Microsoft Azure

ARM - Azure Services で体験" で用いるソリューションのデプロイ

Azure Resource Manager (ARM) - ARM テンプレートとは

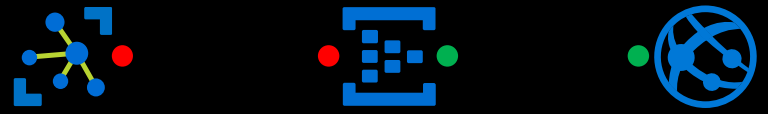
リソースを個別に処理するのではなく、全てのリソースをグループとしてデプロイ、管理、監視

- リソーステンプレートを仕様し、展開後の状態を事前に定義、一環した状態で展開
- 正しい順序でデプロイされるようにリソース間の依存性を定義、
- ロールベースのアクセス制御 (RBAC)
- タグを使用し、リソースを論理的に整理
- リソースグループ、またはタグ単位でコスト表示

ARM テンプレートでデプロイ

- 各種サービスインスタンスの作成
  - 各サービスの設定

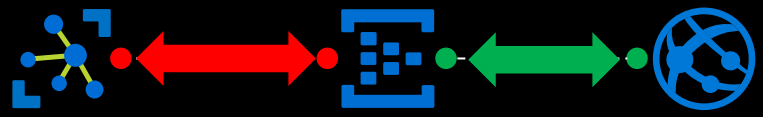
- データの入出力を定義
  - エンドポイント、メッセージルート作成等



- データフローの定義
  - 入出力データの前処理、後処理など

- アプリケーションをデプロイ
  - Github にあるソースコードから CICD

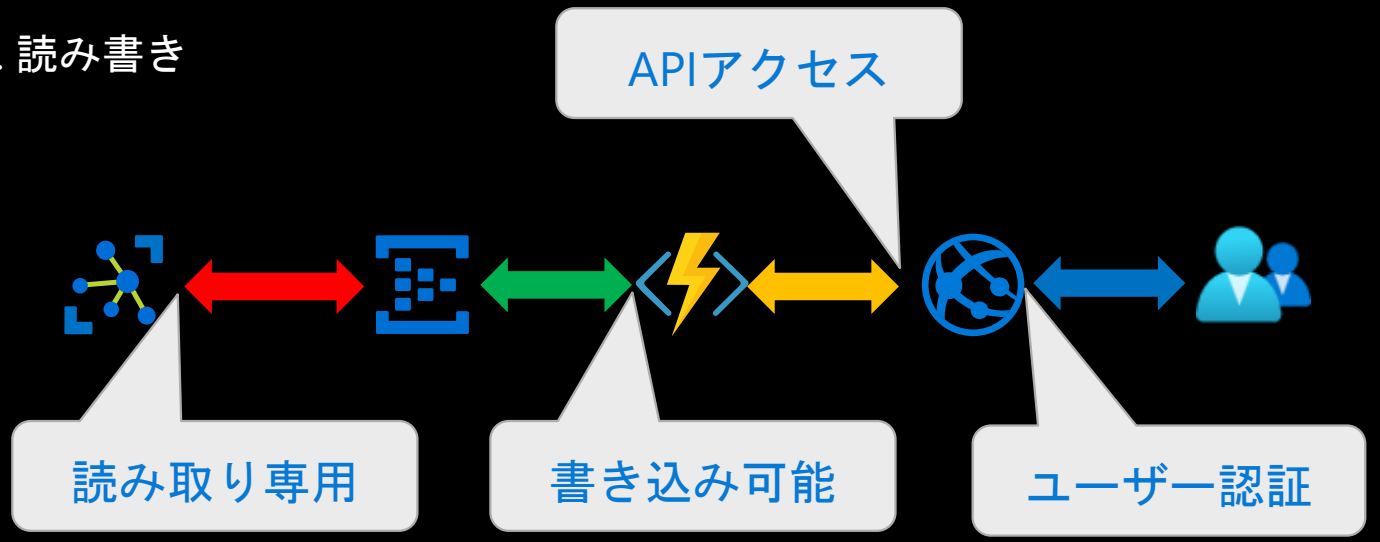
- サービス同士を接続
  - データが流れ、処理されるようになる



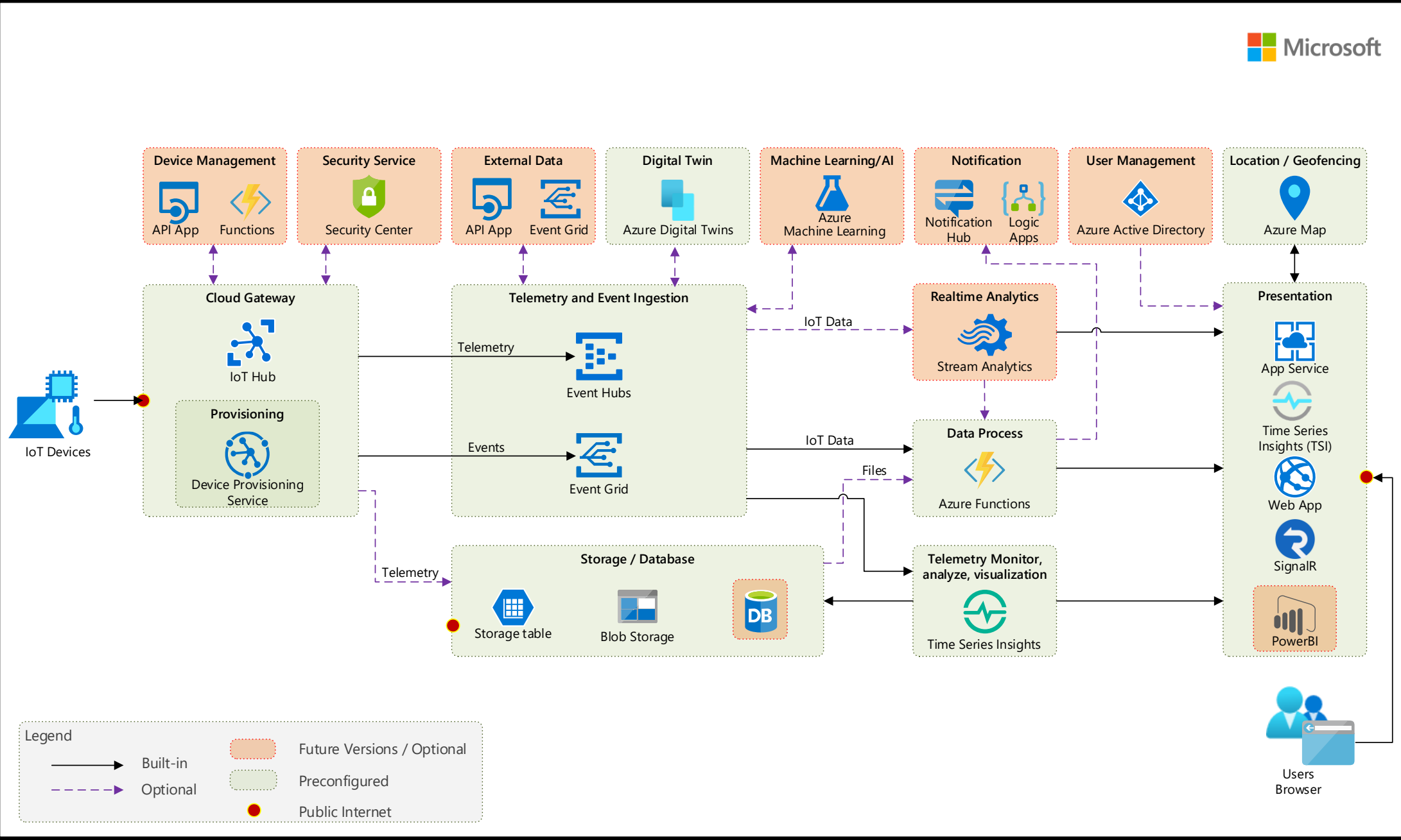
アクセス管理

各サービスの接続やユーザーがデータにアクセスする為にはアクセス権の設定が必要

- ユーザーレベルでのアクセス管理
  - 誰がウェブサイトにアクセスできるべきか？
- アクセスレベルの設定
  - 例：管理者 vs. 一般ユーザー、読み取り専用 vs. 読み書き
- サービス間の認証
  - どのサービスにデータアクセスを許可させるか



