitHub で利用可能な自律型マルチエージェントオーケストレーションに焦点を当てたオープンソースフレームワークです。これは、人間の介入なしに複雑なタスクを解決するために協力する特殊な自律型エージェントのチームを作成するための構造化されたアプローチを提供します。 CrewAI は、ロールベースの調整とタスクの委任を強調します。

の主な機能 CrewAI

CrewAI には、次の主要な機能があります。

- ロールベースのエージェント設計 – 自律エージェントは、特殊な専門知識を実現するために、特定のロール、目標、バックストーリーで定義されます。詳細については、 CrewAIドキュメントの「Crafting Effective Agents」を参照してください。

- タスクの委任 – 機能に基づいて適切なエージェントにタスクを自動的に割り当てるための組み込みメカニズム。詳細については、 CrewAIドキュメントの「タスク」を参照してください。
- エージェントコラボレーション – 人間による仲介のない、エージェント間の自律的なコミュニケーションと知識共有のためのフレームワーク。詳細については、 CrewAIドキュメントの「コラボレーション」を参照してください。
- プロセス管理 – シーケンシャルおよび並列の自律タスク実行のための構造化ワークフロー。詳細については、 CrewAIドキュメントの「プロセス」を参照してください。
- 基盤モデルの選択 – Amazon Bedrock の Anthropic Claude、Amazon Nova モデル (Premier、Pro、Lite、Micro) など、さまざまな基盤モデルをサポートして、さまざまな自律推論タスクを最適化します。詳細については、 CrewAIドキュメントのLLMs」を参照してください。
- LLM API 統合 – Amazon Bedrock、、ローカルモデルのデプロイなどOpenAI、複数の LLM サービスインターフェイスとの柔軟な統合。詳細については、 CrewAIドキュメントの「プロバイダー設定の例」を参照してください。
- マルチモーダルサポート – テキスト、画像、その他のモダリティを処理して、包括的な自律型エージェントインタラクションを実現するための新しい機能。詳細については、 CrewAIドキュメントの「マルチモーダルエージェントの使用」を参照してください。

CrewAI を使用する場合

CrewAI は、次のような自律型エージェントのシナリオに特に適しています。

- 専門的なロールベースの専門知識を自律的に活用できる複雑な問題
- 複数の自律型エージェント間の明示的なコラボレーションを必要とするプロジェクト
- チームベースの問題分解が自律的な問題解決を改善するユースケース
- さまざまな自律エージェントロール間で懸念を明確に分離する必要があるシナリオ

の実装アプローチ CrewAI

CrewAI は、「 ドキュメントの開始方法」で説明されているように、ビジネスステークホルダー向けの AI エージェントアプローチのチームのロールベースの実装を提供します。 CrewAIこのフレームワークにより、組織は次のことが可能になります。

- 特定のロール、目標、専門知識領域を持つ特殊な自律型エージェントを定義します。
- 特殊な機能に基づいてエージェントにタスクを割り当てます。
- タスク間に明確な依存関係を確立して、構造化ワークフローを作成します。

- 複数のエージェント間のコラボレーションを調整して、複雑な問題を解決します。

このロールベースのアプローチは人間のチーム構造を反映しており、ビジネスリーダーは直感的に理解して実装できます。組織は、ヒューマンチームの運営方法と同様に、ビジネス目標を達成するために協力する専門分野を持つ自律型チームを作成できます。ただし、自律型チームは人間の介入なしに継続的に作業できます。

の実例 CrewAI

AWS は、CrewAI公開されたケーススタディで説明されているように、Amazon Bedrock と統合された CrewAI を使用して自律型マルチエージェントシステムを実装 AWS し、ベンダーに依存しない安全なフレームワークCrewAIを開発しました。CrewAI オープンソースの「フローとクルー」アーキテクチャは、Amazon Bedrock 基盤モデル、メモリシステム、コンプライアンスガードレールとシームレスに統合されます。

実装の主な要素は次のとおりです。

- 設計図とオープンソース – AWS および は、CrewAIエージェントを Amazon Bedrock モデルとオブザーバビリティツールにマッピングするリファレンスデザインをCrewAIリリースしました。https://aws.amazon.com/blogs/machine-learning/build-agentic-systems-with-crewai-and-amazon-bedrock/また、マルチエージェント AWS セキュリティ監査クルー、コードモダナイゼーションフロー、コンシューマーパッケージ製品 (CPG) バックオフィスオートメーションなどのサンプルシステムもリリースしました。
- オブザーバビリティスタック統合 – このソリューションは、Amazon CloudWatch、AgentOps、および によるモニタリングを埋め込みLangFuse、概念実証から本番稼働までのトレーサビリティとデバッグを可能にします。
- 実証済みの投資利益率 (ROI) – 初期のパイロットでは、大規模なコードモダナイゼーションプロジェクトの実行が 70#% 高速化され、CPG バックオフィスフローの処理時間が約 90#% 短縮されました。

AutoGen

AutoGen は、 によって最初にリリースされたオープンソースフレームワークMicrosoftです。 は、会話型および共同型の自律型 AI エージェントを有効にすることにAutoGen重点を置いています。これは、複雑な自律ワークフローのためにエージェント間の非同期のイベント駆動型インタラクションに重点を置いたマルチエージェントシステムを構築するための柔軟なアーキテクチャを提供します。

の主な機能 AutoGen

AutoGen には、次の主要な機能があります。

- 会話エージェント – 自律型エージェント間の自然言語会話を中心に構築され、対話を通じて高度な推論を可能にします。詳細については、 AutoGenドキュメントの「マルチエージェント会話フレームワーク」を参照してください。
- 非同期アーキテクチャ – ノンブロッキングの自律型エージェントインタラクションのためのイベント駆動型設計で、複雑な並列ワークフローをサポートします。詳細については、 AutoGenドキュメントの「非同期チャットのシーケンスでの複数のタスクの解決」を参照してください。
- Human-in-the-loop – 必要に応じて、他の自律型エージェントワークフローへのオプションの人間の参加を強力にサポートします。詳細については、 AutoGenドキュメントの「 エージェントでのヒューマンフィードバックの許可」を参照してください。
- コードの生成と実行 — コードを記述して実行できるコードに焦点を当てた自律型エージェント専用の機能。詳細については、 AutoGenドキュメントの「コード実行」を参照してください。
- カスタマイズ可能な動作 – さまざまなユースケースに対応する柔軟な自律型エージェント設定と会話制御。詳細については、 AutoGenドキュメントの agentchat.conversable_agent を参照してください。
- 基盤モデルの選択 – Anthropic Claude、Amazon Bedrock の Amazon Nova モデル (Premier、Pro、Lite、Micro) など、さまざまな基盤モデルをサポートし、さまざまな自律推論機能を提供します。詳細については、 AutoGenドキュメントの「LLM 設定」を参照してください。
- LLM API 統合 – Amazon Bedrock、、 OpenAIなどの複数の LLM サービスインターフェイスの標準化された設定Azure OpenAI。詳細については、 AutoGen API リファレンスの「oai.openai_utils」を参照してください。
- マルチモーダル処理 – リッチなマルチモーダル自律型エージェントインタラクションを可能にするテキストおよび画像処理をサポートします。詳細については、 AutoGenドキュメントの「マルチモーダルモデルの使用: GPT-4VAutoGen」を参照してください。

AutoGen を使用する場合

AutoGen は、次のような自律型エージェントのシナリオに特に適しています。

- 複雑な推論のために自律型エージェント間の自然な会話フローを必要とするアプリケーション
- 完全自律型オペレーションとオプションの人的監視機能の両方を必要とするプロジェクト
- 人間の介入なしでの自律的なコード生成、実行、デバッグを含むユースケース

- 柔軟で非同期の自律型エージェントの通信パターンを必要とするシナリオ

の実装アプローチ AutoGen

AutoGen は、 AutoGenドキュメントの「開始方法」で説明されているように、ビジネスステークホルダー向けの会話型実装アプローチを提供します。このフレームワークにより、組織は次のことが可能になります。

- 自然言語の会話を通じて通信する自律型エージェントを作成します。
- 複数のエージェント間でイベント駆動型の非同期インタラクションを実装します。
- 必要に応じて、完全自律型オペレーションとオプションの人間による監視を組み合わせます。
- 対話を通じてコラボレーションするさまざまなビジネス機能に特化したエージェントを開発します。

この会話型アプローチは、自律システムの推論をビジネスユーザーが透過的に利用できるようにします。意思決定者は、エージェント間の対話を観察して結論にどのように達するかを理解し、必要に応じて人間の判断が必要なときに会話に参加できます。

の実例 AutoGen

Magentic-One は、Microsoft AI Frontiers ブログで説明されているように、さまざまな環境で複雑なマルチステップタスクを自律的に解決するように設計されたオープンソースのゼネラリストマルチエージェントシステムです。その中核となるのはオーケストレーターエージェントです。オーケストレーターエージェントは、高レベルの目標を分解し、構造化台帳を使用して進捗状況を追跡します。このエージェントは、サブタスクを特殊なエージェント (WebSurfer、、FileSurfer、 など ComputerTerminal) に委任しCoder、必要に応じて再計画することで動的に適応します。

システムはAutoGenフレームワーク上に構築され、モデルに依存しず、デフォルトで GPT-4o になります。GAIA、、 などのベンチマーク全体で、タスク固有の調整AssistantBenchWebArenaなしで最先端のパフォーマンスを実現します。さらに、提案によるモジュール式の拡張性と厳格な評価もサポートしていますAutoGenBench。

LlamaIndex

LlamaIndex は、大規模言語モデル (LLMs) を外部データソースに接続して、高度な検索拡張生成 (RAG) およびエージェント AI アプリケーションを可能にするために特別に設計されたデータフレー

ムワークです。このフレームワークは、エージェントシステム、カスタムオーケストレーションパターン、システム統合のための抽象化と高速開発ワークフローを提供し、ナレッジ駆動型 AI ソリューションのtime-to-productionします。

の主な機能 LlamaIndex

LlamaIndex は、エンタープライズエージェント AI アプリケーションに特に適した包括的な機能を提供します。

- データ中心のアーキテクチャ – PDFs、MicrosoftWord ドキュメント、スプレッドシートなど、100 を超えるデータ形式からの情報の取り込み、インデックス作成、取得に優れています。このフレームワークは、エンタープライズデータを AI エージェント用に最適化されたクエリ可能なナレッジベースに変換します。詳細については、LlamaIndex のドキュメントを参照してください。
- 本番環境に対応したデプロイ – LlamaIndexは、 を通じてオープンソースフレームワークとマネージドサービスの両方を提供しLlamaCloud、セキュリティコントロール、スケーラビリティ、オブザーバビリティ統合、デプロイの柔軟性などのエンタープライズグレードの機能を提供します。詳細については、LlamaIndexフレームワークのドキュメントを参照してください。
- 高度なドキュメント処理 – 複雑なレイアウト、ネストされたテーブル、マルチモーダルコンテンツ、さらには手書きのメモを処理するドキュメント解析、抽出、インデックス作成、検索機能LlamaCloudを提供します。この高度な解析により、エージェントはグラフ、図、複雑なフォーマットを含む実際のエンタープライズドキュメントを効果的に操作できます。詳細については、LlamaCloud のドキュメントを参照してください。
- ワークフローオーケストレーション – 複数ステップのエージェントシステムを構築するためのイベント駆動型の非同期ファーストオーケストレーションエンジンLlamaAgentsを提供します。ワークフローは、ループ、並列実行、条件分岐、ステートフル再開などの複雑なパターンをサポートしているため、高度なエージェントインタラクションに最適です。詳細については、LlamaIndex ワークフローのドキュメントを参照してください。
- エージェント取り出し機能 – ハイブリッド検索、セマンティック検索、自動ルーティングなどの高度な取り出しモード。各クエリに最適な取り出し戦略をインテリジェントに決定します。フレームワークは、精度を高めるために再ランク付けして、複数のナレッジベースにわたる複合取得をサポートします。詳細については、LlamaIndexRAG ドキュメントを参照してください。
- オブザーバビリティと評価 – さまざまなオブザーバビリティと評価ツールとLlamaIndex統合されます。この統合機能は、アプリケーションのトレースとデバッグ、パフォーマンスの評価、コストのモニタリングに役立ちます。詳細については、「トレースとデバッグと評価」のLlamaIndexドキュメントを参照してください。 https://developers.llamaindex.ai/python/framework/module_guides/evaluating

LlamaIndex を使用する場合

LlamaIndex は、データ集約型のワークフローとナレッジ管理を強調するエージェント AI シナリオに特に適しています。

- エージェントが契約、レポート、マニュアル、規制申請などの大量のエンタープライズドキュメントからインサイトを処理、分析、抽出する必要があるドキュメントの多いアプリケーション
- 組織が大規模なインフラストラクチャ管理オーバーヘッドなしでドキュメント中心のエージェントを迅速に構築してデプロイしたい実稼働シナリオへの迅速なプロトタイプ
- 特にテーブル、イメージ、構造化データを含む複雑なマルチモーダルドキュメントを操作する場合、取得精度とコンテキストの関連性を優先する RAG ファーストアーキテクチャ
- 解析、分析、要約、コンプライアンスチェックなど、ドキュメント処理のさまざまな側面で特殊なエージェントを必要とするマルチエージェントドキュメントワークフロー

の実装アプローチ LlamaIndex

LlamaIndex は、さまざまな実装アプローチに対応する低レベルの構成要素と高レベルの抽象化の両方を提供します。

- LlamaIndex 高レベル APIs。このアプローチにより、エージェント AI を初めて使用するビジネスチームや開発者がLlamaIndexアクセスできるようになります。
- SharePoint、Amazon Simple Storage Service (Amazon S3)、データベース、APIs などの一般的なエンタープライズシステム向けの LlamaHub を介したエンタープライズ統合。このアプローチにより、既存のデータインフラストラクチャとシームレスに統合できます。
- 最大限の制御のためのオープンソースのセルフホストデプロイと、運用上のオーバーヘッドとエンタープライズ機能を削減するためのLlamaCloudマネージドサービス間の柔軟なデプロイオプション。
- アプリケーションはシンプルなクエリエンジンから開始し、要件の進化に応じてエージェント機能、マルチエージェントオーケストレーション、複雑なワークフローを段階的に追加できます。

の実例 LlamaIndex

この例では、航空ナビゲーションおよび運用ソリューションを専門とする航空会社の子会社に焦点を当てています。調整されていない AI チャットボットトライアルを試すという課題の増大に対処する必要があります。トライアルにより、組織全体で作業が繰り返され、開発サイクルが長くなり、コンプライアンスの障害が発生し、実装が分離されました。

エージェント作成をはるかに効率的にするLlamaIndexオープンソースフレームワーク上に構築された、再利用可能なテンプレートベースのソリューションである統合エージェントフレームワークを開発しました。チェーン指向とグラフベースの競合するフレームワークをいくつか比較しました。最終的に、柔軟な設計、モジュラーコンポーネント、本番稼働対応のオーケストレーションコントロールという 3 つの重要な利点を選択LlamaIndexしました。

このプラットフォームは、エージェントの開発とデプロイの時間を 512 時間から 64 時間に 87% 短縮します。この削減は、チームが約 50 行のコードと JSON 設定ファイルを使用してエージェントを構築できるようにすることで実現されました。チームは、セキュリティ、コンプライアンス、特権システムアクセスが組み込まれた統合フレームワークを活用しました。詳細については、LlamaIndex「 カスタマーのケーススタディ」を参照してください。

エージェント AI フレームワークの比較

自律型エージェント開発用のエージェント AI フレームワークを選択するときは、各オプションが特定の要件にどのように適合するかを検討してください。技術的な能力だけでなく、チームの専門知識、既存のインフラストラクチャ、長期的なメンテナンス要件など、組織の適合性も考慮してください。多くの組織は、自律型 AI エコシステムのさまざまなコンポーネントに複数のフレームワークを活用して、ハイブリッドアプローチの恩恵を受ける可能性があります。

次の表は、主要な技術的側面における各フレームワークの成熟度レベル (最強、強、適切、弱) を比較したものです。この表には、フレームワークごとに、本番デプロイオプションと学習曲線の複雑さに関する情報も含まれています。

Framework	AWS 統合	自動マルチエージェントサポート	自律ワークフローの複雑さ	マルチモーダル機能	基盤モデルの選択	LLM API 統合	本番稼働用デプロイ	学習曲線
AutoGen	弱い	強力	強力	適切	適切	強力	自分で行う (DIY)	急勾配
CrewAI	弱い	強力	適切	弱い	適切	適切	DIY	中

LangChain / LangGraph	適切	強力	最強	最強	最強	最強	プラットフォームまたは DIY	急勾配
LlamaIndex	適切	適切	強力	適切	強力	強力	プラットフォームまたは DIY	中
Strands Agents	最強	強力	最強	強力	強力	最強	DIY	中

エージェント AI フレームワークの選択に関する考慮事項

自律型エージェントを開発するときは、次の主要な要因を考慮してください。

- AWS インフラストラクチャ統合 – に大きく投資されている組織は AWS 、自律型ワークフロー AWS のサービス のために Strands Agentsと のネイティブ統合から最も恩恵を受けます。詳細については、AWS 「週次ラウンドアップ (ブログ)」を参照してください。AWS
- 基盤モデルの選択 – 自律型エージェントの推論要件に基づいて、優先基盤モデル (Amazon Bedrock の Amazon Nova モデルや Anthropic Claude など) に最適なサポートを提供するフレームワークを検討します。詳細については、 Anthropicウェブサイトの「効果的なエージェントの構築」を参照してください。
- LLM API 統合 – 本番デプロイ用の任意の大規模言語モデル (LLM) サービスインターフェイス (Amazon Bedrock や などOpenAI) との統合に基づいてフレームワークを評価します。詳細については、 ドキュメントの「モデルインターフェイス Strands Agents」を参照してください。
- マルチモーダル要件 – テキスト、画像、音声を処理する必要がある自律型エージェントの場合は、各フレームワークのマルチモーダル機能を検討してください。詳細については、 LangChain ドキュメントの「マルチモダリティ」を参照してください。
- 自律型ワークフローの複雑さ — 高度な状態管理を備えたより複雑な自律型ワークフローは、 の高度なステートマシン機能を優先する可能性がありますLangGraph。

