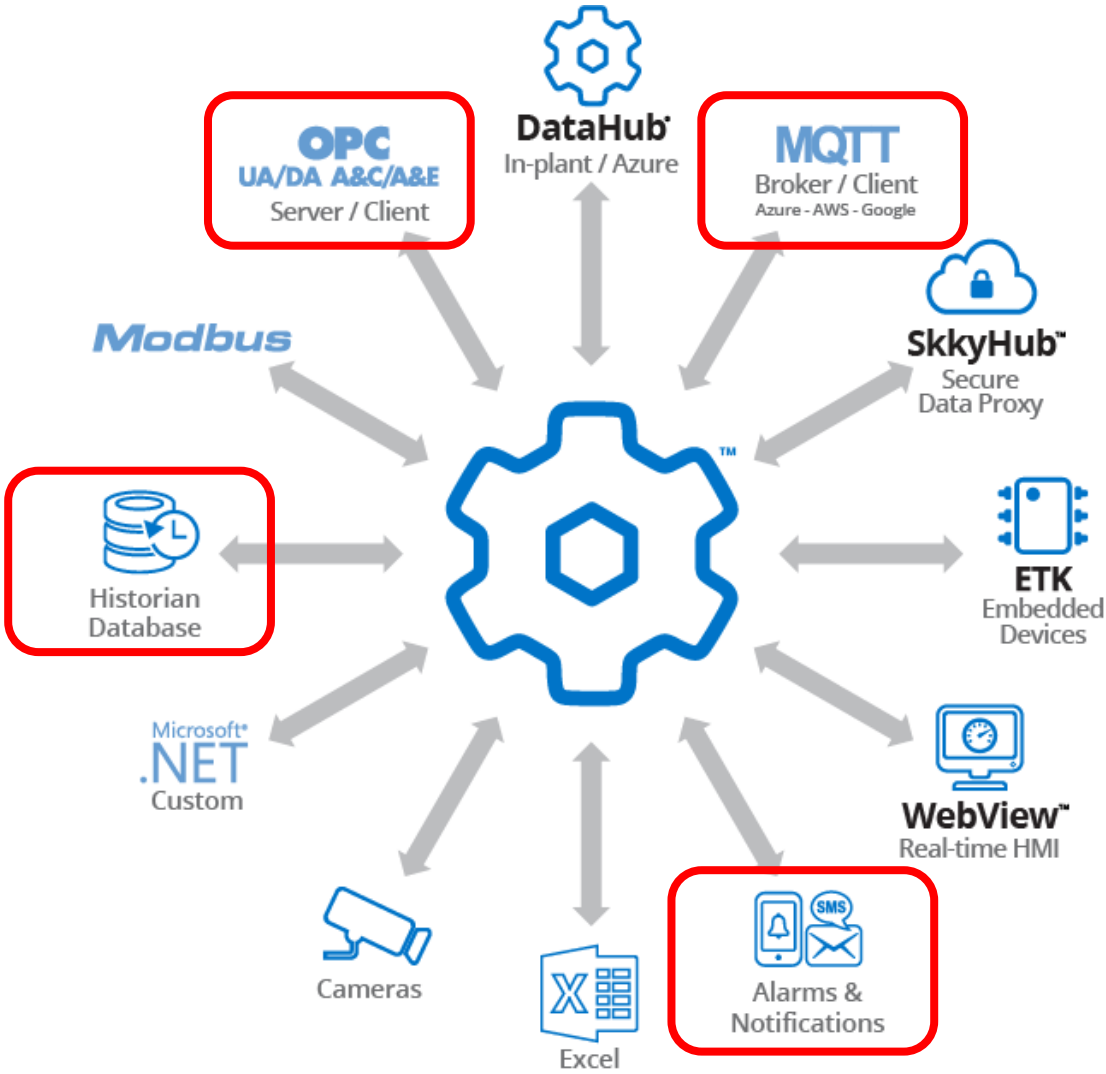
Cogent DataHub® v10 ガイド

Rev 1.0.0

Cogent DataHub V10.0.2



ドキュメントNo.：DH-M0101

作成日：2023/4/7

はじめに

カナダのCogent Real-Time Systems社のリアルタイムデータソリューションである Cogent DataHub®（以下 Cogent DataHub もしくは DataHub と記載）が新しくバージョンアップされ、Cogent DataHub® バージョン10（以下 V10 と記載）がリリースされました。

この『Cogent DataHub® v10 ガイド』は、今回リリースされるCogent DataHub® v10 の新機能の解説書です。

Cogent DataHubは、様々なデータソースのデータを統合し、安全でリアルタイムにデータ活用可能な産業オートメーションのためのミドルウェアです。V10では、バージョン9（以下 V9 と記載）の既存の機能を改善し、一般的な要求に対処、さらにIoTを拡張させるためのいくつかのエキサイティングな機能を追加しました。

- **外部ヒストリアン**（InfluxDB、InfluxDB Cloud、Amazon Kinesis、AVEVA Insight、AVEVA Historian、OSIsoft PI、REST Client）
- **通知（Notification）**
- **OPC UA A&C**
- **構成一括インポート**
- **MQTTの強化**（MQTT Sparkplug B、MQTT Advanced Parser）

従来の機能に関しては、『Cogent DataHub® 入門ガイド』に記載されておりますので、一緒にご活用いただけると幸いに存じます。

全ての仕様、取り扱い方法は Cogent Real-Time Systems 社により開示されているマニュアル（英文）が本書に優先されます。マニュアルは、巻末の付録「資料情報」にURLを記載しております。

2023年4月

株式会社ベルチャイルド

目　次

Cogent DataHub® V10

Cogent DataHub V10 の機能 と サービス

Cogent DataHub では、産業オートメーションや産業用IoTのための多彩な機能とサービスを提供しています。★印は、V10より新しく追加された機能です。

Cogent DataHub コア機能

- **Data Aggregation**（データアグリケーション）：複数のソースからデータを共通のデータセットにマージします。
- **QucikTrend**：選択したデータのリアルタイムトレンドを表示
- **スクリプト**：特定のニーズを満たすカスタムソリューションをプログラム
- **セキュリティ**：アクセスを制御し、ユーザとグループに権限を設定
- **Remote Config**（リモート設定）：DataHubをリモートから、またはサービスとして実行中に設定
- **ライセンス**：DataHubのライセンスの登録と管理

Cogent DataHub 機能

- **OPC UA**：OPC UAのサーバとクライアントをサポート
- **OPC DA**：OPC DAのサーバとクライアントをサポート
- **OPC A&E**：OPC A&Eのサーバとクライアントをサポート
- **★OPC A&C**：OPC A&Cのサーバとクライアントをサポート
- **トンネル/ミラーリング**：ネットワークを介してOPC DA,Modbus,DDEを接続
- **ブリッジ**：OPCサーバとOPCサーバ間のデータ交換
- **Redundancy**（リダンダンシー）：ホットスタンバイによる冗長性をサポート
- **データロギング（書き込み、読み込み）**：ODBC対応データベースへの書き込みと読み込みをサポート
- **Webサーバ**：DataHubに内蔵されたWebサーバによりWebページにライブデータを表示
- **WebView**：高品質なリアルタイムなHMIの作成と表示。
- **MQTTクライアント**：MQTTブローカーに接続
- **MQTTブローカー**：MQTTクライアントに接続
- **Modbus**：Modbus TCPスレーブデバイスに接続
- **★InfluxDB**：InfluxDBサーバの起動や設定、GrafanaやChronofrafの起動や設定、接続
- **★External(外部)ヒストリアン**：他社製品のヒストリアンに接続
  - **★InfluxDB、InfluxDB Cloud**

- ★**Amazon Kinesis**
- ★**AVEVA Insight、AVEVA Histrian**
- ★**OSIsoft PI**
- ★**REST API**

・**ヒストリアン**：高速で大量なデータの保存と取得
・**Eメール/SMS**：データ変更に基づいてEメールまたはSNSメッセージの送信
・**カメラ**：システムのライブビデオや画像をストリーミング。USBカメラと **IPカメラ** サポート
・**Sysモニタ**（システムモニタ）：ネットワークコンピュータのシステム状態を監視
・**DDE**：リアルタイムデータをドラッグ＆ドロップでMicrosoft Excelに表示

その他

・**Vine Excel Add-in**

Excelからネットワーク経由でCogent DataHubやSkkyHub、Skkynet DataHub Serviceへ接続し、データの双方向通信を可能にするMicrosoft ExcelのためのExcelアドインツールです。リアルタイムなデータ分析と複数のExcelユーザー間で分析データの共有を可能にします。

・**DataHub API**

DataHubにカスタムプログラムを接続するためのAPI（Application Programming Interface）を提供しています。DataHubへのデータ書き込み、読み込みを可能にするシンプルな一連のコマンドにより構成されています。オブジェクトコードとソースコードの両方のサンプルを提供しています。

・**SkkyHub DataHub Service**

Skkynet DataHub Service は、Microsoft Azure のマネージドアプリケーションとして提供される、Cogent DataHubソフトウェアをベースとした産業用IoTサービスです。マシンやアプリケーションや組み込みシステム間において、安全なリアルタイムデータ接続を確立します。

Cogent DataHub V10 ライセンス

Cogent DataHub のライセンスは、使用する機能に応じて価格が設定されています。以下の表に示すように、ライセンスパックおよびアドオン により購入が可能です。
Cogent社またはSkkynet社パートナー がお客様のニーズに合った最適なライセンスの組み合わせをご提案いたします。またこちらから お見積りを依頼する ことができます。
★印は、V10より新しく追加されたライセンスです。

ライセンスパック：DataHubのコア機能が含まれるライセンスのベースとなるパッケージ

製品名	製品コード	含まれるDataHub機能
Cogent DataHub	COGDH	全てのCogent DataHub機能
DataHub IOT Gateway	IOTGAT	OPC UA, OPC DA, MQTTクライアント, トンネル/ミラー, MQTT Link License×1, API support
DataHub OPC Gateway	DHGAT	OPC UA, OPC DA
DataHub UA Tunneller	DHTUNUA	OPC UA, OPC DA, トンネル/ミラー, API support
DataHub UA Logger	DHLOGUA	OPC UA, データロギング
DataHub UA Bridge	DHBRGUA	OPC UA, ブリッジ
DataHub Modbus UA Server	DHMBUA	OPC UA, ModbusTCP
DataHub UA WebView	DHWVUA	OPC UA, トンネル/ミラー, WebView, Webサーバ, ヒストリアン, API support, ★InfluxDB
OPC UA DataHub	OPCDHUA	OPC UA, トンネル/ミラー, ブリッジ, データロギング, *DDE*, システムモニタ, Email/SMS Notifications, API support
DataHub DA Tunneller	DHTUN	OPC DA, トンネル/ミラー, API support
DataHub DA Logger	DHLOG	OPC DA, データロギング
DataHub DA Bridge	DHBRG	OPC DA, ブリッジ
DataHub Modbus DA Server	DHMB	OPC DA, ModbusTCP
DataHub DA WebView	DHWV	OPC DA, トンネル/ミラー, WebView, Webサーバ, ヒストリアン, API support, ★InfluxDB
OPC DA DataHub	OPCDH	OPC DA, トンネル/ミラー, ブリッジ, データロギング, DDE, システムモニタ, Email/SMS Notifications, API support
DataHub DDE Tunneller	DHDTUN	トンネル/ミラー, DDE, API support
DataHub Modbus Tunneller	DHMTUN	ModbusTCP, トンネル/ミラー, API support
DataHub Modbus Logger	DHMLOG	ModbusTCP, データロギング
DataHub System Monitor	DHSYS	システムモニタ, トンネル/ミラー, API support
Skkynet Vine Server	SKVINE	トンネル/ミラー, DDE, API support

★DataHub Smart Broker	DHMQB	MQTT Broker, API support,トンネル/ミラー

※上記ライセンスパックには、DataHubのコア機能の **アグリゲーション**、**QuickTrend**、**スクリプト**、**セキュリティ**、**リモート設定** が含まれます。

アドオン ライセンス：ライセンスパックだけでは足りない機能を選択して追加します。

Tunnel/SkkyHub/TCP support	ADDTUN	トんネル/ミラー, API support
OPC DA support	ADDDA	OPC DA（サーバ，クライアント）
OPC UA support	ADDUA	OPC UA（サーバ，クライアント）
★OPC UA A&C support	ADDAC	OPC DA A&C（サーバ，クライアント）
OPC A&E support	ADDAE	OPC DA A&E（サーバ，クライアント）
★OPC A&E + A&C support	ADDAEC	OPC DA A&E（サーバ，クライアント） 、OPC DA A&C（サーバ，クライアント）
MQTT Client	ADDMQC	MQTT クライアント, MQTT Link License×1
MQTT Broker	ADDMQB	MQTT ブローカー, MQTT Link License×1
★AVEVA Insight/Historian	ADDAHI	AVEVA Historian/Insight
★OSIsoft PI Historian	ADDPI	OSIsoft PI Historian
★REST Historian	ADDRST	REST Historian
★InfluxDB support	ADDIFX	InfluxDB
★Alarm & Notification	ADDNOT	通知
Modbus TCP	ADDMB	ModbusTCP（マスタ）
Data Bridging	ADDBRG	ブリッジ
Data Logging	ADDLOG	データロギング（書き込み，読み取り）
Excel/DDE support	ADDDDE	DDE（サーバ，クライアント）
System Monitor	ADDSYS	システムモニタ
Email/SMS Notifications	ADDEML	Email/SMS
Data Redundancy	ADDRED	リダンダンシー
Data Historian	ADDHIS	ヒストリアン
DataHub Web Server	ADDWEB	Webサーバ
DataHub WebView (Add-on)	ADDWV	トンネル/ミラー, WebView, Webサーバ, ヒストリアン, ★InfluxDB
Camera support	ADDCAM	カメラ（USB, IPカメラ）

コネクション ライセンス

TCP Link License x 1	DHTCP	DataHub API, ETKなどの コネクションライセンス
MQTT Client Link License	MQCLNK	MQTT Client とのコネクションライセンス
MQTT Broker Link License	MQBLNK	MQTT Broker とのコネクションライセンス
Skkynet Vine Link License	SKVLNK	Vine Add-in とのコネクションライセンス

1．Cogent DataHub V -10 インストール

1．1．システム要件とインストール

システム要件

Cogent DataHubV10は、下記のオペレーティング・システムをサポートします。

オペレーティング・システム	最小要件
Windows Server 2008＆R2	Service Pack 1（SP1）+ .NET 4.6.1
Windows 7	Service Pack 1（SP1）+ .NET 4.6.1
Windows Server 2012＆R2	.NET 4.6.1
Windows 8.1	.NET 4.6.1
Windows 10	.NET 4.6.1
Windows Server 2016	.NET 4.6.1
Windows Server 2019	.NET 4.6.1
Windows 11	.NET 4.6.1
Wondows Server 2022	.NET 4.6.1

※最小要件以上のバージョンに対応しております。

Cogent DataHub V10ソフトウェアの取得

下記サイトからCogent DataHub V10ソフトウェアをダウンロード申請ができます。

こちら（メーカのダウンロードサイト）

1. 申請情報を入力後、『Send me the download link』 ボタンをクリックします。
2. すぐにSkkynet社よりダウンロード情報がメールで送られてきます。
3. メールにCogent社のダウンロードサイトへのリンクが添付されていますので、
   Cogent DataHub、バージョン：10.0.x 、Windows をダウンロードします。

インストール

DataHubのインストール手順を下記に記載します。

1. Cogent社のWebサイトからダウンロードしたDataHubのインストーラをダブルクリックします。DataHub-10.0-xxxxxx-Windows.exe
2. 後は、メッセージに従ってインストールを完了させます。

アンインストール

1. スタートメニューからコントロールパネルを選択し、プログラムの追加と削除を選択

します。

2. このリストからCogent DataHubをダブルクリックします。
3. 削除ボタンをクリックして、メッセージに従います。

1．2．以前のバージョンの上にバージョン 10.x をインストールする

詳しくは下記のCogent DataHub Webサイトをご参照ください。

こちら（メーカー英文サイト）

以前のバージョンを実行しているシステムへのDataHub V10をインストール

- DataHub V7またはV8を実行しているシステムにDataHub V10をインストールすると、インストーラは、以前のバージョンのDataHubをアンインストールするように要求します。DataHubV10がインストールされ、以前のコンフィグレーション・ファイルの設定が読み込まれます。
- OPC DataHub V6.4を実行しているシステムにDataHub V10をインストールすると、以前のバージョンから構成をコピーするかどうか尋ねられますが、これは以前のバージョンを置き換えるということではありません。同じシステム上でOPC DataHub V6.4とCogent DataHubを実行することができますが、構成の変更を必要とする場合があります。
- 現在のDataHubの設定のバックアップを作成したい場合は、DataHubコンフィグレーション・フォルダの内容をコピーしてください。このフォルダの場所は、実行しているオペレーティング・システムによって異なります。

  コンフィグレーションおよびライセンスファイルの格納場所

  DataHubのコンフィグレーションのバックアップを作成する場合は、DataHubコンフィグレーションディレクトリの内容をコピーすることでバックアップができます。このディレクトリの場所は、実行しているオペレーティングシステムによって異なります。詳細については、このリンクをご参照ください。

DataHub V10とV9、V8、V7またはV6.4の間をトンネリングする

- DataHubソフトウェアV10とV9、V8、V7、V6.4の間をトンネリングで接続することが可能です。V10とV6.4の間をトンネリングすると、DataHubのネットワーク接続が単純なスター構成ではない場合にライセンスに関する既知の問題が発生する可能性があります。
- WebSocketを使用してV10、V9、V8およびV7の間をトンネリングすることが可能ですが、V10とV6.4の間は、非SSL接続を使用のみトンネリングすることが可能です。
- V9/V10がクライアント接続である限り、V10とV9、V8、V7間のプロキシを介してトンネリング接続が可能です。

- DataHub V9/V10のトンネリングに関する既知の問題が一つあります。DataHub V10とV9間のバイナリトンネル接続は成功しますが、V9/V10とそれ以前のバージョン間のバイナリトンネル接続は失敗します。これらは成功したように見えますが、データ値は配信されません。DataHubのイベントログには、接続の方向によってはエラーメッセージが記録される場合があります。これは、タイムスタンプの表現が32ビットから64ビットに変更されたことにより必要となったものです。テキスト接続（非バイナリ）は、V10とそれ以前のバージョン間で互換性があります。

WebViewページとメディアをDataHub V10に移動する

- V10 にアップグレードすると、DataHub インストーラは古い WebView ページを新しいフォルダにコピーし、それらを WebView V10 で表示できるようにします。これを手動で行う場合は、以下の手順を実行してください。
- ユーザによって生成されたコンテンツは、Program Files フォルダでは無く、DataHub コンフィグレーション・フォルダに保存されるようになりました。

デフォルト設定フォルダ

デフォルトのコンフィグレーション・フォルダは次のとおりです。

C : ¥ Users ¥ < Windows ログイン > ¥ AppData ¥ Roaming ¥ Cogent DataHub

ここでの< Windows ログイン >は、Windows へのログインに使用したユーザ名です。コンフィグレーション・フォルダは、DataHub の-H コマンドラインオプションを使用するか、DataHub Service Manager アプリケーションから設定することで変更することが可能です。

個人用ファイル

- admin ユーザーとして WebView にログインして、WebView V8 で作成したページファイルは、下記に保存されています。

  C : ¥Program Files（x86）¥Cogent¥Cogent DataHub¥Plugin¥WebServer¥html¥Silverlight¥Pages¥Users¥admin

  これらのページを WebView V9/10 で使用するには、次の場所にコピーする必要があります。

  < Configuration Folder > ¥ WebContent ¥ Content ¥ Organizations ¥ Local ¥ Users ¥ admin ¥ Pages
- WebView V8 で追加した個人用画像は、下記に保存されています。

  C : ¥Program Files（x86）¥Cogent¥Cogent DataHub¥Plugin¥WebServer¥html¥Silverlight¥Images

  これらの画像を WebView V9/V10 で使用するには、次の場所にコピーする必要があります。

< Configuration Folder > ¥ WebContent ¥ Content ¥ Organizations ¥ Local ¥ Users ¥ admin ¥ Images

