

改めてWeb APIについて
考えてみる

自己紹介

日本マイクロソフト株式会社
マイクロソフトテクノロジーセンター
Azure テクニカルアーキテクト
吉田 雄哉（吉田パクえ）

2015年1月1日入社
得意分野 オープンソース系



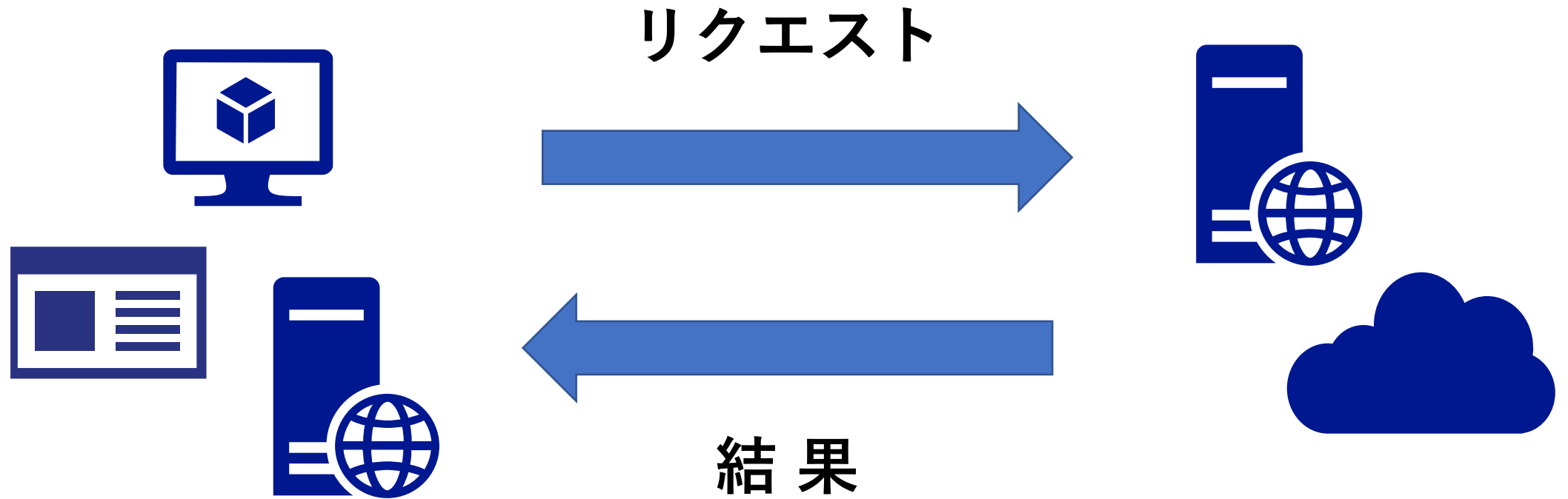
 yuyalush

 yuya_lush

Web APIとは

Web API

Web Application Programming Interface



