特別研究報告書

IoT 環境における状況依存型サービス連携の実現

指導教員　　石田 亨 教授

京都大学工学部情報学科

渡辺 隆弘

平成 28 年 2 月 6 日

IoT 環境における状況依存型サービス連携の実現

渡辺 隆弘

内容梗概

アブストラクト研究の背景と、概要

研究の貢献

1. Web とセンサーを繋ぐ画一化されたプラットフォームが存在しない
2. Web サービスの利用にその都度リクエストを送信しなければならない
3. サービス選択が手動

Realization of situated service composition in IoT environment

Takahiro Watanabe

Abstract

abstract

IoT 環境における状況依存型サービス連携の実現

目次

第1章　はじめに

はじめに

第2章　関連研究

この章では本研究に利用している各用語についての説明を行う．

2.1　IoT

IoT(Internet of Things) とは，様々な物理機器，建物，乗り物などにセンサーやソフトウェアを組み込むことで，情報交換やデータの収集を行えるネットワークを構築する仕組みである．以下のような例が考えられる．

- 離れた場所の環境を知る温度，湿度，気圧．照度といった環境をセンサーによって知ることができる．
- 物体の動きを知る物体の動き (衝撃，振動，移動など) を知ることができる．
- 物体の位置を知る物体の位置 (存在，通過など) を知ることができる．
- 機器の制御を行う空調の制御，照明の制御などを離れた場所から操作することができる．

このように，IoT 環境下ではセンサーデータを用いた周囲の把握や，電気機器の利用，モニタリングが可能となり，より安全かつ快適な生活を実現できるようになる．

ここで，既存の IoT プラットフォームとして，OpenIoT を取り上げる．OpenIoT はオープンソースで実装されている IoT プラットフォームである．OpenIoT ではセンサーから取得したデータをミドルウェアを通じてデータベースに格納している．

2.2　IoS

IoS(Internet of Service) とは，Web アプリケーションやサービスを組み合わせ，新たなサービスを構成するものである．Web 上に点在するサービスを組み合わせ，より複雑な処理やサービス提供が可能となる．IoS 基盤の例として言語グリッドが挙げられる．言語グリッドは，辞書や機械翻訳などの言語資源が言語サービスとして登録され，共有可能とされているインターネット上の多言語サービス基盤である．多数の言語の相互翻訳，用例対訳，言語判別，音声認

識，音声合成などのライブラリを，WebAPI から利用できる．

2.3 CEP

CEP(Complex Event Processing，または複合イベント処理) とは，刻々と生成されるデータをリアルタイムに処理するための方式である．事前に定義したルールに，リアルタイムにデータを挿入し，そのルールに応じて即座に処理を行う．これまでのビッグデータ分析の方法は，データをデータベースに蓄積し，任意のタイミングで参照し，分析するという手法であったために，情報の処理に時間がかかるという問題点があった．CEP は対象のデータを直近の範囲に絞り，メモリ上に読みこんで処理を行うため処理を高速化でき，”直近の数秒以内に ”などの条件に沿ってデータを処理することが可能となる．本研究では，この CEP をストリーム形式であるセンサーデータに対し応用することを考える．

2.4 サービス連携

サービス連携とは，IoS 基盤に集積された各原子サービスを組み合わせ，ユーザの要求を満たす高い品質 (QoS，または Quality of Service) の複合サービス (Composite Service) を構成する技術である．従来，複合サービスを構成するためには，ユーザが自らの要求を満足するような原子サービスを選択する方法が取られていた．また，複合サービスの自動構築を行う方法として，人工知能のプランニング技術を用いてワークフローを自動生成する研究が主流であった．しかし，IoS 環境においては，同種の原子サービスが複数登録されるために，ワークフローを生成することよりむしろ，ワークフローに当てはめる原子サービスの選択が自動化できる必要がある．

第 3 章　提案手法

本章では，現状の課題を説明した後に，センサーのサービス化を行うための手法と，センサーから取得したデータによって，複合サービスのサービス選択，サービス実行を自動で行うための手法を提案する．

3.1 課題点

状況に応じたサービス選択を行うために，センサーから取得した情報によって複合サービスへの入力を変更することを考える．その際に以下の課題点が生じる．

1. センサーの仕様の不統一性

   現状は同じ種類のセンサー (温度センサーや湿度センサーなど) でも，通信手段やデータフォーマットなどに差異がある．Web サービスを利用する際には，その場所に存在するセンサーから値を取得しサービスを実行するが，そのセンサーの仕様が統一されていなければ，それぞれのセンサーの仕様ごとにシステムの実装を行う必要が生まれる．

2. 複合サービス内の原子サービスの選択

   これまで，複合サービス内の原子サービスの選択は，ユーザによって指定する方向で行われてきた．例えば，言語グリッドの翻訳サービスのうち，辞書翻訳を利用することを考える．言語グリッドの辞書翻訳には様々なサービスが登録されており，ユーザがどの辞書を用いるか指定する．

   つまり，原子サービスの選択にユーザの知識や経験が要求されるため，以下のような問題点が生じる．

   - ユーザが初めて複合サービスを利用する際にどのような原子サービスを利用すれば適当かが分からない
   - ユーザのサービスに対しての知識が不足しているために，ユーザのサービス選択がユーザの要求に関わらず固定化されてしまい，ユーザの要求を満たすよりよい原子サービスの組み合わせがあるにもかかわらず，より質の低いサービス選択を行ってしまう

3. 複合サービスのリアルタイム実行

   複合サービスは，複数の Web サービスを組み合わせたものであるため，実行の仕様は Web サービスに基づく．Web サービスはリクエストに応じてレスポンスを返す形式であるため，Web サービスを利用するためには，ユーザは Web サービスにリクエストを送信する必要がある．

これらの課題点を解決するために，以下の 3 つの手法を提案する．

3.2 センサーのサービス化手法

本節では，センサーのサービス化手法を提案する．現状は，前述した通りセンサーの仕様が画一化されていないために，センサーを利用するシステムを実装する際，センサーの種類によって異なる実装が必要であるという問題点が存在する．この問題点を本提案は解決する．

データ定義を OpenIoT のセンサー定義に基づいて画一化する．OpenIoT のセンサー定義の例は以下であり，Observation オブジェクトとして実装される．データの値，取得時間や，温度，湿度，照度といったデータタイプを示す propertyType などが存在する．

センサー定義例

```
1  //Observation
2
3    private String id;
4    private Date times;
5    private String sensorId;
6    private String featureOfInterest="";
7    private ArrayList<ObservedProperty> readings;
8    private String metaGraph;
9    private String dataGraph;
10
11
12 //ObservedProperty
13
14   private static final long serialVersionUID = 1L;
15   private Object value;
16   private Date times;
17   private String propertyType;
18   private String unit;
19   private String observationId;
```

センサーの開発者は，センサーから値を取得した際に，Observation を作成し，各変数に取得した値を格納するようにサービスを構成する．システム開発者はこのサービスの仕様に従ってシステムを実装することで，ユーザからは種々のセンサー間の違いは隠蔽され，画一化されたセンサーサービスとしてデータを

利用することができる．例えば，センサーから温度 20 ℃，湿度 50%のデータを取得した際には以下のように Observation を生成する．

Observation 生成例

```
1 Observation o = new Observation();      //
    Observationオブジェクトの作成
2 ArrayList<ObservedProperty> readings = new ArrayList<
    ObservedProperty>();      //ObservedPropertyのリストの作成
3 ObservedProperty tempProperty = new ObservedProperty();      //
    ObservedPropertyオブジェクトの作成
4 ObservedProperty humdProperty = new ObservedProperty();
5 tempProperty.setPropertyType("http://openiot.eu/ontology/ns/
    AirTemperature");      //propertyTypeの設定
6 humdProperty.setPropertyType("http://openiot.eu/ontology/ns/
    AtmosphereHumidity");
7 tempProperty.setValue(20);      //valueに値を格納
8 humdProperty.setValue(50);
9 readings.add(tempProperty);      //
    ObservedPropertyのリストに追加
10 readings.add(humdProperty);
11 o.setReadings(readings);;      //
    Observationに作成したリストを格納
```

