通信事業者向け カスタム AI エージェントワークショップ
APRIL 23, 2026

AWS で実現するカスタム AI エージェント

Strands Agents &
Amazon Bedrock AgentCore のご紹介

岡本 篤志

ソリューションアーキテクト

アマゾン ウェブ サービス ジャパン 合同会社
技術統括本部　ストラテジックインダストリー技術本部　通信グループ

自己紹介

岡本　篤志

Atsushi Okamoto

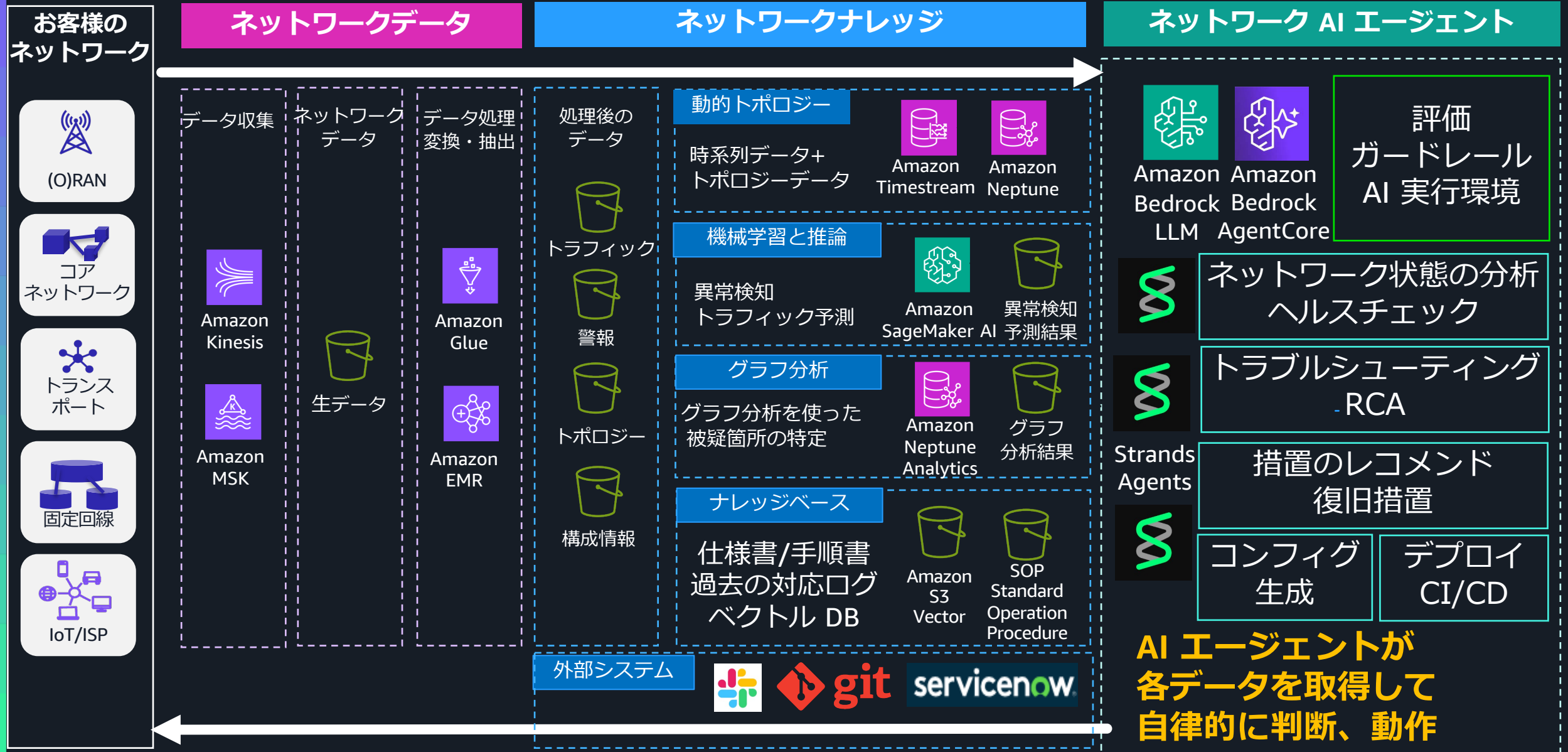
アマゾンウェブサービスジャパン
ソリューションアーキテクト

通信業界のお客様を中心にご支援しています。

好きな AWS サービス：Amazon Bedrock

Autonomous Network 参考アーキテクチャ

ネットワークナレッジ × AI エージェント により自律化を実現

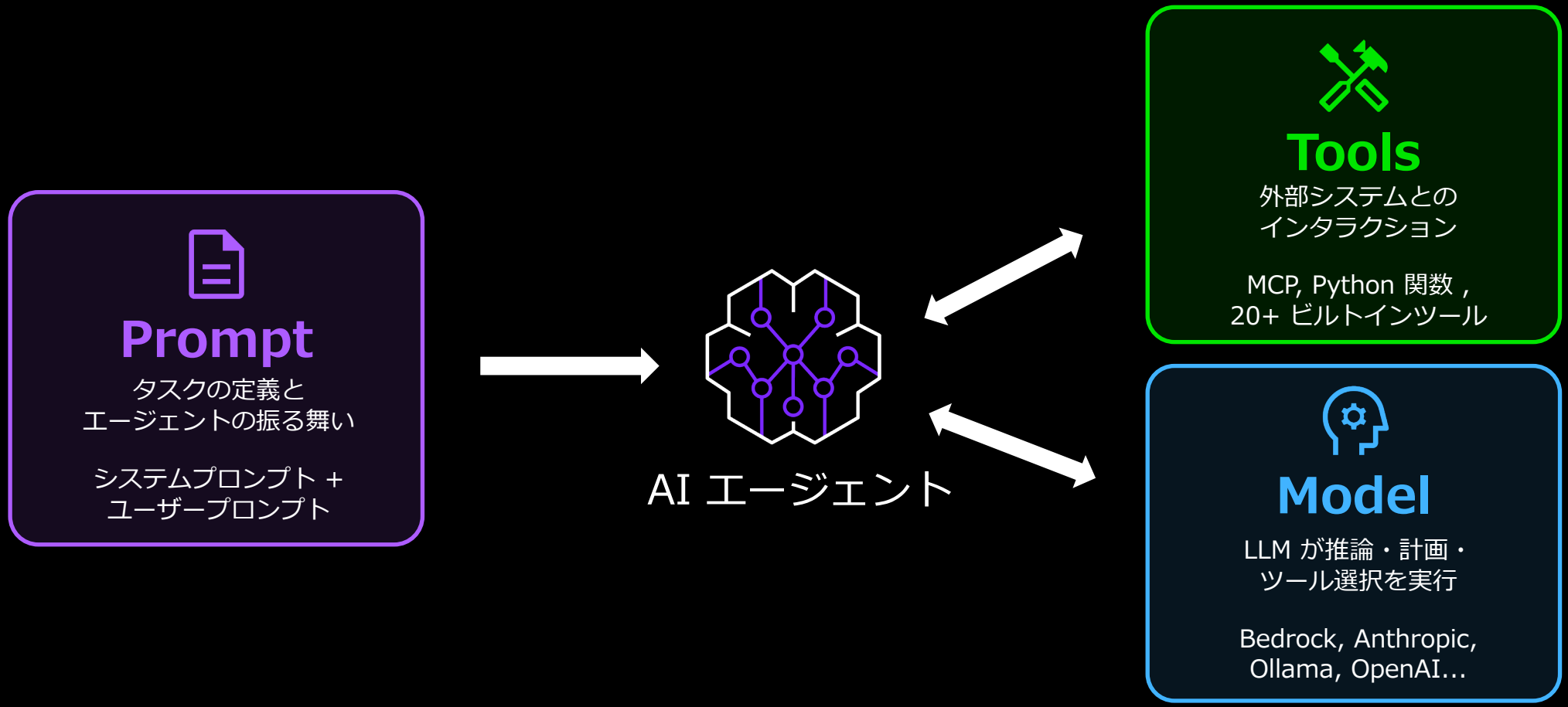


本日のセッション ① カスタム AI エージェント

Strands Agents

AI エージェントの構成要素

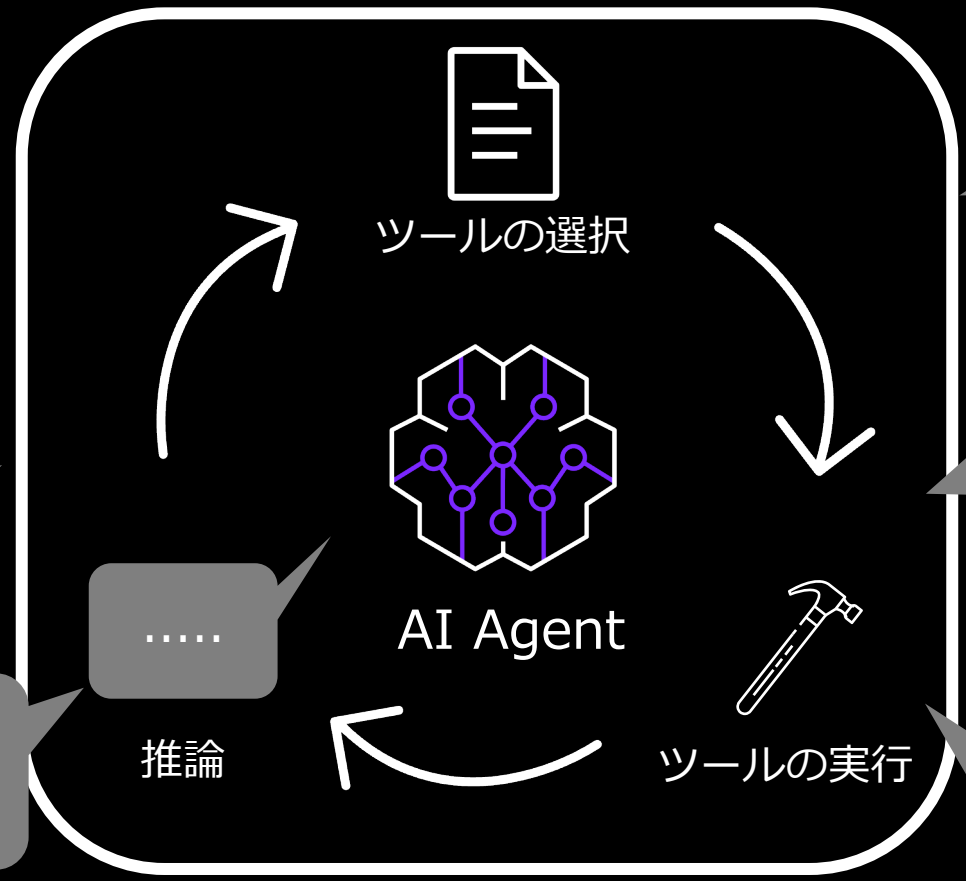
ユーザーからの指示に対して、自律的に計画を立て、ツールを利用してタスクを実行する



https://aws.amazon.com/jp/blogs/news/introducing-strands-agents-an-open-source-ai-agents-sdk/