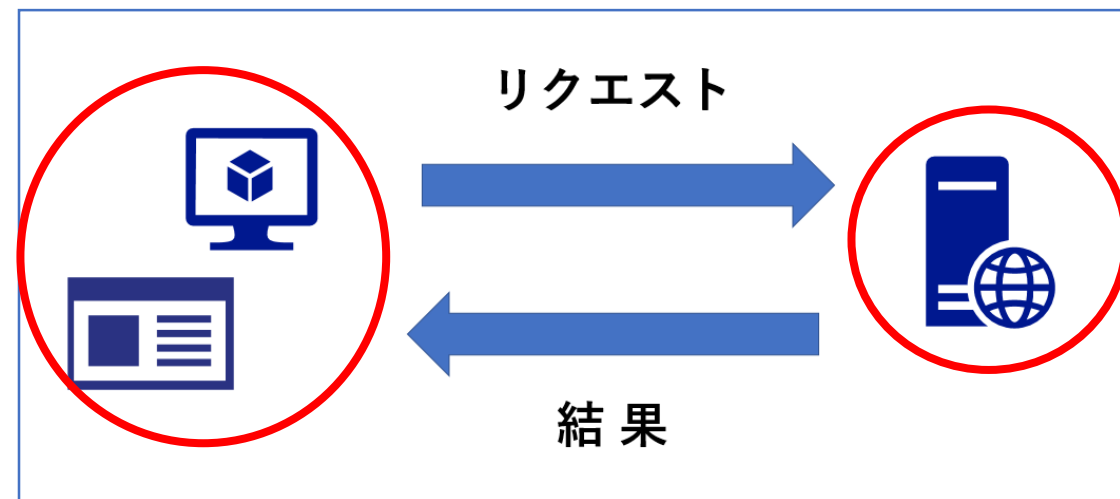
Web APIが登場する場面

リクエストを送る側

ブラウザやスマホアプリ、サーバーサイドのアプリケーション

リクエストを受ける側

サーバーサイドのアプリケーション
クラウドのサービス



Web APIが登場する場面

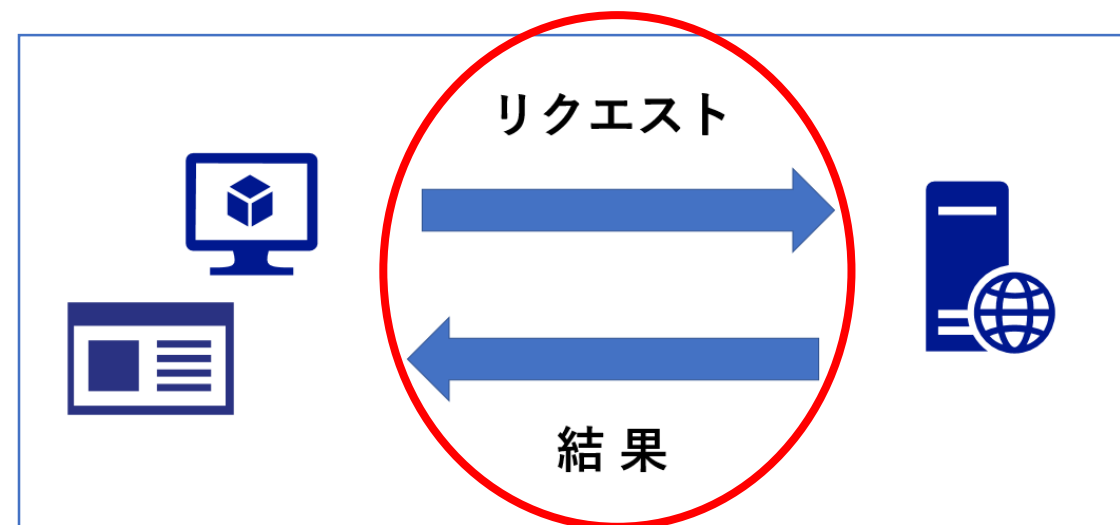
通信の方法

HTTP/HTTPSといった通常のWebで使われる技術

結果のフォーマット