- 自律型チームコラボレーション – 専門エージェント間の明示的なロールベースの自律型コラボレーションを必要とするプロジェクトは、 のチーム指向アーキテクチャの恩恵を受けることができますCrewAI。
- 自律型開発パラダイム – 自律型エージェントの会話型非同期パターンを好むチームは、 のイベント駆動型アーキテクチャを好むかもしれませんAutoGen。
- マネージド型またはコードベースのアプローチ – 最小限のコーディングでフルマネージド型のエクスペリエンスを希望する組織は、Amazon Bedrock エージェントを検討する必要があります。より詳細なカスタマイズを必要とする組織は、特定の自律型エージェントの要件により適した特殊な機能を備えた Strands Agentsやその他のフレームワークを好む場合があります。
- 自律システムの本番稼働準備 — 本番稼働用自律エージェントのデプロイオプション、モニタリング機能、エンタープライズ機能を検討します。

[Platforms] (プラットフォーム)

エージェント AI プラットフォームは、本稼働グレードのエージェントシステムのデプロイ、スケーリング、管理に必要な基本的なランタイム、オーケストレーション、統合レイヤーを提供します。フレームワークは、エージェントの構築方法を定義し、プロトコルはエージェントの通信方法を管理します。プラットフォームは、これらのエージェントが大規模に安全に運用、コラボレーション、進化する環境を提供します。

エージェントプラットフォームは、モデル実行、コンテキスト管理、ツール統合、オブザーバビリティ、ガバナンス機能を統合環境に組み合わせます。これらのプラットフォームにより、組織は実験からエンタープライズ規模のデプロイに移行できます。

このセクションでは、次の操作を行います。

- プラットフォームが重要な理由
- エージェント AI プラットフォームのタイプ
- プラットフォーム選択に関する考慮事項
- Amazon Bedrock エージェント
- Amazon Bedrock AgentCore

プラットフォームが重要な理由

エージェント AI プラットフォームは、本番環境で自律システムを運用しようとする組織にとって重要です。以下の機能を提供します。

- エージェントをホスト、スケーリング、調整するためのランタイムオーケストレーションを提供します。
- マルチエージェントワークフロー全体で状態、コンテキスト、メモリを管理します。
- エンタープライズ標準に沿ったセキュリティ、アイデンティティ、ガバナンスのコントロールを提供します。
- 標準 APIsまたはプロトコルを使用して、ツールエコシステムや外部システムと統合します。
- エージェントインタラクションとイベントフロー全体でオブザーバビリティと監査可能性を有効にします。
- クロスモデル相互運用性をサポートし、エージェントが 1 つの環境で複数の基盤モデルを使用できるようにします。

これらの機能により、個々のエージェントは、エンタープライズおよび規制の境界内で確実に動作できる調整された適応型システムになります。

エージェント AI プラットフォームのタイプ

エージェント AI プラットフォームは通常、次の 1 つ以上のカテゴリに分類されます。

- マネージドエージェント – フルマネージドプラットフォームは、組み込みのインフラストラクチャ、メモリ、オーケストレーション機能を提供します。これにより、運用上のオーバーヘッドが軽減され、本番稼働までの時間を短縮できます。
- オープンソースオーケストレーション – オープンソースのエージェントプラットフォームは、カスタマイズ可能な環境またはオンプレミスデプロイを好む組織に柔軟性と透明性を提供します。
- ハイブリッドエンタープライズ – ハイブリッドプラットフォームは、マネージドコンポーネントとセルフホストコンポーネントを統合し、クラウドマネージドサービスのスケーラビリティとエンタープライズシステムの制御を組み合わせます。

プラットフォーム選択に関する考慮事項

エージェント AI プラットフォームを選択または設計する場合、組織は次の点を考慮する必要があります。

- 統合の深さ — プラットフォームが既存のデータソース、ツール、プロトコルとどの程度統合されているかを評価します。
- スケーラビリティ – プラットフォームが自律的なワークロードとマルチエージェントコラボレーションをサポートするように動的にスケーリングできるようにします。
- セキュリティとコンプライアンス — 組織およびリージョンの要件に照らして、データのプライバシー、暗号化、ガバナンスの機能を評価します。
- 拡張性 – 新しいツール、モデル、またはエージェントを時間の経過とともに追加できるモジュールアーキテクチャのプラットフォームを選択します。
- オブザーバビリティ – エージェントインタラクションの詳細なテレメトリ、トレーサビリティ、監査ログを提供するプラットフォームを優先します。
- コスト効率 – サーバーレスモデルまたは使用量ベースのモデルを検討して、可変ワークロードのコストを最適化します。

Amazon Bedrock エージェント

Amazon Bedrock エージェントは、アプリケーションで自律型エージェントを構築および設定できるフルマネージドサービスです。基盤モデル、データソース、ソフトウェアアプリケーション、ユーザーとの会話間のやり取りをオーケストレーションできます。エージェントの作成に対する合理化されたアプローチでは、容量のプロビジョニング、インフラストラクチャの管理、カスタムコードの記述を行う必要はありません。

Amazon Bedrock エージェントの主な機能

Amazon Bedrock エージェントには、次の主要な機能があります。

- フルマネージドサービス – 容量をプロビジョニングしたり、基盤となるシステムを管理したりすることなく、インフラストラクチャ管理を完了します。詳細については、Amazon Bedrock ドキュメントの「AI エージェントを使用してアプリケーションのタスクを自動化する」を参照してください。
- API 駆動型開発 – モデル、手順、ツール、設定パラメータを指定して、シンプルな API コールを通じてエージェントを定義して実行します。詳細については、Amazon Bedrock ドキュメントの「エージェントを手動で作成および設定する」を参照してください。
- アクショングループ – API スキーマを使用してアクショングループを作成して、エージェントが実行できる特定のアクションを定義します。詳細については、Amazon Bedrock ドキュメントの「アクショングループを使用してエージェントが実行するアクションを定義する」を参照してください。
- ナレッジベースの統合 – Amazon Bedrock ナレッジベースにシームレスに接続して、組織のデータでエージェントのレスポンスを強化します。詳細については、Amazon Bedrock ドキュメントの「ナレッジベースを使用したエージェントの Augment レスポンス生成」を参照してください。
- 高度なプロンプトテンプレート – 前処理、オーケストレーション、ナレッジベースのレスポンス生成、後処理用のプロンプトテンプレートを使用してエージェントの動作をカスタマイズします。詳細については、Amazon Bedrock ドキュメントの「Amazon Bedrock の高度なプロンプトテンプレートを使用したエージェントの精度の向上」を参照してください。
- トレースとオブザーバビリティ – 組み込みトレース機能を使用して、エージェントのstep-by-stepの推論プロセスを追跡します。詳細については、Amazon Bedrock ドキュメントの「トレースを使用してエージェントのstep-by-step推論プロセスを追跡する」を参照してください。
- バージョニングとエイリアス – エージェントの複数のバージョンを作成し、制御されたロールアウトのエイリアスを介してデプロイします。詳細については、「Amazon Bedrock ドキュメント」

の「アプリケーションで Amazon Bedrock エージェントをデプロイして使用する」を参照してください。

Amazon Bedrock エージェントを使用するタイミング

Amazon Bedrock エージェントは、次のような自律型エージェントのシナリオに特に適しています。

- インフラストラクチャを管理せずにエージェントを構築およびデプロイするためのフルマネージド型のエクスペリエンスを必要とする組織
- コードではなく設定によるエージェントの迅速な開発とデプロイを必要とするプロジェクト
- ナレッジベースやガードレールなどの他の Amazon Bedrock 機能との緊密な統合からメリットを得るユースケース
- エージェントをゼロから構築するための社内リソースがないが、本番環境対応の自律機能が必要なチーム

Amazon Bedrock エージェントの実装アプローチ

Amazon Bedrock エージェントは、ビジネスステークホルダー向けに設定ベースの実装アプローチを提供します。このサービスにより、組織は次のことが可能になります。

- 複雑なコードを記述せずに、 AWS マネジメントコンソール または API コールを通じてエージェントを定義します。
- エージェントが実行できる APIsとオペレーションを指定するアクショングループを作成します。
- ナレッジベースを接続して、ドメイン固有の情報をエージェントに提供します。
- ビジュアルインターフェイスを使用して、エージェントの動作をテストして反復処理します。

このマネージド型アプローチにより、ビジネスチームは AI モデル開発やインフラストラクチャ管理に関する深い技術的専門知識を必要とせずに、自律型エージェントを迅速に開発およびデプロイできます。

Amazon Bedrock エージェントの実例

このAWS ブログ記事で説明されている財務オペレーション (FinOps) ソリューションは、Amazon Bedrock マルチエージェントフレームワークを使用して AI 駆動型のクラウドコスト管理アシスタントを作成します。費用対効果の高い Amazon Nova 基盤モデルは、中央の FinOps スーパーバイザーエージェントがタスクを専門エージェントに委任するソリューションを強化します。これらのエー

ジェントは、 を使用して AWS 支出データを取得および分析 AWS Cost Explorer し、 を使用してコスト削減の推奨事項を生成します AWS Trusted Advisor。

このシステムには、Amazon Cognito を介した安全なユーザーアクセス、 でホストされるフロントエンド AWS Amplify、リアルタイム分析と予測のための AWS Lambda アクショングループが含まれています。財務チームは、「2025 年 2 月のコストはいくらだったか」などの自然言語クエリを聞くことができます。システムは、詳細な内訳、最適化の提案、予測で応答します。これらはすべて、 を使用してデプロイされたスケーラブルなサーバーレスアーキテクチャ内で行われます AWS CloudFormation。

Amazon Bedrock AgentCore

Amazon Bedrock AgentCore は、あらゆるフレームワーク、モデル、プロトコルを使用して、高機能エージェントを大規模に安全に構築、デプロイ、運用するためのエージェントプラットフォームです。AgentCore を使用すると、インフラストラクチャ管理なしで、以下を実行できます。

- エージェントをより迅速に構築します。
- エージェントがツールとデータ間でアクションを実行できるようにします。
- 低レイテンシーと拡張ランタイムでエージェントを安全に実行します。
- 本番環境のエージェントをモニタリングします。

AgentCore は、特殊なエージェントインフラストラクチャの構築に伴う差別化されていない負担を排除し、エージェントの本番稼働までの時間を短縮します。そのサービスは一緒に使用することも独立して使用することもできます。また、CrewAI、、LangGraphLlamaIndex、 などのフレームワークと互換性がありますStrands Agents。AgentCore は、Amazon Bedrock の内外で利用可能な基盤モデルとも互換性があり、最高の柔軟性を提供します。

AgentCore は、いくつかの主要なサービスで構成されています。

- Amazon Bedrock AgentCore ランタイム – AI エージェントやツールのデプロイと実行に必要なインフラストラクチャを管理することなく、エージェントをホストして実行するための安全でサーバーレスでスケーラブルな環境を提供します。
- Amazon Bedrock AgentCore Memory – マネージドメモリシステムを提供し、エージェントは即時および長期的な知識を維持することで、よりパーソナライズされた一貫性のある会話のためにインタラクションのコンテキストを保持できます。
- Amazon Bedrock AgentCore Gateway – エージェントに適したツールを作成、保護、検索するプロセスを簡素化します。AgentCore Gateway を使用すると、デベロッパーは APIs、Lambda 関数、

および既存のサービスをモデルコンテキストプロトコル (MCP) 互換ツールに変換し、エージェントに提供できます。

- Amazon Bedrock AgentCore Identity – AI エージェントの開発を加速する、安全でスケーラブルなエージェント ID とアクセス管理サービスを提供します。AgentCore Identity を使用すると、検証可能な一意の ID をエージェントに割り当てることができ、きめ細かなアクセスコントロールと、エージェントによるエンタープライズシステムとの安全なやり取りが可能になります。
- Amazon Bedrock AgentCore 組み込みツール – 組み込みツールを使用して開発とテストのワークフローを強化できます。これらのツールを使用してアプリケーションを効果的に操作し、AI エージェントがサンドボックス環境で安全にコードを記述して実行できるようにします。ブラウザツールを使用して、AI エージェントがウェブサイトを大規模に操作できるようにします。
- Amazon Bedrock AgentCore Observability – ログ記録とモニタリング機能を提供し、エージェントのパフォーマンスと動作をリアルタイムで可視化して、デバッグと最適化を容易にします。

AgentCore の主な機能

AgentCore には、次の主要な機能が含まれています。

- フルマネージド型で拡張可能 – AgentCore はフルマネージド型サービスです。つまり、 は基盤となるインフラストラクチャとメンテナンス AWS を処理します。また、拡張可能で、エージェントの機能をカスタマイズして強化できます。詳細については、AgentCore ドキュメントの「AgentCore ランタイムの開始方法」を参照してください。 AgentCore
- 長期メモリと短期メモリ – 現在の会話や長期的な知識のコンテキストを再現するメモリシステムをエージェントに装備することで、よりパーソナライズされた関連性の高いインタラクションを提供します。詳細については、「 AgentCore ドキュメント」の「AgentCore メモリの開始方法」を参照してください。 AgentCore
- ツールの開発と統合の簡素化 – エージェントが単一の安全なエンドポイントを通じてツールを検出して使用できるようにします。既存のエンタープライズリソースをわずか数行のコードでエージェント対応ツールにすばやく変換できるため、開発者は独自の機能の構築に集中できます。詳細については、AgentCore ドキュメントの「AgentCore Gateway の開始方法」を参照してください。 AgentCore
- 安全でスケーラブルなインフラストラクチャ – AgentCore は、エージェントのデプロイと運用のための安全でスケーラブルな環境を提供します。これには、ID とアクセスの管理、データ暗号化、ネットワークセキュリティの機能が含まれています。詳細については、AgentCore ドキュメントの「AgentCore Identity の開始方法」を参照してください。 AgentCore

- さまざまなツールとの統合 – エージェントを、コードインタープリタや AgentCore 組み込みツールを使用して構築できるブラウザツールなど、さまざまなツールと統合できます。詳細については、AgentCore ドキュメントの「AgentCore Code Interpreter の開始方法」およびAgentCore Browser の開始方法」を参照してください。 AgentCore
- 包括的なオブザーバビリティとモニタリング – 包括的なツールを使用してエージェントを詳細に可視化し、本番環境でのパフォーマンスをトレース、デバッグ、モニタリングします。エージェントの実行パス全体を視覚化して、推論を監査し、障害を解決します。リアルタイムのダッシュボードと標準化されたテレメトリデータを使用して、主要な運用メトリクスを追跡します。詳細については、AgentCore ドキュメントの「Amazon Bedrock AgentCore リソースにオブザーバビリティを追加する」を参照してください。 AgentCore

AgentCore を使用するタイミング

AgentCore は、次のような自律型エージェントシナリオに特に適しています。

- インフラストラクチャ、セキュリティ、組み込みツール、オブザーバビリティ、スケーリングを処理するフルマネージドサービスを使用して、開発を加速し、運用オーバーヘッドを削減したい組織
- 連携または独立して動作し、 CrewAI や などのフレームワークLangGraph、および任意のソースからの基盤モデルと互換性があるモジュラーサービスによる柔軟性を必要とするプロジェクト
- コンテキストを維持し、過去のやり取りから学び、パーソナライズされた関連する応答を提供する必要がある、ステートフルで会話型のエージェントを必要とするユースケース
- さまざまなアプリケーション、データソース、APIs と簡単に統合することで複雑なタスクを実行できるエージェント

AgentCore の実装アプローチ

AgentCore は、オープンソースまたはカスタムエージェントフレームワークを使用して構築された概念実証から本番稼働に移行する組織向けに設計されています。AgentCore を使用すると、組織は以下を実行できます。

- エージェントをサーバーレスインフラストラクチャに安全にデプロイし、エンドツーエンド end-to-end のセキュリティとコンプライアンスのためのセッション分離と組み込みの ID とアクセス管理により、あらゆるフレームワークとモデルをサポートします。スターターツールキットを使用して、主要なエージェントフレームワークの AgentCore ランタイムエージェントをすばやく作成します。

- エージェントを強化するには、永続メモリを統合してコンテキストを保持し、AgentCore Gateway を介したツールの開発と統合を簡素化します。組み込みのブラウザツールとコードインタープリタを高度なワークフローに活用します。
- Amazon CloudWatch Application Insights と を搭載したオブザーバビリティダッシュボードを使用して、本番環境の AI エージェントをトレース、デバッグ、モニタリングし OpenTelemetry、AgentCore リソース (ランタイム、メモリ、ゲートウェイ、ツール) の主要なメトリクスを追跡します。
- フルマネージドのモジュラーサービス、組み合わせ可能なブロックをまとめて、または個別に、エージェントフレームワークとモデルプロバイダーを使用して、デプロイとイノベーションを加速します。この柔軟性により、組織はプロトタイプから本番稼働への移行を迅速化できます。

このマネージド型アプローチにより、組織はあらゆる規模でエンタープライズグレードの AI エージェントとマルチエージェントシステムを迅速かつ安全に構築、デプロイ、実行できます。

AgentCore の実際の例

AWS は、ラテンアメリカ最大の銀行の 1 つが AI/ML を長年使用して、ハイパーパーソナライズされた安全なデジタルバンキングエクスペリエンスを提供していることを確認しました。銀行は AgentCore を使用してエージェント AI サービスを拡張し、直感的なインタラクション、セキュリティの強化、自動化の強化をお客様に提供しています。CTO によると、AgentCore は顧客コミットメントを大規模に達成するための取り組みをサポートすることが期待されています。AgentCore は、開発者にエージェントを構築および管理するためのツールと柔軟性を提供し、財務規制へのコンプライアンスを確保します。

プロトコル

AI エージェントは、他のエージェントやサービスとやり取りするために標準化された通信プロトコルを必要とします。エージェントアーキテクチャを実装している組織は、相互運用性、ベンダーの独立性、投資の将来性に関して大きな課題に直面しています。

このセクションでは、柔軟性と相互運用性を最大化するオープンスタンダードに焦点を当ててagent-to-agentプロトコルランドスケープをナビゲートするのに役立ちます。(agent-to-toolプロトコルの詳細については、このガイドの後半にある「ツール統合戦略」を参照してください)。

このセクションでは、2024 Anthropic 年に によって最初に開発されたオープンスタンダードである Model Context Protocol (MCP) について説明します。現在、 はプロトコルの開発と実装に貢献することで MCP AWS を積極的にサポートしています。 AWS は、、LangGraph、 CrewAIなどの主要なオープンソースエージェントフレームワークと協力してLlamaIndex、プロトコルでのエージェント間通信の未来を形成しています。詳細については、「Open Protocols for Agent Interoperability Part 1: Inter-Agent Communication on MCP」(AWS ブログ) を参照してください。

このセクションでは、次の操作を行います。

- プロトコルの選択が重要な理由
- Agent-to-agentプロトコル
- エージェントプロトコルの選択
- エージェントプロトコルの実装戦略
- MCP の開始方法
- ???

