

IoT Plug and Play で IoT をもっとシンプルに

Daisuke Nakahara, Principal IoT Solution Architect
Koichi Hirao, Senior Program Manager
Takehiro Hirai, IoT Technical Specialist

2020/12/08

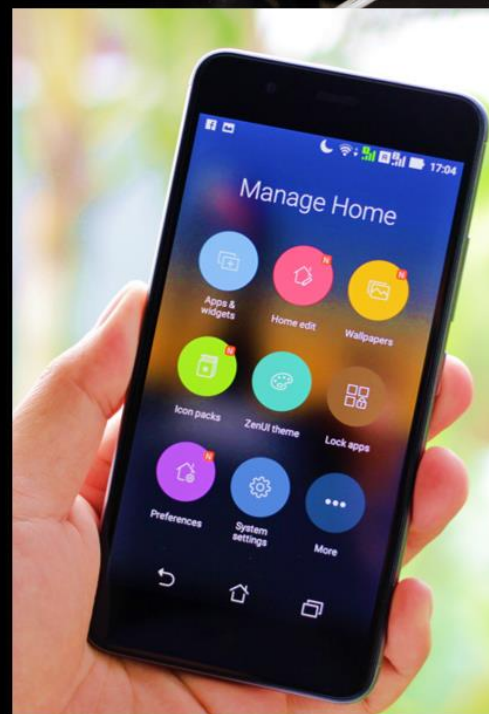


IoT Plug and Play のビジョン

IoT は今も ゲームチェンジャー を必要としている

スマートフォンの登場がモバイル業界を “良い意味”で混乱させた

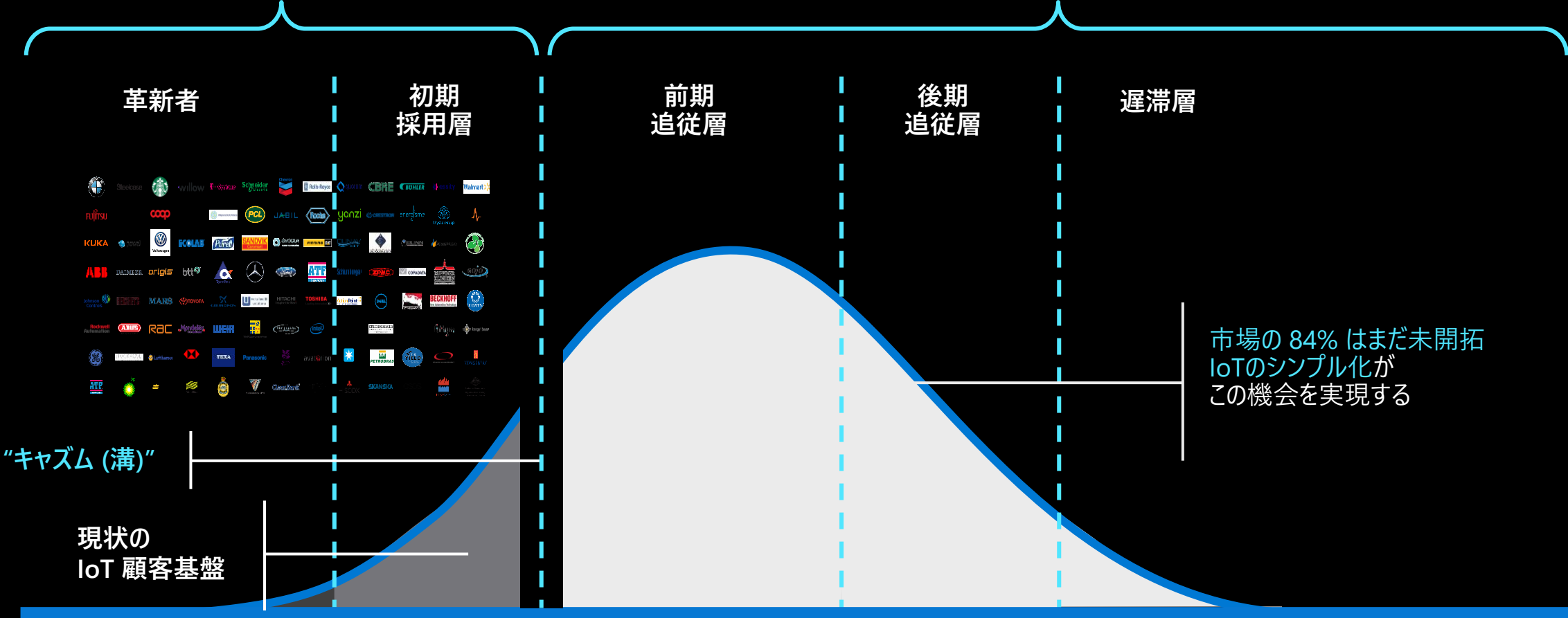
- 細分化された特注のモバイルスペースを シンプル化
- シンプルな体験でユーザーを 喜ばせた
- アプリストアを介して開発者とユーザーを 繋いだ
- モバイル業界で フライホイール効果 (好循環) を創出



シンプル化することが、IoT をメインストリーム化する鍵

メインストリーム化するには、価値創造までの時間を短縮出来るような、使いやすいソリューションが必須

IoT 業界の “今”



技術導入のライフサイクル

IoT の (今も残る) 課題



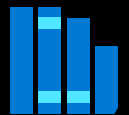
長い 時間軸

ソリューションの開発には長い時間を必要とします



スケールアップが 難しい

既存のソリューションにデバイスを追加するのは非常に複雑です



カスタマイズが 難しい

デバイスの機能を更新するのは難しいかもしれません



敏捷性の欠如

ソリューションは頻繁なアップデートと複数のプレーヤーとのコラボレーションを必要とします



スキル ギャップ

複数の技術にまたがるスキルを持つ技術者は非常に少ないです



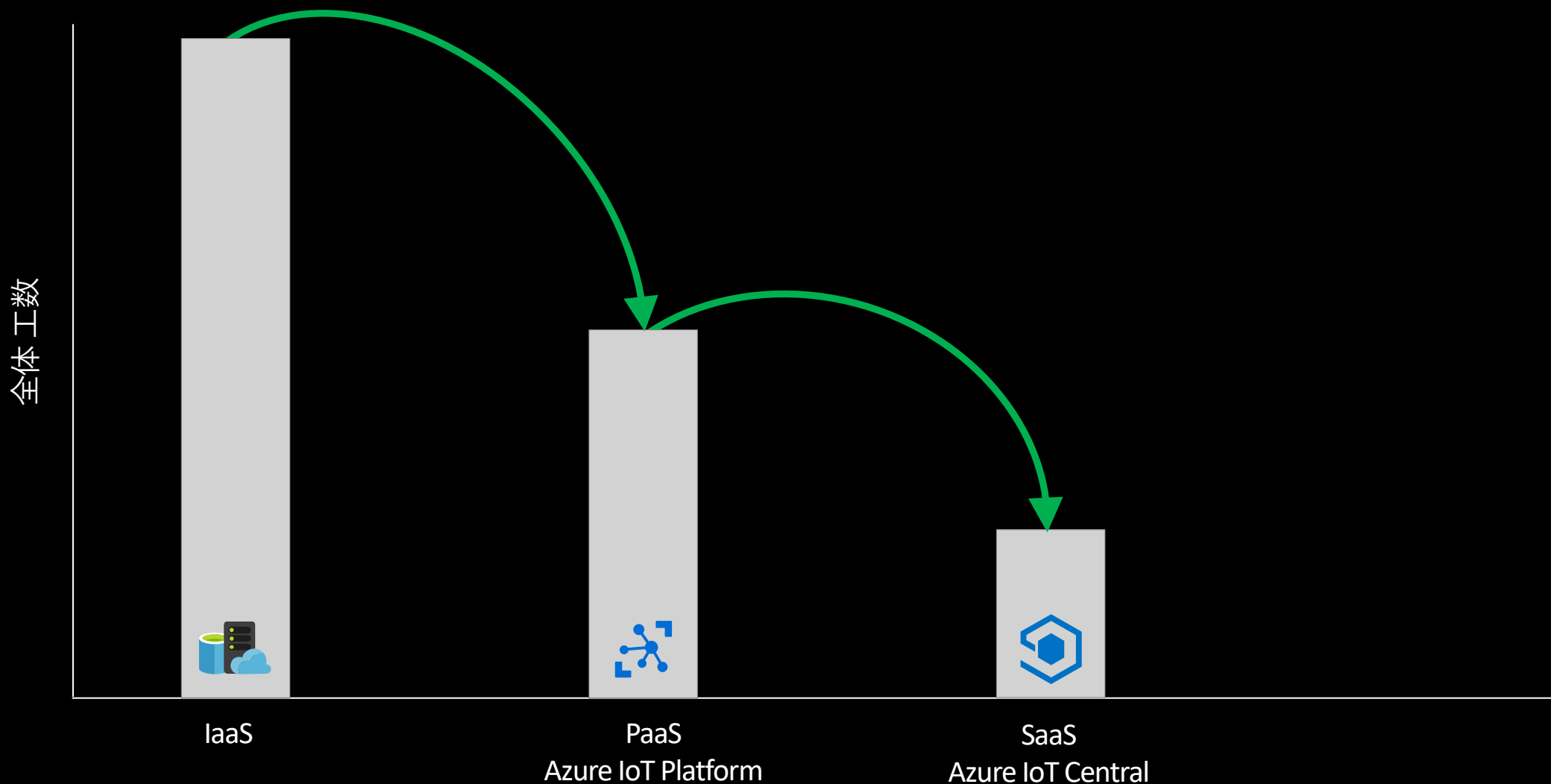
技術的な 複雑さ

異種エンドポイント、サービス、ビッグデータの統合は難しいです

などなど、他にも....

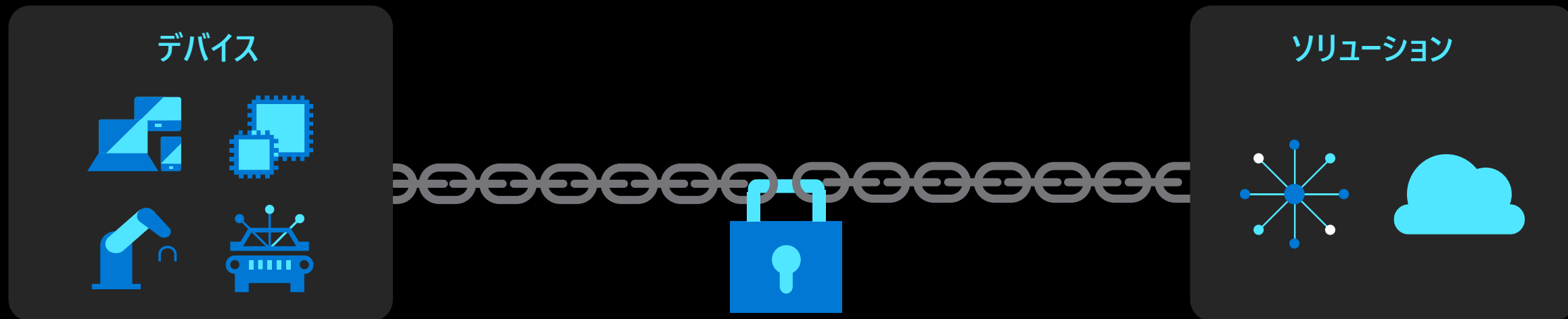
IoT の **メインストリーム化** を加速する

IoT ソリューション の構築、運用に要する工数は、従来と比較すると急速に減少、だが



IoT の現状

クラウド上の IoT ソリューションとデバイス上のソフトウェアが 強く 結合 している



IoT ソリューション 構築の現状

● 設計

デバイスを含む IoT ソリューションの 設計と構築

● 機器提携先 検索

ハードウェアを デザイン、製造出来る デバイスビルダー の検索

● 開発

デバイス向けファームウェア、及びクラウド アプリケーション の実装

● 融合(一体化)

クラウド アプリケーションからデバイスまでの 相互作用モデル を構築

● 導入

大規模なデバイスのプロビジョニング、クラウド アプリケーションからデバイス管理

● 展開

設定を適用して、導入を完了

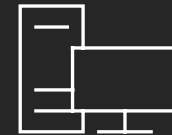
数ヶ月を要し、開発期間
の長期化を招く

過去に似たような課題が ...

