

記事

[Toshihiko Minamoto](#) · 2021年1月14日 6m read

IRISにてMQTTブローカーから気象データを取得しデータベースに格納する

前回のつづき

として、いよいよIRISのインターオペラビリティ機能を使ってMQTTブローカーからメッセージを受信し、データベース

に格納する方

法について解説したいと思います。IRISのインターオペラビリティ機能につきましては[こちら](#)をご参照ください。

ネームスペース作成

インストール時に作成されているUSERネームスペースはInteroperabilityに必要なライブラリを参照するためのマッピングができていません。

そのため、新たにネームスペースを作成する必要があります。

作成方法は、以下の通りです。

1. システム管理ポータルを起動し、「システム管理」「構成」「システム構成」「ネームスペース」をクリックします。
2. 以下のようなネームスペース一覧画面が表示されますので、上にある「新規ネームスペース作成」ボタンをクリックします。

The screenshot shows the InterSystems IRIS Data Platform Management Portal. The breadcrumb navigation is 'システム > 構成 > ネームスペース'. The main heading is 'ネームスペース' (Namespaces). A button labeled '新規ネームスペース作成' (Create New Namespace) is highlighted with a red box. Below the heading, there is a table listing namespaces with columns: Namespace, Globals, Routines, Temp Storage. The first row shows '%ALL', '%DEFAULTDB', '%DEFAULTDB', and 'IRISTEMP'. At the bottom, there are links for 'グローバルマッピング' (Global Mapping), 'ルーチンマッピング' (Routine Mapping), and 'パッケージマッピング' (Package Mapping).

3. 以下のような新規ネームスペース作成画面が表示されますので、「ネームスペース名」欄にネームスペースの名称を入力し、「グローバルのための既存のデータベースを選択」欄、「ルーチンのための既存のデータベースを選択」欄で共にデータベース「USER」を選択します。

- 「相互運用プロダクション用にネームスペースを有効化」欄にチェックが付いていることを確認の上、画面上の「保存」ボタンをクリックします。
4. 以下のような処理ログが表示されますので、画面下の「閉じる」ボタンをクリックします。



以上でネームスペースが作成できました。

気象データテーブルの作成

以下のように観測地点、取得時刻、気温、湿度、気圧を格納するクラス(MQTT.Data.WeatherInfo)を作成します。

```
/// ?????
Class MQTT.Data.WeatherInfo Extends %Persistent
{
    /// ????? (ClientID)
    Property PointName As %String;
    /// ?????
    Property MeasureTime As %TimeStamp;
    /// ??
    Property Temperature As %Numeric;
    /// ??
    Property Humidity As %Numeric;
    /// ??
    Property Pressure As %Numeric;

    Index PointDate On (PointName, MeasureTime);
}
```

気象データ格納オペレーションの作成

以下のように、MQTT.BO.StoreDataというビジネスオペレーションクラスを作成し、MQTTメッセージ(EnsLib.MQTT.Message)からデータを取得しMQTT.Data.WeatherInfoクラスに格納するStoreData()メソッドを追加します。MessageMapの設定も忘れずに追加してください。

```
/// MQTT????????????????
Class MQTT.StoreData Extends Ens.BusinessOperation [ Language = objectscript ]
{

    Parameter INVOCATION = "Queue";
```

```
Method StoreData(pRequest As EnsLib.MQTT.Message, Output pResponse As
Ens.Response) As %Status
{
    set obj=##class(MQTT.Data.WeatherInfo).%New()
    set obj.MeasureTime=$zdatetime($horolog,3)
    set obj.Temperature=$piece(pRequest.StringValue,",")
    set obj.Humidity=$piece(pRequest.StringValue,",",2)
    set obj.Pressure=$piece(pRequest.StringValue,",",3)
    set obj.PointName=$piece(pRequest.Topic,"/",2)

    return obj.%Save()
}

XData MessageMap
{
    <MapItems>
        <MapItem MessageType="EnsLib.MQTT.Message">
            <Method>StoreData</Method>
        </MapItem>
    </MapItems>
}
```

プロダクションの作成

システム管理ポータルより「Interoperability」「構成」「プロダクション」メニューをクリックします。以下のよう
にインターオペラビリティ機能を実行するネームスペースを聞いてきますので、先ほど作成したネームスペース
(TEST)を選択します。

ようこそ, _SYSTEM

表示:  

