\Orchestrating a brighter world NEC

NECモバイルバックエンド基盤入門
基礎編

ver 7.5.0

2018年10月1日

日本電気株式会社

NECモバイルバックエンド基盤入門

NECモバイルバックエンド基盤とは

主な機能の紹介

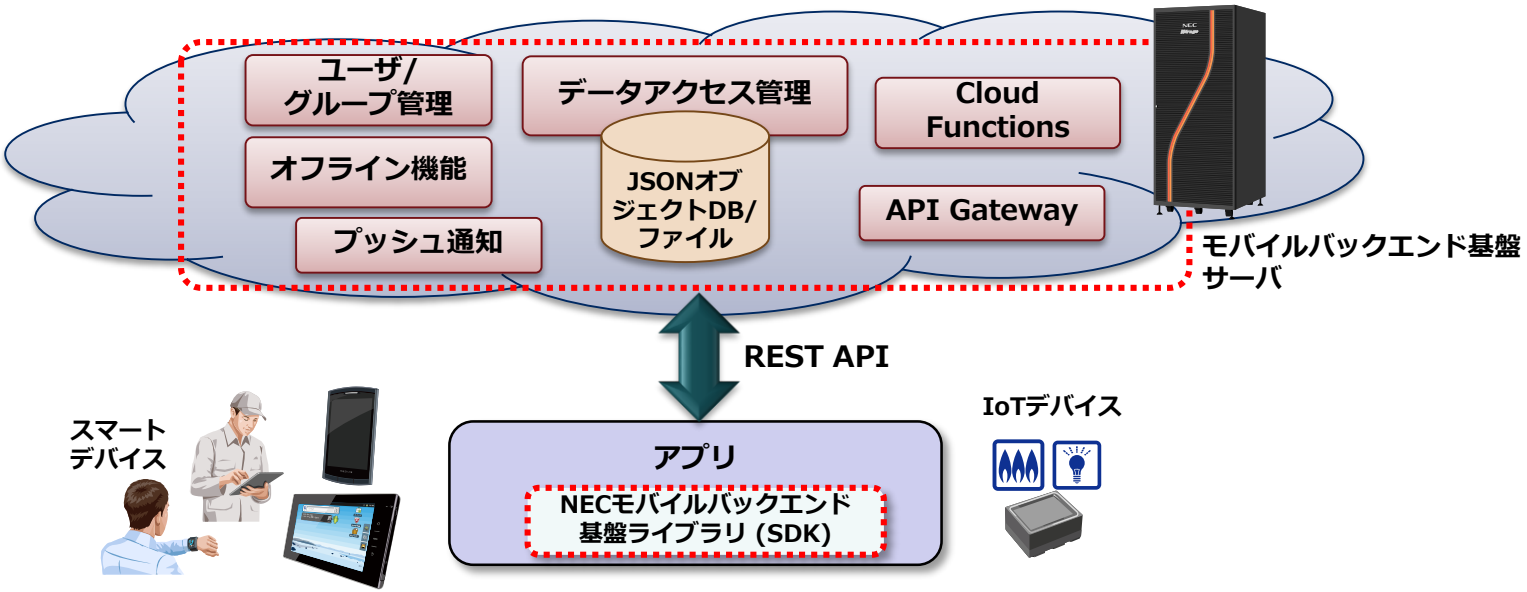
クライアントSDK

ライセンスモデルの概略

NECモバイルバックエンド基盤とは

NECモバイルバックエンド基盤とは

NECモバイルバックエンド基盤は、 PCやスマートデバイスおよびIoTデバイスのクライアントアプリケーションの開発に必要となる汎用的なバックエンド機能を提供する基盤ソフトウェアです。



主な機能	説明
ユーザ管理	ユーザのログインやアクセス権の管理を行う
JSONオブジェクトストレージ	JSON形式の非定形データを管理する
ファイルストレージ	ファイルデータを管理する
Push	クライアントへ Push 通知を行う
オフライン	通信ができない環境でも業務を継続する
API Gateway	カスタム REST API を定義し既存システムと接続する
Cloud Functions	クラウド側にユーザ定義の新たな機能を追加する
認証連携	LDAP/SAML/OpenID Connect など外部認証サーバを使った認証を行う
マルチテナント	異なるシステムを同居させる

NECモバイルバックエンド基盤と BaaS

- NECモバイルバックエンド基盤は「BaaS」を実現するソフトウェア基盤です。
- BaaS とは Backend as a Service の略で、IoT・モバイルアプリ開発で必要なサーバ側機能を提供するクラウドサービスのことです。
- BaaS はクラウドサービスですが、モバイルバックエンド基盤はサーバ機能をソフトウェア部品としても提供します。オンプレミスのサーバやプライベートクラウド上に BaaS 機能を独自に構築することが可能です。