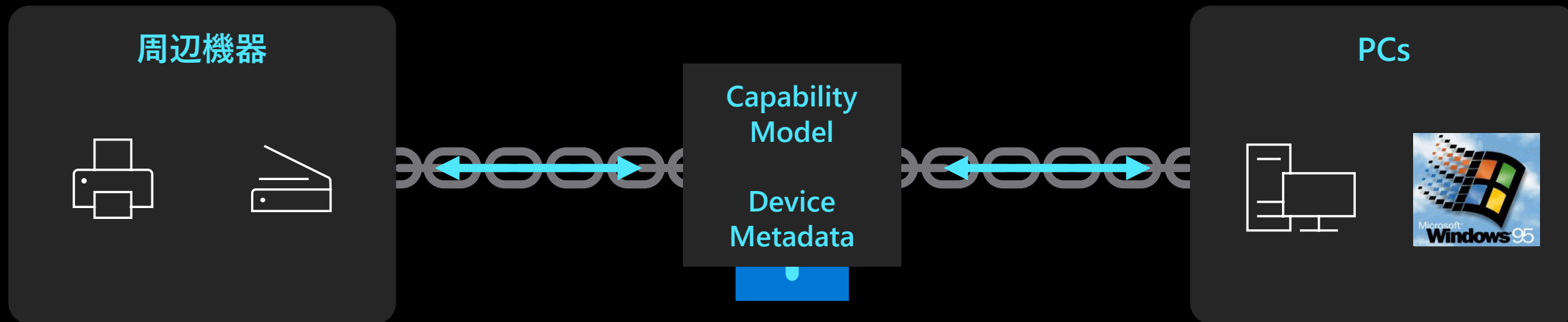
周辺機器



PCs



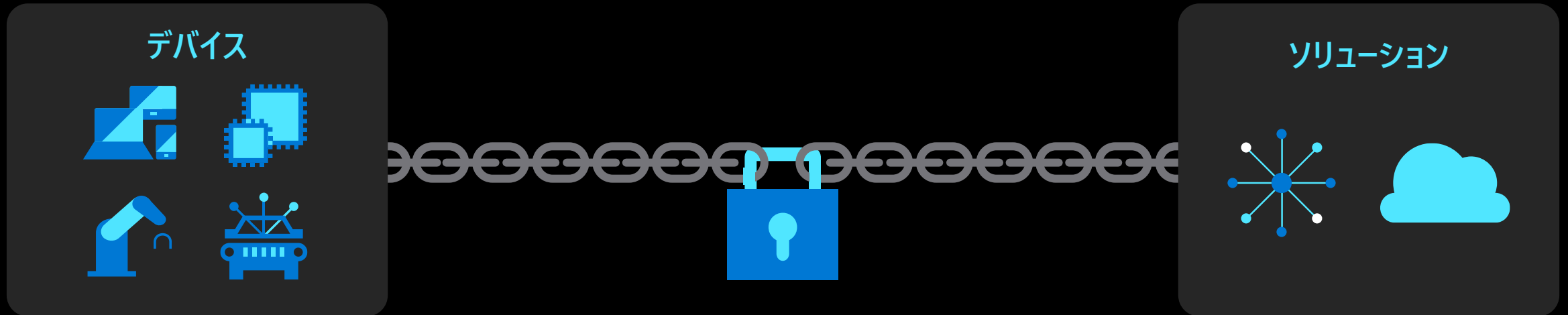
その節 (Windows) は Plug and Play の実装で解決



デバイスはそれ自身の **capability models** を公開し、それに準拠
Windows は **capability model** を用いて、デバイスがどのように **振る舞う** のかを把握

IoT Plug and Play のご紹介

オープンなモデリング言語を用い IoT ソリューションのデバイス相互作用を **シンプル化**



デバイスはオープンな デジタルツインモデリング言語 (DTDL) に基づいて自己記述

ソリューションは それを **自動的に** デバイスに適用可能

一切 **カスタム**のコード無しで

IoT Plug and Play で IoT ソリューションを構築

● 設計

IoT ソリューションの設計と構築 ~~デバイスを含む~~

● ~~機器提携先 検索~~ 機器選定

機器を Azure Device Catalog から 選択



● 開発

デバイス とソリューションを融合(一体化)させるためのカスタマイズは不要に

● 融合(一体化)

IoT Plug and Play により ソリューションはデバイスを理解可能

● 導入 +

IoT Plug and Play 認定済みデバイスは DPS 経由のプロビジョニングをサポート

● 展開

スケーラブルな IoT ソリューションをより簡単、かつ早く構築

● 設計

IoT ソリューションの設計と構築

● 機器選定

機器を Azure Device Catalog から 選択

● 導入 + 展開

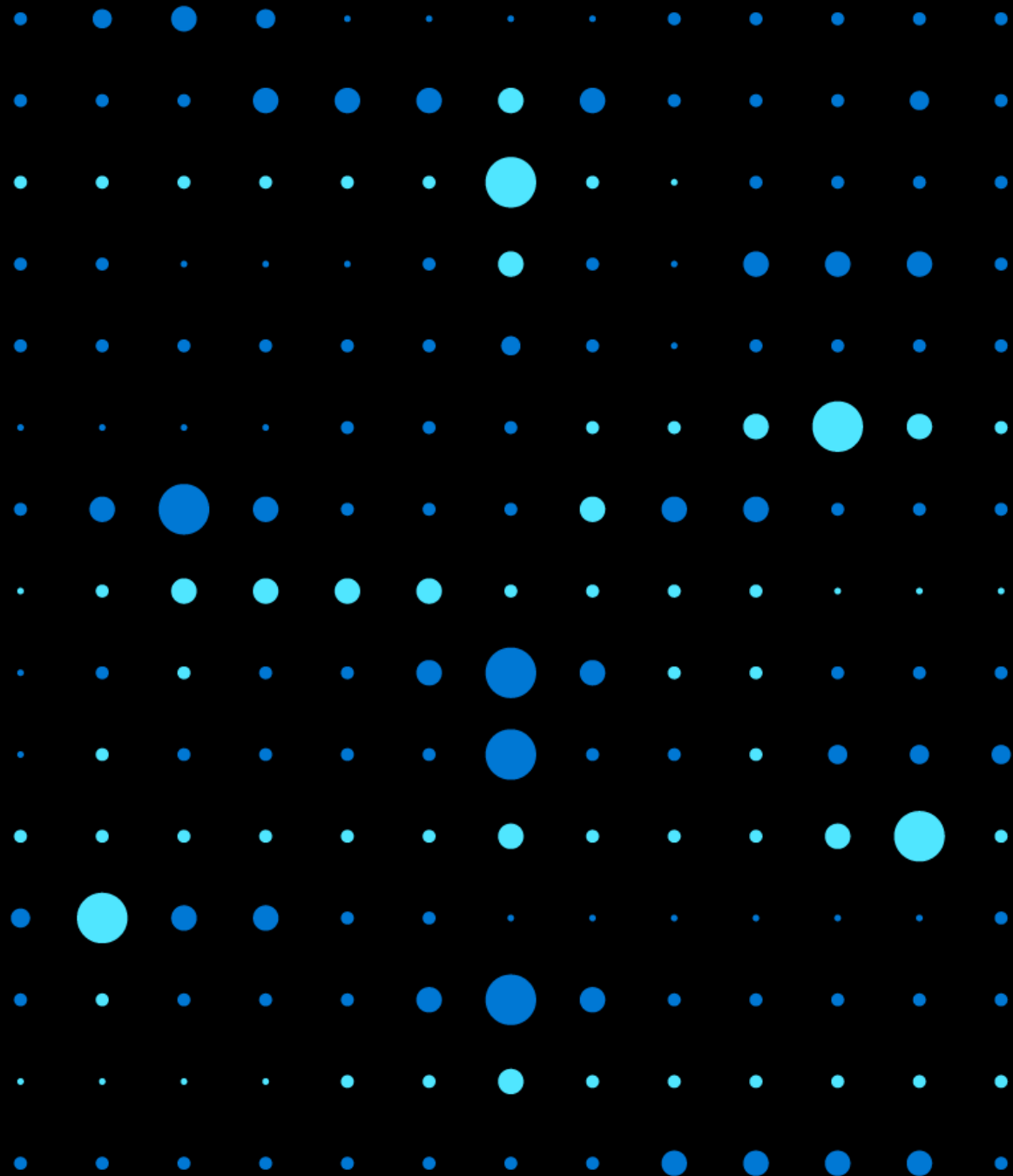
IoT Plug and Play 認定済みデバイスは DPS 経由のプロビジョニングをサポート



