Vol.2019-ARC-235 No.40
Vol.2019-SLDM-187 No.40
Vol.2019-EMB-50 No.40
2019/3/18

# 気が利くシステムのための マイクロサービスアーキテクチャの提案

長田 直也[1]　満田 成紀[2]　福安 直樹[2]　松延 拓生[2]　鯵坂 恒夫[2]

**概要**：気が利くシステムとは，スマートホームのような文脈・状況を把握するための情報（コンテキスト情報）を用いて最適な動作を行うシステムであり，様々な IoT デバイスやサービスを利用することでユーザに適した動作を実現する．このようなシステムにはマイクロサービスアーキテクチャの導入が考えられる．本研究では，ドメイン分析から導出されたアーキテクチャを意識しながらシステムを開発し，アーキテクチャを漸進的に精錬する．その結果，気が利くシステムに必要な 7 つの役割の導出と 3 つのマイクロサービスの識別をすることができた．

## 1. はじめに

近年，ICT の発展によりさまざまなセンシングデバイスやビッグデータを活用することが可能になっている． IoT デバイスは GotAPI[1]，クラウドサービスは OpenAPI 仕様 [2] によって標準化が図られ，利用しやすくなっている．これらの技術を組み合わせることにより，気が利くシステムを開発することができる．気が利くシステムとは，スマートホームのような文脈・状況を把握するための情報（コンテキスト情報）を用いて最適な動作を予測的に行うシステムである．

このような独自に開発されたデバイスやサービスを組み込んだシステムを構築する際には，マイクロサービスアーキテクチャの導入が考えられる．マイクロサービスアーキテクチャは単一のアプリケーションを小さなサービス群として組み合わせて構築するアプローチである．マイクロサービスは，協調して動作する小規模で自律的なサービスである [3].

本研究ではマイクロサービスアーキテクチャに着目し，気が利くシステムのためのアーキテクチャを提案する．気が利くシステムは様々な IoT デバイスやクラウドサービスを利用することでユーザに適した動作を実現する．そのため，マイクロサービスアーキテクチャが適していると考えた．マイクロサービスアーキテクチャの構築のためには実装とモデルがお互いに影響を与えあってモデルを精錬していく重要性が示されている．そこでドメイン分析から導出されたアーキテクチャを意識しながらシステムを開発することで，アーキテクチャを漸進的に精錬していく．

[1] 和歌山大学大学院システム工学研究科
Graduate School of Systems Engineering, Wakayama University
[2] 和歌山大学システム工学部
Faculty of Systems Engineering, Wakayama University

## 2. 気が利くシステムとマイクロサービスアーキテクチャ

### 2.1 気が利くシステム

気が利くシステムとは，スマートホームのような文脈・状況を把握するための情報（コンテキスト情報）を用いて最適な動作を予測的に行うシステムであり，様々な IoT デバイスを利用することでユーザに適した動作を実現する．

#### 2.1.1 様々なデバイス

気が利くシステムには様々なインタラクションデバイスが用いられる．データを取得するためのデバイスはセンシングデバイスと入力デバイスの 2 種類がある．入力デバイスは人間が意識的に入力を行うものであり，マウス，キーボード，マイクなどがある．センシングデバイスは人間が意識的に入力を行わずともデータ取得が可能なものであり，加速度センサや温度センサなどが含まれているデバイスである．人間からセンシングを行う場合はウェアラブルデバイスが多く用いられる．出力のためのデバイスは画面，音，振動などを利用して人間に対して働きかけるものであり，ディスプレイやスピーカなどがある．

#### 2.1.2 状況推定

気が利くシステムには周囲の状況を推定する必要がある．瀬戸口ら [4] は行動状況と場所状況を組み合わせ，ユーザの生活シーン（在宅中，駅構内を移動中など）を表す情報であるユーザ属性を判定する状況推定エンジンを用いた生活支援情報提示のシステムを試作している．木村ら [5] は

ⓒ 2019 Information Processing Society of Japan

ユーザは常に好きな情報を受け取りたいわけではなく，受け取っても良い状況が存在し，ユーザの置かれた状況にあわせて必要な情報を適切なタイミングで配信することが必要であるとしている．

### 2.2 マイクロサービスアーキテクチャ

マイクロサービスアーキテクチャは単一のアプリケーションを小さなサービス群として組み合わせて構築するアプローチである [3].