これらは Azure Active Directory の機能を使って管理するため、ARMテンプレートは不得意


Microsoft
Device Management
API App
Functions
Security Service
Security Center
External Data
API App
Event Grid
Digital Twin
Azure Digital Twins
Machine Learning/AI
Azure Machine Learning
Notification
Notification Hub
Logic Apps
User Management
Azure Active Directory
Location / Geofencing
Azure Map
Cloud Gateway
IoT Hub
Provisioning
Device Provisioning Service
IoT Devices
Telemetry and Event Ingestion
Telemetry
Event Hubs
Events
Event Grid
IoT Data
IoT Data
Files
Realtime Analytics
Stream Analytics
Data Process
Azure Functions
Presentation
App Service
Time Series Insights (TSI)
Web App
SignalR
PowerBI
Telemetry
Storage / Database
Storage table
Blob Storage
DB
Telemetry Monitor, analyze, visualization
Time Series Insights
Users Browser
Legend
Built-in
Optional
Future Versions / Optional
Preconfigured
Public Internet

ARMテンプレートの内容

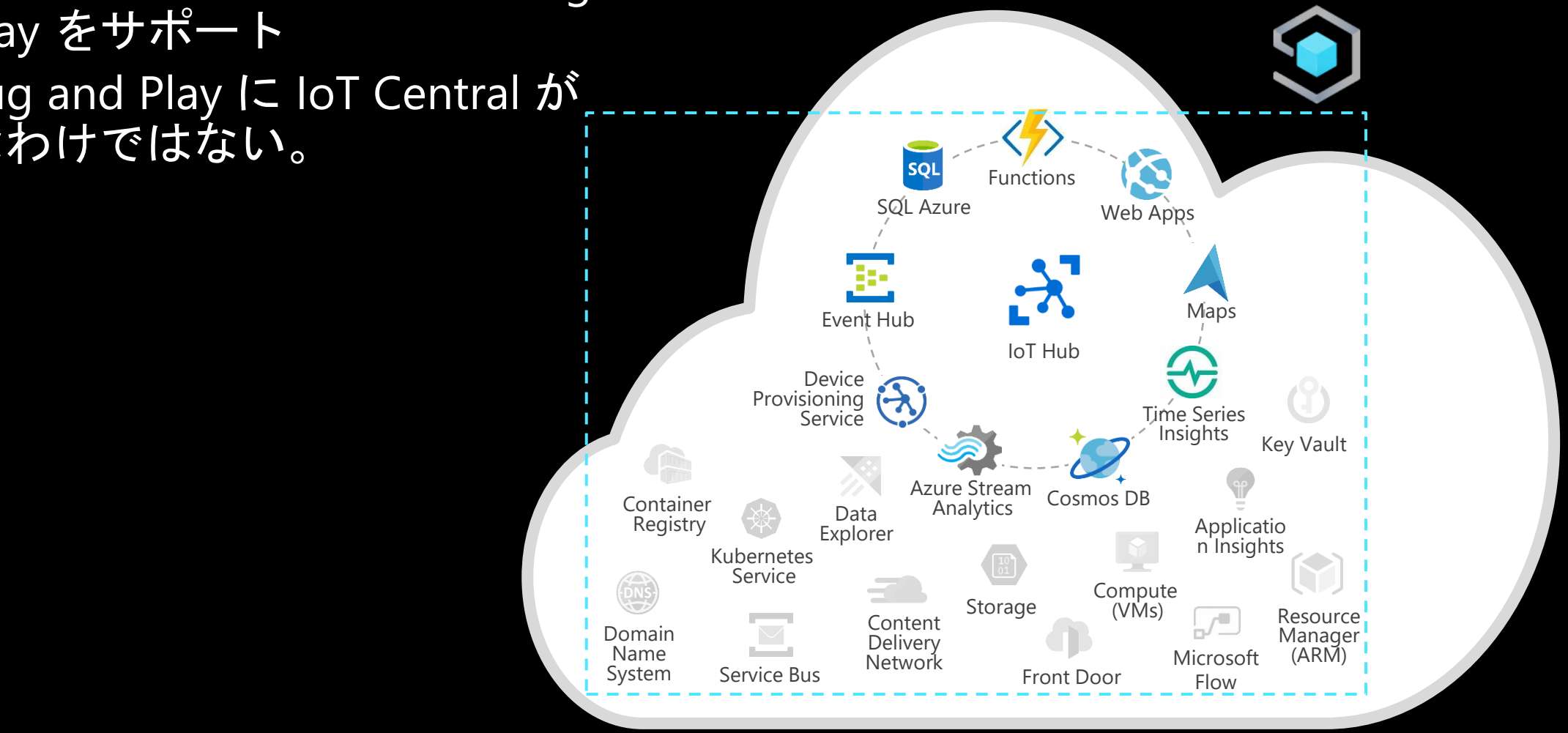
- 16個のリソース
  - IoT Hub, DPS, Event Hubs, SignalR, Web App, Functions, TSI, Azure Mapなど
- 各サービスの設定
  - SKU, ロケーション、サービスごとの設定項目
- データのルート設定
  - Event Grid トピックや、イベントハブのコンシューマーグループ、メッセージルート
- FunctionsやWeb Appのカスタムアプリケーションのデプロイ
  - Github からソースコードからCICD
- サービスの接続
  - 接続文字列などAzure Resourcesで対応できる部分は設定

Microsoft Azure

SaaS - Azure IoT Central で体験

Azure IoT Central - PaaS の集合体としての SaaS

- 2020/12 のアップデートでIoT Plug and Play をサポート
- IoT Plug and Play に IoT Central が必須なわけではない。