- メディアファイルとスクリプトファイルについても同じことを行う必要があります。

公開ファイル

- WebView V8 で作成した、公にアクセス可能なページや画像などを作成したファイルは、下記に保存されています。
  C：¥Program Files（x86）¥Cogent¥Cogent DataHub¥Plugin¥WebServer¥html¥Silverlight¥Pages
  これらのファイルは、次の場所にコピーする必要があります。
  < Configuration Folder > ¥ WebContent ¥ Content ¥ Organizations ¥ Local ¥ Pages
- 画像、メディアファイルおよびスクリプトの同様にコピーする必要があります。

カスタムコントロール

- カスタム WebView コントロールを作成している場合は、Silverlight でしか機能しません。デスクトップ WebView 用にこれらのコントロールを別々に再コンパイルする必要があります。Controls と ControlAssemble は、Silverlight バージョンと WPF バージョンで区別されるようになりました。例えば、MyCustomControl.dll というカスタムコントロール DLL を作成した場合は、DataHub V8 の次の場所にインストールします。

C：¥Program Files（x86）¥Cogent¥Cogent DataHub¥Plugin¥WebServer¥html¥Silverlight¥ControlAssemblies¥Company¥MyCustomControl.dll

- DataHub V9/V10 では、以下のいずれかの場所にインストールしてください。

C：¥Program Files¥Cogent¥Cogent DataHub¥Plugin¥WebServer¥html¥Content¥Common¥ControlAssemblies¥Company¥Silverlight¥MyCustomControl.dll　または、
< Configuration Folder > ¥ WebContent ¥ Content ¥ Organizations ¥ Local ¥ ControlAssemblies ¥ Company ¥ Silverlight ¥ MyCustomControl.dll

- Silverlight バージョンと WPF バージョンで異なる場合は、カスタムコントロールの XAML ファイルと XML ファイルを同様に区別する必要があります。

1．3．以前のバージョンへのダウングレード

バージョン 10 からバージョン 9 以前へのダウングレード

システムに DataHub ソフトウェア V10 をインストールしてから、以前のバージョンに戻す場合は、DataHubV10 ソフトウェアをアンインストールしてから、以前のバージョンをインストールしてください。予期しない動作が発生する場合があります。DataHub ソフトウェア V9.0.11（およびそれ以降）のインストーラは、自動的に V10 ソフトウェアをアンインストールします。

バージョン 9 または 10 をバージョン 8 以前にダウングレードする

DataHub V9 または V10 を最初に実行すると、以前のバージョンから構成フォルダのコピーが作成され、V9 または V10 と互換性があるように構成がアップグレードされます。以前の設定は同じフォルダに #.bak の拡張子で保存されます。＃は、シーケンス番号です。

例えば、DataHub のコンフィグレーション・フォルダが以下の場合

C:¥Users¥MyName¥AppData¥Roaming¥Cogent DataHub

コンフィグレーションバックアップフォルダは次のようになります。

C:¥Users¥MyName¥AppData¥Roaming¥Cogent DataHub.1.bak

以前のバージョンの DataHub の設定に戻す場合は、このバックアップフォルダを元の名前に手動で復元します。手順は下記のとおりです。

1. DataHub と WebView、リモートコンフィグ、DataPid などすべてのツールを停止します。
2. DataHub バージョン 9 または 10 をアンインストールします。
3. 古いバージョンの DataHub をインストールしますが、まだは起動しないでください。
4. Windows のファイルエクスプローラを使用して、今のコンフィグレーション・フォルダの名前を変更するか、削除します。

   C:¥Users¥MyName¥AppData¥Roaming¥Cogent DataHub
5. Windows ファイルエクスプローラを使用して、バックアップフォルダをコンフィギュレーションフォルダ名に変更します。

   例えば、下記のコンフィグレーションバックアップフォルダを

   C:¥Users¥MyName¥AppData¥Roaming¥Cogent DataHub.1.bak

   コンフィグレーション・フォルダ名に変更

   C:¥Users¥MyName¥AppData¥Roaming¥Cogent DataHub
6. DataHub を起動します。

1．4．コンフィグレーション・ファイル

DataHub の設定は、コンフィグレーション・ファイルに記載され保存されています。

こちらをご参照ください。こちら（メーカー英文マニュアル）

1．5．DataHub インストールのバックアップまたは移動

アーカイブ目的または別のマシンに移動するために DataHub のインストールをバックアップすることが可能です。

注意点

Cogent DataHub のエンドユーザ使用承諾契約書では、一度に複数のマシンに DataHub ライセンスをインストールすることを禁止しています。そのため、DataHub をあるマシンから別のマシンに移動する場合は、最初にそのマシンで実行している DataHub からライセンスファイルを削除してから次のマシンにインストールしてください。

DataHub ファイルをバックアップするのには、下記フォルダにすべてをコピーする必要があります。

・DataHub バージョン 9 または 10 の場合：

C:¥Users¥MyName¥AppData¥Roaming¥Cogent DataHub¥

すべてのカスタムスクリプトと設定ファイルは、最初のマシンのフォルダから次のマシンの同じフォルダにコピーする必要があります。

・Windows Vista、Windows 7、Windows 8、Windows 10、Windows Server 2008、Windows Server 2012、Windows 2016 で実行されている DataHub バージョン 7 および 8 の場合：

C:¥Users¥MyName¥AppData¥Roaming¥Cogent DataHub¥

C:¥Program Files (x86)¥Cogent¥Cogent DataHub¥scripts¥

C:¥Program Files (x86)¥Cogent¥Cogent DataHub¥Plugin¥WebServer¥html¥

・Windows XP および Windows Server 2003 で稼働している DataHub バージョン 7 および 8 に場合：

C:¥Documents and Settings¥User Name¥Application Data¥Cogent DataHub¥

C:¥Program Files¥Cogent¥Cogent DataHub¥scripts¥

C:¥Program Files¥Cogent¥Cogent DataHub¥Plugin¥WebServer¥html¥

これに加えて、カスタム・コンフィグレーション・ファイルを作成した場合は、それらも 1 台目のマシンのフォルダから 2 台目のマシンの同一のフォルダにコピーする必要があります。

1．6．既知の問題点

CogentDataHub Web サイトの既知の問題と重大な変更をご参照ください。

1．7．よくある質問

DataHub ソフトウェアの使用に関して、お客様より良くいただく質問をまとめました。以下のリンクからご覧いただけます。

よくある質問

2. Cogent DataHub V10 - 新機能

2．1．外部ヒストリアン

概　要

外部ヒストリアンオプションを使用すると、以下のヒストリアンへの接続を構成することが可能です。

- InfluxDB
- AVEVA Historian、AVEVA Insight
- Amazon Kinesis
- OSISoft PI System
- REST API

マニュアル：こちら（メーカーマニュアル英文）

2．1．1．典型的なシナリオ

1．ローカル接続
2．ネットワーク接続
3．ストア＆フォワードのためのローカルおよびネットワーク接続
4．DMZ を介した分離ネットワークからのストア＆フォワード

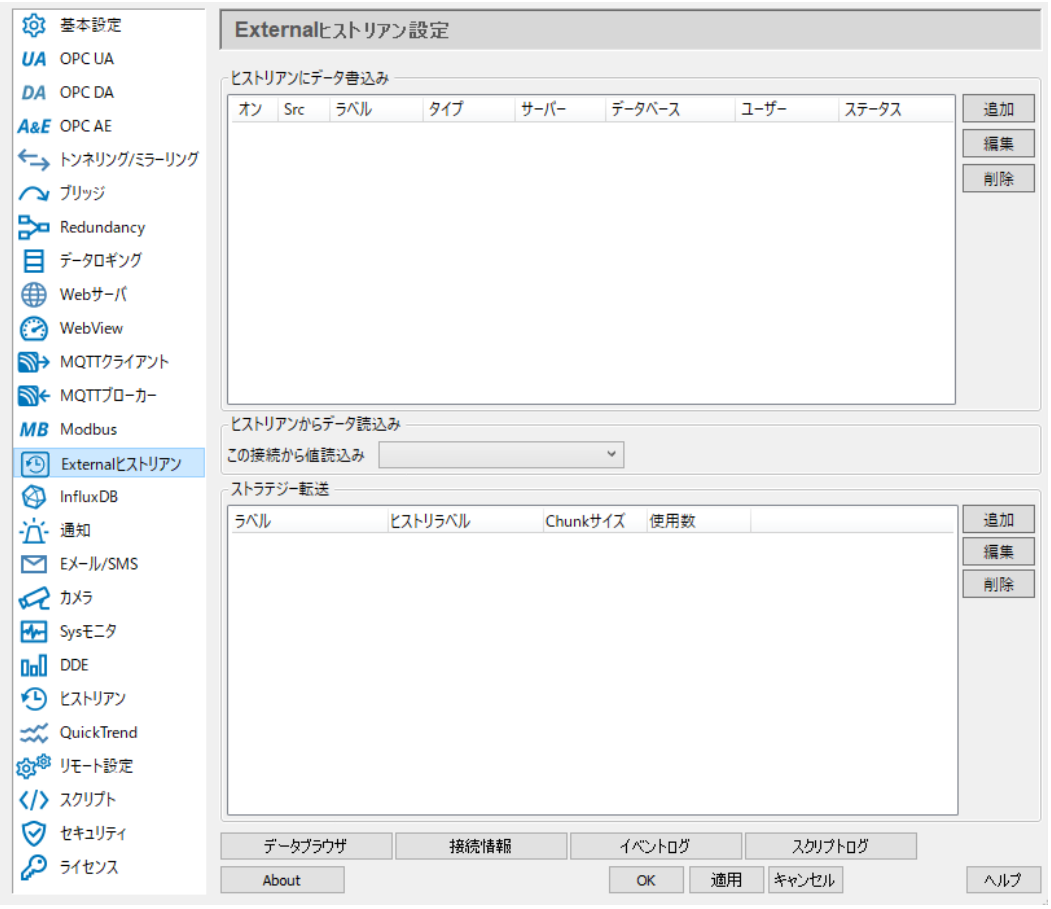
詳しくは、こちら（メーカー マニュアル英文）をご参照ください。

2．1．2．一般的な設定

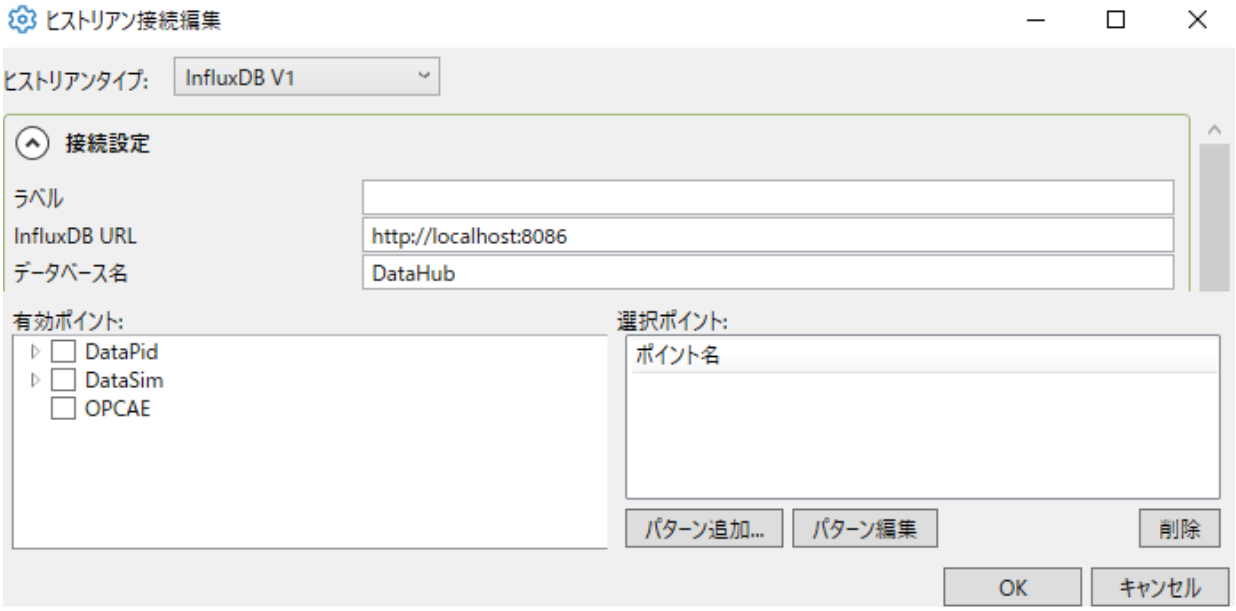
ここでは、全ヒストリアン共通の一般的な設定について記載します。

2．1．2．1．ヒストリアンにデータを書き込む

Cogent DataHub プロパティ画面から、『 External ヒストリアン 』オプションを選択します。
全ての外部ヒストリアンへの接続構成は、このオプションより設定します。

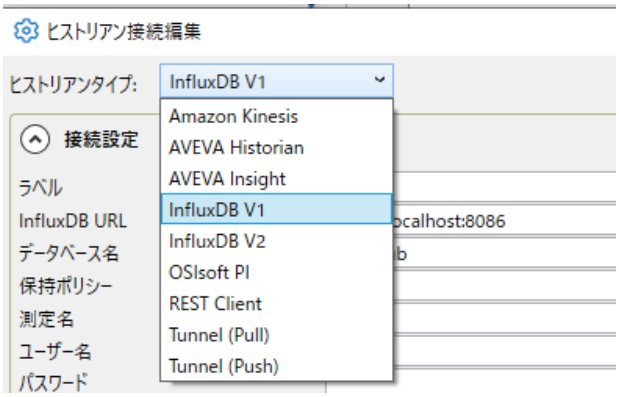


1. ［追加］または［編集］ボタンを押し、ヒストリアン接続編集画面を開きます。
   - ［追加］ボタン：ヒストリアン接続構成の新規作成
   - ［編集］ボタン：ヒストリアン接続構成の編集
   - ［削除］ボタン：ヒストリアン接続構成の削除



2. ［ヒストリアンタイプ］のプルダウンリストより接続するヒストリアンを選択します。各ヒストリアンに対応した接続設定オプションを設定します。現在利用可能なオプションは、次の通りです。

- Amazon Kinesis
- AVEVA Historian
- AVEVA Insight
- InfluxDB V1
- InfluxDB V2
- OSISoft PI
- REST Client
- Tunnel（Pull）
- Tunnel（Push）



設定したいヒストリアンに対応するサブセクションに移動し、設定終了後に、ここに戻って設定を続けます。

2．1．2．2．ポイントの選択（ピッキングポイント）

3. 書き込むポイントを選択する。

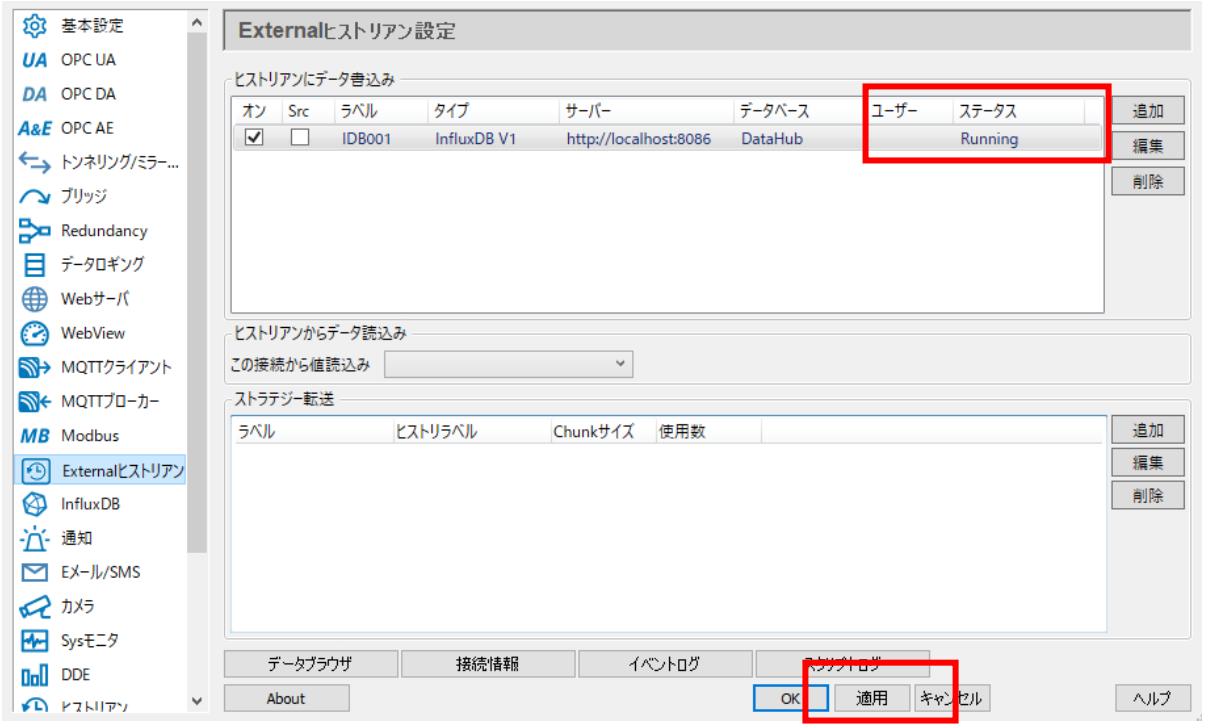
［有効ポイント］からヒストリアンに書き込むポイントを一つずつ選択する。

もしくは、［パターン追加…］ボタンを押して、［パターン入力］画面を開き、.NET 正規表現形式を入力し、共通の特性を持つポイントをグループ選択します。

.NET 正規表現形式：ドキュメントリンク

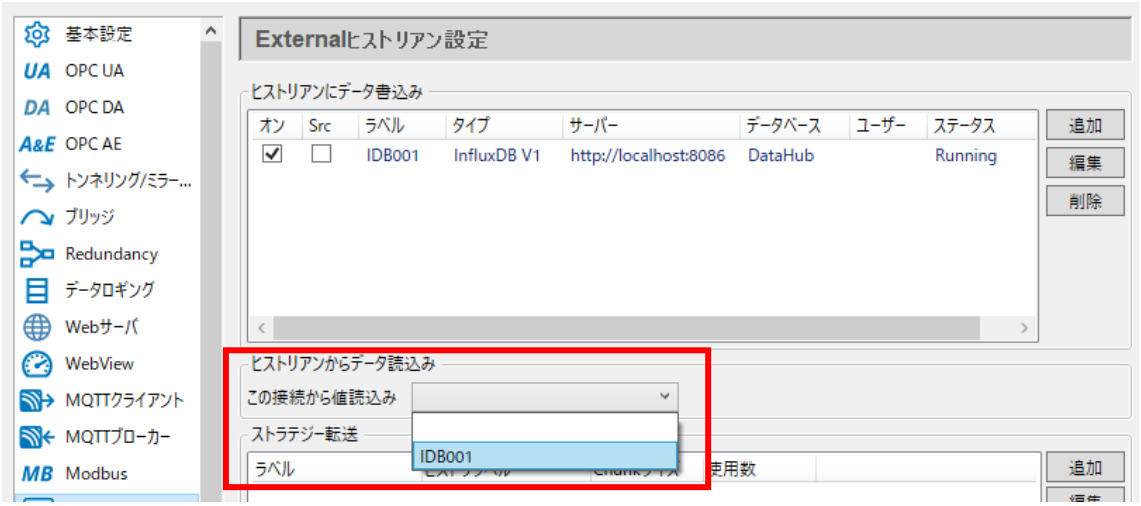
4. ［OK］ボタンを押し、［ヒストリアン接続編集］画面を閉じます。

5. ［適用］ボタンを押し、ステータスが Running になることを確認します。



2．1．2．3．ヒストリアンから履歴データを読み取る

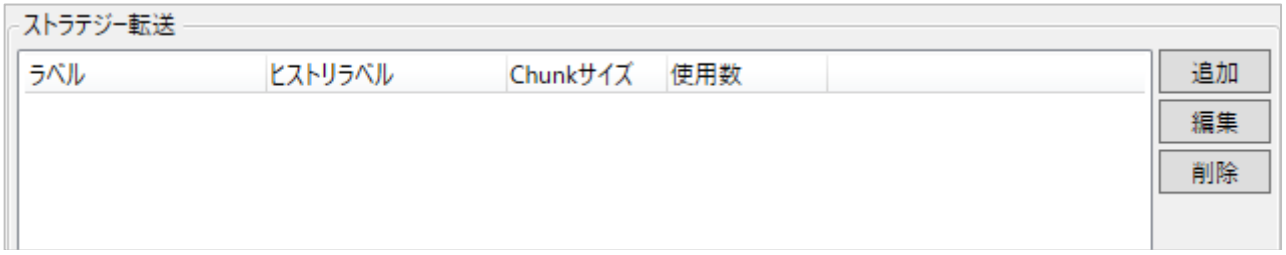
このオプションでは、DataHub クライアントが履歴データを読み出すヒストリアンを選択します。