3.3 状況依存型サービス選択手法

本節では，センサーの値によって複合サービス中の原子サービスを選択する手法を提案する．センサーから取得した値をイベントエンジンによって処理することによってこの手法は実現される．詳細を以下に述べる．

3.3.1 データの取得

前節に基づいて作成されたセンサーサービスオブジェクトがサーバーへ送信される．サーバーはデータを受け取った時点で，このオブジェクトを CEP エンジンに挿入する．

3.3.2 原子サービスの選択

原子サービスの選択においては，ECA ルールを応用することを考える．ECA ルールとは，～～するものであり，

E ：Event

C ：Condition

A ：Action

の 3 つの状態が定義される．イベントが発生した際，その状況に応じてアクションを実行する，というルールの実行を行う．

本研究では，ECA ルールを CEP エンジンで実現する．つまり，ECA ルールを以下のように適用する．

E ：センサーからのデータの取得

C ：センサーから取得した値

A ：選択するサービスとサービスへの入力の生成，サービスの実行

以上から，センサーからデータを取得した際，サーバーから CEP エンジンにデータを挿入し，事前に定義されたルールに基づいて，選択するサービスとサービスへの入力の生成とサービスの実行を行うという一連の処理が実行される．また，本研究では CEP エンジンにおいて適用するルールは事前に定義されているものとし，状況に応じてどのような処理を実行すべきかというルールの構成の点についての議論は行わない．

この手法により以下の 2 点の問題点が解決される．

1. 複合サービス内の原子サービスの選択
   ユーザがサービス選択を行わなければならないという問題点が存在した．一方，本提案では，専門家が一度ルールを作成すれば，センサーの値によって分岐するルールに従って原子サービスの選択を行うことができ，サービス連携においてユーザのサービスに対しての知識や経験に関わらず一定の質の高いサービス合成が可能となる．
2. 複合サービスのリアルタイム実行
   サービス実行のためにユーザは Web サービスにリクエストを送信する必要があった．一方，本提案では，センサーの値をイベントとして CEP エンジンに挿入し，リアルタイムで処理，アクションとして複合サービスへの入力生成とサービス実行を行うことによって，ユーザがサービスのリクエストを送信することなく，リアルタイムかつ自動的なサービス実行が可能となる．

第4章　提案アーキテクチャ

本章では，前章に説明した提案手法に基づいて，IoT 環境下で複合サービスの選択，実行を行うアーキテクチャの提案を行う．

4.1

第5章　実装

本章では，前章に提案したアーキテクチャの実装について説明し，動作確認と評価について述べる．最後に実装の結果に対して考察を行う．

5.1　シチュエーション

体育館を利用するユーザに，温度，湿度などの情報から運動への助言を音声で与えるシステムを実装することを考える．ユーザは様々な言語圏のユーザが想定されるため，それぞれのユーザが利用する言語に基づいてアナウンスを行う必要がある．

5.2　仕様

言語は Java を用いて実装した．以下に各モジュールの詳細を述べる．

5.2.1　センサーデバイス

体育館に設置することを想定するセンサーデバイスは，(株) オムロンの環境センサー”URL”とする．このセンサーによって取得できるデータタイプの中から，今回は温度データと湿度データを利用する．

5.2.2　ユーザデバイス

ユーザが所持している端末を，iOS を搭載した端末とした．この端末はセンサーデバイスからデータを取得し， Observation を形成してサーバーへ送信するデバイスとして働く．センサーデバイスと本デバイス間の通信はBLE(Bluetooth Low Energy) を使用する．BLE は省電力の無線通信技術であり.....
構成要素は以下.

- WaikikiSensor
  取得したデータを端末上に表示する.
- Myviewcontrollor

データを取得し，センサーデータオブジェクトを構成してサーバに送信する．オブジェクトの構成法は 3.1 節で説明した方法に基づく．

1. Observation オブジェクトを生成する．
2. ObservedProperty として tempProperty，humdProperty を作成する．それぞれ，温度のデータ，湿度のデータを格納するオブジェクトである
3. ObservedProperty それぞれに，データタイプを示す PropertyType とデータの値を格納する．
4. tempProperty と humdProperty を Observation オブジェクトに格納する．

5.2.3 サーバー

サーバーと周辺のモジュールについて説明する．
以下のモジュールからなる．

- サーバー
  サーバーは ObservationReceiver クラスとして実装される．デバイスから Observation オブジェクトが送信された際に，CEP エンジンにでデータをイベントとして挿入する．
- CEP エンジン
  CEP エンジンとして，Java で実装されたイベントエンジンである Drools を利用する．ルールとして”badminton.drl”を実装した．ルールの概要を以下に挙げる．

表 1: badminton.drl

ルール	条件	出力
牛丼	並盛	500 円
牛丼	大盛	1,000 円
牛丼	特盛	1,500 円

- 複合サービス
  複合サービスとして，言語グリッドを利用する．言語グリッドは，登録された言語サービスを自由に組み合わせて新しい言語サービスを生み出す複

合サービスである．本研究では，言語グリッドに登録されているサービスの中から，形態素解析サービスと辞書翻訳サービス，音声合成サービスを利用する．仕様は ”URL ”

- 形態素解析
- 辞書翻訳
- 音声合成

5.3　動作確認

5.4　考察

第6章　終わりに

謝辞

本研究を行うにあたり, 貴重な資料をご提供いただきました株式会社オムロン様に深く感謝申し上げます. そして本研究を行うにあたり, 熱心なご指導, ご助言を賜りました石田亨教授に厚く御礼申し上げます. また, 日頃より数々のご助言をいただきました中口孝雄特定研究員, 林冬惠助教をはじめ, 石田・松原研究室の皆様方に心より感謝いたします.

付録

実装のソースコードを添付する.

