\Orchestrating a brighter world NEC

NECモバイルバックエンド基盤入門 応用編

ver 7.0.0

2018年3月26日

NEC IoT基盤開発本部

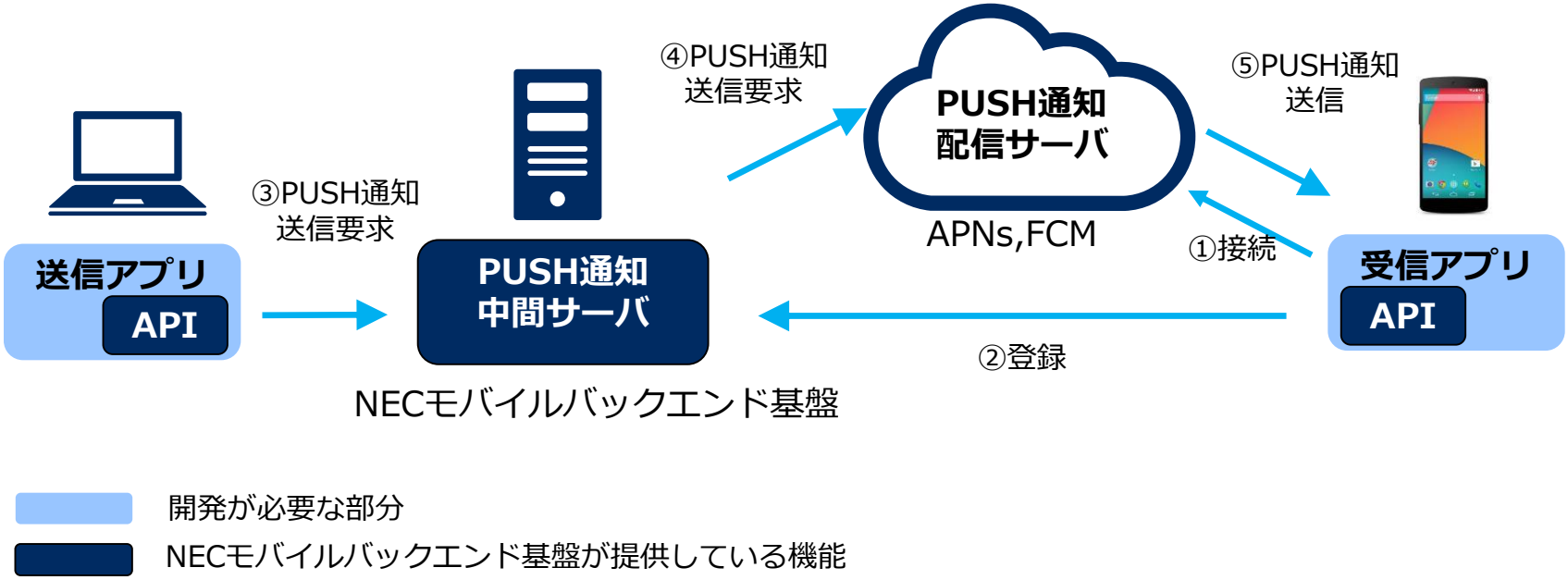
Push通知

クライアントへの Push 通知：APNs / FCM

Android / iOS で使用可能

ユーザがアプリを起動していなくても、サービス提供側からスマートフォンやタブレットにメッセージを送る仕組み。

- モバイル機器はプラットフォーム毎に Push 通知の仕組みが提供されています。これらの仕組みを使う場合には、Push 用のサーバをアプリ提供側で用意する必要があります。モバイルバックエンド基盤はこの PUSH 通知中間サーバ機能を提供しています。
- 使用できる Push 通知の種類
  - Apple 社が提供する APNs(Apple Push Notification Service)：iOS デバイスで利用可能
  - Google 社が提供する FCM (Firebase Cloud Messaging) / GCM (Google Cloud Messaging)：Android デバイスで利用可能

