一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
SC2021-43(2022-03)

異種IoTとプラットフォームの連携を容易化するサービスの研究開発

中橋　友郎† 　陳　　思楠† 　中村　匡秀†,††

† 神戸大学　〒657-8501 神戸市灘区六甲台町 1-1
†† 理化学研究所・革新知能統合研究センター　〒103-0027 東京都中央区日本橋 1-4-1
E-mail: †{tomorrrow,chensinan}@ws.cs.kobe-u.ac.jp, ††masa-n@cs.kobe-u.ac.jp

あらまし　近年より，我が国が目指すべき未来社会として超スマート社会 (Society 5.0) が提唱されている．超スマート社会では，様々な IoT デバイスの活躍が期待され，また種々の IoT デバイスを統合管理し，連携を効率化する土台である IoT プラットフォームの活用も重要視される．しかしながら，IoT プラットフォームと IoT デバイスの連携は容易ではなく，プラットフォームとデバイスの橋渡しをするアダプタの開発や，デバイスをプラットフォームにプロビジョニング (配備) するといった手順が必要である．本研究では，異種 IoT デバイスのプラットフォームへの連携を容易化するサービスを提案することを目的とする．キーアイデアは，IoT デバイスとプラットフォームを仲裁するサービス IoT Mediator の提案である．IoT Mediator は，プラットフォームより送信された制御命令データを変換・規格化することによるアダプタ開発の標準化，またデバイスのプロビジョニング情報の自動生成によるプロビジョニングの簡単化をもって連携を容易化する．提案した IoT Mediator をオープンソースプラットフォーム FIWARE 向けに実装した．ケーススタディでは，夜中に地震が発生した場合を想定した防災アプリの作成を目的とし，IoT Mediator を用いた異種デバイスと FIWARE の連携，FIWARE を介したデバイスの制御を行う．
キーワード　Society 5.0，IoT，IoT プラットフォーム，FIWARE

Study of Service to Assist Platform Deployment of Heterogeneous IoT

Tomoro NAKAHASHI†, Sinan CHEN†, and Masahide NAKAMURA†,††

† Kobe University　Rokkodai-cho 1-1，Nada-ku，Kobe，Hyogo 657-8501 Japan
†† Riken AIP　1-4-1 Nihon-bashi, Chuo-ku, Tokyo 103-0027 Japan
E-mail: †{tomorrrow,chensinan}@ws.cs.kobe-u.ac.jp, ††masa-n@cs.kobe-u.ac.jp

Abstract　In recent years, an ultra-smart society (Society 5.0) has been proposed as the future society that Japan should aim for. In an ultra-smart society, various IoT devices are expected to play an active role, and the use of IoT platforms, which are the foundation for integrated management and efficient coordination of various IoT devices, is also considered important. However, the integration of IoT platforms and IoT devices is not easy, and requires the development of adapters that bridge platforms and devices, and the provisioning of devices to platforms. In this research, we aim to propose a service that facilitates the integration of heterogeneous IoT devices into a platform. The key idea is to propose IoT Mediator, a service that mediates between IoT devices and platforms, to standardize adapter development by converting and standardizing the control instruction data sent from platforms, and to facilitate provisioning by automatically generating device provisioning information. The IoT Mediator simplifies the provisioning process by converting and standardizing the control command data sent from the platform, and by automatically generating the provisioning information of the devices to facilitate the collaboration. In this study, we implemented the proposed IoT mediator for the open source platform FIWARE. The case study aims to create a disaster prevention application that assumes the occurrence of an earthquake in the middle of the night, linking different types of devices with FIWARE using the IoT Mediator and controlling the devices via FIWARE.
Key words　Society 5.0，IoT，IoT platform，FIWARE

1. はじめに

現在の日本社会は，内閣府により**情報社会 (Society 4.0)** [1] と定義づけられている．情報社会とは，インターネットやスマートフォンなどの通信技術の普及により，世界がネットワークで繋がる社会である．そして近年より，情報社会を発展させた**超スマート社会 (Society 5.0)** [2] が提唱されている．情報社会では，サイバー空間に人がアクセスして情報を入手し分析を行うが，膨大な情報の中から有用な情報が見つけれられない，また高齢者や患者など機器を十分に扱えない者は情報を利用しづらいといった課題がある．超スマート社会が実現する社会では，**IoT(Internet of Things)** 技術 [3] により，様々なモノがインターネットに接続される．モノが収集した情報がサイバー空間で分析され，我々に付加価値のある情報として提供する，或いはモノを動作させるなど，モノとモノ・モノとヒトがつながることで，あらゆる人が必要な時に質の高いサービスを享受できるようになる．

またこうしたIoT化の流れを受け，多様なデバイスを統合管理し，連携を効率化するための土台である **IoT プラットフォーム** [4] が提供され始めた．IoT プラットフォームと IoT デバイスが連携することにより，各デバイスの仕様や通信方式の違いが吸収され，共通規格での通信が可能となる．これによりスマートサービス・アプリケーションの生産性が向上し，また分野を超えたデータやサービスの連携が可能になる．

本研究では IoT デバイスとプラットフォームの連携に着目する．実際に IoT デバイスをプラットフォームに連携させる手順は主に２つある．１つ目は，プラットフォームから送られる制御命令を受け取り，デバイスに適した仕様に変換して動作させる，プラットフォームとデバイスの橋渡し役となる**アダプタ**の開発である．２つ目は，連携させるデバイスの情報や対応するアダプタの情報をプラットフォームに**プロビジョニング (配備)** することである．