AI エージェントを実装する際のハードル

ユーザーからの指示に対して、自律的に計画を立て、ツールを利用してタスクを実行する



状態管理はどうする？

オブザーバビリティを
どう確保する？

思考の連鎖を
どう実装する？

安全性をどう確保
する？

Tools

モデルの変更に
どう対応する？

Model

生成AI とツール間の
インターフェースをどう
実装する？

Strands Agents とは

Strands Agents

Strands Agentsは、わずか数行の
コードでエージェントを構築できる
オープンソースのSDKであり、
PythonとTypeScriptの両方をフルサポート

軽量かつシンプル: 数分で使い始められる直感的なエージェント開発

本番稼働対応: 大規模運用に向けた完全なオブザーバビリティ、トレーシング、デプロイ手段

拡張性: 様々なプロバイダの多様なモデルをサポート、ツールの自作、MCP 対応

セキュリティ: データを保護し、責任あるかたちでエージェントを運用

シンプルなAIエージェントの実装

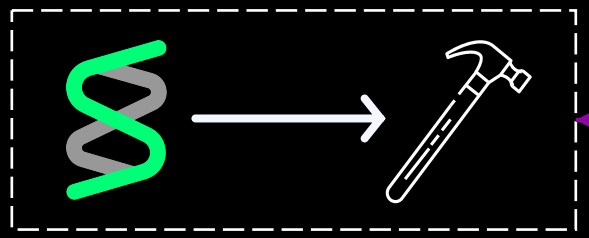
わずか数行のコードで AI エージェントを実装

simple_agent.py > ...

```python
from strands import Agent

# デフォルト設定でエージェントを作成する
agent = Agent()

# エージェントに対して問い合わせる
agent("AI エージェントについて説明して")
```

```
% python3 simple_agent.py
AIエージェントについて詳しく説明いたします。

## AIエージェントとは

AIエージェントは、**自律的に環境を認識し、目標達成のために行動を選択・実行するAIシステム**です。単純な質問応答を超えて、複雑なタスクを独立して処理できる能力を持ちます。

## 主な特徴

### 🎯 **自律性**
- 人間の詳細な指示なしに行動
- 状況に応じた判断と意思決定

### 🔄 **反応性**
- 環境の変化をリアルタイムで感知
- 適切なタイミングでの対応

### 🎯 **目標指向**
- 明確な目的に向かって行動
- 効率的な手段の選択

### 📚 **学習能力**
- 経験から学習し改善
- パフォーマンスの継続的向上

## 主要な種類

| タイプ | 説明 | 具体例 |
|---|---|---|
| **対話型** | 自然言語での会話 | ChatGPT、Claude |
| **タスク実行型** | 特定業務の自動化 | RPAbot、スケジュール管理 |
| **推薦型** | パーソナライズされた提案 | Netflix、Amazon |
| **ゲーム型** | ゲーム環境での戦略実行 | AlphaGo、OpenAI Five |

## 活用分野

- **カスタマーサービス**: 24時間対応のチャットボット
- **業務自動化**: データ処理、レポート作成
- **研究開発**: 科学実験の設計・実行支援
- **教育**: パーソナライズされた学習支援
- **金融**: 投資アドバイス、リスク管理

## 今後の展望

AIエージェントは今後、より高度な推論能力と複数のツールを組み合わせた**マルチモーダル**な能力を獲得し、人間との協働がさらに深化していくと予想されます。
```

様々なツールと接続



Pythonベースのツール

- コードベースに直接定義したPython関数を、@tool デコレーターを付けるだけでエージェントに組み込み可能
- ホットリロードに対応しているため、再起動することなくツールの定義をリアルタイムで更新可能

コミュニティが提供するツール

- ファイル操作やWeb検索など、よく使われる機能があらかじめオープンソースとして実装
- インポートして登録するだけでエージェントの機能をすぐに拡張、開発時間を大幅短縮可能

MCPツール

- MCPサーバーが公開する標準化されたツールとの接続
- ツール呼出、パラメータマッピング、処理結果の取得をエージェントが自動的に処理

Agent-as-tools

- エージェント自身を別のエージェントから呼び出せるツールとして登録
- 複数のエージェントが連携する強力なマルチエージェント構成を実現

複雑なタスクにおける課題

エージェントに**複雑なマルチステップタスク**を任せると ...

一貫性の欠如

同じタスクでも
毎回違う手順で動く
結果が予測できない

再現性の問題

人によって結果が変わる
チーム間で品質がばらつく
評価基準が不統一

ノウハウの属人化

プロンプトが個人に依存
チームで共有できない
引き継ぎが困難

プロンプトを長く書いても、構造がなければ品質はばらつく

Agent SOP

AI エージェント向けの**自然言語ワークフロー**

Markdown ベース
自然言語で記述
誰でも読み書きできる
バージョン管理可能

RFC 2119 準拠
MUST / SHOULD / MAY
推論の自由度と確実性の
バランスを制御

パラメータ化
入力を柔軟に変更
異なるプロジェクトで再利用
チーム間で共有

使い方

MCP Server
strands-agents-sops mcp

Kiro CLI, Claude Code 等から
直接 SOP を呼び出し

Strands Agents 統合
```
from strands_agents_sops import code_assist
agent = Agent(
    system_prompt=code_assist,
    tools=[editor, shell]
)
```

Anthropic Skills
strands-agents-sops skills
Claude.ai / Claude API で利用

Cursor 統合
strands-agents-sops commands --type cursor
/code-assist.sop で呼び出し

SOPの構造

Overview — 目的と使用場面

Parameters — 柔軟な入力定義

Steps — ステップバイステップの手順

Constraints — RFC 2119 制約
MUST 必須 / SHOULD 推奨 / MAY 任意

条件分岐 — Stepsの説明を If [条件], [アクション]. Otherwise, [代替アクション]. で記述
Constraintsでは各条件ごとに If [条件], You MUST [行動] の形式で具体的な制約を列挙

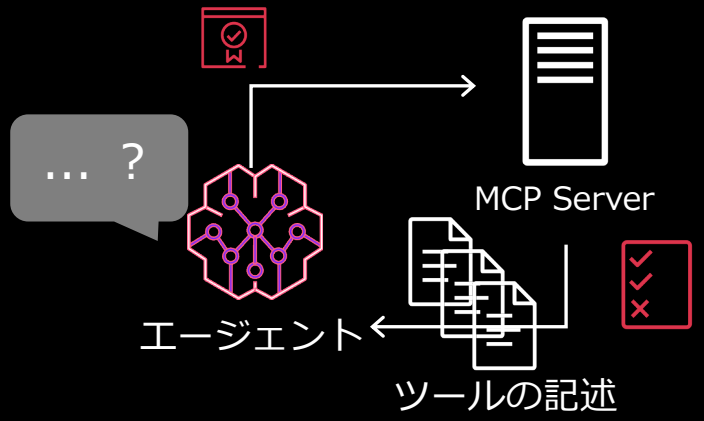
Examples — SOPの入力と出力の具体例

Troubleshooting — よくある問題とその解決策を ### 問題名 + If [状況], [対処法] の形式で記載

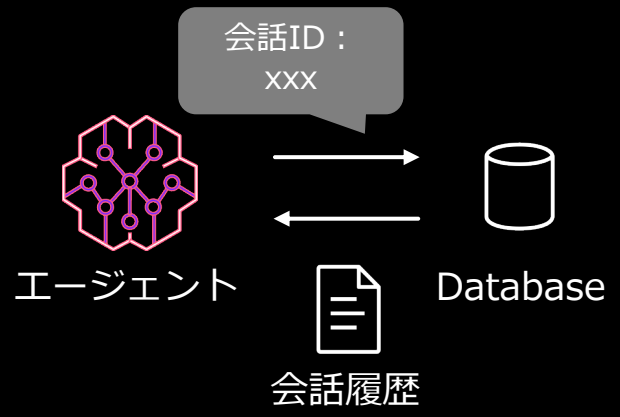

```
# SOP名

## Overview

## Parameters
- **param_name** (required): パラメータの説明
- **optional_param** (optional, default: "値"): パラメータの説明

## Steps
### 1. ステップ名
何をするかの自然言語による説明。

**Constraints:**
- You MUST 必須の行動
- You SHOULD 推奨の行動
- You MAY 任意の行動

### 2. 次のステップ
説明。

**Constraints:**
- You MUST NOT 禁止事項 because 理由

### 3. 条件分岐のあるステップ
If テストが全件パスした場合, デプロイを実行する。Otherwise, エラーログを出力して中断する。

**Constraints:**
- You MUST check テスト結果 before proceeding
- If テストがパスした場合, You MUST デプロイスクリプトを実行する
- If テストが失敗した場合, You MUST 失敗内容をログに記録して処理を中断する

## Examples
### Example Input
ユーザーが `deploy --env staging` を実行した場合

### Example Output
staging環境へのデプロイが完了し、デプロイログが `deploy-log.md` に保存される。

## Troubleshooting
### デプロイがタイムアウトする
If ネットワーク接続が不安定な場合, リトライ回数を増やして再実行する。
### テストが不安定に失敗する
flaky testが疑われる場合、該当テストを特定し `--retry` オプション付きで再実行する。
```