ソリューション開発者は、ソリューション開発に **集中** 出来る

デバイス開発者は、単一のファームウェアが **全てのソリューション** に対応可能になることで、出荷を簡素化 出来る

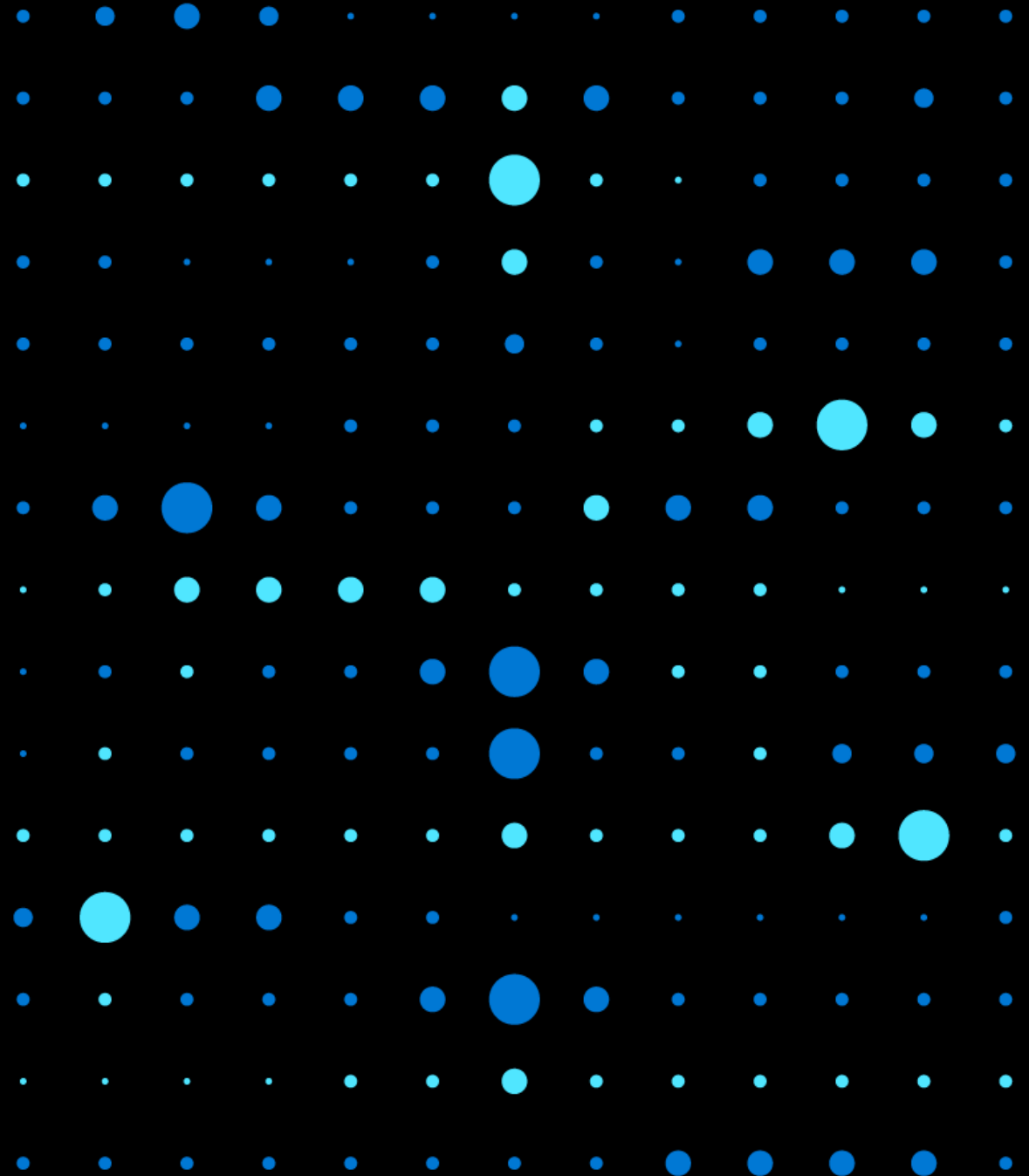
Demo Video



Simplifying IoT with IoT Plug and Play



Demo



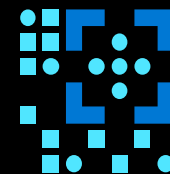
IoT Plug and Play が IoT をシンプル化



強力



高速

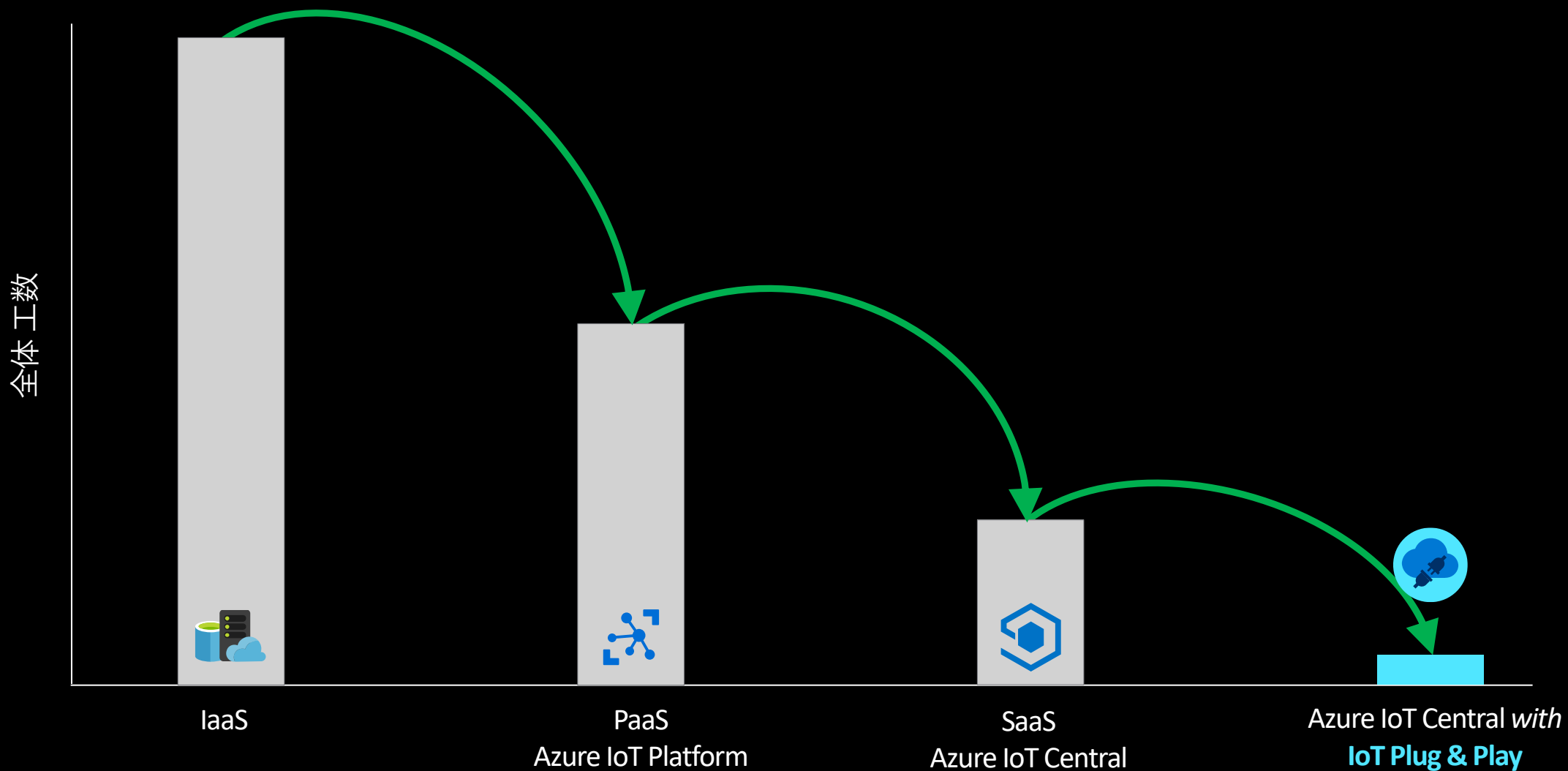


簡単

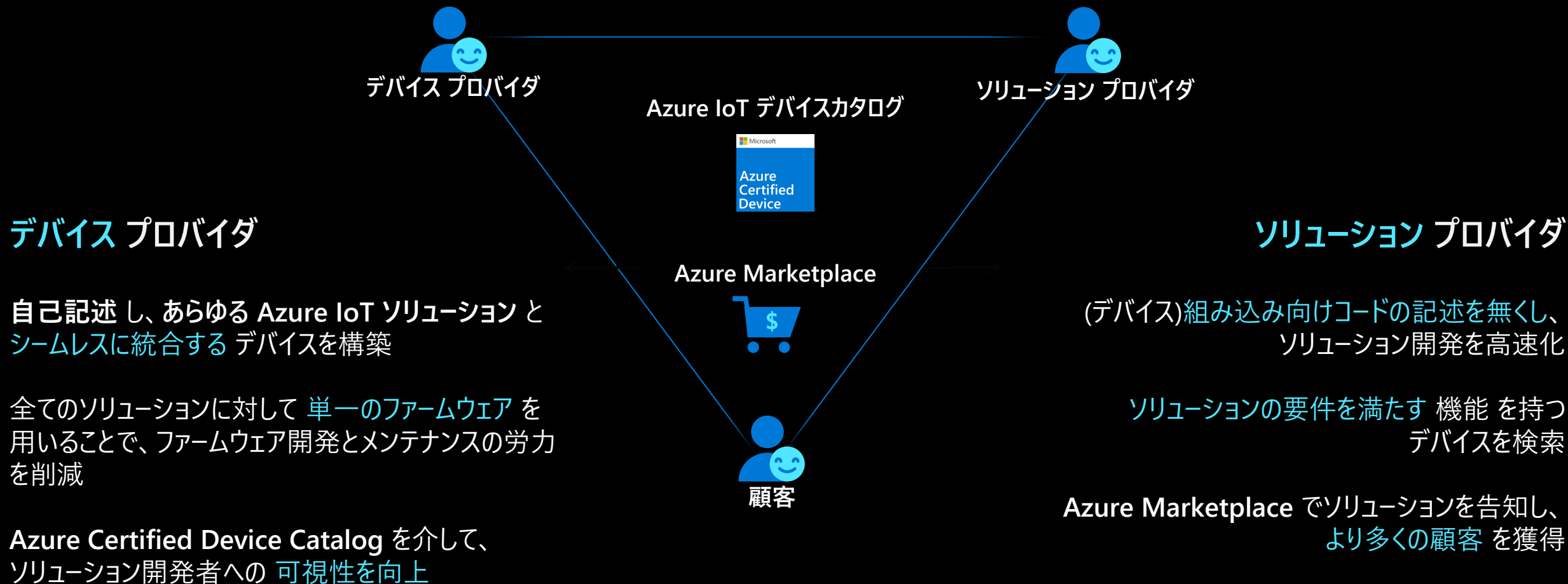
この3つのバランスを取ることは、サービスにとって非常に重要

IoT を加速する

IoT Plug and Play + Azure IoT Central が 全体 工数 を日/月単位から 数分 に短縮



IoT Plug and Play エコシステム を 構築 – 2021年、そしてそれ以降



パートナー連携による IoT Plug and Play エコシステムの構築



"For customers using Azure Digital Twins or IoT Central, being able to leverage IoT Plug and Play on both Octave and Azure will expand capabilities while making solution development even faster and easier."



"The main value in IoT Plug and Play is the ease of developing a device that will be used in a connected fashion. It allows for any company to easily define device telemetry and properties without writing any embedded code."



"IoT Plug and Play has helped our customers who need to acquire sensor data securely and effortlessly. The power of ESP32-Azure IoT Kit, combined with IoT Plug and Play, means our customers will experience firsthand the ease of deploying device-to-cloud solution integration."



"At Bosch we value open industry standards, and with IoT Plug and Play, Microsoft is closing a gap that existed since the inception of IoT. With IoT Plug and Play our customers can focus on the design of decision-making applications with the benefits of having ready to use sensors from the edge. Early adoption of IoT Plug and Play will give the Bosch XDK an advantage over devices that do not support this upcoming standard."



"In a world where the data context is king, solutions should focus on integration and business value instead of handshake mechanics. We are excited to get our devices IoT Plug and Play certified and look forward to an interoperable ecosystem."



TOKYO ELECTRON DEVICE LIMITED

"By making our devices IoT Plug and Play compatible, solution developers can start using the devices without coding. We are confident it will contribute to the rapid construction of IoT services that are easy for everyone to use."



デジタルツイン コンソーシアム

設立企業と先駆的企業

設立企業



先駆的企業

Air Force Research Laboratory
Animated Insights
Asset Management Lab, LLC
Association of Asset Management Professionals
Autosalo Ltd
BEC - Blockchain Engineering Council
BIM6D Consulting

Bandora Systems
Bentley Systems
Building 4.0 CRC
Chain Technology Development Co. Limited

CodeData
Connector Geek Ltd
ConstruWise, Inc.

CumuloCogitus Inc.
Cybertwin
DIGIOTAI
DataCities

e-Magic Inc.
Executive Development
Gafcon, Inc.

Geminus.AI

Healthskouts
IIMBE
IOTA Foundation

IOTIFY
Idun Real Estate Solutions AB
ieLabs
IoT Management

imec
Itus Digital
Jitsuin, Inc.

LINQ Ltd.

LUNO UAB
Lux Modus Ltd.
Monash University

NSW State Government
Neural Concept
Padi LLC
Pirate

PropTechNL
Resonai
Ricardo

Slingshot Simulations

Systems Analytics Solutions
Transforma Insights
Trendspek

Twin Building GmbH
University of Melbourne
UrsaLeo Inc.
WSC Technology
Willow Technology Corporation Pty Ltd
Ynomia
YoGeo, Inc.

IoT Plug and Play 概要



オープンなモデリング言語

Digital Twin Definition Language v. 2 (DTDL)



デバイスモデル

IoT デバイスを記述するための デジタルツインモデル



Azure Digital Twins 連携

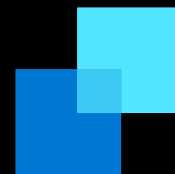
Azure Digital Twins とのシームレスな統合

オープンなモデリング言語



Digital Twin Definition Language v. 2 で IoT デバイスをモデル化

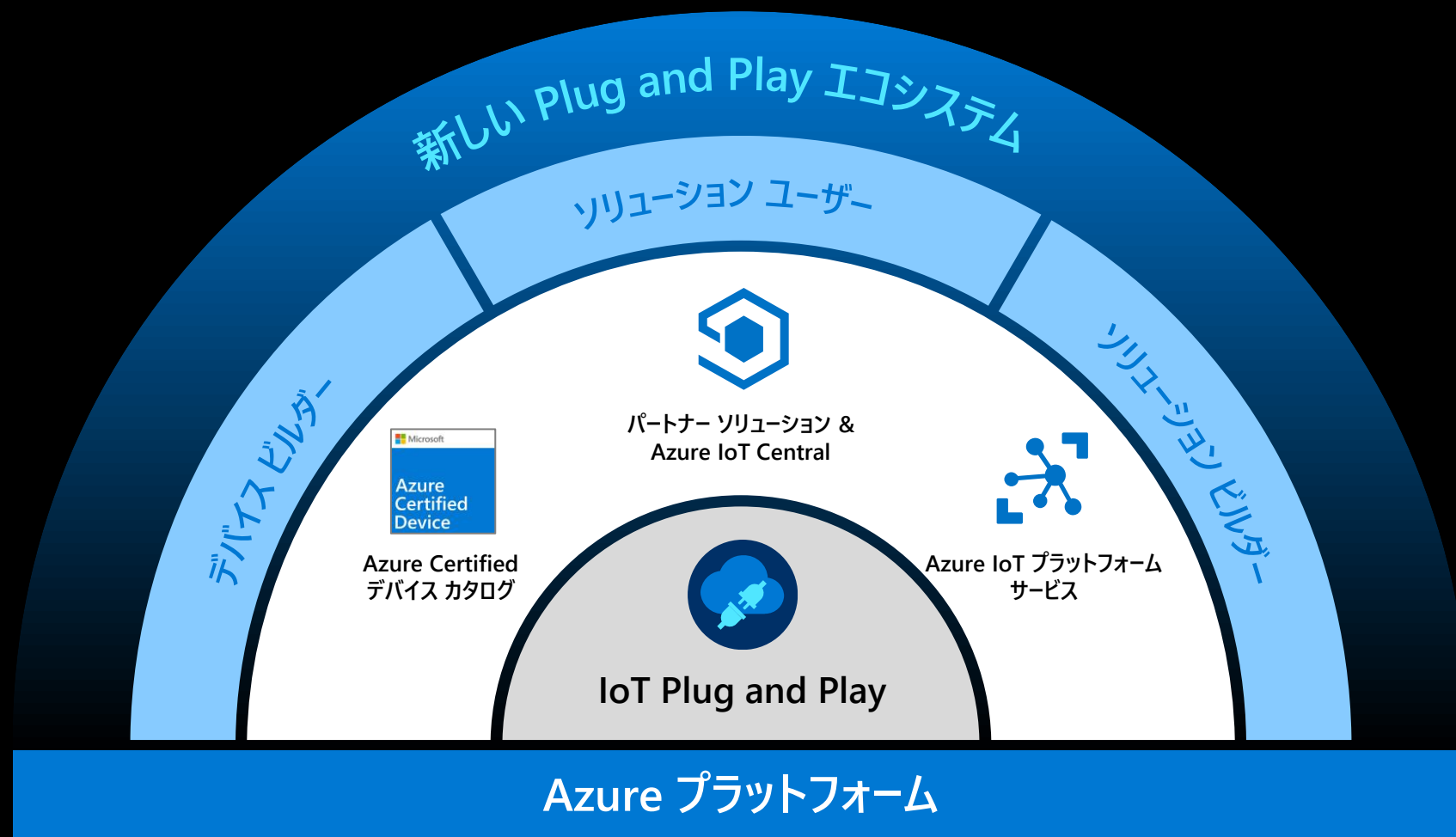
- JSON-LD と RDF に基づくオープン言語:
<https://json-ld.org/>
- インターネット上で 情報を公開したり、
利用したりする際に使用（例: 検索など）