**ホーム**

**Health**

**Analytics**

**Interoperability**

**システムオペレーション**

%SYS ネームスペース

別のネームスペースを選択

プロダクションに使用可能なネームスペース

HSLIB

HSSYS

TEST

以下のようにプロダクション構成画面が表示されますので、画面中央上にある「プロダクション設定」リンクをクリックし、右にある「アクション」タブをクリック、「新規」ボタンをクリックします。

ESTPKG.FoundationProduction)


ン構成

開始する 停止する

カテゴリ: 全て 凡例 **プロダクション設定**


プロセス オペレーション


● Ens.Activity.Operation.Local


ソート: 名前 ステータス 番号 表示: 

プロダクション設定

設定 キュー ログ メッセージ ジョブ **アクション**

**エクスポート**
デプロイ用にエクスポート

**再エクスポート**
既存のデプロイ・パッケージを修正する

**新規**
新しいプロダクションを構築

以下のようなダイアログボックスが表示されますので、パッケージ名、プロダクション名を入力し、「OK」ボタンをクリックします。

プロダクション・ウィザード
新しいプロダクション定義を作成します。

パッケージ: MQTT

プロダクション名: MQTTProduction

プロダクション説明:

プロダクションの種類

- ☒ 一般 - 一般的なプロダクションを作成して後で構成できるようにします
- ☐ HL7 メッセージング - HL7 メッセージルーティングプロダクションを作成します

このフォームを使用して新しいプロダクションを作成します。

このフォームにあるフィールドのヘルプを見るには、フィールド名の上にマウスカースールを移動してください。

キャンセル OK

ビジネスオペレーションの作成

アイテムが無いプロダクション構成画面が表示されますので、画面上の「オペレーション」の右にある「+」ボタンをクリックします。

クリックしますと以下のようなダイアログボックスが表示されますので、「オペレーションクラス」欄に先ほど作成したMQTT.BO.StoreDataクラスを選択し、「オペレーション名」欄にオペレーション名（メッセージ保存）、「有効にする」にチェックを入れ、「OK」ボタンをクリックします。

ビジネス・オペレーション・ウィザード

このプロダクションに新しいビジネス・オペレーションを追加します。

すべてのオペレーション

HL7出力

X12出力

ワークフロー

オペレーションクラス

MQTT.BO.StoreData

オペレーション名

メッセージ保存

カテゴリを表示

コメント

有効にする

☒

このフォームを使用して、新しいビジネス・オペレーションをプロダクションに追加します。

このフォーム内では、設定名の上にカーソルを置くことで設定のヘルプを表示します。

キャンセル

OK

ビジネスサービスの作成

MQTTブローカーからメッセージを受信するMQTTサービスを作成します。画面左上の「サービス」の右にある「+」ボタンをクリックしますと、以下のようなダイアログボックスが表示されます。

teroperability > プロダクション

プロダクションが停止しました

サービス

+

ビジネス・サービス・ウィザード

このプロダクションに新しいビジネス・サービスを追加します。

すべてのサービス

HL7入力

X12入力

ビジネス・メトリック

サービスクラス

EnsLib.MQTT.Service.Passthrough

サービス名

MQTTサービス

カテゴリを表示

コメント

有効にする

☒

このフォームを使用して、新しいビジネス・サービスをプロダクションに追加します。

このフォーム内では、設定名の上にカーソルを置くことで設定のヘルプを表示します。

キャンセル

OK

ここで「サービスクラス」欄にて「EnsLib.MQTT.Service.Passthrough」を選択します。

「サービス名」欄にサービス名（MQTTサービス）を入力し、「有効にする」をチェック、「OK」ボタンをクリックします。

MQTTに接続するための設定

[最初の記事](#)

にて設定したMQTTブローカー（mosquitto）にアクセスするには認証や暗号化通信が必要です。認証に必要な認証情報は以下の手順で作成します。

1. 管理ポータルより「Interoperability」「構成」「認証情報」メニューをクリックします。
2. 以下の画面が表示されますので、「ID」欄にMQTTAccess、「ユーザ名」欄に「iris」、「パスワード」欄に最初の記事で入力したユーザirisのパスワードを入力し、保存ボタンをクリックします。



認証の次は暗号化です。暗号化通信を行うにはCAの証明書が必要になりますので、以下の手順でSSL/TLS構成を作成します。

1. 管理ポータルより「システム管理」「セキュリティ」「SSL/TLS構成」メニューをクリックします。
2. 以下のようにSSL/TLS構成一覧が表示されますので、画面上の「新規構成の作成」ボタンをクリックします。