BaaS のメリット

BaaS を利用することにより次のようなメリットがあります。

- 開発コスト削減
  - サーバ側開発規模の削減により開発コストを削減可能
- 開発期間の短縮
  - サーバ側機能の開発期間削減やサーバとの連携機能の評価期間短縮により、開発期間の短縮が可能
- アプリへの注力
  - サーバ側機能の検討が不要であるため、アプリの開発に注力できる

NECモバイルバックエンド基盤の特徴

NECモバイルバックエンド基盤は業務向けの機能を強化しています。

- サーバ上のデータやファイルへのアクセスコントロールをきめ細かく行えます。
- ユーザや IoT デバイスの認証を統合することができます。
- マイクロサービスを連携したり外部に安全に公開することができます。
- クラウド側にユーザ定義の機能を自由に追加することができます。
- モバイルデバイスで良く使われる Push 通知の機能があります。
- 不安定な通信環境での業務利用を支援するオフライン機能があります。

きめ細かいアクセスコントロール	ユーザ・デバイス認証を統合する	マイクロサービスを連携・公開する
クラウド側に新たな機能を追加する	モバイルデバイスへの Push 通知	不安定な通信環境での業務利用

主な機能の紹介

主な機能の紹介

- ユーザのログインやアクセス権の管理を行う ⇒ ユーザ管理機能
- ユーザやIoTデバイスの認証を統合する ⇒ 認証機能
- 非定形データを管理する ⇒ JSONオブジェクトストレージ
- ファイルを管理する ⇒ ファイルストレージ
- 異なるシステムを同居させる ⇒ マルチテナント機能

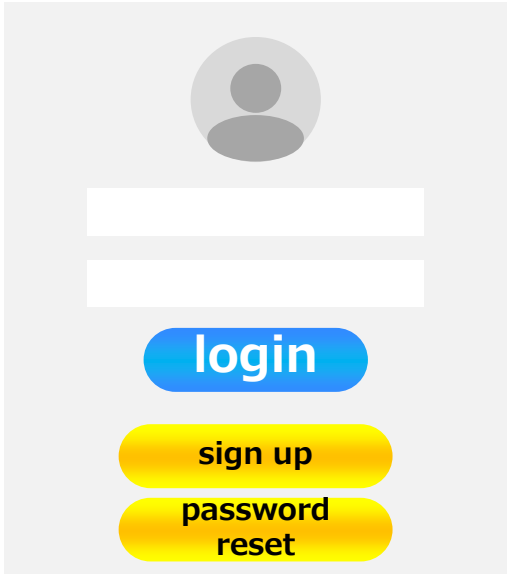
- 以下の機能は応用編で説明しています。
  - クライアントへ Push 通知を行う ⇒ Push
  - マイクロサービスの API 呼び出しを連携・外部公開する ⇒ API Gateway
  - クラウド側に新たな機能を追加する ⇒ Cloud Functions
  - 通信ができない環境でも業務を継続する ⇒ オフライン機能

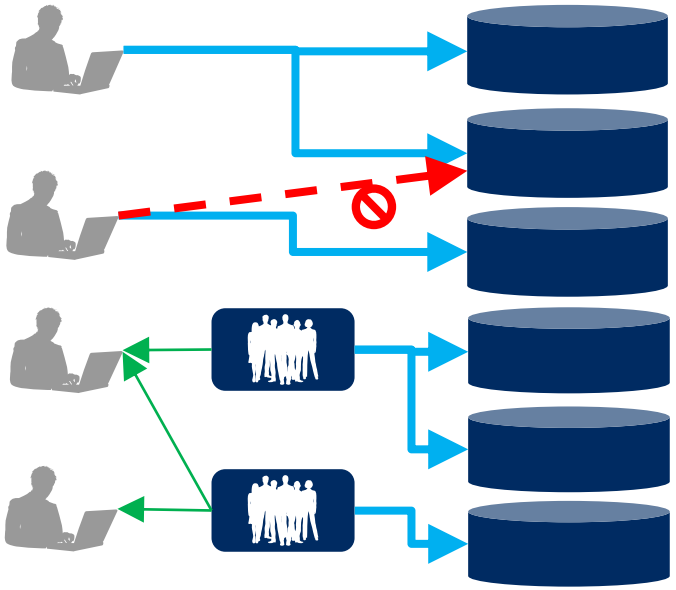
ユーザ・グループ管理

ユーザ・グループ管理 / 認証機能

ユーザ・デバイスのログインの管理やリソースに対するアクセスコントロールのための機能を提供します。

- アプリケーションへユーザを認証するための基本機能を提供しています。API をコールするだけでログイン、ログアウトなどを安全に行う事ができます。
- ユーザのサインアップ、パスワードリセットの機能を提供します。
- 後述するオブジェクトやファイルデータに対してどのユーザがアクセスすることができるかといったアクセス制御を細かく行うことができます。
- グループによりユーザを束ねてアクセスコントロールすることができます。グループは階層化することが可能です。