プロトコルの選択が重要な理由

プロトコルの選択は、AI エージェントアーキテクチャを構築および進化する方法を根本的に形成します。エージェントフレームワーク間の移植性をサポートするプロトコルを選択することで、特定のニーズに合わせてさまざまなエージェントシステムとワークフローを柔軟に組み合わせることができます。

オープンプロトコルを使用すると、エージェントを複数のフレームワークに統合できます。たとえば、 LangChainを使用してラピッドプロトタイプを作成し、 を使用して本番稼働用システムを実装

しStrands Agents、MCP や Agent2Agent (A2A) プロトコルなどの一般的なプロトコルを介して通信します。この柔軟性により、特定の AI プロバイダーへの依存を減らし、既存のシステムとの統合を簡素化し、時間の経過とともにエージェントの機能を強化できます。

また、適切に設計されたプロトコルは、エージェントエコシステム全体で認証と認可のための一貫したセキュリティパターンを確立します。最も重要なのは、プロトコルの移植性により、新しいエージェントフレームワークと機能の導入の自由が維持されることです。オープンプロトコルを選択すると、サードパーティーシステムとの相互運用性を維持しながら、エージェント開発への投資が保護されます。

オープンプロトコルの利点

独自の拡張機能を実装する場合やカスタムエージェントシステムを構築する場合、オープンプロトコルには魅力的な利点があります。

- ドキュメントと透明性 — 通常、包括的なドキュメントと透過的な実装を提供します。
- コミュニティサポート – トラブルシューティングとベストプラクティスのためのより広範な開発者コミュニティへのアクセス
- 相互運用性の保証 – 拡張機能がさまざまな実装で機能する保証を強化
- 将来の互換性 — 変更が中断されたり、廃止されたりするリスクを軽減
- 開発への影響 — プロトコルの進化に貢献する機会

Agent-to-agentプロトコル

次の表は、複数のエージェントがコラボレーション、タスクの委任、および情報の共有を可能にするエージェントプロトコルの概要を示しています。

[プロトコル]	に最適	考慮事項
MCP エージェント間通信	柔軟なエージェントコラボレーションパターンを求める組織	• エージェントagent-to-agent通信の既存の基盤を構築する AWS 、 によって提案されたモデルコンテキストプロトコル (MCP) の拡張 • OAuth ベースのセキュリティによるシームレスなエー

		ジェントコラボレーションを実現
A2A プロトコル	クロスプラットフォームエージェントエコシステム	• によってバックアップ Google • MCP と比較して導入が限られている新しい標準

プロトコルオプション間の決定

agent-to-agent通信を実装する場合は、特定の通信要件を適切なプロトコル機能と一致させます。異なるインタラクションパターンには、異なるプロトコル機能が必要です。次の表は、一般的な通信パターンの概要を示し、各シナリオに最適なプロトコルの選択を推奨しています。

パターン:	説明	理想的なプロトコルの選択
シンプルなリクエストとレスポンス	エージェント間の 1 回限りのインタラクション	ステートレスフローを持つ MCP
ステートフルな対話	コンテキストを使用した継続的な会話	セッション管理による MCP
マルチエージェントコラボレーション	複数のエージェント間の複雑なやり取り	MCP エージェント間または AutoGen
チームベースのワークフロー	ロールが定義された階層エージェントチーム	MCP エージェント間、CrewAI、または AutoGen

コミュニケーションパターン以外にも、いくつかの技術的および組織的な要因がプロトコルの選択に影響を与える可能性があります。次の表は、特定の実装要件に最も近いプロトコルを評価するのに役立つ重要な考慮事項の概要を示しています。

考慮事項	説明	例
セキュリティモデル	認証と認可の要件	MCP の OAuth 2.0

デプロイ環境	エージェントが実行して通信する場所	分散マシンまたは単一マシン
エコシステムの互換性	既存のエージェントフレームワークとの統合	LangChain、または Strands Agents
スケーラビリティのニーズ	エージェントインタラクションの予想される増加	MCP のストリーミング機能

エージェントプロトコルの選択

本番稼働用エージェントシステムを構築するほとんどの組織では、モデルコンテキストプロトコル (MCP) はagent-to-agent通信の最も包括的で十分にサポートされている基盤を提供します。MCP は、 AWS およびオープンソースコミュニティからの積極的な開発貢献からメリットを得ます。

適切なエージェントプロトコルを選択することは、エージェント AI を効果的に実装したい組織にとって重要です。考慮事項は、組織のコンテキストによって異なります。

エージェントプロトコルの選択に関する考慮事項

エージェント AI システムのプロトコルを選択するときは、次のベストプラクティスを考慮する必要があります。

- オープンスタンダードの優先順位付け – 組織は、長期的な相互運用性、拡張性を確保し、ベンダーロックインのリスクを減らすために、MCP などのオープンプロトコルを採用する必要があります。
- スピードと柔軟性のバランス — スタートアップと早期導入者は、迅速な開発のために十分にサポートされている独自のプロトコルから始めることができますが、システムが成熟するにつれて標準を開くための移行パスを定義する必要があります。
- 抽象化レイヤーを実装する – 企業はプロトコル抽象化を実装して、移行を簡素化し、ハイブリッド導入を可能にし、将来を見越した統合戦略を実現する必要があります。
- セキュリティとコンプライアンスを強調する – 規制対象業界の組織は、ガバナンスとコンプライアンスの要件を満たすために、堅牢な認証、暗号化、監査機能を備えたプロトコルを選択する必要があります。
- エコシステムの成熟度を評価する – すべての組織は、持続可能性を確保し、技術的負債を最小限に抑えるために、各プロトコルのヘルス、導入、コミュニティサポートを評価する必要があります。

- 標準の開発に取り組む – 組織は、プロトコルの進化を形成し、ベストプラクティスに影響を与えるために、標準団体またはオープンソースコミュニティに参加する必要があります。
- データ主権の考慮 — 政府機関と規制対象セクターは、プロトコルの選択が、デプロイリージョン全体のデータレジデンシーと主権の要件と一致していることを確認する必要があります。
- マネージドサービスを活用する – 可能な限り、エージェントプロトコルのマネージド実装またはサーバーレス実装を使用して、運用の複雑さを軽減し、デプロイを高速化します。

エージェントプロトコルの実装戦略

組織全体でエージェントプロトコルを効果的に実装するには、以下の戦略的ステップを検討してください。

1. 標準の調整から始める – 可能な場合は、確立されたオープンプロトコルを採用します。
2. 抽象化レイヤーの作成 – システムと特定のプロトコルの間にアダプターを実装します。
3. オープンスタンダードに貢献 — プロトコル開発コミュニティに参加します。
4. プロトコルの進化をモニタリングする – 新しい標準と更新について常に情報を得ます。
5. 相互運用性を定期的にテストする – 実装に互換性があることを確認します。

MCP の開始方法

AWS は、プロトコルの開発と実装に貢献することで、モデルコンテキストプロトコル (MCP) を積極的にサポートしています。 AWS は、LangGraph、、 CrewAIなどの主要なオープンソースエージェントフレームワークと連携してLlamaIndex、プロトコルでのエージェント間通信の未来を形成しています。

エージェントアーキテクチャに MCP を実装するには、次のアクションを実行します。

1. Strands Agents SDK などのフレームワークでの MCP 実装について説明します。
2. Model Context Protocol の技術ドキュメントを確認してください。
3. エージェント相互運用性のオープンプロトコル パート 1: MCP でのエージェント間通信 (AWS ブログ) を読み、エージェントの相互運用性について学びます。
4. MCP コミュニティに参加して、プロトコルの進化に影響を与えます。

MCP は、エージェントが外部データやサービスとやり取りできるようにする通信レイヤーを提供し、エージェントが他のエージェントとやり取りできるようにするためにも使用できます。プロト

コルの Streamable HTTP トランスポート実装により、デベロッパーはホイールを再考案することなく、包括的な一連のインタラクションパターンを使用できます。これらのパターンは、ステートレスリクエスト/レスポンスフローと永続的 IDs を使用したステートフルセッション管理の両方をサポートします。

MCP などのオープンプロトコルを採用することで、AI テクノロジーの進化に合わせて柔軟性、相互運用性、適応性を維持するエージェントシステムを構築するように組織を配置できます。agent-to-toolプロトコルの実装の詳細については、このガイドの後半にある「ツール統合戦略」を参照してください。

A2A の開始方法

Agent2Agent (A2A) プロトコルは、共有セマンティックレイヤーを介してエージェント間の分散コラボレーションを可能にします。A2A では、すべての作業を中央オーケストレーター経由でルーティングする代わりに、エージェントは軽量の JSON ベースのプロトコルを使用して、相互検出、機能のアドバタイズ、タスクのネゴシエート、コンテキストの共有を行うことができます。各エージェントは機能マニフェストを発行します。

次の例は、エージェントのサポートされているアクション、必要な入力、運用メタデータをアドバタイズして検出とタスクネゴシエーションを可能にする、簡略化された A2A 機能マニフェストを示しています。

```
{
  "can": ["summarize.text", "extract.keywords"],
  "needs": ["document.input"],
  "meta": { "version": "1.0.3", "latencyMs": 120 }
}
```

このモデルにより、動的な機能マッチング、タスク中の委任、組織間のコラボレーションが可能になります。エージェントはタスクを中心に自己編成し、一時的な作業グループを形成し、新しい機能がシステムに出入りするにつれて適応できます。

A2A は、シンプルなステートレスリクエストから複数ステップのネゴシエーションセッションまで、次のようなインタラクションをサポートします。

- peer-to-peerのダイレクトメッセージングによる低レイテンシーのコラボレーション
- エージェントが最適なピアを選択するセマンティックタスクネゴシエーション
- 能力ベースの検出により、緊急の作業分担を実現

- ステートフルなマルチステップインタラクションのためのセッションアンカーリング

A2A のようなオープンでエージェントネイティブなプロトコルを採用することで、組織はモジュラーで相互運用可能で、クロスボーダーコラボレーションが可能な AI システムを作成します。A2A は、エージェントのエコシステムを柔軟に保ち、リジッドオーケストレーションレイヤーや事前の結合を必要とせずに、新しいエージェント、チーム、または外部システムの導入に合わせて進化させることができます。

エージェントアーキテクチャに A2A プロトコルを実装するには、次のアクションを実行します。

1. A2A プロトコル仕様を確認する – Agent2Agent (A2A) プロトコル仕様の最新バージョンを読み、機能マニフェスト、ネゴシエーションフロー、エージェントのハンドシェイクがどのように動作するかを確認してください。
2. A2A-compatibleランタイムの探索 – Strands Agents SDK などのフレームワークやA2A-style機能マニフェストとpeer-to-peerネゴシエーションをサポートするカスタムランタイムレイヤーを評価します。
3. エージェントの機能マニフェストを実装する – 各エージェントの `can`、`needs`、および `meta`フィールドを定義して、検出、マッチメーキング、インテントレベルのコラボレーションを有効にします。
4. A2A ネゴシエーションパターンを試す – リクエストオファー承諾ループ、構造化された機能クエリ、または gossip ベースの検出を使用して、エージェントがタスクを処理するべきユーザーについてどのような理由があるかを理解します。
5. 混合インフラストラクチャ環境で A2A をテストする – A2A ピアネゴシエーションを Amazon EventBridge AWS を介して にネイティブなイベントルーティングと組み合わせて、ハイブリッド調整パターンを評価します。
6. A2A コミュニティに参加する – オープンワーキンググループと連携し、拡張機能、セキュリティの推奨事項、ベンダー間の相互運用性の改善について最新の状態を維持し、プロトコルの開発に貢献します。

ツール

AI エージェントは、外部ツール、APIs、データソースを操作して有用なタスクを実行することで価値を提供します。適切なツール統合戦略は、エージェントの機能、セキュリティ体制、長期的な柔軟性に直接影響します。

このセクションでは、自由と柔軟性を最大化するオープンスタンダードに焦点を当てて、ツール統合の状況をナビゲートするのに役立ちます。このセクションでは、ツール統合用の Model Context Protocol (MCP) に焦点を当て、エージェントワークフローを強化するフレームワーク固有のツールと特殊なメタツールを確認します。

このセクションでは、次の操作を行います。

- ツールカテゴリ
- プロトコルベースのツール
- フレームワークネイティブツール
- メタツール
- ツール統合戦略
- ツール統合のセキュリティのベストプラクティス

ツールカテゴリ

エージェントシステムの構築には、主に 3 つのカテゴリのツールが含まれます。

プロトコルベースのツール

プロトコルベースのツールは、agent-to-tool間の通信に標準化されたプロトコルを使用します。

- MCP ツール – ローカル実行オプションとリモート実行オプションの両方でフレームワーク間で機能する標準ツールを開きます。
- OpenAI 関数呼び出し — OpenAIモデルに固有の独自のツール。
- Anthropic ツール – Anthropic Claude モデルに固有の独自のツールを呼び出すOpenAI関数のバリエーション。

フレームワークネイティブツール

フレームワークネイティブツールは、特定のエージェントフレームワークに直接組み込まれています。

- Strands Agents ツール – Strands Agentsフレームワークに固有の軽量でquick-to-implementツール。
- LangChain ツール – LangChainエコシステムと緊密に統合された Pythonベースのツール。
- LlamaIndex ツール – 内のデータの取得と処理に最適化されたツールLlamaIndex。

メタツール

メタツールは、外部アクションを直接実行することなく、エージェントワークフローを強化します。

- ワークフローツール – エージェント実行フロー、分岐ロジック、状態管理を管理します。
- エージェントグラフツール – 複雑なワークフローで複数のエージェントを調整します。
- メモリツール – エージェントセッション全体で永続的なストレージと情報の取得を提供します。
- リフレクションツール – エージェントが独自のパフォーマンスを分析して改善できるようにします。

プロトコルベースのツール

プロトコルベースのツールを検討する場合、モデルコンテキストプロトコル (MCP) はツール統合の最も包括的で柔軟な基盤を提供します。AWS エージェントの相互運用性に関するオープンソースブログ記事で述べたように、 AWS は MCP を戦略的プロトコルとして採用し、その開発に積極的に貢献しています。

次の表に、MCP ツールのデプロイのオプションを示します。

デプロイモデル	説明	に最適	実装
ローカル stdio ベース	エージェントと同じプロセスで実行されるツール	開発、テスト、シンプルなツール	ネットワークオーバーヘッドなしで迅速に実装

ローカルサーバー送信イベント (SSE) ベース	ツールはローカルで実行されますが、HTTP 経由で通信します	懸念を分離したより複雑なローカルツール	分離は向上するが、レイテンシーは低い
リモート HTTP ストリーミング可能	リモートサーバーで実行されるツール	本番環境と共有ツール	スケーラブルで一元管理

公式 MCP SDKs は MCP ツールの構築に使用できます。

- Python SDK – 完全なプロトコルサポートによる包括的な実装
- TypeScript SDK – ウェブアプリケーションの JavaScript/TypeScript 実装
- Java SDK – エンタープライズアプリケーションの Java 実装

これらの SDKsは、プロトコル仕様の一貫した実装により、任意の言語で MCP 互換ツールを作成するための構成要素を提供します。

さらに、 AWS は Strands Agents SDK に MCP を実装しています。Strands Agents SDK を使用すると、MCP 互換ツールを簡単に作成して使用できます。包括的なドキュメントは、Strands AgentsGitHub リポジトリで入手できます。より簡単なユースケースやStrands Agentsフレームワーク外で作業する場合、公式 MCP SDKs は複数の言語でプロトコルを直接実装します。