システム > セキュリティ管理 > SSL/TLS 構成

SSL/TLS 構成

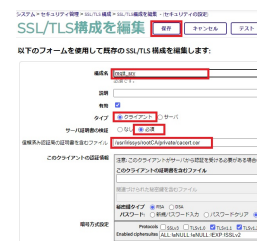
新規構成の作成

以下がSSL/TLS 構成のリストです:

ページサイズ: 最大行数: 結果: 3 ページ: [◀](#) [◀◀](#) **1** [▶▶](#) [▶](#) の 1

名前	説明	有効	タイプ		
ISC.FeatureTracker.SSL.Config		はい	クライアント	編集	削除

3. SSL/TLS構成編集画面にて以下のように構成名（mqttsrv）、タイプ(クライアントをチェック)、サーバ証明書の検証(必須をチェック)を入力し、「信頼済み認証局の証明書を含むファイル」欄に最初の記事の中で作成した自己認証局の証明書のファイルを設定、「保存」ボタンをクリックします。



サービスの設定

前のトピックで作成した認証情報やCAの証明書を使い、MQTTサービスの設定を行います。

以下のようなプロダクション構成画面にて「MQTTサービス」をクリックし、右にある「設定」をクリックします。

ここで以下の項目を設定し、画面上の「適用」をクリックします。

- 基本設定 - ターゲット構成：「メッセージ保存」
- MQTT - ClientID：「MQTTSRV1」
- MQTT - CredentialsName：「MQTTAccess」
- MQTT - SSLConfigName：「mqttsrv」
- MQTT - Topic：「point/#」
- MQTT - Url：「ssl://localhost:8883」

プロダクションの起動

以上で設定は完了しましたので、プロダクション構成画面から「開始する」ボタンをクリックします。

Interoperability > プロダクション構成 - (MQTT.MQTTProduction)



管理ポータルより、「システムエクスプローラ」「SQL」メニューをクリックし、「クエリ実行」タブをクリック、以下のSQLを実行させて、1分おきにレコードが増えていれば成功です。

```
select * from MQTT_Data.WeatherInfo
```

ID	Humidity	MeasureTime	PointName	Pressure	Temperature
1692	86.00	2020-12-16 06:42:06	KFHKXHWD	995.84	13.10
1693	86.00	2020-12-16 06:43:07	KFHKXHWD	995.85	1.40
1694	61.00	2020-12-16 06:44:08	KFHKXHWD	996.06	4.30
1695	57.00	2020-12-16 06:45:09	KFHKXHWD	996.22	2.30
1696	60.00	2020-12-16 06:46:09	KFHKXHWD	996.25	0.80
1697	63.00	2020-12-16 06:47:10	KFHKXHWD	996.28	12.90
1698	65.00	2020-12-16 06:48:11	KFHKXHWD	996.25	13.30
1699	67.00	2020-12-16 06:49:12	KFHKXHWD	996.29	13.70
1700	68.00	2020-12-16 06:50:13	KFHKXHWD	996.29	13.90
1701	69.00	2020-12-16 06:51:14	KFHKXHWD	996.23	14.10
1702	70.00	2020-12-16 06:52:14	KFHKXHWD	996.36	14.10
1703	71.00	2020-12-16 06:53:15	KFHKXHWD	996.22	14.20
1704	71.00	2020-12-16 06:54:16	KFHKXHWD	996.43	14.30
1705	71.00	2020-12-16 06:55:17	KFHKXHWD	996.37	14.40

まとめ

3回にわたって、Mosquittoの設定やESP8266 (arduino)などのマイコンのプログラミング、IRISのプログラム例を紹介しました。
このことから、マイコンからInterSystems IRISにデータが渡せることをご理解いただけたかと思います。
また、IRISの公開鍵基盤を使用して証明書等作成できることもご確認いただけたかと思います。

ご意見、ご質問等ございましたら、ご遠慮なくコメントに追加いただければと思います。
最後までお読みいただきありがとうございました。

[#InterSystems IRIS](#)

ソースURL:

<https://jp.community.intersystems.com/post/iris%E3%81%AB%E3%81%A6mqtt%E3%83%96%E3%83%AD%E3%83%BC%E3%82%AB%E3%83%BC%E3%81%8B%E3%82%89%E6%B0%97%E8%B1%A1%E3%83%87%E3%8>

[3%BC%E3%82%BF%E3%82%92%E5%8F%96%E5%BE%97%E3%81%97%E3%83%87%E3%83%BC%E3%82%BF%E3%83%99%E3%83%BC%E3%82%B9%E3%81%AB%E6%A0%BC%E7%B4%8D%E3%81%99%E3%82%8B](#)