#### 2.2.1 マイクロサービス

マイクロサービスは，協調して動作する小規模で自律的なサービスである．マイクロサービスの特徴として以下のものがある [6].

- 各マイクロサービスが 1 つのビジネス機能を実装する．
- マイクロサービスは小規模な 1 つの開発者チームで作成および管理できる．
- マイクロサービスは個別のプロセスで実行される．
- 各マイクロサービスがそれぞれ自分自身のデータベースを管理する．
- 個別のコードベースを持つ．しかし，共通のユーティリティライブラリは使用することがある．
- 他のサービスと関係なくデプロイできる．

#### 2.2.2 優れたサービス

より良いマイクロサービスのためには疎結合と高凝集性の 2 つの概念に注目する必要がある [7]．疎結合を実現するためには，連携するサービスに関して必要最低限のことだけしか把握しないようにする．高凝集性を実現するためには，あるサービスを変更しても別のサービスを変更する必要がないようにする．

#### 2.2.3 マイクロサービスアーキテクチャの提案手順

マイクロサービスの設計はドメイン駆動設計 (DDD) [8] のアプローチが推奨されている [9]．ドメイン駆動設計では実装とモデルがお互いに影響を与えあってモデルを精錬していく重要性が示されている．そこで本研究では，ドメイン分析から導出されたアーキテクチャを意識しながらシステムを開発することで，アーキテクチャを漸進的に精錬していく．以下が提案マイクロサービスアーキテクチャを提案する手順である．

(1) ドメインの分析
既存の予測システムを分析し，気が利くシステムに必要な機能や役割についての知見を得る．

(2) アーキテクチャの導出
ドメイン分析から得られた機能や役割からアーキテクチャを導出する．

(3) アーキテクチャの精錬
アーキテクチャを意識したシステム開発を試行し，漸進的にアーキテクチャを精錬する．

(4) マイクロサービスの識別
精錬したアーキテクチャをもとにマイクロサービスの識別を行う．

表 1 分類用チェックリスト
**Table 1** Checklist for classification

| チェック項目 | 利用 | チェック項目 | 利用 |
|---|---|---|---|
| センシングデバイス | | 外界予測（履歴あり） | |
| 入力デバイス | | 行動予測（履歴あり） | |
| 外界予測 | | 予期推定（履歴あり） | |
| 行動予測 | | 外界予測（学習あり） | |
| 予期推定 | | 行動予測（学習あり） | |
| ユーザプロファイル | | 予期推定（学習あり） | |

## 3. ドメイン分析から導出されるアーキテクチャ

本章ではドメイン分析からアーキテクチャを導出する．30 個の既存の予測システムを調査し，その結果から共通する役割を抜き出す．

### 3.1 既存の予測システムの分析

30 個の既存の予測システムを表 1 のチェックリストを用いて分析した．その結果，予測システムは 5 つの分類に分けることができた．

- ユーザの状況を重視するシステム
- 外界の状況を重視するシステム
- リアルタイム性が高いシステム
- 意識的な入力により嗜好にあった動作をするシステム
- ユーザと周囲の状況の重み付けが臨機応変に変化するシステム

気が利くシステムを構築するためには，これらの特徴を持つシステムを設計できるようにしなければならない．そこで，これらを包括的に扱えるようなアーキテクチャを考える．予測システムの分類から，次の要素を持つものを気が利くシステムとする．

