Strands Agents ハンズオン
- AI エージェント開発を簡素化しよう –

2025/11/27

Solutions Architect
Kenshiroh Kamiya

自己紹介

神谷 拳四郎（Kamiya Kenshiroh）

通信業界担当 Solutions Architect

Favorites

- AWS サービス：Amazon Bedrock、Amazon CloudFront
- 趣味：スポーツ観戦（野球/バレー）、ゲーム、ガジェット
- ひとこと：カービィのエアライダー漬けの毎日です

Agenda

Strands Agents は、わずか数行のコードで AI エージェントを構築・実行するモデル駆動型アプローチを採用したオープンソース SDK です。ハンズオンを通じてStrands Agents の機能を探索し、この SDK が AI アプリケーション構築へのアプローチをどのように変革できるかを学びましょう。

- Strands Agents とは （20分）
- ハンズオン（60分）

【再掲】 Autonomous Network 参考アーキテクチャ

AI エージェントが各データを取得して自律的に動作

【再掲】① Strands Agents

Strands Agent とは

【再掲】 AI エージェントの定義

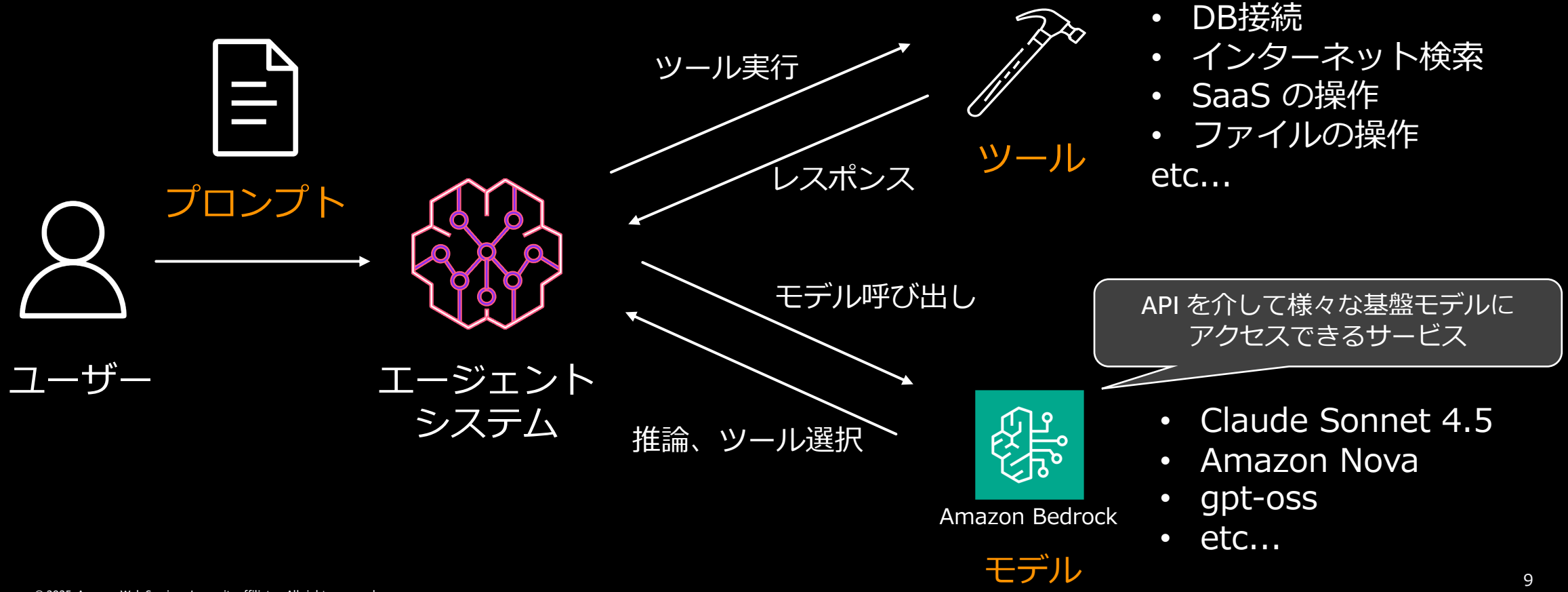
人工知能 (AI) エージェントは、環境と対話し、
データを収集し、そのデータを使用して
自己決定タスクを実行して、
事前に決められた目標を達成するための
ソフトウェアプログラム

