Orchestrating a brighter world NEC

IoT 集配加工マネージドサービス利用ガイド（開発編）

2021 年 11 月 8 日 第 4 版

日本電気株式会社

CPF-WOT-MNG-SVC-19-003

改版履歴

版数	日付	改版内容
1.0	2019 年 5 月 9 日	新規作成
2.0	2020 年 10 月 21 日	「6.3 ユーザアプリケーション API」に蓄積データを扱う API の利用方法を追記
3.0	2021 年 5 月 25 日	「6.2.1.2 アクチュエーション指示の取得」にクエリパラメータ timeout の説明を追記
4.0	2021 年 11 月 8 日	「6.2.2 MQTT API」に暗号化通信のサポートを追加

目次

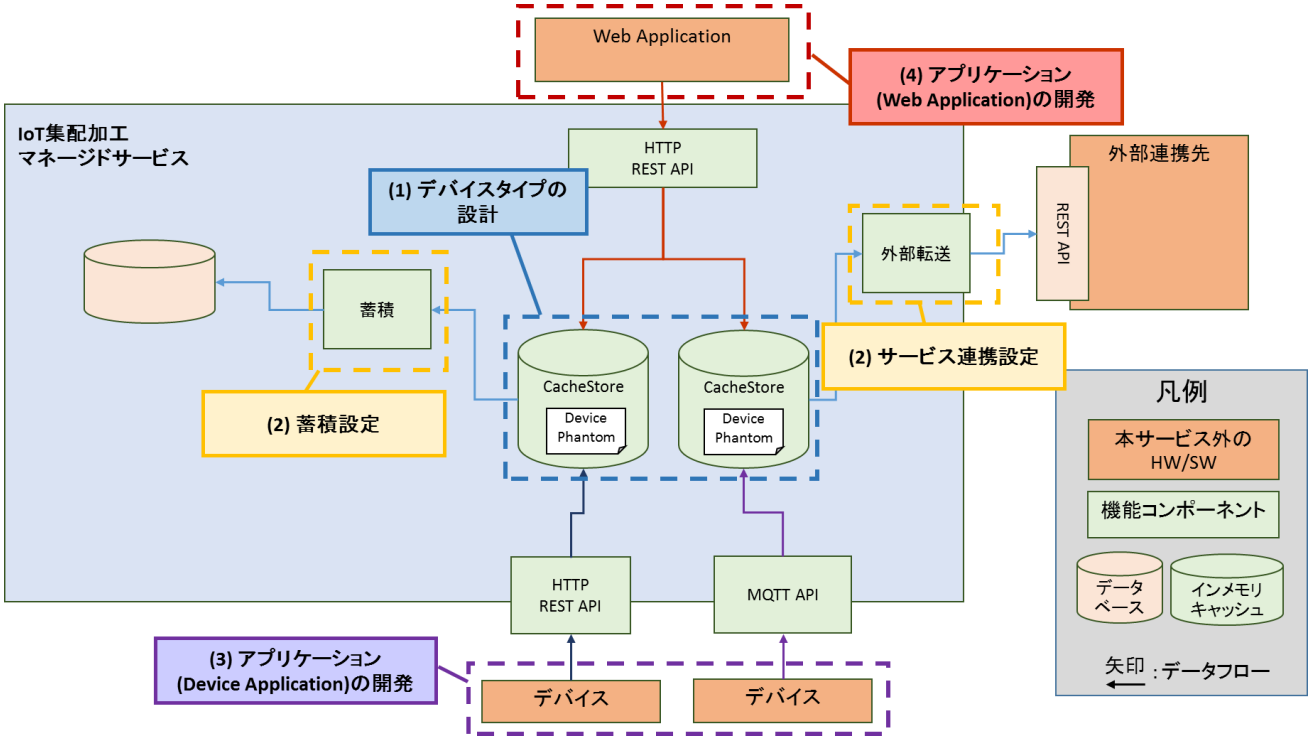
1 はじめに

本書は、IoT 集配加工マネージドサービス(以降、本サービス)を利用するシステム開発者、デバイスアプリケーション開発者およびユーザアプリケーション開発者向けのガイドです。IoT 集配加工マネージドサービスの概要を把握するために事前に「IoT 集配加工マネージドサービス 利用ガイド（概要編)」を一読してください。

1.1 本ドキュメントが対象とする作業の範囲

本書が対象とする作業は、本サービスを用いたシステムを開発するために検討が必要な以下の範囲です。これらの範囲の作業を行うことで本サービスが提供するデバイスデータの蓄積、デバイスデータの転送、デバイスデータの検索機能を活用したシステムが構築できます。

(1) デバイスタイプの設計
(2) 蓄積およびサービス連携設定
(3) デバイスアプリケーションの開発
(4) ユーザアプリケーションの開発



(5) 図 1.1-1 本サービスを利用したシステム開発で行う各作業

それぞれの作業の概要は以下となります。

(1) **デバイスタイプの設計**

本サービスを利用する際には、デバイスタイプの設計が必要です。
デバイス検索機能を持つユーザアプリケーションがクエリによる検索 API を利用する場合、本サービスにデータを蓄積する場合、連携先の外部サービスにデータを転送する場合などの実現したいシステムに合わせ、必要となるデバイスタイプの洗い出しを行い、本サービスに登録します。デバイスタイプの設計を行う際は事前に 3.1 の内容をご確認ください。デバイスタイプの詳細については 0 を参照してください。

(2) **データ蓄積・外部サービス連携の設定**

本サービスが受信したデータは蓄積機能を用いて本サービス内に蓄積できます。また、転送機能を用いることで、本サービスが受信したデータを連携先サービス(ソフトウェア)に転送することもできます。蓄積機能および転送機能が持つ JSON の変換機能を用いることで、蓄積データを活用するソフトウェアや連携先サービスが期待する形に JSON のフォーマットを変更したうえで送信することができます。

(3) **デバイスアプリケーションの開発**

本サービスが公開する HTTP REST API もしくは MQTT API を利用することで、デバイスデータの送信とアクチュエーション指示の受信を行うデバイスアプリケーションが開発できます。

エッジデバイス上で動作するデバイスアプリケーションがセンサデバイスからデータを取得し、本サービスに送信します。本サービスから受信したアクチュエーション指示を受信し、センサデバイスの設定変更を行います。

(4) **ユーザアプリケーションの開発**

本サービスが公開する HTTP REST API を利用し、デバイスの状態確認、デバイスデータの検索、アクチュエーション指示などのデバイス管理操作を行うユーザアプリケーションが開発できます。

2 IoT 集配加工マネージドサービスの代表的なデータフロー

本章では本サービスの代表的な以下のシナリオでのデータの流れについて説明します。

・デバイスからのデバイスデータの受信
・ユーザアプリケーションからデバイスへのアクチュエーション指示の配信
・ユーザアプリケーションからクエリを利用したデバイスデータの検索

2.1 デバイスからのデバイスデータの受信

2.1.1 サービス連携をしない場合のデータフロー

デバイスからデバイスデータを受信し、サービスとの連携を行わずに本サービス内でデバイスの状態を管理する場合のデータフローは以下のとおりです。なお、本データフローは Cache 型デバイスタイプを利用した設定の構成です。

① 本サービスは、デバイスが送信したデータを HTTP REST API/MQTT API のいずれかで受信します。
② 本サービスは受信したデータから Device Phantom を作成し、キャッシュストアに保存します。