- 予測を行う
- コンテキストに応じ，振る舞いを変更する
- ユーザプロファイルを持つ
- ユーザのフィードバックから学習を行い，振る舞いを最適化していく

### 3.2 アーキテクチャの導出

3.1 節で行った分析より図 1 のアーキテクチャを導出した．このアーキテクチャは大きく 3 つの役割に分けることができる．

**データ構築** デバイスのばらつきに対応する役割がある．多様なデバイスをうまく組み合わせるためには，データの取得・解釈をいかに効率よく機能させるのかを考

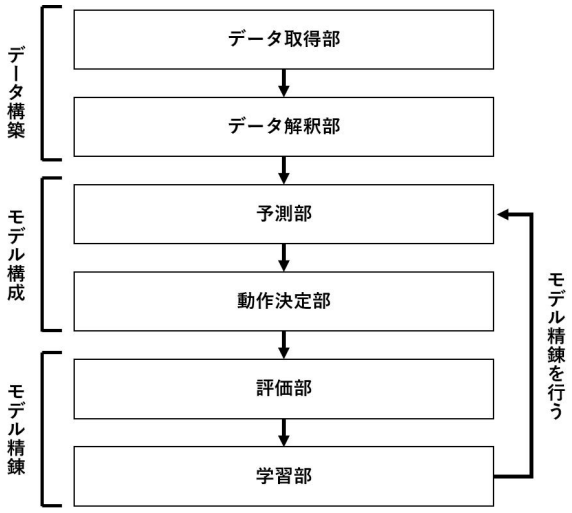


図 1　導出されたアーキテクチャ
**Fig. 1**　Derived architecture

える必要がある．

**モデル構成**　予測や動作決定を行うためのモデルを構成する役割がある．

**モデル精練**　予測モデルを精練するためのデータを作成する役割がある．

このアーキテクチャはさらに 6 つの部分に分けられる．

**データ取得部**　外界から取得した生データを扱う部分である．

**データ解釈部**　データ取得部から送られてきた生データを意味のある加工データに解釈する部分である．

**予測部**　予測に関するモデルを扱う部分である．

**動作決定部**　予測部の結果を受け，最適な動作を決定する部分である．

**評価部**　動作した結果がユーザにとってふさわしかったかどうかを評価する部分である．

**学習部**　評価データを受け取り，予測モデルの精練を行う部分である．

## 4. 導出アーキテクチャを意識したシステム開発の試行

本章では導出アーキテクチャを意識したシステム開発を試行する．開発の目的は漸進的にアーキテクチャを精錬し，マイクロサービスアーキテクチャにより適した構成を導くためである．

### 4.1 システム開発の試行によるアーキテクチャの変遷

システム開発の試行によって生じたアーキテクチャの変遷を示す．

(1) ユーザとデバイスの関係の追加
導出されたアーキテクチャにはユーザとインタラクションデバイスの関係が含まれていなかったので，導出されたアーキテクチャとユーザ，インタラクションデバイスの関わりを追加した．

(2) プロセスの管理をする部分の追加
ユーザがアプリケーションにイベントを送る部分をひとまとめにし，イベントの振り分けとプロセス管理を行う機構を追加した．その結果，気が利くシステムには予測プロセスと改善プロセスの 2 種類のプロセスがあることが分かった．

(3) 評価部の分割
評価部を分割し，より適切な名称に変更した．学習のためのデータを作成する評価部は，予測用データを作成するデータ解釈部の一種であると考え，予測用データ作成部と学習用データ作成部とした．予測プロセスにおけるデータ解釈部が予測用データ作成部，改善プロセスにおけるデータ解釈部が学習用データ作成部である．

(4) 動作表現部の分離
動作を行う部分と動作決定を行う部分に分けることでデバイスやサービスの変更を容易に行えるようにした．動作を行う部分を動作表現部とする．

### 4.2 最終的な開発システムの概要

開発システムは周囲の状況からユーザの気分に合わせた色を予測し，Hue ランプ[*1]を適切な色に点灯させるシステムである．

## 5. マイクロサービスアーキテクチャの提案

本章では 4 章で精錬したアーキテクチャをもとに，各役割をマイクロサービスに識別し，気が利くシステムのためのマイクロサービスアーキテクチャを提案する．

### 5.1 提案アーキテクチャの各部分の役割

提案アーキテクチャにおける，気が利くシステムに必要な役割を持つ部分は次の 7 つである．

(1) データ取得部
(2) 予測用データ作成部
(3) 予測部
(4) 動作決定部
(5) 動作表現部
(6) 学習用データ作成部
(7) 学習部

#### 5.1.1 データ取得部

センサデータや外部リソースから生データを取得する部分である．この部分でデバイスやサービスの違いを吸収する．データ取得はデバイス，外部リソース，内部リソースなどから行う．データ取得のためのデバイスはセンシング