Microsoft Azure

PaaS - Azure Platform Services で体験

Azure IoT device model repository (DMR) & モデル 分解

ソリューションは モデル定義ファイル (JSON ファイル) にアクセスする必要がある

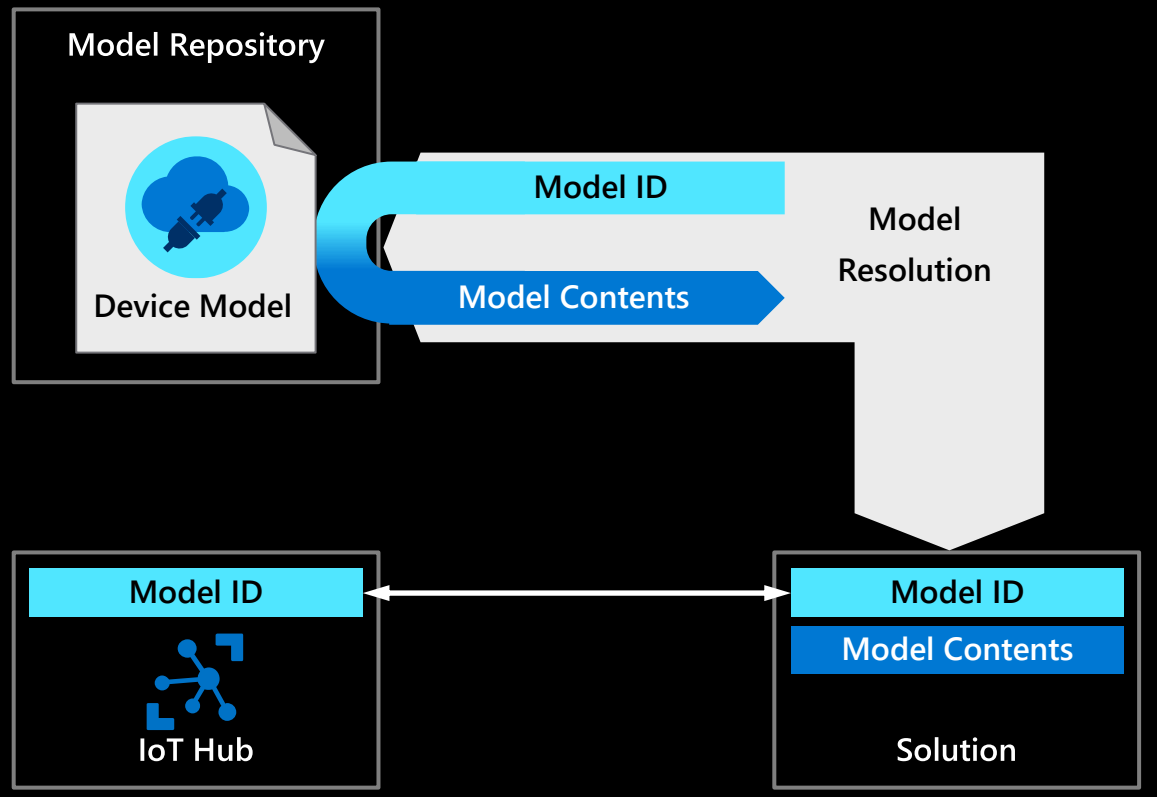
- Azure IoT デバイスモデル レポジトリを介して
- GitHub やファイル共有のような独自のレポジトリを介して

モデルは不変 (変更不可)

- パブリック モデル レポジトリに承認されたモデルは不変

Azure IoT デバイスモデル レポジトリ を介してアクセス

- Microsoft がホストする パブリック モデル レポジトリ
- カスタム モデル レポジトリ
- HTTP API と ローカルファイルアクセスをサポート
- Model Repo クライアント SDK



IoT Plug and Play デバイスモデル 例

温度センサー

テレメトリ ペイロード

{“temp” : “12.34”}

デバイスモデル 無し

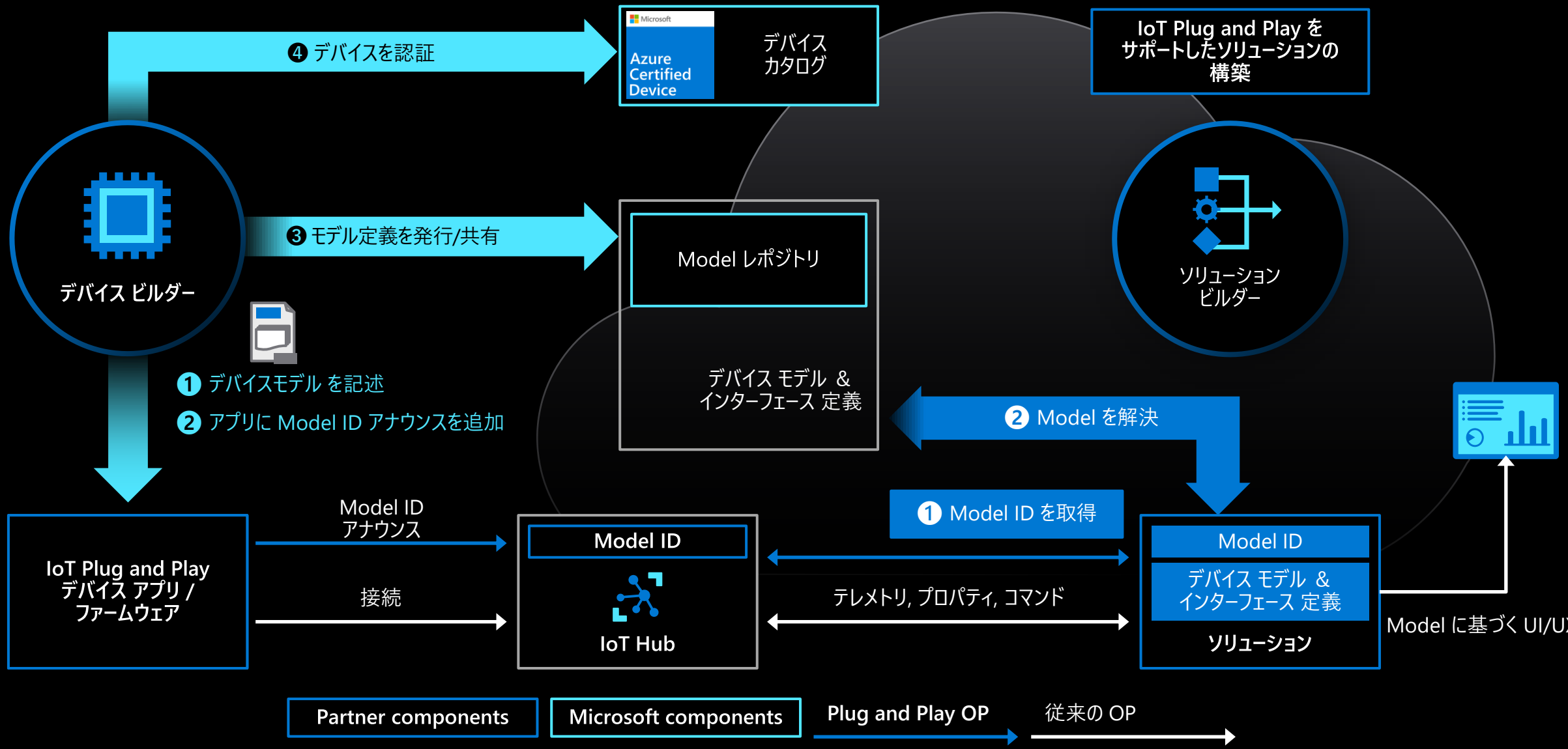
“ temp が 12.34 ”

デバイスモデル 有り

“ 温度が 摂氏12.34度 ”

```
{
  "@context":"dtmi:dtdl:context;2",          DTDL v2
  "@id":"dtmi:com:mycompany:Thermostat;1",   Model ID
  "@type":"Interface",                       Interface type
  "displayName":"IoT Plug and Play Device",
  "description":"Device Model Example",
  "contents": [
    {
      "@type":["Telemetry",                  Telemetry type
               "Temperature"],               Semantic temperature
      "name":"temp",                         Name of data
      "displayName":"Temperature Data",
      "description":"Temperature in degrees Celsius.",
      "schema":"double",                     Schema
      "unit":"degreeCelsius"                 Unit
    },
```