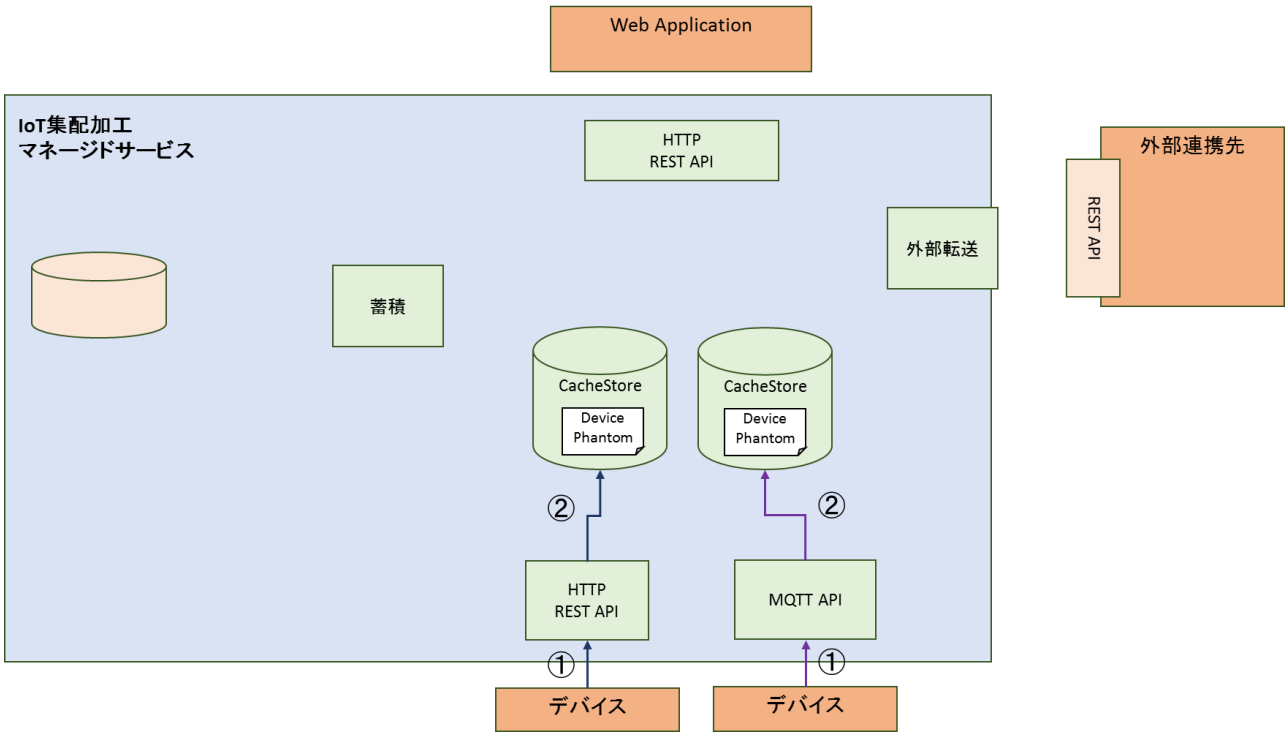


図 2.1-1 デバイスからのデバイスデータ受信のフロー（蓄積・外部転送なし）

2.1.2 サービス連携およびデータ蓄積する場合のデータフロー

デバイスからデバイスデータを受信し、サービス連携を設定した本サービス内でデバイスの状態情報を管理する場合のデータフローは以下のとおりです。

① 本サービスは、デバイスが送信したデータを HTTP REST API/MQTT API のいずれかで受信します。
② 本サービスは受信したデータを Device Phantom の Reported プロパティに設定し、キャッシュストアに格納します。
③ 本サービスは Device Phantom の更新が完了後に、更新した Device Phantom のデータをサービス連携

で設定された外部サービスに転送します。

④ 転送機能では、JSON のフォーマット変更が可能です。JSON フォーマット変更を設定していない場合は、Device Phantom データフォーマット(3.1 節参照)のまま送信されます。