デバイスモデルを介した IoT デバイスと IoT アプリケーション間の共通言語

- デバイス は IoT アプリケーション に機能や属性を 伝達
- IoT アプリケーション は デバイス の機能や属性を 理解

Digital Twin Definition Language モデルがエコシステムの “コア” に



DTDL で 相互運用モデル を記述

デバイスの機能と相互運用モデルを記述

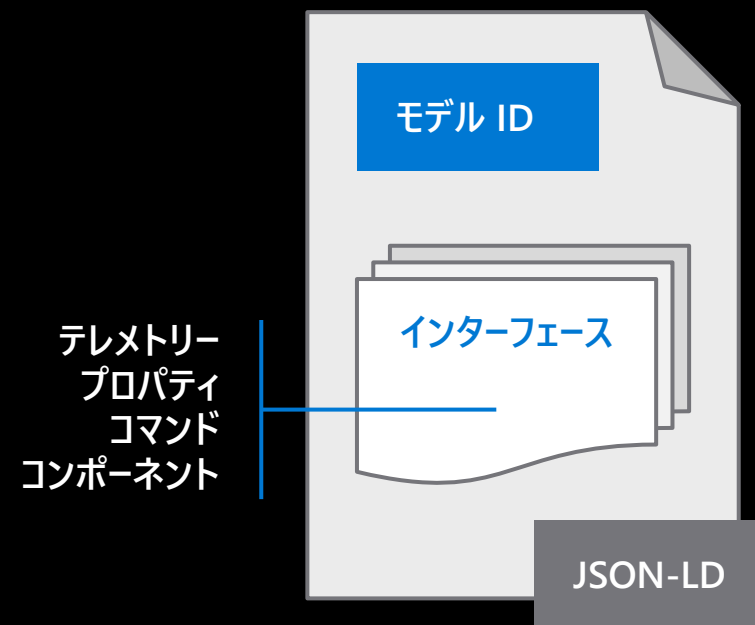
各デバイスモデルは、一意の ID を持つ: **Digital Twin Model ID (DTMI)**

各デバイスモデルは、一連の **インターフェース** で構成

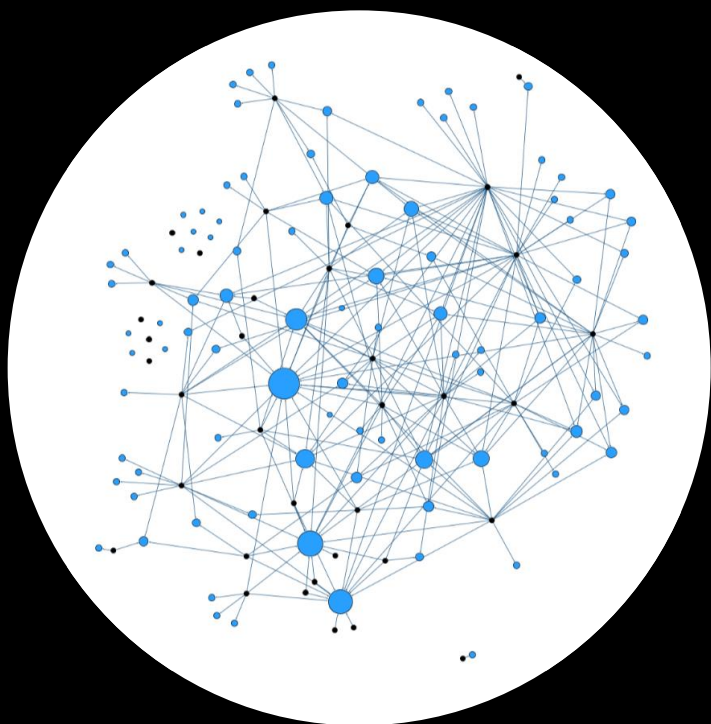
インターフェースはデバイスの属性を記述:

テレメトリー, プロパティ, コマンド, コンポーネント

ソリューションは デバイスモデル をクエリ、及び解析して、相互運用モデルを理解可



Azure Digital Twins との連携



デジタルツインが物理環境の
デジタルモデルに基づくナレッジ
グラフを作成を可能に

ツイン間の関係性を作成

メタデータと属性を追加し、
ツインを強化

IoT Plug and Play は
Azure Digital Twins と
完全に連携



IoT Plug and Play デバイスモデルは、グラフ内のツインの1つに



IoT デバイスをグラフに追加
(plug)

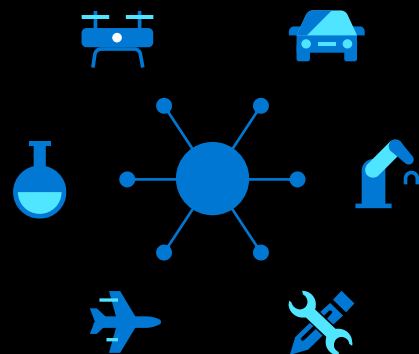


グラフ上での IoT デバイス操
作とデータの参照 (play)

コネクテッド IoT エコシステム という 未来 への進化

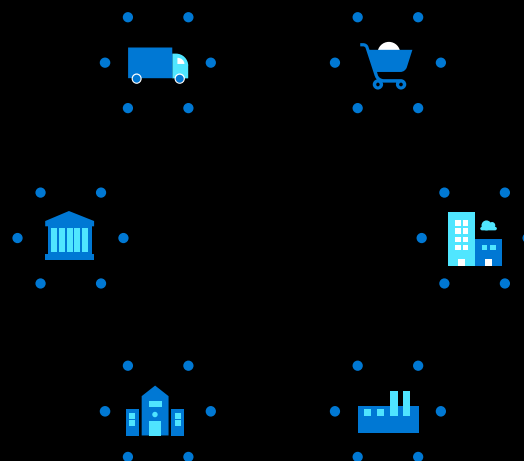
IoT Plug and Play はコネクテッド 資産 からコネクテッド 環境 への移行基盤に

今日



コネクテッド
資産

最新



コネクテッド
環境

未来

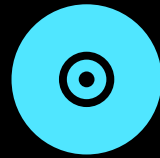
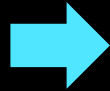


コネクテッド
エコシステム

例: メタデータ で見えるデバイスの進化



カセット
テープ



CD



MP3

アナログ

- ・ 物理的な 商品のみ
- ・ メタデータ 無し
- ・ 棚で整理



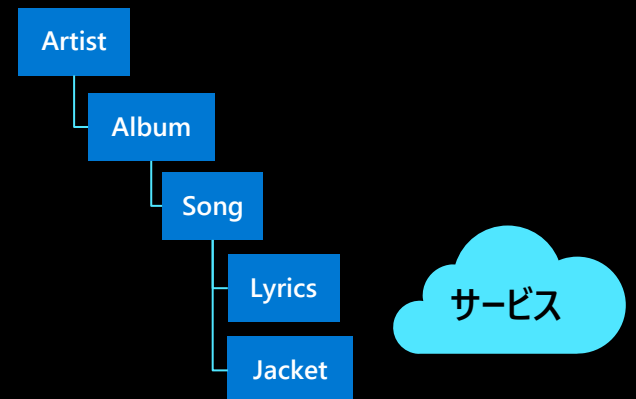
部分的な デジタル化

- ・ 物理的な 資産とデジタルなコンテンツ
- ・ いくつかの メタデータ - ディスク ID
- ・ メタデータを通じた インターネットベースのサービス を実現
- ・ まだ やや限定的

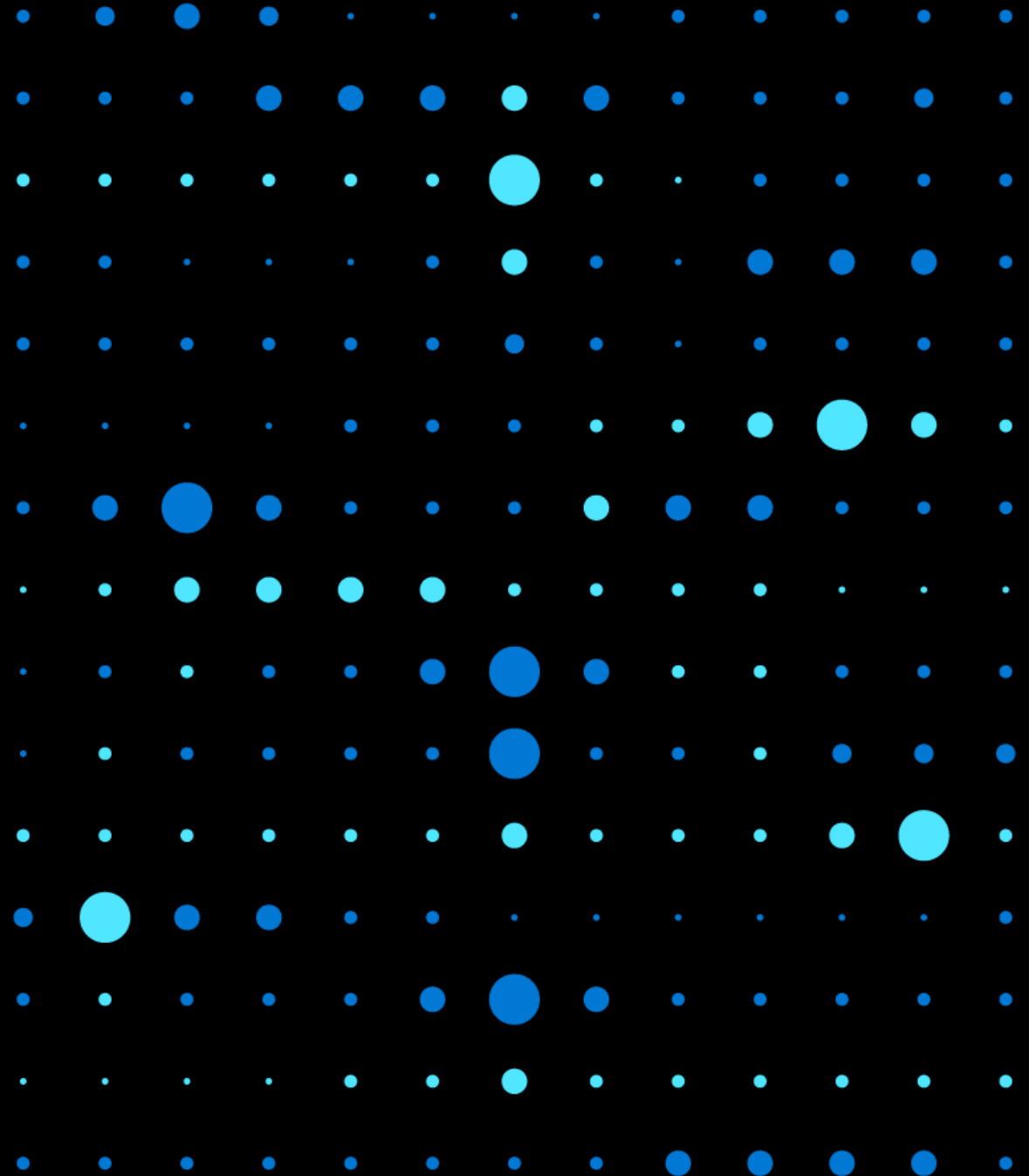


完全 デジタル化

- ・ デジタル資産 のメタデータ添付
- ・ 様々なメタデータ
- ・ 構造化されたメタデータを利用した新しいソリューション、サービス を実現
- ・ 多くの新しい 利用例



Demo

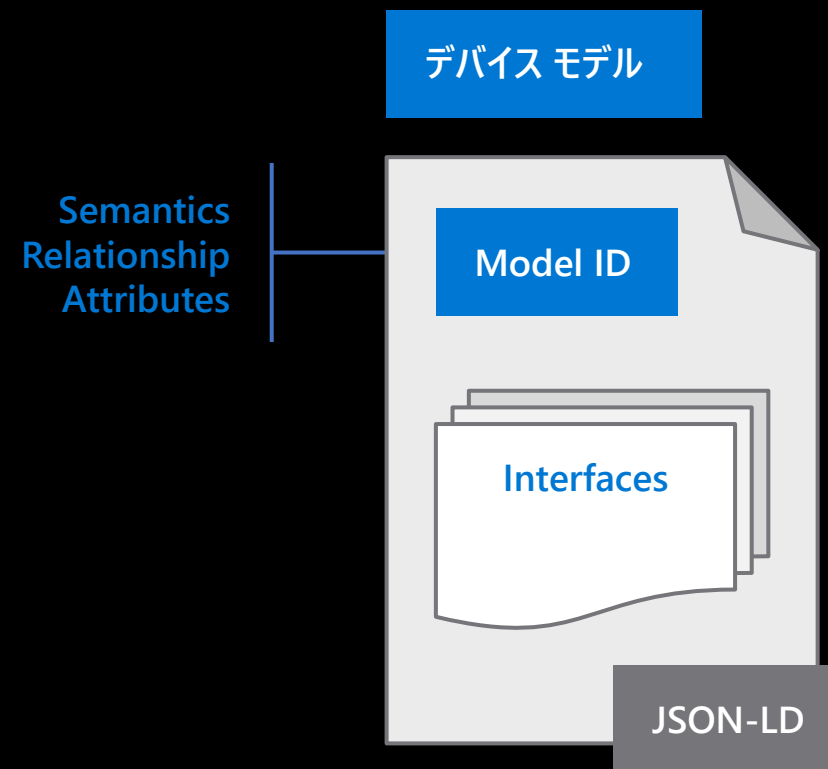


相互作用モデルを理解する鍵となる デバイスモデル

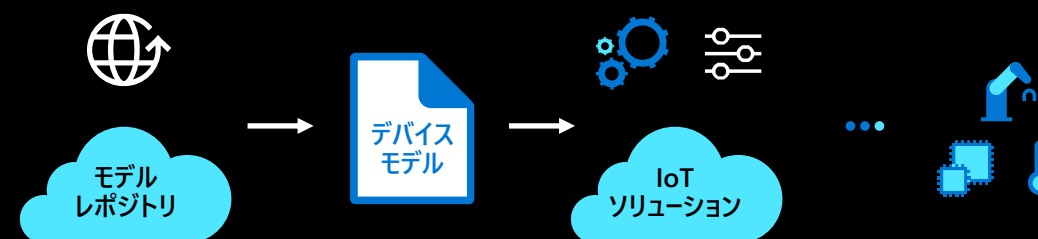
IoT Plug and Play が デバイス や データ に メタデータ を追加する機能を提供