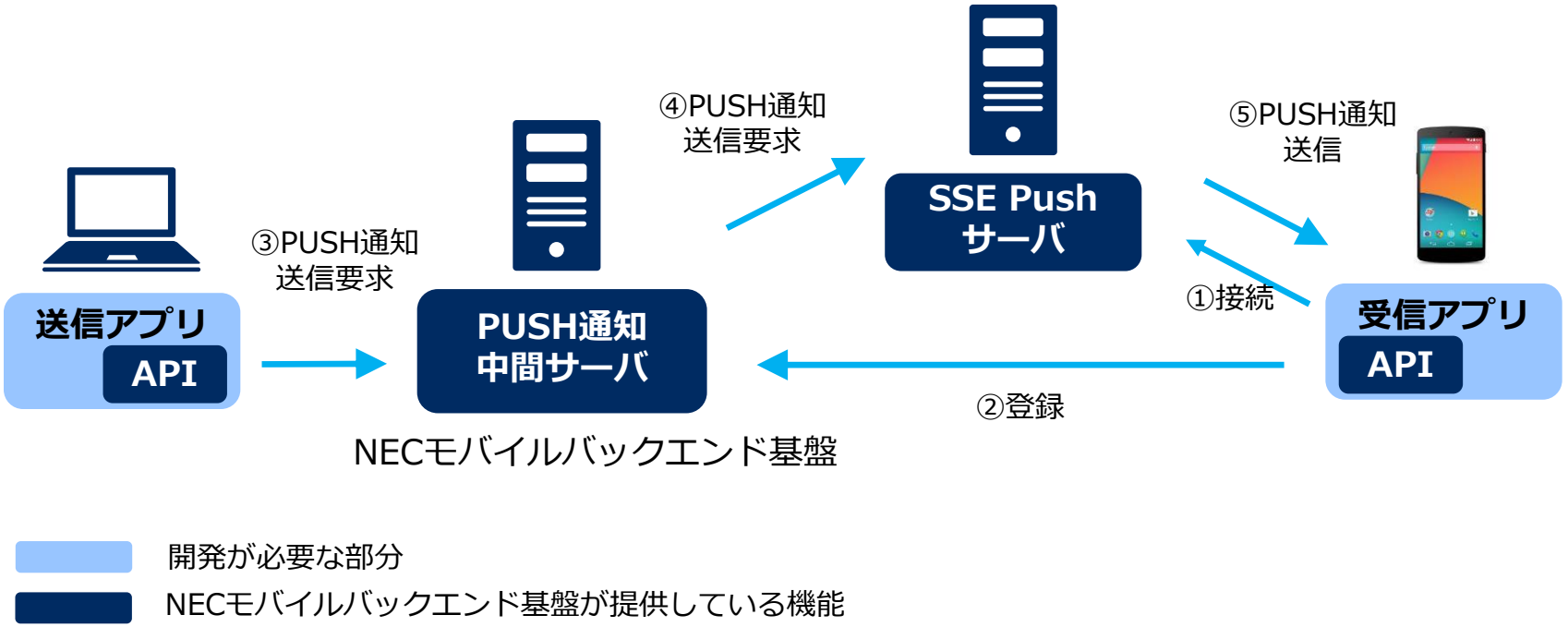


クライアントへの Push 通知：イントラ Push (SSE Push)

Android / .NET / Node.js で使用可能

外部サービスを使用せず、イントラネット内に設置した SSE Push サーバを用いて Push 配信を行うことができます。

- SSE (Server Sent Event) に準拠した「SSE Push サーバ」を提供します。
  - FCM や APNs に依存しないため、イントラネット・オンプレミスで運用することが可能です。
- クライアントとして Android と Windows を使用可能です。
- インスタレーションの概念は FCM / APNs と全く同じです。
- Push 送信側は、APNs, FCM と同じ仕組みで SSE Push も送信することができます。