A.1 デバイスでセンサーデータを取得し、サーバーへ送信するモジュールのソースコード

A.1.1 MyviewController.java

```
1  package org.langrid.waikiki.sensor;
2
3  import java.net.MalformedURLException;
4  import java.net.URL;
5  import java.util.ArrayList;
6
7  import org.langrid.waikiki.sensor.omron.EnvSensor;
8  import org.langrid.waikiki.sensor.omron.EnvSensorScanner;
9  import org.langrid.waikikiws.service.ObservationReceiverImpl;
10 import org.langrid.waikikiws.service.api.ObservationReceiver;
11 import org.openiot.lsm.beans.Observation;
12 import org.openiot.lsm.beans.ObservedProperty;
13 import org.robovm.apple.coregraphics.CGRect;
14 import org.robovm.apple.foundation.NSBundle;
15 import org.robovm.apple.foundation.NSURL;
16 import org.robovm.apple.uikit.UIColor;
17 import org.robovm.apple.uikit.UIView;
18 import org.robovm.apple.uikit.UIViewController;
19 import org.robovm.apple.webkit.WKScriptMessage;
20 import org.robovm.apple.webkit.WKScriptMessageHandlerAdapter;
21 import org.robovm.apple.webkit.WKUserContentController;
22 import org.robovm.apple.webkit.WKWebView;
23 import org.robovm.apple.webkit.WKWebViewConfiguration;
24
25
26 import jp.go.nict.langrid.client.jsonrpc.JsonRpcClientFactory
       ;
27 import net.arnx.jsonic.JSON;
28
```

```
29 public class MyViewController extends UIViewController {
30   public MyViewController() {
31     // Get the view of this view controller.
32     UIView view = getView();
33
34     // Setup background.
35     view.setBackgroundColor(UIColor.white());
36
37     WKUserContentController controller = new
           WKUserContentController();
38     controller.addScriptMessageHandler(new
           WKScriptMessageHandlerAdapter() {
39       @Override
40       public void didReceiveScriptMessage(
             WKUserContentController c, WKScriptMessage message)
             {
41         System.out.println("message: " + message.getBody());
42         if(message.getName().equals("handler")){
43           if(message.getBody().toString().equals("startScan"
                 ))
44             startScan();
45           if(message.getBody().toString().equals("stopScan"))
46             stopScan();
47         }
48       }
49     }, "handler");
50     WKWebViewConfiguration config = new
           WKWebViewConfiguration();
51     config.setUserContentController(controller);
52     CGRect frame = view.getFrame();
53     wv = new WKWebView(
54         new CGRect(frame.getMinX(), frame.getMinY() + 16,
55             frame.getWidth(), frame.getHeight() - 16),
56         config);
57     view.addSubview(wv);
58     NSURL bu = NSBundle.getMainBundle().getBundleURL();
```

```
59      wv.loadFileURL(new NSURL(bu.toString() + "index.html"),
            bu);
60
61      try {
62        client = new JsonRpcClientFactory().create(
63            ObservationReceiver.class,
64            new URL("http://10.229.248.86:8080/waikikiws/
                  services/ObservationReceiver")
65            );
66      } catch (MalformedURLException e) {
67        e.printStackTrace();
68      }
69    }
70
71    private void startScan(){
72      scanner.startScan(s -> {
73        System.out.println(JSON.encode(s).toString());
74        wv.evaluateJavaScript("found(" + JSON.encode(s) + ");",
              null);
75        // 送信
76        client.notify(createObservation(s)); //s = {"brightness
              ":-112,....}
77      });
78    }
79
80    private void stopScan(){
81      scanner.stopScan();
82    }
83  //s.get~で要素の値を取り出してObservationを生成
84    private Observation createObservation(EnvSensor s){
85      String TEMPERATURE = "http://openiot.eu/ontology/ns/
            AirTemperature";
86      String HUMIDITY = "http://openiot.eu/ontology/ns/
            AtmosphereHumidity";
87
88      Observation o = new Observation();
```

```
89      ArrayList<ObservedProperty> readings = new ArrayList<
            ObservedProperty>();
90      ObservedProperty tempProperty = new ObservedProperty();
91      ObservedProperty humdProperty = new ObservedProperty();
92
93      double temperature = s.getTemperature()/100;
94      double humidity = s.getHumidity()/100;
95
96      tempProperty.setPropertyType(TEMPERATURE);
97      humdProperty.setPropertyType(HUMIDITY);
98      tempProperty.setValue(temperature);
99      humdProperty.setValue(humidity);
100     readings.add(tempProperty);
101     readings.add(humdProperty);
102     o.setReadings(readings);
103     return o;
104   }
105
106   private ObservationReceiver client;
107   private EnvSensorScanner scanner = new EnvSensorScanner();
108   private final WKWebView wv;
109 }
```

A.1.2 WaikikiSensor.java

```
1  package org.langrid.waikiki.sensor;
2
3  import org.robovm.apple.foundation.NSAutoreleasePool;
4  import org.robovm.apple.uikit.UIApplication;
5  import org.robovm.apple.uikit.UIApplicationDelegateAdapter;
6  import org.robovm.apple.uikit.UIApplicationLaunchOptions;
7  import org.robovm.apple.uikit.UIScreen;
8  import org.robovm.apple.uikit.UIWindow;
9
10 public class WaikikiSensor extends
       UIApplicationDelegateAdapter {
11     private UIWindow window;
12     private MyViewController rootViewController;
```

```
13
14      @Override
15      public boolean didFinishLaunching(UIApplication
            application, UIApplicationLaunchOptions launchOptions)
            {
16          // Set up the view controller.
17          rootViewController = new MyViewController();
18
19          // Create a new window at screen size.
20          window = new UIWindow(UIScreen.getMainScreen().
                getBounds());
21          // Set the view controller as the root controller for
                the window.
22          window.setRootViewController(rootViewController);
23          // Make the window visible.
24          window.makeKeyAndVisible();
25
26          return true;
27      }
28
29      public static void main(String[] args) {
30          try (NSAutoreleasePool pool = new NSAutoreleasePool
                ()) {
31              UIApplication.main(args, null, WaikikiSensor.
                    class);
32          }
33      }
34  }
```

A.2 受信したデータをルールエンジンに挿入し、状況に応じた出力を得るモジュールのソースコード

A.2.1 ObservationReceiverImpl.java

```
1  package org.langrid.waikikiws.service;
2
3  import java.util.ArrayList;
```

```
4
5  import org.langrid.waikikiws.DroolsManager;
6  import org.langrid.waikikiws.service.api.ObservationReceiver;
7  import org.langrid.waikikiws.service.api.
       ObservationReceiverDebug;
8  import org.openiot.lsm.beans.Observation;
9  import org.openiot.lsm.beans.ObservedProperty;
10 import org.langrid.waikikiws.service.TargetLanguage;
11
12 public class ObservationReceiverImpl
13 implements ObservationReceiver, ObservationReceiverDebug{
14   @Override
15   public void notify(Observation o){
16     //Observationをルールエンジンへ挿入する
17     DroolsManager.getSession().insert(o);
18     //言語の指定
19     DroolsManager.getSession().insert(new TargetLanguage("en"
           ));
20   }
21
22   public static String TEMPERATURE = "http://openiot.eu/
       ontology/ns/AirTemperature";
23   public static String HUMIDITY = "http://openiot.eu/ontology
       /ns/AtmosphereHumidity";
24
25   //デモ用の関数
26   @Override
27   public void dummyNotify(double temperature, double humidity
       ,String tlanguage) {
28
29     Observation o = new Observation();
30     ArrayList<ObservedProperty> readings = new ArrayList<
         ObservedProperty>();
31     ObservedProperty tempProperty = new ObservedProperty();
32     ObservedProperty humdProperty = new ObservedProperty();
33     tempProperty.setPropertyType(TEMPERATURE);
34     humdProperty.setPropertyType(HUMIDITY);
```

```
35      tempProperty.setValue(temperature);
36      humdProperty.setValue(humidity);
37      readings.add(tempProperty);
38      readings.add(humdProperty);
39      o.setReadings(readings);;
40      DroolsManager.getSession().insert(o);
41      DroolsManager.getSession().insert(new TargetLanguage(
            tlanguage));
42    }
43
44  }
```

A.2.2 Translator.java

```
1  package org.langrid.waikikiws.service;
2
3  import java.io.IOException;
4  import java.io.InputStream;
5  import java.io.InputStreamReader;
6  import java.io.Reader;
7  import java.net.MalformedURLException;
8  import java.net.URL;
9  import java.util.Arrays;
10 import java.util.List;
11 import java.util.Properties;
12
13 import org.langrid.waikikiws.Bindings;
14 import org.langrid.waikikiws.service.api.TranslatorService;
15
16 import jp.go.nict.langrid.client.RequestAttributes;
17 import jp.go.nict.langrid.client.soap.SoapClientFactory;
18 import jp.go.nict.langrid.commons.cs.binding.BindingNode;
19 import jp.go.nict.langrid.service_1_2.
       AccessLimitExceededException;
20 import jp.go.nict.langrid.service_1_2.
       InvalidParameterException;
21 import jp.go.nict.langrid.service_1_2.
       NoAccessPermissionException;
```

```
22 import jp.go.nict.langrid.service_1_2.
       NoValidEndpointsException;
23 import jp.go.nict.langrid.service_1_2.ProcessFailedException;
24 import jp.go.nict.langrid.service_1_2.ServerBusyException;
25 import jp.go.nict.langrid.service_1_2.
       ServiceNotActiveException;
26 import jp.go.nict.langrid.service_1_2.
       ServiceNotFoundException;
27 import jp.go.nict.langrid.service_1_2.translation.
       TranslationService;
28
29 public class Translator implements TranslatorService{
30   public Translator() throws IOException {
31     Properties p = new Properties();
32     try(InputStream is = getClass().getResourceAsStream("/
         langrid.properties");
33         Reader r = new InputStreamReader(is, "UTF-8")){
34       p.load(r);
35     }
36     this.url = p.getProperty("url");
37     this.userId = p.getProperty("userId");
38     this.password = p.getProperty("password");
39   }
40   @Override
41   public String translate(String sourceLang, String
       targetLang, String source) {
42     List<BindingNode> bindings = Arrays.asList(
43         new BindingNode("MorphologicalAnalysisPL", "Mecab"),
44         new BindingNode("TranslationPL", "KyotoUJServer")
45         );
46 //     List<BindingNode> bindings = Bindings.getBindings();
47     try {
48       TranslationService trans = new SoapClientFactory().
           create(
49           TranslationService.class,
50           new URL(url + "
                 TranslationCombinedWithBilingualDictionaryWithLongestMatchSearch
```

```
                ”),
51              userId, password
52              );
53          for(BindingNode n : bindings){
54            ((RequestAttributes)trans).getTreeBindings().add(n);
55          }
56          return trans.translate(sourceLang, targetLang, source);
57        } catch (MalformedURLException |
            AccessLimitExceededException |
            InvalidParameterException |
            NoAccessPermissionException | ProcessFailedException |
             NoValidEndpointsException | ServerBusyException |
            ServiceNotActiveException | ServiceNotFoundException e
            ) {
58          throw new RuntimeException(e);
59        }
60      }
61
62      private String url;
63      private String userId;
64      private String password;
65  }
```