［この接続から値を読込み］ドロップダウンリストから、DataHub クライアントから読み取る履歴データを選択します。履歴データは、WebView、QuickTrend、OPC UA HAD で使用することができます。

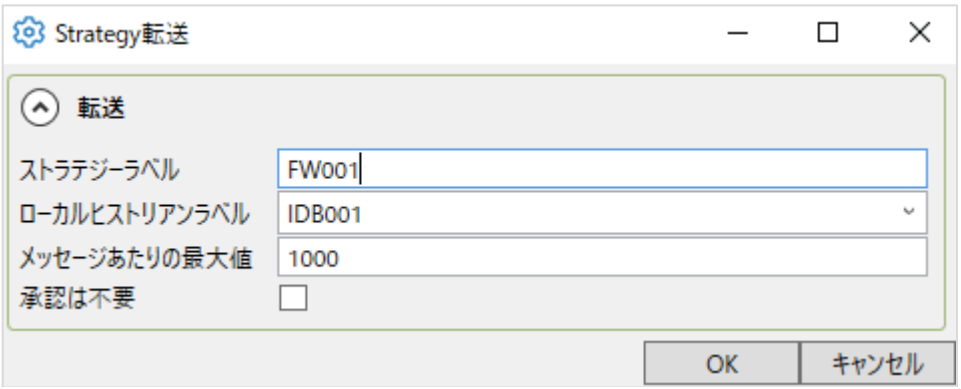
2．1．2．4．ヒストリアンデータを転送する

外部ヒストリアン機能では、データをローカルのヒストリアンに書き込むだけではなく、書き込んだ履歴データを同じヒストリアン製品のリモートインスタンスに転送することができます。



1. ［追加］または［編集］ボタンを押し、Strategy 転送 画面を開きます。
   - ［追加］ボタン：ヒストリアン接続構成の新規作成
   - ［編集］ボタン：ヒストリアン接続構成の編集
   - ［削除］ボタン：ヒストリアン接続構成の削除

2. 転送の設定を行います。



ストラテージラベル：このストラテジーを識別するための名前。文字、数字、アンダースコア（_）のみで構成され、他の文字やスペースは使用できません。

ローカルヒストリアンラベル：転送するヒストリアンを選択します。ヒストリアンのデータ書き込みで割り当てたラベルがプルダウンリストに表示されます。

メッセージあたりの最大値：データは値チャンクでリモートビスとリアンに転送され、１つのメッセージに複数の値が含まれます。ここでは、各メッセージで送信される値の最大数を指定します。

承認は不要：リモートのヒストリアンが受信したメッセージごとに確認応答を送信しないようにする場合は、このボックスをオンにします。

２．１．３．接続設定

サポートされている各ヒストリアン接続には、ポイント名変更、データサンプリングおよび転送オプションが含まれています。

２．１．３．１．ポイント名の変更

このオプションにより、ヒストリアンでポイント名をどのように表現するかを指定することができます。

DataHub インスタンスがヒストリアンに書き込みやデータを照会する際に、まず正規表現による一致と置換を使用してポイント名を変換します。この結果得られた名前が、ヒストリアンにてそのポイントの識別子として使用されます。これらのオプションがポイント名にマッチしない場合、ヒストリアンの識別子は、DataHub エンジンのフルポイント名（ドメイン:プレフィックス）になります。

パターンと置換は、ポイント名の一致に使用される.NET 正規表現とパターンマッチからヒストリアン識別子を生成するための置換指定子です。これらのオプションは、.NET の Regex.Replace フォーマットに従っています。

ポイント名は、ヒストリアンにデータを書き込む時、またはヒストリアンの特定のポイントデータを参照する時に変更されます。ヒストリアンの識別子は、ストア＆フォワードでヒストリアンのデータを読み取る時には変更されません。つまり、ストア＆フォワードでは、ヒストリアン内で検出される識別名が使用されます。

ポイント名パターン：フルポイント名のすべて、もしくは一部が一致する正規表現。
この式には、置換文字列で参照できるキャプチャ式を含めることができます。

ポイント名置き換え：フルポイント名内の ポイント名パターン 使用箇所に置き換える文字列。
パターン内のキャプチャ式は、$記号と数字の組み合わせで参照されます。一致する部分文字列全体は$0、マッチ内の各キャプチャ式は左から$1、$2 などのように番号が付けられます。

例

●ドメイン名の後の最初のポイントパスセグメントを削除する。

パターン： ([^:]+):[^.]*¥.(.*)

置き換え： $1:$2

入力ポイント名：DataPid:PID1.Mv

出力ポイント名：DataPid:Mv

●ドメイン名を削除する。

パターン：[^:]+:(.*)

置き換え： $1

入力ポイント名：DataPid:PID1.Mv

出力ポイント名：PID1.Mv

●ドメイン名を削除する。 (alternate):

パターン：[^:]+:

置き換え： none

入力ポイント名：DataPid:PID1.Mv

出力ポイント名：PID1.Mv

●最初のパスセグメントを取り出し、末尾に配置する。

パターン： ([^:]+:)([^.]*)¥.(.*)

置き換え： $1$3.$2

入力ポイント名：DataPid:PID1.Mv

出力ポイント名：DataPid:Mv.PID1

２．１．３．２．データサンプリング

外部ヒストリアン機能は、フルデータセットではなく、指定した時間間隔でデータサンプルされたデータをサポートしているヒストリアンに送信するように設定することができます。

データサンプリングを有効：データサンプリング機能を有効にします。

サンプリング間隔（ミリ秒）：各サンプルが取得される時間の長さ（ミリ秒単位）。

サンプリングタイプ：

None：サンプリングは行われず、すべての値が送信されます。

First：サンプリング間隔の最初の値が送信されます。

Last：サンプリング間隔の最後の値が送信されます。

Mean：サンプリング間隔内のすべての（Good 品質の）値の平均値が送信されます。

Min：サンプリング間隔内の最小値が送信されます。

Max：サンプリング間隔内の最大値が送信されます・

MinMax：サンプリング間隔の最小値と最大値が時間順に送信されます。

bad データ品質を無視：データ品質が Good ではないデータ変更は無視されます。データ品質が

Good のサンプルは通常どおり処理されます。

タイムスタンプでデータを選別：ドキュメント定義を使用してメッセージ本文を構築する際に、サンプリングされた値を時間順にソートします。サンプリングはポイントごとに行われるため、結果として得られる値の集合は時間順でない可能性があります。例えば、point2 の最大値は point1 の最大値より前に発生した可能性があります。このオプションは、結果として得られる結合された値の集合をソートします。

蓄積時間によるスナップショット：選択されたすべてのデータポイントのスナップショットがヒストリアンに一括送信される前に、周期的に保存されるように設定します。スナップショットを取得する周期は、［接続設定］の［蓄積時間］に設定します。各データポイントは、ソースタイムスタンプと現在値が保存されます。

システムタイムでスナップショット設定：スナップショットを使用してデータを保存する場合、すべてのポイント値のソースタイムスタンプを現在のシステム時刻に置き換えることができます。このオプションは、蓄積時間内で変更されたかどうかにかかわらず、すべてのポイントに適用されます。

2．1．3．3．転送（フォワーディング）

外部ヒストリアン機能は、ローカルでサポートされているヒストリアンにデータを書き込むだけではなく、データを同じサポートされているヒストリアン製品のリモートインスタンスに転送することができます。

ヒストリカルデータ転送を有効：データ転送機能を有効にします。

ストラテジー転送：あらかじめ定義されている 転送ストラテジー を選択します。

固有転送クライアント ID：ヒストリアンが設定を識別するための固有のテキスト文字列。システムで自動的に生成された文字列を使用するか、独自の文字列に変更することができます。

読み込みロケーションの上書き：転送を開始するデータの日時を設定します。開始日時を新しい時間にリセットすることにより、データセットの一部を転送できるようにします。日付/時刻データの形式は、YYYY-MM-DD HH:MM:SS で、必要に応じて編集することができます。カレンダーボタンを使用することで、素早く日付を入力することができます。

2．1．4．サポートしているヒストリアン

External ヒストリアン機能は、以下の接続をサポートしています。

・Amazon Kinesis
・AVEVA Historian
・AVEVA Insight
・InfluxDB V1
・InfluxDB V2
・OSISoft PI
・REST Client
・Tunnel（Pull）
・Tunnel（Push）

2．1．4．1．Amazon Kinesis 接続

概　要

DataHubの外部ヒストリアンを使用して、Amazon Kinesisにリアルタイムにデータをフィードします。

特　徴

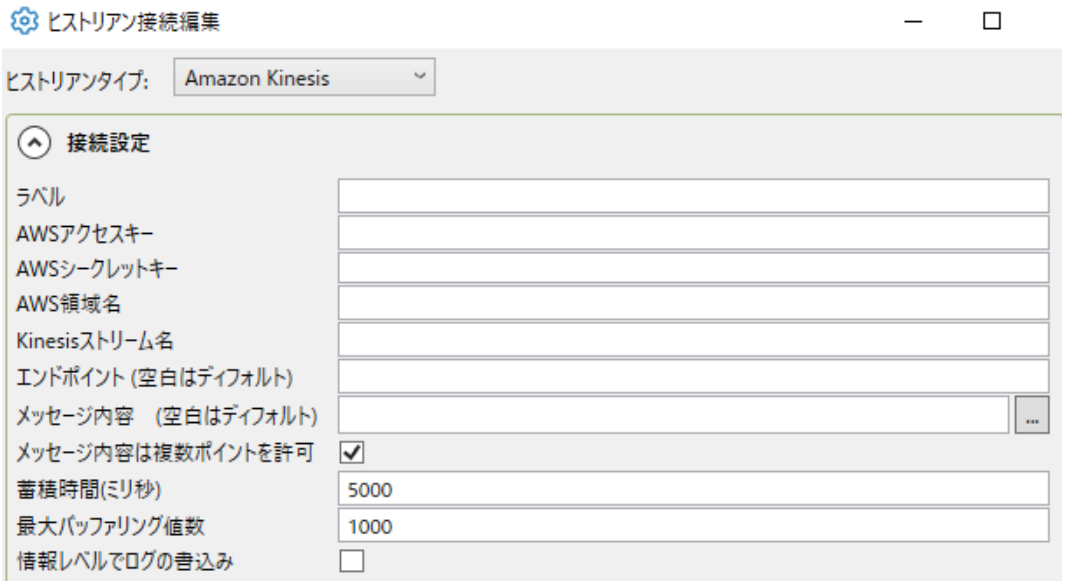
- ミリ秒の時間分解能で Amazon Kinesis Data Streams へ高速データスループット（50,000 書き込み/秒）
- DataHub が収集したデータ（OPC UA および A＆C、OPC DA および A＆E、ModbusTCP スレーブ、ODBC データベース、Excel、カスタムプログラム）の書き込み
- トンネリングと組み合わせる：ネットワーク経由でヒストリアンデータを収集

詳細：こちら（メーカ英文サイト）
マニュアル：こちら（メーカ英文マニュアル）

1.　Amazon Kinesis

ヒストリアン接続編集画面から、Amozon Kinesis を選択できます。

①　接続設定（Connection Settings）



ラベル：接続を識別するために使用される固有のテキスト文字列。

このラベルは、ストア＆フォワードを設定する際に接続を識別するために使用され、一部のヒストリアンでデータベース名または保持ポリシーを作成するために使用されます。ラベルには、文字、数字、およびアンダースコア（_）文字のみを含むことができます。

AWS アクセスキー：Kinesis ストリームへの書き込み権限を持つ AWS アカウントユーザのアクセスキーID。

AWS 秘密鍵：AWS が上記のユーザに割り当てた秘密鍵

AWS 領域名：AWS リージョンコード

Kinesis ストリーム名：Kinesis で設定した、データストリームの名前

エンドポイント（空白はデフォルト）：AWS 上のデータストリームに別のエンドポイントを設定した場合、ここに入力することができます。

メッセージ内容（空白はデフォルト）：ASP でドキュメントフォーマットを明記することができます。[...] ボタンを押すと、メッセージ編集のためのスクリプトエディターが開きます。メッセージ編集に役立つ一般的な数式やサンプルが表示されます。

蓄積時間（ミリ秒）：DataHub インスタンスが Kinesis にデータを送信する前にメモリにバッファリングする時間を設定します。この値を“0”に設定すると、蓄積時間は適用されません。

［重要］AWS の料金は、受信したメッセージの数に基づいているため、このオプションを使用すると、メッセージを一括送信してコストを削減することができます。

この「**蓄積時間**」と「**最大バッファリング値数**」の両方を設定した場合、DataHub インスタンスは、これらの 2 つのうちのどちらかが制限に達した時に、バッファリングされたすべての値を Kinesis に書き込みます。

最大バッファリング値数：DataHub インスタンスが Kinesis に送信する前にメモリにバッファリング可能な値の最大数。この値を“0”に設定するとすべての値が出来る限り早く送信さてます。

情報レベルでログの書き込み：このオプションにチェックを入れると、データベースへ書き込みが成功したログが、DataHub イベントログの情報レベルのログに出力されます。

② **データのサンプリングと転送**

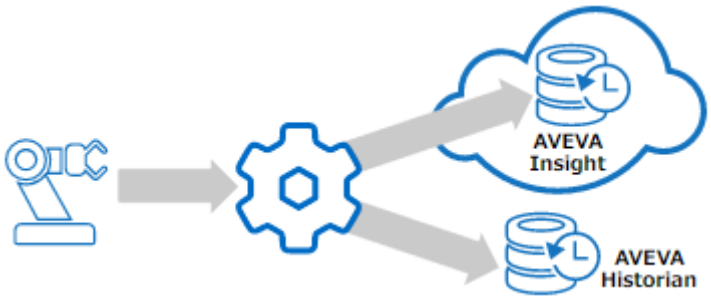
これらのオプションについては、接続設定のデータサンプリング または、転送をご参照ください。

Amazon Kinesis 固有の設定を完了後、「一般的な設定のポイント選択」に戻って設定を続行します。

2．1．4．2．AVEVA Historian、AVEVA Insight 接続

概　要

DataHubの外部ヒストリアンを使用して、AVEVA HistorianまたはAVEVA Insightに直接データを取り込みます。



特　徴

- ミリ秒の時間分解能で Amazon Kinesis Data Streams へ高速データスループット（50,000 書き込み/秒）
- DataHub が収集したデータ（OPC UA および A＆C、OPC DA および A＆E、ModbusTCP スレーブ、ODBC データベース、Excel、カスタムプログラム）の書き込み

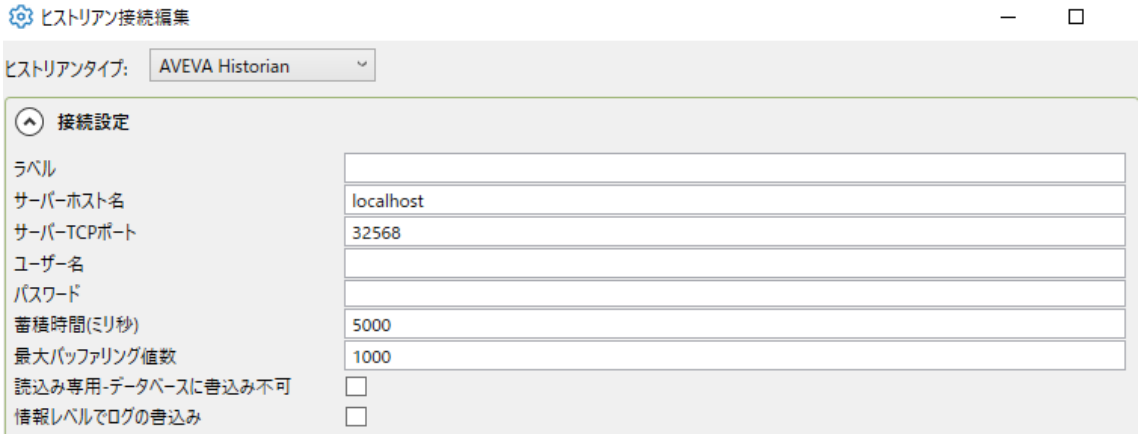
- トンネリングと組み合わせる：ネットワーク経由でヒストリアンデータを収集

詳細：こちら（メーカ英文サイト）
マニュアル：こちら（メーカ英文マニュアル）

1. AVEVA Historian

ヒストリアン接続編集画面から、AVEVA Historian を選択できます。

① 接続設定（Connection Settings）



ラベル：接続を識別するために使用される固有のテキスト文字列。

このラベルは、ストア＆フォワードを設定する際に接続を識別するために使用され、一部のヒストリアンでデータベース名または保持ポリシーを作成するために使用されます。ラベルには、文字、数字、およびアンダースコア（_）文字のみを含むことができます。

サーバーホスト名： AVEVA Historian を実行しているマシンのコンピュータ名またはＩＰアドレス。

サーバ TCP ポート： AVEVA Historian に接続するための TCP ポート。

ユーザ名： AVEVA Historian アカウントのユーザ名。

パスワード： ユーザ名に関連付けられたパスワード。

蓄積時間（ミリ秒）：DataHub インスタンスがデータを AVEVA Historian に送信する前にメモリにバッファリングする時間を設定します。このフィールドを空白のままにすると、データは利用可能になり次第送信されます。

この「**蓄積時間**」と「**最大バッファリング値数**」の両方を設定した場合、DataHub インスタンスは、これらの 2 つのうちのどちらかが制限に達した時に、バッファリングされたすべての値を AVEVA Historian に書き込みます。

最大バッファリング値数：DataHub インスタンスが AVEVA Historian に送信する前に

メモリにバッファリング可能な値の最大数。この値を“0”に設定すると、累計時間内の任意の数の値をバッファリングします。

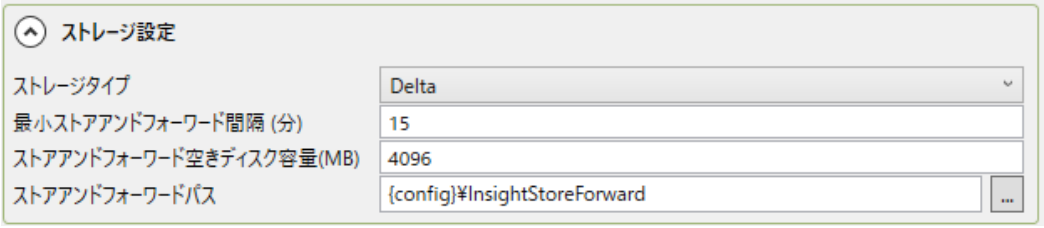
読み込み専用 - データベースへの書き込み不可：

DataHub インスタンスが AVEVA Historian に書き込めないようにします。

情報レベルでログの書き込み：このオプションにチェックを入れると、データベースへ書き込みが成功したログが、DataHub イベントログの情報レベルのログに出力されます。

② **ストレージ設定（Storage Settings）**

DataHub プログラムは、AVEVA Historian への書き込み際に AVEVA Historian ストア&フォワードエージェントを使用します。これらの設定は、そのエージェントの設定を参照しております。詳細は、AVEVA Historian のドキュメントに詳細が記載されています。



ストレージタイプ：転送されたデータの AVEVA Historian 送信タイプ。「**Streamed**」または、「**Non Streamed**」を選択します。

最小ストアアンドフォワード間隔（分）：AVEVA Historian のストレージタイプ。通常、ここは「**Delta**」に設定する必要があります。

最小ストアアンドフォワード間隔（分）：送信待機している間にデータがストアアンドフォワードキューに保存される最小時間。この時間内に送信できないデータは破棄される場合があります。

ストアアンドフォワード空きディスク容量（MB）：ストアアンドフォワードでデータを保持するために必要な最小限のディスク上の空き容量。ディスクの空き容量がこの数値を下回ると、保存されたデータは送信されずに破棄されます。