認証機能

ヒト・モノ・サービスの認証

BaaS では、ヒト・モノ・サービスをそれぞれ認証することができます。
「ヒト」「モノ」はどちらも「ユーザ」として扱います。

1) 「ヒト」の認証

- ID/パスワードベースの認証を使用します。
- ローカル認証、LDAP、SAML、OAuth2/OpenID Connect 認証が使えます。

2) 「モノ」の認証

- クライアント証明書ベースの認証を使用することを推奨しますが、ID/パスワードベース認証も利用できます。

3)「サービス」の認証

- サービスについては、APIキーを使用した認証が基本になります。
- クライアント証明書や ID/パスワードベースの認証も利用できます。

認証方式

以下の認証方式をサポートしています。

1) ローカル認証

- BaaSに内蔵されるユーザデータベースを使用したID/パスワード認証です。
- ユーザのサインアップ・パスワードリセット機能などが利用可能です。

2) LDAP 認証

- LDAP (ActiveDirectory) を使用した認証です。
- ActiveDirectory を使用する場合、ユーザが所属するグループを自動的に BaaS 側に同期することができます。

3) SAML, OAuth2 / OpenID Connect 認証

- 外部の認証サーバ(Identity Provider)を使用した認証連携を行うことが可能です。
- Active Directory Federation Service (ADFS)との連携、および OpenAM, Google などとの認証連携が可能です。