JSONやXMLが一般的

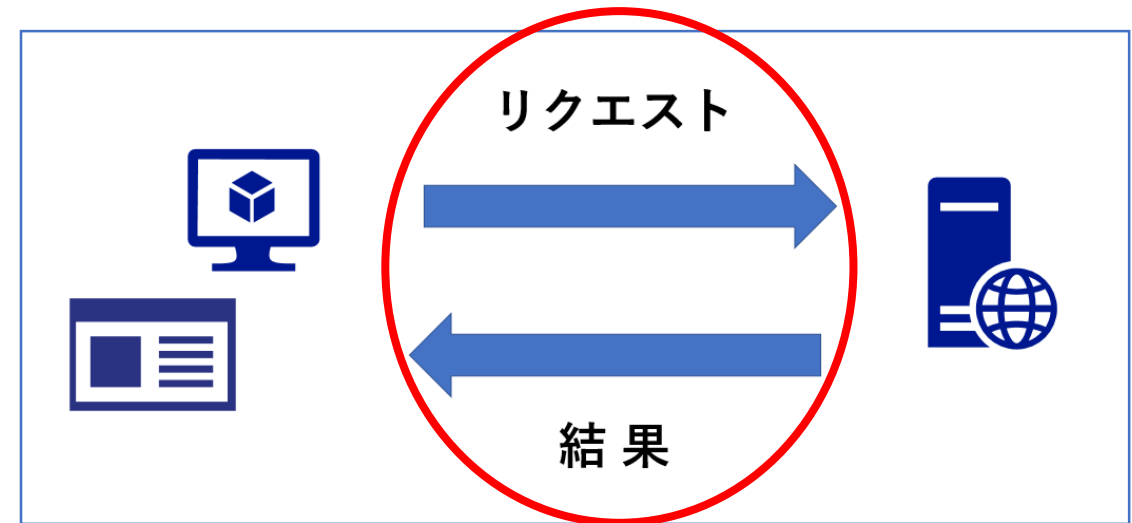
ユーザのビューが必要なく、
あくまでプログラム間でのやり取り



Web APIが登場する場面

なぜ、やり取りが必要か

- 処理をしてもらい結果を得たい
- データを取得したい



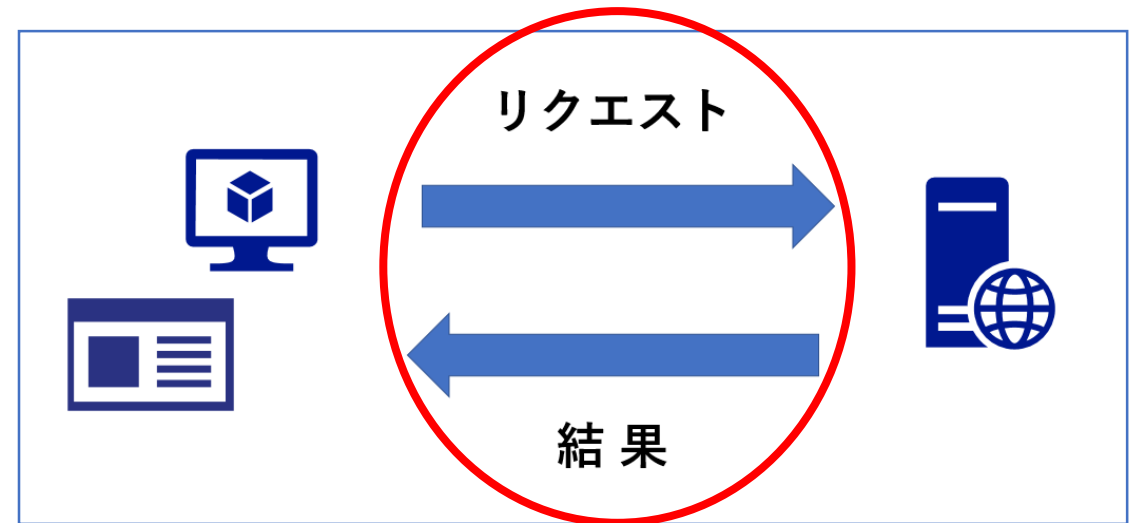
Web APIが登場する場面

さらに

両者がどこにいるのか

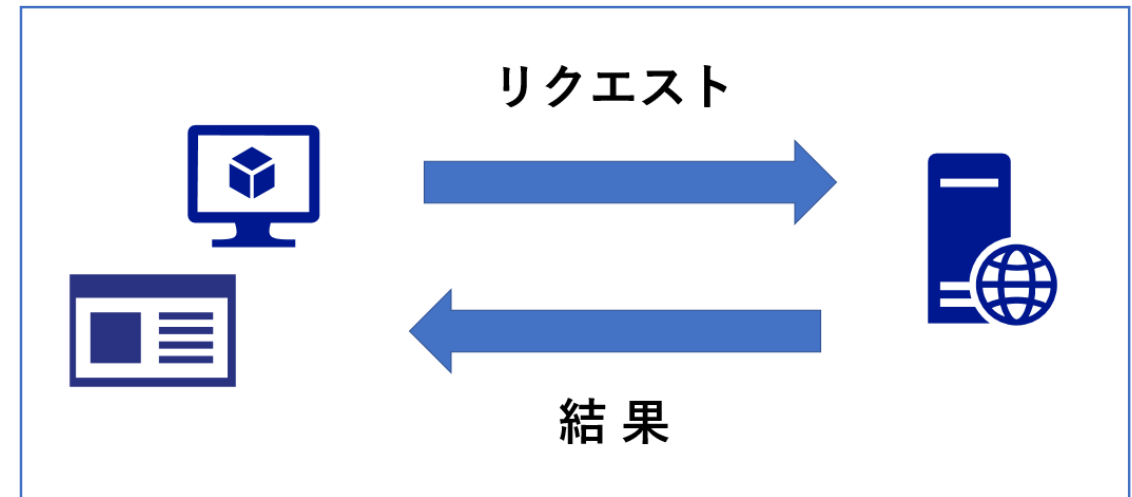