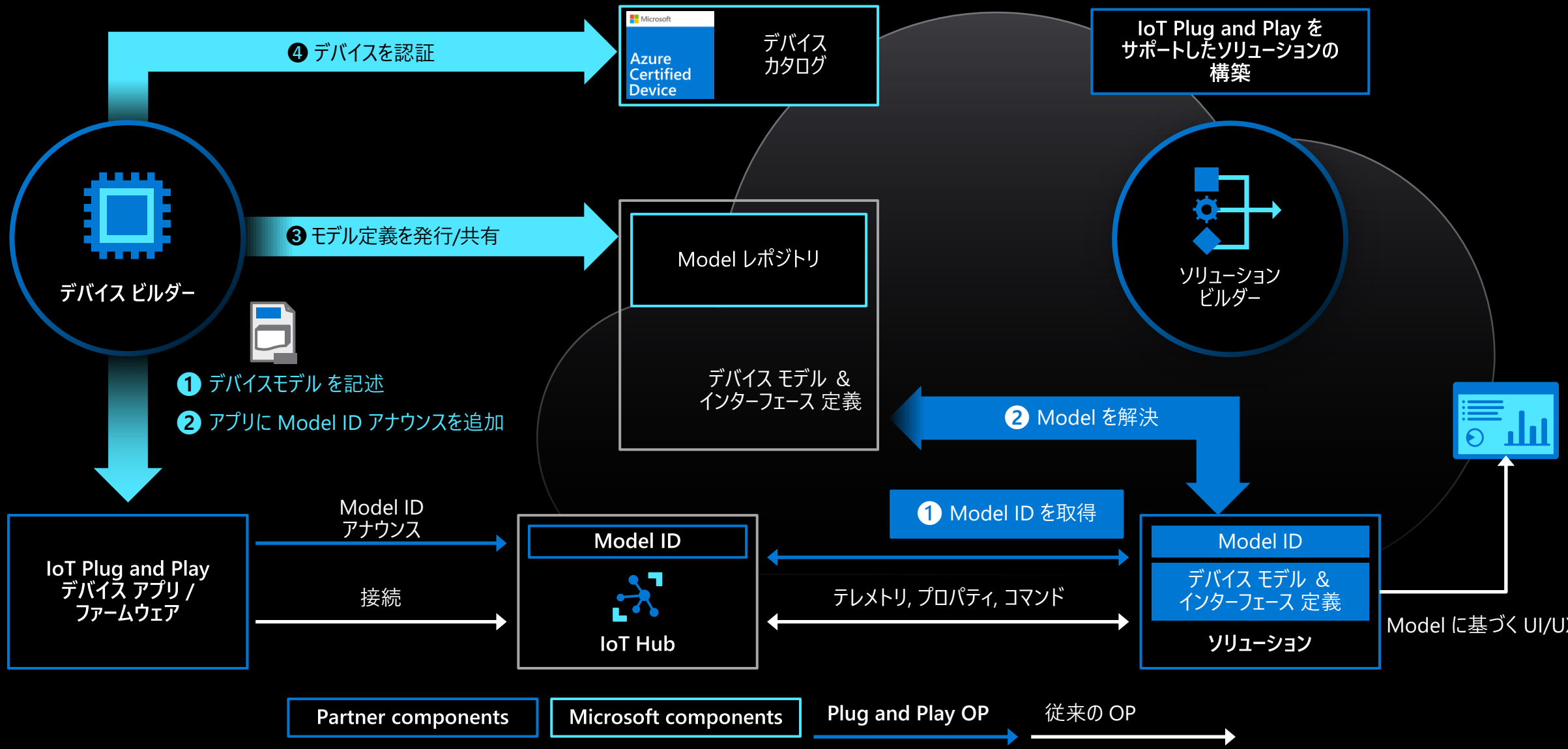
IoT Plug and Play プロセス 概要（デバイス）



DTDL Parser

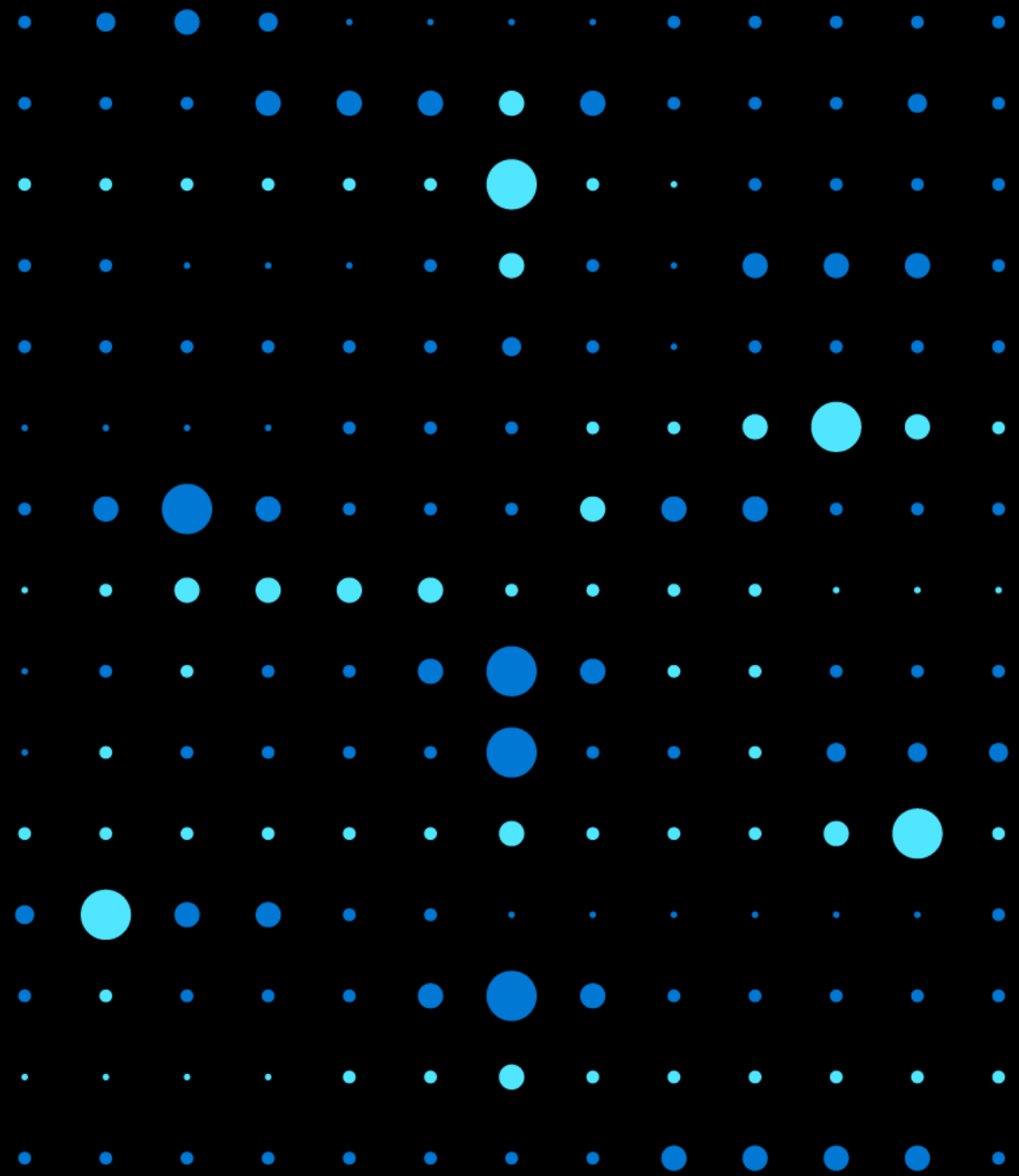
- Parses and validates DTDL
- Provides a C# object model for DTDL models (“reflection” over DTDL)
- Handles DTDL model sets (e.g. inheritance, etc.)
- Can be used by client software:
  - To validate models before service update
  - To find out which properties or methods are defined in a model (taking into account inheritance)
  - To drive UX generation code for model-driven UX
- C# code library distributed via NuGet