Ref. https://aws.amazon.com/jp/what-is/ai-agents/

AI Agent 実装時の課題

AI エージェントの構成要素

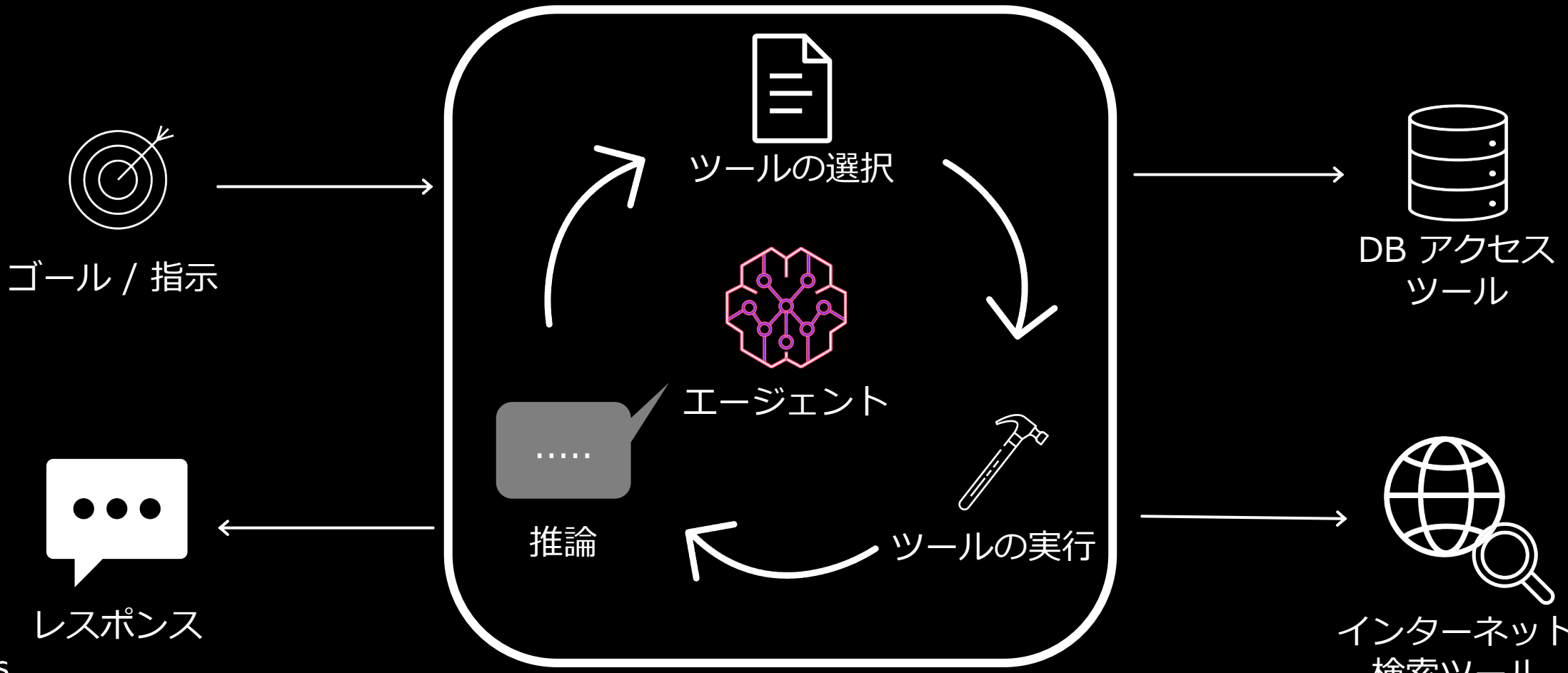
ユーザーからの指示に対して、自律的に計画を立て、ツールを利用してタスクを実行する