デジタルツインモデルは セマンティクス, 関連性, 属性で記述され、ソリューション開発者が相互作用モデルを理解出来るようにすることで、事例やユーザー経験に革新をもたらすことが可能

データの価値 は IoT Plug and Play が実現する機能を通じて、デバイスを解釈し、処理し、相互作用させることから発生



デバイスからソリューションへ、IoT Plug and Play の通信方法



デバイスビルダー

モデル ID でデバイスを IoT ソリューション へ接続,
IoT Plug and Play が デバイスの機能と属性をクラウドへ通信

- Digital Twin Definition Language version 2 (DTDL v2) を用いて デバイスモデル を記述
- デバイスモデル に基づき IoT Plug and Play 規約を実装
- 接続時に IoT Hub / Device Provisioning Service (DPS) に対してモデル ID をアナウンス

ソリューションビルダー

デバイスモデル を介した 相互運用モデル を通じて モデル を運用,
IoT ソリューション はモデル ID を用いて 相互運用モデル を理解

- IoT デバイスと IoT データのメタデータとして モデル ID を取得
- モデル ID を用いて モデルレポジトリ から デバイスモデル を取得
- デバイスモデル を解釈して、相互運用モデル を理解

IoT Plug and Play 提供 早見表



Azure における IoT
Plug and Play 対応

複数の Azure IoT サービスにおいて IoT Plug and Play 対応を実装



実装を開始する
パートナー向けコンテンツ

取りかかるための
オンラインコンテンツ集

ブログ, チュートリアル,
クイックスタート, 動画



確信をもたらすための
認証プログラム

高い確信を持って IoT
Plug and Play デバイス
を構築し、オンラインの
marketplace を通じて
世界中のオーディエンス
にリーチ

認証申請 受付中!



開発者を支援する
SDKs

7つの言語に対応した
SDK を提供

デバイスビルダー
向けの デバイス SDK

ソリューションビルダー
向け サービス SDK

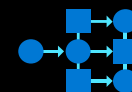


開発者を支援する
ツールやサンプル

それぞれの SDK 向け
サンプルコード
(ドキュメント付属)

デバイス動作確認用
Azure IoT Explorer

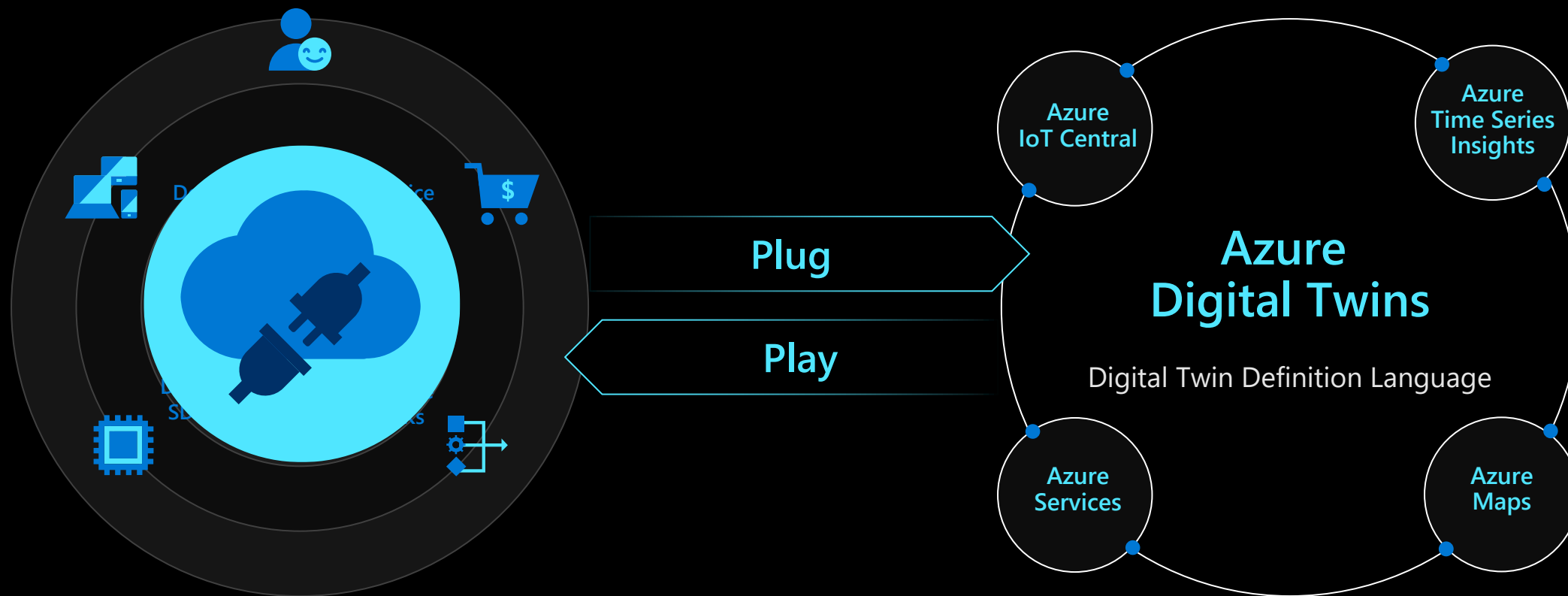
デバイスモデルのオーサリ
ングをサポートするDTDL
拡張機能とパーサー



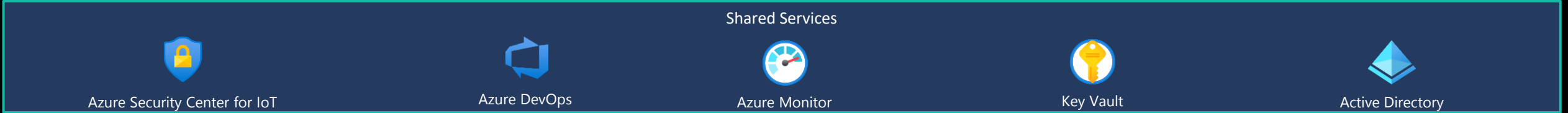
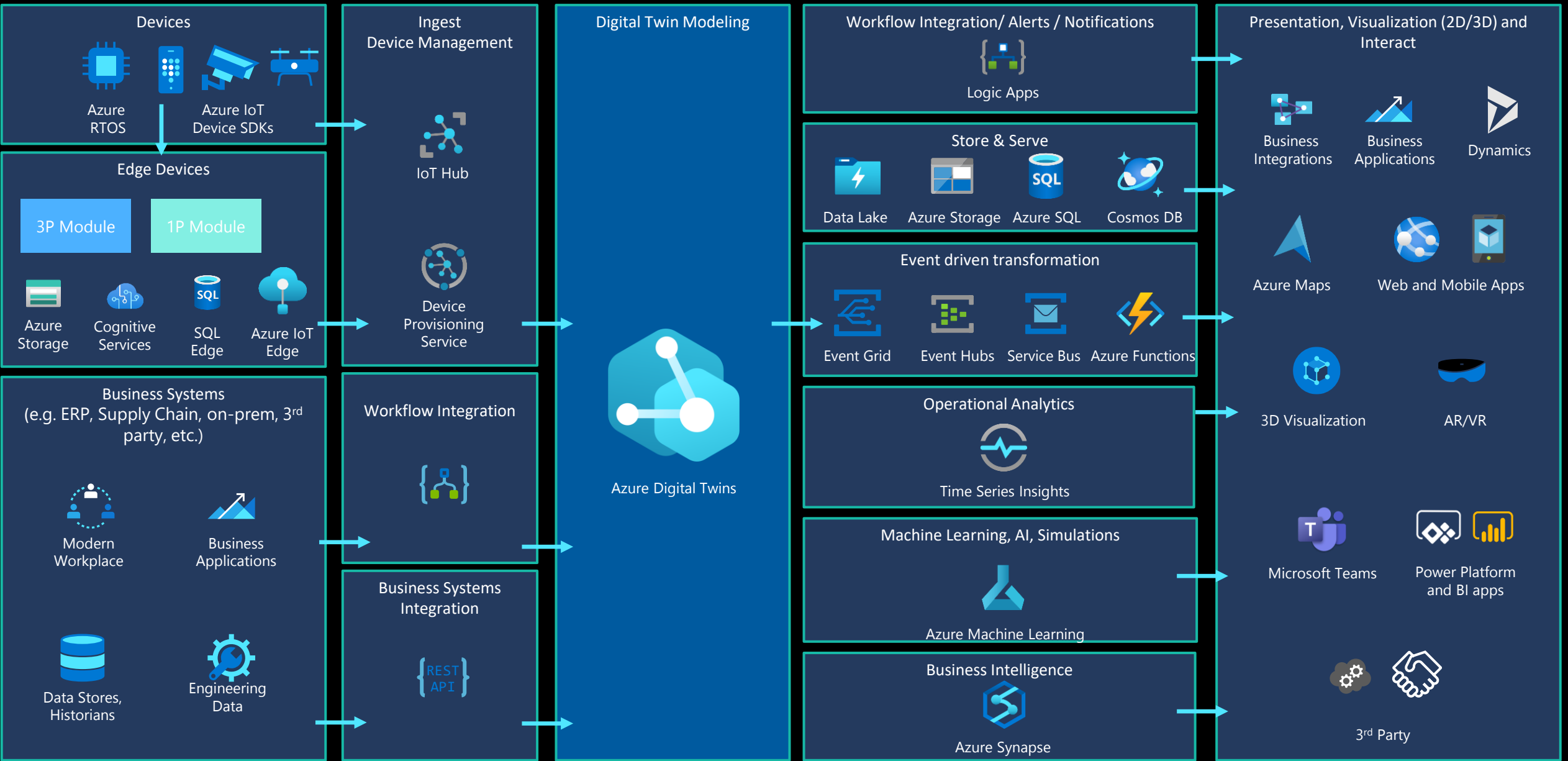
エコシステム構築に
向けたパートナー連携

エコシステム構築のため、
パートナーと協業

エコシステムの 構築 : Microsoft Azure サービスでレイヤリング

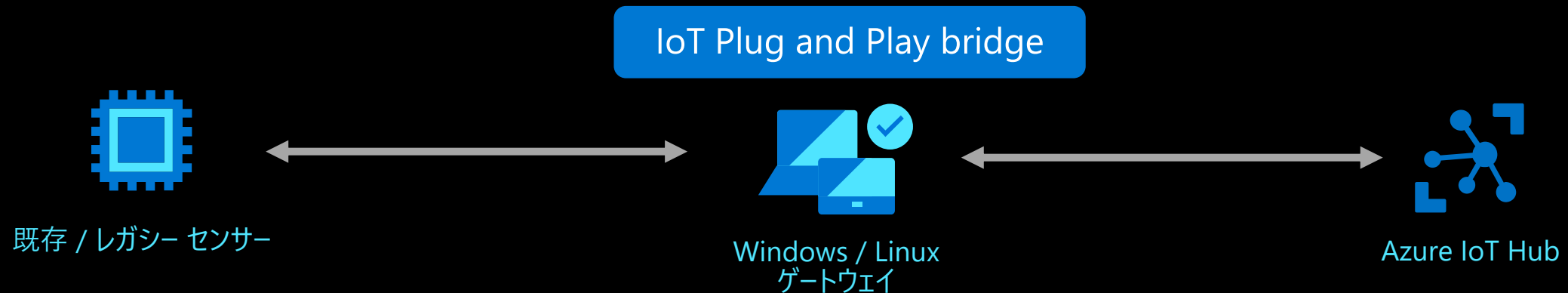


Raw Data を意味のある Insight に変える



IoT Plug and Play bridge

- Windows や Linux のゲートウェイに接続された既存のデバイスを IoT Plug and Play デバイスとして接続するための、オープンソース アプリケーション
- IoT Plug and Play bridge は以下を提供します:
 - Windows / Linux ゲートウェイに接続されたセンサーを、一切の追加コードを書くことなく IoT Plug and Play デバイスに変換
 - プロプライエタリなプロトコルをサポートするために、拡張可能なオープンソースのフレームワーク
 - デバイス と クラウドSDKの 両方に書き込む必要がなく、既存OSの機能を活用



IoT Plug and Play Bridge - アダプタ サポート

Adaptor Protocol	Target OS	
	Windows	Linux
Bluetooth LE Advertisement	Complete	Not Planned
Camera	Complete	Not Planned
Modbus	Complete	Complete
MQTT	Complete	Complete
Serial	Complete	Complete
Windows-PnP (Core Device Health)	Complete	N/A
REST	Planned	Not Planned
WMI	Planned	N/A

ソフトウェア開発キット (SDK) で IoT 開発者を 支援

Azure で構築された ソリューション の開発をシンプル化、そして加速

- IoT Plug and Play には必須ではありませんが、強く推奨

デバイス ビルダー

デバイス SDKs 一般提供 

- C
- .NET
- Python
- Java
- Node.js

ソリューション ビルダー

サービス SDKs 一般提供 

- .NET
- Java
- Python
- Node.js

リソースに制約のあるデバイス向け SDKs 一般提供 

- Embedded C
- Azure RTOS

* [コードサンプル](#) + IoT Plug and Play 規約を実装する方法を示す [関連ドキュメント](#) を含む

ツールで開発者を 支援

Azure IoT Explorer (new)