しかしながら，これらの手順を踏まえてプラットフォームとの連携を実現するには相応の知識が必要であり，容易ではない．プラットフォームとデバイスを橋渡しするアダプタの開発にはプラットフォームとデバイスの両方の知識が必要であり，現状ではアダプタを誰がどのように作るかが明確に決まっていない．また，デバイスのプロビジョニングにはアダプタの情報やプラットフォームの知識が必要であるため煩雑である．

本研究の目的は，異種 IoT デバイスとプラットフォームの連携を容易化することである．そのキーアイデアとして，異種 IoT デバイスとプラットフォームを仲裁するサービス **IoT Mediator** を提案する．IoT Mediator は以下の 4 つの要素で構成される．

A0： アダプタ開発の標準化
A1： アダプタ登録サービス
A2： デバイス登録サービス
A3： コマンドルーチングサービス

プラットフォームによらない標準的なアダプタ開発法を提案することで，アダプタ開発者というアクタを明確化する．また，連携させるデバイスの情報及びアダプタ開発者により IoT Mediator に登録されたアダプタ情報を用いて，システムがプロビジョニング情報を自動生成することによりデバイスのプロビジョニングを簡単化する．

本稿では，提案する IoT Mediator を，我々の研究グループが注目するオープンソース IoT プラットフォーム FIWARE [5] 向けにプロトタイプ実装し，IoT Mediator を利用した異種デバイスの FIWARE への連携，及び FIWARE を利用した異種デバイスを連携動作させるデモアプリの作成をケーススタディとして行った．デモアプリの具体的な内容は，地震発生時の防災を目的としたアプリであり，地震を検知した際に呼び鈴が鳴り，天井照明が点灯し，テレビの情報番組が付き，MMD エージェント [6] が自己防衛を勧告するというものである．ケーススタディより，提案サービスによって FIWARE を意識しない標準的なアダプタ開発を実現し，最低限のデバイス情報の入力で簡単にプロビジョニングを行えたことを確認した．

2. 準　　備

2.1 超スマート社会と IoT

内閣府により現在の日本社会は**情報社会 (Society 4.0)** と定義されている．情報社会とは，インターネットやスマートフォンなどの通信技術の普及により，世界がネットワークで繋がる社会である．そして情報通信技術 (ICT) の発展に伴い，近年より情報社会をさらに発展させた**超スマート社会 (Society 5.0)** が提唱されている．情報社会では，サイバー空間に人がアクセスして情報を入手し分析を行うが，膨大な情報の中から有用な情報が見つけれられない，また高齢者や患者など機器を十分に扱えない者は情報を利用しづらいといった課題がある．一方，超スマート社会は，サイバー空間とフィジカル空間を高度に融合させることを目指しており，IoT(Internet of Things) 技術によりモノとモノ・モノとヒトがつながることで，あらゆる人が必要な時に質の高いサービスを享受できるようになる．

IoT とは，今までインターネットにつながっていなかった様々なモノ (センサ, 建物，車，家電など) がインターネットに接続され，インターネットを経由した通信を行えるようにすることである．現在でも SwitchBot [7] や Nature Remo [8] など，様々な IoT デバイスが各ベンダによって開発・提供されている．IoT デバイスを連携させることにより，交通・製造業・農業・医療・介護といった多分野で活躍するサービスが生み出され，老若男女問わずあらゆる人が恩恵を受けることができる．

2.2 IoT プラットフォーム

今後も様々な IoT デバイスの普及が予期される中，それらを統合管理し，連携を効率的にするための土台となる IoT プラットフォームが提供され出した．現在では多くの IoT プラットフォームが各社より提供されている．中でも，我々の研究グループでは都市 OS の一実装としても利用されるオープンソースプラットフォーム FIWARE を利用した研究が行われている．その他商用のプラットフォームには AWS IoT [9],IoT-EX [10] などがあり，それぞれが強みとする特徴を持つ．IoT プラットフォームは，ベンダによって異なる IoT デバイスの仕様や通信

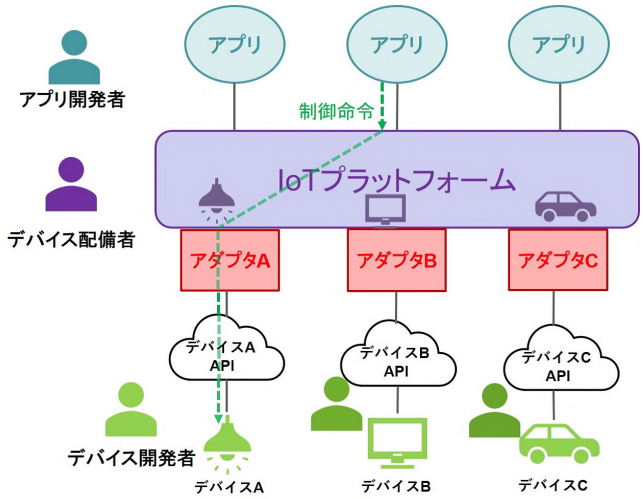

図 1　IoT プラットフォームと異種 IoT の連携アーキテクチャ

方式の違いを吸収し，異種デバイスへの共通規格での通信を実現する．IoT デバイスとプラットフォームが連携することにより，スマートサービス・アプリケーションの生産性が向上する．また，分野ごとに独立したデータやサービスが統合・集約され，分野を超えた連携が可能になる．