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

エージェントループ

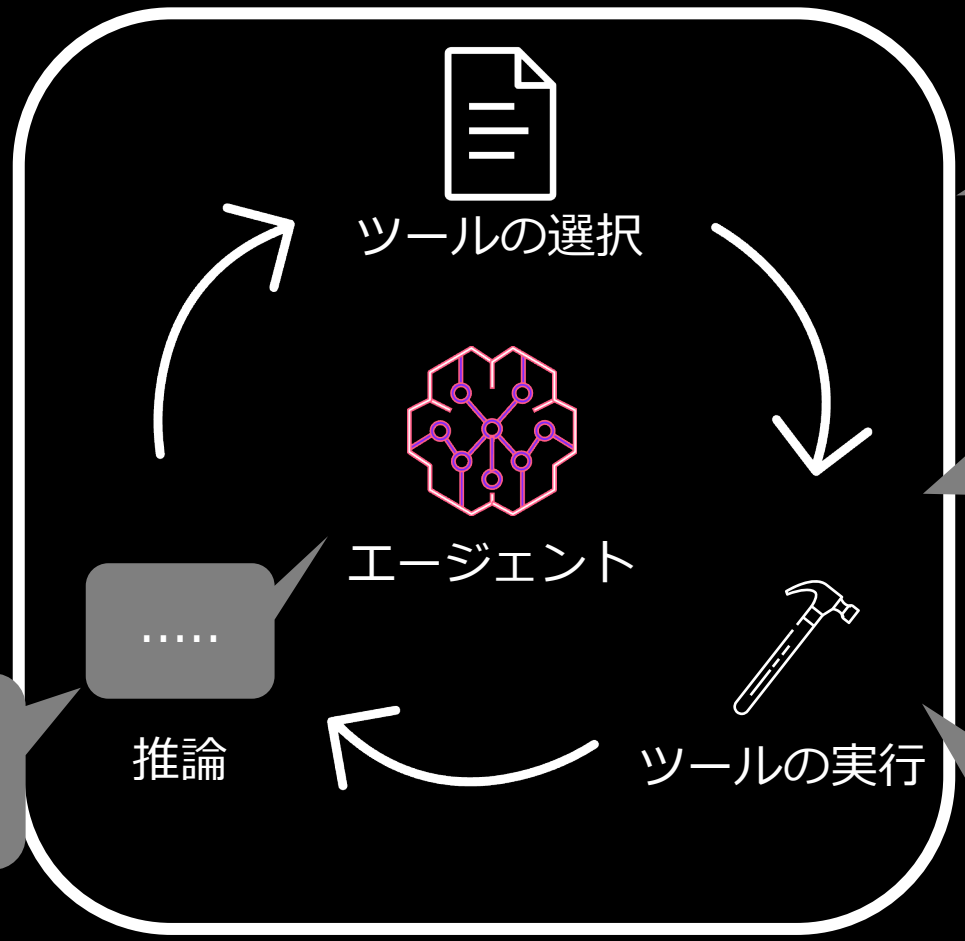
エージェントが自律的にタスクを実行できるようにするための背景技術



ref. https://strandsagents.com/latest/documentation/docs/user-guide/concepts/agents/agent-loop/