IoT Plug and Play プロセス 概要（デバイス）



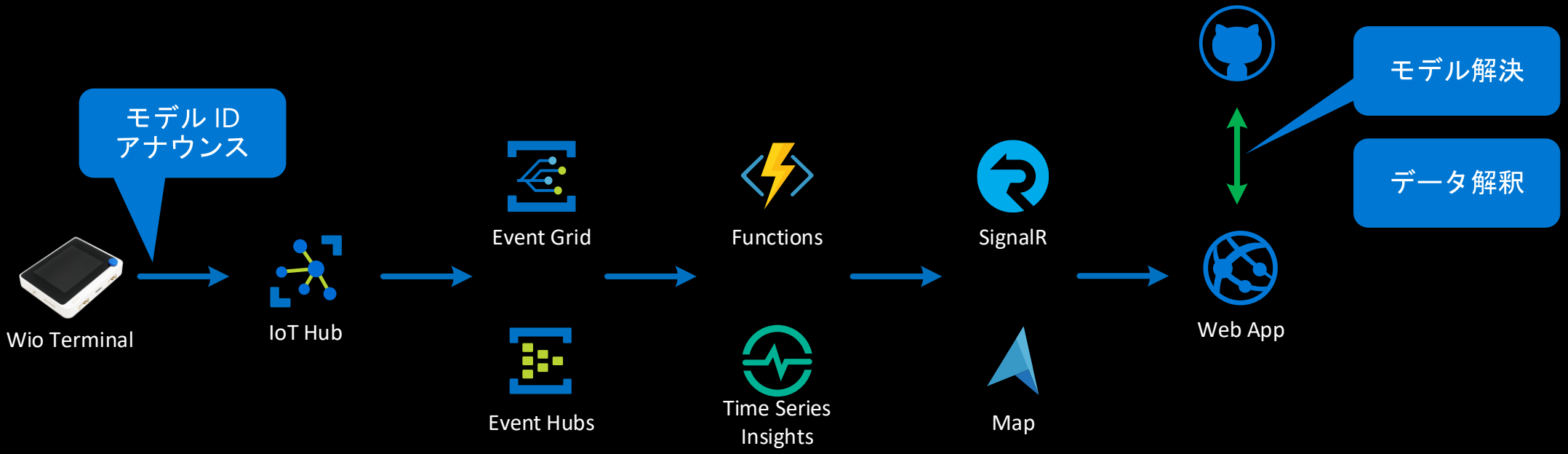
Microsoft Azure

Hands on Lab Day 2

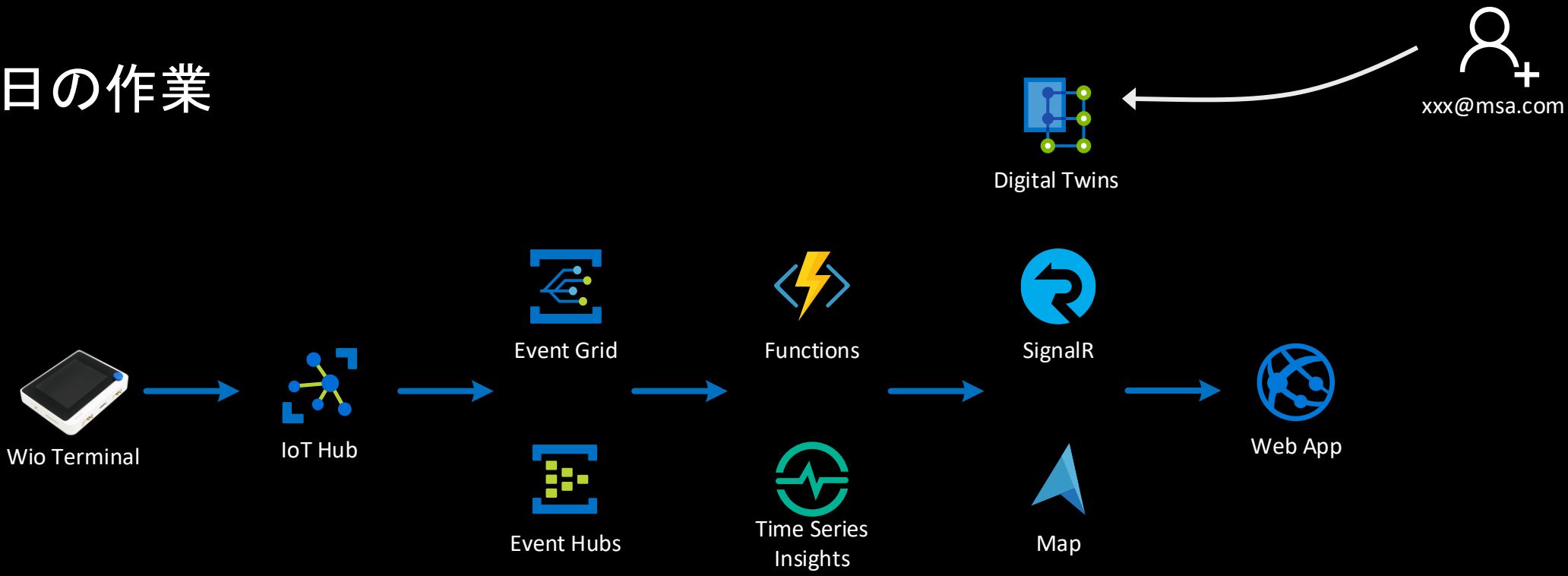


おさらい

- 16個のリソースをデプロイ、データの疎通を確認
- IoT Plug and Play
  - Wio Terminal : Model ID アナウンス
  - IoT Hub, Functions, TSI, Web App でモデル ID を保存・使用
  - Web Appでモデル解決 -> データの解釈（テレメトリーを表示）