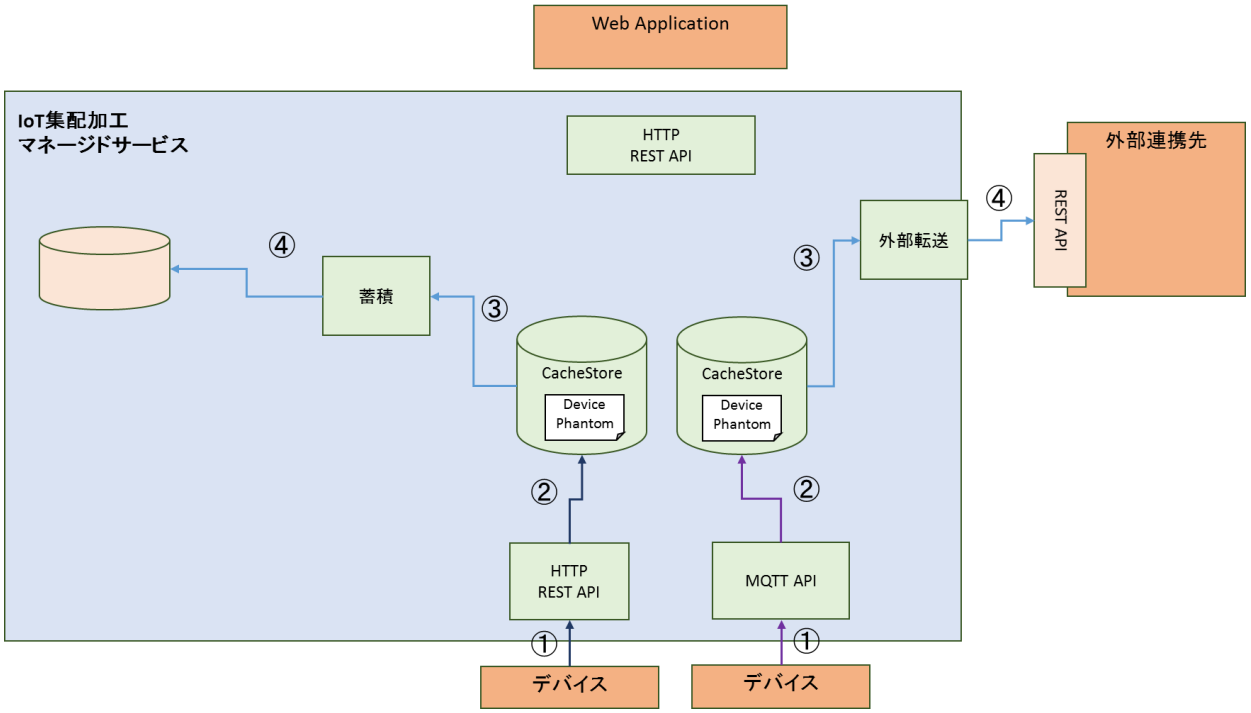


図 2.1-2 デバイスデータ受信のフロー（Cache 型デバイスタイプ/蓄積・外部転送あり）

なお、パススルー型のデバイスタイプを利用することで、デバイスの状態情報を管理しないケースも実現できます。

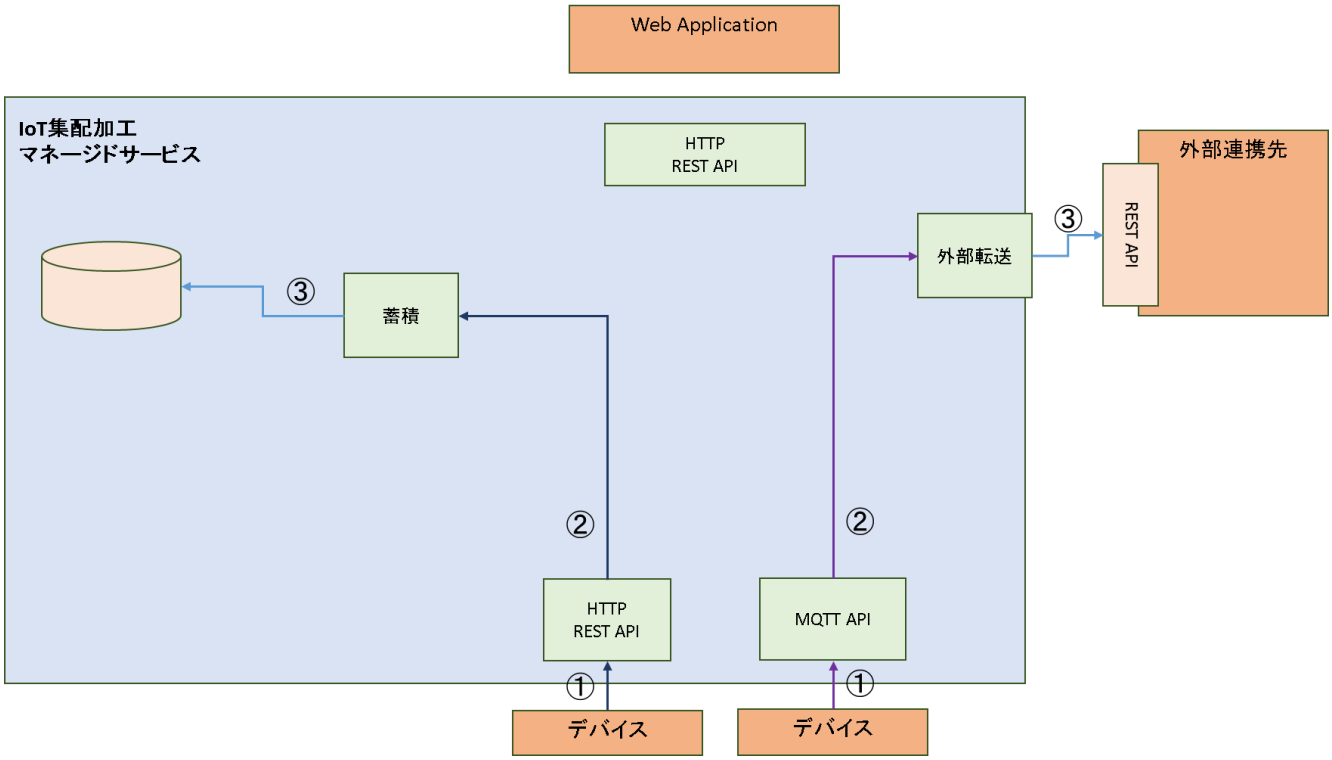


図 2.1-3 デバイスデータ受信のフロー（イベントパススルー型デバイスタイプ/蓄積・外部転送あり）

2.2 ユーザアプリケーションからデバイスへのアクチュエーション指示

本サービスでは、デバイスアプリケーションから本サービスへのデータ送信だけでなく、本サービスからデバイスアプリケーションへのデータ送信(アクチュエーション指示)を行うことができます。

アクチュエーション指示の機能は、Cache 型デバイスタイプを用いることで利用できます。Cache 型デバイスタイプでは、デバイスデータから送信されるデバイスデータの最新の値(reported データ)を保持しています。また、ユーザアプリケーションから送信されるデバイスアプリケーションに期待する値(desired データ)も併せて保持しています。アクチュエーション指示は reported データと desired データが一致しない場合に、本サービスから送信されます。

アクチュエーション指示のデータフローは以下のとおりです。(①→②→③→④は③→④→①→②の順になっても同様です)

① 本サービスはデバイスが送信したデータを REST API/MQTT API のいずれかで受信します。
② 本サービスは受信したデータをキャッシュストア内に格納された Device Phantom の reported プロパティに設定します。
③ ユーザアプリケーションはデバイスに期待する値を設定するため、REST API にデータを送信します。
④ 本サービスは受信したデータをキャッシュストア内に格納された Device Phantom の desired プロパティに設定します。
⑤ 本サービスは reported データ、もしくは desired データの更新を検知すると、reported データと desired データ内に含まれる各プロパティの差分比較を行います。同一プロパティ名で異なる値のデータを検出すると、アクチュエーション指示をデバイスに送信します。
アクチュエーション指示は、差分を検知したプロパティを含んだ JSON です。プロパティの値にはユーザアプリケーションに設定された desired データの値が設定されます。

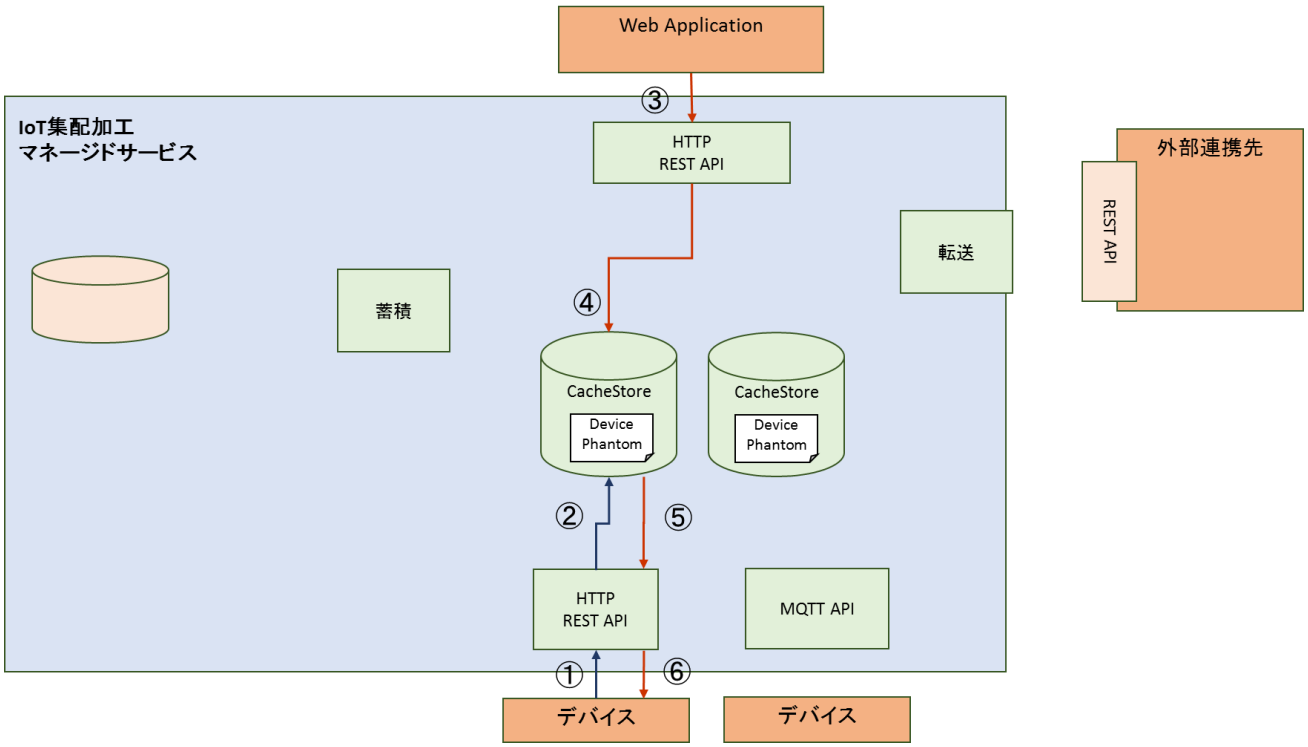


図 2.2-1 アクチュエーション指示の送信フロー

2.3 ユーザアプリケーションによるクエリを利用したデバイス検索

ユーザアプリケーションがクエリを利用しデバイスを検索する場合のデータフローは以下のとおりです。なお、本機能を利用するには Cache 型のデバイスタイプを利用している必要があります。

① ユーザアプリケーションがデバイスの検索のクエリを REST API に送信します。
② 本サービスは指定されたデバイスタイプを対象に受信したクエリを実行し、条件に合致するデバイスの検索を行います。
③ 検索結果を HTTP REST API に戻します。
④ ユーザアプリケーションにデバイスの検索結果が返却されます。