Amazon Bedrock AgentCore

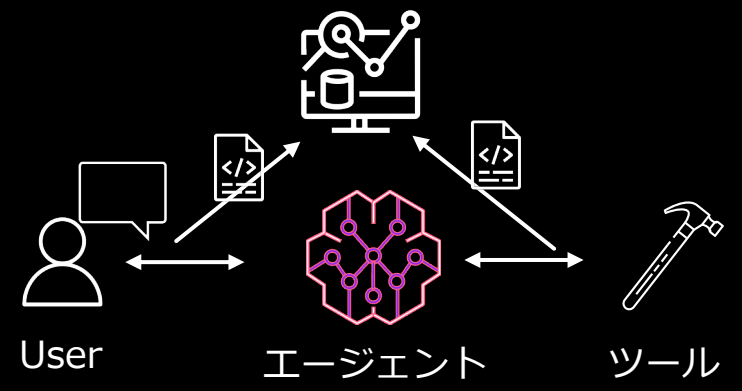
AIエージェントデプロイに関する共通の課題



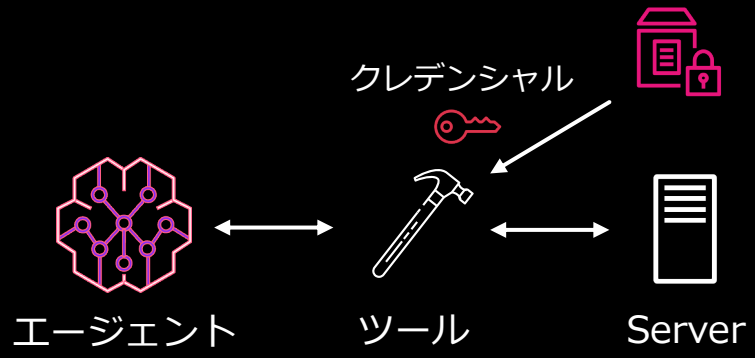
大量のツールの管理・統合
ツールの認証・認可の実装



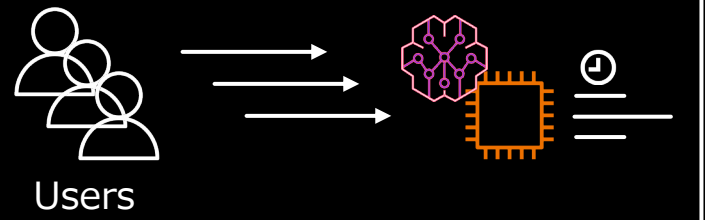
短期・長期の情報を保持・検索する
メモリ基盤の実装



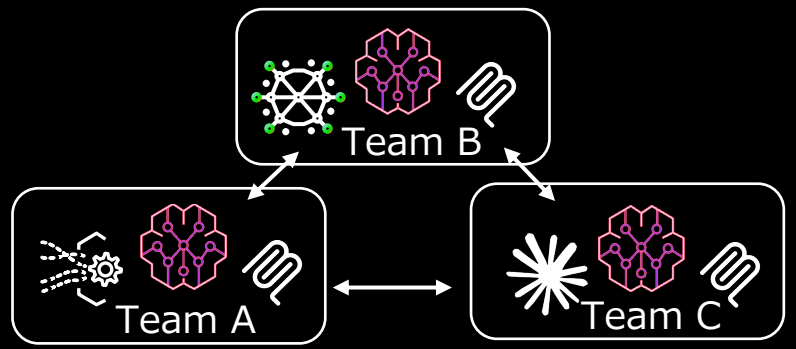
プロンプト、ツールI/O、レスポンス等
各ステップのログとトレース基盤の実装



エージェント固有のアイデンティティ・
資格情報の管理の実装



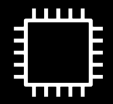
スケーラビリティ、高速な起動
セッション分離、長期ランタイムを備えた
コンピューティング基盤の選定



エージェントがスケールするための、チーム
間でのム間でエージェント・ツールの可視性
とガバナンスの向上 (Agent Sprawl問題)