A.2.3 Binding.java

```
1  package org.langrid.waikikiws;
2
3  import java.util.Arrays;
4  import java.util.List;
5
6  import jp.go.nict.langrid.commons.cs.binding.BindingNode;
7
8  public class Bindings {
9    public static List<BindingNode> getBindings() {
10     return bindings;
11   }
12   public static void binding1(){
13     bindings = Arrays.asList(
```

```
14          new BindingNode("MorphologicalAnalysisPL", "Mecab"),
15          new BindingNode("TranslationPL", "KyotoUJServer")
16          );
17      //System.out.println("binding1");
18    }
19    public static void binding2(){
20      bindings = Arrays.asList(
21          new BindingNode("MorphologicalAnalysisPL", "Mecab"),
22          new BindingNode("
                BilingualDictionaryWithLongestMatchSearchPL", "
                KyotoTourismDictionaryDb"),
23          new BindingNode("TranslationPL", "KyotoUJServer")
24          );
25      //System.out.println("binding2");
26    }
27    public static void setBindings(List<BindingNode> bindings)
        {
28      Bindings.bindings = bindings;
29    }
30    private static List<BindingNode> bindings = Arrays.asList(
31        new BindingNode("MorphologicalAnalysisPL", "Mecab"),
32        new BindingNode("TranslationPL", "KyotoUJServer")
33        );
34  }
```

A.2.4 DroolsManager.java

```
1  package org.langrid.waikikiws;
2
3  import java.io.IOException;
4
5  import org.kie.api.runtime.KieSession;
6
7  public class DroolsManager {
8    public static synchronized KieSession getSession(){
9      if(session == null){
10       try {
```

```
11          session = DroolsUtil.createStreamSessionFromResource(
                "/badminton.drl");
12        } catch (IOException e) {
13          throw new RuntimeException(e);
14        }
15        Thread t = new Thread(() -> {
16          session.fireUntilHalt();
17        });
18        t.setDaemon(true);
19        t.start();
20        org.kie.api.runtime.rule.FactHandle.State.class.getName
              ();
21      }
22      return session;
23    }
24
25    private static KieSession session;
26  }
```

A.2.5 DroolsUtil.java

```
1  package org.langrid.waikikiws;
2
3  import java.io.IOException;
4  import java.io.InputStream;
5
6  import org.kie.api.KieBase;
7  import org.kie.api.KieBaseConfiguration;
8  import org.kie.api.KieServices;
9  import org.kie.api.builder.KieBuilder;
10 import org.kie.api.builder.KieFileSystem;
11 import org.kie.api.builder.Message;
12 import org.kie.api.builder.Results;
13 import org.kie.api.conf.EventProcessingOption;
14 import org.kie.api.runtime.KieContainer;
15 import org.kie.api.runtime.KieSession;
16
17 public class DroolsUtil {
```

```
18  public static KieSession createSessionFromResource(Package
       pkg, String rulePath) throws IOException{
19    return createSessionFromResource(
20        "/" + pkg.getName().replaceAll("\\.", "/") + "/" +
             rulePath);
21  }
22
23  public static KieSession createSessionFromResource(String
       rulePath)
24  throws IOException{
25    KieServices kieServices = KieServices.Factory.get();
26    KieFileSystem kfs = kieServices.newKieFileSystem();
27    try(InputStream is = DroolsUtil.class.getResourceAsStream
         (rulePath)){
28      // for each DRL file, referenced by a plain old path
           name:
29      kfs.write("src/main/resources" + rulePath,
30          kieServices.getResources().newInputStreamResource(
               is));
31      KieBuilder kieBuilder = kieServices.newKieBuilder( kfs
           ).buildAll();
32      Results results = kieBuilder.getResults();
33      if( results.hasMessages( Message.Level.ERROR ) ){
34        System.out.println( results.getMessages() );
35        throw new RuntimeException("### errors ###" );
36      }
37      KieContainer kieContainer = kieServices.newKieContainer
           (
38          kieServices.getRepository().getDefaultReleaseId()
               );
39      KieBase kieBase = kieContainer.getKieBase();
40      return kieBase.newKieSession();
41    }
42  }
43
44  public static KieSession createStreamSessionFromResource(
       Package pkg, String rulePath) throws IOException{
```

```
45      return createStreamSessionFromResource(
46          "/" + pkg.getName().replaceAll("\\.", "/") + "/" +
              rulePath);
47    }
48    public static KieSession createStreamSessionFromResource(
        String rulePath)
49    throws IOException{
50      KieServices kieServices = KieServices.Factory.get();
51      KieFileSystem kfs = kieServices.newKieFileSystem();
52      try(InputStream is = DroolsUtil.class.getResourceAsStream
          (rulePath)){
53        // for each DRL file, referenced by a plain old path
            name:
54        kfs.write("src/main/resources" + rulePath,
55            kieServices.getResources().newInputStreamResource(
                is));
56        KieBuilder kieBuilder = kieServices.newKieBuilder( kfs
            ).buildAll();
57        Results results = kieBuilder.getResults();
58        if( results.hasMessages( Message.Level.ERROR ) ){
59          System.out.println( results.getMessages() );
60          throw new RuntimeException("### errors ###" );
61        }
62        KieContainer kieContainer = kieServices.newKieContainer
            (
63            kieServices.getRepository().getDefaultReleaseId()
                );
64        KieBaseConfiguration config = KieServices.Factory.get
            ().newKieBaseConfiguration();
65        config.setOption( EventProcessingOption.STREAM );
66        KieBase kieBase = kieContainer.newKieBase(config);
67        return kieBase.newKieSession();
68      }
69    }
70  }
```

A.2.6 TargetLanguage

```java
package org.langrid.waikikiws.service;


public class TargetLanguage {
  public TargetLanguage(){
  }

  public TargetLanguage(String targetlang) {
    super();
    this.targetlanguage = targetlang;
  }

  public String getTargetlang() {
    return targetlanguage;
  }

  public void setTargetlang(String targetlang){
    this.targetlanguage = targetlang;
  }

  private String targetlanguage;

}
```

A.2.7 VoiceText.java

```java
package org.langrid.waikikiws;

import java.net.URL;
import jp.go.nict.langrid.client.soap.SoapClientFactory;
import jp.go.nict.langrid.service_1_2.speech.Speech;
import jp.go.nict.langrid.service_1_2.speech.
    TextToSpeechService;
import javax.sound.sampled.*;


import java.io.*;