どのようなビジネスメリットを求めているのか

- 社内システム連携
- APIによるエコシステム
- クラウド上でのサービス連携



色々 必要になる

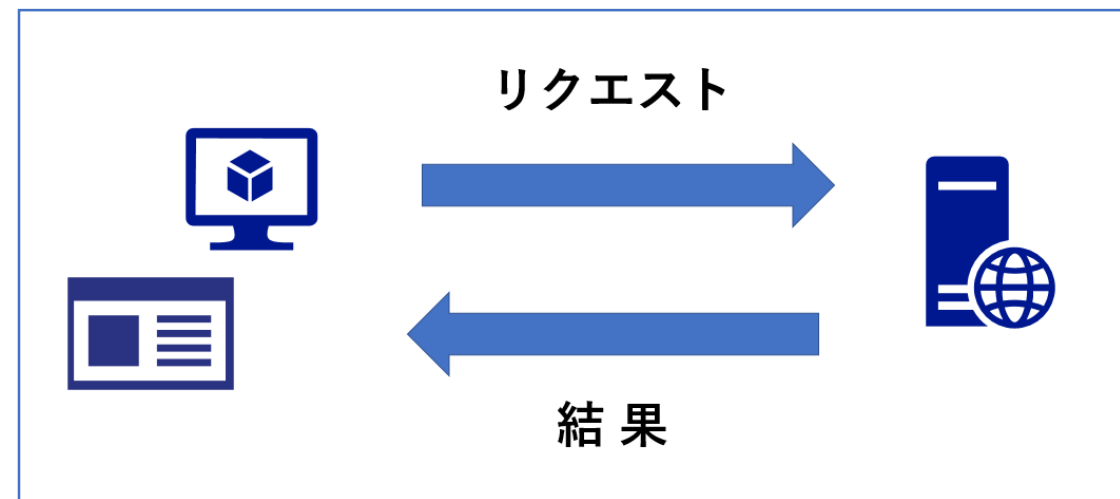
- 認証：リクエスト元を特定する
- 認可：アクセスを許可する
- 監査：活動内容、適正なリクエストか
- 流量制限：リソースの保護
- 提供：APIに関する情報を公開
- 請求：利用負担を求める
- 開示：利用状況を見せる