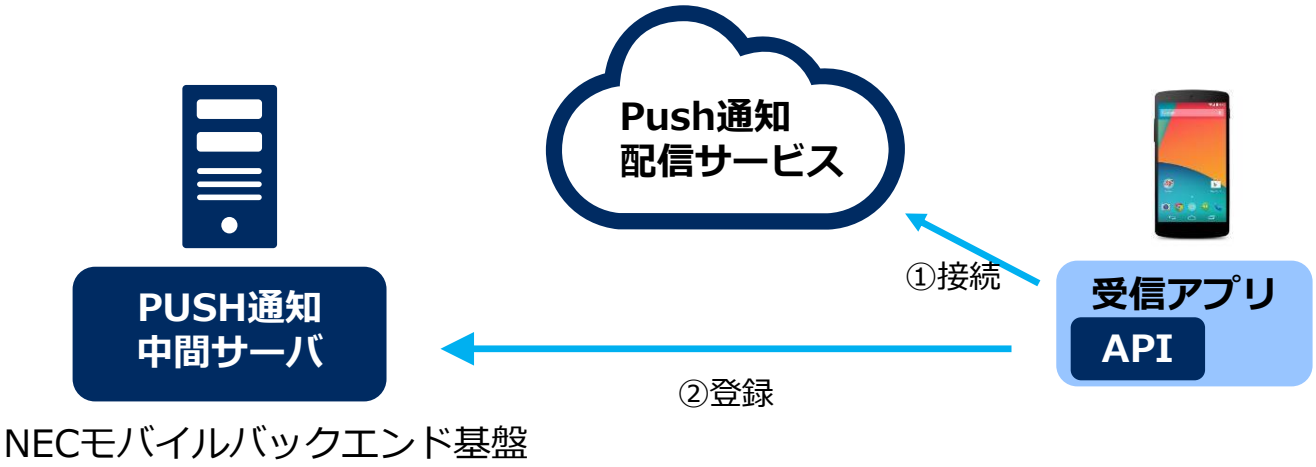
Push 機能: 受信アプリとインスタレーション

インスタレーション

- 端末にインストールされた受信アプリのインスタンスのことを「インスタレーション」と呼びます。 Push送信は、このインスタレーションに対して行います。

受信アプリの動作

- ①接続：
  - アプリインストール後にデバイスを識別するトークン(FCM の場合 Registration ID, APNs の場合 Device Token) が OS からアプリに対して払い出されます。 (アプリは開発時に FCM や APNs などへアプリ登録しておく必要があります。)
  - SSE Push の場合は、SDK 内でトークンが生成されアプリに渡されます。
- ②登録：
  - 受信アプリは、トークンをモバイルバックエンド基盤へ送信しインスタレーション登録を行います。送信側でインスタレーションを絞り込む必要がある場合には、インスタレーション登録時に購読するチャネルを指定します(Pub/Subモデル)。チャネルには任意の文字列が使えます。



Push 機能: 送信アプリと Push 通知送信要求

送信アプリの動作

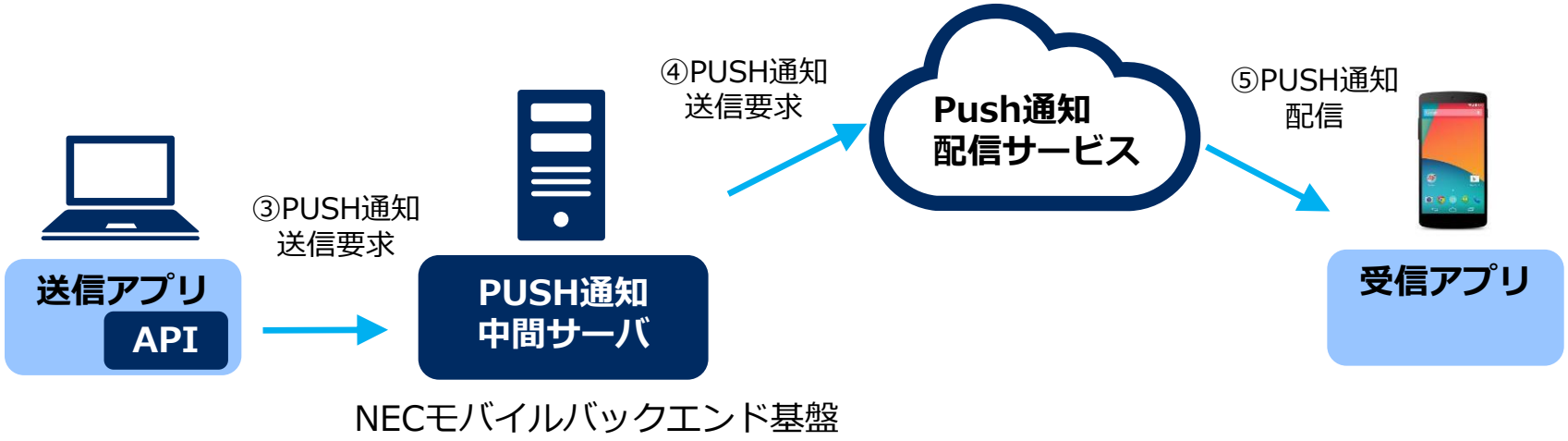
- 送信アプリはモバイルバックエンド基盤上で Push 通知対象となるインスタレーションを検索条件で絞り込みます。
  - アプリを使用する全端末、アプリが特定のバージョンである端末、特定のチャネルを購読する端末等
  - 特定のグループやユーザに通知を絞り込むことが可能です。
- ③ PUSH通知送信要求： 絞り込んだインスタレーションへPush 通知要求を行います。