エージェントの本番稼働に向けて実装すべき事項

安全なエージェントを大規模に展開するためには、
エージェントの周辺機能の開発に多くの工数を割く必要がある



エージェントのホスト

認証・認可

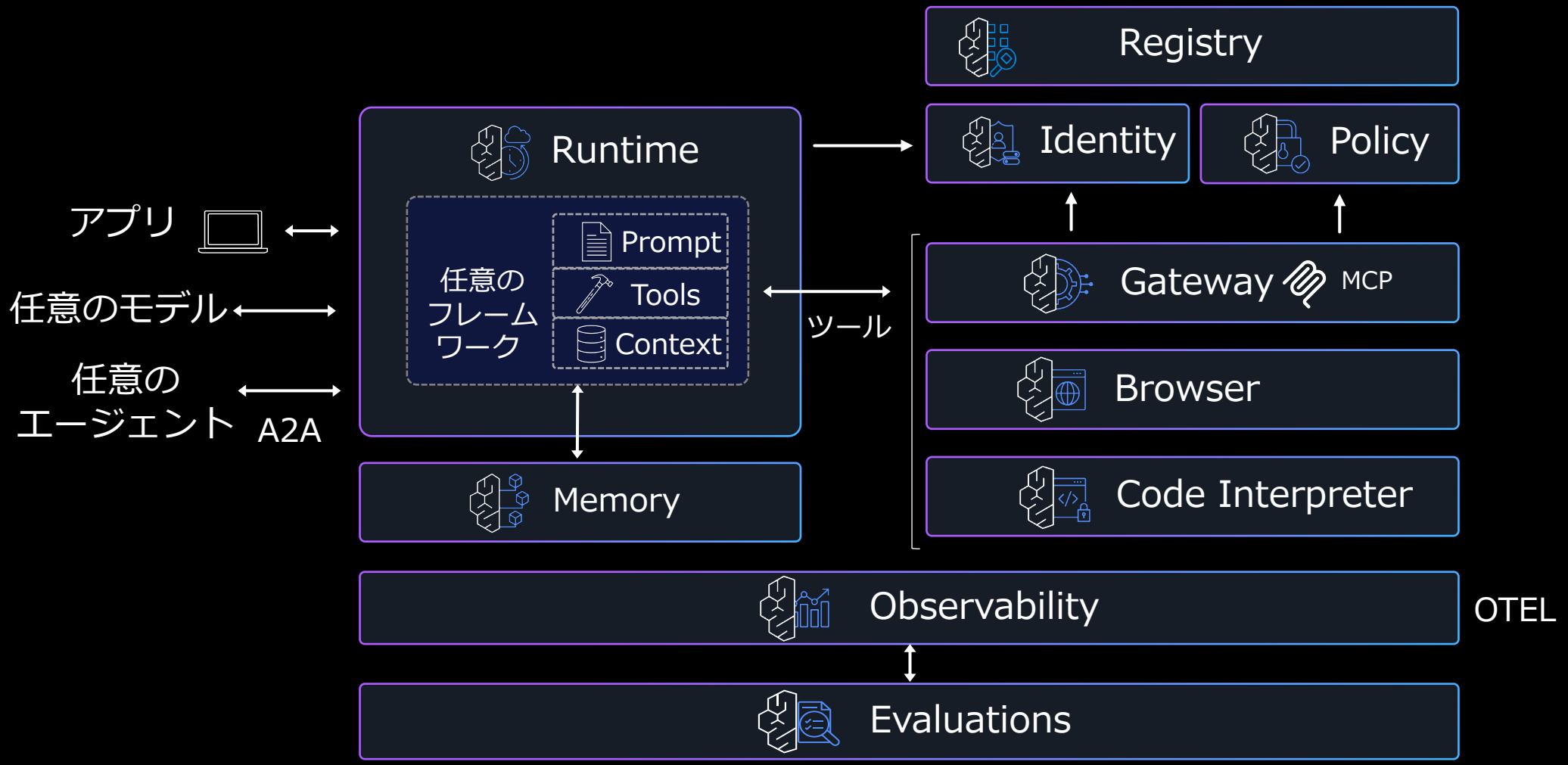
ツールの管理

セッション管理

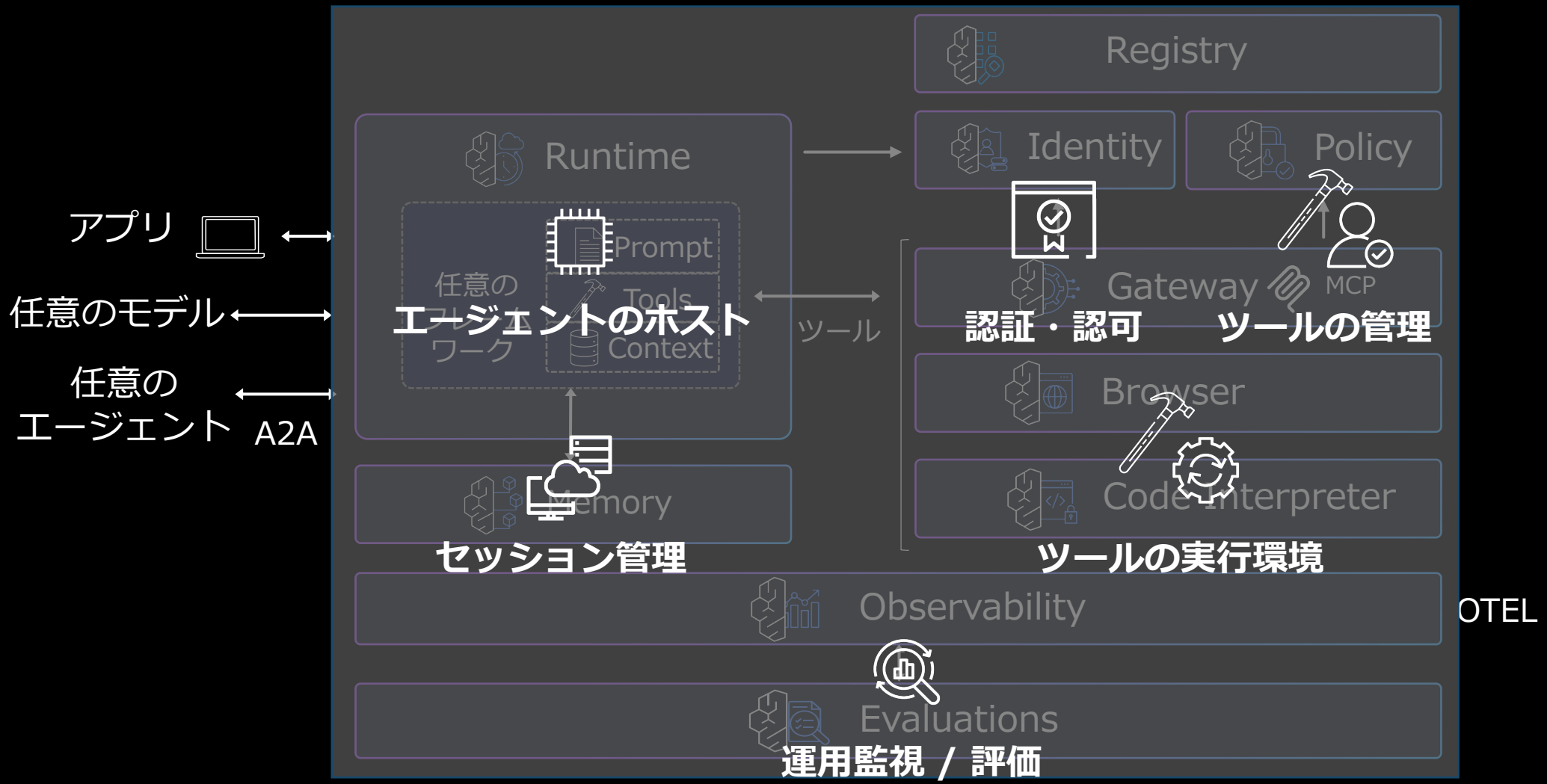
ツールの実行環境

運用監視 / 評価

Amazon Bedrock AgentCore プラットフォーム

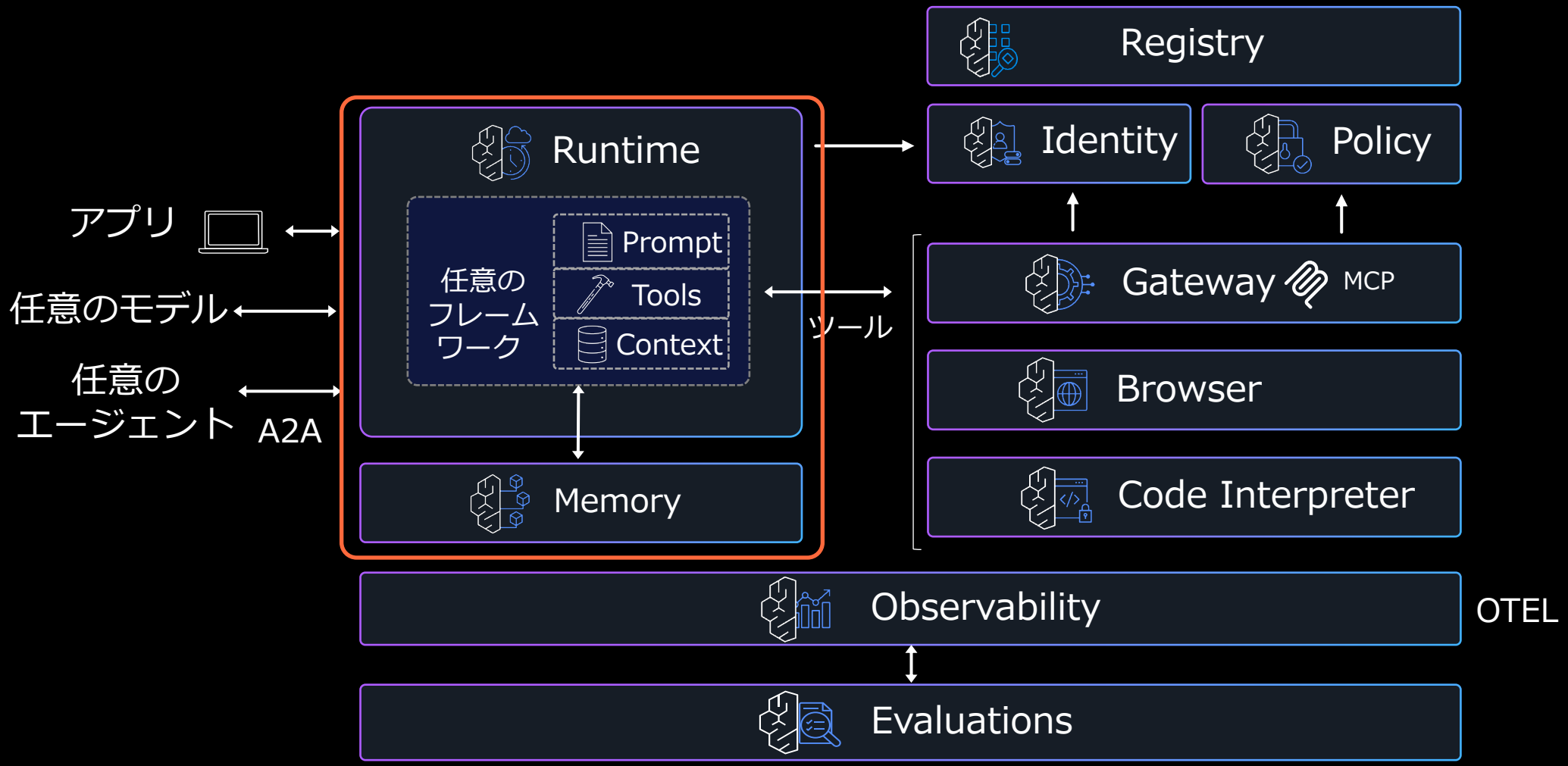


Amazon Bedrock AgentCore プラットフォーム

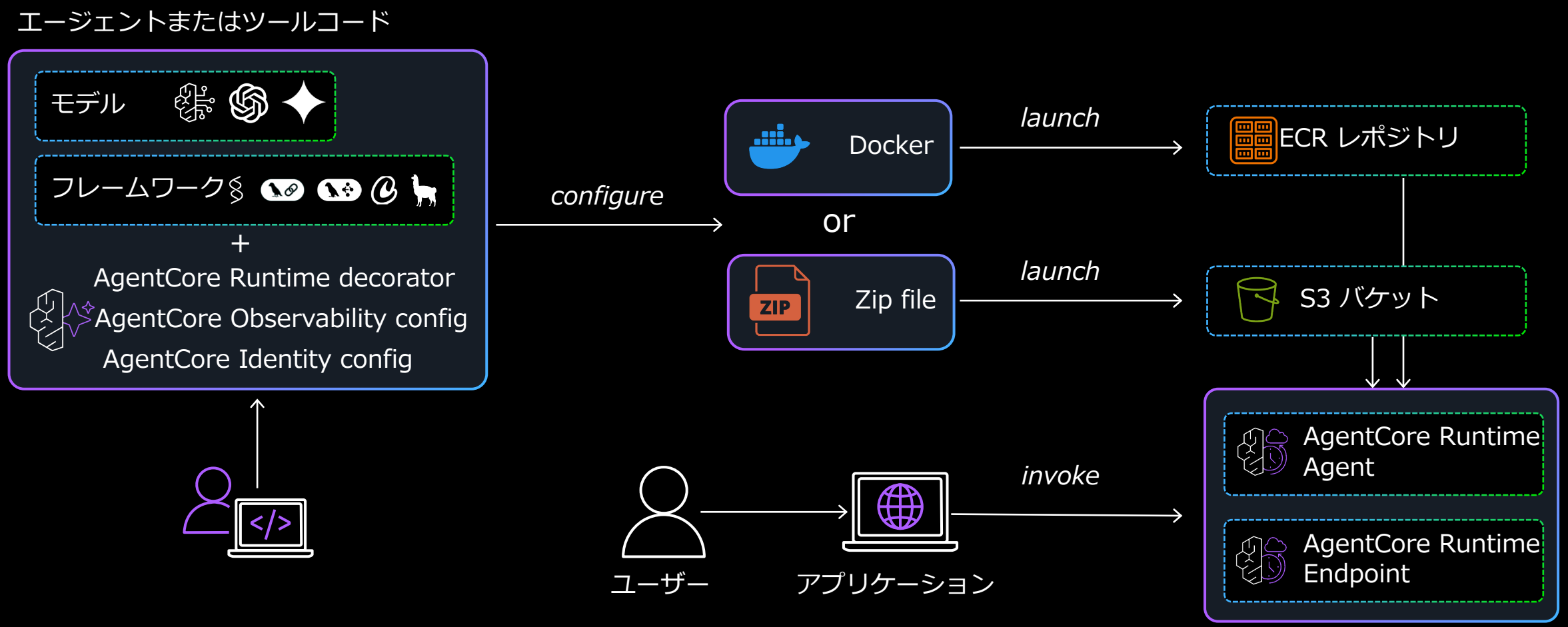


Amazon Bedrock AgentCore プラットフォーム

エージェント起動の基本

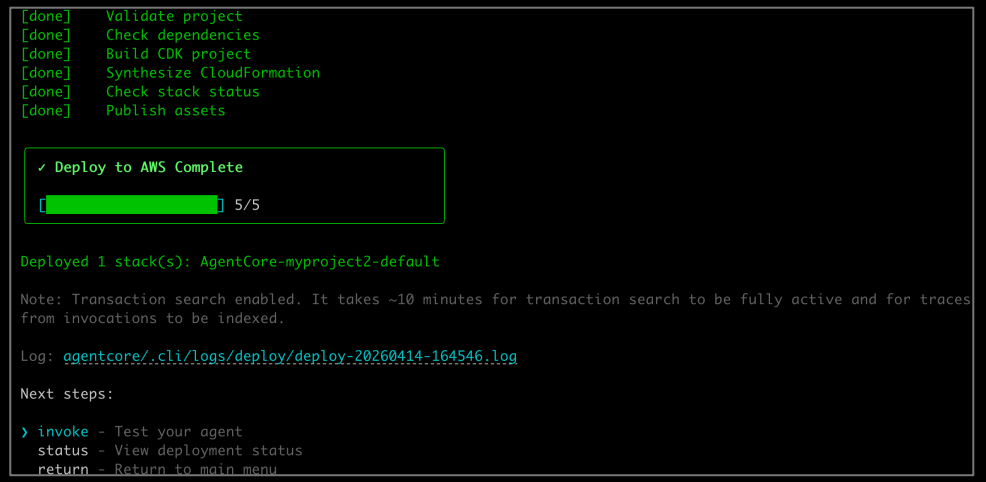


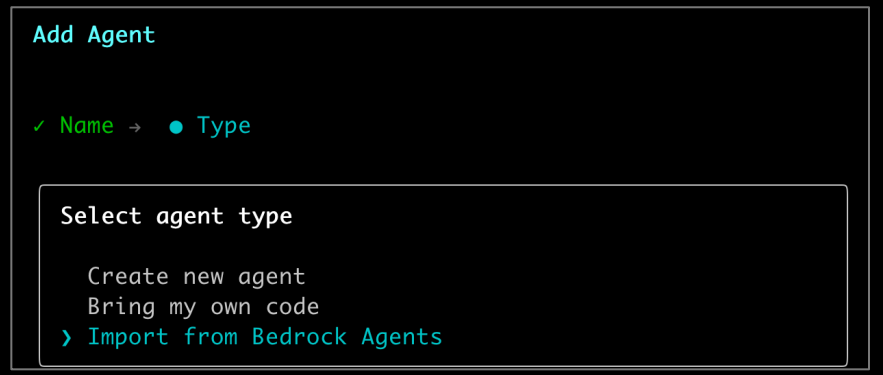
エージェント起動の基本
Amazon Bedrock AgentCore Runtime