2.3　IoT プラットフォームと IoT デバイスの連携

2.3.1　連携の手順

IoT デバイスをプラットフォームにつなげ，IoT プラットフォームを介したデバイスの制御を実現するまでに必要な手順は以下の 2 点である．

(1)　アダプタの開発

(2)　プラットフォームへデバイスをプロビジョニング (配備)

(1) について，仕様の異なる IoT デバイスに対して共通規格での接続を実現するには，共通規格での制御命令を受け取り，デバイスに適した仕様に変換する，プラットフォームとデバイスの橋渡し役がデバイスの仕様ごとに必要になる．この役割を持つコンポーネントを本研究ではアダプタと呼称する．

(2) について，IoT デバイスの情報や，そのデバイスに対応したアダプタの情報をプラットフォームに送信し，登録することをプロビジョニングという．プロビジョニングを行うと，プラットフォーム内にプロビジョニングしたデバイスのエンティティが生成される．このエンティティはデバイスの状態管理や制御命令のルーチングに用いられる．

(1)(2) を踏まえて IoT プラットフォームと IoT デバイスの連携が実現した全体のアーキテクチャを図 1 に示す．図 1 のデバイス A を例に，アプリがデバイスを動作させる流れを示す．始めに，アプリがデバイス A に対する制御命令をプラットフォームに送信する．次に，プラットフォームはデバイス A のプロビジョニング情報を参照してアダプタ A に制御命令を送信する．最後に，アダプタ A はデバイス A の動作 API を呼び出してデバイス A を動作させる．

2.3.2　連携に関わるアクタ

IoT プラットフォームによる異種 IoT デバイスの連携に想定されるアクタは，アプリ開発者・デバイス配備者・デバイス開発者の三者である．デバイス開発者は IoT デバイスを開発した企業・個人等であり，デバイスの操作 API を提供する．デバイス配備者はデバイスの情報やアダプタの情報を用いてデバイスをプラットフォームにプロビジョニングする．アプリ開発者はプラットフォームを利用してアプリを開発する．

異種 IoT デバイスがプラットフォームを通して連携動作するまでの流れとしては，デバイス開発者が各 IoT デバイスを開発，デバイス配備者がそれらのデバイスをプラットフォームにプロビジョニング，アプリ開発者がプロビジョニングされたデバイスに対してプラットフォームを介した通信で動作させるアプリを開発する，という流れになる．アプリ開発者はデバイス毎の仕様の違いを意識せずに共通規格での制御命令をプラットフォームに送信することでデバイスを動作させることができるため，IoT デバイスを用いた様々な目的のアプリを効率的に開発することができるようになる．

2.4　着目する課題

しかしながら，IoT デバイスとプラットフォームの連携は容易ではない．着目する問題点は以下の 2 つである．

P1：　アダプタを誰がどのように作るかが決まっていないこと

P2：　プロビジョニングが煩雑であること

P1 について，プラットフォームとデバイスの橋渡し役であるアダプタの開発にはプラットフォームの知識とデバイスの仕様に関する知識の両方が必要である．現状ではデバイスの仕様に関する知識を持つデバイス開発者が IoT プラットフォーム用にアダプタを開発・提供する場合や，プラットフォームの知識を持つデバイス配備者，またはアプリ開発者が目的の IoT プラットフォームで好みのデバイスを動作させるために自前で開発する場合がある．また，プラットフォームによってアダプタの実装方法もばらばらであり，技術的な責任分界点が曖昧になっている．

P2 について，デバイスのプロビジョニングにはアダプタの情報が必要であることが煩雑さの主な要因になる．アプリからプラットフォームへ送信された制御命令を適切なアダプタへとルーチングするために，プロビジョニングにはアダプタのエンドポイントや制御コマンド群が必要である．デバイス配備者が，既に他者が用意したアダプタを利用してプロビジョニングする場合でも，結局はアダプタの詳細を知る必要があるため，負担がかかる．また，アダプタの種類が増えると情報の管理も煩雑になる．

3.　提 案 手 法

3.1　研究目的とキーアイデア

本研究の目的は，**異種 IoT と IoT プラットフォームの連携を容易化すること**である．本研究のキーアイデアは，異種 IoT デバイスと IoT プラットフォームを仲裁するサービス IoT Mediator の提案である．

3.2　全体アーキテクチャ

図 2 は IoT Mediator を導入した，IoT デバイスとプラットフォームの連携の全体図である．図 2 において，提案する IoT Mediator は IoT プラットフォームとアダプタの間に位置し，