ストアアンドフォワードパス：転送前にストアアンドフォワードメカニズムによってデータが格納されるパス。

③ **データのサンプリングと転送**

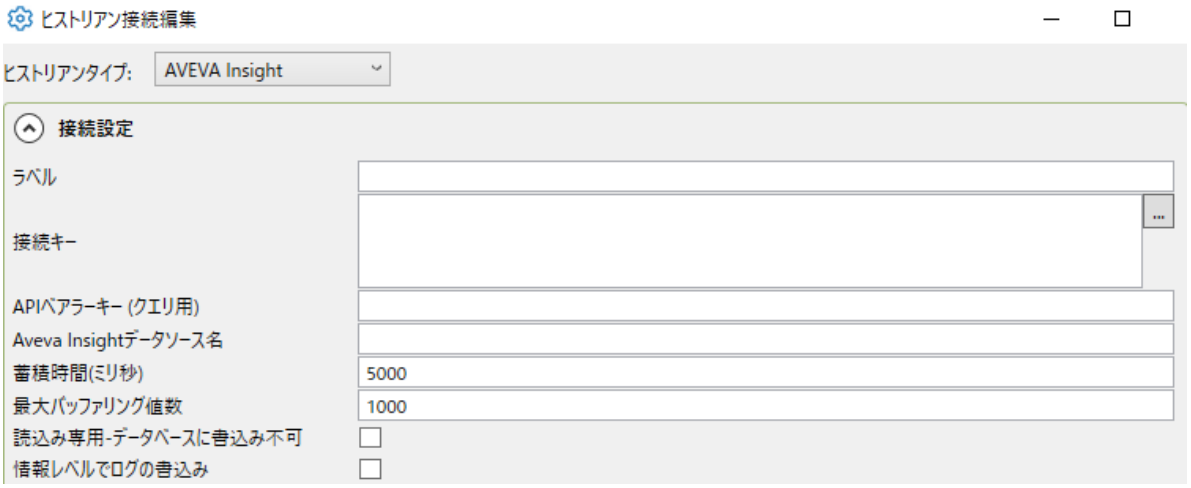
これらのオプションについては、接続設定のデータサンプリング または、転送をご参照ください。

AVEVA Historian 固有の設定を完了後、「一般的な設定のポイント選択」に戻って設定を続行します。

2. AVEVA Insight

ヒストリアン接続編集画面から、AVEVA Insight を選択できます。

① 接続設定（Connection Settings）



ラベル：接続を識別するために使用される固有のテキスト文字列。

このラベルは、ストア＆フォワードを設定する際に接続を識別するために使用され、一部のヒストリアンでデータベース名または保持ポリシーを作成するために使用されます。ラベルには、文字、数字、およびアンダースコア（_）文字のみを含むことができます。

接続キー： Insight Publisher アプリケーションによって生成された接続キー。このフィールドの横にある［...］ボタンを押すと Insight Publisher が開きます。または、AVEVA Insight ポータルからダウンロードすることもできます。

API ベアラーキー（クエリー用）：AWS が上記のユーザーに割り当てた秘密鍵

Aveva Insight データソース名：AVEVA Insight ポータルで構成したソース名、または Insight Publisher でのキー生成プロセス中の作成したソース名。

蓄積時間（ミリ秒）：DataHub インスタンスが AVEVA Insight にデータを送信する前にメモリにバッファリングする時間を設定します。このフィールドを空白のままにすると、データは利用可能になり次第送信されます。送信前にデータを蓄積すると、ネットワークパフォーマンスは向上しますが、遅延が増加します。

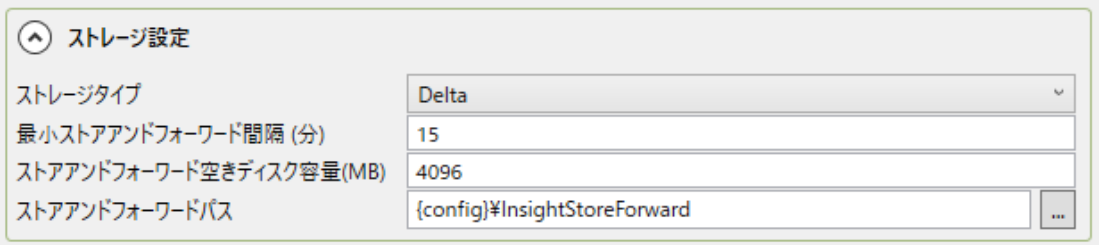
この「**蓄積時間**」と「**最大バッファリング値数**」の両方を設定した場合、DataHub インスタンスは、これらの 2 つのうちのどちらかが制限に達した時に、バッファリングされたすべての値を Kinesis に書き込みます。

最大バッファリング値数：DataHub インスタンスが Kinesis に送信する前にメモリにバッファリング可能な値の最大数。この値を“0”に設定するとすべての値が出来る限り早く送信さてます。

情報レベルでログの書き込み：このオプションにチェックを入れると、データベースへ書き込みが成功したログが、DataHub イベントログの情報レベルのログに出力されます。

④ **ストレーレージ設定**

DataHub プログラムは、AVEVA Insight への書き込み時に Aveva Insight ストア＆フォワード エージェントを使用します。これらの設定は、そのエージェントの構成を参照しており、Aveva Insight のドキュメントに詳細が記載されています。



ストレージタイプ：Aveva Insight のストレージタイプ。通常、ここは Delta に設定する必要があります。

最小ストアアンドフォワード間隔（分）：送信待機している間にデータがストア＆フォワード キューに保存される最小時間。この時間内に送信できないデータは破棄される場合があります。

ストアアンドフォワード空きディスク容量（MB）：ストア＆フォワードでデータを保持するために必要なディスク上の空き容量の最小量。ディスクの空き容量がこの数値を下回ると、保存されたデータは送信されずに破棄されます。

ストアアンドフォワードパス：転送前にストア＆フォワードメカニズムによってデータが格納されるパス。

⑤ **データのサンプリングと転送**

これらのオプションについては、接続設定のデータサンプリング または、転送をご参照

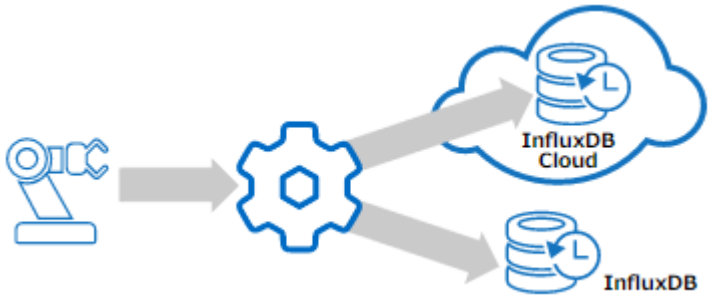
ください。

AVEVA Insight 固有の設定を完了後、「一般的な設定のポイント選択」に戻って設定を続行します。

2．1．4．3．InfluxDB 接続

概　要

DataHub の外部ヒストリアンを使用して、InfluxDB および InfluxDB Cloud へデータ送信します。InfluxDB は、オープンソースの時系列データベースで、付属する Chronograf と Grafana を使ってダッシュボードの作成と分析を行います。InfluxDB Cloud は、サービスとして提供される時系列データベースです。



特　徴

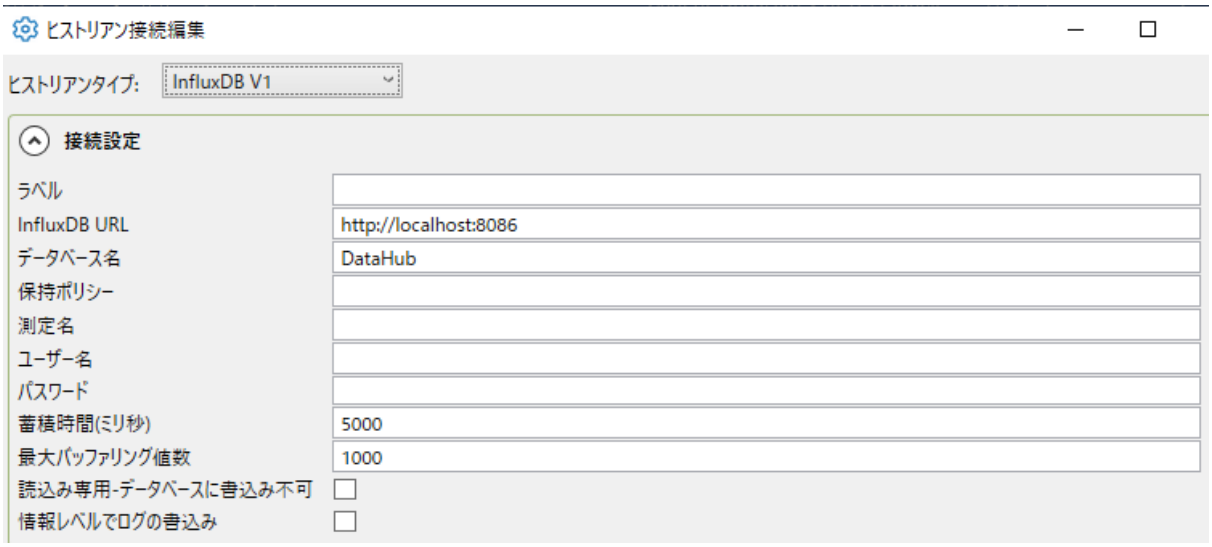
- ミリ秒の時間分解能で高速データ収集（書き込み）
- DataHub が収集したデータ（OPC UA および A＆C、OPC DA および A＆E、ModbusTCP スレーブ、ODBC データベース、Excel、カスタムプログラム）の書き込み
- トンネリングと組み合わせる：ネットワーク経由でヒストリアンデータを収集
- 履歴データを WebView のトレンドチャートへ表示
- OPC UA HAD クライアントからアクセス
- InfluxDB V1、InfluxDB V2 をサポート

詳細：こちら（メーカ英文サイト）
マニュアル：InfluxDB 1　こちら（メーカ英文マニュアル）
　　InfluxDB 2　こちら（メーカ英文マニュアル）

1.　InfluxDB v1

ヒストリアン接続編集画面から、InfluxDB V1 を選択できます。

①　接続設定（Connection Settings）



ラベル：接続を識別するために使用される固有のテキスト文字列。

このラベルは、ストア＆フォワードを設定する際に接続を識別するために使用され、一部のヒストリアンでデータベース名または保持ポリシーを作成するために使用されます。ラベルには、文字、数字、およびアンダースコア（_）文字のみを含むことができます。

InfluxDB URL：InfluxDB データベースの URL。

データベース名：設定された InfluxDB インスタンス c に対して、DataHub インスタンスで設定された名前。

保持ポリシー：使用する InfluxDB 保持ポリシーの名前。存在しない場合は、DataHub インスタンスが作成します。空白の場合は、接続食べるが保持ポリシー名として使用されます。

測定名（Measurement Name）：使用する InfluxDB のメジャーメント名。存在しない場合は、DataHub インスタンスが作成します。空白にすると、DataPoins というメジャーメント名が使用されます。

ユーザー名：設定された InfluxDB インスタンスのユーザ名。

パスワード：ユーザ名に関連付けられたパスワード。

蓄積時間（ミリ秒）：DataHub インスタンスが InfiuxDB にデータを送信する前にメモリにバッファリングする時間を設定します。この値を“0”に設定すると、蓄積時間は適用されません。

この「**蓄積時間**」と「**最大バッファリング値数**」の両方を設定した場合、DataHub インスタンスは、これらの 2 つのうちのどちらかが制限に達した時に、バッファリングされたすべての値を InfluxDB に書き込みます。

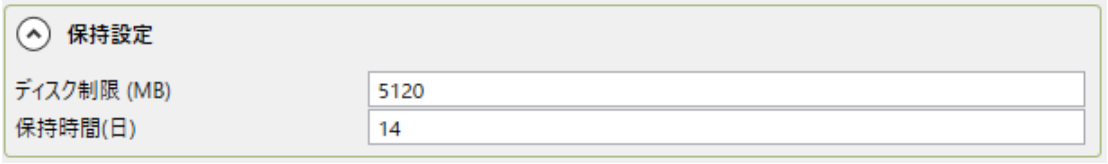
最大バッファリング値数：DataHub インスタンスが InfiuxDB に送信する前にメモリにバッファリング可能な値の最大数。この値を“0”に設定するとバッファリングされません。

読込み専用-データベースに書込み不可：DataHub インスタンスが InfiuxDB のデータセットに書き込みを行わないようにします。

情報レベルでログの書き込み：ここのオプションにチェックを入れると、データベースへ書き込みが成功したログが、DataHub イベントログの情報レベルのログに出力されます。

② **保持設定**

InfluxDB に保持されるデータの量と期間。



DataHub インスタンスは、自らデータベースと保持ポリシーを作成します。データベースがすでに存在する場合は、DataHub インスタンスはそれを使用します。DataHub インスタンスは、database_name.label という名前の保持ポリシーを作成します。database は指定されたデータベース名、label は接続ラベルです。

保持ディスクの上限を“0”に設定すると、ディスク容量は無制限に増加するようになり、制限を無効化することができます。制限値を“0”以外に設定した場合は、DataHub インスタンスは、データのディスク使用量を推定し、ディスク使用量をその数値に近づけるために、時間をかけてデータを削除しようとします。InfiuxDB はディスク上のデータを圧縮し、データを削除してからディスクから削除するまでに時間がかかることがあるため、この推定値は正確ではありません。実際に利用可能なディスクの空き容量に近いディスク使用量の制限を設定しないでください。

保持期間は、0.042(約 1 時間)から 1491308 までの任意の日数で設定することができます。保持期間を“0”に設定すると、保持時間は無限になります。

ディスク制限（MB）：履歴データの保存に割り当てられたメガバイト単位のディスク容量。

保持期間（日）：データが保持される日数。この制限より古いデータは毎日削減されます。

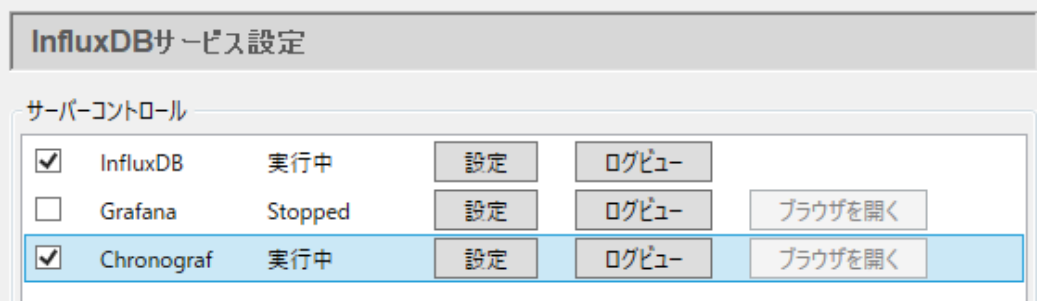
③ **データのサンプリングと転送**

これらのオプションについては、接続設定のデータサンプリング または、転送をご参照ください。

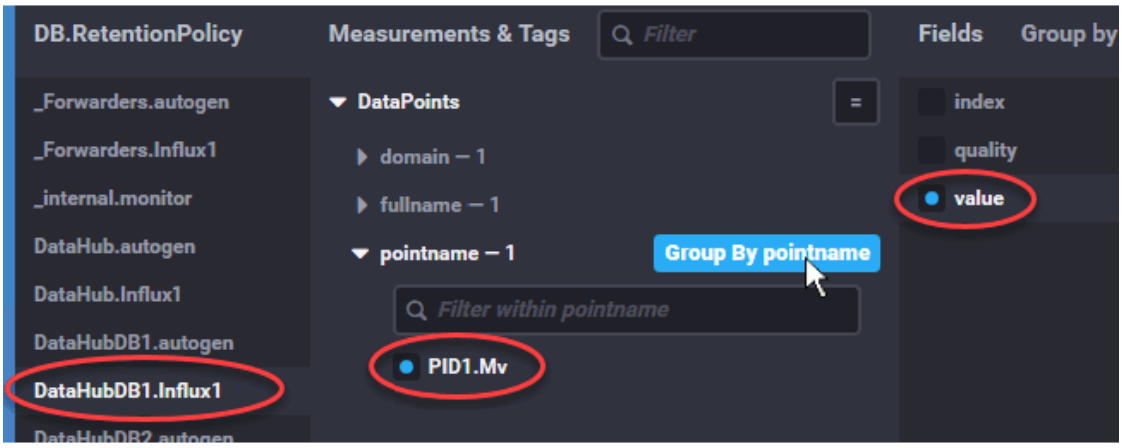
④ **データを確認する**

Chronograf で InfiuxDB データベースの更新を表示できます。

1. InfiuxDB と Chronograf の両方が実行されていることを確認し、[ブラウザを開く]ボタンをクリックします。



2. ダッシュボードに移動し、**+ダッシュボードの作成** をクリックしてから、**+データの追加** をクリックします。
3. 構成したデータベースと、pointname の下の DataPoints に表示するポイントを選択します。
4. 複数のポイントを表示するには、[Group By pointname] をオンにし、Fields で［value］を選択します。



ディスプレイ上部のトレンドにいくつかのデータが表示され始めます。ダッシュボード名を変更し、緑色のチェックボックスをクリックしてダッシュボードを保存することができます。

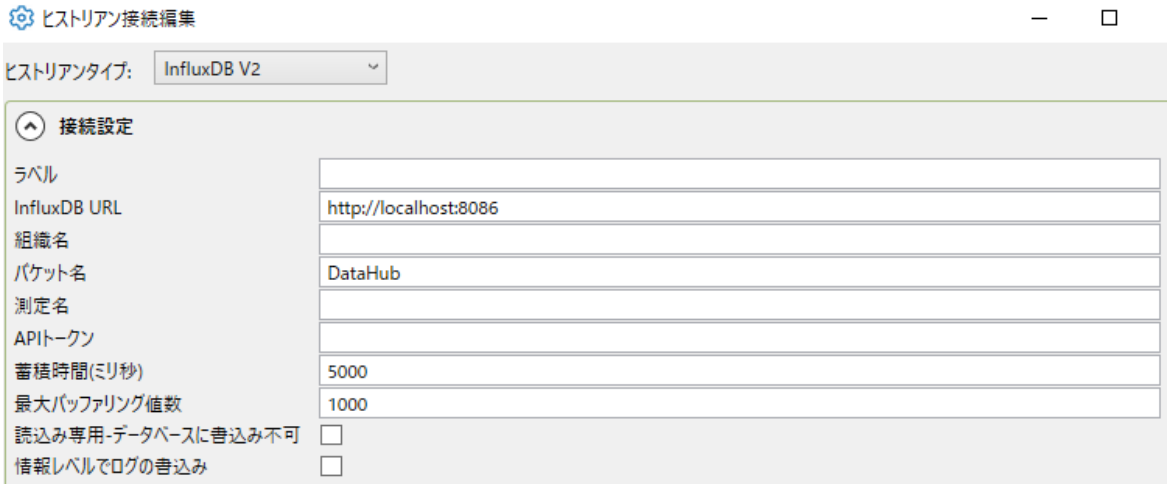
詳細については、Chronograf のドキュメント をご参照ください。

InfluxDB v1 固有の設定を完了後、「一般的な設定のポイント選択」に戻って設定を続行します。

2. InfluxDB v2

ヒストリアン接続編集画面から、InfluxDB V2 を選択できます。

① 接続設定（Connection Settings）



ラベル：接続を識別するために使用される固有のテキスト文字列。

このラベルは、ストア＆フォワードを構成する際に接続を識別するために使用され、InfluxDB などの一部のヒストリアンでデータベース名または保持ポリシーを作成するために使用されます。ラベルには、文字、数字、およびアンダースコア（_）文字のみを含むことができます。

InfluxDB URL：InfluxDB データベースの URL。

組織名：InfluxDB アカウント用に構成された組織名。

パケット名：データを保存する組織の InfluxDB パケット。

測定名（Mesurement Name）：使用する InfluxDB のメジャーメント名。存在しない場合は、DataHub インスタンスが作成します。空白にすると、DataPoins というメジャーメント名が使用されます。**API トークン**：InfluxDB 認証トークン。

蓄積時間（ミリ秒）：DataHub インスタンスが InfiuxDB にデータを送信する前にメモリにバッファリングする時間を設定します。この値を“0”に設定すると、蓄積時間は適用されません。

この「**蓄積時間**」と「**最大バッファリング値数**」の両方を設定した場合、DataHub インスタンスは、これらの 2 つのうちのどちらかが制限に達した時に、バッファリングされたすべての値を InfluxDB に書き込みます。