4) クライアント証明書認証

- IoTエッジデバイスなどから、X.509クライアント証明書認証を使用した認証が可能です。

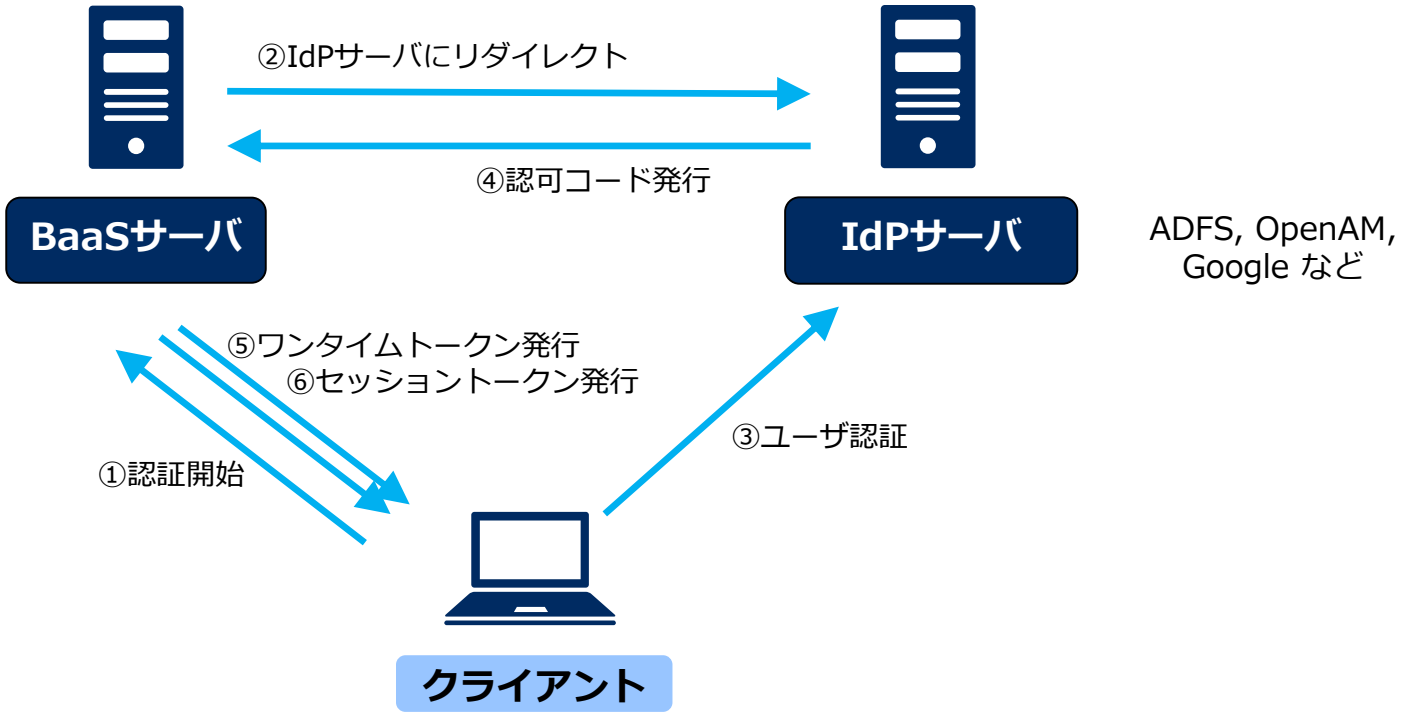
5) APIキー認証

- アプリケーション・サービス毎に払い出される API キーをベースとした認証です。
- 1) ～ 4) と同時に実施することができます。

SAML / OpenID Connect

外部認証サーバ(IdP: Identity Provider) を使用した認証連携が可能です。

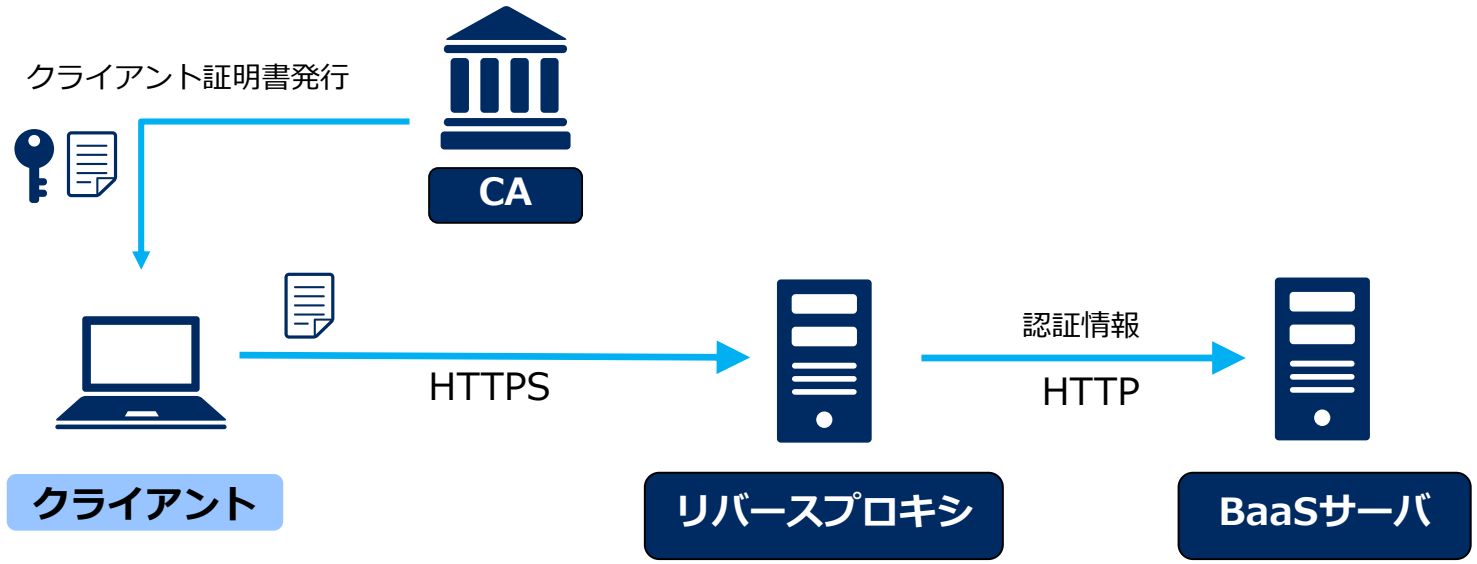
- BaaSサーバからIdPサーバに認証を委譲する形になります。BaaSサーバ側には自動的にユーザ情報が転記されます。
- ブラウザ、ネイティブアプリケーション(Java/Android, C#) で利用することができます。



クライアント証明書認証

クライアント証明書を使用したクライアント認証が可能です。

- クライアント証明書認証を使用する場合は、プライベートCA(認証局)を用意し、クライアント(IoTデバイスなど)毎に証明書を発行します。
  - クライアントの識別子は、クライアント証明書の CN (Common Name) フィールドに格納します (他に uid なども使用可能)
  - BaaSサーバは、CN の値を「username」としてユーザにマッピングします。必要に応じて、BaaSサーバ上にユーザを自動作成することができます。
- BaaSサーバの前段に HAProxy などのリバースプロキシを配置し、クライアント証明書認証を実施します。
  - リバースプロキシは、HTTPヘッダを使用して BaaS サーバに認証情報を引き渡すように設定します。



JSON オブジェクトストレージ

JSONオブジェクトストレージ

モバイルバックエンド基盤ではJSONオブジェクトストレージというオブジェクトデータベースによりデータを扱います。

- JSON 形式のオブジェクトを取り扱うことができます。
- リレーショナルデータベース(以降RDB）との比較：
  - ○ RDB はタプルでデータを表現するのに対し、JSONオブジェクトを直接扱うことができます。複雑な形式のデータでも変換せずにそのまま扱うことができます。
  - ○ RDB はスキーマを決める必要があるのに対し、 スキーマレスで運用できます。データ構造の変更や複数の形式のデータの混在も容易です。
  - × RDB では複数レコードの変更をトランザクションでアトミックに扱えるのに対し、複数のオブジェクトの更新をアトミックに扱うのは苦手です (部分的には可能)