IoT Plug and Play デバイスを操作、テストするためのグラフィカルなツール

DTDL パーサー ライブラリ

IoT アプリケーションのデバイスモデルのオーサリングと、IoT アプリケーション内でのデバイスモデルの利用をサポート

Azure IoT CLI 拡張

IoT Hub などの Azure リソースを管理するためのオープンソースでクロスプラットフォームなコマンドラインツール

DTDL 拡張

構文の検証と Intellisense を備えた Visual Studio 、及び Visual Studio Code の拡張機能の活用

IoT Plug and Play Talk/Ask with Partner - Seeed Japan

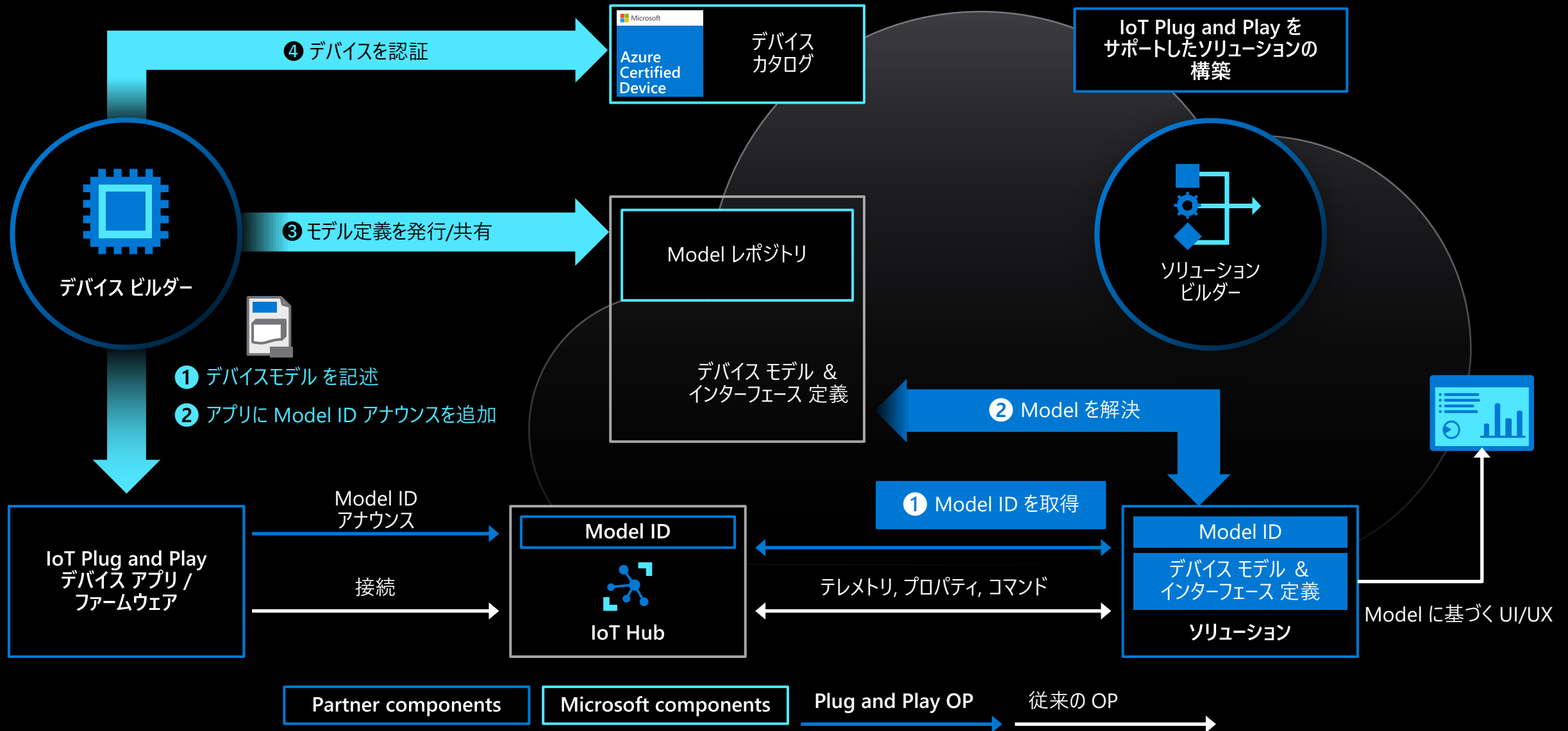
Takashi Matsuoka, Developer

(break) - 10:30-10:40

IoT Plug and Play technical deep dive

Daisuke Nakahara, Principal IoT Solution Architect

IoT Plug and Play プロセス 概要



IoT Plug and Play デバイス モデル

デバイス モデル

- タイプ、スキーマ、セマンティクス を使用して、デバイスの機能と属性を説明
- 1つまたはそれ以上のインターフェースで構成
- Digital Twin Model ID (DTMI) で一意に識別

インターフェース

- デバイスモデル のトップレベル アイテム
- テレメトリー、プロパティ、コマンド、そして コンポーネント でデバイスを記述

```
@context: DTDL2
@id: Model ID
@type: Interface
Contents [
```

```
  @type: Telemetry
```

```
  @type: Property
```

```
  @type: Command
```

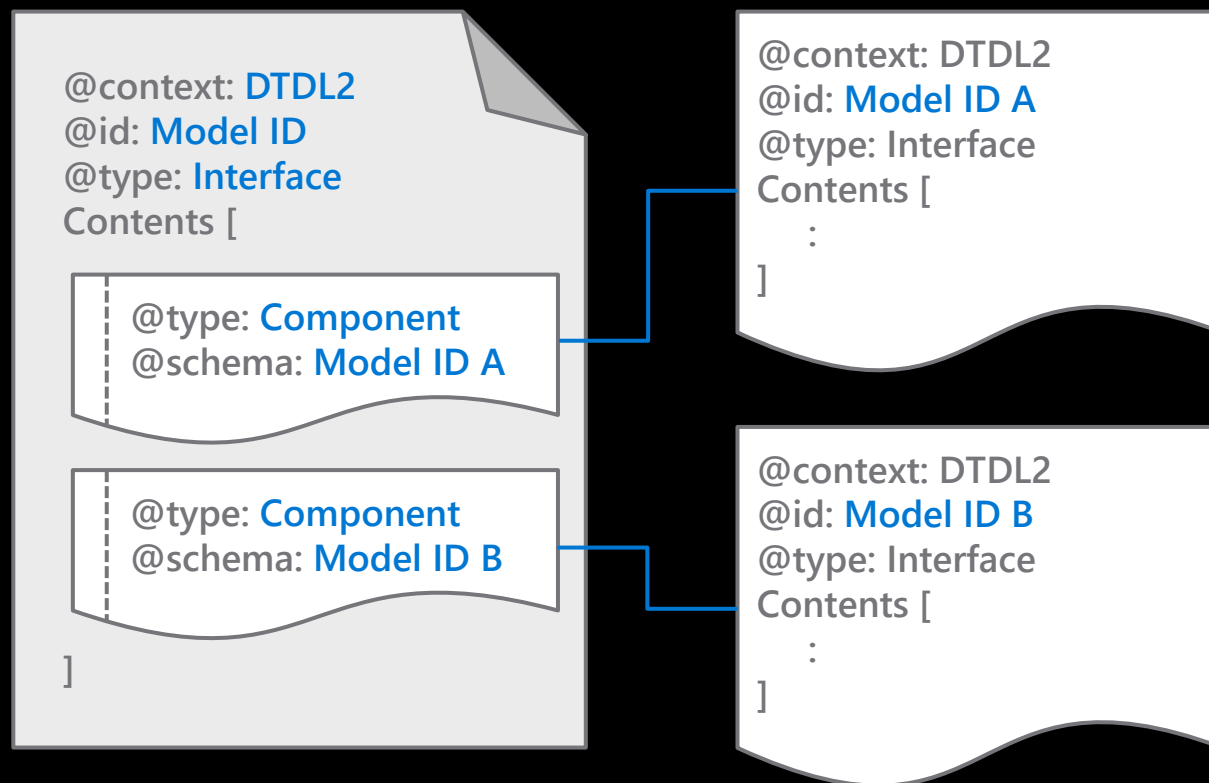
```
  @type: Component
```

```
]
```

IoT Plug and Play デバイスモデルのコンポーネント化

他のインターフェースの“集合体”としての デバイスモデル インターフェースの作成が可能

標準 デバイスモデルは 単一 インターフェース (または コンポーネントレス) モデル



IoT Plug and Play デバイスモデル のオーサリング

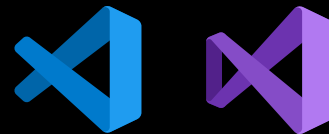
IoT Plug and Play デバイス モデル = Digital Twin モデル

- スキーマは デバイスの 機能 と 属性 を記述
- JSON-LD に基づく DTDL v2 で記述
- DTDL v2 が スキーマ と セマンティクス を管理
<https://aka.ms/dtdl>

デバイスモデル オーサリングのためのツール

- Visual Studio / Visual Studio Code
 - モデルオーサリングのための拡張
- DTDL Parser Library
 - syntax チェック

```
{
  "@context": "dtmi:dtdl:context;2",
  "@id": "dtmi:com:example:Thermostat;1",
  "@type": "Interface",
  "displayName": "Thermostat",
  "description": "IoT Plug and Play Device Model for Thermostat",
  "contents": [
    {
      "@type": [
        "Telemetry",
        "Temperature"
      ],
      "name": "temperature",
      "displayName": "Temperature",
      "description": "Temperature in degrees Celsius.",
      "schema": "double",
      "unit": "degreeCelsius"
    }
  ]
}
```



This new industry standard is a collaborative effort between Microsoft and the [Digital Twin Consortium](https://digitaltwins Consortium).

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"}`

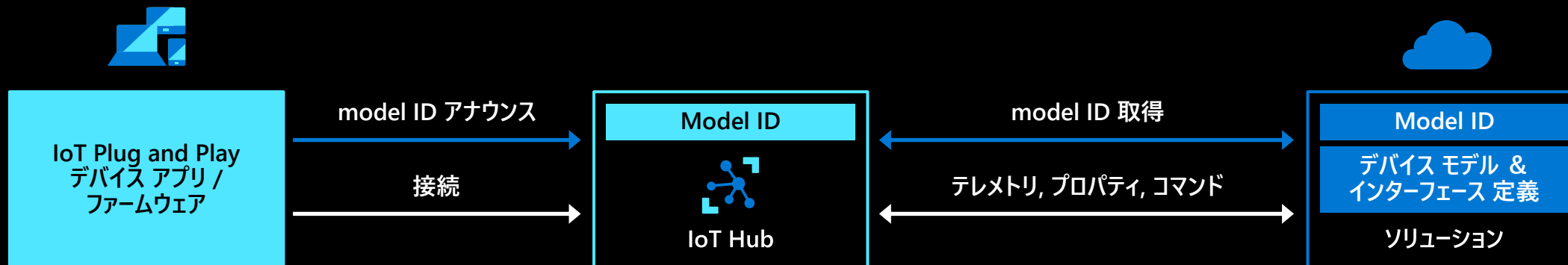
デバイスモデル 無し
"temp が 12.34 "

デバイスモデル 有り
" 温度が 摂氏12.34度 "

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

Model ID アナウンス

既存 APIs を利用



デバイスの通信方法

デバイスは IoT Hub DPS へのプロビジョニング、または IoT Hub へ接続中に model ID を **既存の** API を用いてアナウンス

例 : IoT Hub

```
IoTHubDeviceClient_LL_SetOption(deviceHandle,  
                                OPTION_MODEL_ID,  
                                "dtmi:com:example:IoTPnPDevice;1");
```

例 : DPS

```
Prov_Device_LL_Set_Provisioning_Payload(provDeviceHandle,  
    {"modelId": "dtmi:com:example:IoTPnPDevice;1"});
```

ソリューションの model ID 取得方法

IoT Hub が model ID を受け取り、Device Twin と Digital Twin に保存

Device Twin

```
{  
  "modelId": "dtmi:com:example:IoTPnPDevice;1",  
  "deviceId": "IoTPnPDevice",  
}
```

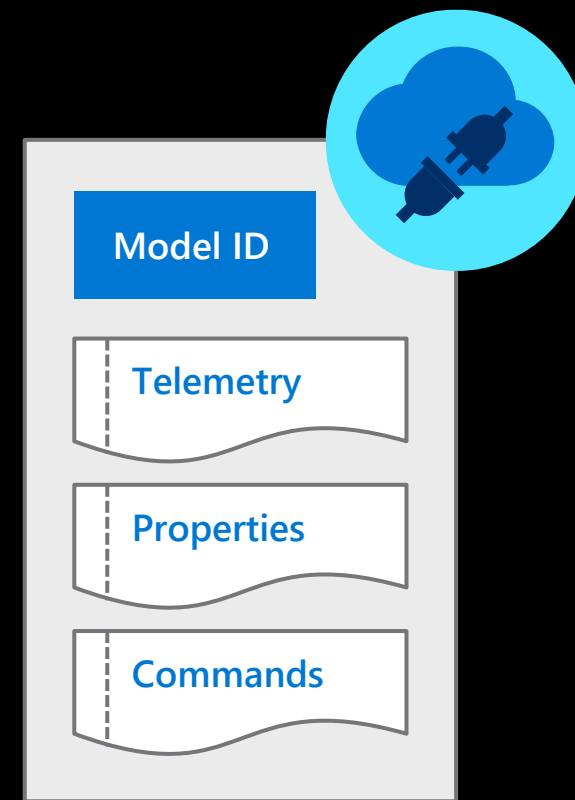
Digital Twin

```
{  
  "$dtId": "pnp-thermostat136",  
  "DisplayName": "Thermostat in Room 136",  
  "$metadata": {  
    "$model": "dtmi:iotpnpadt:DigitalTwins:IoTPnPDevice;1",  
    :  
  }  
}
```