さらに

- スケーラビリティ
- セルフサービス化（トークンの発行、ポータルサイト）
- 機能強化（バージョン管理）
- SDKの提供

などなど



ビジネス要件を含めて考えるととても大変だ

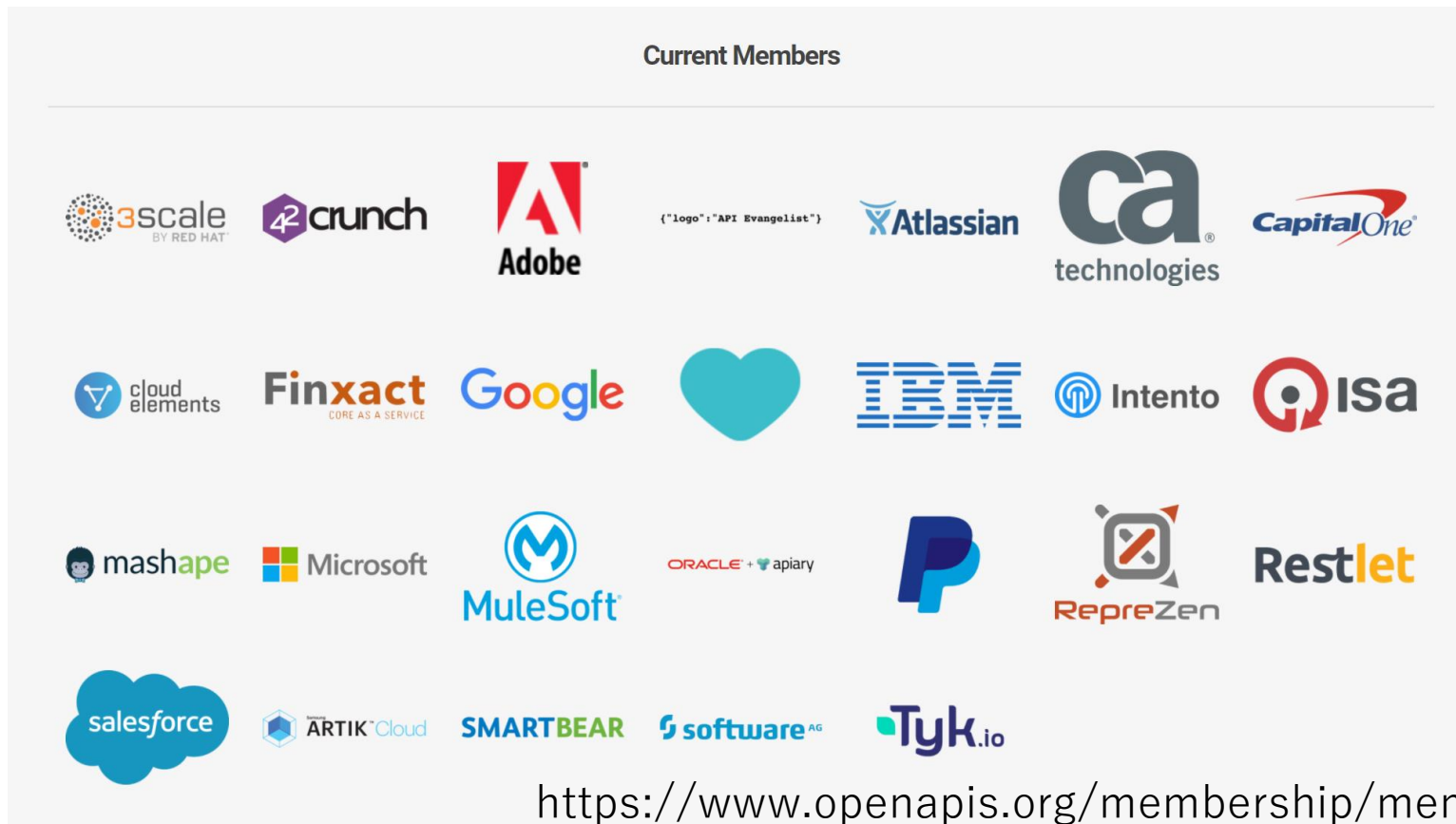
- 非機能要件が実は非常に多い
- 機能要件を満たすスピードをどう出すか
- 標準化は無視できない

Web APIの動向



The Open API Initiative (OAI) was created by a consortium of forward-looking industry experts who recognize the immense value of standardizing on how REST APIs are described.

As an open governance structure under the Linux Foundation, the OAI is focused on creating, evolving and promoting a vendor neutral description format.