JSON 形式のデータがあつかえる

スキーマレスでの運用が可能

複数レコードのアトミック更新が苦手

オブジェクトバケット

「オブジェクト」と「オブジェクトバケット」という単位でデータを管理します。バケット、オブジェクト毎にアクセス権の設定ができます。

- オブジェクト
  - オブジェクトは JSON 形式のデータです。 JSON形式のメリットとして、Java や JavaScript からの利用が容易です。
- オブジェクトバケット
  - バケットは RDB でいうところの「テーブル」に該当する概念です（JSONオブジェクトは RDB ではテーブルの１行に相当します）。
  - バケットは用途や種類別に用意することで、管理や検索が容易になります。

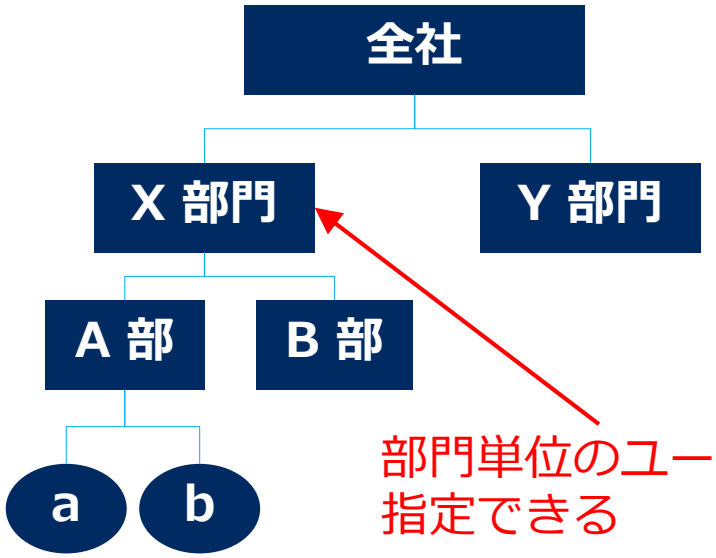
オブジェクトの操作

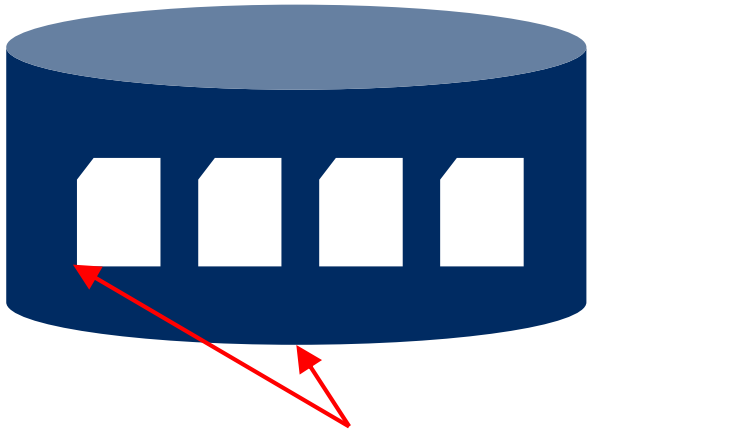
オブジェクトに対して次のような操作を行えます。

- 生成：
  - JSON 形式のオブジェクトを格納できます。
- 検索： バケットからオブジェクトを検索する際、様々な条件を付与して検索を行うことができます。アクセス権のあるオブジェクトが対象になります。
  - クエリ式 (JSON) による検索、ソート条件の指定、検索件数上限の指定など
- 更新： オブジェクトの更新を次のような条件で行えます。
  - フルアップデート/部分更新、楽観ロックあり/なし
- 削除： オブジェクトを削除します。
  - 論理削除：オブジェクトに削除フラグを付けます。
  - 物理削除：オブジェクトそのものを削除します。
  - 一括削除： 検索同様にクエリ式で指定したオブジェクトを削除します。
- バッチオペレーション：複数オブジェクトを同時に追加・更新・削除できます。
  - MongoDB v4.0の環境では、トランザクション内でバッチオペレーションを実行することで、アトミックな更新もサポートします。(v7.5以降)
- 集約(Aggregation)：高度なクエリを行なうことができます
  - RDB の JOIN, GROUP BY 相当の操作を行なうことができます。(v7.0 以降)