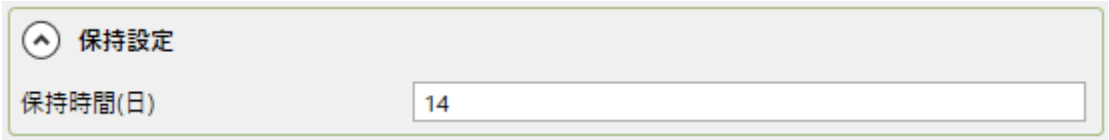
最大バッファリング値数：DataHub インスタンスが InfluxDB に送信する前にメモリにバッファリング可能な値の最大数。この値を“0”に設定すると値はバッファリングされません。

読込み専用-データベースに書込み不可：DataHub インスタンスが InfiuxDB のデータセットに書き込みを行わないようにします。

情報レベルでログの書き込み：このオプションにチェックを入れると、データベースへ書き込みが成功したログが、DataHub イベントログの情報レベルのログに出力されます。

② **保持設定**

InfluxDB に保持される期間。



DataHub インスタンスは、自らデータベースと保持ポリシーを作成します。データベースがすでに存在する場合は、DataHub インスタンスはそれを使用します。DataHub インスタンスは、database_name.label という名前の保持ポリシーを作成します。database は指定されたデータベース名、label は接続ラベルです。

保持期間は、0.042(約 1 時間)から 1491308 までの任意の日数で設定することができます。保持期間を“0”に設定すると、保持時間は無限になります。

保持時間（日）：データが保持される日数。この制限より古いデータは毎日削除されます。

③ **データのサンプリングと転送**

これらのオプションについては、接続設定のデータサンプリング または、転送をご参照ください。

InfluxDB v2 有の設定を完了後、「一般的な設定のポイント選択」に戻って設定を続行します。

2．1．4．4．OSISoft PI System 接続

概　要

DataHubの外部ヒストリアン機能を使用して、RESTfulインターフェイスであるPI Web APIを使用して、OSISoft PI Systemにリアルタイムにデータを共有することが可能です。

特　徴

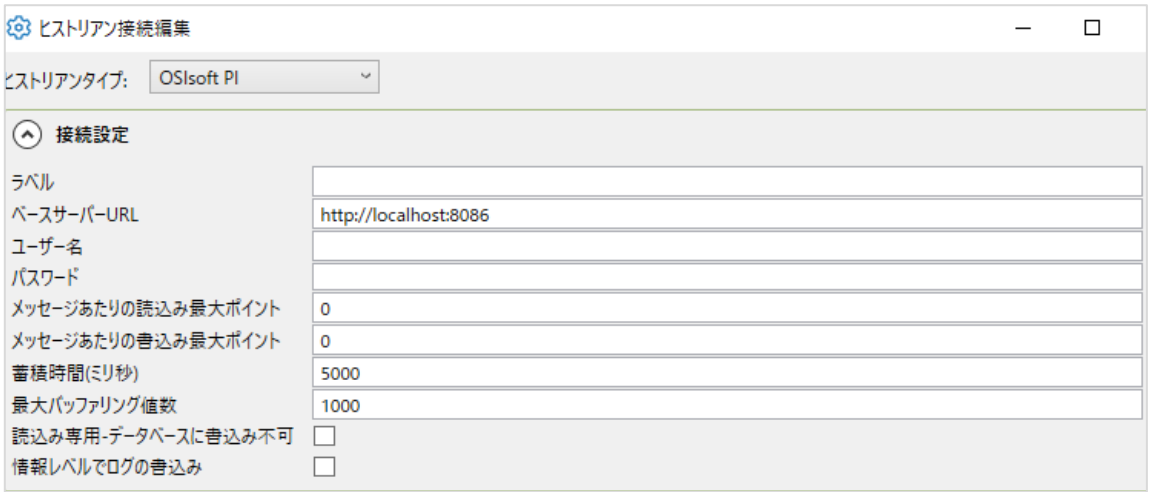
- ミリ秒の時間分解能で PI System へ高速データスループット（20,000 書き込み/秒）
- DataHub が収集したデータ（OPC UA および A＆C、OPC DA および A＆E、ModbusTCP スレーブ、ODBC データベース、Excel、カスタムプログラム）の書き込み
- データ駆動、周期的な書き込み
- トンネリングと組み合わせる：ネットワーク経由でヒストリアンデータを収集

詳細：こちら（メーカ英文サイト）
マニュアル：こちら（メーカ英文マニュアル）

1. OSISoft PI System

ヒストリアン接続編集画面から、データの読み取りおよび書き込みに OSTsoft PI を選択できます。DataHub インスタンスは、PI Web API を使用して、ローカルまたはネットワーク接続を介して PI ヒストリアンと通信します。読み取りおよび書き込みトランザクションは、HTTPS 経由で行われます。

① **接続設定（Connection Settings）**



ラベル：接続を識別するために使用される固有のテキスト文字列。

このラベルは、ストア＆フォワードを設定する際に接続を識別するために使用され、InfluxDB などの一部のヒストリアンでデータベース名または保持ポリシーを作成するために使用されます。ラベルには、文字、数字、およびアンダースコア（_）文字のみを含むことができます。

ベースサーバーURL：PI Web API エンドポイント。

これは https://hostname[:port]/piwebapi となります。Hostname は、PI を実行しているサーバのホスト名または IP アドレスです。PI Web API サービスが 443 以外のポートを使用するように設定されている場合は、コロンの後にポート番号も含める必要があります。（例）https://192.168.1.17:8080/piwebapi

ユーザー名：この接続を認証するために使用するユーザ名。ユーザ erver でユーザとパスワードを設定する必要があります。

パスワード：ユーザ名に関連付けられたパスワード。

メッセージあたりの読み込み最大ポイント：1 回のトランザクションで読み取られる値の最大数。デフォルトでは、PI は読み取りを 150,000 個に制限しています。この値を小さく設定することで、大きなクエリーによって PI サーバに遅延が発生する可能性を減らすことができます。必要に応じて、DataHub インスタンスは複数の読み込みトランザクションを使用して、一定期間にわたってデータを完全に取得します。

メッセージあたりの書き込み最大ポイント：DataHub インスタンスが 1 回のトランザクションで書き込む値の最大数です。DataHub インスタンスが送信待ちの値がこの値より多くある場合は、値はこのサイズ以下の複数のトランザクションに分割されます。

蓄積時間（ミリ秒）：DataHub インスタンスが PI にデータを送信する前にメモリに

バッファリングする時間を設定します。この値を“0”に設定すると、蓄積時間が強制されないことを意味します。この場合は、「**最大バッファリング数値**」によって、データが書き込まれる頻度が決まります。

最大バッファリング値数：DataHub インスタンスが PI に送信する前にメモリにバッファリング可能な値の最大数。この値を“0”に設定すると、値はバッファリングされず、代わりに利用可能になり次第送信されます。

DataHub インスタンスは、ネットワーク効率を高めるために、PI に送信する前に値をバッファリングします。「**蓄積時間**」と「**最大バッファリング値数**」の 2 つのオプションは、バッファリングの制限を表す代替え手段です。DataHub インスタンスは、どちらかが制限に達した時に、バッファリングされた値を PI に送信します。

読込み専用-データベースに書込み不可：DataHub インスタンスが PI に書き込みを行わないようにします。

情報レベルでログの書き込み：このオプションにチェックを入れると、データベースへ書き込みが成功したログが、DataHub イベントログの情報レベルのログに出力されます。

④ **データのサンプリングと転送**

これらのオプションについては、接続設定のデータサンプリング または、転送をご参照ください。

OSIsoft PI 固有の設定を完了後、「一般的な設定のポイント選択」に戻って設定を続行します。

2．1．4．5．REST Client 接続

概　要

DataHubの外部ヒストリアンを使用して、任意のRESTfulインターフェイスにリアルタイムデータをフィードすることが可能です。

特　徴

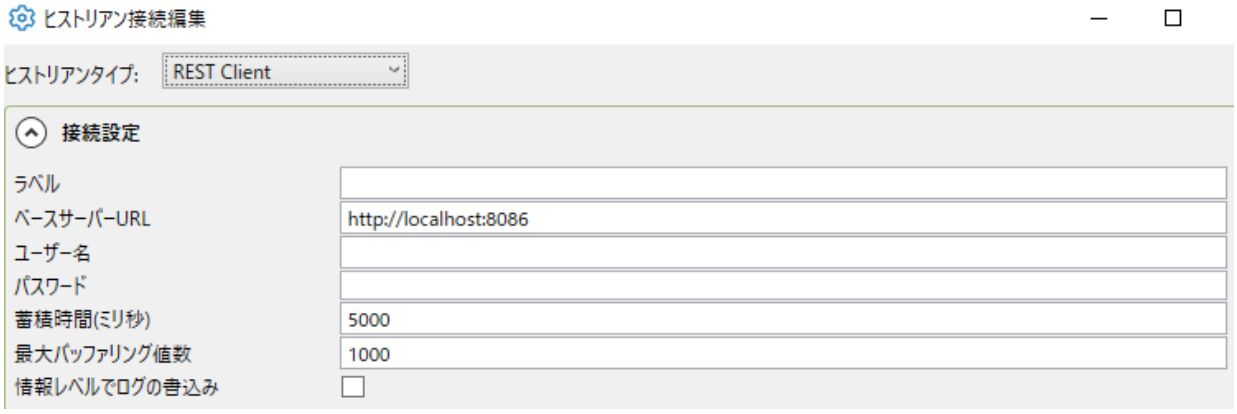
- RESTful システムに対して、ミリ秒の時間分解能で高速データスループット（最大 50,000 書き込み/秒）
- DataHub が収集したデータ（OPC UA および A＆C、OPC DA および A＆E、ModbusTCP スレーブ、ODBC データベース、Excel、カスタムプログラム）の書き込み
- データ駆動、周期的な書き込み
- トンネリングと組み合わせる：ネットワーク経由でヒストリアンデータを収集

詳細：こちら（メーカ英文サイト）
マニュアル：こちら（メーカ英文マニュアル）

1. REST クライアント

ヒストリアン接続編集画面から、REST クライアントを選択できます。外部ヒストリアンのこの実装では、DataHub インスタンスは、REST クライアントとして機能し、REST サーバにデータを送信します。DataHub インスタンスは、ペイロードとしてデータポイント値を含む HTTP 呼び出しを行います。サーバは、成功または失敗の表示で応答します。

① 接続設定（Connection Settings）



ラベル：接続を識別するために使用される固有のテキスト文字列。

このラベルは、ストア＆フォワードを構成するときに接続を識別するために使用され、InfluxDB などの一部のヒストリアンでデータベース名または保持ポリシーを作成するために使用されます。ラベルには、文字、数字、およびアンダースコア（_）文字のみを含むことができます。

ベースサーバーURL：REST サービスの URL。これは、**相対パス** と組み合わせて、データの送信先の完全な URL を形成するプレフィックスです。

ユーザー名：DataHub インスタンス REST クライアントのユーザー名。

パスワード：ユーザ名に関連付けられたパスワード。

蓄積時間（ミリ秒）：DataHub インスタンスが Kinesis にデータを送信する前にメモリにバッファリングする時間を設定します。この値を“0”に設定すると、蓄積時間は適用されなくなります。

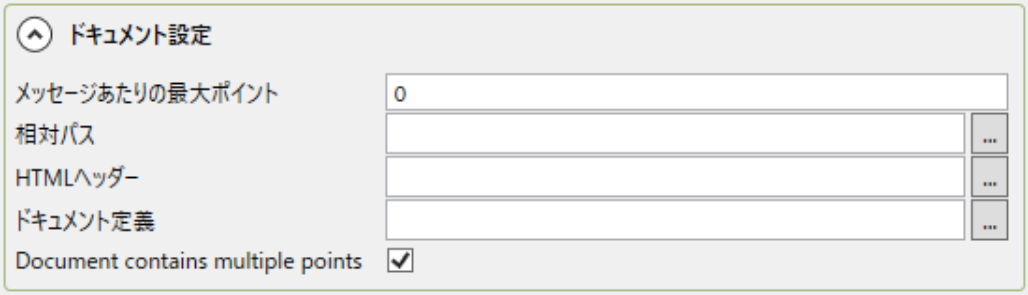
DataHub インスタンスは、データをバッチごとに REST サーバに書き込みます。これらの設定を使用して、バッチ処理の動作を変更できます。「**蓄積時間**」と「**最大バッファリング値数**」の 2 つのオプションは、バッファリングの制限を表す代替え手段です。DataHub インスタンスは、どちらかが制限に達した時に、サーバリングされたすべての値を REST サーバに書き込みます。

最大バッファリング値数：DataHub インスタンスが Kinesis に送信する前にメモリにバッファリング可能な値の最大数。この値を“0”に設定するとすべての値が出来る限り早く送信さてます。

情報レベルでログの書き込み：このオプションにチェックを入れると、データベースへ書き込みが成功したログが、DataHub イベントログの情報レベルのログに出力されます。

② **ドキュメント設定**

REST 接続を確立するために使用される JSON、XML またはその他のドキュメントの属性。



メッセージあたりの最大ポイント：1 メッセージあたりの最大ポイント

REST クライアントは、蓄積されたデータを 1 つまたは複数のメッセージで送信することができます。この設定は、1 つの HTTP メッセージで送信できるポイント数を決定します。蓄積されたポイント（上記の Accumulation time と Max # of buffered values で設定）がこの数を超えると、値が複数のメッセージに分割されます。値 0 は、すべての蓄積されたポイントが単一のメッセージで送信されることを意味します。

次の 3 つの項目（相対パス、HTML ヘッダー、ドキュメント定義）は、ASP プロセッサで処理されるスクリプトを使用して値を定義しています。これにより、実行時に変

化するデータなどに応じて、これらの設定を変化させるように指定することができます。文書定義の構文と例については、以下を参照してください。

相対パス：これは、メッセージの送信先となる URL のパス部分です。これは Base server の URL に追加されます。相対パスは空白にすることができ、その場合はベースサーバの URL がメッセージの送信先となる URL を完全に指定することになります。相対パスが空白でない場合、これは文字列を返すスクリプトとなります。簡単な例を記載します。

```
<%= "/input" %>
```

あるいは、

```
/input
```

他の情報をもとの REST サーバの URL が変更された場合にこのエントリーを使用します。

HTML ヘッダー：HTTP POST メッセージに付加する HTML ヘッダーを指定します。これは、name: value の形式で 1 行に 1 つずつ文字列を並べたドキュメントです。

たとえば

```
Content-Type: text/json
```

このエントリは、REST サーバが特別な HTTP ヘッダー（通常は認証用）を請求している場合に使用します。

ドキュメント定義：POST メッセージのボディを ASP 形式で構築するスクリプトです。これにより、JSON、XML、またはポイントデータ値とメッセージ構造を混合したカスタムメッセージボディを構築することができます。

Document contains multiple points（ドキュメントに複数のポイントを記載）：このオプションを使用すると、ドキュメントをポイントごとに 1 回生成するか、ポイントのリストで操作するかを選択できます。このオプションを設定すると、ドキュメントの処理時に変数ポイントが定義されます。このオプションをオフにすると、ドキュメントの処理時に変数ポイントが定義されます。

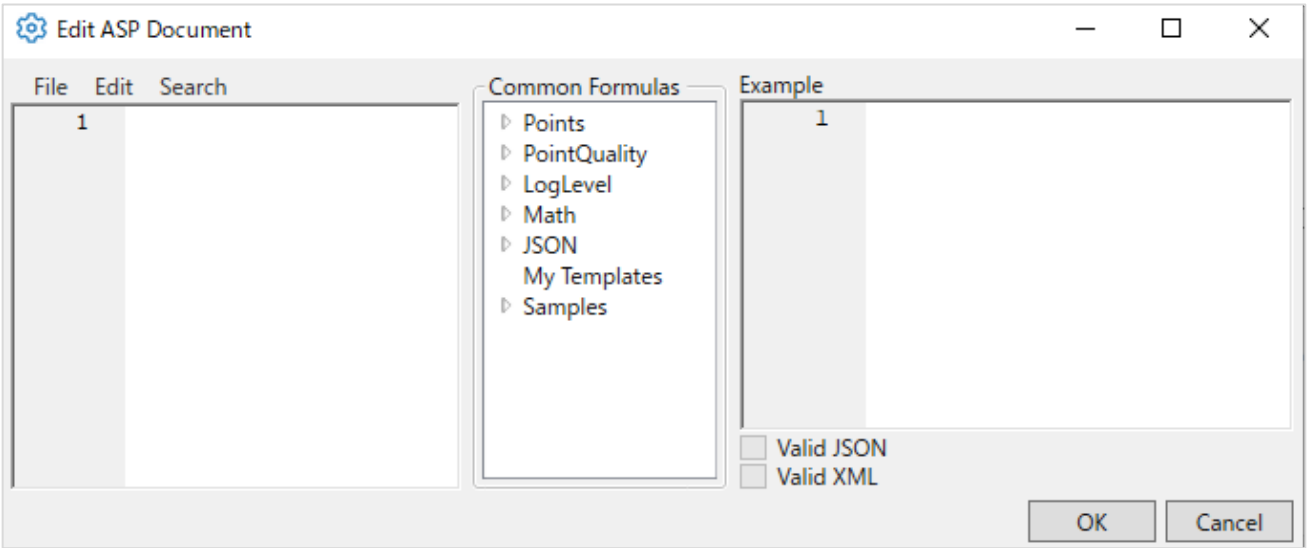
③ データのサンプリングと転送

これらのオプションについては、接続設定のデータサンプリング または、転送をご参照ください。

REST クライアント固有の設定を完了後、「一般的な設定のポイント選択」に戻って設定を続行します。

④ **APS ドキュメント定義**

ASP 値は、次のダイアログで定義されます。



左側ペイン：値を定義する ASP ドキュメントが含まれています。ここにドキュメントを入力してください。

中央ペイン：保存された例題とテンプレートをご用意しており、テンプレート名をダブルクリックすることで読み込むことができます。文章定義を作成したら、「ファイル」→「名前を付けて保存」を選択して、テンプレートに名前を付けて保存することができます。

右側ペイン：左側ペインにあるドキュメント定義を使用して、いくつかのポイント変更イベントの例を処理した結果が表示されます。「Valid JSON」と「Valid XML」の 2 つのチェックボックスは、Example ペインの結果のドキュメントが有効な JSON または有効な XML であれば、自動的にチェックされます。これは、結果が JSON または XML である場黄ときに、ドキュメント定義のエラーを識別するためのものです。

ASP ドキュメントは、通常のテキストと特殊な ASP ディレクティブを組み合わせたあらゆるテキスト文章です。ASP 指示文の内容は、S-Sharp で記述されたスクリプトです。スクリプト式とスクリプト文は、それぞれ有効な S-Sharp の式と文です。ASP 指示文には、符号（=）の違いだけで、 2 つの ASP 指示文が

<% *one or more script statements* >

このディテクティブは、ASP プロセッサにスクリプトを実行し、出力独メントに何も出力しないように指示します。このディレクティブは、関数の定義、変数の初期化、ログメッセージなど、出力文書にテキストを追加しない場合に使用することができます。スクリプトは、複数の<% %>ディレクティブに分割して記述

することができ、ディレクティブの合計が有効な式の並びである限り、複数のディテクティブに分割することができます。

<%= *one or more script statements* >

このディテクティブは、ASP プロセッサにスクリプト文を実行し、その結果を出力ドキュメントに挿入するように指示します。このディレクティブを使用して、ポイント名、値、タイムスタンプ、品質などを挿入することができます。スクリプト文はステートメントであってはなりません。つまり、ステートメントの終わりを示すセミコロンやブレースで終わらないでください。

JSON の例

これは、Microsoft Power BI へのデータ送信に有効な JSON ドキュメントの例です。

```
[
<%
  var point; var i;
  for (i = 0; i < points.Length; i++)
  {
    point = points[i];
%><%= i == 0 ? "" : ",¥n" %>  {
    "<%= point.Name %>": <%= point.JsonValue %>,
    "Date": "<%= point.IsoTimestamp %>"
  }<% } %>
]
```