IoT Plug and Play デバイス アプリ / ファームウェア 開発

デバイスモデルに基づく IoT Plug and Play 規約の実装

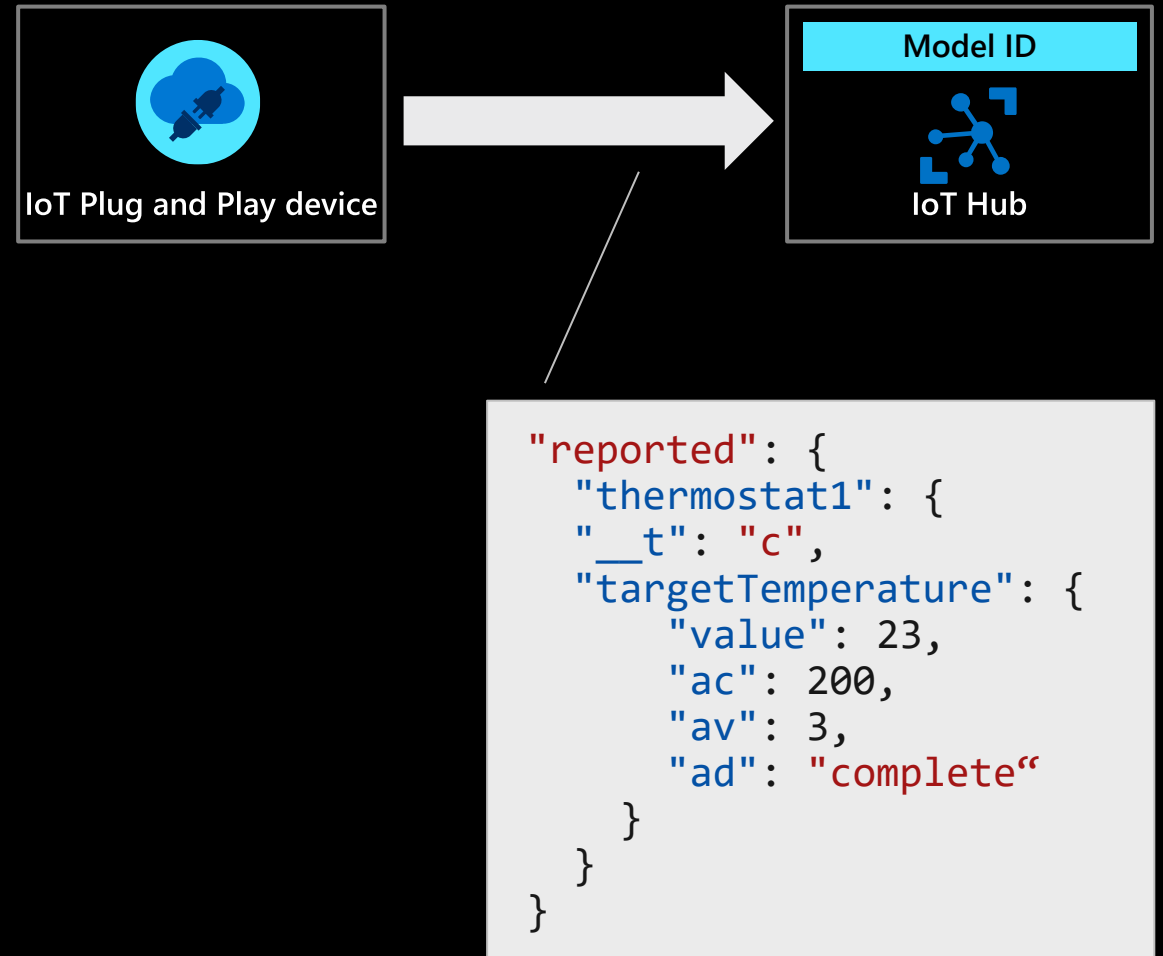
- Model ID アナウンス: 最低限必要
- テレメトリー – Device to Cloud データ
- 書き込み可能 プロパティ – From Cloud to Device 設定
- 読み取り専用 プロパティ – デバイスによって報告
- コマンド – クラウドからデバイス上のメソッドを呼び出す
- IoT Plug and Play 規約に準拠



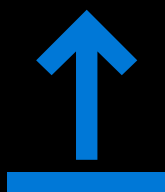
IoT Plug and Play 規則

IoT Plug and Play 規則

- デバイスは IoT Hub とメッセージを交換する際、必須要件として規約に従う
- それにより デバイスモデル とメッセージ内のメタデータの一貫性を確保
- [IoT Plug and Play 規約 の詳細はこちら](#)

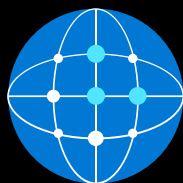


モデルの公開



デバイスモデルを IoT ソリューションから利用出来るようにする必要がある

- Azure IoT デバイスモデル レポジトリ により
デバイスビルダーが IoT Plug and Play
デバイスモデルを 公開、共有可能に



モデル レポジトリ のオプション

- [パブリック デバイス モデル レポジトリ](#)
Microsoft によってホストされ、GitHub の
プルリクエスト 検証ワークフローを利用
- [カスタム デバイス モデル レポジトリ](#)
選択されたストレージによる カスタムモデル
レポジトリをサポート



IoT Plug and Play デバイス認定における要件

- デバイスモデルを
[パブリック デバイス モデル レポジトリ](#) に公開
- エンドカスタマー が デバイスモデル に
アクセス可能にする

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

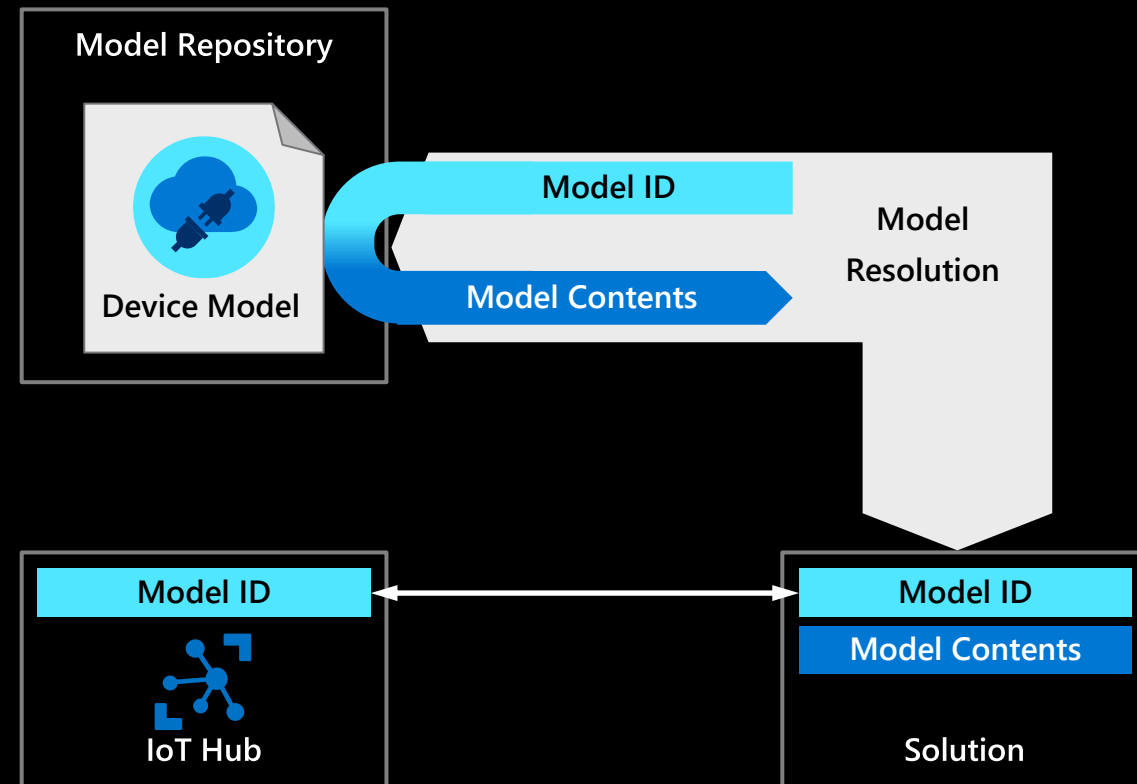
- Azure IoT デバイスモデル レポジトリを介して
- GitHub やファイル共有のような独自のレポジトリを介して

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



オンライン リソース

Published new online resources

- Blogs, documentation, and IoT Show videos

Blogs

- [IoT Plug and Play Public Preview](#)
- [IoT Plug and Play Certification program](#)
- [Connect existing sensors with IoT Plug and Play bridge](#)

IoT Show video series

- [Add "Plug and Play" to your IoT solutions](#)
- [Prepare and certify your devices for IoT Plug and Play](#)
- [Deep Dive : Build and certify your devices through the Azure Certification Device Program](#)

Documentation

- [IoT Plug and Play documentation](#)
- [IoT Plug and Play certification tutorial](#)

Sample code

[Sample code available for supported languages](#)

Azure Certified Device Program

Koichi Hirao, Senior Program Manager

Microsoft による IoT を加速させる 包括的なアプローチ



クラウド
ソリューション



Azure IoT Central



Azure IoT Edge



Azure IoT Hub



Azure Digital Twins



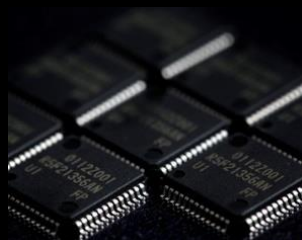
Azure Time Series Insights



Azure Maps



デバイス
サポート



Sensors + control



IoT devices
Azure IoT Device SDK



Sensors to interactive

IoT Edge devices
Azure IoT Edge

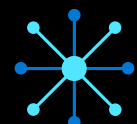


Edge appliances
Azure Stack Edge



Integrated platform

Edge stack
Azure Stack Hub



認定
プログラム




← Azure Certified Device →

← Edge Managed →

← IoT Plug and Play →

Azure Certified Device Program 概要

Azure Certified Device Program は 3種類 のデバイス認定を提供中

認定	確約	要件	対象デバイス
 Azure Certified Device	IoT Hub で動作	<ul style="list-style-type: none">• Device to Cloud (D2C) テレメトリー• DPS によるデバイスプロビジョニング	<ul style="list-style-type: none">• マイクロコントローラー• IoT デバイス• IoT Edge デバイス• Edge アプライアンス• Edge Stack
 Edge Managed	IoT Edge ランタイム が動作	<ul style="list-style-type: none">• DPS によるデバイスプロビジョニング• Moby 上で IoT Edge ランタイムが動作• Moby & Edge セキュリティアマネージャーがプリインストール	<ul style="list-style-type: none">• IoT Edge デバイス• Edge アプライアンス• Edge Stack
 IoT Plug and Play – New!	IoT ソリューション のシンプル化	<ul style="list-style-type: none">• IoT Plug and Play の device model 準拠• Azure Certified Device 要件を含む<ul style="list-style-type: none">• D2C テレメトリー• DPS によるデバイスプロビジョニング	<ul style="list-style-type: none">• マイクロコントローラー• IoT デバイス• IoT Edge デバイス

IoT Plug and Play 認定 要件



技術要件

- デバイスとペリフェラルを記述した デバイスモデル
- IoT Plug and Play 規約に準拠した Device to Cloud (D2C) メッセージをサポート
- 1つ以上のデバイス認証タイプによる Device Provisioning Service (DPS) サポート:
 - 対称鍵
 - X.509
 - Trusted Platform Module (TPM)
- Model ID アナウンス
 - DPS による デバイスプロビジョニング中
 - MQTT 接続中
- Microsoft の提供する モデルレポジトリに デバイスモデル を公開