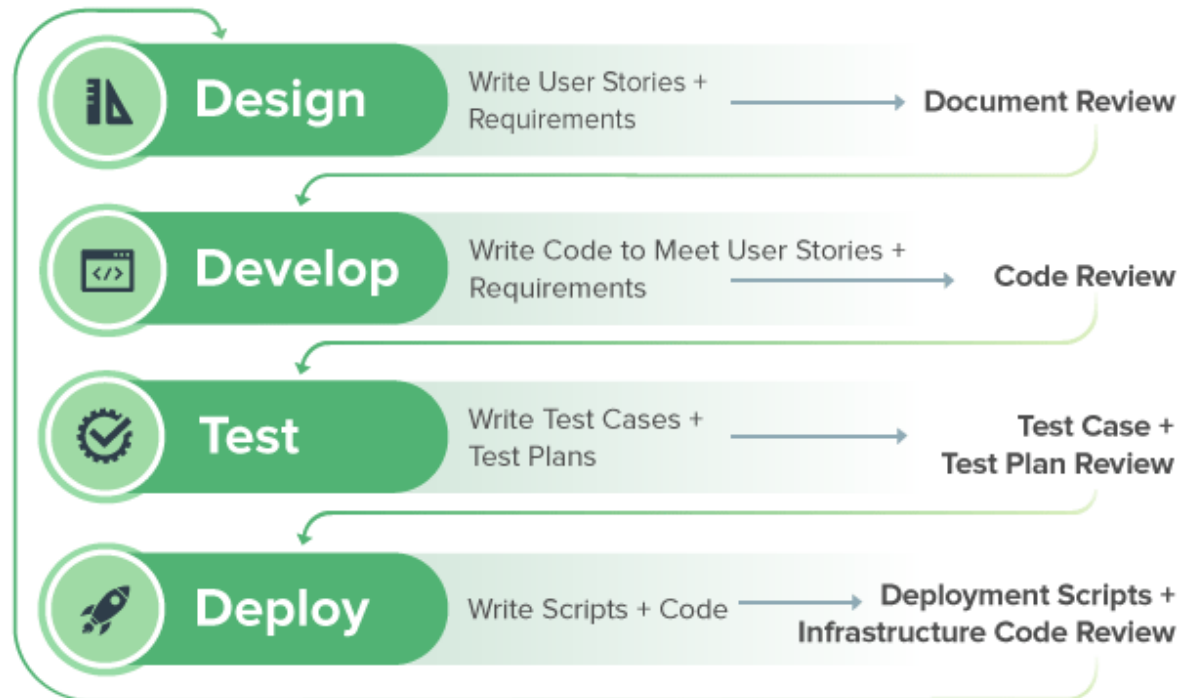


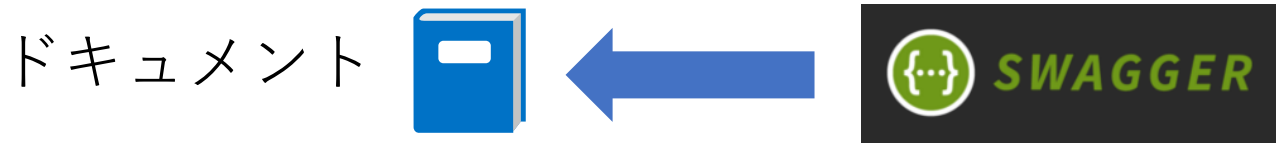


SWAGGER

THE WORLD'S MOST POPULAR API TOOLING

Swagger is the world's largest framework of API developer tools for the OpenAPI Specification(OAS), enabling development across the entire API lifecycle, from design and documentation, to test and deployment.





SDKと
アプリケーション
コード

定義ファイル

サーバーコード

pet Everything about your Pets

POST /pet Add a new pet to the store

Parameters

Name Description

body * required Pet object that needs to be added

(body)

Example ValueModel

```
{
  "id": 0,
  "category": {
    "id": 0,
    "name": "string"
  },
  "name": "doggie",
  "photoUrls": [
    "string"
  ],
  "tags": [
    {
      "id": 0,
      "name": "string"
    }
  ]
}
```

Generate Server ▾

Generate Client ▾

[Switch back to previous editor](#)