モバイルバックエンドの動作

- ④ PUSH通知送信要求： 対象インスタレーションに対する Push 通知要求を Push 通知配信サービスへ送信します。

Push通知配信サービス、デバイスの動作

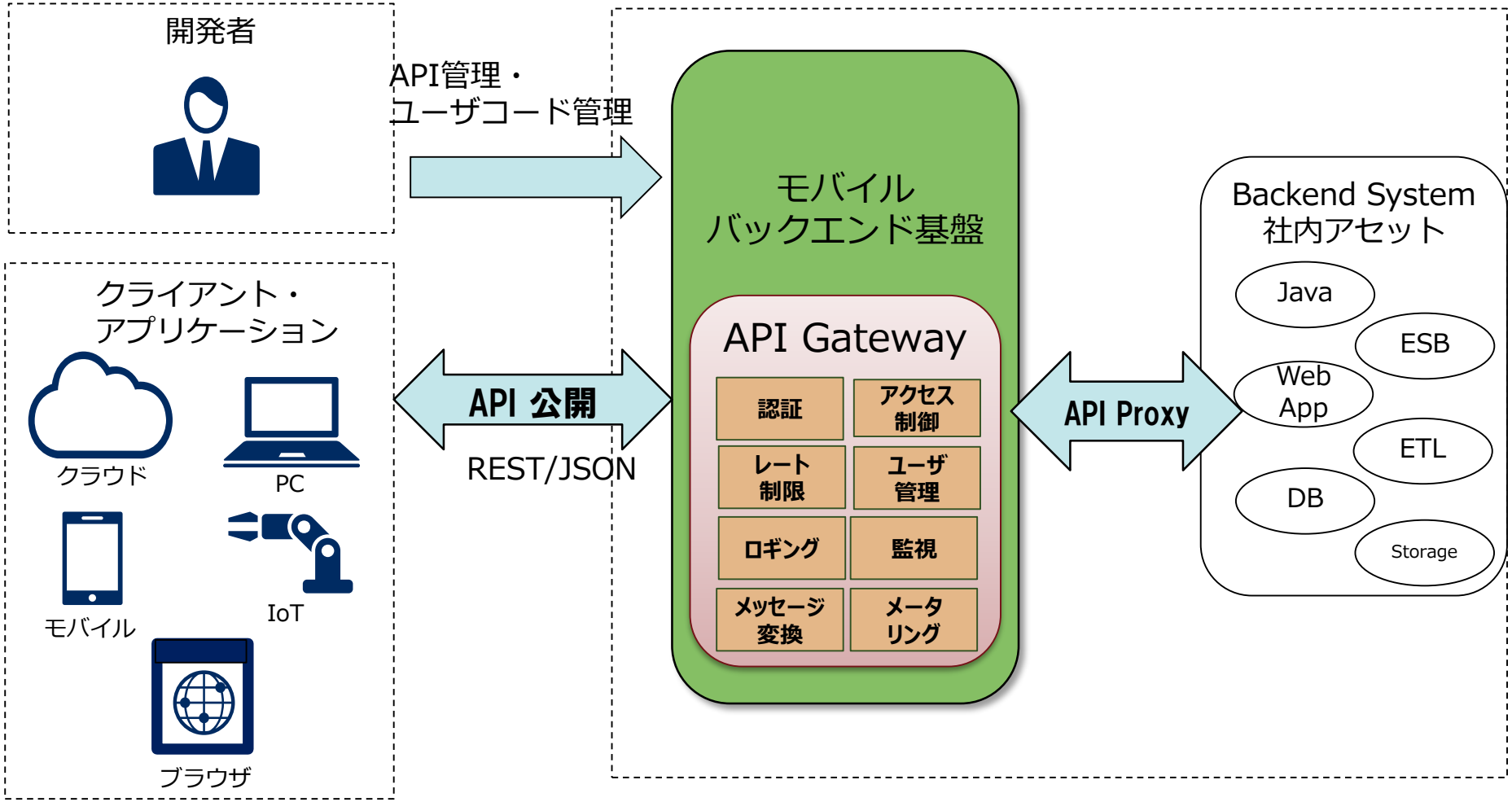
- Push通知配信サービスからデバイスに対して Push 通知の配信が行われます。受信アプリは API を使用して PUSH通知を受け取ります。