AI エージェントを実装する際のハードル

AI エージェントを自力で実装するには多くの困難が存在する



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？

DB アクセス
ツール

モデルの変更に
どう対応する？

生成AI とツール間の
インターフェースをどう
実装する？

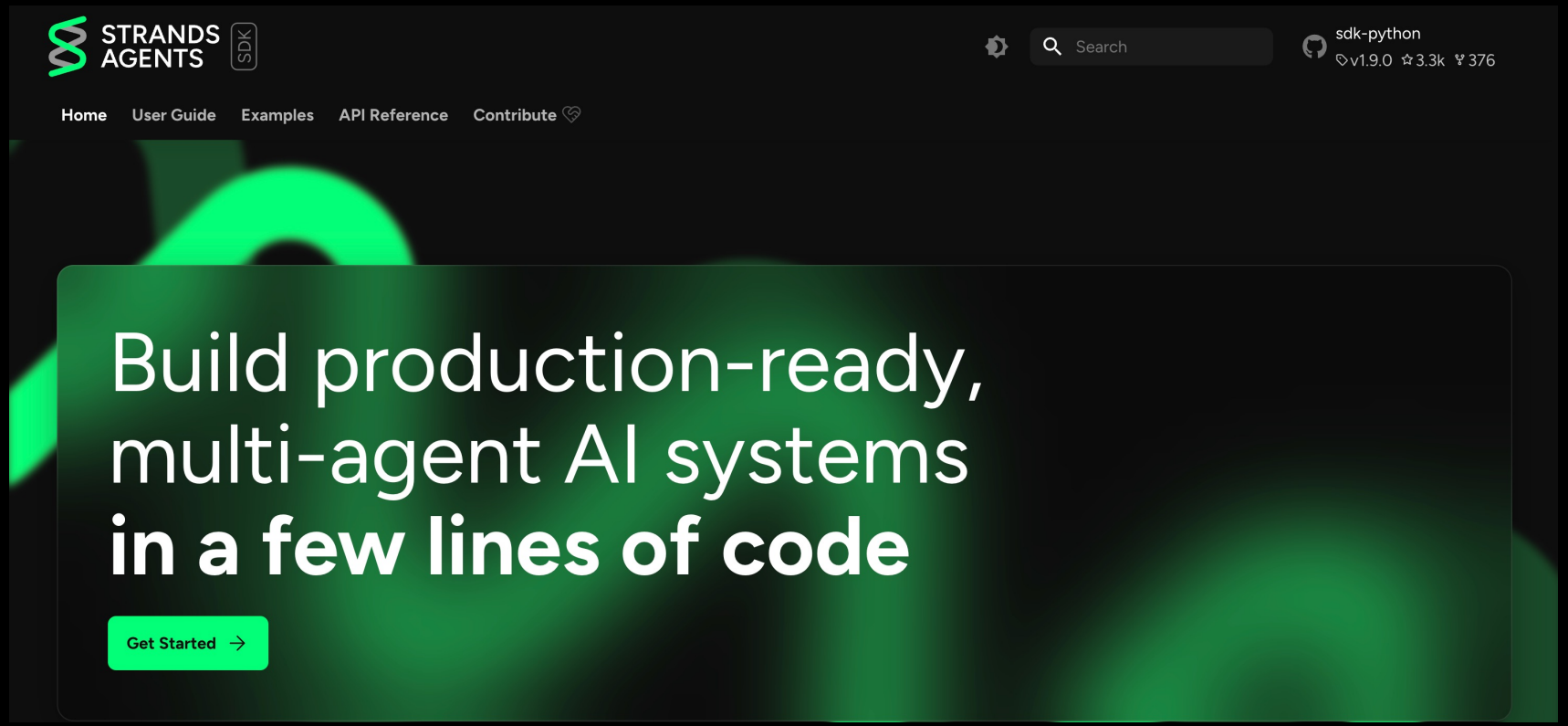
検索ツール

ref. https://strandsagents.com/latest/documentation/docs/user-guide/concepts/agents/agent-loop/

Strands Agents のご紹介

Strands Agents とは

わずか数行のコードで実装できるオープンソース AI エージェント開発用 SDK

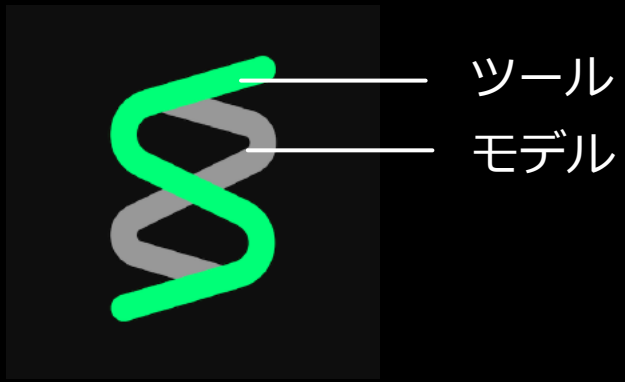


https://strandsagents.com/latest/

Strands ?

DNAの構成要素である
らせん構造の鎖のこと

エージェントの主要構成要素である「モデル」と「ツール」が結び合わさることが由来



【再掲】AWS の AI エージェントサービス

要件に合わせたエージェントの構築とデプロイの広い選択肢

高 フルマネージド 低

低 カスタマイズ性 高

ビジネスユーザーが作成

Amazon Quick Suite

構築済みのDeep Research機能やカスタマイズエージェントをGUIで作成する機能

開発者が簡単に作成

Amazon Bedrock Agents

組み込みの基盤モデルのオーケストレーション

完全にカスタムなエージェントを実装

Strands Agents

エージェント構築用のシンプルで柔軟で軽量なオープンソース SDK

x

Amazon Bedrock AgentCore

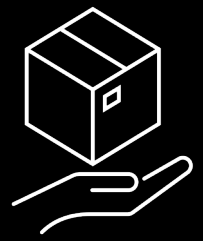
任意のフレームワーク・モデルを使ったAI エージェントを安全・大規模に運用

サーバーレス：インフラを意識しない

コンテナなど Compute リソース上で動作

Strands Agents の特徴

軽量かつシンプル



シンプルな
エージェント
ループがそのまま
動作し、完全に
カスタマイズ可能

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
（可観測性）、
トレーシング、
デプロイ手段を備
える

モデル/プロバイダ/デプロイ非依存

Amazon Bedrock、
Anthropic、
Gemini、OpenAI
など、さまざまな
プロバイダの
多様なモデルを
サポート

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

※抜粋

ref. https://strandsagents.com/latest/documentation/docs/

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。
```

※特に設定しない場合、Amazon Bedrock の Claud Sonnet 4が基盤モデルとして利用されます

ref. https://strandsagents.com/latest/documentation/docs/

柔軟なモデルの選択

Amazon Bedrock で利用可能なモデルの他、様々なモデルを
引数を変えるだけで簡単に選択可能

```python
from strands import Agent
from strands.models import BedrockModel

# Amazon Bedrock 内で利用できるモデルを指定
bedrock_model = BedrockModel(
    model_id="amazon.nova-pro-v1:0",
    region_name="us-east-1",
    temperature=0.3,
)