[*1] Philips Hue：スマートフォンでコントロールできるスマート LED 照明．https://www2.meethue.com/ja-jp

デバイスと入力デバイスの 2 種類とする．外部リソースはクラウドサービスなど，開発システム外から取得するリソースである．内部リソースはシステムが有しているリソースである．PC 内の時間情報などがこれにあたる．

#### 5.1.2 予測用データ作成部

センシングまたは入力された生データを意味のある加工データにし，予測のためのデータを作成する部分である．加工の例として，画像データを挙げる．画像データの中には様々な利用可能なデータがある．画像内の人の動きを解析し，行動履歴とすることができる．また，画像内の車の動きを解析することで渋滞の度合いを判断することも可能である．このように生データを加工データに解釈すると意味合いが大きく変化するものがある．気温データや湿度データなど，解釈の変更がないデータも，解釈の変更がないと解釈すれば加工データであるとする．また，予測アルゴリズムによってはダミー変数に変換するといった処理も必要である．

#### 5.1.3 予測部

予測に関するモデルを扱う部分である．予測に利用するデータは予測用データ作成部から取得してきたものである．予測モデルの作成を行うために必要なユーザのデータは，ユーザ毎にユーザプロファイルとしてデータベースに保存されている．この部分で予測されるものとしては，ユーザが関係しない事象の予測や，ユーザの行動予測などがある．得られた結果は動作決定部に渡される．この部分ではユーザに対してどのような振る舞いをするかまでは決定しない．次に述べる動作決定部で振る舞いは決定される．

#### 5.1.4 動作決定部

予測部の結果を受け，最適な動作を決定する部分である．動作決定部が受け取る予測結果は一つとは限らない．複数の予測部から送られてきたデータを利用して動作の決定を行うことも考えられる．開発システムの動作決定部は開発者が振る舞いを決定しているが，この部分にも機械学習の機構を導入することが可能である．その場合，ユーザからのフィードバックが動作決定の結果に対するものであるのか，予測結果に対するものなのか区別する仕組みが必要である．

#### 5.1.5 動作表現部

動作決定部から受け取った動作を実行するためのデータ（デバイス名，属性名など）をもとに動作の実行を行う部分である．この部分により，システムは状況に応じて臨機応変に出力するデバイスを取り換えることができる．動作決定部はいくつもの動作表現部を持つことができ，動作決定部は統一的な方法で動作表現部にデータを送ることができることが望ましい．

#### 5.1.6 学習用データ作成部

ここではユーザからのフィードバックデータを受け，よりよい学習のためのデータを構築する，学習用データ作成を行う部分である．ユーザからのフィードバックはデータ取得部を通して受け取る．学習用データの内容はモデルに依存する．ベイジアンネットワークであれば，確率データの操作などが考えられる．

#### 5.1.7 学習部

学習用データ作成部で作成されたデータをもとに予測モデルを精錬する部分である．ここで精錬するモデルは予測部で使用される予測モデルのことであり，予測アルゴリズムが変更されれば，学習部も変更しなければならない．

### 5.2 アーキテクチャに含まれるマイクロサービスの識別

疎結合と高凝集性を意識しながら，アーキテクチャに含まれるマイクロサービスを識別を行った．その結果，3 つのマイクロサービスに分けることができた．

(1) データ取得サービス
(2) 予測サービス
(3) 動作サービス

また，気が利くシステムには予測プロセスと改善プロセスの 2 つのプロセスがある．これらプロセスは送られてくるイベントによって振り分けられる．この 2 つのプロセスやアプリケーション全体の処理の流れを管理する部分も必要である．

#### 5.2.1 データ取得サービス

データ取得を行うサービスである．データ取得のみを行い，データに意味付けをする機能はない．また，この部分は実装するビジネスロジックに依存しない．データ取得サービスはデータ取得部のみで構成している．予測サービスがどのようなデータを必要とするかによって選択されるデータ取得サービスが変わるため，このサービスを実装する際に，予測サービスを意識する必要はない．