API Gateway / Cloud Functions

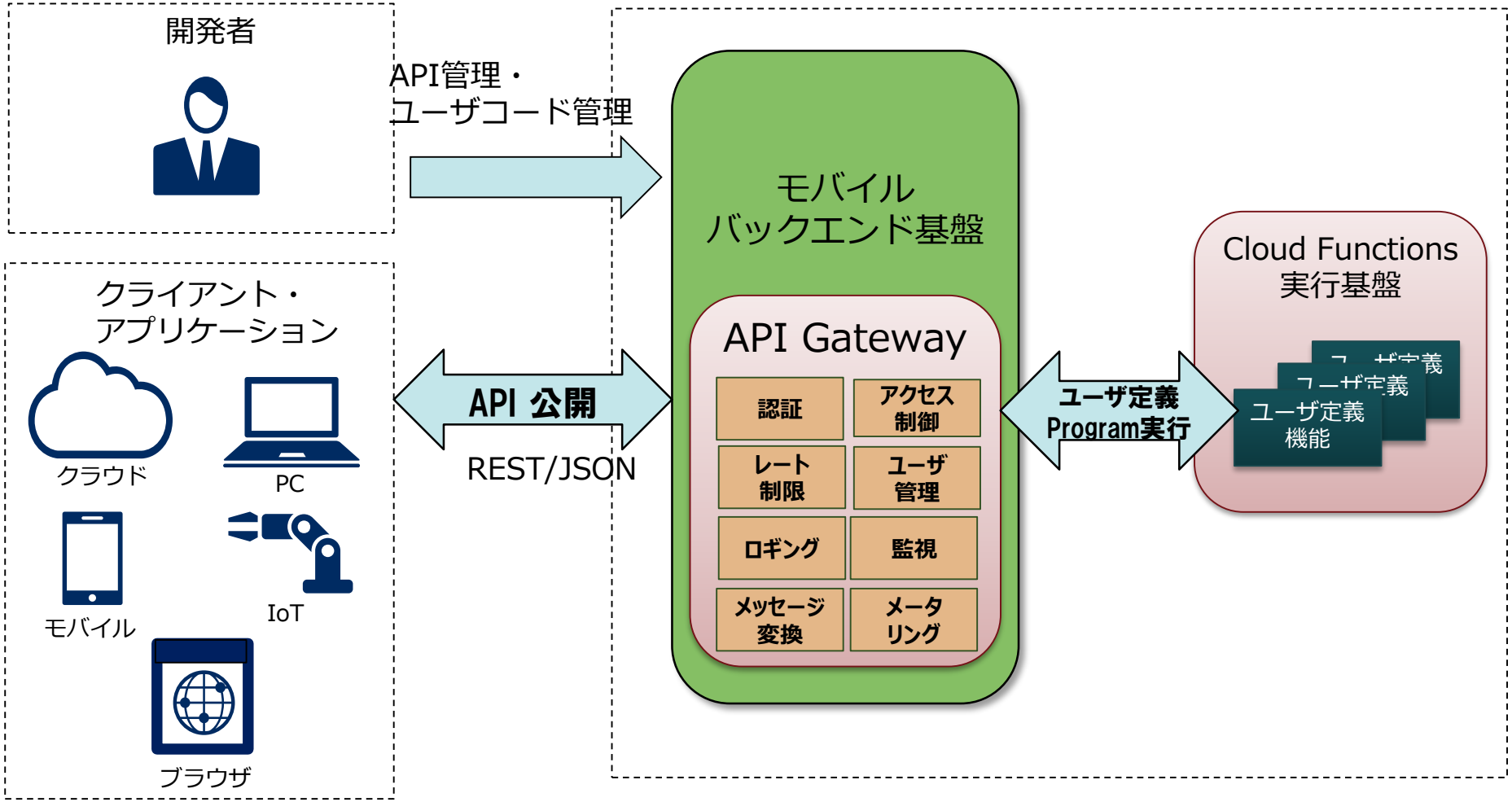
API Gateway とは

バックエンドに対する API 呼び出しを中継します。認証・認可・メッセージ変換・ロギングなどを集約することができます。



Cloud Functions とは

ユーザが独自に定義した機能をモバイルバックエンド基盤上で実行することができ、REST API で外部に公開できます。



API Gateway：機能

API Gateway は以下 2 つの機能を持ちます

- 1) モバイルバックエンド基盤以外のシステム(マイクロサービスなど)に対して REST API を Proxy することができます。以下の 2 つのプロトコルに対応しています
  - HTTP / HTTPS
  - RabbitMQ：Publish / Consume の両方が可能
  - Proxy 時に簡単なヘッダ変換・ボディ変換を行うことができます。
    - ヘッダ書き換え、クエリパラメータ書き換え、ボディ書き換え (JSON Patch など)
- 2) Cloud Functions に搭載した機能を外部システムから呼び出すことができます。

API Gateway：定義

REST API はユーザが独自に定義することができます。

- API は Open API 仕様 (Swagger仕様) 形式で定義します。
- API 定義はデベロッパーコンソールで設定することも、REST API やコマンドラインツールから投入することもできます。

定義する REST API は、以下のような URL となります。

- api-path の部分を自由に定義できます。

https://*{server-name}*/api/1/*{tenant-name}*/api/*{api-path}*

API Gateway：認証・認可

認証：認証は他の BaaS の REST API と同じように実施されます。

- ID/パスワードベース認証、クライアント証明書認証、APIキー認証、すべて使用できます。
- 認証時のログインユーザの情報などを後段のサービスに引き渡すことが可能です (HTTP Proxy の場合)

認可：API Gateway に認可制御情報を記述することで、アクセス制御が可能です。

- 特定のユーザやグループだけが API を実行可能なように指定することができます。

API Gateway：レートリミット・メータリング

レートリミット

- API Gateway の API 呼び出し数をレートリミット(スロットリング)により制限することができます。1分毎の呼び出し回数上限で指定します。
- 各 API 定義毎に上限数を個別に指定できます。

メータリング

- API Gateway の呼び出し回数は自動的に記録(メータリング)されます。
- 呼び出し回数は10分単位で記録されます。
- 集計用の REST API を呼び出すことで、メータリング情報の取得が可能です。また、デベロッパーコンソール上で呼び出し状況グラフを確認することができます。