```

```
12 public class VoiceText {
13     public void voicetext(String text,String lang) throws
           Exception {
14
15     // TODO 自動生成されたメソッド・スタブ
16       TextToSpeechService c =
17           new SoapClientFactory().create(
18               TextToSpeechService.class,
19               new URL("http://langrid.org/service_manager/
                     invoker/kyoto1.langrid:VoiceText"),
20               "ishida.kyoto-u", "tWJaakYm");
21         Speech s = c.speak(lang, text, "woman","audio/x-wav"
               );
22
23         byte[] buf = s.getAudio();
24         ByteArrayInputStream stream = new
               ByteArrayInputStream(buf);
25       AudioInputStream ais = AudioSystem.getAudioInputStream(
             stream);
26       byte[] data = new byte [ais.available()];
27       ais.read(data);
28       ais.close();
29         AudioFormat af = ais.getFormat();
30         DataLine.Info info = new DataLine.Info(SourceDataLine
               .class, af);
31         SourceDataLine line = (SourceDataLine)AudioSystem.
               getLine(info);
32         line.open();
33         line.start();
34         line.write(buf,0,buf.length);
35         line.drain();
36         line.close();
37     }
38 }
```

A.2.8 TransTextToSpeech.java

```
1 package org.langrid.waikikiws;
```

```
2
3 import org.langrid.waikikiws.VoiceText;
4 import org.langrid.waikikiws.service.Translator;
5
6 public class TransTextToSpeech {
7
8   public void transtexttospeech(String text,int i) throws
      Exception{
9     Translator trans = new Translator();
10    VoiceText tts = new VoiceText();
11    String lang;
12    if (i == 0) {
13      lang = "en";
14    }
15    else{
16      lang = "zh-CN";
17    }
18    String transtext = trans.translate("ja",lang,text);
19    tts.voicetext(transtext,lang);
20  }
21 }
```