# デフォルト設定でエージェントを作成する
agent = Agent(model=bedrock_model)
```

model_id を変更するだけで、Bedrock で利用可能な基盤モデルを柔軟に変更可能

ref. https://strandsagents.com/latest/documentation/docs/user-guide/concepts/model-providers/amazon-bedrock/

柔軟なモデルの選択

モデルプロバイダー各社の API 呼び出しも簡単に実装可能

```python
from strands import Agent
from strands.models.openai import OpenAIModel

model = OpenAIModel(
    client_args={
        "api_key": "<KEY>",
    },
    # **model_config
    model_id="gpt-4o",
    params={
        "max_tokens": 1000,
        "temperature": 0.7,
    }
)

agent = Agent(model=model)
```

OpenAI のほか、Anthoripic、LiteLLM、Llama API、Ollama、SageMaker、Gemini、Writerなど主要なプロバイダー各社の API 呼び出しをサポート

ref. https://strandsagents.com/latest/documentation/docs/user-guide/concepts/model-providers/openai/

ツールの使用

エージェントの定義にツールをリスト形式で渡すだけで、
ツールを持ったエージェントを作成可能

```python
from strands_tools import calculator
from strands import Agent

# エージェントに計算機ツール（calculator）を渡す
agent = Agent(tools=[calculator])

# エージェントに対して問い合わせる
agent("25+25を5で割った数はいくつ？")
```

Strands Agents SDK の
組み込みツールである計算機
（calculator）を使用

calculator が必要なタスクと
判断し、ツールを実行

```
日本語の問題を解いてみますね。「25+25を5で割った数」を計算します。
Tool #1: calculator
答えは **10** です。

計算の流れ：
1. まず 25 + 25 = 50
2. 次に 50 ÷ 5 = 10

したがって、25+25を5で割った数は10になります。
```

計算機を使った正確な
出力を実現

ref. https://strandsagents.com/latest/documentation/docs/user-guide/concepts/tools/tools_overview/

Strands Agents の組み込みツール

典型的なのツールを 20 以上標準で組み込み
import 文一つで AI エージェントにアタッチ可能

ファイル操作

file_read, file_write

ウェブ検索

tavily_search, exa_search

RAG

retrieve

シェル統合

shell, cron

コード実行

code_interpreter

パソコン操作

browser, use_computer

メモリ

mem0_memory, agent_core_memory

AWS 連携

use_aws

画像・動画生成

generate_image_stability, generate_image

ref. https://github.com/strands-agents/tools

自作ツールの使用

@tool デコレータを使用することで、既存の関数を簡単にツール化可能
関数内でコメントでツールの情報を記述することでコンテキストに取り込み

```python
from strands import Agent, tool

@tool
def letter_counter(word: str, letter: str) -> int:
    """
    与えられた文字列中の指定した文字の数をカウントする。

    Args:
        word (str): 走査対象の文字列
        letter (str): カウントしたい文字

    Returns:
        int: 与えられた文字列中の指定した文字の数
    """
    return word.lower().count(letter.lower())

agent = Agent(tools=[letter_counter])
```

ツールの説明

引数と返り値の説明

※これ以外にも自由記述、jsonによるツールの記述も可能

ref. https://strandsagents.com/latest/documentation/docs/user-guide/concepts/tools/python-tools/

自作ツールの使用

@tool デコレータを使用することで、既存の関数を簡単にツール化可能
関数内でコメントでツールの情報を記述することでコンテキストに取り込み

```python
    return word.lower().count(letter.lower())

agent = Agent(tools=[letter_counter])
# agent = Agent()
message = """
「となりのきゃくはよくかきくうきゃくだ」という文章の中に「き」が何文字あるか？
"""
agent(message)
```

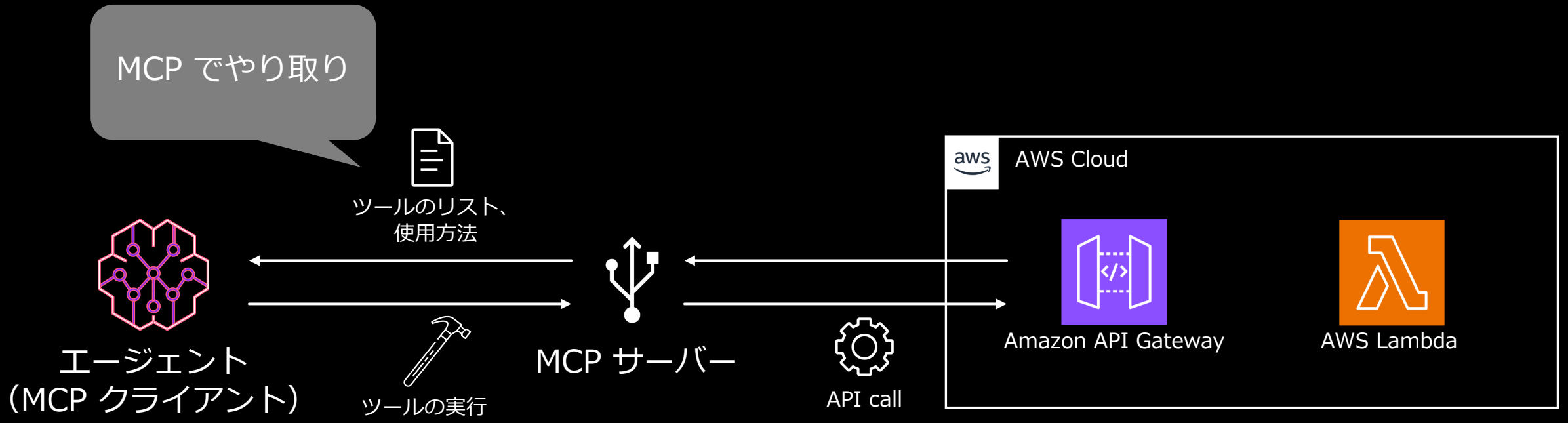
コメントの通り、
正しくツールを利用している