MCP ツールのセキュリティ機能

MCP ツールのセキュリティ機能には以下が含まれます。

- OAuth 2.0/2.1 認証 – 業界標準認証
- アクセス許可のスコープ — ツールのきめ細かなアクセスコントロール
- ツール機能検出 — 使用可能なツールの動的検出
- 構造化エラー処理 — 一貫したエラーパターン

MCP ツールの開始方法

ツール統合用の MCP を実装するには、次のアクションを実行します。

1. 本番環境対応の MCP 実装については、 Strands Agents SDK をご覧ください。
2. MCP 技術ドキュメントを確認して、主要な概念を理解します。

3. このAWS オープンソースブログ記事で説明されている実用的な例を使用します。
4. リモートツールに進む前に、シンプルなローカルツールから始めます。
5. MCP コミュニティに参加して、プロトコルの進化に影響を与えます。

AgentCore Gateway の詳細

Amazon Bedrock AgentCore Gateway は、開発者が MCP ツールやその他のターゲットエンドポイントを大規模に構築、デプロイ、検出、接続するための簡単で安全な方法を提供します。AgentCore Gateway を使用すると、デベロッパーAPIs、 AWS Lambda 関数、既存のサービスを MCP 互換ツールに変換できます。次に、わずか数行のコードで、これらのツールを AgentCore Gateway エンドポイントを介してエージェントが使用できるようにします。AgentCore Gateway は OpenAPI、、Smithy、および Lambda を入力タイプとしてサポートし、フルマネージドサービスで包括的な進入認証と退出認証の両方を提供する唯一のソリューションです。

フレームワークネイティブツール

Model Context Protocol (MCP) は最も柔軟な基盤を提供しますが、フレームワークネイティブツールは特定のユースケースに利点を提供します。

Strands Agents SDK は、シンプルなオペレーションに最小限のオーバーヘッドを必要とする軽量設計を特徴とする Pythonベースのツールを提供します。これにより、迅速な実装が可能になり、開発者はわずか数行のコードでツールを作成できます。さらに、Strands Agentsフレームワーク内でシームレスに機能するように緊密に統合されています。

次の例は、 を使用してシンプルな気象ツールを作成する方法を示していますStrands Agents。開発者は、最小限のコードオーバーヘッドでPython関数をエージェントアクセス可能なツールにすばやく変換し、関数のドキュメントから適切なドキュメントを自動的に生成できます。

```
#Example of a simple Strands native tool

@tool

def weather(location: str) -> str:

"""Get the current weather for a location""" #

Implementation here

return f"The weather in {location} is sunny."
```

迅速なプロトタイプ作成やシンプルなユースケースでは、フレームワークネイティブツールが開発を加速できます。ただし、本番稼働用システムの場合、MCP ツールはフレームワークネイティブツールよりも相互運用性と将来の柔軟性に優れています。

次の表は、他のフレームワーク固有のツールの概要を示しています。

Framework	ツールタイプ	利点	考慮事項
AutoGen	関数定義	強力なマルチエージェントサポート	Microsoft エコシステム
LangChain	Python クラス	構築済みのツールの大規模なエコシステム	フレームワークのロックイン
LlamaIndex	Python 関数	データオペレーション用に最適化	に制限 LlamaIndex

メタツール

メタツールは外部システムと直接やり取りしません。代わりに、エージェントパターンを実装することでエージェントの機能を強化します。このセクションでは、ワークフロー、エージェントグラフ、メモリメタツールについて説明します。

ワークフローメタツール

ワークフローメタツールは、エージェント実行のフローを管理します。

- 状態管理 – 複数のエージェントインタラクションのコンテキストを維持する
- 分岐ロジック – 条件付き実行パスを有効にする
- 再試行メカニズム – 高度な再試行戦略で障害を処理する

ワークフローメタツールを使用したフレームワークの例には、 LangGraphおよび Strands Agents ワークフロー機能が含まれます。

エージェントグラフのメタツール

エージェントグラフのメタツールは、複数のエージェントを連携させて調整します。

- タスクの委任 – 専門エージェントにサブタスクを割り当てる
- 結果の集約 – 複数のエージェントからの出力を結合する
- 競合の解決 – エージェント間の不一致を解決する

AutoGen や などのフレームワークは、エージェントグラフの調整にCrewAI特化しています。

メモリメタツール

メモリメタツールは、永続的なストレージと取り出しを提供します。

- 会話履歴 – セッション間でコンテキストを維持する
- ナレッジベース – ドメイン固有の情報を保存および取得する
- ベクトルストア – セマンティック検索機能を有効にする

MCP のリソースシステムは、さまざまなエージェントフレームワークで動作するメモリメタツールを実装するための標準化された方法を提供します。

ツール統合戦略

ツール統合戦略の選択は、エージェントが達成できることとシステムの進化の容易さに直接影響します。フレームワークネイティブツールとメタツールを戦略的に使用しながら、モデルコンテキストプロトコル (MCP) などのオープンプロトコルを優先します。これにより、AI テクノロジーの進歩に合わせて柔軟で強力なツールエコシステムを構築できます。

ツール統合に対する以下の戦略的アプローチは、組織の当面のニーズを満たしながら柔軟性を最大化します。

1. 基盤として MCP を採用する – MCP は、強力なセキュリティ機能を持つツールにエージェントを接続するための標準化された方法を提供します。以下の主要なツールプロトコルとして MCP から始めます。
   - 複数のエージェント実装で使用される戦略的ツール。
   - 堅牢な認証と認可を必要とするセキュリティ重視のツール。
   - 本番環境でリモート実行が必要なツール。
2. 必要に応じてフレームワークネイティブツールを使用する — 以下のフレームワークネイティブツールを検討します。

- 初期開発中の迅速なプロトタイピング。
- セキュリティ要件を最小限に抑えたシンプルな重要ではないツール。
- 独自の機能を活用するフレームワーク固有の機能。

3. 複雑なワークフローにメタツールを実装する – メタツールを追加してエージェントアーキテクチャを強化します。
   - 基本的なワークフローパターンから簡単に開始します。
   - ユースケースが成熟するにつれて複雑さを追加します。
   - エージェントとメタツール間のインターフェイスを標準化します。
4. 進化の計画 – 将来の柔軟性を念頭に置いて構築します。
   - 実装とは無関係にツールインターフェイスを文書化します。
   - エージェントとツールの間に抽象化レイヤーを作成します。
   - 独自のプロトコルからオープンプロトコルへの移行パスを確立します。

ツール統合のセキュリティのベストプラクティス

ツールの統合は、セキュリティ体制に直接影響します。このセクションでは、組織で考慮すべきベストプラクティスの概要を説明します。

認証と認可

次の堅牢なアクセスコントロールを使用します。

- OAuth 2.0/2.1 を使用する – リモートツールに業界標準の認証を実装します。
- 最小特権を実装する – ツールに必要なアクセス許可のみを付与します。
- 認証情報のローテーション – API キーとアクセストークンを定期的に更新します。

データ保護

データを保護するために、以下の対策を実施してください。

- 入力と出力を検証する – すべてのツールインタラクションにスキーマ検証を実装します。
- 機密データの暗号化 – すべてのリモートツール通信に TLS を使用します。
- データ最小化の実装 – 必要な情報のみをツールに渡します。

モニタリングと監査

次のメカニズムを使用して、可視性と制御を維持します。

- すべてのツール呼び出しを口グに記録する – 包括的な監査証跡を維持します。
- 異常をモニタリングする – 異常なツールの使用パターンを検出します。
- レート制限の実装 — 過剰なツール呼び出しによる不正使用を防止します。

Model Context Protocol (MCP) セキュリティモデルは、これらの懸念に包括的に対処します。詳細については、MCP ドキュメントの「セキュリティに関する考慮事項」を参照してください。

結論

エージェント AI の環境は急速に進化し続けており、組織はインテリジェントで自律的なシステムを構築するための強力な新しい方法を提供します。このガイドでは、実装を成功させるための 3 つの重要なコンポーネント、基盤を提供するフレームワーク、環境を提供するプラットフォーム、通信を可能にするプロトコル、機能を拡張するツールについて説明しました。

フレームワークが成熟するにつれて、相互運用性の向上、モデルコンテキストプロトコル (MCP) などのプロトコルの標準化、自律型エージェントのより高度なオーケストレーション機能が期待できます。現在、これらのフレームワークに関する専門知識を確立している組織は、ビジネスに大きな価値をもたらす、ますます自律的でインテリジェントなエージェントを構築する態勢が整っています。

プラットフォームは、エージェントシステムが動作する実行、ガバナンス、ライフサイクル環境を提供します。アイデンティティ、セキュリティ境界、オブザーバビリティ、メモリ管理、セッショングラウンディング、ツールやデータとの安全なやり取りなどの懸念に対処します。環境では AWS 、マネージドエージェントランタイムやオーケストレーションサービスなどのプラットフォームにより、組織は自律型エージェントとエージェントシステムを大規模にデプロイ、モニタリング、進化、管理できます。プラットフォームは、基本的なフレームワークを実際の運用要件と橋渡しします。

エージェントプロトコルの選択は、即時の開発ニーズと長期的な柔軟性と相互運用性のバランスを取る戦略的決定を表します。オープンプロトコルを優先し、適切な抽象化レイヤーを作成することで、組織は現在のビジネス要件を満たしながら、進化するテクノロジーに適応可能なエージェントシステムを構築できます。

ほとんどの組織にとって、MCP はオープンスタンダード、成長するエコシステム、agent-to-agent 通信パターンのサポート、ツール統合機能による強力な基盤です。 AWS は MCP と Agent2Agent (A2A) を戦略的プロトコルとして採用し、Strands AgentsSDK などのサービス全体での開発と実装に積極的に貢献しています。MCP または A2A を適切なフレームワークネイティブツールやメタツールとともに使用することで、将来のイノベーションに適応しながらすぐに価値をもたらすエージェントシステムを構築できます。

リソース

次のリソース AWS と、自律型エージェント開発に関連するその他のリソースを使用します。

AWS ブログ

- Amazon Bedrock AgentCore Memory: コンテキスト対応エージェントの構築
- Best practices for building robust generative AI applications with Amazon Bedrock Agents – Part 1
- Best practices for building robust generative AI applications with Amazon Bedrock Agents – Part 2
- LlamaIndexと Amazon Bedrock を使用して強力な RAG パイプラインを構築する
- Amazon Bedrock AgentCore Observability を使用して信頼できる AI エージェントを構築する
- Amazon Bedrock LlamaIndexと RAGAS を使用して RAG レスポンスを評価する
- Amazon Bedrock AgentCore コードインタープリタの紹介
- Amazon Bedrock AgentCore Gateway の紹介: エンタープライズ AI エージェントツール開発の変革
- Amazon Bedrock AgentCore Identity の紹介: エージェント AI を大規模に保護する
- オープンソース AI エージェント SDK Strands Agentsである の紹介
- エージェント相互運用性のオープンプロトコル パート 1: MCP でのエージェント間通信
- Amazon Bedrock AgentCore ランタイムでエージェントとツールを安全に起動およびスケーリングする
- AWS Transform for .NET、.NET アプリケーションを大規模にモダナイズするための最初のエージェント AI サービス
- AWS 毎週の切り上げ: Strands Agents

AWS 規範ガイダンス

- でのエージェント AI の運用 AWS
- でのエージェント AI の基礎 AWS
- でのエージェント AI のパターンとワークフロー AWS
- でのエージェント AI 用のサーバーレスアーキテクチャの構築 AWS
- でのエージェント AI 用のマルチテナントアーキテクチャの構築 AWS

- でのエージェント AI のセキュリティ AWS
- で拡張生成オプションとアーキテクチャを取得する AWS

AWS リソース

- Amazon Bedrock ドキュメント
- Amazon Bedrock AgentCore ドキュメント
- Amazon Bedrock AgentCore スターターツールキット (GitHub リポジトリ)
- Amazon Nova ドキュメント
- AWS MCP サーバー (GitHub リポジトリ)

その他のリソース

- AutoGen ドキュメント (Microsoft)
- 効果的なエージェントの構築 (Anthropic)
- CrewAI GitHub リポジトリ
- LangChainドキュメント
- LangGraph プラットフォーム
- LlamaIndexドキュメント
- Model Context Protocol ドキュメント
- Strands Agentsドキュメント
- Strands Agents ツールの概要
- Strands Agents クイックスタートガイド

ドキュメント履歴

以下の表は、本ガイドの重要な変更点について説明したものです。今後の更新に関する通知を受け取る場合は、RSS フィードをサブスクライブできます。

変更	説明	日付
新規セクション	プラットフォームセクションを追加	2026 年 1 月 16 日
初版発行	—	2025 年 7 月 14 日

AWS 規範ガイダンスの用語集

以下は、 AWS 規範ガイダンスによって提供される戦略、ガイド、パターンで一般的に使用される用語です。エントリを提案するには、用語集の最後のフィードバックの提供リンクを使用します。

数字

7 Rs

アプリケーションをクラウドに移行するための 7 つの一般的な移行戦略。これらの戦略は、ガートナーが 2011 年に特定した 5 Rs に基づいて構築され、以下で構成されています。

- リファクタリング/アーキテクチャの再設計 — クラウドネイティブ特徴を最大限に活用して、俊敏性、パフォーマンス、スケーラビリティを向上させ、アプリケーションを移動させ、アーキテクチャを変更します。これには、通常、オペレーティングシステムとデータベースの移植が含まれます。例: オンプレミスの Oracle データベースを Amazon Aurora PostgreSQL 互換エディションに移行する。
- リプラットフォーム (リフトアンドリシェイプ) — アプリケーションをクラウドに移行し、クラウド機能を活用するための最適化レベルを導入します。例: お客様のオンプレミスの Oracle データベースを AWS クラウドの Oracle 用の Amazon Relational Database Service (Amazon RDS) に移行する。
- 再購入 (ドロップアンドショップ) — 通常、従来のライセンスから SaaS モデルに移行して、別の製品に切り替えます。例: 顧客関係管理 (CRM) システムを Salesforce.com に移行する。
- リホスト (リフトアンドシフト) — クラウド機能を活用するための変更を加えずに、アプリケーションをクラウドに移行します。例: お客様のオンプレミスの Oracle データベースを AWS クラウドの EC2 インスタンス上の Oracle に移行する。
- 再配置 (ハイパーバイザーレベルのリフトアンドシフト) – 新しいハードウェアを購入したり、アプリケーションを書き換えたり、既存の運用を変更したりすることなく、インフラストラクチャをクラウドに移行できます。オンプレミスプラットフォームから同じプラットフォームのクラウドサービスにサーバーを移行します。例： Microsoft Hyper-Vアプリケーションを に移行します AWS。
- 保持 (再アクセス) — アプリケーションをお客様のソース環境で保持します。これには、主要なリファクタリングを必要とするアプリケーションや、お客様がその作業を後日まで延期したいアプリケーション、およびそれらを移行するためのビジネス上の正当性がないため、お客様が保持するレガシーアプリケーションなどがあります。
- 廃止 — お客様のソース環境で不要になったアプリケーションを停止または削除します。

A

ABAC

「属性ベースのアクセス制御」をご覧ください。

抽象化されたサービス

「マネージドユーザー」をご覧ください。

ACID

「原子性、一貫性、分離性、耐久性 (ACID)」をご覧ください。

アクティブ/アクティブ移行

(双方向レプリケーションツールまたは二重書き込み操作を使用して) ソースデータベースとターゲットデータベースを同期させ、移行中に両方のデータベースが接続アプリケーションからのトランザクションを処理するデータベース移行方法。この方法では、1 回限りのカットオーバーの必要がなく、管理された小規模なバッチで移行できます。アクティブ/パッシブ移行よりも柔軟な方法ですが、さらに多くの作業が必要となります。

アクティブ/パッシブ移行

ソースデータベースとターゲットデータベースを同期させながら、データがターゲットデータベースにレプリケートされている間、接続しているアプリケーションからのトランザクションをソースデータベースのみで処理するデータベース移行方法。移行中、ターゲットデータベースはトランザクションを受け付けません。

集計関数

複数行に処理を行い、グループ全体を対象に単一の戻り値を計算する SQL 関数。集計関数の例としては、SUM や MAX などがあります。

AI

「人工知能」をご覧ください。

AIOps

「AI オペレーション」をご覧ください。

匿名化

データセット内の個人情報を完全に削除するプロセス。匿名化は個人のプライバシー保護に役立ちます。匿名化されたデータは、もはや個人データとは見なされません。

アンチパターン

繰り返し起こる問題に対して頻繁に用いられる解決策で、その解決策が逆効果であったり、効果がなかったり、代替案よりも効果が低かったりするもの。

アプリケーション制御

マルウェアからシステムを保護するために、承認されたアプリケーションのみを使用できるようにするセキュリティアプローチ。

アプリケーションポートフォリオ

アプリケーションの構築と維持にかかるコスト、およびそのビジネス価値を含む、組織が使用する各アプリケーションに関する詳細情報の集まり。この情報は、ポートフォリオの検出と分析プロセス の重要な要素であり、移行、モダナイズ、最適化するアプリケーションを特定し、優先順位を付けるのに役立ちます。

人工知能 (AI)

コンピューティングテクノロジーを使用し、学習、問題の解決、パターンの認識など、通常は人間に関連づけられる認知機能の実行に特化したコンピュータサイエンスの分野。詳細については、「人工知能 (AI) とは何ですか?」をご覧ください。

AI オペレーション (AIOps)

機械学習技術を使用して運用上の問題を解決し、運用上のインシデントと人の介入を減らし、サービス品質を向上させるプロセス。 AWS 移行戦略での AIOps の使用方法については、オペレーション統合ガイド を参照してください。