A.2.9 badminton.drl

```
1 import org.openiot.lsm.beans.Observation;
2 import org.openiot.lsm.beans.ObservedProperty;
3 import org.langrid.waikikiws.TransTextToSpeech;
4 import org.langrid.waikikiws.service.TargetLanguage;
5 import java.util.ArrayList;
6
7 //テキストを音声出力する関数
8 function void TTTS(String text,int t){
9   TransTextToSpeech ttts = new TransTextToSpeech();
10  ttts.transtexttospeech(text,t);
11 }
12
13 //WBGTの計算
14 rule "WBGTcalc1"
```

```
15 when
16   $o: Observation()
17   $op1: ObservedProperty(
18     propertyType == "http://openiot.eu/ontology/ns/
         AirTemperature"
19     )
20     from $o.readings
21   $op2: ObservedProperty(
22     value > 80 &&
23     propertyType == "http://openiot.eu/ontology/ns/
         AtmosphereHumidity"
24     )
25     from $o.readings
26   $t: TargetLanguage()
27 then
28   double tmp = Double.parseDouble($op1.getValue().toString
       ());
29   double hmd = Double.parseDouble($op2.getValue().toString
       ());
30   double WBGT = tmp + (hmd - 80) / 5;
31   System.out.println("WBGT :" + WBGT + "tmp: " + tmp + "hmd "
       + hmd);
32     Observation o1 = new Observation();
33     Observation o2 = new Observation();
34     Observation o3 = new Observation();
35   ArrayList<ObservedProperty> readings1 = new ArrayList<
       ObservedProperty>();
36   ArrayList<ObservedProperty> readings2 = new ArrayList<
       ObservedProperty>();
37   ArrayList<ObservedProperty> readings3 = new ArrayList<
       ObservedProperty>();
38   ObservedProperty wbgtProperty = new ObservedProperty();
39   ObservedProperty tlangProperty = new ObservedProperty();
40   ObservedProperty tmpProperty = new ObservedProperty();
41   ObservedProperty hmdProperty = new ObservedProperty();
42   wbgtProperty.setPropertyType("http://ishida.kyoto-u/
       watanabe/WetBulbGlobTemperature");
```

```
43    tmpProperty.setPropertyType("http://openiot.eu/ontology/ns/
          AirTemperature");
44    hmdProperty.setPropertyType("http://openiot.eu/ontology/ns/
          AtmosphereHumidity");
45    tlangProperty.setPropertyType("http://ishida.kyoto-u/
          watanabe/TargetTransLanguage");
46    wbgtProperty.setValue(WBGT);
47    tmpProperty.setValue(tmp);
48    hmdProperty.setValue(hmd);
49    String st =  $t.getTargetlang();
50    if(st.equals("en")){
51      tlangProperty.setValue(0);
52    }else if (st.equals("zh-CN")) {
53      tlangProperty.setValue(1);
54    }
55    readings1.add(wbgtProperty);
56    readings1.add(tlangProperty);
57    readings2.add(tmpProperty);
58    readings2.add(tlangProperty);
59    readings3.add(hmdProperty);
60    readings3.add(tlangProperty);
61    o1.setReadings(readings1);
62    o2.setReadings(readings2);
63    o3.setReadings(readings3);
64    insert(o1);
65    insert(o2);
66    insert(o3);
67  end
68
69  rule "WBGTcalc2"
70  when
71    $o: Observation()
72    $op1: ObservedProperty(
73      propertyType == "http://openiot.eu/ontology/ns/
          AirTemperature"
74      )
75      from $o.readings
```

```
76    $op2: ObservedProperty(
77      value <= 80 &&
78      propertyType == "http://openiot.eu/ontology/ns/
            AtmosphereHumidity"
79      )
80      from $o.readings
81    $t: TargetLanguage()
82  then
83    double tmp = Double.parseDouble($op1.getValue().toString
          ());
84    double hmd = Double.parseDouble($op2.getValue().toString
          ());
85    double WBGT = tmp - (80 - hmd) / 5;
86    System.out.println("WBGT :" + WBGT + "tmp: " + tmp + "hmd "
          + hmd);
87      Observation o1 = new Observation();
88      Observation o2 = new Observation();
89      Observation o3 = new Observation();
90    ArrayList<ObservedProperty> readings1 = new ArrayList<
          ObservedProperty>();
91    ArrayList<ObservedProperty> readings2 = new ArrayList<
          ObservedProperty>();
92    ArrayList<ObservedProperty> readings3 = new ArrayList<
          ObservedProperty>();
93    ObservedProperty wbgtProperty = new ObservedProperty();
94    ObservedProperty tlangProperty = new ObservedProperty();
95    ObservedProperty tmpProperty = new ObservedProperty();
96    ObservedProperty hmdProperty = new ObservedProperty();
97    wbgtProperty.setPropertyType("http://ishida.kyoto-u/
          watanabe/WetBulbGlobTemperature");
98    tmpProperty.setPropertyType("http://openiot.eu/ontology/ns/
          AirTemperature");
99    hmdProperty.setPropertyType("http://openiot.eu/ontology/ns/
          AtmosphereHumidity");
100   tlangProperty.setPropertyType("http://ishida.kyoto-u/
          watanabe/TargetTransLanguage");
101   wbgtProperty.setValue(WBGT);
```

```
102     tmpProperty.setValue(tmp);
103     hmdProperty.setValue(hmd);
104     String st =  $t.getTargetlang();
105     if(st.equals("en")){
106       tlangProperty.setValue(0);
107     }else if (st.equals("zh-CN")) {
108       tlangProperty.setValue(1);
109     }
110     readings1.add(wbgtProperty);
111     readings1.add(tlangProperty);
112     readings2.add(tmpProperty);
113     readings2.add(tlangProperty);
114     readings3.add(hmdProperty);
115     readings3.add(tlangProperty);
116     o1.setReadings(readings1);
117     o2.setReadings(readings2);
118     o3.setReadings(readings3);
119     insert(o1);
120     insert(o2);
121     insert(o3);
122 end
123
124 //WBGTによって運動時の注意喚起を行うようにする
125 rule "phase5"  //WBGT>=31
126 when
127     $o: Observation()
128     $op1: ObservedProperty(
129       value >= 31 &&
130       propertyType == "http://ishida.kyoto-u/watanabe/
            WetBulbGlobTemperature"
131       )
132       from $o.readings
133     $op2: ObservedProperty(
134       propertyType == "http://ishida.kyoto-u/watanabe/
            TargetTransLanguage"
135       )
136       from $o.readings
```

```
137 then
138   System.out.println("運動を中止しましょう.");
139   TTTS("運動を中止しましょう
          .",Integer.parseInt($op2.getValue().toString()));
140 end
141
142 rule "phase4"   //28<=WBGT<31
143 when
144   $o: Observation()
145   $op1: ObservedProperty(
146     value < 31 && value >= 28 &&
147     propertyType == "http://ishida.kyoto-u/watanabe/
          WetBulbGlobTemperature"
148     )
149     from $o.readings
150   $op2: ObservedProperty(
151     propertyType == "http://ishida.kyoto-u/watanabe/
          TargetTransLanguage"
152     )
153     from $o.readings
154 then
155   System.out.println("激しい運動は避け、積極的に休息と水分補給を行いま
          しょう.");
156   TTTS("激しい運動は避け、積極的に休息と水分補給を行いましょう
          .",Integer.parseInt($op2.getValue().toString()));
157 end
158
159 rule "phase3"   //25<=WBGT<28
160 when
161   $o: Observation()
162   $op1: ObservedProperty(
163     value < 28 && value >= 25 &&
164     propertyType == "http://ishida.kyoto-u/watanabe/
          WetBulbGlobTemperature"
165     )
166     from $o.readings
167   $op2: ObservedProperty(
```

```
168       propertyType == "http://ishida.kyoto-u/watanabe/
              TargetTransLanguage"
169       )
170       from $o.readings
171   then
172     System.out.println("激しい運動を行う際は
              ,30分おきくらいに休息をとりましょう.");
173     TTTS("激しい運動を行う際は,30分おきくらいに休息をとりましょう."
              ,Integer.parseInt($op2.getValue().toString()));
174   end
175
176   rule "phase2"   //21<=WBGT<25
177   when
178     $o: Observation()
179     $op1: ObservedProperty(
180       value < 25 && value >= 21 &&
181       propertyType == "http://ishida.kyoto-u/watanabe/
              WetBulbGlobTemperature"
182       )
183       from $o.readings
184     $op2: ObservedProperty(
185       propertyType == "http://ishida.kyoto-u/watanabe/
              TargetTransLanguage"
186       )
187       from $o.readings
188   then
189     System.out.println("水分補給には十分気をつけましょう.");
190     TTTS("水分補給には十分気をつけましょう
              .",Integer.parseInt($op2.getValue().toString()));
191   end
192
193   rule "phase1"    //WBGT<21
194   when
195     $o: Observation()
196     $op1: ObservedProperty(
197       value < 21 &&
198       propertyType == "http://ishida.kyoto-u/watanabe/
              WetBulbGlobTemperature"
```

```
199       )
200       from $o.readings
201     $op2: ObservedProperty(
202       propertyType == "http://ishida.kyoto-u/watanabe/
              TargetTransLanguage"
203       )
204       from $o.readings
205   then
206     System.out.println("熱中症の危険は少ないですが
          ,適宜水分補給をしましょう.");
207     TTTS("熱中症の危険は少ないですが,適宜水分補給をしましょう.",
          Integer.parseInt($op2.getValue().toString()));
208   end
209
210   //湿度が高いと床が滑りやすくなる注意
211   rule "floor"
212   when
213     $o: Observation()
214     $h: ObservedProperty(
215       value >= 90 &&
216       propertyType == "http://openiot.eu/ontology/ns/
              AtmosphereHumidity"
217       )
218       from $o.readings
219     $op2: ObservedProperty(
220       propertyType == "http://ishida.kyoto-u/watanabe/
              TargetTransLanguage"
221       )
222       from $o.readings
223   then
224     System.out.println("湿度が高く床が滑りやすくなっています
          .気をつけましょう.");
225     TTTS("湿度が高く床が滑りやすくなっています.気をつけましょう.",
          Integer.parseInt($op2.getValue().toString()));
226   end
227
228   //温度によって適切なシャトルの番号を提示する
```

```
229 rule "shuttle1"    //1番シャトル
230 when
231   $o: Observation()
232   $op1: ObservedProperty(
233     value >= 33 &&
234     propertyType == "http://openiot.eu/ontology/ns/
          AirTemperature"
235     )
236     from $o.readings
237   $op2: ObservedProperty(
238     propertyType == "http://ishida.kyoto-u/watanabe/
          TargetTransLanguage"
239     )
240     from $o.readings
241 then
242   System.out.println("1番のシャトルを使いましょう.");
243   TTTS("1番のシャトルを使いましょう.",Integer.parseInt($op2.
        getValue().toString()));
244 end
245
246 rule "shuttle2"    //2番シャトル
247 when
248   $o: Observation()
249   $op1: ObservedProperty(
250     value < 33 && value >= 27 &&
251     propertyType == "http://openiot.eu/ontology/ns/
          AirTemperature"
252     )
253     from $o.readings
254   $op2: ObservedProperty(
255     propertyType == "http://ishida.kyoto-u/watanabe/
          TargetTransLanguage"
256     )
257     from $o.readings
258 then
259   System.out.println("2番のシャトルを使いましょう.");
```

```
260   TTTS("2番のシャトルを使いましょう.",Integer.parseInt($op2.
          getValue().toString()));
261 end
262
263 rule "shuttle3"    //3番シャトル
264 when
265   $o: Observation()
266   $op1: ObservedProperty(
267     value < 27 && value >= 22 &&
268     propertyType == "http://openiot.eu/ontology/ns/
          AirTemperature"
269     )
270     from $o.readings
271   $op2: ObservedProperty(
272     propertyType == "http://ishida.kyoto-u/watanabe/
          TargetTransLanguage"
273     )
274     from $o.readings
275 then
276   System.out.println("3番のシャトルを使いましょう.");
277   TTTS("3番のシャトルを使いましょう.",Integer.parseInt($op2.
          getValue().toString()));
278 end
279
280 rule "shuttle4"    //4番シャトル
281 when
282   $o: Observation()
283   $op1: ObservedProperty(
284     value < 22 && value >= 17 &&
285     propertyType == "http://openiot.eu/ontology/ns/
          AirTemperature"
286     )
287     from $o.readings
288   $op2: ObservedProperty(
289     propertyType == "http://ishida.kyoto-u/watanabe/
          TargetTransLanguage"
290     )
```

```
291       from $o.readings
292   then
293     System.out.println("4番のシャトルを使いましょう.");
294     TTTS("4番のシャトルを使いましょう.",Integer.parseInt($op2.
            getValue().toString()));
295   end
296
297   rule "shuttle5"    //5番シャトル
298   when
299     $o: Observation()
300     $op1: ObservedProperty(
301       value < 17 && value >= 12 &&
302       propertyType == "http://openiot.eu/ontology/ns/
            AirTemperature"
303       )
304       from $o.readings
305     $op2: ObservedProperty(
306       propertyType == "http://ishida.kyoto-u/watanabe/
            TargetTransLanguage"
307       )
308       from $o.readings
309   then
310     System.out.println("5番のシャトルを使いましょう.");
311     TTTS("5番のシャトルを使いましょう.",Integer.parseInt($op2.
            getValue().toString()));
312   end
313
314   rule "shuttle6"    //6番シャトル
315   when
316     $o: Observation()
317     $op1: ObservedProperty(
318       value < 12 &&
319       propertyType == "http://openiot.eu/ontology/ns/
            AirTemperature"
320       )
321       from $o.readings
322     $op2: ObservedProperty(
```

```
323       propertyType == "http://ishida.kyoto-u/watanabe/
              TargetTransLanguage"
324       )
325       from $o.readings
326   then
327     System.out.println("6番のシャトルを使いましょう.");
328     TTTS("6番のシャトルを使いましょう.",Integer.parseInt($op2.
            getValue().toString()));
329   end
330
331   //温度によってラケットに張るストリングのテンションの助言をする
332   rule "stringsh"     //温度が高いときのストリング
333   when
334     $o: Observation()
335     $op1: ObservedProperty(
336       value > 25 &&
337       propertyType == "http://openiot.eu/ontology/ns/
              AirTemperature"
338       )
339       from $o.readings
340     $op2: ObservedProperty(
341       propertyType == "http://ishida.kyoto-u/watanabe/
              TargetTransLanguage"
342       )
343       from $o.readings
344   then
345     System.out.println("適正温度のときより
          +1ポンドのガットが適切です.");
346     TTTS("適正温度のときより+1ポンドのガットが適切です.",Integer.
            parseInt($op2.getValue().toString()));
347   end
348
349   rule "stringsn"     //適正温度のときのストリング
350   when
351     $o: Observation()
352     $op1: ObservedProperty(
353       value <= 25 && value >= 15 &&
```

```
354       propertyType == "http://openiot.eu/ontology/ns/
              AirTemperature"
355       )
356       from $o.readings
357     $op2: ObservedProperty(
358       propertyType == "http://ishida.kyoto-u/watanabe/
              TargetTransLanguage"
359       )
360       from $o.readings
361 then
362     System.out.println("適正");
363     TTTS("ガットの適正温度です
          .",Integer.parseInt($op2.getValue().toString()));
364 end
365
366 rule "stringsl"    //温度が低いときのストリング
367 when
368     $o: Observation()
369     $op1: ObservedProperty(
370       value < 15 &&
371       propertyType == "http://openiot.eu/ontology/ns/
              AirTemperature"
372       )
373       from $o.readings
374     $op2: ObservedProperty(
375       propertyType == "http://ishida.kyoto-u/watanabe/
              TargetTransLanguage"
376       )
377       from $o.readings
378 then
379     System.out.println("適正温度のときより
          -1ポンドのガットが適切です.");
380     TTTS("適正温度のときより-1ポンドのガットが適切です.",Integer.
          parseInt($op2.getValue().toString()));
381 end
```