#### 5.2.2 予測サービス

予測に関する処理を行うサービスである．予測用データ作成部・予測部・学習用データ作成部・学習部を含んでいる．このサービスは動作サービスに対し予測結果を渡すだけであり，どのような動作が実行されるかについては把握しない．予測サービスは自分が呼び出されたとき，予測プロセスであるのか改善プロセスであるのかを判断する必要がある．図 2 で予測サービスでの処理の流れとデータベースとの関係を示す．破線の矢印は応答を表す．予測サービスはプロセスの種類によって振る舞いが異なる．評価プロセスの場合，データ取得サービスから送られてきたデータをもとに予測用データ作成部で予測用データの作成を行う．そして作成した予測用データを予測部に送信し，予測モデルをもとに予測を行う．予測から得られた結果は動作サービスに送られる．改善プロセスの場合，データ取得サービスから送られてきたデータをもとに学習用データ作成部で学習用データの作成を行う．そして作成した学習用データをデータベースに保存する．学習部はデータベースに保存

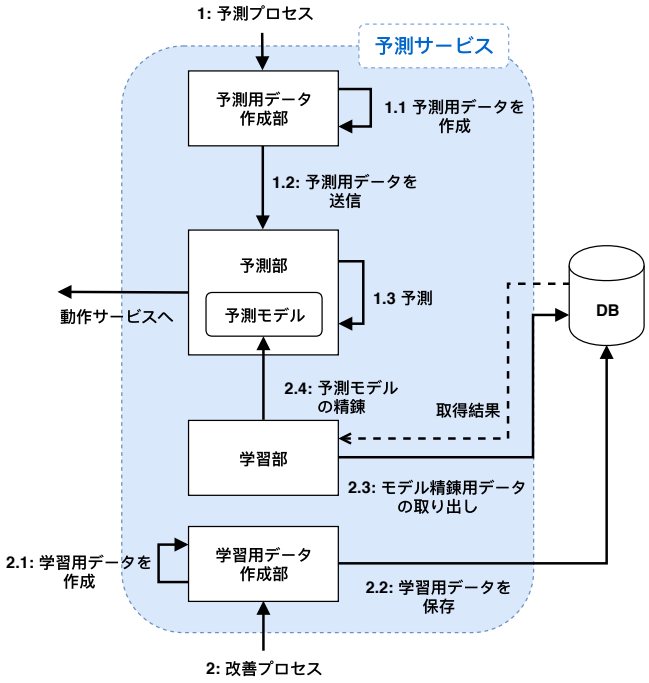


図 2　予測サービスの処理の流れ
**Fig. 2**　Flow of prediction service

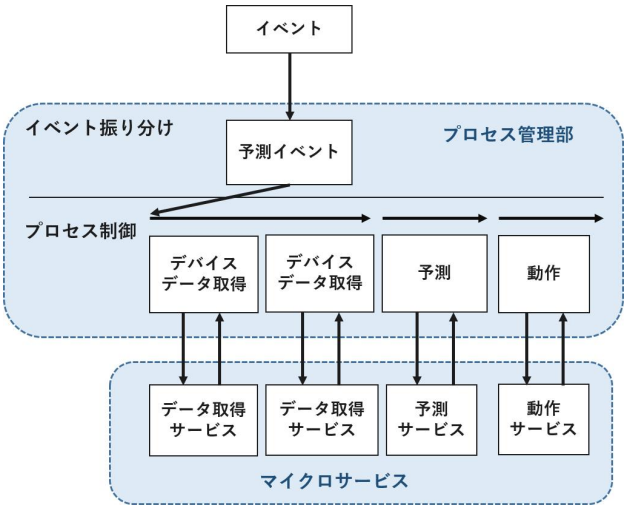


図 3　予測プロセス
**Fig. 3**　Prediction process

されたモデル精錬用データを用いて予測モデルの精錬を行う．

#### 5.2.3　動作サービス

動作に関する処理を行うサービスである．動作決定部・動作表現部を含んでいる．このサービスは予測サービスから送られてきた予測結果をもとに動作を行う．このサービスは予測サービスがどのような予測アルゴリズムを用いているかを把握する必要はなく，予測結果として何が渡されるかを知っておくだけでよい．