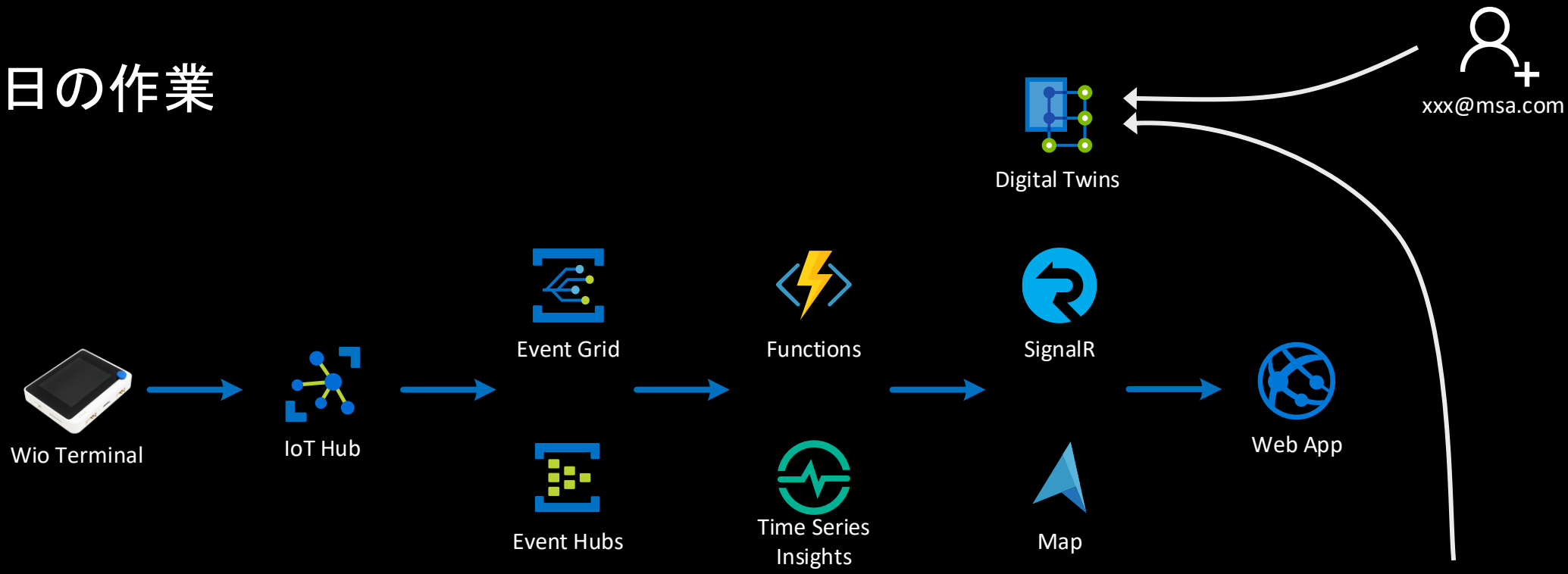


本日の作業

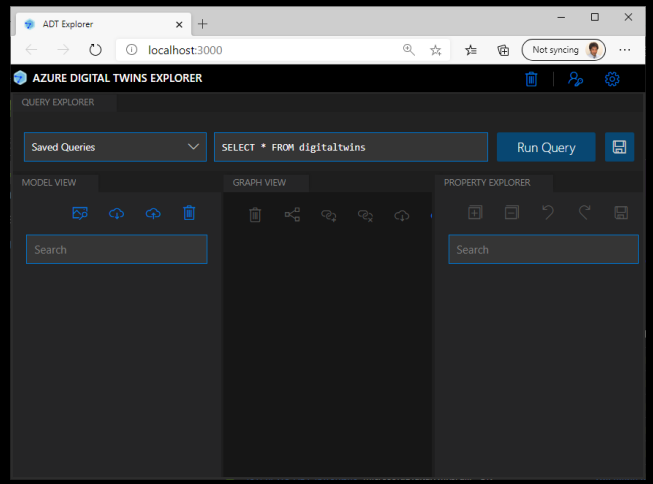


- Azure Digital Twin にモデルを構築するための準備
  - コマンドラインツール ADT クライアントを使用
- 必須項目
  - ADT のアクセス権
    マイクロソフトアカウントを Digital Twin Owner として追加
  - .net core 3.1 実行環境
    net core 3.1 ランタイムインストール。Cloud Shell でも実行可能

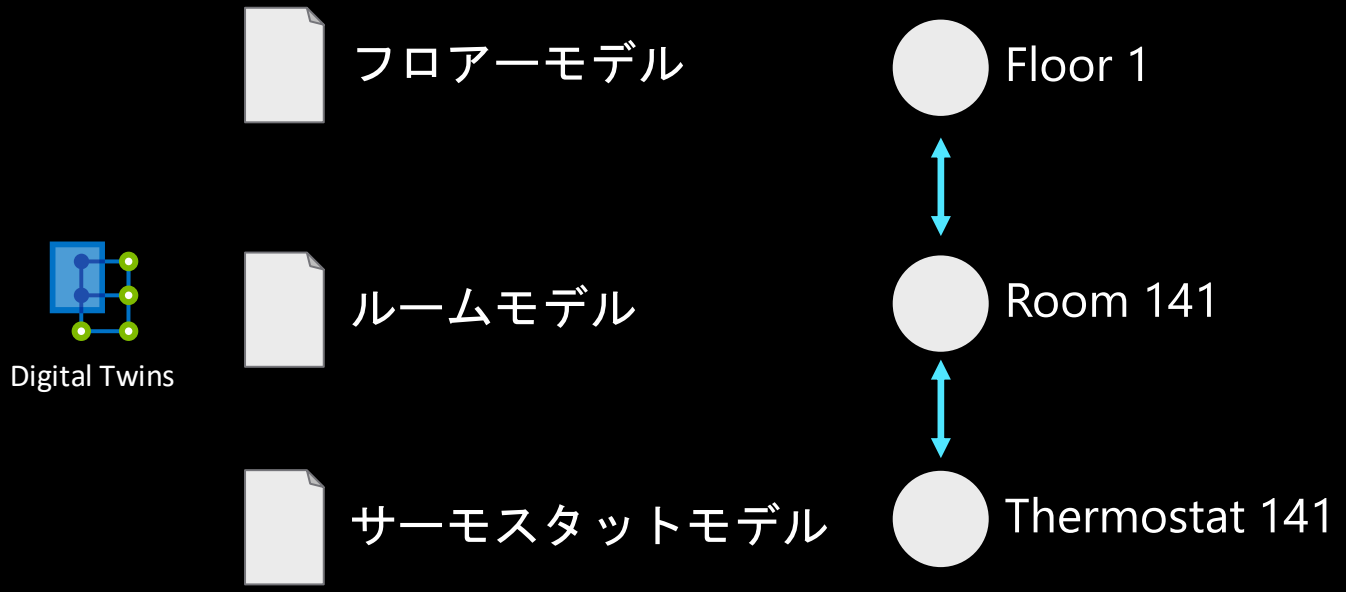
本日の作業

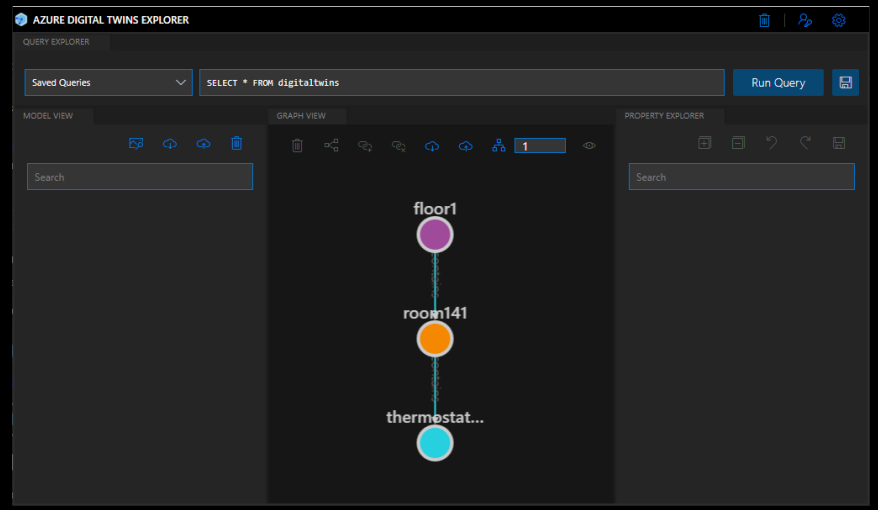


- Azure Digital Twin のモデル・データの可視化の準備
  - GUIツール Azure Digital Twins Explorer を使用
- 必須項目
  - Node.js の実行環境
    Node.js version 12 以降をインストール
  - ADT のアクセス権
    マイクロソフトアカウントを Digital Twin Owner として追加