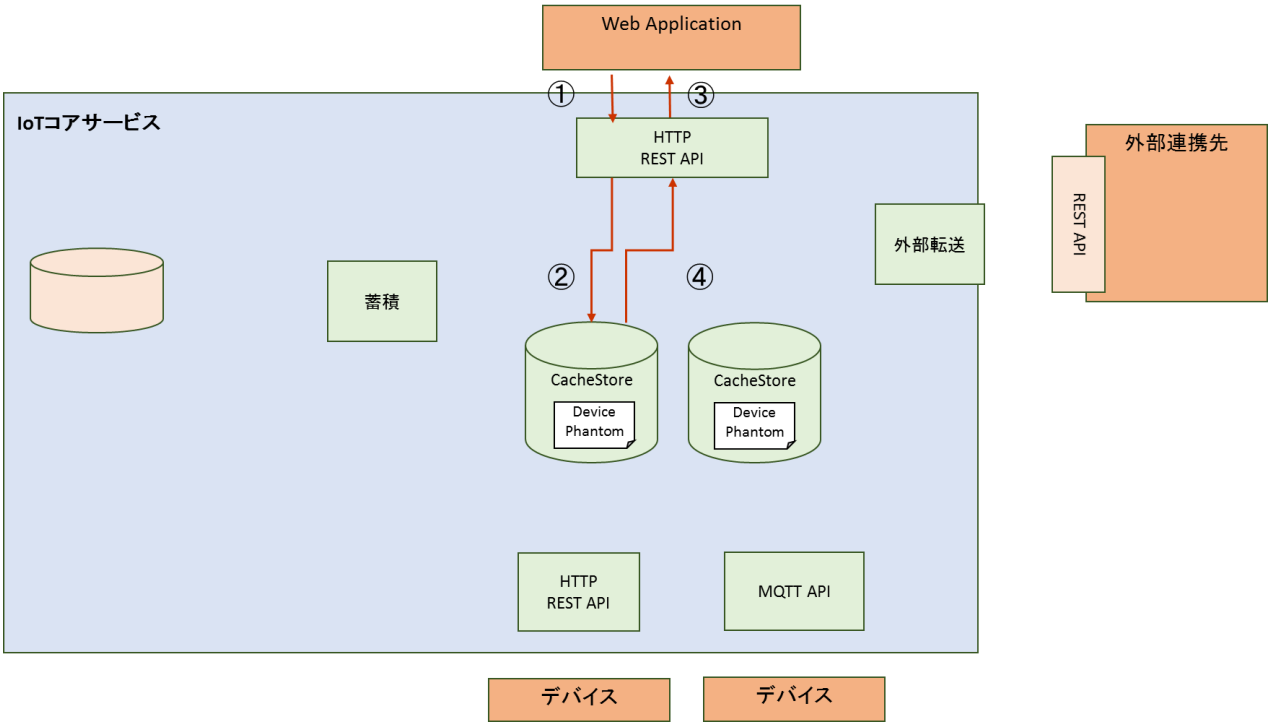


図 2.3-1 ユーザアプリケーションによるクエリを利用したデバイス検索のフロー

3 本サービス集配設定の設計

本章では、本サービスを利用したシステムを構築する前に行うべき設計作業、および設計の事前知識として必要となる仕様・用語について説明します。

3.1 Device Phantom の仕様理解

Device Phantom とは、本サービスがデバイスの状態情報を管理するためのデータ構造です。本サービスが規定するデータフォーマットに従った JSON ドキュメントとしてキャッシュストアに保持されます。

以下の温度センサ(temperature)と湿度センサ(humidity)を例に説明します。JSON ドキュメント内の赤字のプロパティ名は本サービスの予約語です。

```
{
    "deviceId" : "ieMSI_1FH1ne",
    "attributes" : {
       "reported" : {
          "temperature" : 17.0,
          "humidity" : 48.6
       },
       "desired" : {
          "temperature" : 25.0
       },
       "delta" : {
          "temperature" : 25.0
       }
    },
    "metadata" : {
       "reported" : {
          "temperature" : {
              "timestamp" : 1501475759191
          },
          "humidity": {
           "timestamp" : 1501475759191
          }
       }
    },
    "lastUpdate" : 1501475759191,
    "version": 15
}
```

JSON ドキュメントの各プロパティについて説明します。

プロパティ名	説明
deviceId	各デバイスに一意に割り振ったデバイスの識別子を設定してください。 使用できる文字列は最大 100 文字の英数字と記号(.-)です。未指定の場合、デバイス送信時の HTTP、MQTT のトピックに指定したクライアント ID の値が設定されます
attributes	-
reported	デバイスが送信するデバイスデータ(=センサの実測値)を設定してください。本フィールドは任意のスキーマの JSON オブジェクトが設定できます。オブジェクトはネスト構造が可能で、ネストの深さは 5 階層まで利用いただけます
desired	運用者や管理アプリがデバイスに送信する期待値を設定してください。本プロパティに設定された JSON で、attributes.reported に設定された JSON のスキーマ(プロパティ名・構造)が一致しているものを対象に後述の delta が計算されます。
delta	reported データと desired データに差分がある場合に、desired データの値を設定してください。本サービスは attributes.reported や attributes.desired のデータ受信時に自動で delta の計算をします。delta が作成・更新されると、デバイスに delta が通知されます。なお、デバイスへの通知の負荷軽減のため、再計算された delta が前回送信時の delta 値と同じ場合、delta は送信されません。
metadata	-
reported	reported データのメタ情報を設定してください。デバイスやアプリケーションは本フィールドに管理用の任意の情報をメタ情報に設定することができます。 メタ情報は、本サービスが設定する属性の最終更新時刻を格納する timestamp フィールドを含みます。timestamp は本サービスにより自動的に更新され、デバイスやアプリケーションから更新できません。時刻の形式は unix-time で精度はミリ秒です。本サービスが動作するホストマシンで取得した時刻が使用されます。
desired	desired データのメタ情報を設定してください。デバイスやアプリケーションは本フィールドに管理用の任意の情報を設定することができます。
delta	reported データと desired データの差分(delta データ)のメタ情報を設定してください。デバイス、アプリケーションは本フィールドに管理用の任意の情報を設定することができます。
lastUpdate	Device Phantom の最終更新時刻が格納されるプロパティです。この属性は本サービスにより自動的に更新され、デバイスやアプリケーションからは更新できません。時刻の形式は unix-time で精度はミリ秒です。本サービスが動作するホストマシンから取得した時刻が使用されます。
version	Device Phantom のバージョンを示す long 型のプロパティです。送信データに version が指定されていない場合、Device Phantom の初回作成時に 0 が設定され、以降は Device Phantom が更新される度に version フィールドの値がインクリメントされます。デバイスやアプリケーションが古いバージョン番号を指定して Device Phantom を更新しようとした場合、その操作は失敗します。本サービスはメッセージの順序保証をしないため、古いデータでの上書きを防止したい場合は本 version フィールドを利用してください。 本フィールドは任意属性のため、Device Phantom のバージョンによらず、クライアントから Device Phantom を更新したい場合は、本フィールドを省略して送信します。version の値にはエポックミリ秒を推奨します。

3.2 デバイスタイプの設計

本節ではデバイスタイプの設計について説明します。デバイスタイプの説明は概要ガイドを参照ください。デバイスタイプの設計は、システムの全体の構成、システムの目的に合わせて行います。デバイスタイプは以下の 2 つの観点で設計を行います。

- Cache 型もしくはパススルー型の選択
- デバイスから受信するデバイスデータの型の登録

3.2.1 Cache 型・パススルー型の選択

「IoT 集配加工マネージドサービス利用ガイド(概要編)」の「3.1 デバイスタイプ」の説明を参照し、Cache 型・パススルー型のどちらを利用するか決定してください。

3.2.2 デバイス検索のための設計（Cache 型デバイスタイプのみ）

Cache 型デバイスタイプを選択した場合、デバイスの検索機能を利用できます。デバイス検索機能は API 実行時に指定した Cache 型デバイスタイプのキャッシュストアを対象に実行されます。デバイスの種類ごとにデバイスデータの送信先のデバイスタイプを分けることで、事前に検索範囲を限定し、運用対象を絞りこむことができます。

検索機能を利用する場合はデバイスタイプの登録の際に、検索時にクエリに指定するプロパティの型の情報の設定が必要になります。検索機能を利用しない場合は型情報の設定は不要です。

本サービスのクエリ検索は SQL ライクな構文を用いて実行します。SQL 文の table 指定に該当する箇所にはデバイスタイプを指定し、クエリはデバイスタイプを対象に検索を実行されます。