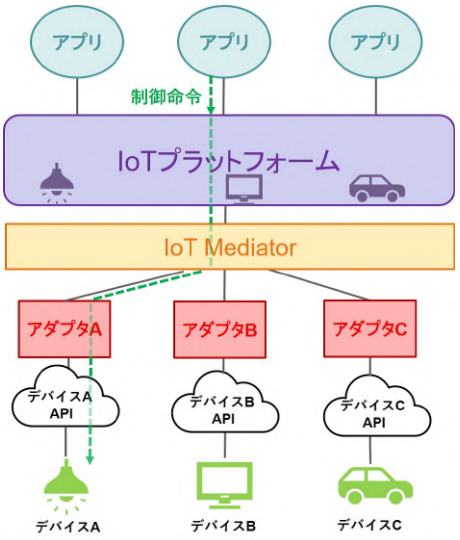

図 2　IoT Mediator 導入後の全体アーキテクチャ

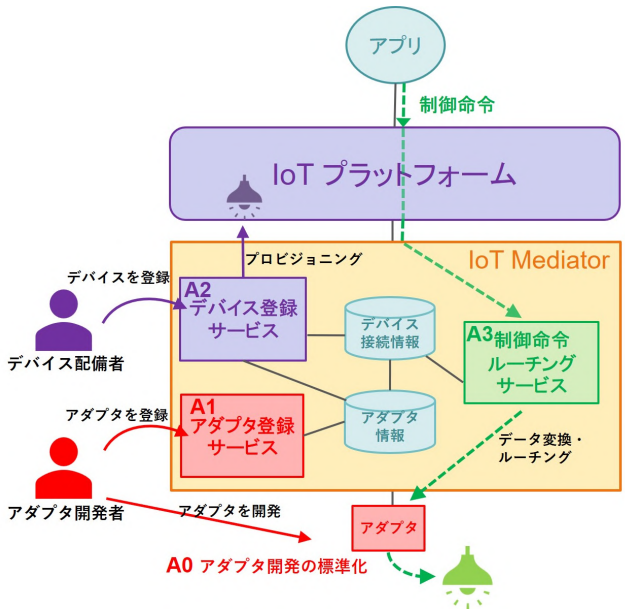

図 3　IoT Mediator アーキテクチャ

制御命令を適切なアダプタへ送信する役割をプラットフォームの代わりに IoT Mediator が担っている．

IoT Mediator は以下の要素で構成される．

A0：　アダプタ開発の標準化
A1：　アダプタ登録サービス
A2：　デバイス登録サービス
A3：　制御命令ルーチングサービス

図 3 は IoTMediator のアーキテクチャである．IoT Mediator はプラットフォームによらない標準的なアダプタ開発を提案することで，アダプタ開発者というアクタを明確化する．アダプタ開発者はデバイスの動作仕様の知識を持ち，アダプタを開発する者である．IoT Mediator の利用者は，アダプタ開発者とデバイス配備者であり，それぞれがアダプタ登録サービス・デバイス登録サービスを利用することで，デバイスとプラットフォームの連携が実現する．

3.3　A0: アダプタ開発の標準化

アダプタの役割はデバイスとプラットフォームを橋渡しすることであるため，アダプタを開発するにはデバイスの知識に加え，プラットフォームの知識が必要である．これはプラットフォームの種類が増えるとアダプタ開発者にとって大きな負担になる．そこで提案サービスでは，アダプタに求める役割を，**ある規格化されたデータを受け取ってデバイスを動作させる**という役割にすることで，プラットフォームによらない標準的なアダプタ開発を提案し，アダプタ開発にかかる負担を低減させる．これを実現させるために，IoT デバイスとプラットフォームを仲裁する IoT Mediator を各プラットフォームに向けて実装し，IoT Mediator がプラットフォームから送信された制御命令データを規格化データに変換する．提案する規格化データの形式は JSON で，含まれる情報は以下のとおりである．

- deviceId: デバイスの識別子
- command: 操作コマンドを表す文字列
- value: コマンドに与えるパラメータ
- metaAttributes: その他必要なメタ情報

metaAttributes は deviceId や command 以外にデバイスを動作させるのに必要な静的情報であり，例としてアクセストークン等が挙げられる．また，データ送信の実行形式は HTTP POST である．IoT Meditator により上記の形式に規格化された制御命令データがアダプタに送信されるため，アダプタ開発者はプラットフォームを意識せずにアダプタを開発できる．

3.4　A1: アダプタ登録サービス

アダプタ登録サービスは，開発したアダプタを様々な利用者が利用できるよう，アダプタ開発者がアダプタ情報を IoT Mediator に登録するサービスである．IoT Mediator ではアダプタの情報を，デバイスの型 (デバイスタイプ) と型の集合 (デバイスグループ) として管理する．デバイスタイプはアダプタが動作させることのできるデバイスの種類である．デバイスの型名，デバイスが受け付けるコマンドリスト，その他動作に必要なメタ情報の属性名を持つ．例えばエアコンの場合，型名「AirConditioner」・コマンドリスト「turnOn,turnOff,setAll」・メタ情報「accesstoken」等である．デバイスグループは同じ会社の製品群など，デバイスタイプをまとめて管理するものであり，グループ名とアダプタのエンドポイントを持つ．例えば SwitchBot 製品に対応するアダプタを用意した場合，グループ名「SwitchBot」・アダプタエンドポイント「http://〜」といったものになり，このデバイスグループに SwitchBot ボットや SwitchBot カーテンといった SwitchBot 製品のデバイスタイプが含まれている．

アダプタ開発者は，開発したアダプタの情報を元にデバイスグループ・デバイスタイプを登録する．登録されたデバイスグループ・デバイスタイプの情報はデバイスのプロビジョニングや，プラットフォームから送信された制御命令をルーチングする際に利用される．