非対称暗号化

暗号化用のパブリックキーと復号用のプライベートキーから成る 1 組のキーを使用した、暗号化のアルゴリズム。パブリックキーは復号には使用されないため共有しても問題ありませんが、プライベートキーの利用は厳しく制限する必要があります。

原子性、一貫性、分離性、耐久性 (ACID)

エラー、停電、その他の問題が発生した場合でも、データベースのデータ有効性と運用上の信頼性を保証する一連のソフトウェアプロパティ。

属性ベースのアクセス制御 (ABAC)

部署、役職、チーム名など、ユーザーの属性に基づいてアクセス許可をきめ細かく設定する方法。詳細については、 AWS Identity and Access Management (IAM) ドキュメントの「 の ABAC AWS」を参照してください。

信頼できるデータソース

最も信頼性のある情報源とされるデータのプライマリーバージョンを保存する場所。匿名化、編集、仮名化など、データを処理または変更する目的で、信頼できるデータソースから他の場所にデータをコピーすることができます。

アベイラビリティーゾーン (AZ)

他のアベイラビリティーゾーンの障害から AWS リージョン 隔離され、同じリージョン内の他のアベイラビリティーゾーンへの低コストで低レイテンシーのネットワーク接続を提供する 内の別の場所。

AWS クラウド導入フレームワーク (AWS CAF)

組織がクラウドへの移行を成功させるための効率的で効果的な計画を立て AWS るための、 のガイドラインとベストプラクティスのフレームワークです。 AWS CAF は、ビジネス、人材、ガバナンス、プラットフォーム、セキュリティ、運用という 6 つの重点分野にガイダンスを整理しています。ビジネス、人材、ガバナンスの観点では、ビジネススキルとプロセスに重点を置き、プラットフォーム、セキュリティ、オペレーションの視点は技術的なスキルとプロセスに焦点を当てています。例えば、人材の観点では、人事 (HR)、人材派遣機能、および人材管理を扱うステークホルダーを対象としています。この観点から、 AWS CAF は、クラウド導入を成功させるための準備に役立つ人材開発、トレーニング、コミュニケーションに関するガイダンスを提供します。詳細については、AWS CAF ウェブサイトと AWS CAF のホワイトペーパー を参照してください。

AWS ワークロード認定フレームワーク (AWS WQF)

データベース移行ワークロードを評価し、移行戦略を推奨し、作業見積もりを提供するツール。AWS WQF は AWS Schema Conversion Tool (AWS SCT) に含まれています。データベーススキーマとコードオブジェクト、アプリケーションコード、依存関係、およびパフォーマンス特性を分析し、評価レポートを提供します。

B

不正なボット

個人や組織に混乱や損害を与えることを目的としたボット。

BCP

「ビジネス継続性計画 (BCP)」をご覧ください。

動作グラフ

リソースの動作とインタラクションを経時的に示した、一元的なインタラクティブビュー。Amazon Detective の動作グラフを使用すると、失敗したログオンの試行、不審な API 呼び出し、その他同様のアクションを調べることができます。詳細については、Detective ドキュメントの「動作グラフのデータ」を参照してください。

ビッグエンディアンシステム

最上位バイトを最初に格納するシステム。「エンディアン性」もご覧ください。

二項分類

バイナリ結果 (2 つの可能なクラスのうちの 1 つ) を予測するプロセス。例えば、お客様の機械学習モデルで「この E メールはスパムですか、それともスパムではありませんか」などの問題を予測する必要があるかもしれません。または「この製品は書籍ですか、車ですか」などの問題を予測する必要があるかもしれません。

ブルームフィルター

要素がセットのメンバーであるかどうかをテストするために使用される、確率的でメモリ効率の高いデータ構造。

ブルー/グリーンデプロイ

それぞれが独立しているが、同一の環境を 2 つ作成するデプロイ戦略。現在のアプリケーションバージョンを 1 つの環境 (ブルー) で実行し、新しいアプリケーションバージョンを別の環境 (グリーン) で実行します。この戦略は、最小限の影響で迅速にロールバックするのに役立ちます。

ボット

インターネット経由で自動タスクを実行し、人間のアクティビティややり取りをシミュレートするソフトウェアアプリケーション。インターネット上の情報のインデックスを作成するウェブクローラーなど、一部のボットは有用または有益です。悪質なボットと呼ばれる他のボットの中には、個人や組織を混乱させたり、損害を与えたりすることを意図したものもあります。

ボットネット

マルウェアに感染しており、ボットハーダーまたはボットオペレーターと呼ばれる単一の当事者によって制御されているボットのネットワーク。ボットネットは、ボットとその影響力を拡大する仕組みとして、非常によく知られています。

ブランチ

コードリポジトリに含まれる領域。リポジトリに最初に作成するブランチは、メインブランチといいます。既存のブランチから新しいブランチを作成し、その新しいブランチで機能を開発した

り、バグを修正したりできます。機能を構築するために作成するブランチは、通常、機能ブランチと呼ばれます。機能をリリースする準備ができたら、機能ブランチをメインブランチに統合します。詳細については、「ブランチの概要」(GitHub ドキュメント) を参照してください。

ブレークグラスアクセス

例外的な状況では、承認されたプロセスを通じて、ユーザーが AWS アカウント 通常アクセス許可を持たない にすばやくアクセスできるようにします。詳細については、 AWS Well-Architected ガイダンスの「ブレークグラス手順の実装」インジケータを参照してください。

ブラウンフィールド戦略

環境の既存インフラストラクチャ。システムアーキテクチャにブラウンフィールド戦略を導入する場合、現在のシステムとインフラストラクチャの制約に基づいてアーキテクチャを設計します。既存のインフラストラクチャを拡張している場合は、ブラウンフィールド戦略とグリーンフィールド戦略を融合させることもできます。

バッファキャッシュ

アクセス頻度が最も高いデータが保存されるメモリ領域。

ビジネス能力

価値を生み出すためにビジネスが行うこと (営業、カスタマーサービス、マーケティングなど)。マイクロサービスのアーキテクチャと開発の決定は、ビジネス能力によって推進できます。詳細については、AWSでのコンテナ化されたマイクロサービスの実行ホワイトペーパーの「ビジネス機能を中心に組織化」セクションを参照してください。

ビジネス継続性計画 (BCP)

大規模移行など、中断を伴うイベントが運用に与える潜在的な影響に対処し、ビジネスを迅速に再開できるようにする計画。

C

CAF

「AWS クラウド導入フレームワーク」を参照してください

カナリアデプロイ

エンドユーザーへのバージョンリリースを、時間をかけて段階的に行うこと。確信が持てたら新規バージョンをデプロイして、現在のバージョン全体を置き換えます。

CCoE

「Cloud Center of Excellence」を参照してください。

CDC

「変更データキャプチャ」を参照してください。

変更データキャプチャ (CDC)

データソース (データベーステーブルなど) の変更を追跡し、その変更に関するメタデータを記録するプロセス。CDC は、ターゲットシステムでの変更を監査またはレプリケートして同期を維持するなど、さまざまな目的に使用できます。

カオスエンジニアリング

障害や破壊的なイベントを意図的に導入して、システムの耐障害性をテストすること。AWS Fault Injection Service (AWS FIS) を使用して、 AWS ワークロードにストレスを与え、その応答を評価する実験を実行できます。

CI/CD

「継続的インテグレーションと継続的デリバリー」を参照してください。

分類

予測を生成するのに役立つ分類プロセス。分類問題の機械学習モデルは、離散値を予測します。離散値は、常に互いに区別されます。例えば、モデルがイメージ内に車があるかどうかを評価する必要がある場合があります。

クライアント側の暗号化

ターゲットがデータ AWS のサービス を受信する前のローカルでのデータの暗号化。

Cloud Center of Excellence (CCoE)

クラウドのベストプラクティスの作成、リソースの移動、移行のタイムラインの確立、大規模変革を通じて組織をリードするなど、組織全体のクラウド導入の取り組みを推進する学際的なチーム。詳細については、 AWS クラウド エンタープライズ戦略ブログの CCoE 投稿を参照してください。

クラウドコンピューティング

リモートデータストレージと IoT デバイス管理に通常使用されるクラウドテクノロジー。クラウドコンピューティングは、一般的に、エッジコンピューティングに接続されています。

クラウド運用モデル

IT 組織において、1 つ以上のクラウド環境を構築、成熟、最適化するために使用される運用モデル。詳細については、「クラウド運用モデルの構築」を参照してください。

導入のクラウドステージ

組織が、 AWS クラウドへの移行時に通常実行する 4 つの段階。

- プロジェクト — 概念実証と学習を目的として、クラウド関連のプロジェクトをいくつか実行する
- 基礎固め — お客様のクラウドの導入を拡大するための基礎的な投資 (ランディングゾーンの作成、CCoE の定義、運用モデルの確立など)
- 移行 — 個々のアプリケーションの移行
- 再発明 — 製品とサービスの最適化、クラウドでのイノベーション

これらのステージは、 AWS クラウド エンタープライズ戦略ブログのブログ記事「クラウドファーストへのジャーニー」と「導入のステージ」で Stephen Orban によって定義されました。移行戦略との関連性については、 AWS「移行準備ガイド」を参照してください。

CMDB

「構成管理データベース (CMDB)」を参照してください。

コードリポジトリ

ソースコードやその他の資産 (ドキュメント、サンプル、スクリプトなど) が保存され、バージョン管理プロセスを通じて更新される場所。一般的なクラウドリポジトリには、GitHub や Bitbucket Cloud があります。コードの各バージョンはブランチと呼ばれます。マイクロサービスの構造では、各リポジトリは 1 つの機能専用です。1 つの CI/CD パイプラインで複数のリポジトリを使用できます。

コールドキャッシュ

空である、または、かなり空きがある、もしくは、古いデータや無関係なデータが含まれているバッファキャッシュ。データベースインスタンスはメインメモリまたはディスクから読み取る必要があり、バッファキャッシュから読み取るよりも時間がかかるため、パフォーマンスに影響します。

コールドデータ

めったにアクセスされず、通常は過去のデータです。この種類のデータをクエリする場合、通常は低速なクエリでも問題ありません。このデータを低パフォーマンスで安価なストレージ階層またはクラスに移動すると、コストを削減することができます。

コンピュータビジョン (CV)

機械学習を使用してデジタルイメージやビデオといった、ビジュアル形式の情報を分析および抽出する AI の分野。例えば、Amazon SageMaker AI では、CV 用の画像処理アルゴリズムを利用できます。

設定ドリフト

ワークロードにおいて、設定が想定した状態から変化すること。これによって、ワークロードが非準拠になる可能性がありますが、この状態は、徐々に生じ、意図的なものではありません。

構成管理データベース (CMDB)

データベースとその IT 環境 (ハードウェアとソフトウェアの両方のコンポーネントとその設定を含む) に関する情報を保存、管理するリポジトリ。通常、CMDB のデータは、移行のポートフォリオの検出と分析の段階で使用します。

コンフォーマンスパック

コンプライアンスチェックとセキュリティチェックをカスタマイズするためにアセンブルできる AWS Config ルールと修復アクションのコレクション。YAML テンプレートを使用して、コンフォーマンスパックを AWS アカウント および リージョンの単一のエンティティとしてデプロイすることも、組織全体にデプロイすることもできます。詳細については、 AWS Config ドキュメントの「コンフォーマンスパック」を参照してください。

継続的インテグレーションと継続的デリバリー (CI/CD)

ソフトウェアリリースプロセスのソース、ビルド、テスト、ステージング、本番の各ステージを自動化するプロセス。CI/CD は一般的にパイプラインと呼ばれます。プロセスの自動化、生産性の向上、コード品質の向上、配信の加速化を可能にします。詳細については、「継続的デリバリーの利点」を参照してください。CD は継続的デプロイ (Continuous Deployment) の略語でもあります。詳細については「継続的デリバリーと継続的なデプロイ」を参照してください。

CV

「コンピュータビジョン」を参照してください。

D

保管中のデータ

ストレージ内にあるデータなど、常に自社のネットワーク内にあるデータ。

データ分類

ネットワーク内のデータを重要度と機密性に基づいて識別、分類するプロセス。データに適した保護および保持のコントロールを判断する際に役立つため、あらゆるサイバーセキュリティのリスク管理戦略において重要な要素です。データ分類は、 AWS Well-Architected フレームワークのセキュリティの柱のコンポーネントです。詳細については、「データ分類」を参照してください。

データドリフト

実稼働データと ML モデルのトレーニングに使用されたデータとの間に有意な差異が生じたり、入力データが時間の経過と共に有意に変化したりすることです。データドリフトは、ML モデル予測の全体的な品質、精度、公平性を低下させる可能性があります。

転送中のデータ

ネットワーク内 (ネットワークリソース間など) を活発に移動するデータ。

データメッシュ

非一元的で分散型のデータ所有権を持つとともに、一元的な管理およびガバナンスを行えるアーキテクチャフレームワーク。

データ最小化

厳密に必要なデータのみを収集し、処理するという原則。でデータ最小化を実践 AWS クラウドすることで、プライバシーリスク、コスト、分析のカーボンフットプリントを削減できます。

データ境界

AWS 環境内の一連の予防ガードレール。信頼された ID のみが、期待されるネットワークから信頼されたリソースにアクセスできるようにします。詳細については、「 でのデータ境界の構築 AWS」を参照してください。

データの前処理

raw データをお客様の機械学習モデルで簡単に解析できる形式に変換すること。データの前処理とは、特定の列または行を削除して、欠落している、矛盾している、または重複する値に対処することを意味します。

データ出所

データの生成、送信、保存の方法など、データのライフサイクル全体を通じてデータの出所と履歴を追跡するプロセス。

データ件名

データを収集、処理している個人。

データウェアハウス

分析などのビジネスインテリジェンスをサポートするデータ管理システム。データウェアハウスには、一般的に、大量の履歴データが含まれており、多くの場合、それらはクエリや分析に使用されます。

データベース定義言語 (DDL)

データベース内のテーブルやオブジェクトの構造を作成または変更するためのステートメントまたはコマンド。

データベース操作言語 (DML)

データベース内の情報を変更 (挿入、更新、削除) するためのステートメントまたはコマンド。

DDL

「データベース定義言語」を参照してください。

ディープアンサンブル

予測のために複数の深層学習モデルを組み合わせます。ディープアンサンブルを使用して、より正確な予測を取得したり、予測の不確実性を推定したりできます。

深層学習

人工ニューラルネットワークの複数層を使用して、入力データと対象のターゲット変数の間のマッピングを識別する機械学習サブフィールド。

多層防御

一連のセキュリティメカニズムとコントロールをコンピュータネットワーク全体に層状に重ねて、ネットワークとその内部にあるデータの機密性、整合性、可用性を保護する情報セキュリティの手法。この戦略を採用するときは AWS、 AWS Organizations 構造の異なるレイヤーに複数のコントロールを追加して、リソースの安全性を確保します。たとえば、多層防御アプローチでは、多要素認証、ネットワークセグメンテーション、暗号化を組み合わせることができます。

委任管理者

では AWS Organizations、互換性のあるサービスが AWS メンバーアカウントを登録して組織のアカウントを管理し、そのサービスのアクセス許可を管理できます。このアカウントを、そのサービスの委任管理者と呼びます。詳細、および互換性のあるサービスの一覧は、 AWS

Organizations ドキュメントの「AWS Organizationsで使用できるサービス」を参照してください。

トラブルシューティング

アプリケーション、新機能、コードの修正をターゲットの環境で利用できるようにするプロセス。デプロイでは、コードベースに変更を施した後、アプリケーションの環境でそのコードベースを構築して実行します。

開発環境

「環境」を参照してください。

検出管理

イベントが発生したときに、検出、ログ記録、警告を行うように設計されたセキュリティコントロール。これらのコントロールは副次的な防衛手段であり、実行中の予防的コントロールをすり抜けたセキュリティイベントをユーザーに警告します。詳細については、「 AWSでのセキュリティコントロールの実装」の「検出的コントロール」を参照してください。

開発バリューストリームマッピング (DVSM)

ソフトウェア開発ライフサイクルのスピードと品質に悪影響を及ぼす制約を特定し、優先順位を付けるために使用されるプロセス。DVSM は、もともとリーンマニュファクチャリング・プラクティスのために設計されたバリューストリームマッピング・プロセスを拡張したものです。ソフトウェア開発プロセスを通じて価値を創造し、動かすために必要なステップとチームに焦点を当てています。

デジタルツイン

建物、工場、産業機器、生産ラインなど、現実世界のシステムを仮想的に表現したものです。デジタルツインは、予知保全、リモートモニタリング、生産最適化をサポートします。

ディメンションテーブル

スタースキーマにおいて、ファクトテーブルの定量データに関するデータ属性が含まれる小さいテーブル。ディメンションテーブルの属性は、通常、テキストフィールド、またはテキストのように扱える個別の数値で示されます。これらの属性は、一般的に、クエリの制約、フィルタリング、結果セットのラベル付けに使用されます。

ディザスタ

ワークロードまたはシステムが、導入されている主要な場所でのビジネス目標の達成を妨げるイベント。これらのイベントは、自然災害、技術的障害、または意図しない設定ミスやマルウェア攻撃などの人間の行動の結果である場合があります。

ディザスタリカバリ (DR)

ディザスタによるダウンタイムとデータ損失を最小限に抑えるための戦略とプロセス。詳細については、 AWS Well-Architected フレームワークの「 でのワークロードのディザスタリカバリ AWS: クラウドでのリカバリ」を参照してください。