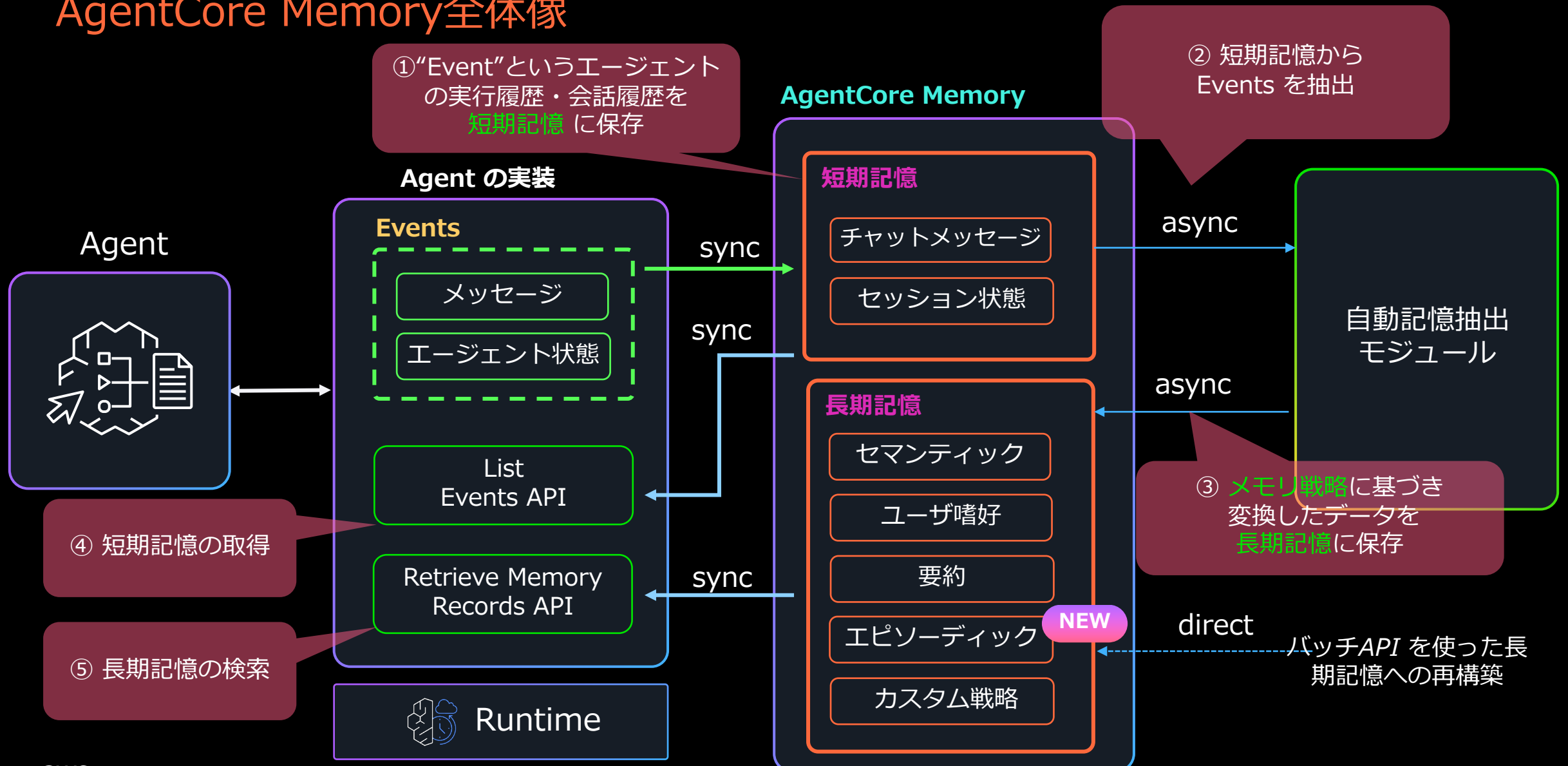
(参考) AgentCore CLI

- AgentCore CLIは、より簡単に、より素早くAgentの生成フローを実行するためのCLIおよびTUI(Terminal User Interface)を提供
- 以下主要なコマンドにより、ゼロから数分でエージェントの起動を可能にする
  - agentcore create – デフォルト設定で即実行可能な新規エージェントプロジェクトを自動生成 (複数のモデル、エージェントフレームワーク、プロトコルの選択、メモリ利用有無)
  - agentcore dev – Starts local development loop with hot-reload
  - agentcore deploy – Deploys agent to AgentCore
- Bedrock Agentsからの移行も可能
  - 移行ターゲットとなるエージェントフレームワークを選択
  - 既存のツールやナレッジベースを活用