A.3 オムロンのセンサー定義

A.3.1 EnvSensor.java

```
1  package org.langrid.waikiki.sensor.omron;
2
3  public class EnvSensor {
4    public EnvSensor() {
5    }
6    //EnvSensor 11個の変数
7    public EnvSensor(
8        String uuid,
9        int lineNo, int temperature, int humidity, int
             brightness,
10       int uvIndex, int pressure, int noise,
11       int discomfortIndex, int heatstrokeIndex, int
             cellVoltage) {
12     this.uuid = uuid;
13     this.lineNo = lineNo;
14     this.temperature = temperature;
15     this.humidity = humidity;
16     this.brightness = brightness;
17     this.uvIndex = uvIndex;
18     this.pressure = pressure;
19     this.noise = noise;
20     this.discomfortIndex = discomfortIndex;
21     this.heatstrokeIndex = heatstrokeIndex;
22     this.cellVoltage = cellVoltage;
23   }
24 //行番号が1,その他が2バイトずつの19バイト
25   public static EnvSensor create(String uuid, byte[] bytes){
26     if(bytes.length != 19) throw new RuntimeException("The
           length of bytes must be 19");
27     return new EnvSensor(
28         uuid,
29         (int)bytes[0],
30         bytes[1] + (bytes[2] << 8),
31         bytes[3] + (bytes[4] << 8),
```

```
32          bytes[5] + (bytes[6] << 8),
33          bytes[7] + (bytes[8] << 8),
34          bytes[9] + (bytes[10] << 8),
35          bytes[11] + (bytes[12] << 8),
36          bytes[13] + (bytes[14] << 8),
37          bytes[15] + (bytes[16] << 8),
38          bytes[17] + (bytes[18] << 8));
39    }
40
41    public String getUuid() {
42      return uuid;
43    }
44    public void setUuid(String uuid) {
45      this.uuid = uuid;
46    }
47    public int getLineNo() {
48      return lineNo;
49    }
50    public void setLineNo(int lineNo) {
51      this.lineNo = lineNo;
52    }
53    public int getTemperature() {
54      return temperature;
55    }
56    public void setTemperature(int temperature) {
57      this.temperature = temperature;
58    }
59    public int getHumidity() {
60      return humidity;
61    }
62    public void setHumidity(int humidity) {
63      this.humidity = humidity;
64    }
65    public int getBrightness() {
66      return brightness;
67    }
68    public void setBrightness(int brightness) {
```

```
69      this.brightness = brightness;
70    }
71    public int getUvIndex() {
72      return uvIndex;
73    }
74    public void setUvIndex(int uvIndex) {
75      this.uvIndex = uvIndex;
76    }
77    public int getPressure() {
78      return pressure;
79    }
80    public void setPressure(int pressure) {
81      this.pressure = pressure;
82    }
83    public int getNoise() {
84      return noise;
85    }
86    public void setNoise(int noise) {
87      this.noise = noise;
88    }
89    public int getDiscomfortIndex() {
90      return discomfortIndex;
91    }
92    public void setDiscomfortIndex(int discomfortIndex) {
93      this.discomfortIndex = discomfortIndex;
94    }
95    public int getHeatstrokeIndex() {
96      return heatstrokeIndex;
97    }
98    public void setHeatstrokeIndex(int heatstrokeIndex) {
99      this.heatstrokeIndex = heatstrokeIndex;
100   }
101   public int getCellVoltage() {
102     return cellVoltage;
103   }
104   public void setCellVoltage(int cellVoltage) {
105     this.cellVoltage = cellVoltage;
```

```
106     }
107
108     private String uuid;
109     private int lineNo; // ("行番号: " + bytes[0]);
110     private int temperature; // ("温度: " + (bytes[1] + (bytes
            [2] << 8)));
111     private int humidity; // ("相対湿度: " + (bytes[3] + (bytes
            [4] << 8)));
112     private int brightness; // ("照度: " + (bytes[5] + (bytes
            [6] << 8)));
113     private int uvIndex; // ("UVI: " + (bytes[7] + (bytes[8] <<
             8)));
114     private int pressure; // ("気圧: " + (bytes[9] + (bytes[10]
            << 8)));
115     private int noise; // ("騒音: " + (bytes[11] + (bytes[12]
           << 8)));
116     private int discomfortIndex; // ("不快指数: " + (bytes[13]
            + (bytes[14] << 8)));
117     private int heatstrokeIndex; // ("熱中症危険度: " + (bytes
            [15] + (bytes[16] << 8)));
118     private int cellVoltage; // ("電池電圧: " + (bytes[17] + (
            bytes[18] << 8)));
119 }
```

A.3.2 EnvSensorListener.java

```
1 package org.langrid.waikiki.sensor.omron;
2
3 public interface EnvSensorListener {
4   void onFound(EnvSensor sensor);
5 }
```