きめ細かなアクセスコントロール

- データやリソースに対してアクセスコントロールを行う事ができます。業務アプリを作成する場合に便利なようにきめ細かいアクセスコントロールを提供します。
  - グループを階層的に定義することが可能で階層的な組織構造に合わせたアクセスコントロールが可能です。
  - アクセス権は、ACL (Access Control List)で指定します。
    - Read/Write/Create/Update/Delete等の権限をユーザやグループ毎に指定できます。
  - 複数のデータを保持しているバケット単位と、１件ごとのデータ(オブジェクト)の両方にアクセス権の設定が可能なので、アプリケーションの要件に合わせて柔軟に運用することができます。
    - バケットのみにACLを設定し、オブジェクトのACLを省略する運用も可能です (v7.5以降)





ファイルストレージ

ファイルストレージ

- ファイルストレージは、バイナリ形式でのファイルを保存、読み出しすることができるストレージ機能です。
  - ファイルは、オブジェクトストレージと同様にバケット単位で管理することができます。アクセス制限をファイル単位で設定することが可能です。
  - ファイルは「公開」を行うことができます。公開を行うと、そのファイルには認証なしでアクセスすることができるようになります。

ファイルストレージ：バックエンド

ファイルストレージのバックエンドには以下のものが使用できます。

1) MongoDB

- デフォルトではファイルは MongoDB 内の GridFS に格納されます。
- 1個のファイルサイズの上限はおよそ 100MB 程度となります。(上限は設定で変更可能)

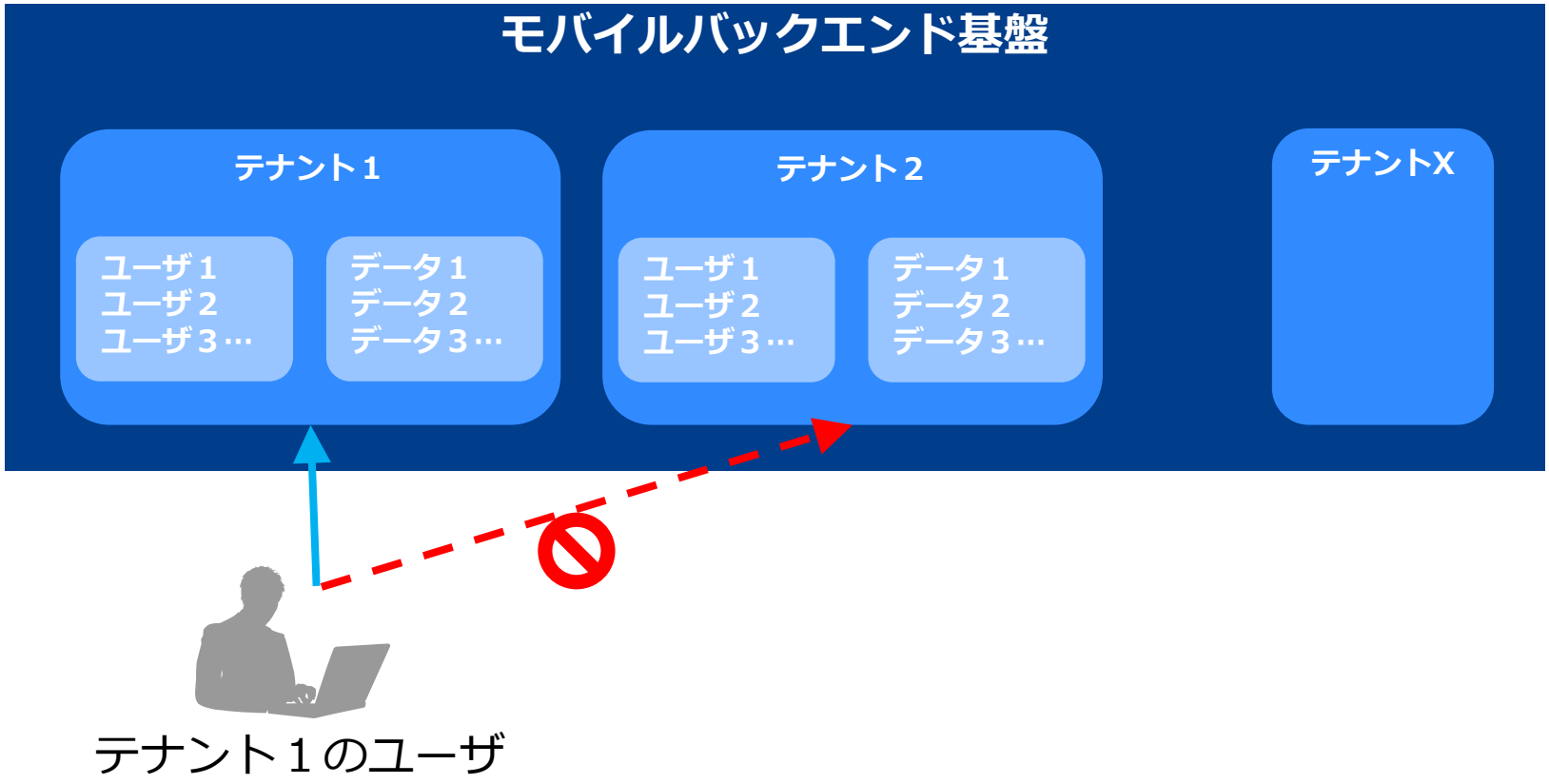
2) S3互換オブジェクトストレージ

- AWS S3 互換オブジェクトストレージをバックエンドに使用可能です。
- 1個のファイルサイズの上限はありません。 (S3オブジェクトストレージ側の制限のみ)
- 大量のファイルを安価なストレージ上に置くことが可能になります。