### 5.3　プロセス管理部

イベントに応じてプロセスを振り分ける部分であり，イベント駆動型の実装となる．プロセス管理部にはイベント振り分けとプロセス制御を行う 2 つの役割がある．イベント振り分けを行う部分では，デバイスやサービスから送られてきたデータが予測のためのイベントなのか，改善のためのイベントなのかを判断し，各プロセスに振り分ける役割がある．イベントは人間の意識的な入力によって発生するものとセンシングデバイスや外部サービスがユーザの意思に関係なく発生させるものがある．プロセス制御を行う部分は使用するサービス間の処理の順番や，タイミングを制御する役割がある．プロセス管理部は各マイクロサービスを管理するため，マイクロサービスの領域の外に配置する．

#### 5.3.1　各プロセスと各サービス

プロセス管理を行う際に，各プロセスが各サービスとどのように関係しているかを述べる．図 3，図 4 中のプロセス制御部分に記載されている右向きの矢印は，一つの処理の区切りを表しており，この処理が終了したら，次の処理に移る．

#### 5.3.2　予測プロセス

データ取得サービスと予測サービス，動作サービスが含まれる（図 3）．このプロセスは送られてきたイベントが予測イベントであるときに実行される．データ取得サービスは並行して複数の処理を行うことができる．予測サービスと動作サービスは前の処理が終わってから実行される．実行されるサービスはレスポンスを返すようにする．データ取得サービスと予測サービスは処理したデータをプロセス管理部に返す必要が必要であるのでレスポンスが必要である．動作サービスは処理したデータを返す必要はないが，処理が成功したかどうか確認したり，エラー処理を行う場合にレスポンスが必要である．

#### 5.3.3　改善プロセス

データ取得サービスと予測サービスが含まれる（図 4）．このプロセスは送られてきたイベントが改善イベントであるとき実行される．構造は予測プロセスとほぼ同じである．動作を行う必要がないので動作サービスに関する処理がない．予測サービスにデータを送るが，起動される処理は学習である．

### 5.4　提案するマイクロサービスアーキテクチャ

これまでのアーキテクチャの変更やデータベースとの関係，外部サービスとの関係を考慮し，最終的に提案する気が利くシステムのためのマイクロサービスアーキテクチャが図 5 である．クライアント側にはユーザが干渉することでイベントが発生するデバイスとユーザに関係のないイベントを発生させるデバイスを配置することができる．これは 5.3 で述べた通り，イベントにはユーザの意識的な入力