Cloud Functions

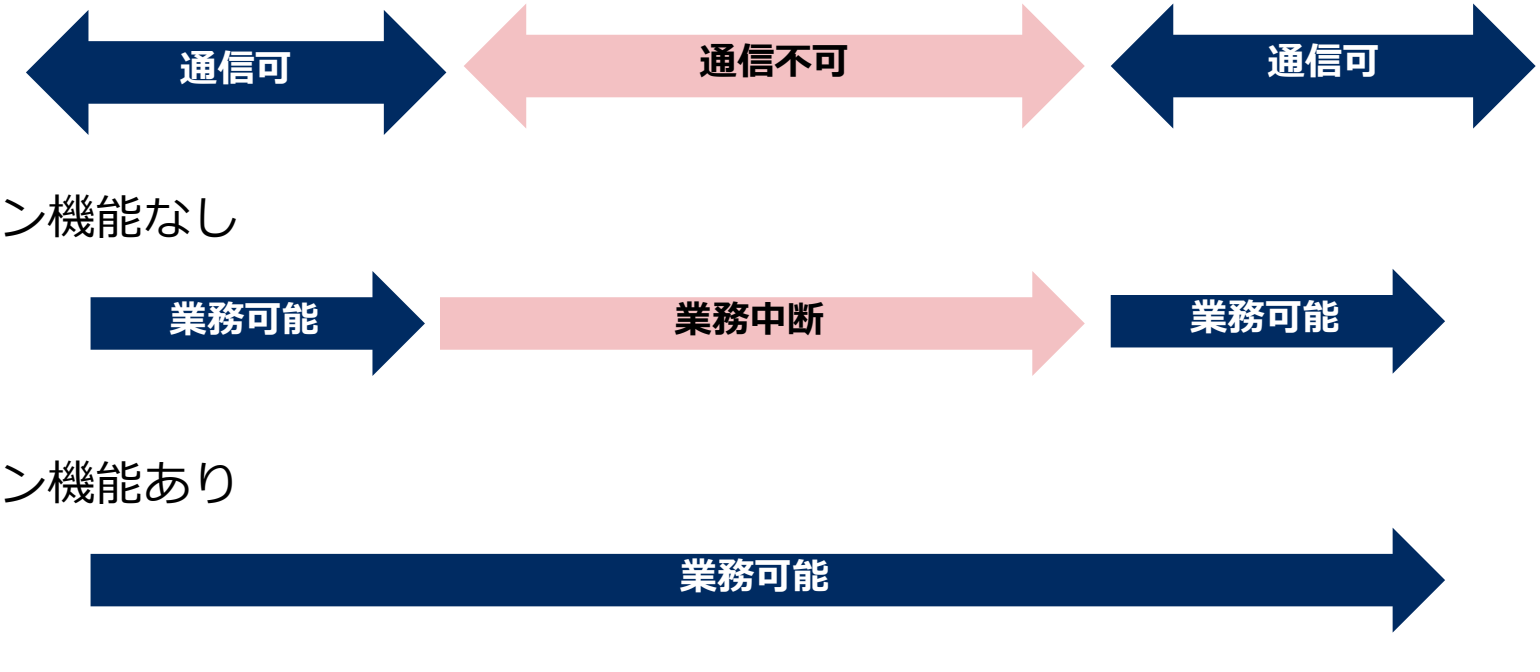
- ユーザが独自に定義した機能(ファンクション・プログラム)をモバイルバックエンド基盤上で実行することができます。
  - AWS Lambda や Google Cloud Functions に類似の機能です。
- いわゆる「サーバーレス」「Function as a Service」を実現することができます。
- 機能は Node.js または Java を使用して実装することができます。
  - ローカル環境でファンクションを開発し、コンパイル・パッケージングした圧縮ファイルを BaaS サーバに登録しておきます。
  - API Gateway 経由でファンクションを呼び出します。
  - ファンクションは、サーバの設定次第で Docker コンテナ上で動作させることも、サーバ上で直接動作させることも可能です。

オフライン機能

オフライン機能：不安定な通信環境での業務利用

モバイル機器は常に安定した通信が可能とは限りません。通信が不安定な環境でもアプリの機能を継続して利用するためにオブジェクトのオフライン機能を提供します。

- モバイルアプリとサーバとの間でオブジェクトの集まりを同期できます。
- 安定した通信環境下でデータをサーバとあらかじめ同期しておくことにより、不安定な通信環境下でもオブジェクトを参照、更新することができます。更新されたオブジェクトは次に同期された際に自動的にサーバに反映されます。