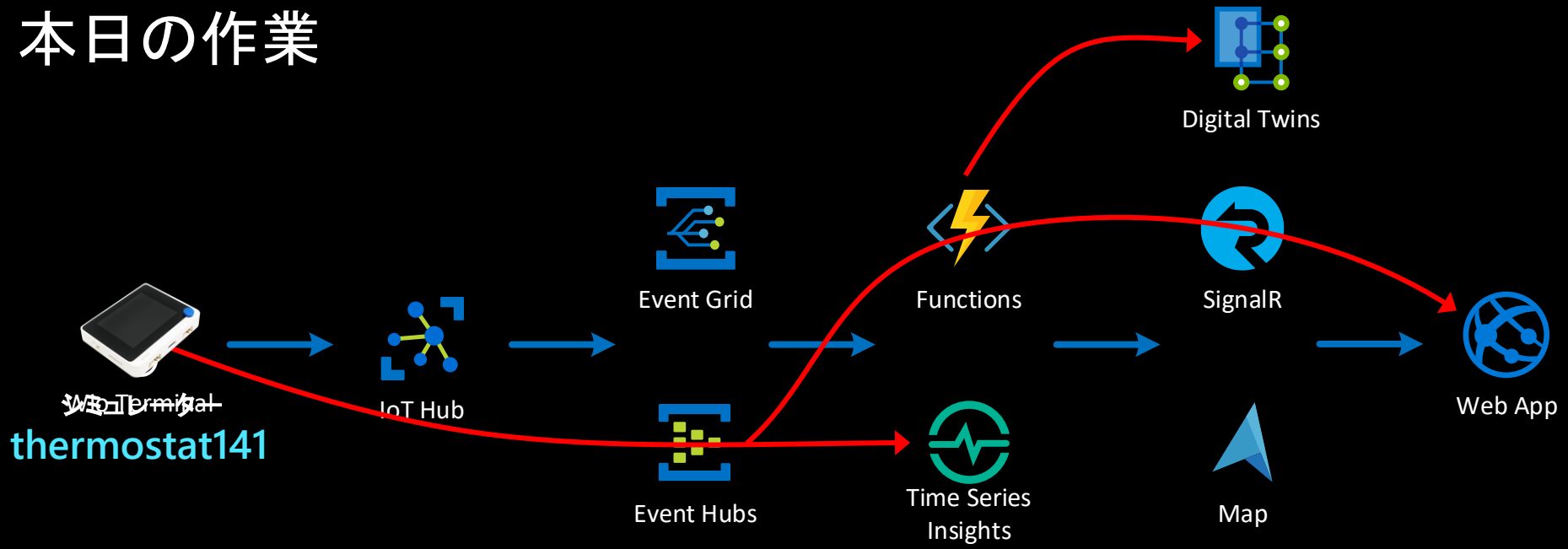
本日の作業





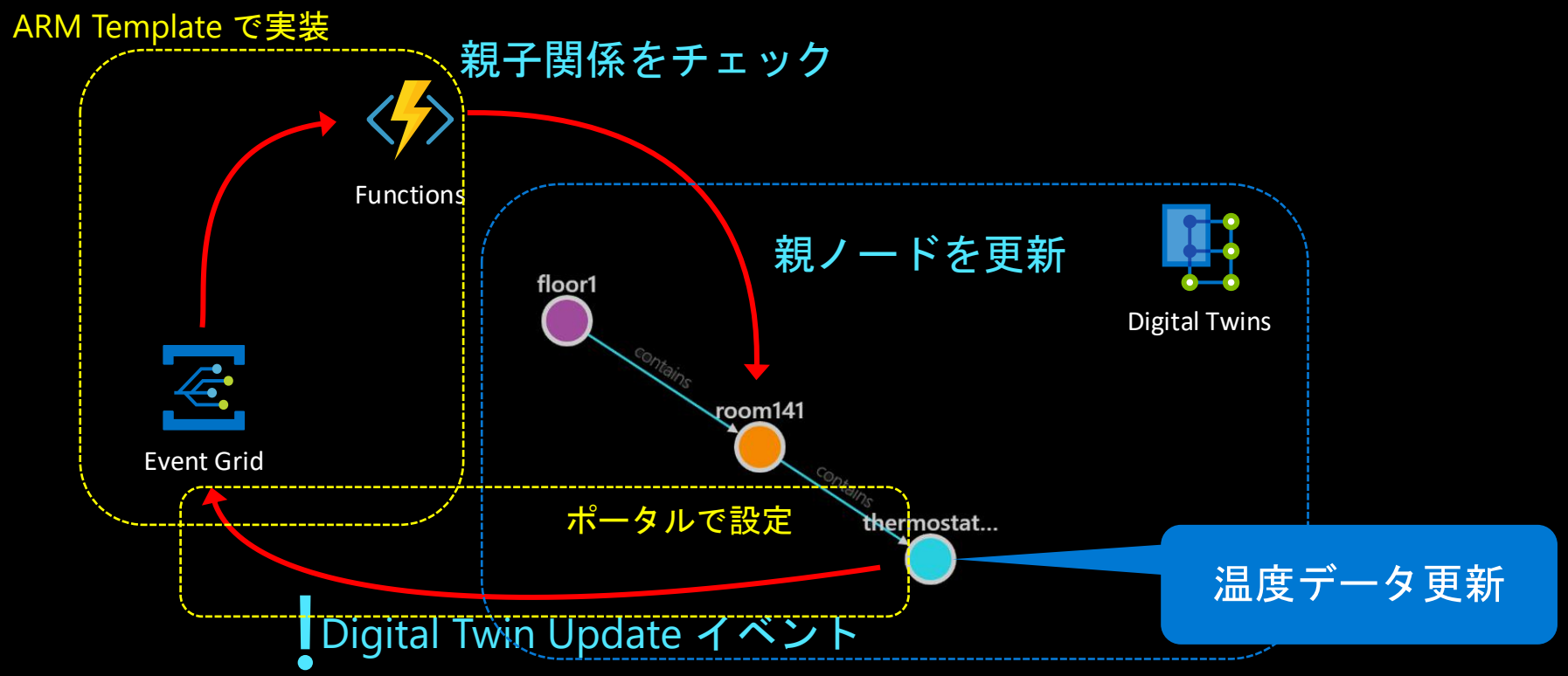
- デジタルツインに、モデルを作成
  - IoT Plug and Play デバイスと同様に、各ツインの基になるモデルを作成します
  - Floor Model, Room Model, Thermostat Model
- 作成したモデルをベースにツインのノードを作成します
  - Floor 1, Room 141, Thermostat 141
- 各ツインの関係性を構築します
- ADT Explorer で確認

本日の作業



- デバイスからのデータをデジタルツインモデルに送信
  - シミュレーターの IoT Plug and Play 温度計デバイスをソリューションに追加
  - 注：今回はデモ目的でデバイス名を “ thermostat141 “ に固定しています
- デバイスからソリューションにデータが流れていることを確認します
- デジタルツイン “thermostat141” の “Temperature” プロパティーがデバイスからのデータによって更新されていることを確認します
  - Azure Functions でデータ転送がすでに実装済みです。
  - ADT クライアントでモニター、もしくは ADT Explorer で確認