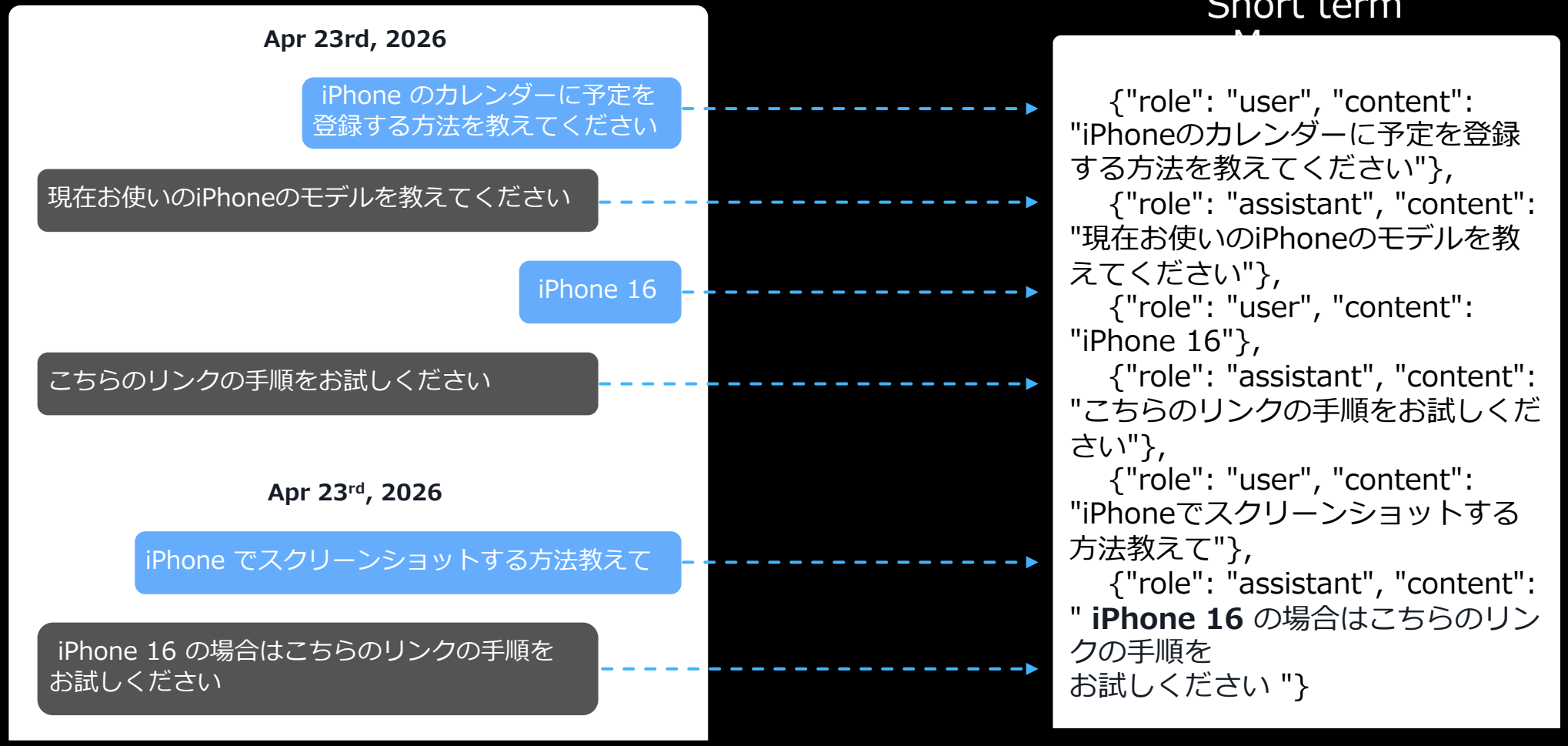
エージェント起動の基本
AgentCore Memory全体像



エージェント起動の基本

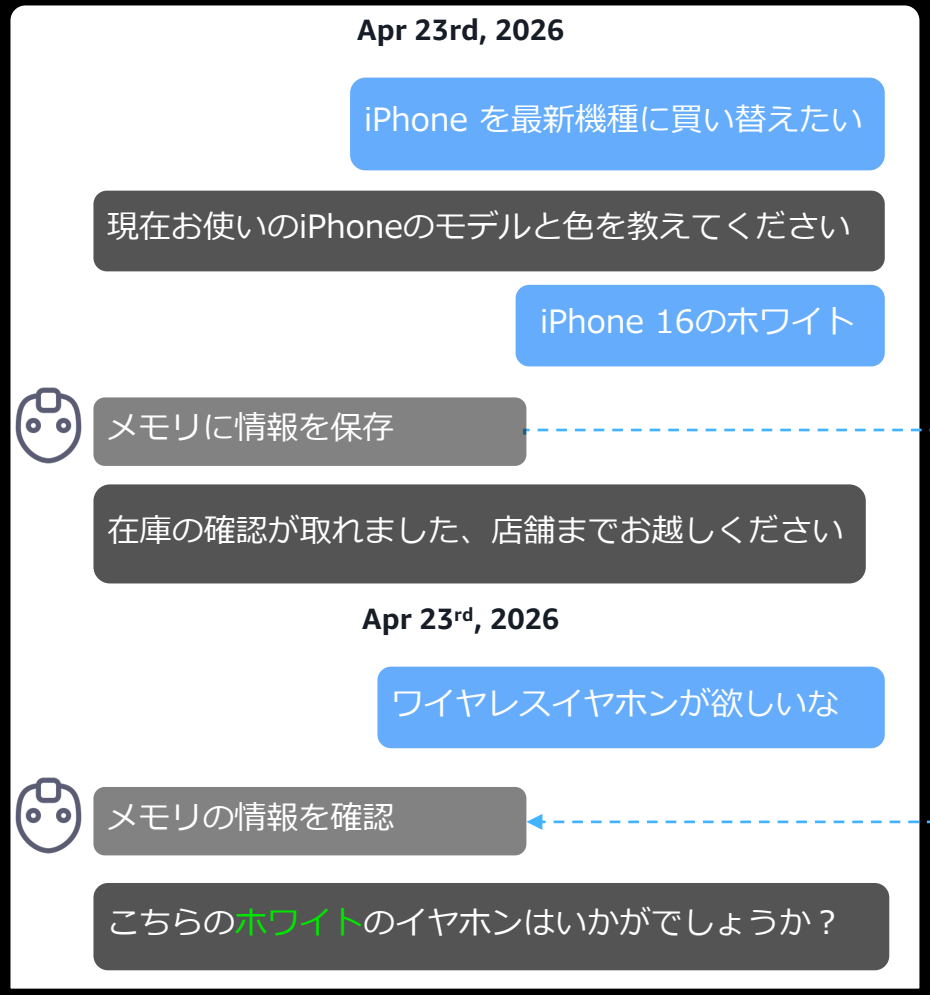
AgentCore Memory Short-term Memory概要

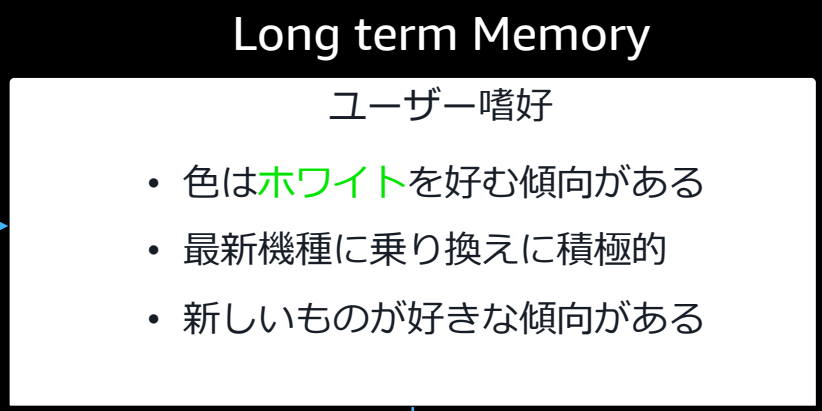
ユーザーの入力、Agentの実行・会話履歴などの生データを保存しコンテキストの維持、中断した会話の再開を可能にする