aspnet5

aspnetcore

erlang-server

finch

bash

CsharpDotNet2

go-server

haskell

inflector

jaxrs ml

elixir

html

jaxrs-cxf

jaxrs-cxf-cdi

jaxrs-resteasy

jaxrs-re

javascript-closure-angular

jaxrs-spec

lumen

msf4j

nancyfx

perl

nodejs-server

python-flask

rails5

scalatra ml

ruby

swift

silex-PHP

sinatra

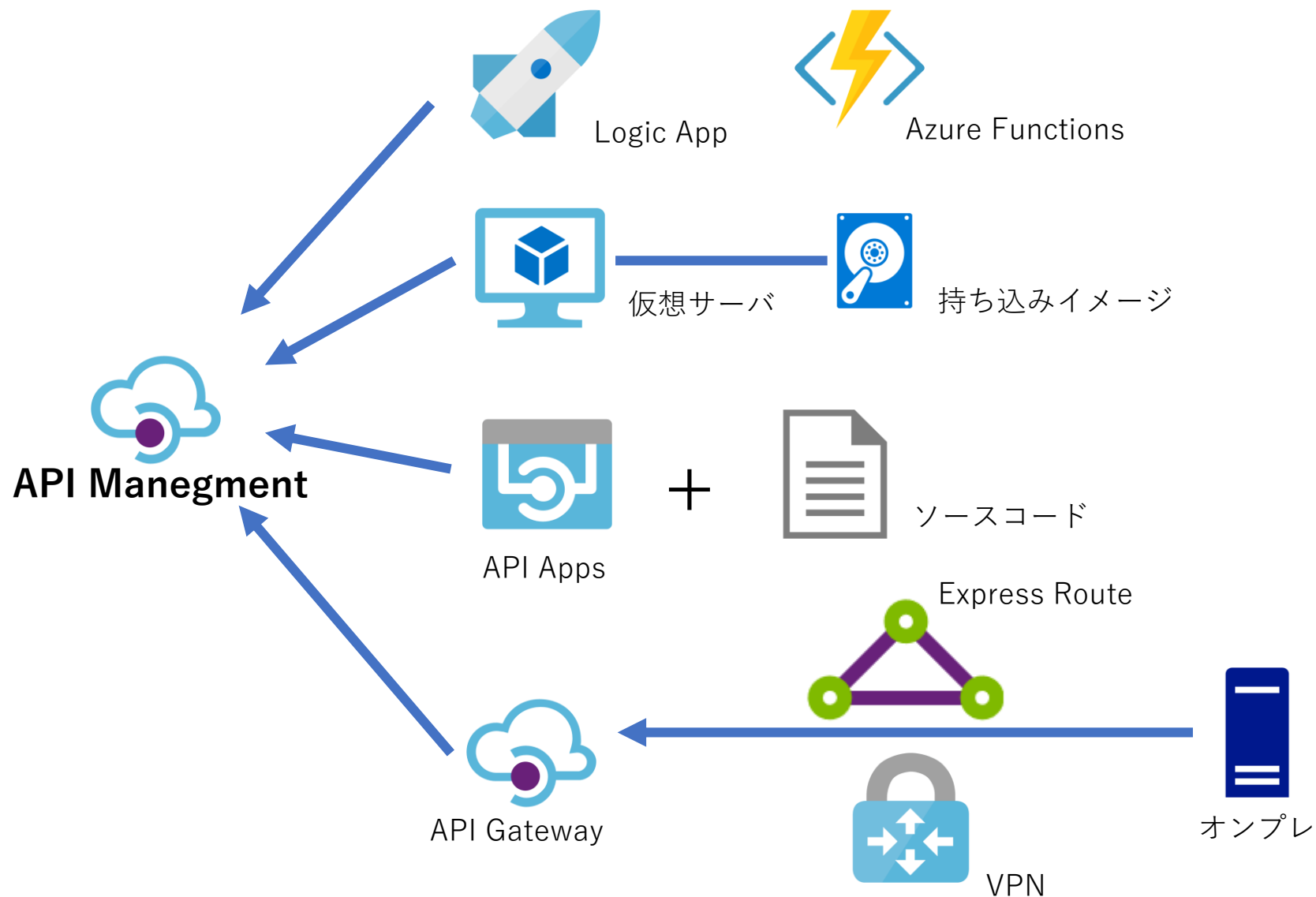
slim

spring ingular

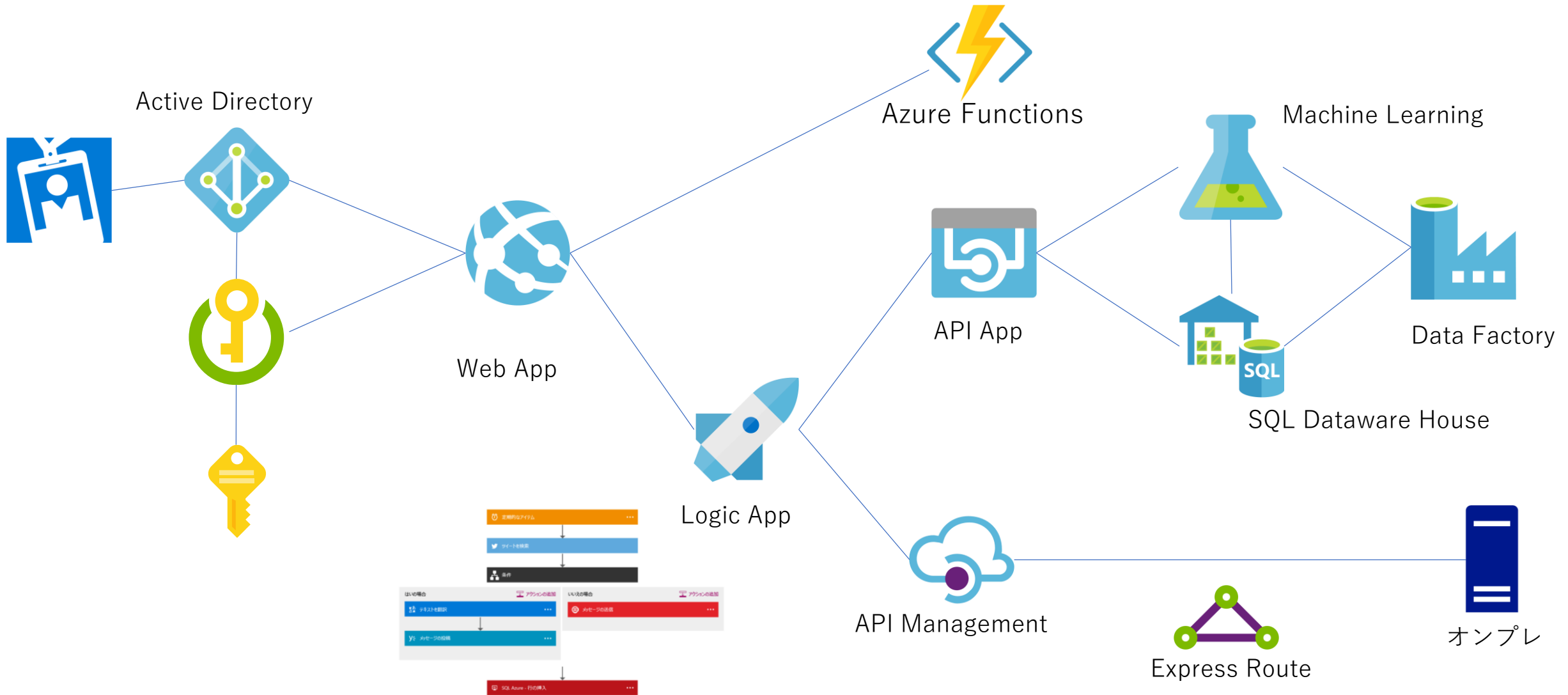
typescript-angular2

node

多彩なバックサイドの実装方法



Web APIベースでのシステム化の複雑化



クラウドらしい考え方を学ぶ

アーキテクチャーのガイドが公開されています

<https://docs.microsoft.com/ja-jp/azure/architecture/>

Azure Architecture Center



Azure Application Architecture Guide

A guide to designing scalable, resilient, and highly available applications, based on proven practices that we have learned from customer engagements.



Reference Architectures

A set of recommended architectures for Azure. Each architecture includes best practices, prescriptive steps, and a deployable solution.



Cloud Design Patterns

Design patterns for developers and solution architects. Each pattern describes a problem, a pattern that addresses the problem, and an example based on Azure.

ベストプラクティス

API design

<https://docs.microsoft.com/en-us/azure/architecture/best-practices/api-design>

API implementation

<https://docs.microsoft.com/en-us/azure/architecture/best-practices/api-implementation>

API Security Checklist

<https://github.com/shieldfy/API-Security-Checklist>

参考：RESTとHATEOAS

<http://postd.cc/sprinkle-some-hateoas-on-your-rails-apis/>