マルチテナント

マルチテナント

- モバイルバックエンド基盤のテナントとはユーザーやデータ、アプリケーションを保持する単位です。
  - アプリケーションからは自分が属していないテナントのデータにアクセスすることはできません。
  - ユーザーもテナント単位で登録可能です。



デベロッパーコンソール

デベロッパーコンソール

デベロッパーコンソールは、管理者に対してテナント、アプリケーション、管理者などの管理機能を提供するWebサービスです。

- デベロッパーコンソールはブラウザからアクセスし Web 画面で操作します。
- 管理者は権限の範囲でテナントの管理が行えます。

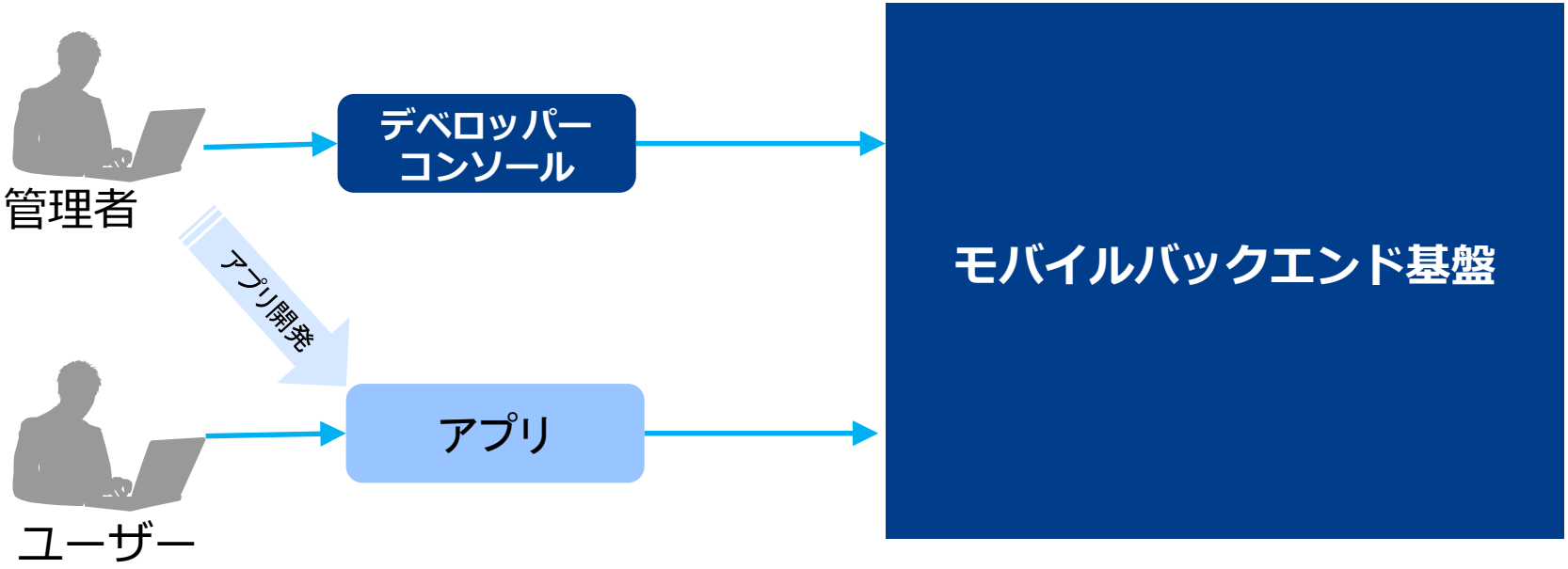
（デベロッパコンソール表示例）

デベロッパーコンソールの提供機能

項目	機能	機能概要
テナント	アプリケーション管理	アプリケーションの表示・編集、アプリケーションキーの発行等
	ユーザ管理	ユーザの表示・検索・編集、無効化、ユーザの所属グループ設定等
	グループ管理	グループの表示・編集
	オブジェクトバケット管理	オブジェクトバケットの表示・編集
	オブジェクト管理	オブジェクトバケット内のオブジェクト一覧表示、オブジェクトのインポート/エクスポート、ACL変更等
	ファイルバケット管理	ファイルバケットの表示・編集
	ファイル管理	ファイルバケット内のファイル一覧表示、アップロード・ダウンロード
	Push	Push送信
	API Gateway管理	API Gateway 定義の表示・編集
	Cloud Functions管理	Cloud Functions 定義の編集、ログ確認
	Event Subscription管理	イベント駆動のサブスクリプション定義の表示・投入
	API統計情報表示	APIコール数などの統計情報の確認
	管理者管理	テナントの管理権設定
	テナント設定	テナント詳細設定、S3互換ストレージ設定、クライアント証明書認証設定、テンプレート設定
システム	テナント管理	テナント作成・管理
	管理者管理	デベロッパーコンソール管理者作成・管理
	システム設定	一般設定、データベース設定、AMQP設定、メール設定など
	システムログ	システムログ確認

管理者とユーザ

- モバイルバックエンド基盤の利用者には管理者、ユーザーの２種類があります。