エージェント起動の基本
AgentCore Memory Long-term Memory概要

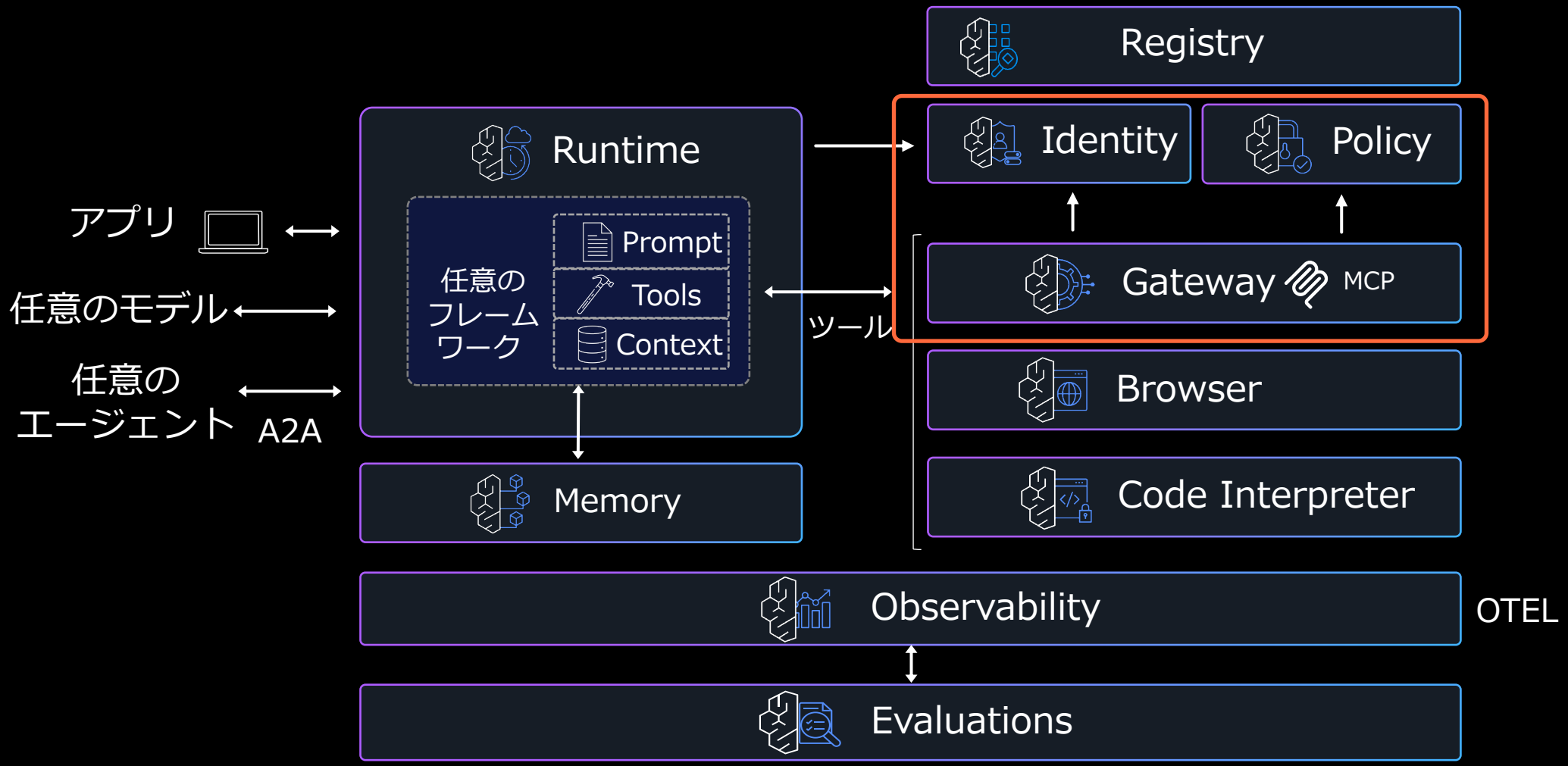
ユーザーの好みや傾向、会話の要約、会話から得られた事実や関係を保存する機能





Amazon Bedrock AgentCore プラットフォーム

エージェントアクションの制御

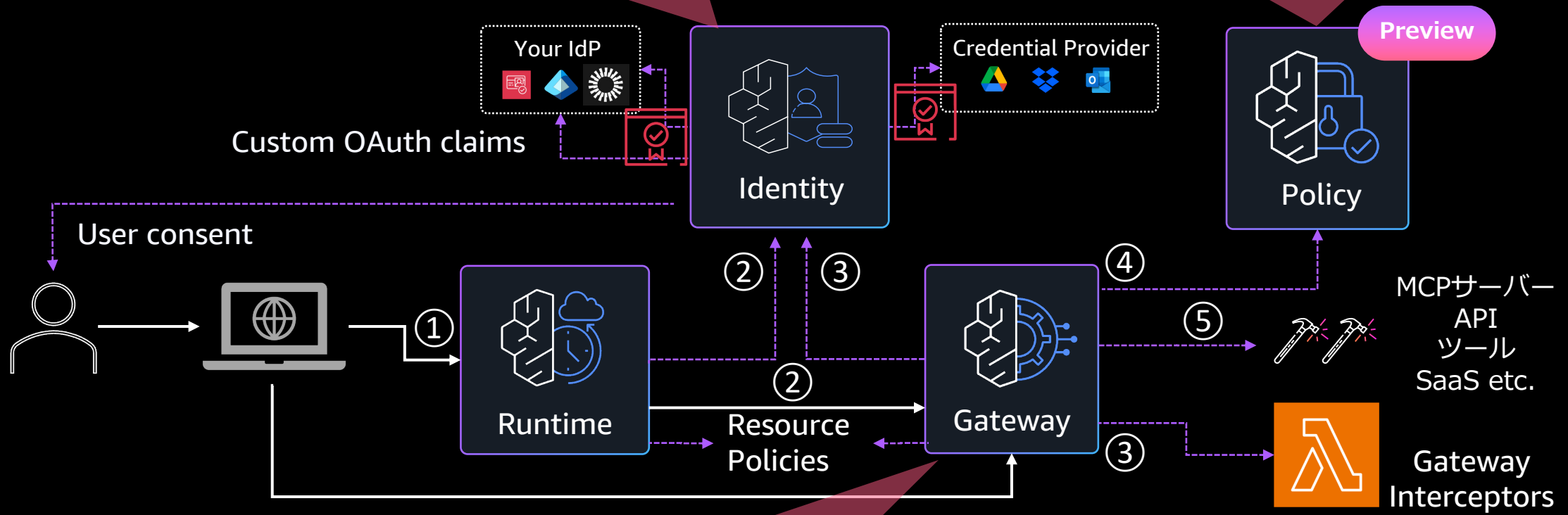


エージェントアクションの制御

Amazon Bedrock AgentCore Identity/Gateway/Policy

AIエージェントにおける適切なIDや権限管理を行いエージェントからツールへのアクセスを安全に制御

AgentCore Gatewayと統合した動的なポリシーの評価、自動Cedar変換により自然言語を使用してアクセスポリシーを作成・管理

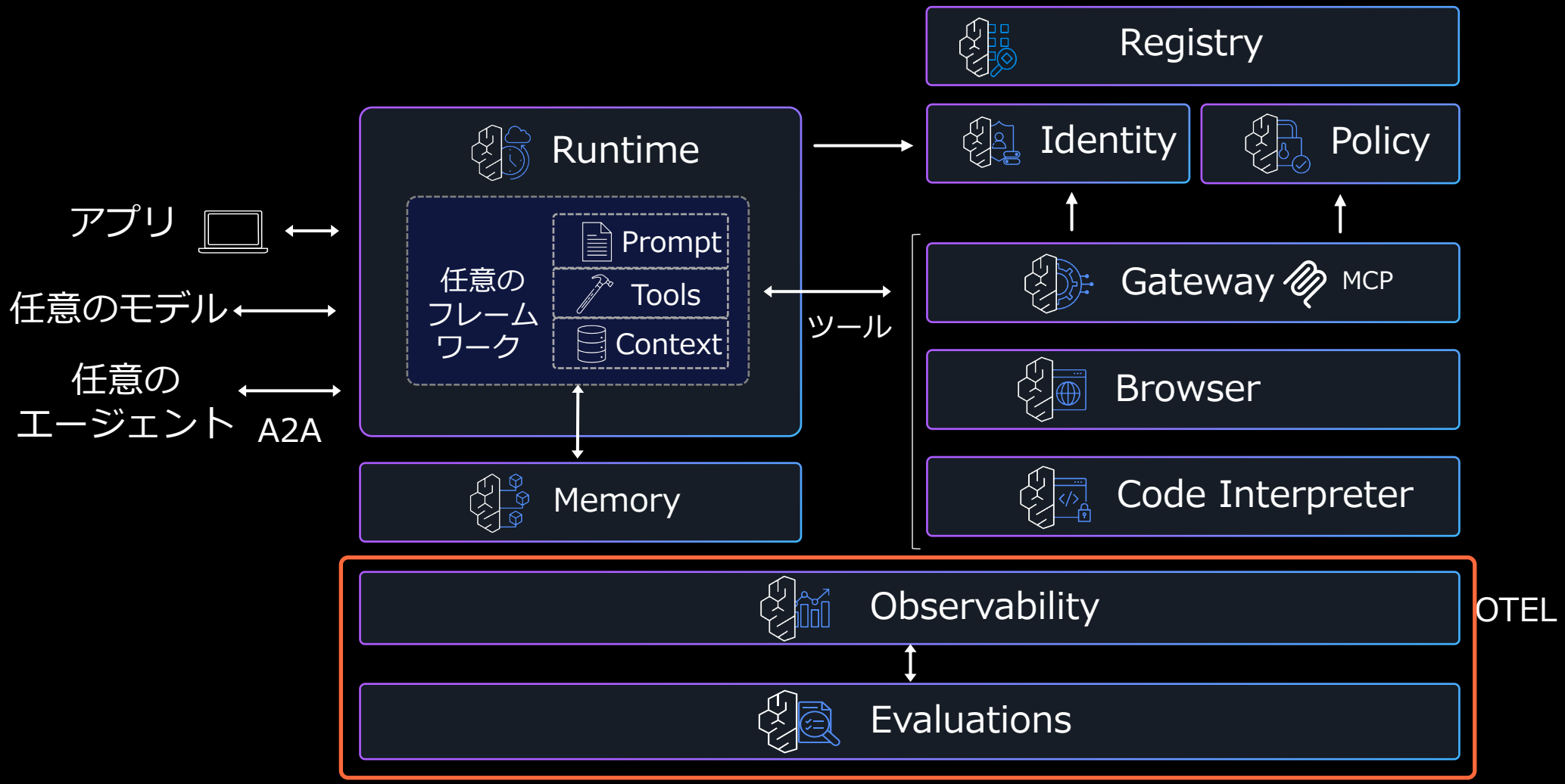


開発者がツールを大規模に「構築・デプロイ・発見・接続」するための、簡便かつ安全な手段

リクエスト/レスポンス2つのインターセプターにより、きめ細かなアクセス制御、ツール検出の動的なフィルタリング、カスタムヘッダーの伝播を実装

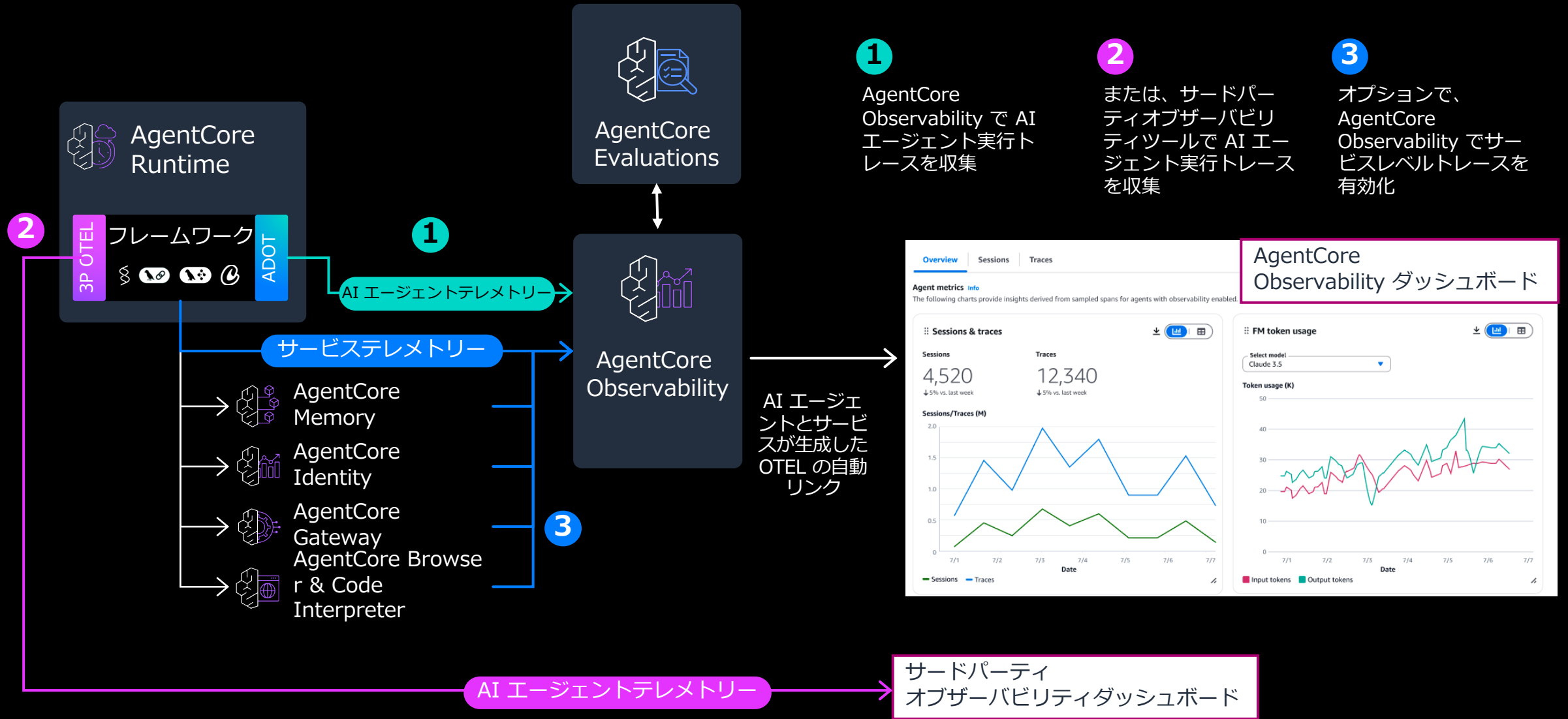
Amazon Bedrock AgentCore プラットフォーム

エージェントのモニタリングと評価



エージェントのモニタリングと評価

Amazon Bedrock AgentCore Observability

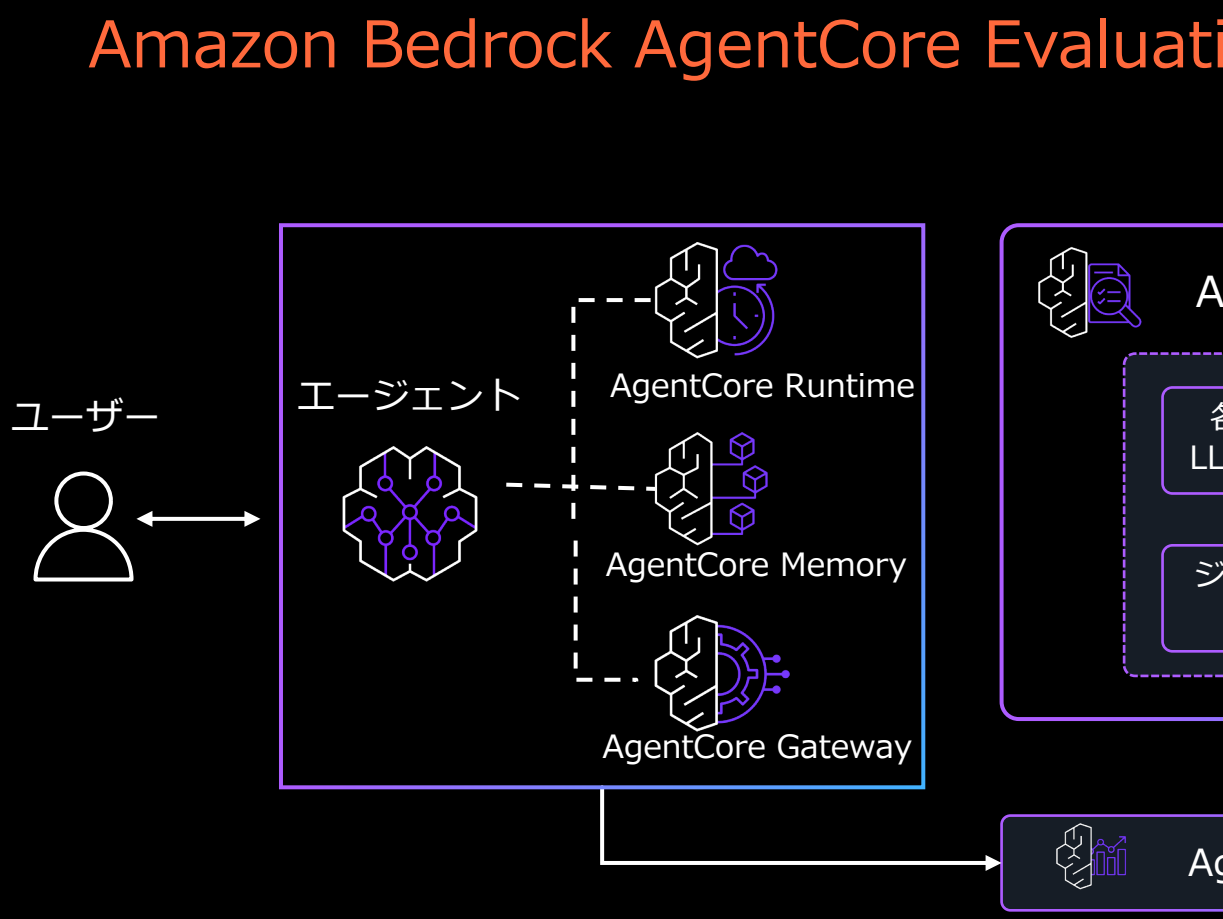


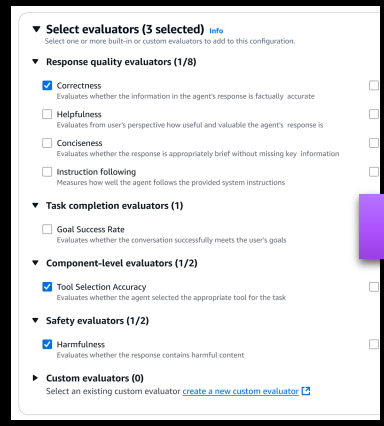
1 AgentCore Observability で AI エージェント実行トレースを収集