デバイスから送信される JSON 形式のデータのプロパティには、以下の 4 つの型が指定できます。

・整数値（LONG）
・小数値（DOUBLE）
・文字列（STRING）
・真偽値（BOOLEAN）

デバイスタイプの登録方法については導入ガイドをご参照ください。

3.2.3 送信データの JSON のスキーマの設計

本サービスのサービス連携機能は送信処理前に JSON スキーマの変換を行うことができます。連携先サービスに送信後のデータの利用方法を検討し、送信データのスキーマを設計します。

送信データは、サービス運用画面から設定することで、期待したフォーマットに変換することができます。

4 サービス連携設定

本章では、サービス連携のための転送機能の設定について説明します。

サービス連携の機能は、本サービスが受信したデバイスおよびユーザアプリケーションからのデータを外部サービスへデータを転送します。

サービス連携機能は、全てのデバイスタイプの更新を検知して動作するケースと、デバイスタイプ毎に登録し特定のデバイスタイプの更新を検知して動作するケースが選択できます。１つのデバイスタイプに複数のサービス連携設定を行うことができます。

本サービスのサービス連携設定で利用できる機能は以下があります。

・データ保管機能

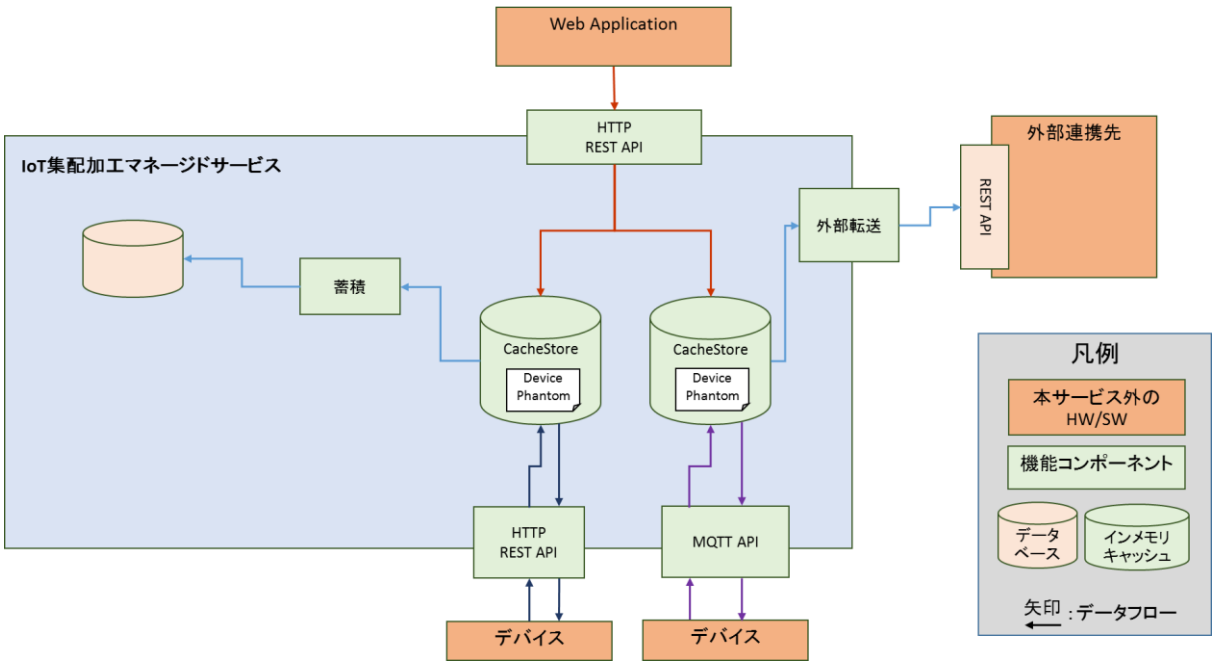
・外部サービス連携機能

5 アプリケーションの開発

5.1 IoT 集配加工マネージドサービスの API 概要

本サービスでは、デバイスアプリケーション向けとユーザアプリケーション向けそれぞれに API を公開しています。デバイスアプリケーション向けには HTTP REST API と MQTT API を提供し、ユーザアプリケーション向けには HTTP REST API を提供しています。

これらの API を呼び出すことで、本サービスが提供するデバイスデータの蓄積機能や外部サービスへの転送機能、デバイスデータの検索機能を利用することができます。



5.2 デバイスアプリケーションの開発

本サービスが公開している API を用いて、デバイスアプリケーションを開発します。API には HTTP または MQTT を利用することができます。

REST API については 6.1 のデバイス向け API リファレンスをご参照ください。

5.3 ユーザアプリケーションの開発

本サービスが公開している REST API を用いて、ユーザアプリケーションを開発します。
例えば、以下のような機能をもったユーザアプリケーションを開発することができます。

・本サービスにデータを登録したデバイスの一覧を取得する
・特定のデバイスの状態(センサデータ)を監視、表示する
・条件に沿ったデバイス一覧を取得する
・上記の取得情報を基にデバイスの状態の期待値を設定し、デバイスの状態を変更する

REST API については 0 の REST API リファレンスをご参照ください。
なお、REST API を操作するためには、API サーバの認証トークンが必要になります。REST API をご利用の

際は事前に本サービスが提供するサービス運用画面からユーザアカウントを登録してください。アカウントの追加方法については、導入ガイドを参照してください

6 API リファレンス

6.1 デバイスアプリケーション・ユーザアプリケーション向け共通 API

6.1.1 HTTP REST API

6.1.1.1 ログイン

デバイスアプリケーション・デバイス向け API・ユーザアプリケーション向けの HTTP REST API の実行に必要なセッショントークンを取得するための API です。

項目	説明
URI	https://{ホスト名}.nec-iot-da2.com/api/login
HTTP メソッド	POST
HTTP ヘッダ	• X-Application-Id: アプリケーション ID (必須) • Content-Type: application/json (必須)
リクエストボディ	以下のプロパティを含む JSON データを送信します。 username: API を利用するユーザの名前を指定します。(必須) password: API を利用するユーザのパスワードを指定します。(必須)
レスポンスボディ	以下の情報を含む JSON が返却されます。 sessionToken: セッショントークン
備考	

6.1.1.2 デバイスの状態取得

指定したデバイス ID の Device Phantom を取得します。

項目	説明
URI	https://{ホスト名}.nec-iot-da2.com/api/devices/{deviceType}/{deviceId}
HTTP メソッド	GET
HTTP ヘッダ	• X-Application-Id: アプリケーション ID (必須) • X-Session-Token: セッショントークン (必須)
リクエストボディ	無し
レスポンスボディ	Device Phantom の JSON データが返却されます。
備考	

6.1.1.3 デバイスの状態削除

指定したデバイス ID の Device Phantom を削除します。

項目	説明
URI	https://{ホスト名}.nec-iot-da2.com/api/devices/{deviceType}/{deviceId}
HTTP メソッド	DELETE
HTTP ヘッダ	• X-Application-Id: アプリケーション ID (必須) • X-Session-Token: セッショントークン (必須)
リクエストボディ	無し
レスポンスボディ	無し
備考	

6.2 デバイス向け API

デバイス向け API はデバイスがデバイスデータを送信するための API と本サービスが保持する最新データをデバイスが取得するための API の 2 種類があります。各 API は HTTP と MQTT のプロトコルに対応しています。

なお、{}で括った箇所は実行したい API の対象に合わせて設定してください。{deviceId}と書かれた箇所は、サービス運用画面で登録したデバイスアカウントの ID を指定します。

6.2.1 HTTP REST API

6.2.1.1 デバイスデータの送信

デバイスが本サービスにデバイスデータを送信するための API です。

項目	説明
URI	https://{ホスト名}.nec-iot-da2.com/api/devices/{deviceType}/{deviceId}/reported
HTTP メソッド	PUT
HTTP ヘッダ	• X-Application-Id: アプリケーション ID (必須) • X-Session-Token: セッショントークン (必須) • Content-Type: application/json (必須)
リクエストボディ	登録したいデバイスデータを含む JSON データを送信します。 Cache 型デバイスタイプの場合は Device Phantom のフォーマット(「3.1 Device Phantom の仕様理解」を参照)に従い、以下のデータを送信する必要があります。 • deviceId: デバイス ID を指定します。(任意) • attributes.reported: デバイスデータを設定します。(任意) • metadata.reported: デバイスデータのメタデータを設定します。(任意)
レスポンスボディ	無し
備考	