以下に XML の例を記載します。

Power BI の例を分解してみると、より理解しやすくなります。一番外側の角括弧 [ ] は、JSON 配列を囲むリテラル括弧です。この配列の中で送信する必要があるポイントのリストを繰り返し処理したいと思います。ポイントの配列は、変数 points に自動的に格納されます。ポイントの配列ループには、リテラルテキストと ASP ディレクティブ内のスクリプトの両方が含まれています。コードを色付けすることで、理解し易くなります。

```
[
<%
  var point; var i;
  for (i = 0; i < points.Length; i++)
  {
    point = points[i];
%><%= i == 0 ? "" : ",\n" %>  {
    "<%= point.Name %>": <%= point.JsonValue %>,
    "Date": "<%= point.IsoTimestamp %>"
  }<% } %>
]
```

<% %>ディレクティブ内のテキストは、出力を生成しないスクリプトコードです。これはドキュメント コンテキストでは無く、スクリプトの構造を表しています。そのスクリプトコードを抜き出すと、次のような構造になっていることがわかります。

```
var point; var i;
  for (i = 0; i < points.Length; i++)
  {
    point = points[i];
  }
```

ドキュメント コンテキストは、リテラルテキストとディレクティブ<% %>内のコードの評価結果の組み合わせです。これを抽出することで、文章内容の構造を知ることができます。

```
[
<%= i == 0 ? "" : ",\n" %>  {
    "<%= point.Name %>": <%= point.JsonValue %>,
    "Date": "<%= point.IsoTimestamp %>"
  }
]
```

このディレクティブ `<%= i == 0 ? "" : ",\n" %>` は、ポイントリストの最終の項目を省くすべての項目に対して、結果のドキュメントにカンマと改行文字を挿入します。

`"<%= point.Name %>"`このディテクティブは、二重引用符で囲まれたポイント名を結果のドキュメントに挿入します。二十引用符を明示的に追加しなければならないことに注意してください。これは、二重引用符やバックスラッシュ文字を含むポイント名に対しては失敗します。バックスラッシュやダブルクォート文字でポイント名をエスケープする必要がある場合は、次を使用します。

```
"<%= point.Name.Replace("\\","\\\\").Replace("\"","\\\"") %>"
```

このディテクティブ <%= point.JsonValue %>は、JSON 形式のポイント値を出力に挿入します。ポイント値が文字列の場合、文字列を二重引用符で囲み、引用符とバックスラッシュ文字をエスケープすることも含まれます。

このディレクティブ"<%= point.IsoTimestamp %>"は、ポイントのタイムスタンプを ISO 8601 フォーマットで挿入します。

データポイントは、ドキュメントに挿入することができるいくつかのメンバーが含まれています。

Value – オブジェクトとしてのポイント値。これは、任意の文字列または数値型を含むことができます。

IntVal – integer に変換されたポイント値。

DbVal – double に変換されたポイント値。

StrVal – ポイントの値を文字列に変換した値

Name – ドメイン名を含む、ポイントの完全な名前（文字列）。

DomainName – ポイントの名前のドメイン名部分（文字列）。

Quality – PointQuality 値としてのポイントの品質。整数または文字列に変換可能。

Timestamp – UTC でのポイントのタイムスタンプを OLE オートメーション日付（OADate）として、double で表します。

DateTime – DateTime としての UTC でのポイント・タイムスタンプ。

IsoTimestamp – ISO 8601 形式のポイント・タイムスタンプ（文字列）。

JsonTimestamp – JSON 形式のポイント・タイムスタンプ。UNIX エポックスタート（1970 年 1 月 1 日 00:00:00 UCT）から何ミリ経過したか。

JsonValue – 有効な JSON フォーマットでのポイント値です。文字列は二重引用符で囲まれ、引用符のバックスラッシュ文字はエスケープされます。

XML の例

この例は、グローバル・タイムスタンプと、名前、値、品質、タイムスタンプなどの一連のデータポイントを送信する XML ドキュメントの例です。

```
<DataPoints>
  <Timestamp><%= DateTime.UtcNow.ToString("o") %></Timestamp>
  <Points>
```

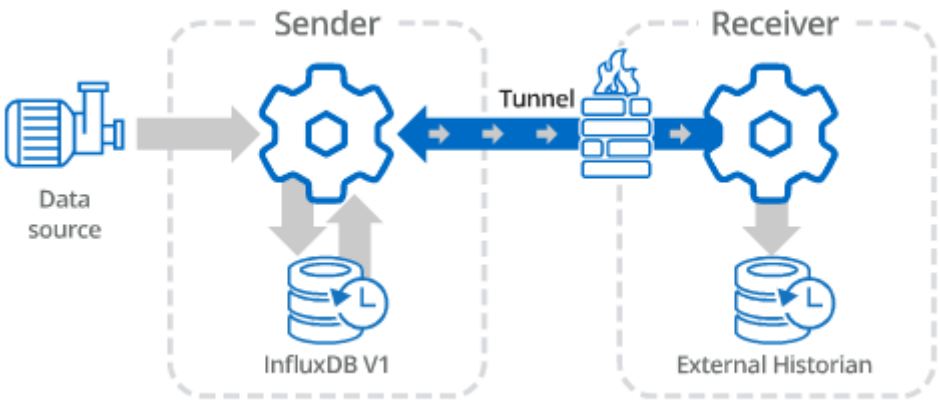
```
  <%
    var point; var i;
    for (i = 0; i < points.Length; i++)
    {
      point = points[i];
  %>  <Point>
      <Name><%= point.Name %></Name>
      <Value><%= point.Value %></Value>
      <Quality><%= (int)point.Quality %></Quality>
      <Timestamp><%= point.IsoTimestamp %></Timestamp>
    </Point>
  <% } %>
  </Points>
</DataPoints>
```

2．1．4．6．Tunnel（Pull）接続

概　要

DataHubの外部ヒストリアンは、DataHubのトンネリングを介して送信側の外部ヒストリアンのデータをプルで、受信側の外部ヒストリアンに格納することができます。

送信側のヒストリアンは、InfluxDB V1である必要があります。受信側のヒストリアンは、サポートされている任意のヒストリアンにすることができます。

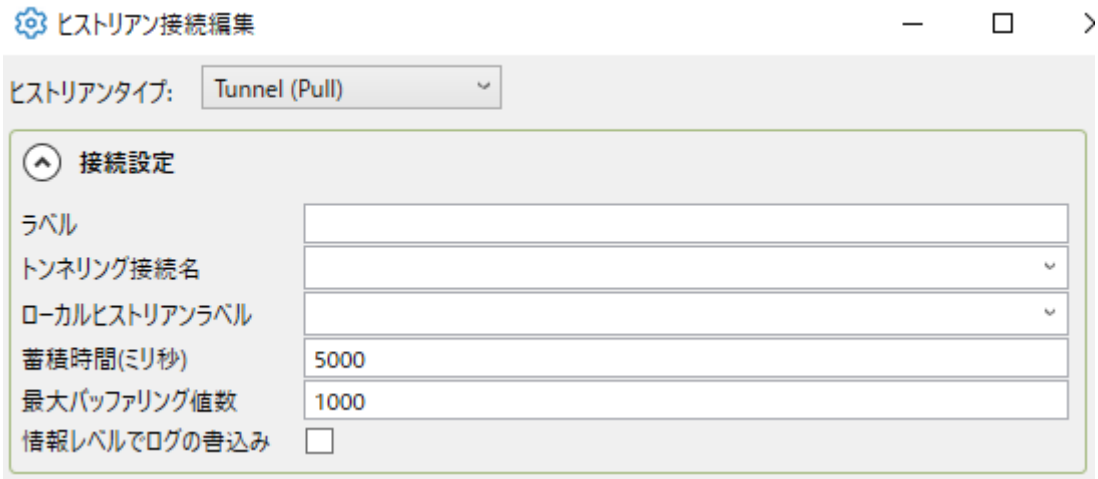


1. Tunnel（Pull）

ヒストリアン接続編集画面から、Tunnel（Pull）を選択します。

トンネル（プル）ヒストリアンタイプでは、リモートマシン上で動作する DataHub トンネルマスターに接続し、リモート外部ヒストリアンからデータを取得し、ローカルヒストリアンへロギングするトンネルスレーブとして DataHub インスタンスを構成することが可能です。詳細は、外部ヒストリアンの典型的はシナリオ をご参照ください。

① 接続設定（Connection Settings）



ラベル：接続を識別するために使用される固有のテキスト文字列。

このラベルは、ストア＆フォワードを設定する際に接続を識別するために使用され、一部のヒストリアンでデータベース名または保持ポリシーを作成するために使用されます。ラベルには、文字、数字、およびアンダースコア（_）文字のみを含むことができます。

トンネリング接続名：リモート DataHub インスタンスからデータをプルするために設定したトンネリングの接続名。詳細については、チュートリアルの 受信側セクション と トンネル（Pull） をご参照ください。

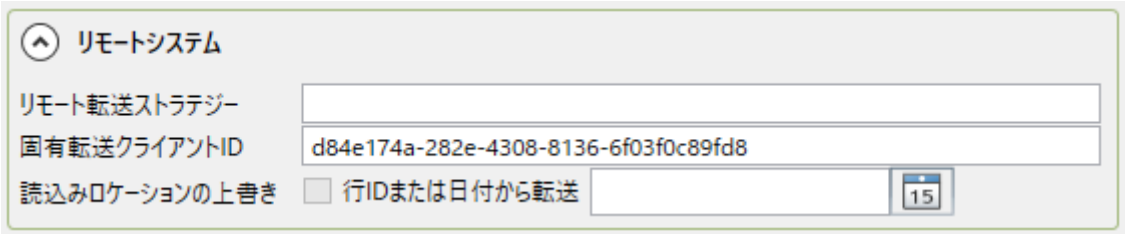
ローカルヒストリアンラベル：リモート側のヒストリアン用に構成されたラベル。

蓄積時間（ミリ秒）：送信側の DataHub インスタンスが、トンネル経由でデータをプルだれる前にメモリにバッファリングする時間（ミリ秒）を設定します。このフィールドを空白にすると、蓄積時間が強制されないことを意味します。この「**蓄積時間**」と「**最大バッファリング値数**」の両方を設定した場合、DataHub インスタンスは、これらの 2 つのうちのどちらかが制限に達した場合、トンネリング経由ですべてのバッファリングされた値を引き出し（プル）ます。

最大バッファリング値数：送信側の DataHub インスタンスがトンネリング経由でプルされる前にメモリにバッファリングする値の最大数。この値を空白にすると、バッファリングされないことを意味します。

情報レベルでログの書き込み：このオプションにチェックを入れると、データベースへ書き込みが成功したログが、DataHub イベントログの情報レベルのログに出力されます。

② **リモートシステム**



リモート転送ストラテジー：送信側の DataHub インスタンスの外部ヒストリアン設定で設定した転送ストラテジーのラベル。

固定転送クライアント ID：この転送設定を識別するための固有のテキスト文字列。システムで生成された文字列を使用するか、独自の文字列を入力することができます。

読み取りロケーションの上書き：開始点をより新しい時間にリセットすることにより、データセットの一部を転送できるようにします。日付/時刻データの形式は、YYYY-MM-DD HH:MM:SS で、必要に応じて編集することができます。カレンダーボタンで素早く日付を入力することができます。

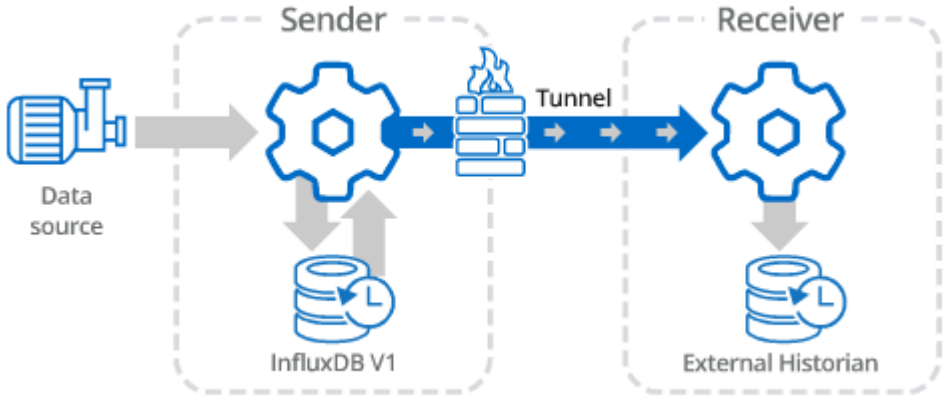
トンネル（プル）固有の設定が完了後、「一般的な設定のポイント選択」に戻って設定を続行します。

2．1．4．7．Tunnel（Push）接続

概　要

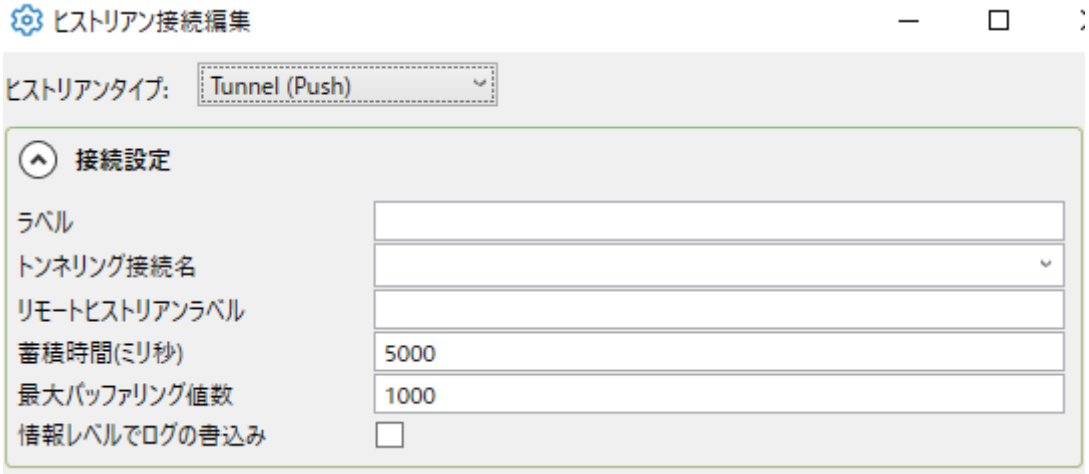
DataHubの外部ヒストリアンは、DataHubのトンネリングを介して送信側の外部ヒストリアンのデータをプッシュで、受信側の外部ヒストリアンに格納することができます。

送信側のヒストリアンは、InfluxDB V1である必要があります。受信側のヒストリアンは、サポートされている任意のヒストリアンにすることができます。



1. Tunnel(Push)

① 接続設定（Connection Settings）



ラベル：接続を識別するために使用される固有のテキスト文字列。

このラベルは、ストア＆フォワードを設定する際に接続を識別するために使用され、一部のヒストリアンでデータベース名または保持ポリシーを作成するために使用されます。ラベルには、文字、数字、およびアンダースコア（_）文字のみを含むことができます。

トンネリング接続名：リモート DataHub インスタンスからデータをプルするために設定したトンネリングの接続名。詳細については、チュートリアルの 受信側セクション と トンネル（Pull） をご参照ください。

ローカルヒストリアンラベル：リモート側のヒストリアン用に構成されたラベル。

蓄積時間（ミリ秒）：送信側の DataHub インスタンスが、トンネル経由でデータを取得する前にメモリにバッファリングする時間（ミリ秒）を設定します。このフィールドを空白にすると、蓄積時間が強制されないことを意味します。この「**蓄積時間**」と「**最大バッファリング値数**」の両方を設定した場合、DataHub インスタンスは、これらの２つ

のうちのどちらかが制限に達した場合、トンネリング経由ですべてのバッファリングされた値を引き出し（プル）ます。

最大バッファリング値数：DataHub インスタンスがトンネリング経由でバッファリングされた値を引き出す（プル）前にメモリにバッファリング可能な最大数。この値を空白にすると、バッファリングされないことを意味します。

情報レベルでログの書き込み：このオプションにチェックを入れると、データベースへ書き込みが成功したログが、DataHub イベントログの情報レベルのログに出力されます。

② **データのサンプリングと転送**

これらのオプションについては、接続設定のデータサンプリング または、転送をご参照ください。

Tunnel(Push)固有の設定を完了後、「一般的な設定のポイント選択」に戻って設定を続行します。

2．2．InfluxDB

概　要

InfluxDB は、DataHub プログラムのプラグインと「して提供されるオープンソースの時系列データヒストリアンです。InfluxDB の DataHub プログラム実装には、InfluxDB ダッシュボードである Chronograf とともに、InfluxDB と連携すうように構成できるオープンソースの分析および監視ツールである Grafana が含まれています。

特　徴

・InfluxDB、Grafana、Chronograf の起動、構成、接続

注 意 点

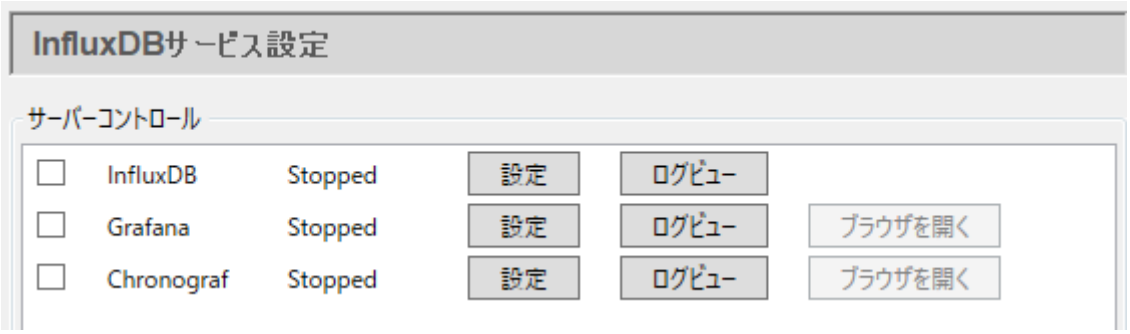
InfluxDB は、64 ビットオペレーティングシステムでのみサポートしています。

DataHub をインストールすると、InfluxDB、Grafana および Chronograf が一緒にインストールされます。すでにご自身で InfluxDB、Grafana および Chronograf をインストールして構成している場合や異なるバージョンをインストールしたい場合は、DataHub と同じフォルダにインストールしないようにしてください。DataHub インストール時に上書きされます。

マニュアル：こちら（メーカ英文マニュアル）

サーバー制御

ここで、３つの InfluxDB コンポーネント（InfluxDB、Grafana、Chronograf）を起動、構成および接続を行います。

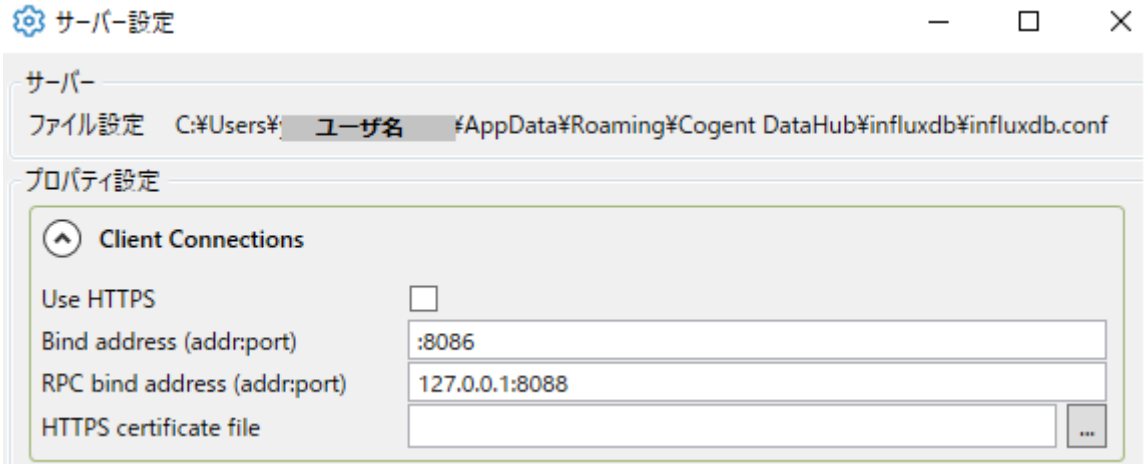


DataHub プログラムは、InfluxDB とそのコンポーネント用に事前設定されています。チェックボックスをクリックし、「適用」ボタンを押してそれらのいずれかを開始します。ステータスが **Staeting** に変わり、次に **Running** に変わります。