DML

「データベース操作言語」を参照してください。

ドメイン駆動型設計

各コンポーネントが提供している変化を続けるドメイン、またはコアビジネス目標にコンポーネントを接続して、複雑なソフトウェアシステムを開発するアプローチ。この概念は、エリック・エヴァンスの著書、Domain-Driven Design: Tackling Complexity in the Heart of Software (ドメイン駆動設計:ソフトウェアの中心における複雑さへの取り組み) で紹介されています (ボストン: Addison-Wesley Professional、2003)。strangler fig パターンでドメイン駆動型設計を使用する方法の詳細については、「コンテナと Amazon API Gateway を使用して、従来の Microsoft ASP.NET (ASMX) ウェブサービスを段階的にモダナイズ」を参照してください。

DR

「ディザスタリカバリ」を参照してください。

ドリフト検出

ベースライン設定からの偏差を追跡します。例えば、 AWS CloudFormation を使用してシステムリソースのドリフトを検出したり、 を使用して AWS Control Tower 、ガバナンス要件への準拠に影響するランディングゾーンの変更を検出したりできます。

DVSM

「開発バリューストリームマッピング」を参照してください。

E

EDA

「探索的データ分析」を参照してください。

EDI

「電子データ交換」を参照してください。

エッジコンピューティング

IoT ネットワークのエッジにあるスマートデバイスの計算能力を高めるテクノロジー。クラウドコンピューティングと比較すると、エッジコンピューティングは通信レイテンシーを短縮し、応答時間を改善できます。

電子データ交換 (EDI)

組織間で行う、ビジネスドキュメントの自動交換。詳細については、「電子データ交換とは」を参照してください。

暗号化

人間が読み取り可能なプレーンテキストデータを暗号文に変換するコンピューティング処理。

暗号化キー

暗号化アルゴリズムが生成した、ランダム化されたビットからなる暗号文字列。キーの長さは決まっておらず、各キーは予測できないように、一意になるように設計されています。

エンディアン

コンピュータメモリにバイトが格納される順序。ビッグエンディアンシステムでは、最上位バイトが最初に格納されます。リトルエンディアンシステムでは、最下位バイトが最初に格納されます。

エンドポイント

「サービスエンドポイント」を参照してください。

エンドポイントサービス

仮想プライベートクラウド (VPC) 内でホストして、他のユーザーと共有できるサービス。を使用してエンドポイントサービスを作成し AWS PrivateLink 、他の AWS アカウント または AWS Identity and Access Management (IAM) プリンシパルにアクセス許可を付与できます。これらのアカウントまたはプリンシパルは、インターフェイス VPC エンドポイントを作成することで、エンドポイントサービスにプライベートに接続できます。詳細については、Amazon Virtual Private Cloud (Amazon VPC) ドキュメントの「エンドポイントサービスを作成する」を参照してください。

エンタープライズリソースプランニング (ERP)

エンタープライズの主要なビジネスプロセス (会計、MES、プロジェクト管理など) を自動化および管理するシステム。

エンベロープ暗号化

暗号化キーを、別の暗号化キーを使用して暗号化するプロセス。詳細については、 AWS Key Management Service (AWS KMS) ドキュメントの「エンベロープ暗号化」を参照してください。

環境

実行中のアプリケーションのインスタンス。クラウドコンピューティングにおける一般的な環境の種類は以下のとおりです。

- 開発環境 — アプリケーションのメンテナンスを担当するコアチームのみが使用できる、実行中のアプリケーションのインスタンス。開発環境は、上位の環境に昇格させる変更をテストするときに使用します。このタイプの環境は、テスト環境と呼ばれることもあります。
- 下位環境 — 初期ビルドやテストに使用される環境など、アプリケーションのすべての開発環境。
- 本番環境 — エンドユーザーがアクセスできる、実行中のアプリケーションのインスタンス。CI/CD パイプラインでは、本番環境が最後のデプロイ環境になります。
- 上位環境 — コア開発チーム以外のユーザーがアクセスできるすべての環境。これには、本番環境、本番前環境、ユーザー承認テスト環境などが含まれます。

エピック

アジャイル方法論で、お客様の作業の整理と優先順位付けに役立つ機能カテゴリ。エピックでは、要件と実装タスクの概要についてハイレベルな説明を提供します。例えば、 AWS CAF セキュリティエピックには、ID とアクセスの管理、検出コントロール、インフラストラクチャセキュリティ、データ保護、インシデント対応が含まれます。 AWS 移行戦略のエピックの詳細については、プログラム実装ガイドを参照してください。

ERP

「エンタープライズリソース計画」を参照してください。

探索的データ分析 (EDA)

データセットを分析してその主な特性を理解するプロセス。お客様は、データを収集または集計してから、パターンの検出、異常の検出、および前提条件のチェックのための初期調査を実行します。EDA は、統計の概要を計算し、データの可視化を作成することによって実行されます。

F

ファクトテーブル

スタースキーマの中央にあるテーブル。ビジネスオペレーションに関する定量的データが保存されます。一般的に、ファクトテーブルは、2 種類の列で構成されます。1 つは測定値が含まれる列、もう 1 つはディメンションテーブルへの外部キーが含まれる列です。

フェイルファスト

開発ライフサイクルを短縮するために、頻繁かつ段階的にテストを行う哲学であり、アジャイルアプローチでは、この考え方がきわめて重要です。

障害分離境界

では AWS クラウド、障害の影響を制限し、ワークロードの耐障害性を高めるのに役立つアベイラビリティーゾーン AWS リージョン、コントロールプレーン、データプレーンなどの境界。詳細については、「AWS 障害分離境界」を参照してください。

機能ブランチ

「ブランチ」を参照してください。

特徴量

お客様が予測に使用する入力データ。例えば、製造コンテキストでは、特徴量は製造ラインから定期的にキャプチャされるイメージの可能性もあります。

特徴量重要度

モデルの予測に対する特徴量の重要性。これは通常、Shapley Additive Deskonations (SHAP) や積分勾配など、さまざまな手法で計算できる数値スコアで表されます。詳細については、「 を使用した機械学習モデルの解釈可能性 AWS」を参照してください。

機能変換

追加のソースによるデータのエンリッチ化、値のスケーリング、単一のデータフィールドからの複数の情報セットの抽出など、機械学習プロセスのデータを最適化すること。これにより、機械学習モデルはデータの恩恵を受けることができます。例えば、「2021-05-27 00:15:37」の日付を「2021 年」、「5 月」、「木」、「15」に分解すると、学習アルゴリズムがさまざまなデータコンポーネントに関連する微妙に異なるパターンを学習するのに役立ちます。

数ショットプロンプト

LLM に、タスクと望ましい出力を示す例を少数提示した後に、類似のタスクを実行させること。この手法は、プロンプトに記述された例 (ショット) からモデルが学習する「インコンテキスト学

習」の一種です。数ショットプロンプトは、特定のフォーマット、推論、専門知識が必要なタスクに効果的です。「ゼロショットプロンプト」も参照してください。

FGAC

「きめ細かなアクセス制御」を参照してください。

きめ細かなアクセス制御 (FGAC)

複数の条件を使用してアクセス要求を許可または拒否すること。

フラッシュカット移行

変更データのキャプチャによる継続的なデータ複製を利用して、段階的なアプローチではなく、可能な限り短時間でデータを移行するデータベース移行方法。目的はダウンタイムを最小限に抑えることです。

FM

「基盤モデル」を参照してください。

基盤モデル (FM)

大規模な深層学習ニューラルネットワークであり、一般化およびラベル付けされていないデータからなる大規模データセットでトレーニングされています。FM により、言語理解、テキストおよび画像生成、自然言語での会話といった、一般的な各種タスクを実行できます。詳細については、「基盤モデルとは何ですか?」を参照してください。

G

生成 AI

AI モデルのサブセット。大量のデータでトレーニングされており、シンプルなテキストプロンプトを使用して、画像、動画、テキスト、オーディオなどの新しいコンテンツやアーティファクトを作成できます。詳細については、「生成 AI とは何ですか?」を参照してください。

ジオブロッキング

「地理的制限」を参照してください。

地理的制限 (ジオブロッキング)

特定の国のユーザーがコンテンツ配信にアクセスできないようにするための、Amazon CloudFront のオプション。アクセスを許可する国と禁止する国は、許可リストまたは禁止リスト

を使って指定します。詳細については、CloudFront ドキュメントの「コンテンツの地理的ディストリビューションの制限」を参照してください。

Gitflow ワークフロー

下位環境と上位環境が、ソースコードリポジトリでそれぞれ異なるブランチを使用する方法。Gitflow ワークフローは古いと見なされている方法であり、トランクベースのワークフローは推奨されている新しい方法です。

ゴールデンイメージ

システムまたはソフトウェアのスナップショットであり、システムまたはソフトウェアの新規インスタンスをデプロイするテンプレートとして使用されます。製造の例で言えば、ゴールデンイメージを使用すると、複数のデバイスにソフトウェアをプロビジョニングして、デバイス製造オペレーションの速度、スケーラビリティ、生産性を向上させることができます。

グリーンフィールド戦略

新しい環境に既存のインフラストラクチャが存在しないこと。システムアーキテクチャにグリーンフィールド戦略を導入する場合、既存のインフラストラクチャ (別名ブラウンフィールド) との互換性の制約を受けることなく、あらゆる新しいテクノロジーを選択できます。既存のインフラストラクチャを拡張している場合は、ブラウンフィールド戦略とグリーンフィールド戦略を融合させることもできます。

ガードレール

組織単位 (OU) 全般のリソース、ポリシー、コンプライアンスを管理するのに役立つ概略的なルール。予防ガードレールは、コンプライアンス基準に一致するようにポリシーを実施します。これらは、サービスコントロールポリシーと IAM アクセス許可の境界を使用して実装されます。検出ガードレールは、ポリシー違反やコンプライアンス上の問題を検出し、修復のためのアラートを発信します。これらは AWS Config、、Amazon GuardDuty AWS Security Hub CSPM、、 AWS Trusted Advisor Amazon Inspector、およびカスタム AWS Lambda チェックを使用して実装されます。

H

HA

「高可用性」を参照してください。

異種混在データベースの移行

別のデータベースエンジンを使用するターゲットデータベースへお客様の出典データベースの移行 (例えば、Oracle から Amazon Aurora)。異種間移行は通常、アーキテクチャの再設計作業の一部であり、スキーマの変換は複雑なタスクになる可能性があります。AWS は、スキーマの変換に役立つ AWS SCTを提供します。

高可用性 (HA)

課題や災害が発生した場合に、介入なしにワークロードを継続的に運用できること。HA システムは、自動的にフェイルオーバーし、一貫して高品質のパフォーマンスを提供し、パフォーマンスへの影響を最小限に抑えながらさまざまな負荷や障害を処理するように設計されています。

ヒストリアンのモダナイゼーション

製造業のニーズによりよく応えるために、オペレーショントロジー (OT) システムをモダナイズし、アップグレードするためのアプローチ。ヒストリアンは、工場内のさまざまなソースからデータを収集して保存するために使用されるデータベースの一種です。

ホールドアウトデータ

機械学習モデルのトレーニング用データセットから保留される、ラベル付き履歴データの一部。ホールドアウトデータを使用すると、モデル予測をホールドアウトデータと比較して、モデルのパフォーマンスを評価できます。

同種データベースの移行

お客様の出典データベースを、同じデータベースエンジンを共有するターゲットデータベース (Microsoft SQL Server から Amazon RDS for SQL Server など) に移行する。同種間移行は、通常、リホストまたはリプラットフォーム化の作業の一部です。ネイティブデータベースユーティリティを使用して、スキーマを移行できます。

ホットデータ

リアルタイムデータや最近の翻訳データなど、頻繁にアクセスされるデータ。通常、このデータには高速なクエリ応答を提供する高性能なストレージ階層またはクラスが必要です。

ホットフィックス

本番環境の重大な問題を修正するために緊急で配布されるプログラム。緊急性が高いため、通常の DevOps のリリースワークフローからは外れた形で実施されます。

ハイパーケア期間

カットオーバー直後、移行したアプリケーションを移行チームがクラウドで管理、監視して問題に対処する期間。通常、この期間は 1～4 日です。ハイパーケア期間が終了すると、アプリケーションに対する責任は一般的に移行チームからクラウドオペレーションチームに移ります。

I

IaC

「Infrastructure as Code」を参照してください。

ID ベースのポリシー

AWS クラウド 環境内のアクセス許可を定義する 1 つ以上の IAM プリンシパルにアタッチされたポリシー。

アイドル状態のアプリケーション

90 日間の平均的な CPU およびメモリ使用率が 5～20% のアプリケーション。移行プロジェクトでは、これらのアプリケーションを廃止するか、オンプレミスに保持するのが一般的です。

IIoT

「インダストリアル IoT」を参照してください。

イミュータブルインフラストラクチャ

既存インフラストラクチャの更新、パッチ適用、変更などを行わずに、本番環境ワークロードに使用する新規インフラストラクチャをデプロイするモデル。本質的に、イミュータブルインフラストラクチャは、ミュータブルインフラストラクチャよりも一貫性、信頼性、予測性に優れています。詳細については、 AWS Well-Architected フレームワークにある「イミュータブルインフラストラクチャを使用してデプロイする」のベストプラクティスを参照してください。

インバウンド (受信) VPC

AWS マルチアカウントアーキテクチャでは、アプリケーションの外部からネットワーク接続を受け入れ、検査し、ルーティングする VPC。AWS Security Reference Architecture では、アプリケーションとより広範なインターネット間の双方向のインターフェイスを保護するために、インバウンド、アウトバウンド、インスペクションの各 VPC を使用してネットワークアカウントを設定することを推奨しています。

増分移行

アプリケーションを 1 回ですべてカットオーバーするのではなく、小さい要素に分けて移行するカットオーバー戦略。例えば、最初は少数のマイクロサービスまたはユーザーのみを新しいシステムに移行する場合があります。すべてが正常に機能することを確認できたら、残りのマイクロサービスやユーザーを段階的に移行し、レガシーシステムを廃止できるようにします。この戦略により、大規模な移行に伴うリスクが軽減されます。

インダストリー 4.0

2016 年に Klaus Schwab 氏が提唱した用語で、接続、リアルタイムデータ、オートメーション、分析、AI/ML の進歩による、ビジネスプロセスのモダナイズを意味します。

インフラストラクチャ

アプリケーションの環境に含まれるすべてのリソースとアセット。

Infrastructure as Code (IaC)

アプリケーションのインフラストラクチャを一連の設定ファイルを使用してプロビジョニングし、管理するプロセス。IaC は、新しい環境を再現可能で信頼性が高く、一貫性のあるものにするため、インフラストラクチャを一元的に管理し、リソースを標準化し、スケールを迅速に行えるように設計されています。

インダストリアル IoT (IIoT)

製造、エネルギー、自動車、ヘルスケア、ライフサイエンス、農業などの産業部門におけるインターネットに接続されたセンサーやデバイスの使用。詳細については、「インダストリアル IoT (IIoT) デジタルトランスフォーメーション戦略の構築」を参照してください。

インスペクション VPC

AWS マルチアカウントアーキテクチャでは、VPC (同一または異なる 内 AWS リージョン)、インターネット、オンプレミスネットワーク間のネットワークトラフィックの検査を管理する一元化された VPCs。AWS Security Reference Architecture では、アプリケーションとより広範なインターネット間の双方向のインターフェイスを保護するために、インバウンド、アウトバウンド、インスペクションの各 VPC を使用してネットワークアカウントを設定することを推奨しています。

IoT

インターネットまたはローカル通信ネットワークを介して他のデバイスやシステムと通信する、センサーまたはプロセッサが組み込まれた接続済み物理オブジェクトのネットワーク。詳細については、「IoT とは」を参照してください。

解釈可能性

機械学習モデルの特性で、モデルの予測がその入力にどのように依存するかを人間が理解できる度合いを表します。詳細については、「 を使用した機械学習モデルの解釈可能性 AWS」を参照してください。

IoT

「IoT」を参照してください。

IT 情報ライブラリ (ITIL)

IT サービスを提供し、これらのサービスをビジネス要件に合わせるための一連のベストプラクティス。ITIL は ITSM の基盤を提供します。

IT サービス管理 (ITSM)

組織の IT サービスの設計、実装、管理、およびサポートに関連する活動。クラウドオペレーションと ITSM ツールの統合については、オペレーション統合ガイドを参照してください。

ITIL

「IT 情報ライブラリ」を参照してください。

ITSM

「IT サービス管理」を参照してください。

L

ラベルベースアクセス制御 (LBAC)

強制アクセス制御 (MAC) の実装で、ユーザーとデータ自体にそれぞれセキュリティラベル値が明示的に割り当てられます。ユーザーセキュリティラベルとデータセキュリティラベルが交差する部分によって、ユーザーに表示される行と列が決まります。

ランディングゾーン

ランディングゾーンは、スケーラブルで安全な、適切に設計されたマルチアカウント AWS 環境です。これは、組織がセキュリティおよびインフラストラクチャ環境に自信を持ってワークロードとアプリケーションを迅速に起動してデプロイできる出発点です。ランディングゾーンの詳細については、「安全でスケーラブルなマルチアカウント AWS 環境のセットアップ」を参照してください。

大規模言語モデル (LLM)

大量のデータで事前トレーニングされた深層学習 AI モデル。LLM では、質問への回答、ドキュメントの要約、他言語へのテキスト翻訳、文を完成させるなど、さまざまなタスクを実行できます。詳細については、「大規模言語モデル (LLM) とは何ですか?」を参照してください。