3.5　A2: デバイス登録サービス

デバイスのプラットフォームへのプロビジョニングには，デバイス情報 (deviceId 等) に加え，そのデバイスに対する制御命

令を適切なアダプタへと送るためにアダプタの情報 (エンドポイント，アダプタが受け付ける操作コマンド群等) が必要になる．デバイス配備者はアダプタの情報とプラットフォームの知識が要求されるため，プロビジョニングは容易ではない．デバイス登録サービスは，A1 のアダプタ登録サービスにより IoT Mediator で管理されているアダプタ情報 (デバイスグループ・デバイスタイプ) を用いることで，必要最小限の情報入力でプロビジョニングを簡単に行えるようにするサービスである．具体的には，デバイス配備者はデバイス登録サービス上で，配備したいデバイスに該当するデバイスグループ・デバイスタイプを選択し，デバイス情報を入力する．デバイス登録サービスは，選択されたデバイスグループ・デバイスタイプが持つアダプタ情報と入力されたデバイス情報を元にプラットフォームの仕様に沿ったプロビジョニング情報を生成し，プラットフォームにプロビジョニングする．これにより，アダプタの情報を必要としない，簡単なプロビジョニングが実現する．

また，プロビジョニングすると同時に，デバイス登録サービスはデバイス接続情報というエンティティを生成し，IoT Mediator 内で管理する．このエンティティは入力されたデバイス情報とアダプタ情報を持ち，プラットフォームから送信される制御命令データを A0 で示したデータ規格に変換する際に用いられる．生成したエンティティの ID はプロビジョニング情報に付加されプラットフォームで管理される．

3.6 A3: 制御命令ルーチングサービス

従来では, アプリからプラットフォームに送信された制御命令はプラットフォームが適切なアダプタへルーチングしていた．一方で IoT Mediator を導入することにより，アプリからプラットフォームに送信された制御命令は全て IoT Mediator に送信されるため，IoT Mediator がアダプタへルーチングする役割を代行する必要がある．制御命令ルーチングサービスはプラットフォームから送信された制御命令データを A0 で示したデータ規格に変換し，適切なアダプタへ送信するサービスである．図 4 に制御命令ルーチングサービスのアーキテクチャを示す．データ変換・ルーチングには A2 のデバイス登録サービスで生成され，IoT Mediator で管理されているデバイス接続情報エンティティが用いられる．デバイス接続情報は，制御命令を転送するアダプタのエンドポイントとプロビジョニング時に入力したデバイスの情報を持つ．プラットフォームから送信される制御命令データには，A2 のデバイス登録サービスにより付加されたデバイス接続情報のエンティティ Id が含まれており，ルーチングサービスは Id に紐づいたデバイス接続情報と，制御命令データを用いて規格化データに変換し，適切なアダプタにルーチングする．

3.7 IoT Mediator による異種 IoT の連携

IoT Mediator を用いて，IoT デバイスとプラットフォームの連携からデバイス同士の連携を実現するまでの流れを以下に示す．

Step1: アダプタ開発
Step2: アダプタ登録
Step3: デバイス登録

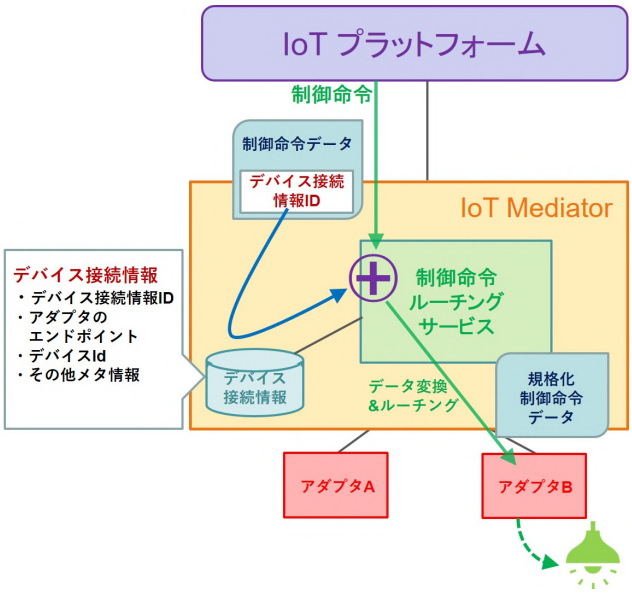

図 4 制御命令ルーチングサービスによるルーチング・データ変換

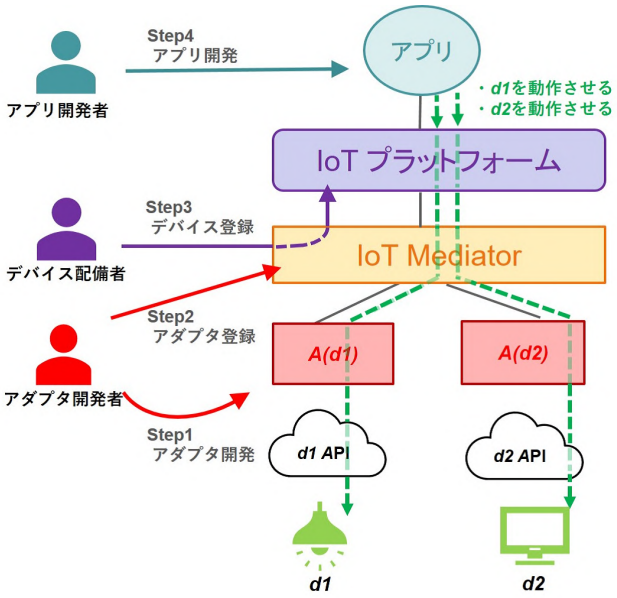

図 5 $d1, d2$ の連携までの流れ

Step4: アプリ開発

ここでは，IoT デバイス $d1, d2$ を IoT Mediator に基づいてプラットフォームに連携させることを考える．図 5 に $d1, d2$ を連携動作させるまでの流れを示す．