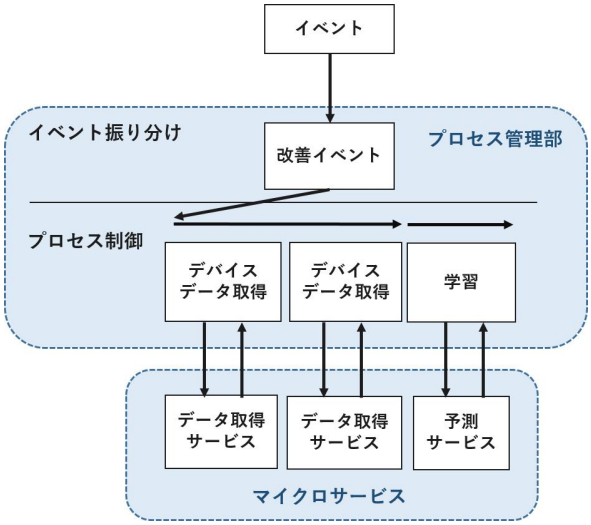


図 4　改善プロセス
**Fig. 4**　Improvement process

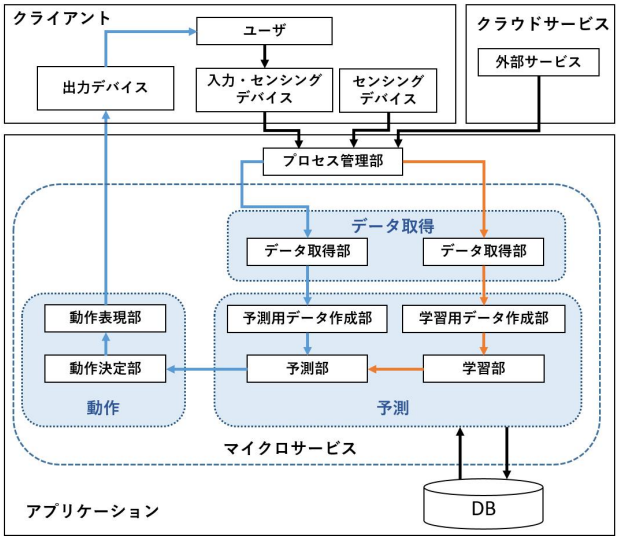


図 5　提案するマイクロサービスアーキテクチャ
**Fig. 5**　Proposed microservice architecture

によって起こるものとユーザの意思に関係なく起こるものがあり，それらに対応するためである．クラウドサービスも同様に利用できるようにしている．データベースは予測サービスで用いるために配置する．

## 6.　考察

疎結合，高凝集性の 2 つの観点とモノリシックサービスとの比較から提案マイクロサービスの有用性を考察する．

提案した各マイクロサービスはお互いの実装の中身を知らずとも，受け渡しされるデータを明確に定義していれば機能するものとなっている．また，各マイクロサービスはほかのマイクロサービスの変更に影響されることなく，自分の役割を実行することができる．したがって，優れたマイクロサービスに必要な疎結合，高凝集性があると考えられる．

モノリシックサービスで気が利くシステムを構築する場合，変更があった時に関連する部分を探しだし，システム全体を配置しなおす必要があり手間がかかる．しかし，マイクロサービスを利用することにより，変更のあるサービスを変更するだけ良いので変化に対応しやすいと考える．

## 7.　おわりに

本研究では多様なデバイスを用いた気が利くシステムのマイクロサービスアーキテクチャを提案した．そして気が利くシステムに必要な 7 つの役割の導出と 3 つのマイクロサービスの識別を行った．しかし，これらのサービスを独立したものとして扱うことでデメリットも考えられる．各サービスはほかのサービスから送られてくるデータを利用するので，正確なタイミングの管理を行わなければならない．もしデータが取得できなかった場合や予測ができなかった場合，ほかサービスが代用して必要なデータを送る，今ある情報の中で最適な動作を決定する，といった機能を有しておく必要があると考える．

### 参考文献

[1] OMA Specworks，“GotAPI”，入手先 ⟨https://www.omaspecworks.org/what-is-oma-specworks/iot/gotapi/⟩（参照 2019-2-8）．
[2] OpenAPI Initiative，“The OpenAPI Specification”，入手先 ⟨https://github.com/OAI/OpenAPI-Specification⟩（参照 2019-2-8）．
[3] James Lewis，Martin Fowler，“Microservices”，入手先 ⟨https://martinfowler.com/articles/microservices.html⟩，（参照 2019-1-11）．
[4] 瀬戸口 久雄，岡本 雄三，長 健太，川村 隆浩，“ユーザ属性に応じた生活支援情報提示を行うエージェントシステム”，情報処理学会シンポジウム論文集，第 2011 巻，第 3 号，pp.409-412，2011．
[5] 木村 壮，“個人の TPO を考慮に入れた情報配信システムの開発”，大学院研究年報 理工学研究科篇，第 36 号，2006．
[6] Microsoft，“Azure でのマイクロサービスの設計、構築、および操作”，入手先 ⟨https://docs.microsoft.com/ja-jp/azure/architecture/microservices/index⟩（参照 2019-2-1）．
[7] Sam Newman，佐藤 直生（監訳），木下 哲也（訳），“マイクロサービスアーキテクチャ”，オライリー・ジャパン，2016．
[8] Eric Evans，今関 剛（監訳），和智 右桂・牧野 祐子（訳），“エリック・エヴァンスのドメイン駆動設計”，翔泳社，2011．
[9] Microsoft，“マイクロサービスの設計:ドメイン分析”，入手先 ⟨https://docs.microsoft.com/ja-jp/azure/architecture/microservices/domain-analysis⟩（参照 2019-2-1）．