大規模な移行

300 台以上のサーバの移行。

LBAC

「ラベルベースアクセス制御」を参照してください。

最小特権

タスクの実行には必要最低限の権限を付与するという、セキュリティのベストプラクティス。詳細については、IAM ドキュメントの「最小特権アクセス許可を適用する」を参照してください。

リフトアンドシフト

「7 Rs」を参照してください。

リトルエンディアンシステム

最下位バイトを最初に格納するシステム。「エンディアン性」もご覧ください。

LLM

「大規模言語モデル」を参照してください。

下位環境

「環境」を参照してください。

M

機械学習 (ML)

パターン認識と学習にアルゴリズムと手法を使用する人工知能の一種。ML は、モノのインターネット (IoT) データなどの記録されたデータを分析して学習し、パターンに基づく統計モデルを生成します。詳細については、「機械学習」を参照してください。

メインブランチ

「ブランチ」を参照してください。

マルウェア

コンピュータのセキュリティやプライバシーを侵害するように設計されたソフトウェア。マルウェアは、コンピュータシステムの中断、機密情報の漏洩、不正アクセスを招く可能性があります。マルウェアの例には、ウイルス、ワーム、ランサムウェア、トロイの木馬、スパイウェア、キーロガーなどがあります。

マネージドサービス

AWS のサービス はインフラストラクチャレイヤー、オペレーティングシステム、プラットフォーム AWS を運用し、エンドポイントにアクセスしてデータを保存および取得します。マネージドサービスの例として、Amazon Simple Storage Service (Amazon S3) と Amazon DynamoDB が挙げられます。このサービスは、抽象化されたサービスとも呼ばれます。

製造実行システム (MES)

生産プロセスを追跡、モニタリング、文書化、制御するソフトウェアシステムであり、工場では、これによって、原材料から製品を完成させます。

MAP

「Migration Acceleration Program」を参照してください。

メカニズム

ツールを作成してその導入を推進し、導入結果を調べて調整を行うための包括的なプロセス。メカニズムとは、運用中にそれ自体を強化し改善するサイクルを意味します。詳細については、AWS 「 Well-Architected フレームワーク」の「メカニズムの構築」を参照してください。

メンバーアカウント

組織の一部である管理アカウント AWS アカウント 以外のすべて AWS Organizations。 アカウントが 組織のメンバーになることができるのは、一度に 1 つのみです。

MES

「製造実行システム」を参照してください。

Message Queuing Telemetry Transport (MQTT)

発行/サブスクライブのパターンに基づく、軽量のマシンツーマシン (M2M) 通信プロトコルであり、リソースに限りのある IoT デバイスに使用されます。

マイクロサービス

明確に定義された API を介して通信し、通常は小規模な自己完結型のチームが所有する、小規模で独立したサービスです。例えば、保険システムには、販売やマーケティングなどのビジネス

機能、または購買、請求、分析などのサブドメインにマッピングするマイクロサービスが含まれる場合があります。マイクロサービスの利点には、俊敏性、柔軟なスケーリング、容易なデプロイ、再利用可能なコード、回復力などがあります。詳細については、AWS「サーバーレスサービスを使用したマイクロサービスの統合」を参照してください。

マイクロサービスアーキテクチャ

各アプリケーションプロセスをマイクロサービスとして実行する独立したコンポーネントを使用してアプリケーションを構築するアプローチ。これらのマイクロサービスは、軽量 API を使用して、明確に定義されたインターフェイスを介して通信します。このアーキテクチャの各マイクロサービスは、アプリケーションの特定の機能に対する需要を満たすように更新、デプロイ、およびスケーリングできます。詳細については、「 でのマイクロサービスの実装 AWS」を参照してください。

Migration Acceleration Program (MAP)

組織がクラウドに移行するための強力な運用基盤を構築し、移行の初期コストを相殺するのに役立つコンサルティングサポート、トレーニング、サービスを提供する AWS プログラム。MAP には、組織的な方法でレガシー移行を実行するための移行方法論と、一般的な移行シナリオを自動化および高速化する一連のツールが含まれています。

大規模な移行

アプリケーションポートフォリオの大部分を次々にクラウドに移行し、各ウェーブでより多くのアプリケーションを高速に移動させるプロセス。この段階では、以前の段階から学んだベストプラクティスと教訓を使用して、移行ファクトリー チーム、ツール、プロセスのうち、オートメーションとアジャイルデリバリーによってワークロードの移行を合理化します。これは、AWS 移行戦略 の第 3 段階です。

移行ファクトリー

自動化された俊敏性のあるアプローチにより、ワークロードの移行を合理化する部門横断的なチーム。移行ファクトリーチームには、通常、運用、ビジネスアナリストおよび所有者、移行エンジニア、デベロッパー、およびスプリントで作業する DevOps プロフェッショナルが含まれます。エンタープライズアプリケーションポートフォリオの 20～50% は、ファクトリーのアプローチによって最適化できる反復パターンで構成されています。詳細については、このコンテンツセットの移行ファクトリーに関する解説とCloud Migration Factory ガイドを参照してください。

移行メタデータ

移行を完了するために必要なアプリケーションおよびサーバーに関する情報。移行パターンごとに、異なる一連の移行メタデータが必要です。移行メタデータの例としては、ターゲットサブネット、セキュリティグループ、 AWS アカウントなどがあります。

移行パターン

移行戦略、移行先、および使用する移行アプリケーションまたはサービスを詳述する、反復可能な移行タスク。例: AWS Application Migration Service を使用して Amazon EC2 への移行をリホストします。

Migration Portfolio Assessment (MPA)

オンラインツール。これによって、 AWS クラウドに移行するビジネスケースの検証に必要な情報を得られます。MPA は、詳細なポートフォリオ評価 (サーバーの適切なサイジング、価格設定、TCO 比較、移行コスト分析) および移行プラン (アプリケーションデータの分析とデータ収集、アプリケーションのグループ化、移行の優先順位付け、およびウェーブプランニング) を提供します。MPA ツール (ログインが必要) は、すべての AWS コンサルタントと APN パートナーコンサルタントが無料で利用できます。

移行準備状況評価 (MRA)

AWS CAF を使用して、組織のクラウド準備状況に関するインサイトを取得し、長所と短所を特定し、特定されたギャップを埋めるためのアクションプランを構築するプロセス。詳細については、移行準備状況ガイドを参照してください。MRA は、AWS 移行戦略の第一段階です。

移行戦略

ワークロードを AWS クラウドに移行するために使用するアプローチ。詳細については、この用語集の 7 Rs エントリと、「組織を動員して大規模な移行を加速する」を参照してください。

ML

「機械学習」を参照してください。

モダナイゼーション

古い (レガシーまたはモノリシック) アプリケーションとそのインフラストラクチャをクラウド内の俊敏で弾力性のある高可用性システムに変換して、コストを削減し、効率を高め、イノベーションを活用します。詳細については、「AWS クラウドでのアプリケーションのモダナイズ戦略」を参照してください。

モダナイゼーション準備状況評価

組織のアプリケーションのモダナイゼーションの準備状況を判断し、利点、リスク、依存関係を特定し、組織がこれらのアプリケーションの将来の状態をどの程度適切にサポートできるかを決定するのに役立つ評価。評価の結果として、ターゲットアーキテクチャのブループリント、モダナイゼーションプロセスの開発段階とマイルストーンを詳述したロードマップ、特定されたギャップに対処するためのアクションプランが得られます。詳細については、「AWS クラウドでのアプリケーションのモダナイゼーションの準備状況を評価する」を参照してください。

モノリシックアプリケーション (モノリス)

緊密に結合されたプロセスを持つ単一のサービスとして実行されるアプリケーション。モノリシックアプリケーションにはいくつかの欠点があります。1 つのアプリケーション機能エクスペリエンスの需要が急増する場合は、アーキテクチャ全体をスケーリングする必要があります。モノリシックアプリケーションの特徴を追加または改善することは、コードベースが大きくなると複雑になります。これらの問題に対処するには、マイクロサービスアーキテクチャを使用できます。詳細については、「モノリスをマイクロサービスに分解する」を参照してください。

MPA

「Migration Portfolio Assessment」を参照してください。

MQTT

「Message Queuing Telemetry Transport」を参照してください。

多クラス分類

複数のクラスの予測を生成するプロセス (2 つ以上の結果の 1 つを予測します)。例えば、機械学習モデルが、「この製品は書籍、自動車、電話のいずれですか?」 または、「このお客様にとって最も関心のある商品のカテゴリはどれですか?」と聞くかもしれません。

ミュータブルなインフラストラクチャ

本番ワークロードに使用する既存のインフラストラクチャを更新および変更するためのモデル。Well-Architected AWS フレームワークでは、一貫性、信頼性、予測可能性を向上させるために、イミュータブルインフラストラクチャの使用をベストプラクティスとして推奨しています。

O

OAC

「オリジンアクセス制御」を参照してください。

OAI

「オリジンアクセスアイデンティティ」を参照してください。

OCM

「組織変更管理」を参照してください。

オフライン移行

移行プロセス中にソースワークロードを停止させる移行方法。この方法はダウンタイムが長くなるため、通常は重要ではない小規模なワークロードに使用されます。

OI

「オペレーション統合」を参照してください。

Ola

「オペレーショナルレベルアグリーメント」を参照してください。

オンライン移行

ソースワークロードをオフラインにせずにターゲットシステムにコピーする移行方法。ワークロードに接続されているアプリケーションは、移行中も動作し続けることができます。この方法はダウンタイムがゼロから最小限で済むため、通常は重要な本番稼働環境のワークロードに使用されます。

OPC-UA

「Open Process Communications - Unified Architecture」を参照してください。

Open Process Communications - Unified Architecture (OPC-UA)

産業オートメーション用のマシンツーマシン (M2M) 通信プロトコル。OPC-UA により、相互運用の際に、データ暗号化、認証、認可の各スキームを標準化できます。

オペレーショナルレベルアグリーメント (OLA)

サービスレベルアグリーメント (SLA) をサポートするために、どの機能的 IT グループが互いに提供することを約束するかを明確にする契約。

運用準備状況レビュー (ORR)

質問と関連するベストプラクティスのチェックリスト。インシデントや起こり得る障害を理解、評価、防止したり、その範囲を縮小したりする際に役立ちます。詳細については、 AWS Well-Architected フレームワークの「Operational Readiness Reviews (ORR)」を参照してください。

運用テクノロジー (OT)

産業オペレーション、機器、インフラストラクチャを制御するために物理環境と連携させるハードウェアおよびソフトウェアシステム。製造分野では、Industry 4.0 への変革を進める上で、OT と情報技術 (IT) システムの統合に焦点が当てられています。

オペレーション統合 (OI)

クラウドでオペレーションをモダナイズするプロセスには、準備計画、オートメーション、統合が含まれます。詳細については、オペレーション統合ガイドを参照してください。

組織の証跡

組織 AWS アカウント 内のすべての のすべてのイベント AWS CloudTrail を ログに記録する によって作成された証跡 AWS Organizations。証跡は、組織に含まれている各 AWS アカウント に作成され、各アカウントのアクティビティを追跡します。詳細については、CloudTrail ドキュメントの「組織の証跡の作成」を参照してください。

組織変更管理 (OCM)

人材、文化、リーダーシップの観点から、主要な破壊的なビジネス変革を管理するためのフレームワーク。OCM は、変化の導入を加速し、移行問題に対処し、文化や組織の変化を推進することで、組織が新しいシステムと戦略の準備と移行するのを支援します。 AWS 移行戦略では、クラウド導入プロジェクトに必要な変化のスピードにより、このフレームワークは人材アクセラレーションと呼ばれます。詳細については、OCM ガイドを参照してください。

オリジンアクセス制御 (OAC)

Amazon Simple Storage Service (Amazon S3) コンテンツを保護するための、CloudFront のアクセス制限の強化オプション。OAC は AWS リージョン、すべての S3 バケット、 AWS KMS (SSE-KMS) によるサーバー側の暗号化、S3 バケットへの動的 PUT および DELETEリクエストをサポートします。

オリジンアクセスアイデンティティ (OAI)

CloudFront の、Amazon S3 コンテンツを保護するためのアクセス制限オプション。OAI を使用すると、CloudFront が、Amazon S3 に認証可能なプリンシパルを作成します。認証されたプリンシパルは、S3 バケット内のコンテンツに、特定の CloudFront ディストリビューションを介してのみアクセスできます。OAC も併せて参照してください。OAC では、より詳細な、強化されたアクセス制御が可能です。

ORR

「運用準備状況レビュー」を参照してください。

OT

「運用テクノロジー」を参照してください。

アウトバウンド (送信) VPC

AWS マルチアカウントアーキテクチャでは、アプリケーション内から開始されたネットワーク接続を処理する VPC。AWS Security Reference Architecture では、アプリケーションとより広範なインターネット間の双方向のインターフェイスを保護するために、インバウンド、アウトバウンド、インスペクションの各 VPC を使用してネットワークアカウントを設定することを推奨しています。

P

アクセス許可の境界

ユーザーまたはロールが使用できるアクセス許可の上限を設定する、IAM プリンシパルにアタッチされる IAM 管理ポリシー。詳細については、IAM ドキュメントのアクセス許可の境界を参照してください。

個人を特定できる情報 (PII)

直接閲覧した場合、または他の関連データと組み合わせた場合に、個人の身元を合理的に推測するために使用できる情報。PII の例には、氏名、住所、連絡先情報などがあります。

PII

「個人を特定できる情報」を参照してください。

プレイブック

クラウドでのコアオペレーション機能の提供など、移行に関連する作業を取り込む、事前定義された一連のステップ。プレイブックは、スクリプト、自動ランブック、またはお客様のモダナイズされた環境を運用するために必要なプロセスや手順の要約などの形式をとることができます。

PLC

「プログラマブルロジックコントローラー」を参照してください。

PLM

「製品ライフサイクル管理」を参照してください。

ポリシー

次の操作を可能にするオブジェクト: アクセス許可を定義する (ID ベースのポリシーを参照)。アクセス条件を指定する (リソースベースのポリシーを参照)。 AWS Organizations の組織における全アカウントにアクセス許可の上限を定義する (サービスコントロールポリシーを参照)。

多言語の永続性

データアクセスパターンやその他の要件に基づいて、マイクロサービスのデータストレージテクノロジーを個別に選択します。マイクロサービスが同じデータストレージテクノロジーを使用している場合、実装上の問題が発生したり、パフォーマンスが低下する可能性があります。マイクロサービスは、要件に最も適合したデータストアを使用すると、より簡単に実装でき、パフォーマンスとスケーラビリティが向上します。

ポートフォリオ評価

移行を計画するために、アプリケーションポートフォリオの検出、分析、優先順位付けを行うプロセス。詳細については、「移行の準備状況の評価」を参照してください。

述語

`true` または `false` を返すためのクエリ条件。一般的に、`WHERE` 句に記述されます。

述語プッシュダウン

データベースクエリを最適化する手法。これによって、転送前にクエリ内のデータをフィルタリングします。この手法を取ると、リレーショナルデータベースから取得し処理する必要のあるデータの量が減少するため、クエリのパフォーマンスが向上します。

予防的コントロール

イベントの発生を防ぐように設計されたセキュリティコントロール。このコントロールは、ネットワークへの不正アクセスや好ましくない変更を防ぐ最前線の防御です。詳細については、「AWSでのセキュリティコントロールの実装」の「予防的コントロール」を参照してください。

プリンシパル

アクションを実行し AWS 、リソースにアクセスできる のエンティティ。このエンティティは通常、、IAM AWS アカウントロール、または ユーザーのルートユーザーです。詳細については、IAM ドキュメントの「ロールに関する用語と概念」にあるプリンシパルを参照してください。

プライバシーバイデザイン

開発プロセス全体を通してプライバシーが考慮されているシステムエンジニアリングのアプローチ。

プライベートホストゾーン

1 つ以上の VPC 内のドメインとそのサブドメインへの DNS クエリに対し、Amazon Route 53 がどのように応答するかに関する情報を保持するコンテナ。詳細については、Route 53 ドキュメントの「プライベートホストゾーンの使用」を参照してください。

プロアクティブコントロール

非準拠リソースのデプロイ防止を目的としたセキュリティコントロール。このコントロールにより、プロビジョニング前にリソースをスキャンします。コントロールに準拠していないリソースは、プロビジョニングされません。詳細については、 AWS Control Tower ドキュメントの「コントロールリファレンスガイド」および「 セキュリティコントロールの実装」の「プロアクティブコントロール」を参照してください。 AWS

製品ライフサイクル管理 (PLM)

製品の設計、開発、発売から、成長、成熟、衰退、廃棄に至る、製品のライフサイクル全体を通してデータとプロセスを管理すること。

本番環境

「環境」を参照してください。

プログラマブルロジックコントローラー (PLC)

製造分野で使用される、信頼性と適応性に優れたコンピュータであり、これによって、マシンをモニタリングするとともに、製造プロセスを自動化します。

プロンプトチェイニング

1 つの LLM プロンプトによる出力を次のプロンプトの入力に使用して、より良いレスポンスを生成します。この手法を使用すると、複雑なタスクをサブタスクに分割したり、事前レスポンスを繰り返し改良または拡張したりできます。これによって、モデルのレスポンスの精度と関連性が向上し、粒度の高いパーソナライズされた結果を得られます。

仮名化

データセット内の個人識別子をプレースホルダー値に置き換えるプロセス。仮名化は個人のプライバシー保護に役立ちます。仮名化されたデータは、依然として個人データとみなされます。

発行/サブスクライブ (pub/sub)