Step1 では,$d1$ のアダプタ開発者は A0 で提案したデータ規格に従って $d1$ のアダプタ $A(d1)$ を開発する．Step2 では,$d1$ のアダプタ開発者は A1 のアダプタ登録サービスを用いて $A(d1)$ の情報を元にデバイスグループ・デバイスタイプを IoT Mediator に登録する．Step3 では, デバイス配備者は A2 のデバイス登録サービスを用いてデバイス情報を入力し，プラットフォームにプロビジョニングする．$d2$ についても，Step1〜Step3 を同様に行う．以上により，$d1, d2$ のプラットフォームとの連携が完了する．Step4 では, アプリ開発者は $d1, d2$ を連携動作させるアプリを開発する．プラットフォームに $d1, d2$ が連携している

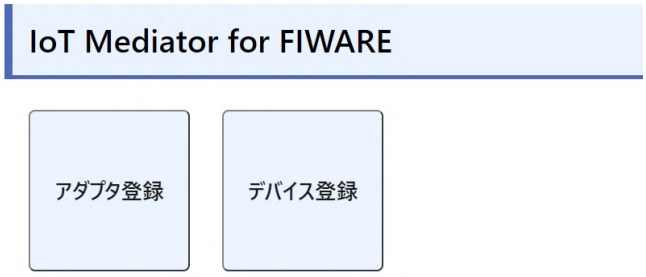

図 6　開 始 画 面

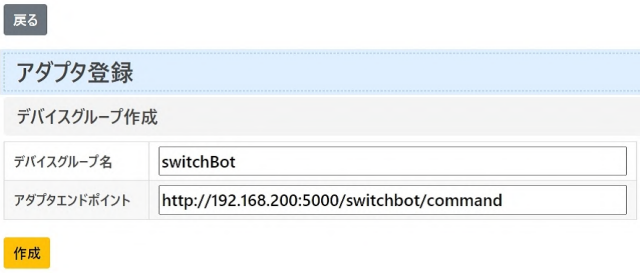

図 7　アダプタ登録: デバイスグループ作成画面

ことにより，アプリ開発者は $d1, d2$ それぞれの仕様を意識せず，共通規格の制御命令をプラットフォームに送信して動作させることができる．制御命令の流れを $d1$ を例として示す．$d1$ への制御命令をプラットフォームに送信すると，プラットフォームはデバイス接続情報エンティティの Id とともに IoT Mediator に制御命令を送信する．IoT Mediator のコマンドルーチングサービスは，受け取ったデバイス接続情報エンティティの Id を参照してデータ変換し，$A(d1)$ に送信する．$A(d1)$ は受け取った規格化データをもとにデバイスを動作させる．

4.　実　　装

4.1　利用した技術

本稿では，提案する IoT Mediator を FIWARE プラットフォームに向けてプロトタイプ実装した．バックエンド実装は Java 言語と SpringBoot を用いた．データベースサーバは MySQL を利用し，Web サーバとして Apache Tomcat を利用した．また，アダプタやデモアプリ等のサーバアプリは Node.js を用いて実装した．

4.2　画面の説明

実装した IoT Mediator はアダプタ登録サービス・デバイス登録サービスの UI を提供している．図 6 は本サービスの開始ページである．アダプタ開発者は開始ページからアダプタ登録に進み, デバイスグループの作成 (図 7)・デバイスタイプの作成 (図 8) を行う．デバイス配備者は開始ページからデバイス登録に進み，登録するデバイスに該当するデバイスグループを選択 (図 9), デバイスグループ内のデバイスタイプを選択し (図 10), 新規登録画面 (図 11) でデバイスの情報を入力して登録ボタンを押すことでプラットフォームにプロビジョニング，またデバイス接続情報のエンティティがシステム内に生成される．

図 8　アダプタ登録: デバイスタイプ作成画面

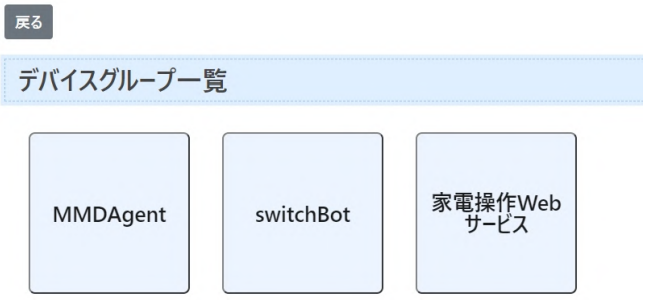

図 9　デバイス登録: デバイスグループ選択画面

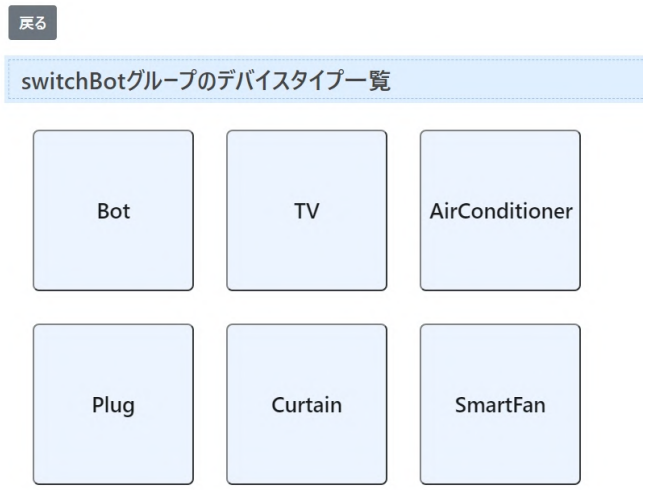

図 10　デバイス登録: デバイスタイプ選択画面

5.　ケーススタディ

5.1　アプリケーションの概要

ケーススタディでは，FIWARE を利用した異種 IoT の同時制御を行うデモアプリの制作を目的とし，各 IoT の FIWARE との連携を 4. で実装した IoT Mediator for FIWARE を用いて