```
「となりのきゃくはよくかきくうきゃくだ」という文章中の「き」の文字数を数えてみますね。
Tool #1: letter_counter
「となりのきゃくはよくかきくうきゃくだ」という文章中に「き」は**3文字**あります。

具体的には：
- 「となりの**き**ゃくは」の「き」
- 「よくか**き**くう」の「き」
- 「くうき**き**ゃくだ」の「き」

の3箇所に「き」が含まれています。
```


ref. https://strandsagents.com/latest/documentation/docs/user-guide/concepts/tools/python-tools/

MCP：Model Context Protocol とは？

外部のAPIなどを AI エージェントが使える形に変換するための仕組み



ref. https://strandsagents.com/latest/documentation/docs/user-guide/concepts/tools/mcp-tools/

MCP ツールの利用

内外の MCP サーバーと容易に連携可能

```python
from mcp import stdio_client, StdioServerParameters
from strands import Agent
from strands.tools.mcp import MCPClient

# MCPサーバーへの接続
stdio_mcp_client = MCPClient(lambda: stdio_client(
    StdioServerParameters(
        command="uvx",
        args=["awslabs.aws-documentation-mcp-server@latest"]
    )
))

# MCP ツールを持ったエージェントを定義
with stdio_mcp_client:
    # MCP サーバーからツールを取得
    tools = stdio_mcp_client.list_tools_sync()
    print(tools)

    agent = Agent(tools=tools)
    agent("Amazon EC2のユーザーデータはどういう機能ですか？")
```

stdio トランスポート※で
MCPサーバーと接続

awsのドキュメントの検索・
閲覧が可能なMCPサーバー

※クライアントがMCPサーバーをローカルで立ち上げ、
標準入出力でデータをやり取りする方式

ref. https://modelcontextprotocol.io/specification/2025-06-18/basic/transports

MCP ツールの利用

内外の MCP サーバーと容易に連携可能

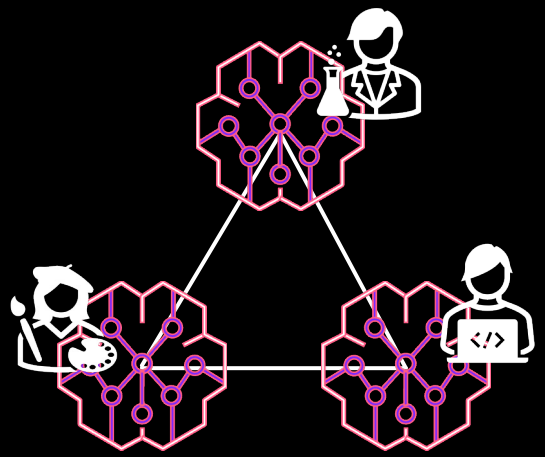
```
[09/29/25 19:41:29] INFO     Processing request of type ListToolsRequest
[<strands.tools.mcp.mcp_agent_tool.MCPAgentTool object at 0x1190df230>, <strands.tools.mcp.mcp_agent_tool.MCPAgentTool object at 0x11905
MCPAgentTool object at 0x11905b390>]
Amazon EC2のユーザーデータについて詳しく調べてみます。
Tool #1: search_documentation
[09/29/25 19:41:32] INFO     Processing request of type CallToolRequest
[09/29/25 19:41:33] INFO     HTTP Request: POST https://proxy.search.docs.aws.amazon.com/search?session=fff50eed-6e97-477a-b125-00863ce5
                             200 OK"
最初の結果が最も関連性が高そうなので、詳細を確認します。
Tool #2: read_documentation
[09/29/25 19:42:45] INFO     Processing request of type CallToolRequest
                    INFO     HTTP Request: GET
                             https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-metadata.html?session=f
                             ff50eed-6e97-477a-b125-00863ce570c3 "HTTP/1.1 200 OK"
ユーザーデータについてより具体的な情報を取得するため、専用のページを確認しましょう。
Tool #3: search_documentation
[09/29/25 19:42:55] INFO     Processing request of type CallToolRequest
[09/29/25 19:42:57] INFO     HTTP Request: POST
                             https://proxy.search.docs.aws.amazon.com/search?session=fff50eed-6e97-477a-b125-00863ce5
                             70c3 "HTTP/1.1 200 OK"
```

MCP サーバーを利用して AWS ドキュメントから情報収集

ドキュメントを踏まえた回答の生成