6.2.1.2 アクチュエーション指示の取得

デバイスがアクチュエーション指示を取得する API です。アクチュエーション指示がない場合は、発生するまで一定時間待機します。

項目	説明
URI	https://{ホスト名}.nec-iot-da2.com/api/devices/{deviceType}/{deviceId}/event
HTTP メソッド	GET
HTTP ヘッダ	• X-Application-Id: アプリケーション ID (必須) • X-Session-Token: セッショントークン (必須)
クエリパラメータ	• timeout: タイムアウト(秒)の値を指定します。指定可能な値は 0 以上 25 以下です。
リクエストボディ	無し
レスポンスボディ	JSON 形式で目標状態値、最後に報告された状態値、状態の設定値とデバイスの最新状態の差分、メタデータなどを受信します。 詳細は Device Phantom の節を参照して下さい。
備考	本 API は定期的に呼び出し(ポーリング)して利用することを推奨します。アクチュエーション指示が発生した場合は、HTTP ステータスコード：200 OK と共にアクチュエーション指示が得られます。アクチュエーション指示が発生しない場合、REST API は一定時間(30 秒間)ブロックした後に、HTTP ステータスコード：204 No Content を返します。

6.2.2 MQTT API

MQTT 用の API は非同期で処理されるため、API の呼び出し(処理実行)と結果取得(結果確認)はそれぞれ異なるトピックを用いて実施する必要があります。API の呼び出しは対象となるトピックに publish し、結果の取得は対象となるトピックを subscribe します。結果の取得や確認を行う場合は、API を呼び出す前に subscribe を実施して、結果の待ち受け状態にしてから、API のリクエスト送信してください。

6.2.2.1 暗号化通信のサポート

本サービスでは通常の MQTT プロトコルによるデバイスとの通信に加え、TLSv1.2 を利用した暗号化通信をサポートします。MQTT の暗号化通信を利用する場合はデバイスの MQTT クライアントの暗号化通信を有効化し本サービスの 8883 を接続先として指定してください。

6.2.2.2 デバイスデータの送信

(1) デバイスデータの送信（API 呼び出し）

デバイスが本サービスにデバイスデータを送信するための API です。

項目	説明
接続先ドメイン	{ホスト名}-mqtt.nec-iot-da2.com
接続先ポート	MQTT: 1883、MQTTS: 8883
トピック	eventhub/v1/{ホスト名}/{deviceType}/{deviceId}/phantom/report
送信データ	登録したいデバイスデータを含む JSON データを送信します。 Cache 型デバイスタイプの場合は Device Phantom のフォーマット(「3.1 Device Phantom の仕様理解」を参照)に従い、以下のデータを送信する必要があります。 • deviceId: デバイス ID を指定します。(任意) • attributes.reported: デバイスデータを設定します。(任意) • metadata.reported: デバイスデータのメタデータを設定します。(任意)

(2) デバイスデータ送信の結果取得

項目	説明
接続先ドメイン	{ホスト名}-mqtt.nec-iot-da2.com
接続先ポート	MQTT: 1883、MQTTS: 8883
トピック	成功/失敗を確認するため、以下のトピック指定でサブスクライブします。 eventhub/v1/{ホスト名}/{deviceType}/{deviceId}/phantom/report/+ ・IoT Event Hub が受理した場合、以下のトピックにデータが返却されます。 eventhub/v1/{ホスト名}/{deviceType}/{deviceId}/phantom/report/accepted ・IoT Event Hub が拒否した場合、以下のトピックにデータが返却されます。 eventhub/v1/{ホスト名}/{deviceType}/{deviceId}/phantom/report/rejected
受信データ	accepted の場合、Device Phantom の JSON を受信します。 rejected の場合は以下のプロパティを持つ JSON を受信します。 - code: サーバが出力するエラーコード - message: エラーメッセージ

6.2.2.3 Device Phantom の取得

(1) デバイスデータの取得指示（API 呼び出し）

デバイスが本サービスから Device Phantom の値を取得する API です。

項目	説明
接続先ドメイン	{ホスト名}-mqtt.nec-iot-da2.com
接続先ポート	MQTT: 1883、MQTTS: 8883
トピック	eventhub/v1/{ホスト名}/{deviceType}/{deviceId}/phantom/get
送信データ	空データ

(2) デバイスデータの取得指示の結果取得

項目	説明
接続先ドメイン	{ホスト名}-mqtt.nec-iot-da2.com
接続先ポート	MQTT: 1883、MQTTS: 8883
トピック	成功/失敗を確認するため、以下のトピック指定でサブスクライブします。 eventhub/v1/{ホスト名}/{deviceType}/{deviceId}/phantom/get/+ ・IoT Event Hub が受理した場合、以下のトピックにデータが返却されます。 eventhub/v1/{ホスト名}/{deviceType}/{deviceId}/phantom/get/accepted ・IoT Event Hub が拒否した場合、以下のトピックにデータが返却されます。 eventhub/v1/{ホスト名}/{deviceType}/{deviceId}/phantom/get/rejected
受信データ	accepted の場合、Device Phantom の JSON を受信します rejected の場合は以下のプロパティを持つ JSON を受信します。 - code: サーバが出力するエラーコード - message: エラーメッセージ

6.2.2.4 Actuation の取得

デバイスが本サービスからアクチュエーション指示を取得する API です。subscribe でアクチュエーション指示のイベント発生を待つため、API 呼び出し(publish)は不要です。

項目	説明
接続先ドメイン	{ホスト名}-mqtt.nec-iot-da2.com
接続先ポート	MQTT: 1883、MQTTS: 8883
トピック	eventhub/v1/{ホスト名}/{deviceType}/{deviceId}/phantom/actuate
受信データ	Device Phantom の JSON を受信します

6.3 ユーザアプリケーション向け API

ユーザアプリケーション向けに提供する API です。

6.3.1 HTTP REST API

6.3.1.1 期待するデバイス状態の設定

ユーザアプリケーションから本サービスへ期待するデバイスの状態の設定を行う API です。本サービスはデータ受信後に Device Phantom の登録・更新を行います。

項目	説明
URI	https://{ホスト名}.nec-iot-da2.com/api/devices/{deviceType}/{deviceId}/desired
HTTP メソッド	PUT
HTTP ヘッダ	• X-Application-Id: アプリケーション ID (必須) • X-Session-Token: セッショントークン (必須) • Content-Type: application/json (必須)
リクエストボディ	期待するデバイスの状態の設定を含む JSON データを送信します。 Cache 型デバイスタイプの場合は Device Phantom のフォーマット(「3.1 Device Phantom の仕様理解」を参照)に従い、以下のデータを送信する必要があります。 deviceId : デバイス ID を指定します。(任意) attributes.desired: 期待するデバイスの状態の設定値を設定します。(任意) metadata.desired: 期待するデバイスの状態の設定情報のメタデータを設定します。(任意)
レスポンスボディ	無し
備考	

6.3.1.2 デバイス ID の一覧取得

指定したデバイスタイプに含まれるデバイス ID の一覧を返します。

項目	説明
URI	https://{ホスト名}.nec-iot-da2.com/api/deviceids/{deviceType}
HTTP メソッド	GET
HTTP ヘッダ	• X-Application-Id: アプリケーション ID (必須) • X-Session-Token: セッショントークン (必須)
リクエストボディ	無し
レスポンスボディ	JSON 形式. { "deviceIds": ["デバイス ID0", "デバイス ID1", …] }
備考	