コンポーネントの構成を編集する必要がある場合は、「設定」ボタンをクリックして、InfluxDB、Grafana または Chronograf のサーバー設定画面を開きます。

２．２．１．InfluxDB サーバー設定

これは、InfluxDB プログラムのサーバー設定画面です。通常、設定の目的がわからない場合は、変更する必要がありません。



① クライアント接続（Client Connections）

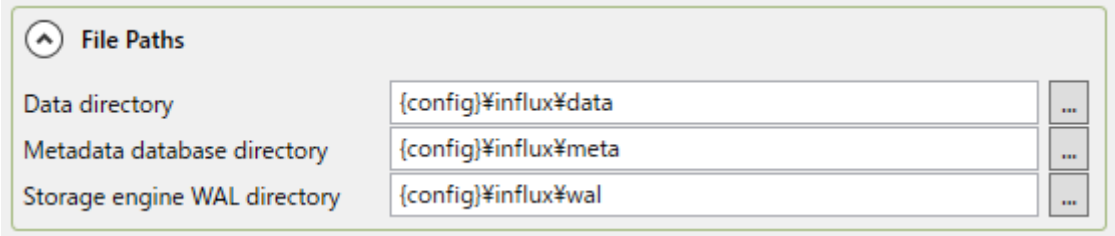
Use HTTPS：選択すると、サーバーは HTTP ではなく HTTPS として接続をリッスンします。速度よりもセキュリティが重要な場合に使用します、クライアントアプリケーションは、このプロトコルを使用してこの InfluxDB サーバーに接続する必要があります。

Bind address（addr:port）：サーバーがリッスンするポートを指定します（HTTP または HTTPS）。InfluxDB を使用すると、リッスンするネットワーク・インターフェイスとポート番号を制限できます。デフォルト値は、**:8086** です。特殊な構文 **:1234** は、すべてのネットワーク・インターフェイスでリッスンすることを示します。

RPC bind address（addr:port）：TBD

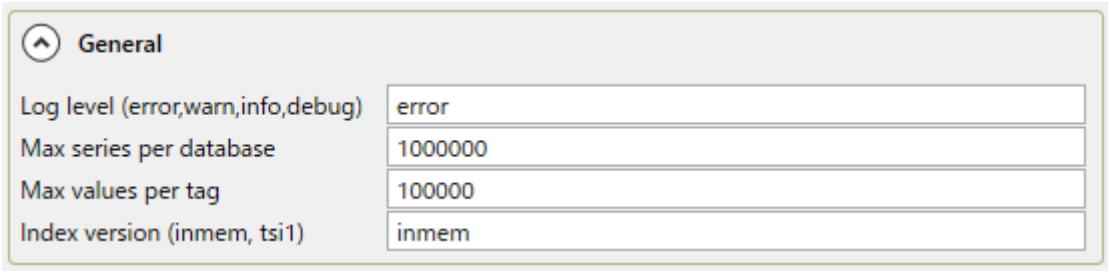
HTTPS certificate file：SSL 証明書の格納場所を入力します。証明書は、秘密鍵を含む PEM 形式のファイルである必要があります。

② ファイルパス（File Paths）



ディスク上の InfluxDB データベースへのファイルパスを設定します。文字列｛config｝は、DataHub の設定フォルダに｛install｝は、DataHub インストールフォルダに置き換わります。通常、data、metadata、WAL の各フォルダは同じ親フォルダ内に配置する必要があります。

③ 全般的（General）



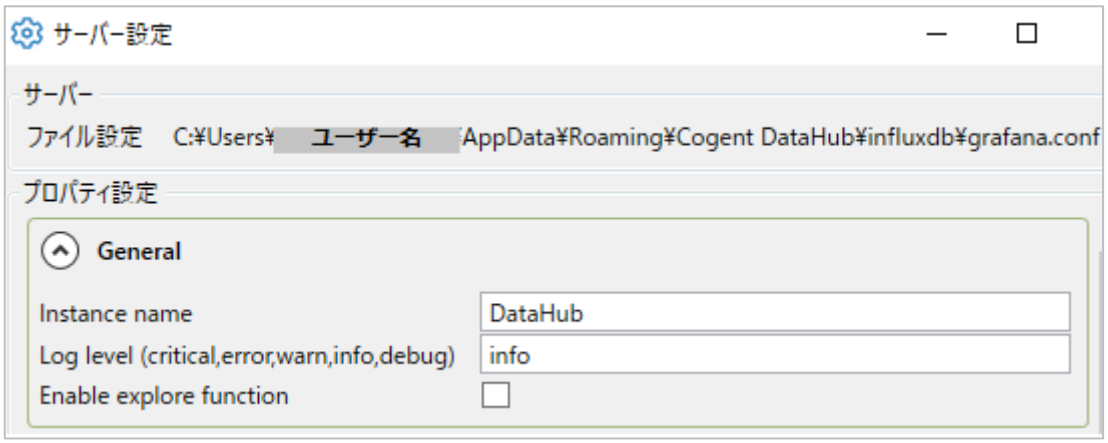
Log level（error,warm,info,debug）：InfluxDB サーバーがログファイルに書き込むメッセージの重要度レベルを指定します。

Max series per database：TBD

Max values per tag：TBD

2．2．2．Grafana（グラファナ）

これは、InfluxDB と連携する分析・監視ツールである Grafana のサーバ設定画面です。



① **全般的（General）**

Instance name：デフォルト名（DataHub）を変更することができます。

Log level（critical,error,warn,info,debug）：Grafana がログファイルに書き込むべきメッセージの重要度レベルを指定します。設定フォルダ

Enable explore function：Grafana の検索機能を有効にします。

詳細：**https://grafana.com/docs/features/explore/**

② **ファイルパス（File Paths）**

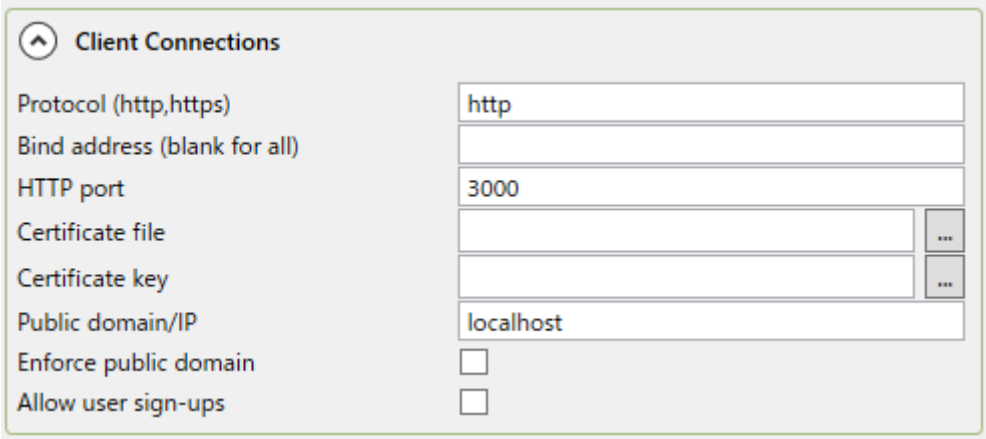
Grafana のデータ、ログおよびプラグイン フォルダのファイルの場所を設定します。文字列 {config} は、**dh** に置き換えられます。{install}は、インストールファルダに置き換わります。通常は、これらのフォルダは、同じ親フォルダ内に配置する必要があります。

Data directory：データフォルダーのパス

Log directory：ログフォルダーのパス

Plugin directory：プラグインフォルダーのパス

③ **クライアント接続（Client Connection）**



Protocol（http,https）：Grafana でサポートするプロトコルを入力します。

Bind address（blank for all）：Grafana は、ここで指定されたアドレスのみをリッスンします。これが空白のままの場合は、すべてのアドレスをリッスンします。

HTTP port：HTTP または HTTPS クライアントが接続するためのポート。

Certificate file：証明書ファイルへのファイルパス

Certificate key：キーファイルへのファイルパス

Public domain/IP：この Grafana サーバのパブリックドメイン名または IP アドレス。

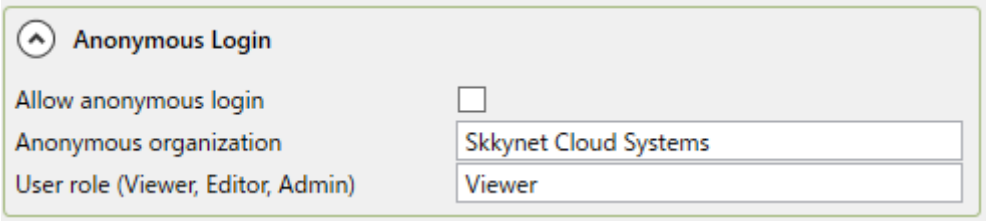
詳細：**https://grafana.com/docs/installation/configuration/#domain**

Enforce public domain：パブリックドメインの使用を強制します。

詳細： https://grafana.com/docs/installation/configuration/#enforce-domain

Allow user sign-ups：ユーザのサインアップを許可する。このボックスをチェックすると、ユーザがアカウントにサインアップし、独自のユーザ名とお明日ワードを作成することができます。このボックスをチェックしない場合は、Grafana 管理者の新しいユーザアカウントを作成する必要があります。

④ **匿名ログイン（Anonumous Login）**

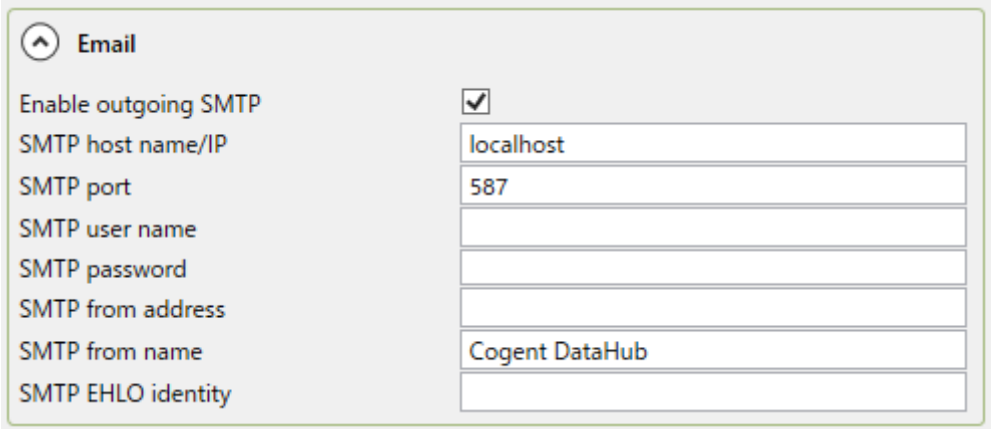


Allow anonymous login：匿名でログインを許可する。

Grafana がメールの送信に使用するメールアカウントの適切な情報を入力します。

詳細：https://grafana.com/docs/installation/configuration/#smtp

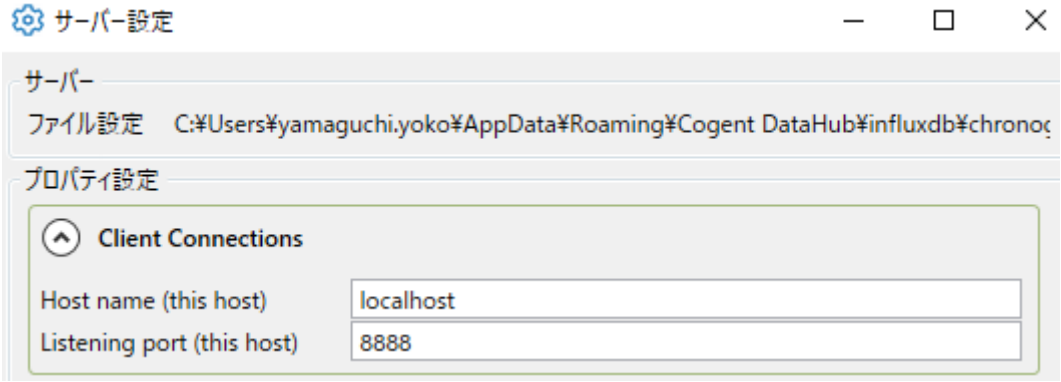
⑤ **Email**



⑥ **セキュリティ（Security）**

2．2．3．Chronograf（クロノグラフ）

これは、InfluxDB ダッシュボードである Chronograf のサーバ設定画面です。



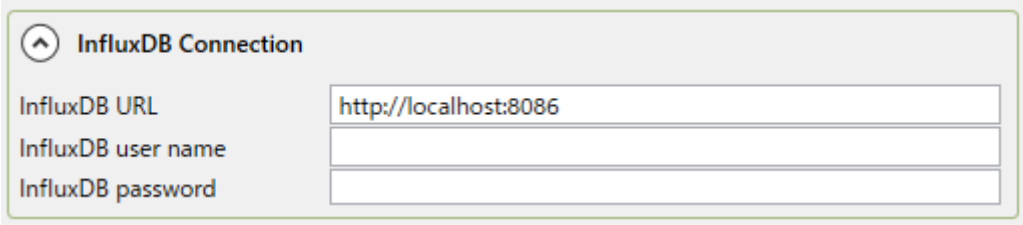
① **クライアント接続（Client Connections）**

Chronograf は、適切に設定されたクライアントからインバウンドコネクションを受け付けます。デフォルトでは、認証なしですべての接続を許可するように設定されています。

Host name（this host）：Chronograf および DataHub プログラムを実行しているコンピュータ名または IP アドレス。

Listening port（this host）：Chronograf クライアント接続を受け入れるために開かれたポート。

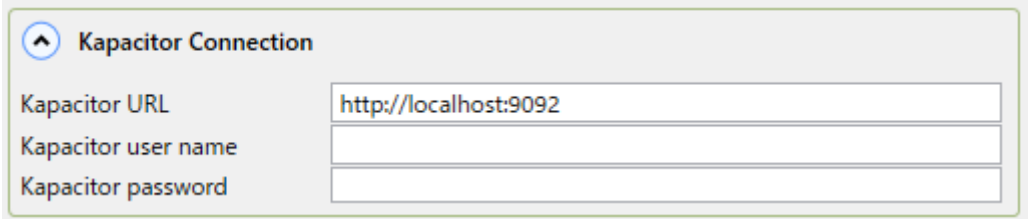
② **InfluxDB 接続（InfluxDB Connection）**



Chronograf は、ここで指定された URL、ユーザー名およびパスワードを InfluxDB に接続します。

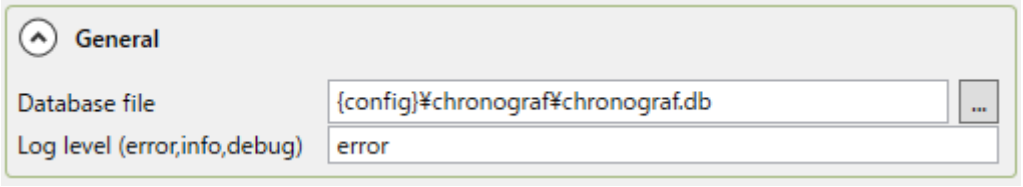
③ **Kapacitor Connection**

Kapacitor は、ストリーミング解析や異常検知に使用できる InfluxDB 用のデータ処理エンジンです。



Kapacitor の URL、ユーザ名、パスワードは、ここで入力、編集することができます。

④ **全般（General）**



Database file：Chronograf のデータベースファイル名。{config} は、DataHub の設定フォルダ、{install} は、DataHub のインストールフォルダと置き換わります。

Log level (eroor, info, debug)：Chronograf がログファイルに書き出すメッセージの重要度を指定します。

2．2．4．高度なトピック

他アプリケーションによって書き込まれたデータへのアクセス については、以下のリンクをご参照ください。DataHub プログラムは、他のアプリケーションによって書き込まれた InfluxDB データを読み取ることができますが、特にストア＆フォワードについてはいくつか制限があります。

マニュアル：こちら（メーカ英文マニュアル）

2．3．通知（Notification）

概　要

データ変更に基づいて、アラーム、イベントおよび条件に対する通知を生成することができます。条件が満たされると、データポイントの変更、電子メール、SMS、OPC A＆E またはソーシャルメディアメッセージを送信することによって通知を発信することができます。さまざまな状態や条件に対してテンプレートを定義し、データにバインドしてメッセージを送信、カスタムスクリプトを実行することができます。

特　徴

- OPC A＆E を作成、集約および送信します。
- DataHub を OPC A&E および OPC UA A&C サーバに変えます。
- OPC A&E と OPC UA A&C 間を変換します。
- 電子メール、SMS、ソーシャルメッセージを生成します。
- DataHub イベントに基づいて通知またはアラームを送信します。
- OPC UA、DA、A＆E、A＆C サーバまたはクライアント、Modbus スレーブ、ODBC データベース、Excel およびカスタムプログラムからのデータを扱うことができます。

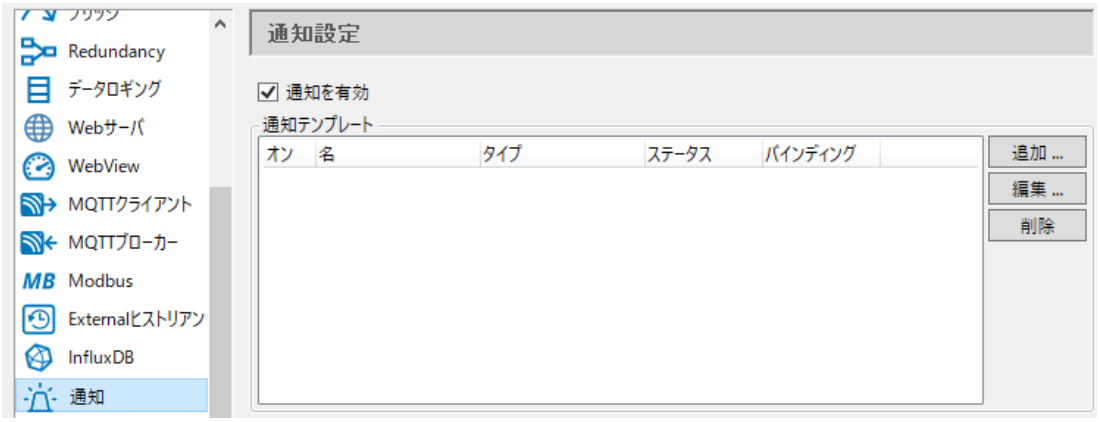
マニュアル：こちら（メーカ英文マニュアル）

このオプションの使用を開始するための手順については、通知の作成 をご参照ください。

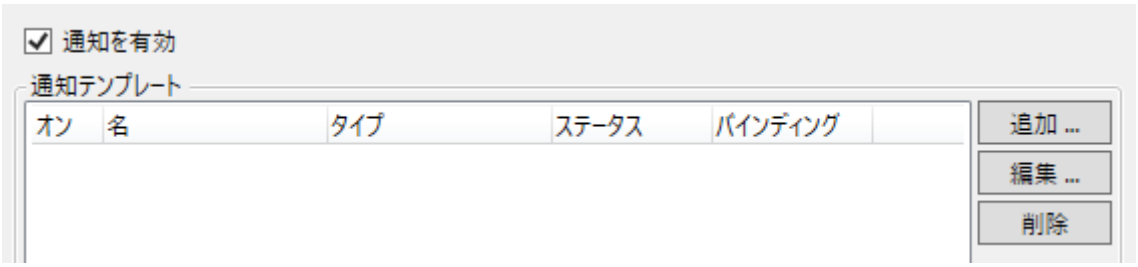
2．3．1．通知設定

DataHub の通知をクリックし、通知設定プロパティ画面を開きます。

通知を有効：ここチェックを入れると通知機能が有効になります。



2．3．1．1．通知テンプレート

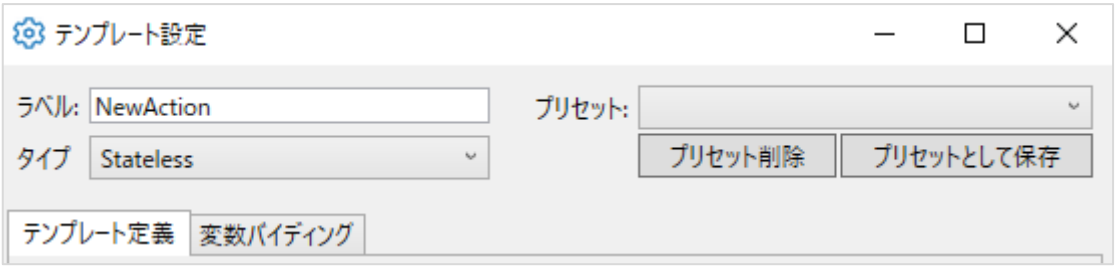


追加：通知テンプレートを設定します。

編集：通知テンプレートの設定を編集します。

削除：通知テンプレートの設定を削除します。

テンプレートは、［テンプレート設定］画面で作成および変更します。



ラベル：このテンプレートを識別するテキスト文字列。

タイプ：このテンプレートのタイプ。

Stateless（ステートレス）：定義された各状態の各条件を順番に調べ、最初に「真」になった条件の IfTure スクリプトを実行する。

データポイントの変更により、最大 1 つの IfTure スクリプトが実行される。これは、ポイント変更ごとに、ポイント値に応じたアクションを実行したい場合に有効です。