```
Amazon EC2のユーザーデータについて詳しく説明いたします。

## Amazon EC2ユーザーデータとは

**ユーザーデータ**は、EC2インスタンスを起動する際に渡すことができるデータで、イン
たタスクを実行できます。

## 主な機能と特徴

### 1. **自動化されたインスタンス設定**
- インスタンス起動時に自動的にソフトウェアのインストールや設定を実行
- Webサーバー、データベース、アプリケーションなどの環境構築を自動化
- システムの初期設定やユーザー作成などを自動実行

### 2. **対応する形式**
**Linuxインスタンス：**
- シェルスクリプト（例：bash、sh）
```

その他の Strands Agents の機能



マルチエージェントの実装

- A2Aプロトコルのサポートによる、マルチエージェントワークフローの実装
- ツールとしてのエージェントの公開

ref.
https://strandsagents.com/latest/documentation/docs/user-guide/concepts/multi-agent/agent-to-agent/

セキュリティ、安全性の確保

- Guardrail の組み込みをサポートし、責任のあるAIを実現
- LLM Guard 等の3rd Party ツールと連携したPIIリダクション

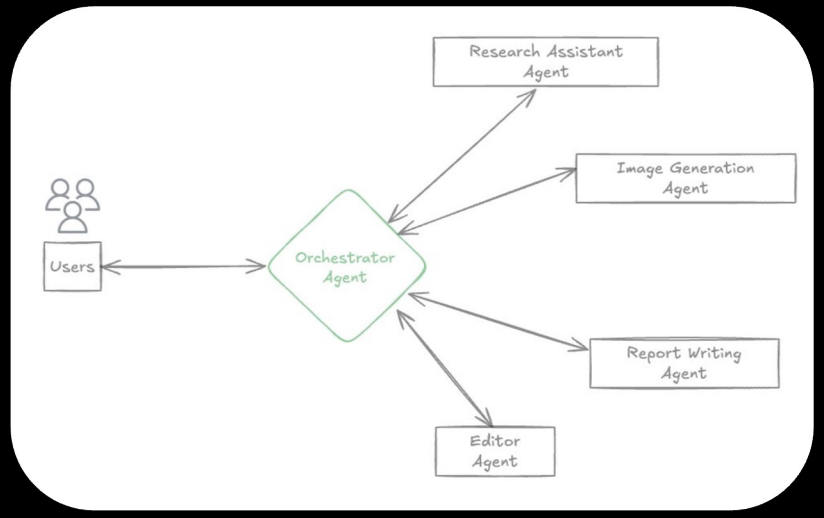
ref.
https://strandsagents.com/latest/documentation/docs/user-guide/safety-security/responsible-ai/

オブザーバビリティ

- オブザーバビリティを SDK に直接統合
- モデルとのインタラクション、推論ステップ、ツール使用などAI 特有のシグナルも含む各種テレメトリを取得・監視

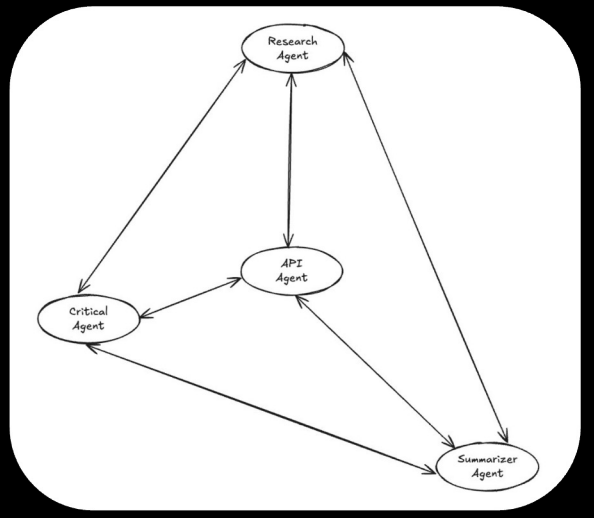
ref.
https://strandsagents.com/latest/documentation/docs/user-guide/observability-evaluation/observability/

