短期連載

AIエージェント同士で連携!

通信プロトコルAgent2Agent

第2回 Hello Worldで見るA2Aサーバ作りと通信

鶴 英雄

エージェント同士の協調動作と仕事の割り振りを円滑にするため，A2A（Agent 2 Agent）プロトコルという，エージェント間のやり取りに特化した標準化が検討されています．

今回はA2Aの最小サンプルとも言えるHello WorldをPythonで作成し，どのような作りになるのかを説明します（**図1**）．

Agentサーバ側のコード

● Agentサーバの役割

本Agentサーバは，クライアントに対して常にHello Worldを返すA2A Python SDKを使用したエージェントです（**リスト1**，13 ～ 19行目，22 ～ 34行目）．A2A SDKはVersion 0.3.12を使っています．また，エージェントの本体でTASKの処理を行う主体でinvokeで呼び出されます（14 ～ 18行目，33行目）．ここではHello Worldを返すだけでクライアントの要求を受け付けます（18行目）．

● 処理内容と対応機能

ここでは実行（excute），キャンセル（cancel）を実装します（28 ～ 39行目）．ただし，本エージェントはキャンセルをサポートしません（36 ～ 39行目）[注1]．HTTPサーバを司るクラスです（43 ～ 95行目）．

● 設定情報とサーバ構成

Agent Cardを用意して，クライアントからの呼び出しを受け付けます（50 ～ 70行目）．また，Agent Cardのcapabilitiesでサーバ側の能力を指定できます（67行目）．ここではstreaming応答が可能としています（67行目）．このstreamingにFalseを設定すると，クライアントがstreamingを指定しても通信はnon-streamingとなります．

Agent Cardは，/.well-known/agent-card.jsonにあるものとされますが，それはSDK側で処理してくれます（60 ～ 70行目）．サーバ本体としては，A2AStarletteApplicationクラスを利用しますが（78行目），ここでは受信データのダンプも可能にするため派生クラスLoggingA2AStarletteApplicationを作成しています（78 ～ 88行目）．

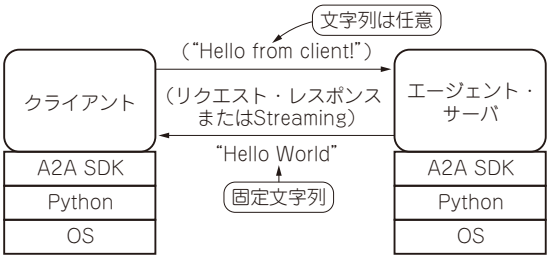

図1 Hello Worldサンプル動作イメージ

クライアント側のコード

● 事前準備と接続設定

クライアントでは，Agent Cardの取得が必要です．ここではA2A SDKのA2ACardResolverを利用してAgent Cardを取得します．次にA2Aサーバにアクセスするためにclientインスタンスを作成します（**リスト2**，5 ～ 8行目）．作成はClientFactoryで行います（5行目）．

● 通信方式の指定

サーバとのやり取りにstreamingを使用するかどうかはこの段階で指定しています（ClientConfigのstreamingパラメータ，1 ～ 4行目）．本クライアント・スクリプトの引数にstreamingを指定すると，streaming変数にTrueが設定されます（streamingを要望します）．streamingがなければstreaming変数はFalseとなります．

ここで指定はクライアント側の要望であって，実際

注1：A2Aエージェントとして必要な実行とキャンセルの入口は実装しているが，キャンセルは仕様上あえて非対応にしている．