6.3.1.3 デバイス検索

条件によるデバイスの検索を行います。

項目	説明
URI	https://{ホスト名}.nec-iot-da2.com/api/devices/query/{deviceType}
HTTP メソッド	POST
HTTP ヘッダ	• X-Application-Id: アプリケーション ID (必須) • X-Session-Token: セッショントークン (必須)
リクエストボディ	文字列 (SQL クエリ) ※を設定します。クエリの仕様については後述の説明を参照ください。
レスポンスボディ	JSON 形式。クエリ結果の文字列を { "result": ["クエリ結果文字列 1", "クエリ結果文字列 2", …] } という JSON で返します。 個々のクエリ結果文字列の仕様は IoT Event Hub の仕様を参照して下さい。
備考	

(1) デバイス検索のクエリについて

本サービスでは以下の例のような SQL ライクなクエリでの検索が実行可能です。

```
{
  “sql” : “select * from <デバイスタイプ名> where <デバイスタイプに登録した ailias 名> ≥ 条件”
}
```

下線太字の文字列は固定値です。<>の値は本サービスに設定した内容に合わせてください。

クエリでは複数のデバイスタイプを利用した副問い合わせも可能です。

```
{
  “sql” : “select * from <デバイスタイプ名> XX <デバイスタイプ名> XX where XX.<デバイスタイプ XX に登録した ailias 名> in (select YY.<デバイスタイプ YY に登録した alias 名> form <デバイスタイプ名> YY where YY.value ≤ 条件式)”
}
```

6.3.1.4 デバイスの状態一括削除

指定したデバイスタイプの Device Phantom を一括削除します。

項目	説明
URI	https://{ホスト名}.nec-iot-da2.com/api/devices/{deviceType}
HTTP メソッド	DELETE
HTTP ヘッダ	• X-Application-Id: アプリケーション ID (必須) • X-Session-Token: セッショントークン (必須)
リクエストボディ	無し
レスポンスボディ	無し
備考	

6.3.1.5 蓄積データのクエリによる取得(同期)

蓄積したデータを検索し、返します。

項目	説明
URI	https://{ホスト名}.nec-iot-da2.com/api/1/storedata/query/{データ保存先名}
HTTP メソッド	GET
HTTP ヘッダ	• X-Application-Id: アプリケーション ID (必須) • X-User-Name: ユーザ ID (必須) • X-User-Password: ユーザパスワード (必須)
クエリパラメータ	• filter: 検索条件を JSON 形式で指定します。JSON の内容は URL エンコードを行う必要があります。 • order: ソートを行うキー名を指定します。昇順で検索する場合はキー名を指定します。降順で検索する場合はキー名の前に"-"を付与して指定します。複数のキーを指定するときは、キー名を優先順に","で区切ります。 • limit: 返却する検索数の上限を指定します。デフォルト値は 100 件です。limit パラメータを指定しなかった場合、上限は 100 件となります。limit パラメータに-1 を指定した場合は上限無しとなります。 • skip: 検索結果の先頭からのスキップ数を指定します。デフォルト値は 0 件です。
リクエストボディ	無し
レスポンスボディ	クエリ結果の文字列を {"results": ["クエリ結果文字列 1", "クエリ結果文字列 2", …], "currentTime": "クエリ実行時刻"} という形式の JSON で返します。
備考	

6.3.1.6 蓄積データのクエリによる更新(同期)

蓄積したデータを検索し、指定した JSON で更新します。

項目	説明
URI	https://{ホスト名}.nec-iot-da2.com/api/1/storedata/query/{データ保存先名}
HTTP メソッド	PUT
HTTP ヘッダ	• X-Application-Id: アプリケーション ID (必須) • X-User-Name: ユーザ ID (必須) • X-User-Password: ユーザパスワード (必須) • Content-Type: application/json (必須) • X-Full-Update: JSON の置換動作/更新動作を true/false で指定します。(任意) true を指定すると置換動作となります。デフォルト値は false で更新動作です。
クエリパラメータ	• filter: 検索条件を JSON 形式で指定します。JSON の内容は URL エンコードを行う必要があります。 • order: ソートを行うキー名を指定します。昇順で検索する場合はキー名を指定します。降順で検索する場合はキー名の前に"-"を付与して指定します。複数のキーを指定するときは、キー名を優先順に","で区切ります。 • limit: 更新対象とする検索数の上限を指定します。デフォルト値は 100 件です。limit パラメータを指定しなかった場合、上限は 100 件となります。limit パラメータに-1 を指定した場合は上限無しとなります。 • skip: 検索結果の先頭からのスキップ数を指定します。デフォルト値は 0 件です。
リクエストボディ	更新する JSON データ (必須)
レスポンスボディ	無し
備考	

6.3.1.7 蓄積データのクエリによる削除(同期)

蓄積したデータを検索し、削除します。

項目	説明
URI	https://{ホスト名}.nec-iot-da2.com/api/1/storedata/query/{データ保存先名}
HTTP メソッド	DELETE
HTTP ヘッダ	• X-Application-Id: アプリケーション ID (必須) • X-User-Name: ユーザ ID (必須) • X-User-Password: ユーザパスワード (必須) • X-Query-Safeguard: クエリパラメータを指定しない場合の動作を true/false で指定します。(任意) false を指定すると、クエリパラメータを指定しない場合の動作が全件削除となります。 デフォルト値は true です。
クエリパラメータ	• filter: 検索条件を JSON 形式で指定します。JSON の内容は URL エンコードを行う必要があります。
リクエストボディ	無し
レスポンスボディ	無し
備考	

6.3.1.8 蓄積データの取得(非同期)

蓄積したデータを検索し、結果を非同期で返します。本 API は以下の流れで使用します。

1. リクエストを送信します。
2. レスポンスボディの statusUrl プロパティの値の URL にアクセスして status の値を確認し、値が"DONE"になるまで待ちます。
   リクエスト受付時の status の値は"RUNNING"です。処理が完了すると status の値が"DONE"になります。status の値が"DONE"になるまでの時間は蓄積データの量やクエリ内容によって変わります。
3. status の値が"DONE"になっていることを確認後、レスポンスボディの resultUrl プロパティの値の URL にアクセスして検索結果を取得します。

※statusUrl および resultUrl の URL にアクセス可能な期間はリクエスト受付後から約 6 時間です。

項目	説明
URI	https://{ホスト名}.nec-iot-da2.com/api/1/storedata/asyncquery/{データ保存先名}
HTTP メソッド	GET
HTTP ヘッダ	• X-Application-Id: アプリケーション ID (必須) • X-User-Name: ユーザ ID (必須) • X-User-Password: ユーザパスワード (必須)
クエリパラメータ	• filter: 検索条件を JSON 形式で指定します。JSON の内容は URL エンコードを行う必要があります。 • order: ソートを行うキー名を指定します。昇順で検索する場合はキー名を指定します。降順で検索する場合はキー名の前に"-"を付与して指定します。複数のキーを指定するときは、キー名を優先順に","で区切ります。 • limit: 返却する検索数の上限を指定します。デフォルト値は 100 件です。limit パラメータを指定しなかった場合、上限は 100 件となります。limit パラメータに-1 を指定した場合は上限無しとなります。 • skip: 検索結果の先頭からのスキップ数を指定します。デフォルト値は 0 件です。
リクエストボディ	無し
レスポンスボディ	以下のプロパティを持った JSON 形式。 • id: リクエストごとに発行される ID • query: リクエストされたクエリ文字列 • statusUrl: 実行状況のステータスを表すファイルの URL • resultUrl: 検索結果ファイルの URL
statusUrl のファイル内容	以下のプロパティを持った JSON 形式。 • id: リクエストごとに発行される ID • status: “RUNNING”/”DONE”/”CANCEL”のいずれかの値 • description: 実行中断の理由 (status が“CANCEL”となった場合のみ)
resultUrl のファイル内容	{"results": ["クエリ結果文字列 1", "クエリ結果文字列 2", …], "currentTime": "クエリ実行時刻"} という形式の JSON。
備考	

6.3.1.9 蓄積データの更新(非同期)

蓄積したデータを検索し、結果を非同期で更新します。本 API は以下の流れで使用します。

1. リクエストを送信します。
2. レスポンスボディの statusUrl プロパティの値の URL にアクセスして status の値を確認し、値が"DONE"になるまで待ちます。
   リクエスト受付時の status の値は"RUNNING"です。処理が完了すると status の値が"DONE"になります。status の値が"DONE"になるまでの時間は蓄積データの量やクエリ内容によって変わります。
3. status の値が"DONE"になっていれば、蓄積データの更新処理は完了しています。