マイクロサービス間の非同期通信を可能にするパターン。これにより、スケーラビリティと応答性を向上させます。例えば、マイクロサービスベースの MES の場合、マイクロサービスは、他のマイクロサービスがサブスクライブ可能なチャネルにイベントメッセージを発行できます。このシステムでは、発行サービスの変更なしに、新規マイクロサービスを追加できます。

Q

クエリプラン

手順などの一連のステップであり、SQL リレーショナルデータベースシステムのデータにアクセスするために使用されます。

クエリプランのリグレッション

データベースサービスのオプティマイザーが、データベース環境に特定の変更が加えられる前に選択されたプランよりも最適性の低いプランを選択すること。これは、統計、制限事項、環境設定、クエリパラメータのバインディングの変更、およびデータベースエンジンの更新などが原因である可能性があります。

R

RACI マトリックス

「実行責任者、説明責任者、協業先、報告先 (RACI)」を参照してください。

RAG

「検索拡張生成」を参照してください。

ランサムウェア

決済が完了するまでコンピュータシステムまたはデータへのアクセスをブロックするように設計された、悪意のあるソフトウェア。

RASCI マトリックス

「実行責任者、説明責任者、協業先、報告先 (RACI)」を参照してください。

RCAC

「行と列のアクセス制御」を参照してください。

リードレプリカ

読み取り専用に使用されるデータベースのコピー。クエリをリードレプリカにルーティングして、プライマリデータベースへの負荷を軽減できます。

リアーキテクト

「7 Rs」を参照してください。

目標復旧時点 (RPO)

最後のデータリカバリポイントからの最大許容時間です。これにより、最後の回復時点からサービスが中断されるまでの間に許容できるデータ損失の程度が決まります。

目標復旧時間 (RTO)

サービスの中断から復旧までの最大許容遅延時間。

リファクタリング

「7 Rs」を参照してください。

リージョン

地理的エリア内の AWS リソースのコレクション。各 AWS リージョン は、耐障害性、安定性、耐障害性を提供するために、他の から分離され、独立しています。詳細については、「アカウントが使用できる AWS リージョン を指定する」を参照してください。

リグレッション

数値を予測する機械学習手法。例えば、「この家はどれくらいの値段で売れるでしょうか?」という問題を解決するために、機械学習モデルは、線形回帰モデルを使用して、この家に関する既知の事実 (平方フィートなど) に基づいて家の販売価格を予測できます。

リホスト

「7 Rs」を参照してください。

リリース

デプロイプロセスで、変更を本番環境に昇格させること。

再配置

「7 Rs」を参照してください。

リプラットフォーム

「7 Rs」を参照してください。

再購入

「7 Rs」を参照してください。

回復性

中断に抵抗または中断から回復するアプリケーションの機能。 AWS クラウドでの回復力を計画する際には、一般的に、高可用性とディザスタリカバリが考慮されます。詳細については、「AWS クラウド の耐障害性」を参照してください。

リソースベースのポリシー

Amazon S3 バケット、エンドポイント、暗号化キーなどのリソースにアタッチされたポリシー。このタイプのポリシーは、アクセスが許可されているプリンシパル、サポートされているアクション、その他の満たすべき条件を指定します。

実行責任者、説明責任者、協業先、報告先 (RACI) に基づくマトリックス

移行活動とクラウド運用に関わるすべての関係者の役割と責任を定義したマトリックス。マトリックスの名前は、マトリックスで定義されている責任の種類、すなわち責任 (R) 、説明責任 (A) 、協議 (C) 、情報提供 (I) に由来します。サポート (S) タイプはオプションです。サポートが含まれる場合は RASCI マトリックスと呼ばれ、含まれない場合は RACI マトリックスと呼ばれます。

レスポンシブコントロール

有害事象やセキュリティベースラインからの逸脱について、修復を促すように設計されたセキュリティコントロール。詳細については、「 AWSでのセキュリティコントロールの実装」の「レスポンシブコントロール」を参照してください。

保持

「7 Rs」を参照してください。

廃止

「7 Rs」を参照してください。

検索拡張生成 (RAG)

生成 AI の技術。これにより、LLM では、レスポンスの生成前に、トレーニングデータソースの外部にある信頼できるデータソースが参照されます。例えば、RAG モデルによって、組織のナレッジベースまたはカスタムデータのセマンティック検索を実行できる場合があります。細については、「RAG (検索拡張生成) とは何ですか?」を参照してください。

ローテーション

定期的にシークレット情報を更新して、攻撃者が認証情報にアクセスするのをより困難にするプロセス。

行と列のアクセス制御 (RCAC)

アクセスルールが定義された、基本的で柔軟な SQL 表現の使用。RCAC は行権限と列マスクで構成されています。

RPO

「目標復旧時点」を参照してください。

RTO

「目標復旧時間」を参照してください。

ランブック

特定のタスクを実行するために必要な手動または自動化された一連の手順。これらは通常、エラー率の高い反復操作や手順を合理化するために構築されています。

S

SAML 2.0

多くの ID プロバイダー (IdP) が使用しているオープンスタンダード。この機能を使用すると、フェデレーティッドシングルサインオン (SSO) が有効になるため、ユーザーは組織内のすべてのユーザーを IAM で作成しなくても、 AWS マネジメントコンソール にログインしたり AWS 、API オペレーションを呼び出すことができます。SAML 2.0 ベースのフェデレーションの詳細については、IAM ドキュメントの「SAML 2.0 ベースのフェデレーションについて」を参照してください。

SCADA

「監視制御とデータ取得」を参照してください。

SCP

「サービスコントロールポリシー」を参照してください。

シークレット

暗号化された形式で保存する AWS Secrets Managerパスワードやユーザー認証情報などの機密情報または制限付き情報。シークレット値とそのメタデータで構成されます。シークレット値には、バイナリ、1 つの文字列、複数の文字列を指定できます。詳細については、Secrets Manager ドキュメントの「Secrets Manager シークレットの概要」を参照してください。

セキュリティバイデザイン

開発プロセス全体を通してセキュリティが考慮されているシステムエンジニアリングのアプローチ。

セキュリティコントロール

脅威アクターによるセキュリティ脆弱性の悪用を防止、検出、軽減するための、技術上または管理上のガードレール。セキュリティコントロールには、主に 4 つの種類があります。4 つとは、予防、検出、レスポンシブ、プロアクティブです。

セキュリティ強化

アタックサーフェスを狭めて攻撃への耐性を高めるプロセス。このプロセスには、不要になったリソースの削除、最小特権を付与するセキュリティのベストプラクティスの実装、設定ファイル内の不要な機能の無効化、といったアクションが含まれています。

Security Information and Event Management (SIEM) システム

セキュリティ情報管理 (SIM) とセキュリティイベント管理 (SEM) のシステムを組み合わせたツールとサービス。SIEM システムは、サーバー、ネットワーク、デバイス、その他ソースからデータを収集、モニタリング、分析して、脅威やセキュリティ違反を検出し、アラートを発信します。

セキュリティレスポンスの自動化

セキュリティイベントへの自動レスポンスまたは自動修復を目的として、事前定義およびプログラムされたアクション。これらの自動化は、セキュリティのベストプラクティスを実装するのに役立つ検出的または応答的な AWS セキュリティコントロールとして機能します。自動レスポンスアクションの例には、VPC セキュリティグループの変更、Amazon EC2 インスタンスへのパッチ適用、認証情報の更新などがあります。

サーバー側の暗号化

送信先で、それ AWS のサービス を受け取る によるデータの暗号化。

サービスコントロールポリシー (SCP)

AWS Organizationsの組織内の、すべてのアカウントのアクセス許可を一元的に管理するポリシー。SCP は、管理者がユーザーまたはロールに委任するアクションに、ガードレールを定義したり、アクションの制限を設定したりします。SCP は、許可リストまたは拒否リストとして、許可または禁止するサービスやアクションを指定する際に使用できます。詳細については、 AWS Organizations ドキュメントの「サービスコントロールポリシー」を参照してください。

サービスエンドポイント

のエントリポイントの URL AWS のサービス。ターゲットサービスにプログラムで接続するには、エンドポイントを使用します。詳細については、「AWS 全般のリファレンス」の「AWS のサービス エンドポイント」を参照してください。

サービスレベルアグリーメント (SLA)

サービスのアップタイムやパフォーマンスなど、IT チームがお客様に提供すると約束したものを明示した合意書。

サービスレベルインジケータ (SLI)

エラー率、可用性、スループットといった、サービスパフォーマンス面の指標。

サービスレベル目標 (SLO)

サービスレベルインジケータによって測定され、サービスの状態を表すターゲットメトリクス。

責任共有モデル

クラウドのセキュリティとコンプライアンス AWS について と共有する責任を説明するモデル。AWS はクラウドのセキュリティを担当しますが、 はクラウドのセキュリティを担当します。詳細については、「責任共有モデル」を参照してください。

SIEM

「Security Information and Event Management システム」を参照してください。

単一障害点 (SPOF)

特定のアプリケーションを構成する単一の重要なコンポーネントで発生し、システム稼働に支障をきたす可能性のある障害。

SLA

「サービスレベルアグリーメント」を参照してください。

SLI

「サービスレベルインジケータ」を参照してください。

SLO

「サービスレベルの目標」を参照してください。

スプリットアンドシードモデル

モダナイゼーションプロジェクトのスケーリングと加速のためのパターン。新機能と製品リリースが定義されると、コアチームは解放されて新しい製品チームを作成します。これにより、お客様の組織の能力とサービスの拡張、デベロッパーの生産性の向上、迅速なイノベーションのサポートに役立ちます。詳細については、「AWS クラウドでのアプリケーションをモダナイズするための段階的アプローチ」を参照してください。

SPOF

「単一障害点」を参照してください。

スタースキーマ

データベースの編成構造を意味し、1 つの大きいファクトテーブルにトランザクションデータまたは測定データが保存され、1 つ以上の小さいディメンションテーブルにデータ属性が保存されます。この構造は、データウェアハウスやビジネスインテリジェンスを用途とするように設計されています。

strangler fig パターン

レガシーシステムが廃止されるまで、システム機能を段階的に書き換えて置き換えることにより、モノリシックシステムをモダナイズするアプローチ。このパターンは、宿主の樹木から根を成長させ、最終的にその宿主を包み込み、宿主に取って代わるイチジクのつるを例えています。そのパターンは、モノリシックシステムを書き換えるときのリスクを管理する方法として Martin Fowler により提唱されました。このパターンの適用方法の例については、「コンテナと Amazon API Gateway を使用して、従来の Microsoft ASP.NET (ASMX) ウェブサービスを段階的にモダナイズ」を参照してください。

サブネット

VPC 内の IP アドレスの範囲。サブネットは、1 つのアベイラビリティーゾーンに存在する必要があります。

監視制御とデータ取得 (SCADA)

製造分野において、ハードウェアとソフトウェアを使用して物理アセットと本番運用をモニタリングするシステム。

対称暗号化

データの暗号化と復号に同じキーを使用する暗号化のアルゴリズム。

合成テスト

ユーザーとのやり取りをシミュレートして、起こり得る問題を検出したり、パフォーマンスをモニタリングしたりすることで、システムをテストします。Amazon CloudWatch Synthetics を使用すると、こうしたテストを作成できます。

システムプロンプト

コンテキスト、指示、ガイドラインなどを提示して、LLM に動作を指示する手法。システムプロンプトは、コンテキストを設定して、ユーザーとやり取りするルールを確立するのに有用です。

T

タグ

AWS リソースを整理するためのメタデータとして機能するキーと値のペア。タグは、 リソースの管理、識別、整理、検索、フィルタリングに役立ちます。詳細については、「AWS リソースのタグ付け」を参照してください。

ターゲット変数

監督された機械学習でお客様が予測しようとしている値。これは、結果変数のことも指します。例えば、製造設定では、ターゲット変数が製品の欠陥である可能性があります。

タスクリスト

ランブックの進行状況を追跡するために使用されるツール。タスクリストには、ランブックの概要と完了する必要のある一般的なタスクのリストが含まれています。各一般的なタスクには、推定所要時間、所有者、進捗状況が含まれています。

テスト環境

「環境」を参照してください。

トレーニング

お客様の機械学習モデルに学習するデータを提供すること。トレーニングデータには正しい答えが含まれている必要があります。学習アルゴリズムは入力データ属性をターゲット (お客様が予測したい答え) にマッピングするトレーニングデータのパターンを検出します。これらのパターンをキャプチャする機械学習モデルを出力します。そして、お客様が機械学習モデルを使用して、ターゲットがわからない新しいデータでターゲットを予測できます。

トランジットゲートウェイ

VPC とオンプレミスネットワークを相互接続するために使用できる、ネットワークの中継ハブ。詳細については、 AWS Transit Gateway ドキュメントの「トランジットゲートウェイとは」を参照してください。

トランクベースのワークフロー

デベロッパーが機能ブランチで機能をローカルにビルドしてテストし、その変更をメインブランチにマージするアプローチ。メインブランチはその後、開発環境、本番前環境、本番環境に合わせて順次構築されます。

信頼されたアクセス

ユーザーに代わって AWS Organizations およびそのアカウントで組織内でタスクを実行するために指定したサービスにアクセス許可を付与します。信頼されたサービスは、サービスにリンクされたロールを必要なときに各アカウントに作成し、ユーザーに代わって管理タスクを実行します。詳細については、 ドキュメントの「Using AWS Organizations with other AWS services AWS Organizations 」を参照してください。

チューニング

機械学習モデルの精度を向上させるために、お客様のトレーニングプロセスの側面を変更する。例えば、お客様が機械学習モデルをトレーニングするには、ラベル付けセットを生成し、ラベルを追加します。これらのステップを、異なる設定で複数回繰り返して、モデルを最適化します。

ツーピザチーム

2 枚のピザを分け合えることができるくらい小さな DevOps チーム。ツーピザチームの規模では、ソフトウェア開発におけるコラボレーションに最適な機会が確保されます。

U

不確実性

予測機械学習モデルの信頼性を損なう可能性がある、不正確、不完全、または未知の情報を指す概念。不確実性には、次の 2 つのタイプがあります。認識論的不確実性は、限られた、不完全なデータによって引き起こされ、弁論的不確実性 は、データに固有のノイズとランダム性によって引き起こされます。詳細については、深層学習システムにおける不確実性の定量化ガイドを参照してください。

未分化なタスク

ヘビーリフティングとも呼ばれ、アプリケーションの作成と運用には必要だが、エンドユーザーに直接的な価値をもたらさなかったり、競争上の優位性をもたらしたりしない作業です。未分化なタスクの例としては、調達、メンテナンス、キャパシティプランニングなどがあります。

上位環境

「環境」を参照してください。

V

バキューミング

ストレージを再利用してパフォーマンスを向上させるために、増分更新後にクリーンアップを行うデータベースのメンテナンス操作。

バージョンコントロール

リポジトリ内のソースコードへの変更など、変更を追跡するプロセスとツール。

VPC ピアリング

プライベート IP アドレスを使用してトラフィックをルーティングできる、2 つの VPC 間の接続。詳細については、Amazon VPC ドキュメントの「VPC ピア機能とは」を参照してください。

脆弱性

システムのセキュリティを脅かすソフトウェアまたはハードウェアの欠陥。

W

ウォームキャッシュ

頻繁にアクセスされる最新の関連データを含むバッファキャッシュ。データベースインスタンスはバッファキャッシュから、メインメモリまたはディスクからよりも短い時間で読み取りを行うことができます。

ウォームデータ

アクセス頻度の低いデータ。この種類のデータをクエリする場合、通常は適度に遅いクエリでも問題ありません。

ウィンドウ関数

現在のレコードに何らかの形で関連している行のグループに計算を実行する SQL 関数。ウィンドウ関数は、移動平均を計算したり、現在の行の相対位置に基づいて他の行の値にアクセスするといったタスクの処理に役立ちます。

ワークロード

ビジネス価値をもたらすリソースとコード (顧客向けアプリケーションやバックエンドプロセスなど) の総称。

ワークストリーム

特定のタスクセットを担当する移行プロジェクト内の機能グループ。各ワークストリームは独立していますが、プロジェクト内の他のワークストリームをサポートしています。たとえば、ポートフォリオワークストリームは、アプリケーションの優先順位付け、ウェーブ計画、および移行メタデータの収集を担当します。ポートフォリオワークストリームは、これらの設備を移行ワークストリームで実現し、サーバーとアプリケーションを移行します。

WORM

「Write-Once-Read-Many」を参照してください。

WQF

「AWS ワークロード資格フレームワーク」を参照してください

Write-Once-Read-Many (WORM)

データを 1 回のみ書き込むことで、データの削除や変更を防ぐストレージモデル。承認済みユーザーは、必要な回数だけデータを読み取ることができますが、変更することはできません。このデータストレージインフラストラクチャは、イミュータブルと見なされます。

Z

ゼロデイエクスプロイト

ゼロデイ脆弱性を悪用した攻撃 (一般的にマルウェアによる)。

ゼロデイ脆弱性

実稼働システムにおける未解決の欠陥または脆弱性。脅威アクターは、このような脆弱性を利用してシステムを攻撃する可能性があります。開発者は、よく攻撃の結果で脆弱性に気付きます。

ゼロショットプロンプト

LLM にタスク実行の手順は提示するが、実行のガイドとして役立つ例 (ショット) は提示しない方法。LLM は、事前トレーニング済みの知識を使用してタスクを処理する必要があります。ゼロショットプロンプトの有効性は、タスクの複雑さとプロンプトの品質によって異なります。「数ショットプロンプト」も参照してください。

ゾンビアプリケーション

平均 CPU およびメモリ使用率が 5% 未満のアプリケーション。移行プロジェクトでは、これらのアプリケーションを廃止するのが一般的です。

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。