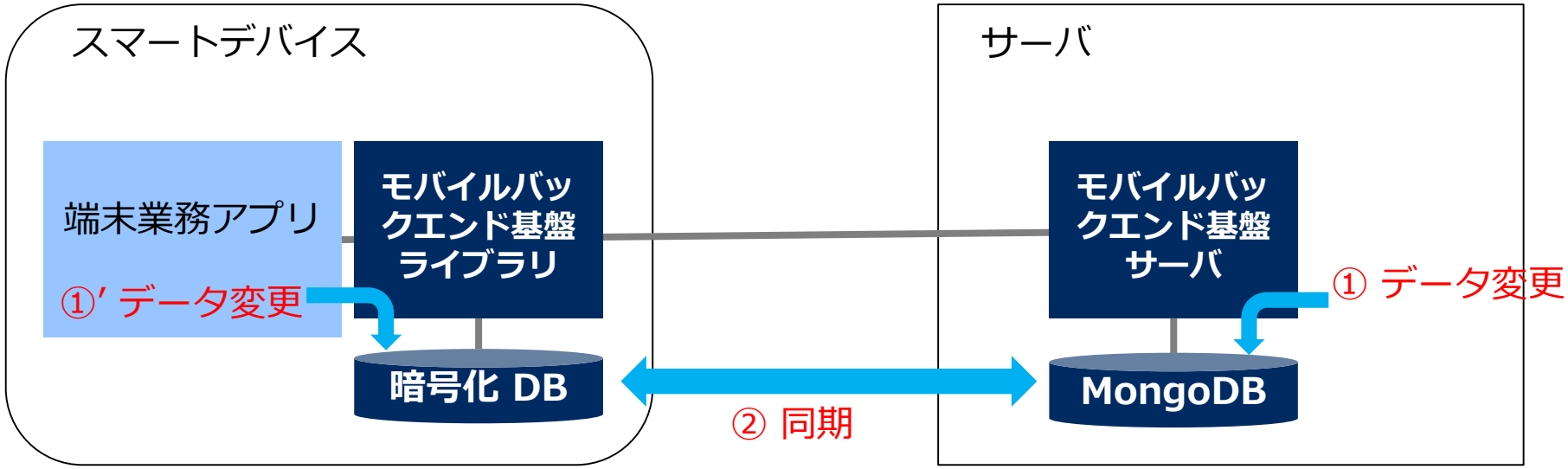


オフライン機能

Android (SDE4SD含む) / .NET で使用可能

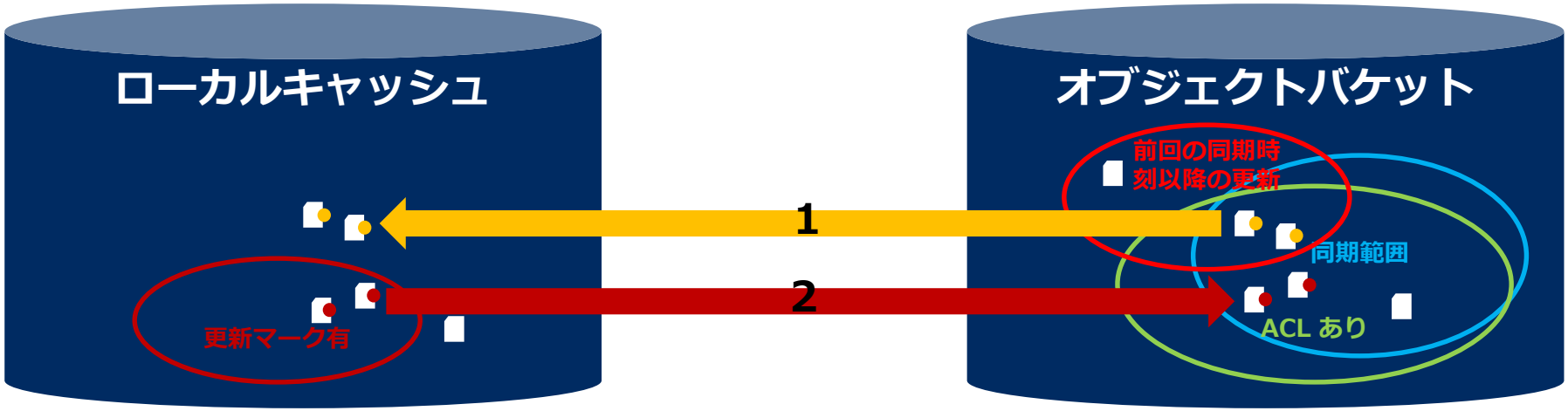
オフライン機能は、サーバ上にあるオブジェクトバケットの情報をクライアントのローカルストレージ(暗号化DB)にキャッシュします。ネットワークに接続していないオフライン環境でも、ローカルストレージに情報があるため、必要なデータにアクセス可能です。

- ローカルストレージのデータとサーバ上のデータはお互いに更新されるため、ずれが蓄積されます。同期を実行することでずれを解消できます。
- 同期範囲を指定することにより、サーバ上のオブジェクトバケットから一部の情報だけを同期することができます。
- ローカルのストレージ上のデータは暗号化され保持します。



オフライン機能: 同期の仕組み

- 同期処理では、クライアント端末のローカルDBとサーバ側のDBとの間で更新のあったデータを同期します。
- 実際の処理の流れ：
  - 通信量を削減するため、前回同期から更新のあった差分データを同期します。
  - 同期処理では、下記 1, 2 の処理を行っています。
    1. サーバ側で更新のあったデータを受信(「前回の同期時刻以降の更新 and 同期範囲内 and ログインユーザの利用権あり」の条件で対象バケットを検索)
    2. クライアント側で更新したデータをサーバへ送信(ローカルで更新のあったデータにはマークが付けられており、同期成功後にマークは削除される)




Orchestrating a brighter world
NEC