※statusUrl の URL にアクセス可能な期間はリクエスト受付後から約 6 時間です。

項目	説明
URI	https://{ホスト名}.nec-iot-da2.com/api/1/storedata/asyncquery/{データ保存先名}
HTTP メソッド	PUT
HTTP ヘッダ	• X-Application-Id: アプリケーション ID (必須) • X-User-Name: ユーザ ID (必須) • X-User-Password: ユーザパスワード (必須) • Content-Type: application/json (必須) • X-Full-Update: JSON の置換動作/更新動作を true/false で指定します。(任意) true を指定すると置換動作となります。デフォルト値は false で更新動作です。
クエリパラメータ	• filter: 検索条件を JSON 形式で指定します。JSON の内容は URL エンコードを行う必要があります。 • order: ソートを行うキー名を指定します。昇順で検索する場合はキー名を指定します。降順で検索する場合はキー名の前に"-"を付与して指定します。複数のキーを指定するときは、キー名を優先順に","で区切ります。 • limit: 更新対象とする検索数の上限を指定します。デフォルト値は 100 件です。limit パラメータを指定しなかった場合、上限は 100 件となります。limit パラメータに-1 を指定した場合は上限無しとなります。 • skip: 検索結果の先頭からのスキップ数を指定します。デフォルト値は 0 件です。
リクエストボディ	更新する JSON データ (必須)
レスポンスボディ	以下のプロパティを持った JSON 形式。 • id: リクエストごとに発行される ID • query: リクエストされたクエリ文字列 • statusUrl: 実行状況のステータスを表すファイルの URL
statusUrl のファイル内容	以下のプロパティを持った JSON 形式。 • id: リクエストごとに発行される ID • status: “RUNNING”/”DONE”/”CANCEL”のいずれかの値 • description: 実行中断の理由 (status が“CANCEL”となった場合のみ)
備考	

6.3.1.10 蓄積データの削除(非同期)

蓄積したデータを検索し、結果を非同期で削除します。本 API は以下の流れで使用します。

1. リクエストを送信します。
2. レスポンスボディの statusUrl プロパティの値の URL にアクセスして status の値を確認し、値が"DONE"になるまで待ちます。
   リクエスト受付時の status の値は"RUNNING"です。処理が完了すると status の値が"DONE"になります。status の値が"DONE"になるまでの時間は蓄積データの量やクエリ内容によって変わります。
3. status の値が"DONE"になっていれば、蓄積データの削除処理は完了しています。

※statusUrl の URL にアクセス可能な期間はリクエスト受付後から約 6 時間です。

項目	説明
URI	https://{ホスト名}.nec-iot-da2.com/api/1/storedata/asyncquery/{データ保存先名}
HTTP メソッド	DELETE
HTTP ヘッダ	• X-Application-Id: アプリケーション ID (必須) • X-User-Name: ユーザ ID (必須) • X-User-Password: ユーザパスワード (必須) • X-Query-Safeguard: クエリパラメータを指定しない場合の動作を true/false で指定します。(任意) false を指定すると、クエリパラメータを指定しない場合の動作が全件削除となります。 デフォルト値は true です。
クエリパラメータ	• filter: 検索条件を JSON 形式で指定します。JSON の内容は URL エンコードを行う必要があります。
リクエストボディ	無し
レスポンスボディ	以下のプロパティを持った JSON 形式。 • id: リクエストごとに発行される ID • query: リクエストされたクエリ文字列 • statusUrl: 実行状況のステータスを表すファイルの URL
statusUrl のファイル内容	以下のプロパティを持った JSON 形式。 • id: リクエストごとに発行される ID • status: “RUNNING”/”DONE”/”CANCEL”のいずれかの値 • description: 実行中断の理由 (status が“CANCEL”となった場合のみ)
備考	

6.3.1.11 蓄積データの転送

蓄積したデータを検索し、指定した Amazon S3 のバケットに検索結果を転送する API です。本 API は以下の流れで使用します。

1. リクエストを送信します。
2. レスポンスボディの statusUrl プロパティの値の URL にアクセスして status の値を確認し、値が”DONE”になるまで待ちます。
   リクエスト受付時の status の値は”RUNNING”です。処理が完了すると status の値が”DONE”になります。status の値が”DONE”になるまでの時間は蓄積データの量やクエリ内容によって変わります。
3. status の値が”DONE”になっていれば、指定した S3 バケットに検索結果のファイルがアップロードされています。オブジェクト URL はレスポンスボディの resultUrl プロパティの値です。

※statusUrl の URL にアクセス可能な期間はリクエスト受付後から約 6 時間です。

項目	説明
URI	https://{ホスト名}.nec-iot-da2.com/api/1/storedata/transporterquery/{データ保存先名}
HTTP メソッド	GET
HTTP ヘッダ	• X-Application-Id: アプリケーション ID (必須) • X-User-Name: ユーザ ID (必須) • X-User-Password: ユーザパスワード (必須)
クエリパラメータ	• bucket: 検索結果ファイル転送先の Amazon S3 のバケット名を指定します。 (必須) • folder: 検索結果ファイルを保存するバケット内のフォルダ名を指定します。検索結果ファイルの S3 バケット上のキーは”{folder パラメータに指定したフォルダ名}/{リクエスト ID}.json”となります。 folder パラメータを指定しない場合はバケット直下に検索結果ファイルが保存されます。検索結果ファイルの S3 バケット上のキーは”{リクエスト ID}.json”となります。 ※{リクエスト ID}はレスポンスボディの id プロパティの値です。 • filter: 検索条件を JSON 形式で指定します。JSON の内容は URL エンコードを行う必要があります。 • order: ソートを行うキー名を指定します。昇順で検索する場合はキー名を指定します。降順で検索する場合はキー名の前に"-"を付与して指定します。複数のキーを指定するときは、キー名を優先順に","で区切ります。 • limit: 返却する検索数の上限を指定します。デフォルト値は 100 件です。limit パラメータを指定しなかった場合、上限は 100 件となります。limit パラメータに-1 を指定した場合は上限無しとなります。 • skip: 検索結果の先頭からのスキップ数を指定します。デフォルト値は 0 件です。
リクエストボディ	無し
レスポンスボディ	以下のプロパティを持った JSON 形式。 • id: リクエストごとに発行される ID • query: リクエストされたクエリ文字列 • statusUrl: 実行状況のステータスを表すファイルの URL • resultUrl: 検索結果ファイルのオブジェクト URL
statusUrl のファイル内容	以下のプロパティを持った JSON 形式。 • id: リクエストごとに発行される ID • status: “RUNNING”/”DONE”/”CANCEL”のいずれかの値 • description: 実行中断の理由 (status が“CANCEL”となった場合のみ)

resultUrl のファイル内容	{"results": ["クエリ結果文字列 1", "クエリ結果文字列 2", …], "currentTime": "クエリ実行時刻"} という形式の JSON。
備考	

この API を使用するためには、検索結果ファイルの転送先として指定する S3 のバケット(クエリパラメータ ”bucket” の値として指定するバケット)に、DA2 の VPC エンドポイント「vpce-0f1df52c59f2c1c4f」からのアクセスを許可する設定が事前に必要です。
※バケットのリージョンは「アジアパシフィック（東京)」のみ対応しています。

以下は、DA2 の VPC エンドポイントからのアクセスを許可するバケットポリシーの例です。

```
{
    "Version": "2012-10-17",
    "Id": "xxxxxxxx",
    "Statement": [
        {
            "Sid": "xxxxxxxx",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::<バケット名>",
                "arn:aws:s3:::<バケット名>/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:SourceVpce": " vpce-0f1df52c59f2c1c4f"
                }
            }
        }
    ]
}
```

IoT 集配加工マネージドサービス 利用ガイド（開発編）

2021 年 5 月
日本電気株式会社