2 または、サードパーティオブザーバビリティツールで AI エージェント実行トレースを収集

3 オプションで、AgentCore Observability でサービスレベルトレースを有効化

エージェントのモニタリングと評価
Amazon Bedrock AgentCore Evaluations

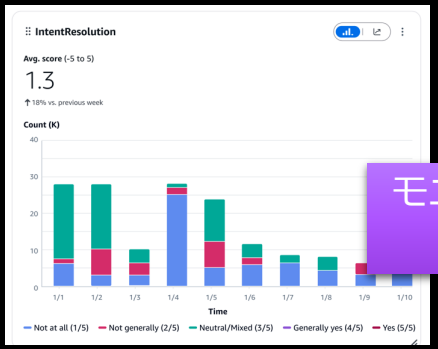




設定

オンデマンド

エージェント
オーナー



モニタリング
と評価

品質の継続的な検証
実世界での振る舞いは
商用環境で評価

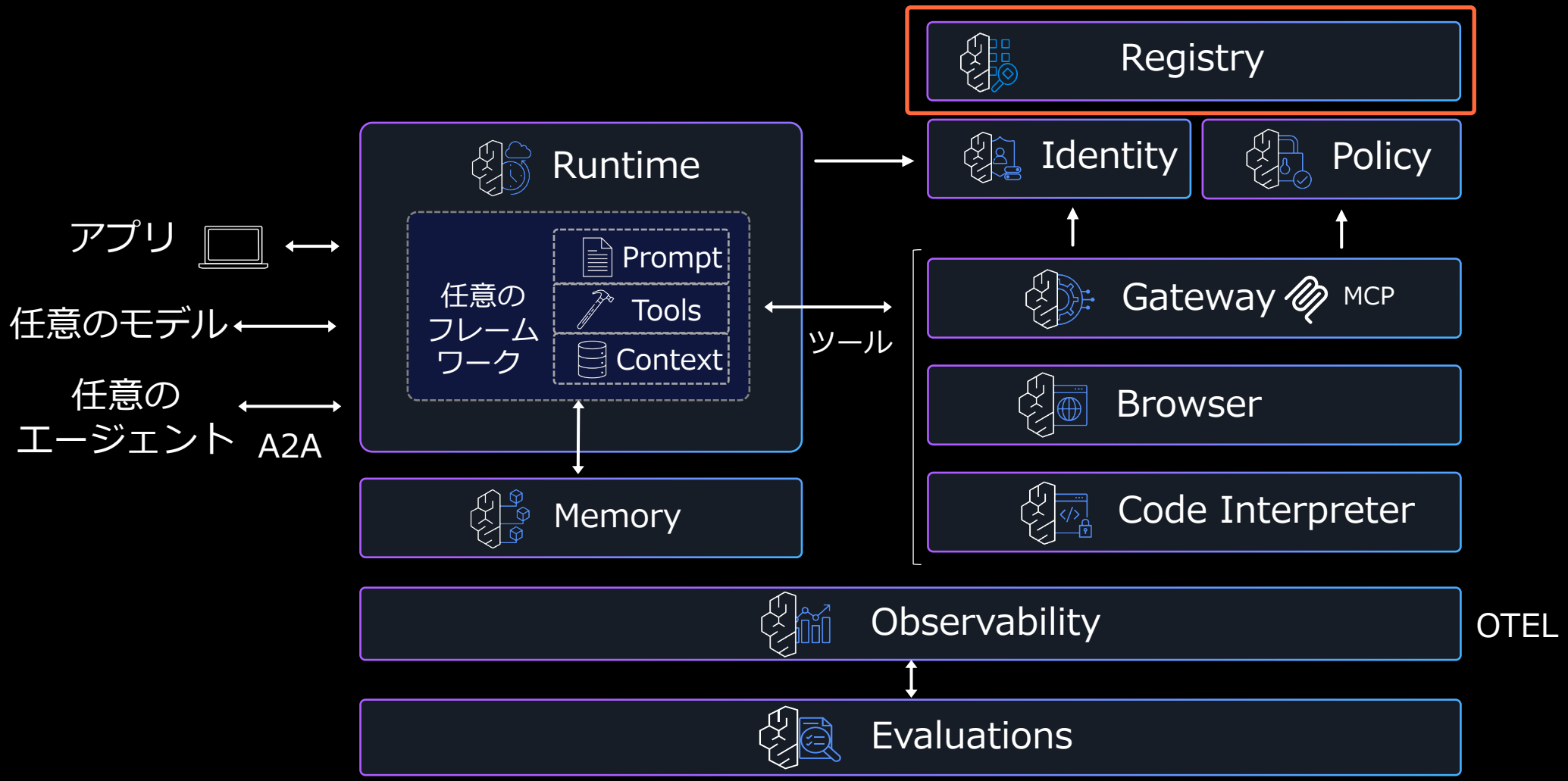
事前定義とカスタム
事前に定義された評価と
カスタマイズされた評価

オンデマンドでの評価
本番稼働前に検証し、
スムーズな適用を実現

オンラインでの評価
本番環境でのリアルタイム評価
品質低下、サイレント障害検知

Amazon Bedrock AgentCore プラットフォーム

エージェント・ツールの統一管理



エージェント・ツールの統一管理

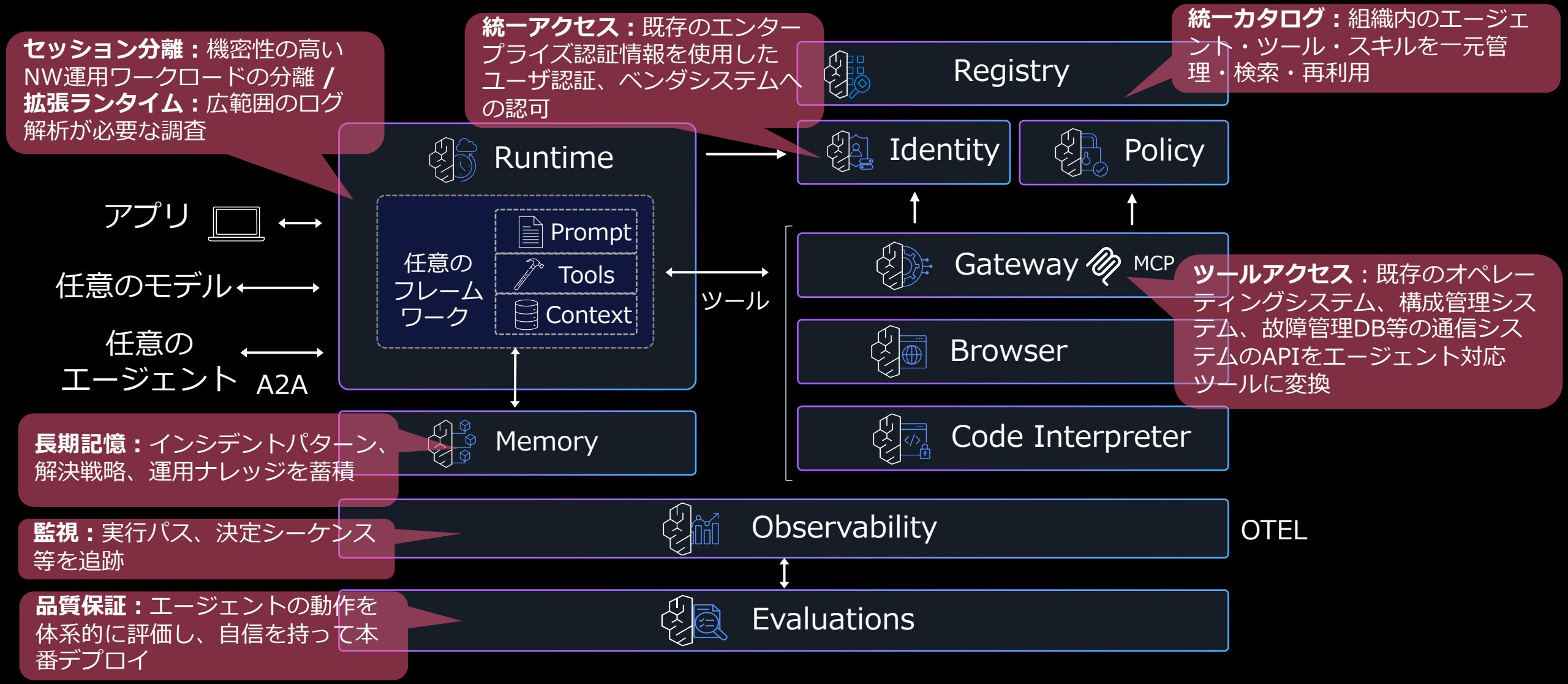
AWS Agent Registry

Preview

Amazon Bedrock AgentCoreの一部として提供される統制付きカタログ。これを中心として Agent Sprawl 問題を解消

ペルソナ	対象者	主なタスク
Admin (管理者)	IT管理者 DevOpsエンジニア	レジストリ管理, 承認ワークフロー設定, セキュリティポリシー
Publisher (公開者)	エージェント開発者 MCP/ツール提供者	レコード作成, 承認申請, CI/CDパイプライン統合
Consumer (利用者)	エージェント開発者 Autonomousエージェント	セマンティック検索, アクセス権限取得, IDE統合

通信ネットワークにおける価値



Thank you!