図 11　デバイス登録: 新規デバイス登録画面

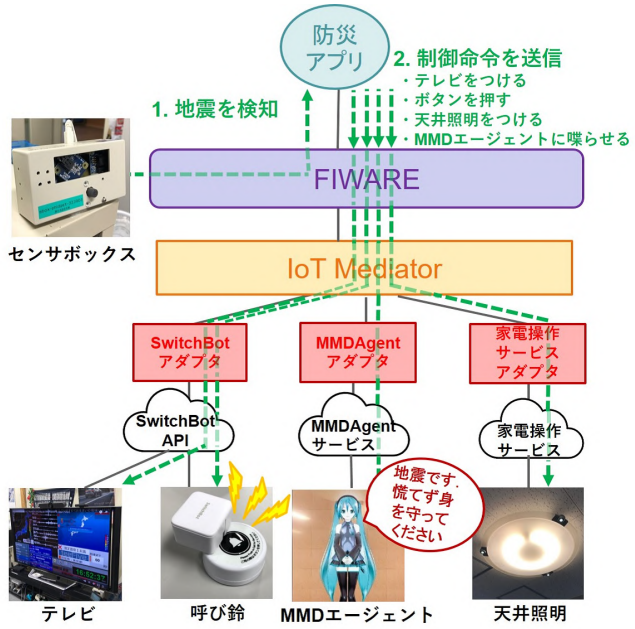

図 12　防災アプリの動作の流れ

行う．具体的なアプリの内容は，夜中に地震が発生した際，研究室の天井照明が点灯し，テレビの情報番組が付き，呼び鈴が鳴り，MMD エージェントが自己防衛を促す防災アプリである．図 12 に防災アプリの動作の流れを示す．地震の検知は我々の研究グループで制作された環境センサ “センサボックス “の振動センサの測定値を逐時 FIWARE に送信し，FIWARE が提供するサブスクリプション機能により，測定値が一定の水準を超えるとアプリに通知をすることで実装する．テレビの操作は SwitchBot hub を用いた赤外線通信で行い，呼び鈴のボタン押下は SwitchBot Bot で行い，天井照明・MMD エージェントの操作は我々の研究グループが開発した操作サービスの WebAPI を用いて行う．

5.2　IoT Mediator を用いた実装

5.2.1　Step1: アダプタ開発

SwitchBot 用アダプタ・MMD エージェント操作サービス用アダプタ・家電操作サービス用アダプタをそれぞれ作成した．各アダプタは Node.js による Web サーバとして IoT Mediator から送信される A0 の規格化データを待ち受け，それぞれが受け持つサービスの API を呼び出してデバイスを動作させる．

5.2.2　Step2: アダプタ登録

各アダプタの情報を元に，アダプタ登録サービスにデバイスグループ・デバイスタイプを作成する．デバイスグループの作成画面は図 7, デバイスタイプ作成画面は図 8 のとおりであり，以下に SwitchBot 用アダプタを用いたデバイスグループ・デバイスタイプ作成時の入力情報を示す．

デバイスグループ作成

- グループ名
  - SwitchBot
- アダプタエンドポイント
  - http://192.168.0.128:5000/switchbot/command

デバイスタイプ作成 (テレビの場合)

- デバイスタイプ名
  - TV
- 受け付けるコマンド
  - `turnOn,turnOff,setChannel,volumeAdd, volumeSub,channelAdd,channelSub`
- その他メタ情報
  - accesstoken

5.2.3　Step3: デバイス登録

Step2 により，デバイス登録サービスの画面に各デバイスグループとデバイスタイプを選択する項目が追加される．プラットフォームにプロビジョニングしたいデバイスに当てはまるデバイスグループ名とデバイスタイプ名を選択し，deviceId などの必要な情報を入力・フォーム送信する．以下に，防災アプリで動作させるテレビをデバイス登録サービスによりプロビジョニングする流れを示す．

1. デバイスグループ “SwitchBot” を選択 (図 9)
2. デバイスタイプ “TV” を選択 (図 10)
3. 新規登録フォームにデバイス名・deviceId・accesstoken を入力し，フォームを送信する (図 11)
4. IoT Mediator が入力情報を元に FIWARE にプロビジョニング

5.2.4　Step4: アプリ開発

Step1〜Step3 により，IoT Mediator を用いた IoT デバイスとプラットフォームの連携が完了した．これにより，FIWARE に対して共通規格での制御命令による異種デバイスの制御が可能になった．具体的な制御方法としては，FIWARE に向け Http-Patch リクエストを送信し，プロビジョニングにより FIWARE に生成されたエンティティのコマンド属性を更新することで，FIWARE の内部機能により IoT Mediator へ制御命令が送信される．開発した防災アプリは Node.js による Web サーバとして，FIWARE からセンサボックスの振動センサの