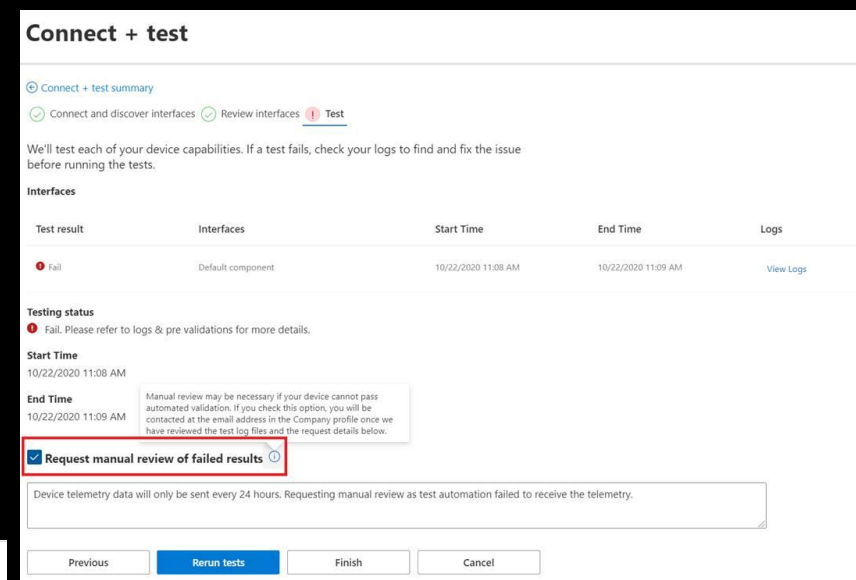
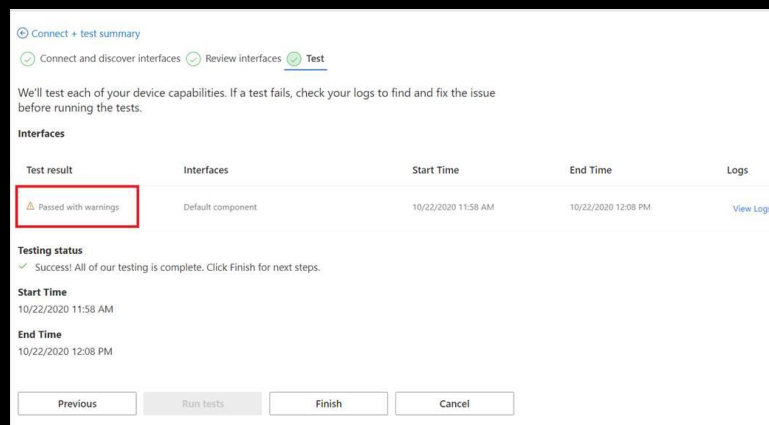


オプション (機能)

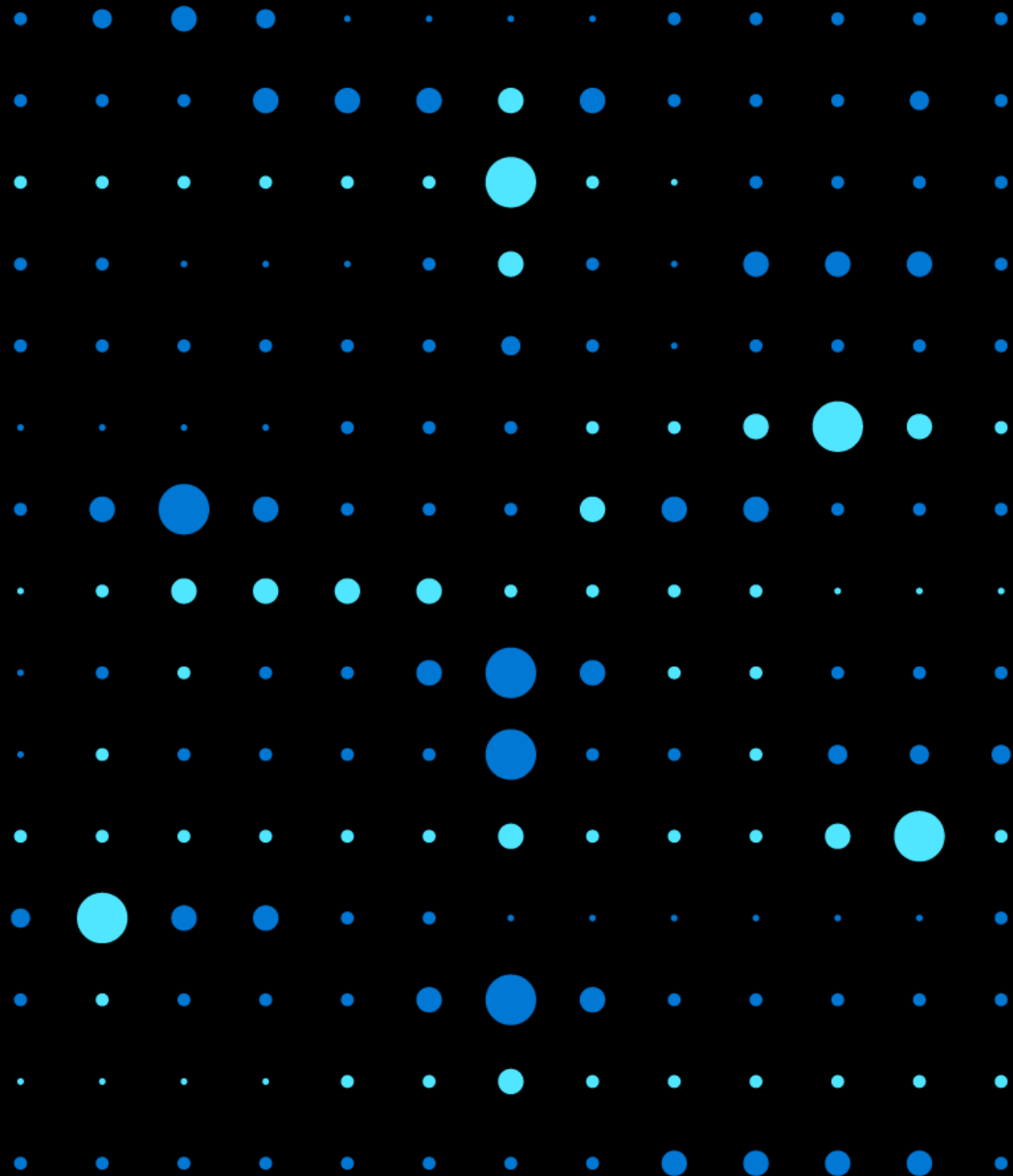
- Cloud to Device (C2D) メッセージ
- デバイス メソッド (または ダイレクト メソッド)
- デバイス プロパティ, 書き込み可能 または 読み取り専用

GA(General Availability) 以降ポータル変更点

- IoT Central 未サポートData Type対応(Exemption 対応)
- アップデートしたDevice modelの再テスト
- 新Model Repository対応
- Exemption 対応プロセスの追加
- 新ステート: “Pass with warning”追加
- モバイルデバイスへのUI最適化



実演 (ポータル, デバイス スカタログ)



Thank you

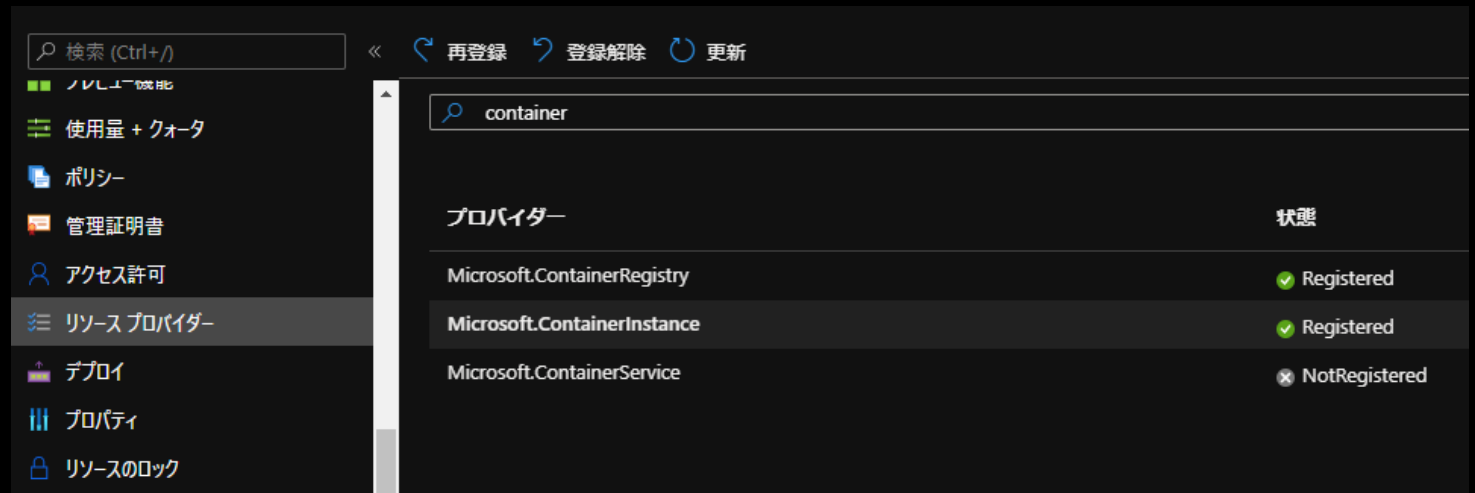


DAY.1 Homework

ワークショップに参加の方は、Azure Portal で以下をご確認ください

サブスクリプション -> リソース プロバイダー -> Container で検索
Microsoft.ContainerInstance が Registered になっていますか？

もしなっていない場合は、Cloud Shell で
az provider register --namespace Microsoft.ContainerInstance
を実行しておいてください

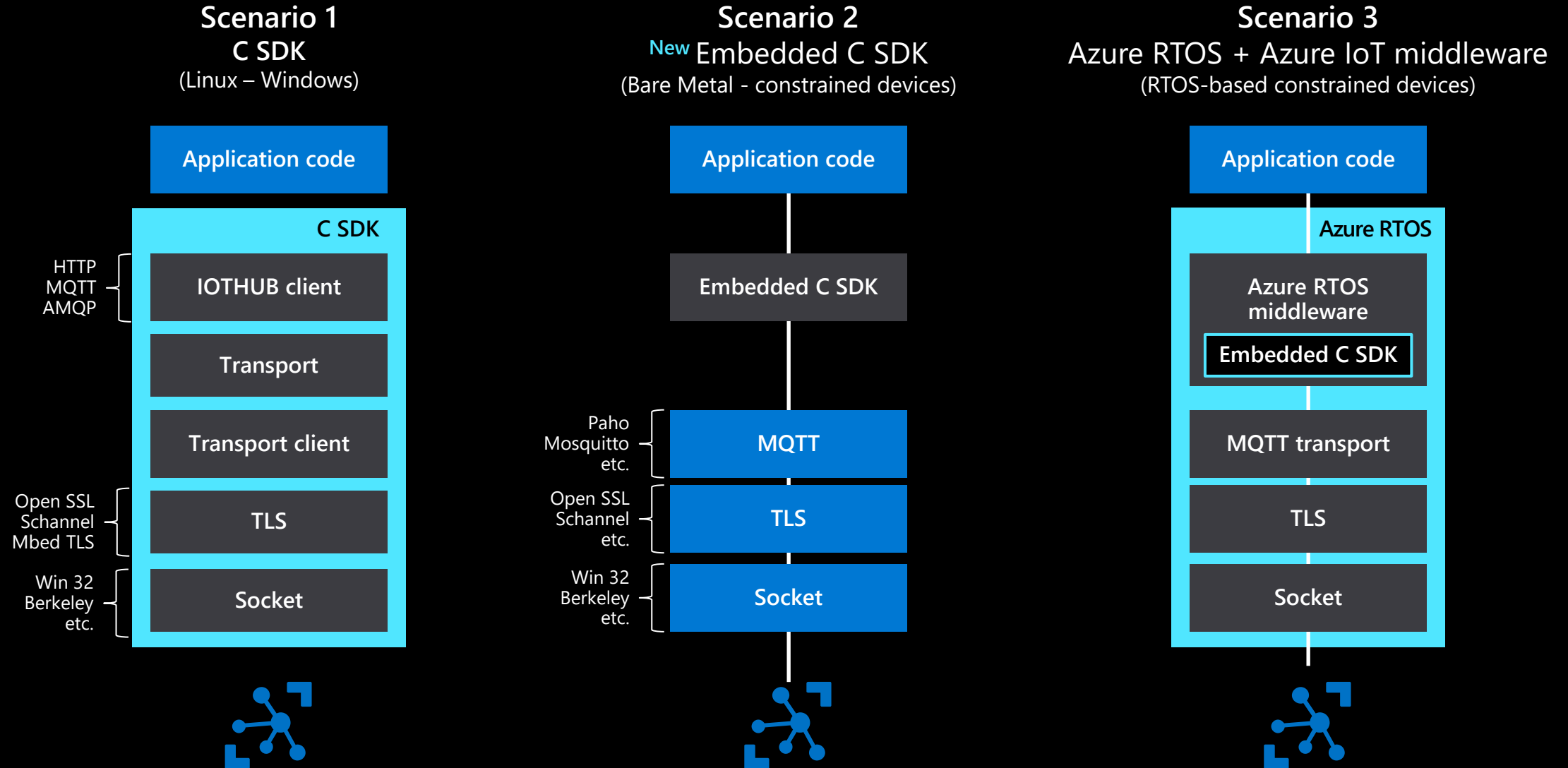


Appendix

Embedded C SDK

Azure IoT Device SDK C

IoT developer's responsibility Microsoft's responsibility



C SDK 推奨

シナリオにより 推奨される C SDK は異なります

シナリオ	プラットフォーム	推奨
MPU-based device	Windows	C SDK
MPU-based device	Linux	C SDK
MCU / constrained device	No RTOS / bare metal	Embedded C SDK
MCU / constrained device	RTOS	Azure RTOS (includes Embedded C SDK)
MCU / constrained device	Mbed	C SDK
Device multiplexing	[any]	C SDK