Strands の Multi-agent パターン 1/2



Agents as Tools pattern

- オーケストレーターエージェントが専門エージェントをツールとして呼び出し、各結果を統合して最終回答を生成する階層型パターン
- 明確に分離可能なタスク、異なる入力形式を扱うタスクに向く

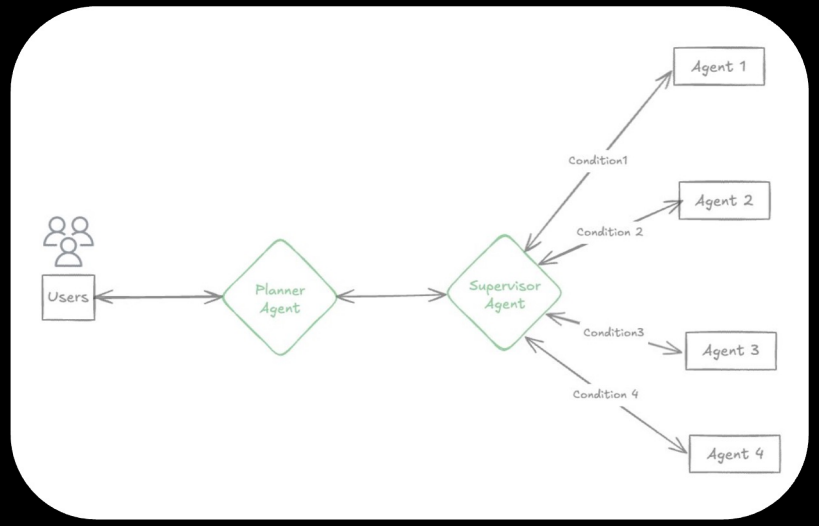


Swarm pattern

- 複数のエージェントが対等に協力しながら問題解決する分散型パターン
- 多様な視点やアプローチが価値を生むタスクに向く

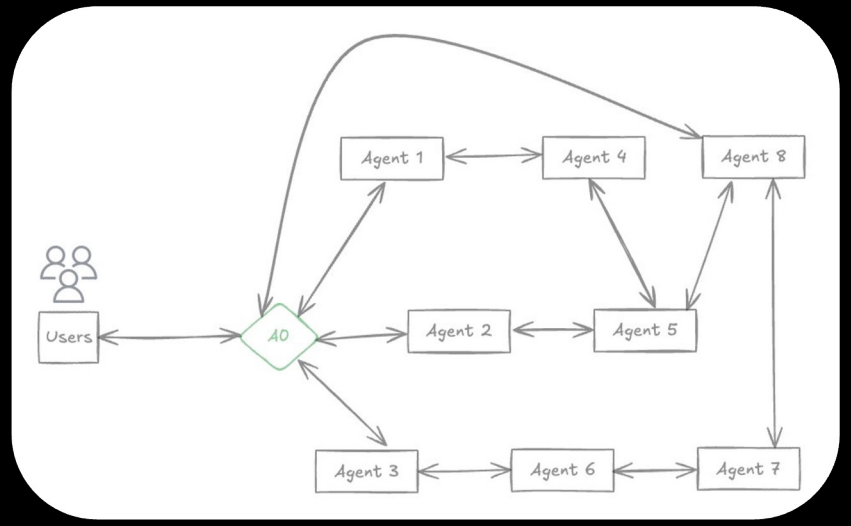
ref. https://aws.amazon.com/jp/blogs/machine-learning/multi-agent-collaboration-patterns-with-strands-agents-and-amazon-nova/

Strands の Multi-agent パターン 2/2



Graph pattern

- エージェントを有向グラフで接続し、データの流れる経路と方向を明示制御するパターン
- 複雑な多段階の意思決定プロセスがあるタスクに向く



Workflow pattern

- 事前定義された依存関係とシーケンスに従って実行し、各ステップを専門エージェントが処理するパターン
- タスクの依存関係が明確で、段階的な実行が重要な場合に向く

ref. https://aws.amazon.com/jp/blogs/machine-learning/multi-agent-collaboration-patterns-with-strands-agents-and-amazon-nova/

まとめ

- Strands Agent を利用することで Production-Ready な AI エージェントを容易に開発することが可能
- Strands Agent は AI エージェントに必須の機能であるツールと生成 AI をシンプルに統合し、様々な基盤モデルに対応、高い拡張性を持つ
- A2A プロトコルのサポートによるマルチエージェントワークフローの実装も可能

Thank you!

Thank you!