A.3.3 EnvSensorScanner.java

```
1 package org.langrid.waikiki.sensor.omron;
2
3 import java.util.LinkedHashSet;
4 import java.util.Set;
5
```

```
6  import org.robovm.apple.corebluetooth.CBAdvertisementData;
7  import org.robovm.apple.corebluetooth.CBCentralManager;
8  import org.robovm.apple.corebluetooth.
       CBCentralManagerDelegateAdapter;
9  import org.robovm.apple.corebluetooth.CBCharacteristic;
10 import org.robovm.apple.corebluetooth.CBPeripheral;
11 import org.robovm.apple.corebluetooth.
       CBPeripheralDelegateAdapter;
12 import org.robovm.apple.corebluetooth.CBService;
13 import org.robovm.apple.foundation.NSData;
14 import org.robovm.apple.foundation.NSError;
15 import org.robovm.apple.foundation.NSNumber;
16
17 import jp.go.nict.langrid.client.jsonrpc.JsonRpcClientFactory
       ;
18 import jp.go.nict.langrid.repackaged.net.arnx.jsonic.JSON;
19
20 public class EnvSensorScanner {
21   private Set<CBPeripheral> peripherals;
22   private CBCentralManager central;
23   public void startScan(EnvSensorListener listener){
24     this.peripherals = new LinkedHashSet<>();
25     this.central = new CBCentralManager(
26         new CBCentralManagerDelegateAdapter(){
27           @Override
28           public void didUpdateState(CBCentralManager central
                ) {
29             switch(central.getState().toString()){
30               case "PoweredOn":
31                 central.scanForPeripherals(null, null);
32                 break;
33             }
34             super.didUpdateState(central);
35           }
36           @Override
37           public void didDiscoverPeripheral(CBCentralManager
                central, CBPeripheral peripheral,
```

```
        CBAdvertisementData advertisementData, NSNumber
        rssi) {
38          if(peripherals.contains(peripheral)) return;
39          peripherals.add(peripheral);
40          System.out.println(peripheral);
41          if("EnvSensor-BL01".equals(peripheral.getName
                ())){
42            central.connectPeripheral(peripheral, null);
43          }
44        }
45        @Override
46        public void didConnectPeripheral(CBCentralManager
            central, CBPeripheral peripheral) {
47          peripheral.setDelegate(new
              CBPeripheralDelegateAdapter(){
48            @Override
49            public void didDiscoverServices(CBPeripheral
                peripheral, NSError error) {
50              for(CBService s : peripheral.getServices()){
51                if(s.getUUID().toString().equals("0C4C3000
                    -7700-46F4-AA96-D5E974E32A54")){
52                  peripheral.discoverCharacteristics(null,
                      s);
53                }
54              }
55            }
56            @Override
57            public void didDiscoverCharacteristics(
                CBPeripheral peripheral, CBService service,
58                NSError error) {
59              for(CBCharacteristic c : service.
                    getCharacteristics()){
60                if(c.getUUID().toString().equals("0C4C3001
                    -7700-46F4-AA96-D5E974E32A54")){
61                  peripheral.readValue(c);
62                }
63              }
```

```
64                  }
65                  @Override
66                  public void didUpdateValue(CBPeripheral
                        peripheral, CBCharacteristic characteristic,
67                      NSError error) {
68                    NSData data = characteristic.getValue();
69                    listener.onFound(EnvSensor.create(peripheral.
                          getIdentifier().toString(), data.getBytes
                          ()));
70                  }
71                });
72                peripheral.discoverServices(null);
73              }
74              @Override
75              public void didFailToConnectPeripheral(
                    CBCentralManager central, CBPeripheral
                    peripheral,
76                    NSError error) {
77                System.out.println("failed to connected to " +
                      peripheral);
78              }
79            }, null
80            );
81    }
82    public void stopScan(){
83      central.stopScan();
84    }
85  }
```

A.4 OpenIoTのデータ定義

A.4.1 Observation.java

```
1  package org.openiot.lsm.beans;
2  /**
3  *     Copyright (c) 2011-2014, OpenIoT
4  *
5  *     This file is part of OpenIoT.
```

```
6 *
7 *     OpenIoT is free software: you can redistribute it and/or
      modify
8 *     it under the terms of the GNU Lesser General Public
      License as published by
9 *     the Free Software Foundation, version 3 of the License.
10 *
11 *     OpenIoT is distributed in the hope that it will be
      useful,
12 *     but WITHOUT ANY WARRANTY; without even the implied
      warranty of
13 *     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
      See the
14 *     GNU Lesser General Public License for more details.
15 *
16 *     You should have received a copy of the GNU Lesser
      General Public License
17 *     along with OpenIoT.  If not, see <http://www.gnu.org/
      licenses/>.
18 *
19 *      Contact: OpenIoT mailto: info@openiot.eu
20 */
21 import java.util.ArrayList;
22 import java.util.Date;
23
24 /**
25  *
26  * @author Hoan Nguyen Mau Quoc
27  *
28  */
29 public class Observation implements java.io.Serializable {
30   private String id;
31   private Date times;
32   private String sensorId;
33   private String featureOfInterest="";
34   private ArrayList<ObservedProperty> readings;
35   private String metaGraph;
```

```
36    private String dataGraph;
37
38    public Observation(){
39      id = ""+System.nanoTime();
40      readings = new ArrayList<ObservedProperty>();
41    }
42
43    public String getId() {
44      return id;
45    }
46    public void setId(String id) {
47      this.id = id;
48    }
49    public Date getTimes() {
50      return times;
51    }
52    public void setTimes(Date times) {
53      this.times = times;
54    }
55    public String getSensor() {
56      return sensorId;
57    }
58    public void setSensor(String sensorId) {
59      this.sensorId = sensorId;
60    }
61    public String getFeatureOfInterest() {
62      return featureOfInterest;
63    }
64    public void setFeatureOfInterest(String featureOfInterest)
        {
65      this.featureOfInterest = featureOfInterest;
66    }
67    public ArrayList<ObservedProperty> getReadings() {
68      return readings;
69    }
70    public void setReadings(ArrayList<ObservedProperty>
        readings) {
```

```
71      this.readings = readings;
72    }
73
74    public void addReading(ObservedProperty reading){
75      readings.add(reading);
76    }
77
78    public void removeReading(ObservedProperty reading){
79      readings.remove(reading);
80    }
81
82    public String getMetaGraph() {
83      return metaGraph;
84    }
85
86    public void setMetaGraph(String metaGraph) {
87      this.metaGraph = metaGraph;
88    }
89
90    public String getDataGraph() {
91      return dataGraph;
92    }
93
94    public void setDataGraph(String dataGraph) {
95      this.dataGraph = dataGraph;
96    }
97
98  }
```

A.4.2 ObserbatonProperty.java

```
1  package org.openiot.lsm.beans;
2  /**
3  *     Copyright (c) 2011-2014, OpenIoT
4  *
5  *     This file is part of OpenIoT.
6  *
```

```
 7 *     OpenIoT is free software: you can redistribute it and/or
       modify
 8 *     it under the terms of the GNU Lesser General Public
       License as published by
 9 *     the Free Software Foundation, version 3 of the License.
10 *
11 *     OpenIoT is distributed in the hope that it will be
       useful,
12 *     but WITHOUT ANY WARRANTY; without even the implied
       warranty of
13 *     MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
       See the
14 *     GNU Lesser General Public License for more details.
15 *
16 *     You should have received a copy of the GNU Lesser
       General Public License
17 *     along with OpenIoT.  If not, see <http://www.gnu.org/
       licenses/>.
18 *
19 *      Contact: OpenIoT mailto: info@openiot.eu
20 */
21 import java.util.ArrayList;
22 import java.util.Date;
23
24 /**
25  *
26  * @author Hoan Nguyen Mau Quoc
27  *
28  */
29 public class Observation implements java.io.Serializable {
30   private String id;
31   private Date times;
32   private String sensorId;
33   private String featureOfInterest="";
34   private ArrayList<ObservedProperty> readings;
35   private String metaGraph;
36   private String dataGraph;
```

```
37
38    public Observation(){
39      id = ""+System.nanoTime();
40      readings = new ArrayList<ObservedProperty>();
41    }
42
43    public String getId() {
44      return id;
45    }
46    public void setId(String id) {
47      this.id = id;
48    }
49    public Date getTimes() {
50      return times;
51    }
52    public void setTimes(Date times) {
53      this.times = times;
54    }
55    public String getSensor() {
56      return sensorId;
57    }
58    public void setSensor(String sensorId) {
59      this.sensorId = sensorId;
60    }
61    public String getFeatureOfInterest() {
62      return featureOfInterest;
63    }
64    public void setFeatureOfInterest(String featureOfInterest)
        {
65      this.featureOfInterest = featureOfInterest;
66    }
67    public ArrayList<ObservedProperty> getReadings() {
68      return readings;
69    }
70    public void setReadings(ArrayList<ObservedProperty>
        readings) {
71      this.readings = readings;
```

```
72    }
73
74    public void addReading(ObservedProperty reading){
75      readings.add(reading);
76    }
77
78    public void removeReading(ObservedProperty reading){
79      readings.remove(reading);
80    }
81
82    public String getMetaGraph() {
83      return metaGraph;
84    }
85
86    public void setMetaGraph(String metaGraph) {
87      this.metaGraph = metaGraph;
88    }
89
90    public String getDataGraph() {
91      return dataGraph;
92    }
93
94    public void setDataGraph(String dataGraph) {
95      this.dataGraph = dataGraph;
96    }
97
98  }
```