測定値が一定水準を超えた通知を受け取り，FIWARE に生成されたテレビ・MMD エージェント・天井照明のエンティティに対して Patch リクエストを送信する．実際に送信されるリクエスト body の内容を以下に示す．

テレビ

```
% テレビをつける
{"turnOn":  {"type:"command","value":""}}
% チャンネルを変える
{"setCannel":  {"type:"command","value":"1"}}
```

呼び鈴

```
% 呼び鈴を鳴らす (Bot によりボタンを押下する)
{"press":  {"type:"command","value":""}}
```

MMD エージェント

```
% 自己防衛を喚起する
{"startSpeech":  {"type:"command","value":"地震です．慌てずに身を守ってください．"}}
```

天井照明

```
% 天井照明を点灯する
{"on":  {"type:"command","value":""}}
```

このように，本来は動作仕様が異なるテレビ・呼び鈴・MMD エージェント・天井照明を FIWARE に対する共通規格の通信で動作させることができた．

5.3 考　　察

以上により，IoT Mediator を用いた IoT デバイスと FIWARE の連携，FIWARE を利用した防災アプリの開発・動作を確認した．章 2.4 の P1 への回答として，IoT Mediator の導入により FIWARE の知識を要さないアダプタ開発を実現したことで，デバイスの仕様に関する知識のみを持つ者がアダプタ開発者としてアダプタ開発を行うことが期待できる．章 2.4 の P2 への回答として，デバイス登録サービスによるアダプタ情報入力の省略・プロビジョニング情報の自動生成により煩雑さを低減した．

さらに，MMD エージェントのように動作 API を提供しているものであれば，物理的なデバイスではない Web サービスも FIWARE と連携が可能であることが確認できた．同様に，WebAPI が公開されているモノ・コトに対しても，提案手法を適用することで，異種 IoT とクラウドサービスの豊かな連携が期待できる．より高度なサービス連携とその評価については今後の課題とする．

また，アダプタ開発者がデバイスグループやデバイスタイプを作成する際に、別の開発者によって再利用されることを意識した名前付けを行う必要がある．しかしながら，現状は命名規約についての規定は提案サービスでは考慮しきれていない．その結果，同種のデバイスのタイプに対しても表記ゆれが発生するため，再利用がしにくくなるという問題が生じる．この問題に対しては，あらかじめ IoT Mediator が標準的な型名を準備し，開発者に選択してもらうことで軽減すると考えられるが，これについても今後の課題としたい．

6. ま と め

本研究では，異種 IoT と IoT プラットフォームの連携を容易化することを目的とし，プラットフォームとデバイスを仲裁するサービス IoT Mediator を提案した．IoT Mediator は制御命令データを変換・規格化することによるアダプタ開発の標準化，アダプタ登録サービスによるアダプタ情報の管理，デバイス登録サービスによるプラットフォームへのデバイスプロビジョニングの簡単化，ルーチングサービスによるデバイス接続情報に基づいた制御命令データ変換・ルーチングを要素として構成される．本研究では，ケーススタディとして，IoT Mediator を用いて複数のデバイスと FIWARE の連携，またそれらのデバイスを用いた防災アプリを開発した．今後の課題としては，一般的なセンサのプラットフォームとの連携容易化を実現することを考えている．また，クラウドサービスをプラットフォームに連携し，サービス同士の連携を効率化させることも今後の課題としたい．

謝辞　本研究の一部は JSPS 科研費 JP19H01138, JP18H03242, JP18H03342, JP19H04154, JP19K02973, JP20K11059, JP20H04014, JP20H05706 および，立石科学技術振興財団の研究助成を受けて行われている．

文　　献

[1] 山田豊通，“豊かなコミュニケーション社会とは 情報化社会への心構え，” 高分子，vol.49，no.7，pp.428–430，2000.
[2] 原辰次，本多敏，“超スマート社会におけるシステム科学技術概論，” 計測と制御，vol.55，no.4，pp.284–287，2016.
[3] 岩野和生，高島洋典，“サイバーフィジカルシステムと iot (モノのインターネット) 実世界と情報を結びつける，” 情報管理，vol.57，no.11，pp.826–834，2015.
[4] 森郁海，鷲尾元太郎，田村孝之他，“Iot プラットフォームを用いた機器制御アプリケーションの実装，” 第 78 回全国大会講演論文集，vol.2016，no.1，pp.3–4，2016.
[5] 石井和彦，山中淳史，“Fiware を活用したスマートシティ向け共通プラットフォームの構築 (高松市事例)(データを活用した持続可能な都市経営特集)–(データ利活用型スマートシティの実証・実装事例)，” NEC 技報= NEC technical journal，vol.71，no.1，pp.29–32，2018.
[6] 中谷将大，佐伯幸郎，中村匡秀，安田清，“バーチャルエージェントを活用した個人オントロジー構築システムの試作，” 電子情報通信学会技術研究報告; 信学技報，vol.118，no.293，pp.7–12，2018.
[7] “Switchbot（スイッチボット），” https://www.switchbot.jp/.（Accessed on 02/06/2022).
[8] “Nature remo（ネイチャーリモ），” https://nature.global/nature-remo/.（Accessed on 02/06/2022).
[9] “Aws iot，” https://aws.amazon.com/jp/iot/.（Accessed on 02/06/2022).
[10] “Iot-ex，” https://www.iot-ex.co.jp/. (Accessed on 02/06/2022).