  - 管理者：
    - アプリケーション開発やテナントの管理を行う利用者。
    - デベロッパーコンソール機能を通してモバイルバックエンド基盤へアクセスする。
  - ユーザー：
    - デベロッパーが開発したアプリケーションを利用する利用者。
    - デベロッパーが開発したアプリケーションを通してモバイルバックエンド基盤にアクセスする。



クライアント SDK

クライアント SDKの構成

ネイティブ・ハイブリッドアプリケーション対応

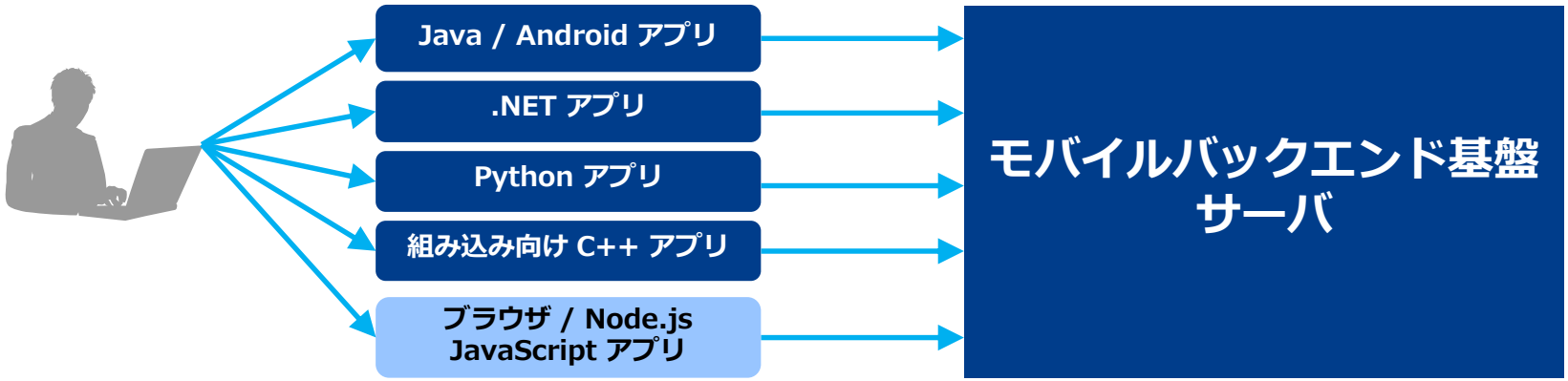
- Java / Android SDK：Pure Java および Android Java 向け
- JavaScript SDK：Node.js, WebKit(Cordovaなど)向け
- iOS SDK：iOS 向け (Objective-C / Swift) (※β版)
- Embedded SDK：組込デバイス向け (C++)
- Python SDK：Python 向け

ブラウザアプリケーション対応

- ブラウザから利用可能な JavaScript SDK を提供します。
- Android や iOS のブラウザからも利用可能です。

クライアント SDK はすべて OSS として公開しています。

- https://github.com/nec-baas



ライセンスモデルの概略

ライセンスモデルの考え方

モバイルバックエンド基盤を利用する際は、サーバのライセンス購入が必要となります。

- ライセンスは、「システム単位」で購入する必要があります。
  - テナント数・マシン台数・CPU数などには依存しません。マシンが何台あっても、またテナントが何個あっても、システムが1つならば必要なライセンス数は1個です。
  - システム数は、MongoDB に格納するモバイルバックエンド基盤のマスターデータベース数でカウントします。
- ライセンスには、「モデル」が何種類かあり、それぞれごとに性能上限が設けられています。
  - 具体的には、毎秒あたりのAPIコール数、最大登録ユーザ数などに上限があります。

SDK は OSS で提供しており、無償で利用可能です。


Orchestrating a brighter world
NEC