本日の作業

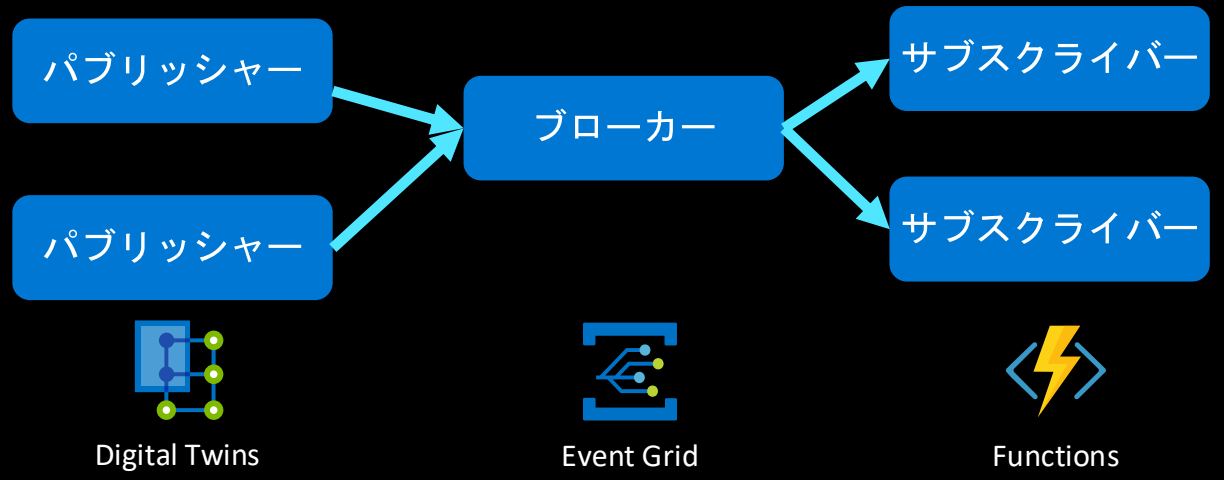


- デジタルツインの親子関係をベースに温度計のデータ（プロパティー）を親である Room 141 に反映させる
  - ADT はツインのプロパティーの更新時にイベントを発生させる
  - イベントをトリガーに温度データを Room 141 に反映させる
- Event Grid にイベントをパブリッシュ
- Azure Functions がイベントをサブスクライブ
  - Azure Functionsでツインの親子関係を調べ、親ノードの Room 141 のプロパティーを更新
- 設定項目
  - Event Grid の設定は ARM Template を使用 (Subscribe)
  - Azure Digital Twins のイベントルートの設定は手動 (Publish)

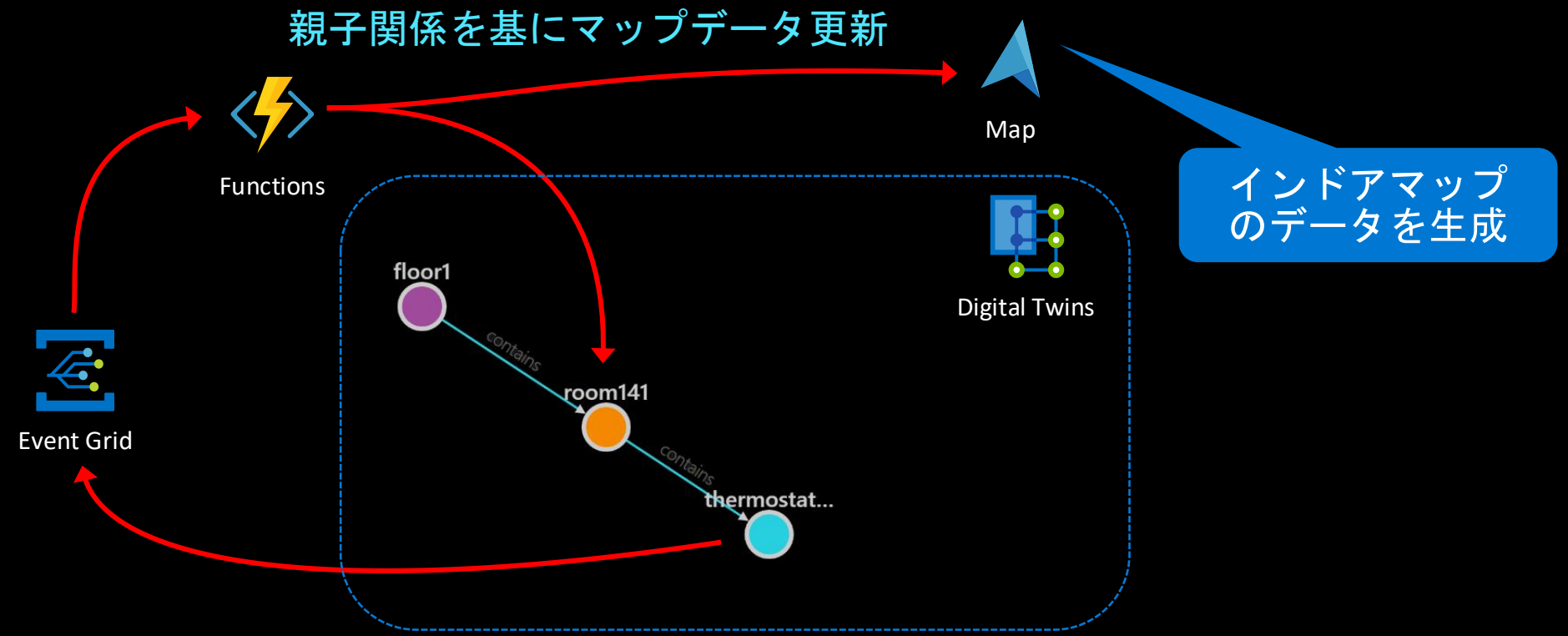
おさらい：通信方法

- クライアント・サーバー
  - 基本的には１対１の関係
  - クライアント
    - 受信側が何をするか、何を期待するか分かった状態でデータを送信
    - レスポンスを受け取れる
  - サーバー
    - データ・リクエストを受信・処理
    - レスポンスを返す
- 送受信両側がオンラインでないと成立しない
  - クラウド環境には不向き

- パブサブ (パブリッシャー・サブスクライバー)
  - 間にブローカーと呼ばれる仲介役が存在する
  - 多対多の関係
  - パブリッシャー
    - データをパブリッシュ（発行）する
  - サブスクライバー
    - データをサブスクライブ（消費）する
  - ブローカー
    - 仲介役になりコミュニケーションを調整
  - トピック
    - やり取りするデータ



本日の作業



- デジタルツインを可視化する
  - ツインのプロパティをインドアアップを使用し可視化します
- Azure Indoor Mapのデータ作成
  - CADデータからインドアマップのデータを作成します
- ツインの親子関係を基に、マップデータを更新

Microsoft Azure

IoT Plug and Play Certified Program

認定プログラムのシンプルステップ

1. MPN 登録する
   必須：会社メールアドレス
   必須：Azure AD (Azure トライアルで作成出来る無償版で可）
   https://partner.microsoft.com/ja-JP/
2. デバイスモデルをモデルレポジトリアップロードする
   https://docs.microsoft.com/en-us/azure/iot-pnp/concepts-model-repository
   # この認証は GitHub のもの
3. 認証テストを行う
   https://certify.azure.com/
   # ここで企業アカウント (AAD) ログインが必須