Stateful（ステートフル）：定義された各ステートの Entry Condition を順番に調べ、最初の True Condition に対して OnEntry スクリプトを実行する。ステートに入ると、そのステートの Exit Condition が満たされるまで、それ以上のスクリプトは実行されません。Exit Condition が満たされると、OnExit スクリプトが実行され、各 Entry Condition が再び調べられ、新しいステートに入る可能性があります。

データポイントが変更されると、最大で 1 つの OnExit スクリプトと 1 つの OnEntry スクリプトが処理されることになります。これは、OPC A＆E 条件と同様に、複数の相互に排他的な状態からなる条件に使用されます。

Every State（すべての条件）：定義されたすべての状態の条件を調べ、その条件が「真」であれば、IfTure スクリプトを実行します。

１回のデータ変更で複数の IfTure イベントがトリガーされる可能性があります。これは、ポイント変更毎に再評価される条件/アクションを指定したい場合に使用します。

OPC A&E Condition（OPC UA A&E コンディション）：ステートフルテンプレートの形式の一つで、OPC A&E コンディションのデータポイント形式を実装したデータポイントを生成します、OPC A&E サーバ機能は、このポイントを利用して OPC 条件を生成し、接続されている OPC A&E クライアントに送信します。また、本テンプレートは、各コンディションの OPC A&E 確認ポイントも監視して、外部クライアントからの確認に応答します。これは、プロセスデータから OPC A&E アラームを生成および管理する場合に使用されます。

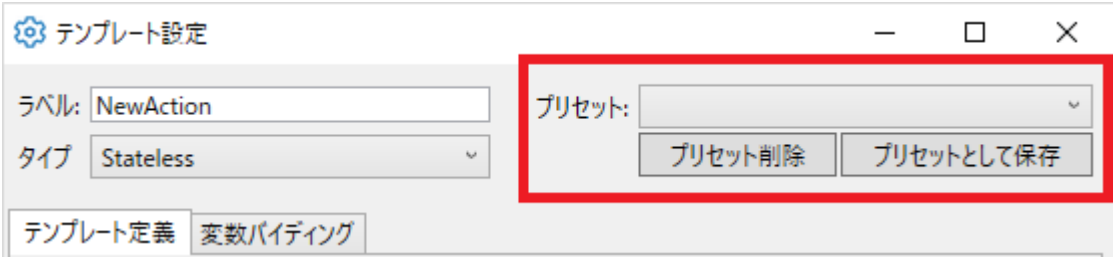
OPC A&E Event（OPC A&E イベント）：これは、OPC A&E Simple Event を実装するデータポイントも生成する Stateless テンプレートの形式です。
これは、プロセスデータから OPC A&E SImple Event を生成する場合に使用されます。

OPC A&E Condition Alerter（OPC A&E コンディション アラーター）：これは、外部の OPC A&E サーバーが提供する OPC A&E コンディションポイントを監視するステートレステンプレートの形式の一つです。A＆E コンディションイベントが発生すると、このテンプレートはコンディション情報を抽出し、コンディションスクリプトと IfTrue スクリプトを利用できるようにし、A＆E 情報に基づく通知を可能にします。
外部の OPC A＆E サーバーからの OPC A&E イベントに基づいて、メールや SMS などを送信したい場合に使用します。

OPC A&E Event Alerter（OPC A&E イベント アラーター）：これは、外部 OPC A&E サーバーによって提供される OPC A&E Event ポイントを監視する Stateless テンプレートの形式の一つです。OPC A&E Simple イベントや Tracking イベントが発生すると、このテンプレートはイベント情報を抽出し、コンディションスクリプトや IfTrue スクリプトで利用できるようにします。
これは、外部の OPC A&E サーバーからの OPC A&E イベントに基づいて電子メール、SMS などを送信する場合に使用されます。

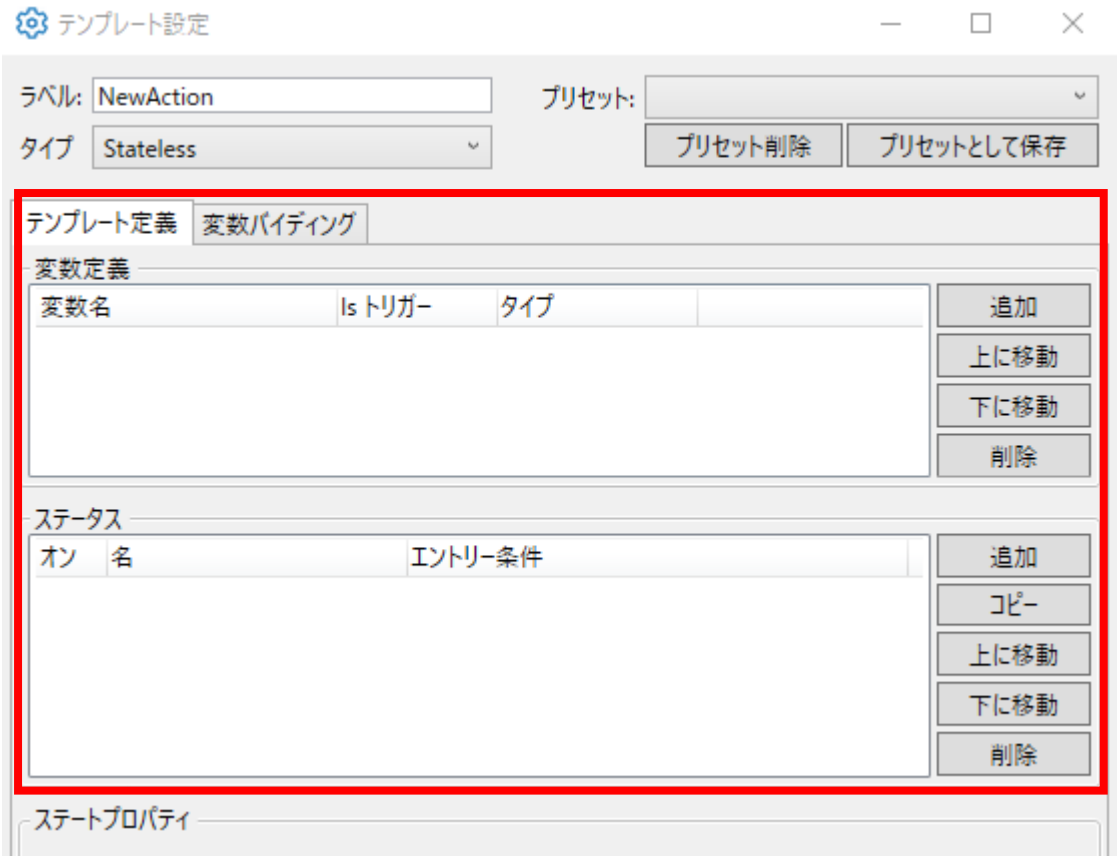
プリセット

プリセットとして保存することで、任意の通知テンプレートを再利用するために保存することができます。一度保存されると、テンプレート名はプリセットドロップダウンリストに表示されます。その後、プリセットリストから新しいテンプレートを選択し、修正することができます。テンプレートを変更しても、プリセットは変更されません。



テンプレート定義

DataHub のインスタンスでは、１つのソースから多くの類似した通知を作成できるようにテンプレートを使用しています。各テンプレートは、ステートとスクリプトに関連する変数で構成され、ポイント値を割り当てることができます。



A&E 設定：タイプ選択が「OPC A＆E Condition」、「OPC A＆E Event」の場合表示されます。

◆タイプ選択が「OPC A＆E Condition」の場合。

A&E データドメイン：OPC A&E データ用に設定された DataHub のデータドメインです。OPC A&E が書き込まれるのは、このデータドメインです。OPC A&E 機能もインストールされている場合は、これらのイベントは OPC A&E クライアントで利用可能です。

条件名：OPC A&E コンディションの場合のみ、このコンディションの名称を指定します。

ソース（スクリプト）：スクリプトをトリガーできる A&E コンディションまたはイベントのソースです。以下の「イベント内容（スクリプト）」をご参照ください。

カテゴリ：このコンディションまたは、イベントの OPC A&E カテゴリーです。

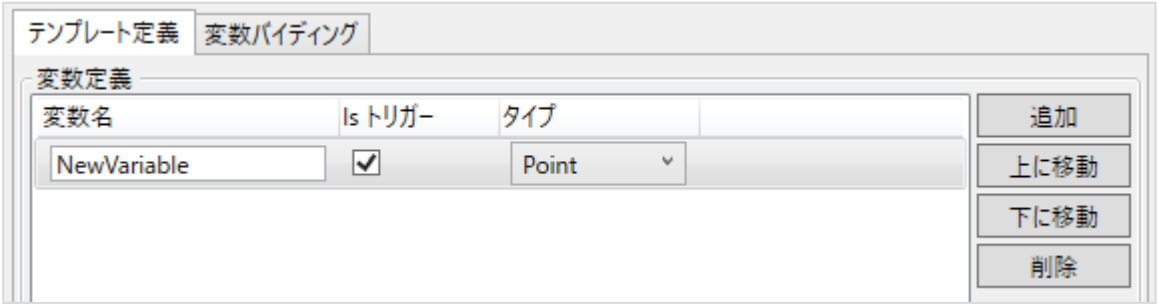
OnAcknowlege スクリプト：OPC A&E コンディションの場合のみ、この条件が承認された時に実行されるスクリプトです。

◆タイプ選択が「OPC A&E Event」の場合。

A&E データドメイン：PC A&E データ用に設定された DataHub のデータドメインです。OPC A&E が書き込まれるのは、このデータドメインです。OPC A&E 機能もインストールされている場合は、これらのイベントは OPC A&E クライアントで利用可能です。

カテゴリ：このコンディションまたは、イベントの OPC A&E カテゴリです。

変数の定義



変数名：この名前はスクリプトの識別子として使用されます。スペースを含まない、文字、数字、アンダースコア（_）文字のみのテキスト文字列。

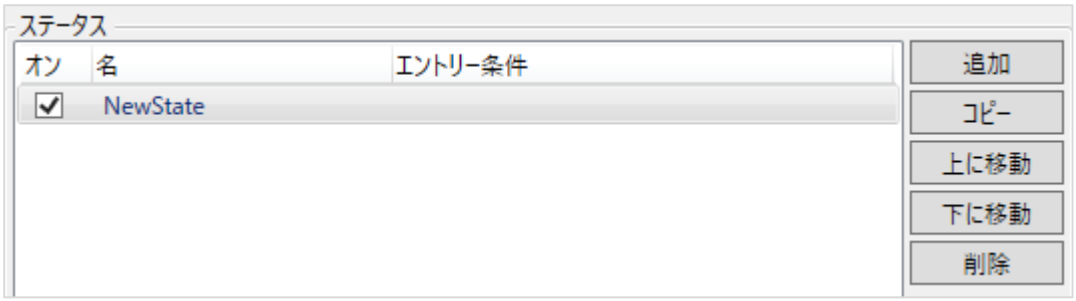
Is トリガー：ここにチェックを入れると、この変数に割り当てられたポイントが変化すると、ステートの再評価が実行されます。トリガーにできるのは、「Point」のみです。

タイプ：「Point」、「Integer」、「Double」、「String」から選択。

- Point：値、品質、およびタイムスタンプが、条件およびアクションスクリプト使用できる DataHub ポイント。
- Integer：64 ビットの整数リテラル
- Double：倍精度の浮動小数点リテラル
- String：文字列

ステータス

通知は、システムの状態に基づいており、ポイント値やその他の要因によって決定されます。



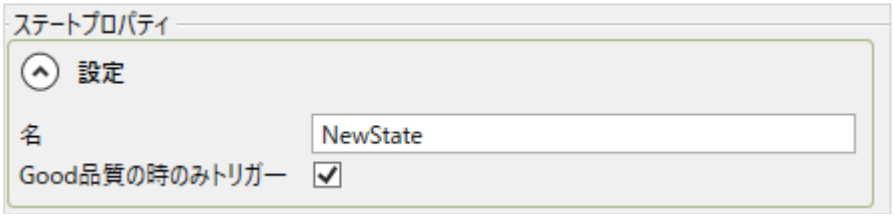
各状態には、「名（Name）」があり、通常は、「エントリー条件」があり、以下の「ステータスプロパティ」で定義されています。ステータスは、リストの上から順に評価されます。「上に移動」ボタンと「下へ移動」ボタンを使用して、評価の順序を変更できます。

ステートプロパティ

使用可能なステートプロパティは、テンプレートに定義されたタイプによって異なります。

設定

設定では、「**名**（Name）」は状態を識別するテキスト文字列です。



名：状態を識別するテキスト文字列です。

Good 品質の時のみトリガー：ここにチェックを入れると、品質が Good 以外の場合のポイントの変更を無視します。

スクリプト

関連する条件が満たされると、スクリプトが実行されます。スクリプトと条件のオプションは、通知の種類によって異なります。

◆タイプ選択が「Every State」、「OPC A＆E Condition Alerter」、「OPC A＆E Event Alerter」、「Stateless」の場合表示される。

条件が満たされるたびに、**IfTrue** スクリプトが実行されます。

◆タイプ選択が「OPC A＆E Condition」、「OPC A＆E Event」、「Stateful」の場合表示される。

OnEntry スクリプトは、アクションがその状態にある間に**エントリー条件**が満たされるたびに実行されます。**OnExit スクリプト**が指定されいない場合は、**終了条件**は、**エントリー条件**の逆になるように定義されます。

イベント内容（スクリプト）Event Setting(Script)

スクリプトとして定義されるイベント設定プロパティでは、トリガーポイント名などのアクション固有の情報を含む文字列を作成することができます。

例えば、トリガー変数が MyTrigger と呼ばれる場合、ソース (Script) (上記参照)を MyTrigger.Name と Message として定義すると便利でしょう。

```
MyTrigger.Name + " has reached " + MyTrigger.Value
```

◆タイプ選択が「OPC A＆E Condition」の場合表示される。

メッセージ：A&E の状態に関連するメッセージ。

説明：A&E 状態の説明。

定義：A&E 条件の定義。

◆タイプ選択が「OPC A＆E Event」の場合表示される。

メッセージ：A&E のイベントに関連するメッセージ。

ソース：実行されるスクリプト。

イベント設定

◆タイプ選択が「OPC A＆E Condition」の場合表示される。

重要度：A＆E コンディションの重要度。重要度は、1～1000 の範囲でなければならない。

Ack 必要：OPC A&E コンディションがこの状態になった場合、確認応答を要求するように設定します。

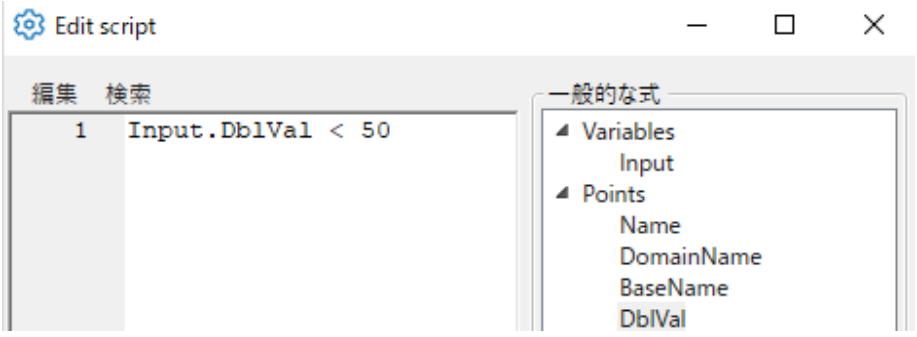
非アクティブ状態：この A＆E コンディションのこの状態を非アクティブ状態に設定します。OPC A&E クライアントは、このインジケータを使用して、条件をオペレータに表示するかどうかを決定します。

◆タイプ選択が「OPC A＆E Event」の場合表示される。

重要度：A＆E イベントの重要度。重要度は、1～1000 の範囲でなければならない。

スクリプト

スクリプトは、コンディションの基準を設定し、通知アクションを割り当てるために使用されます。



スクリプトの言語は、S#です。S#については、こちら に記載されています。

一般的な計算式

スクリプトエディターには、一般的なスクリプトの数式と

- **Variables**：このテンプレートで定義したすべての変数。
- **Points**：Points: 変数が Point 型の場合、変数名にドット（．）構文を使用してアクセスされる DataHub ポイントのプロパティ。
- **Application**：アプリケーションによって提供される機能。
- **Notifiers**：設定したすべての通知者（Notisiers）
- **LogLevel**：app.Log 関数の「レベル」引数で使用されます。
- **PointQuality**：DataHub のポイント品質の各可能な値。これは、app.WritePoint 関数で quality 引数として使用されます。
- **Math**：一般的な数学関数の選択。
- **Statistics(統計)**：app.Statistics 関数を使用して追跡されているポイントの利用可能な統計。

通知スクリプトは、スライディング画面統計を使って、より複雑な条件を支援することができます。たとえば、あるデータ・ポイント値がしきい値を超え、過去 60 秒間に強く上昇する傾向がある場合にのみ通知を生成したい場合があります。そのためには、その点の最近の値の傾きを知る必要があります。

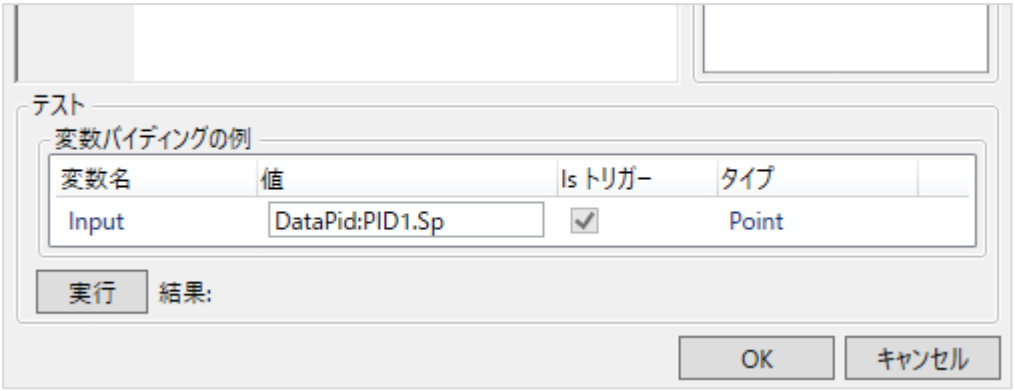
スクリプトに app.Statistics(point, seconds) という式を含めると、スクリプトエンジンは自動的に、スクリプトの最初の実行から、秒単位で指定した期間、そのポイントの統計情報の収集を開始します。その後、同じ数式を呼び出すと、期間（統計情報の収集が期間より短い場合はそれ以下）にわたって計算された、利用可能なすべての統計的測定値が利用できるようになります。利用可能な統計値は以下の通りです。

- Average：期間内のすべての測定値の単純平均。
- TimeWeightedAverage：期間内のすべての測定値の時間加重平均。より長い時間値を保持する測定値は、この平均に大きな影響を与えます。
- Count：期間内の測定回数。
- Slope：勾配。一定期間にわたる線形回帰 Y=mX+B の勾配で、delta Y/day を生成します。
- Intercept：期間中の線形回帰 Y＝mX+B の X 切片、Y＝0 の日付を生成します。

・SampleStdDev：期間内のすべての測定値のサンプル標準偏差（N-1）
・SampleVariance：サンプル分散（N-1）。
・PopulationStdDev：母集団の標準偏差（N）。
・PopulationVariance：母集団分散（N）

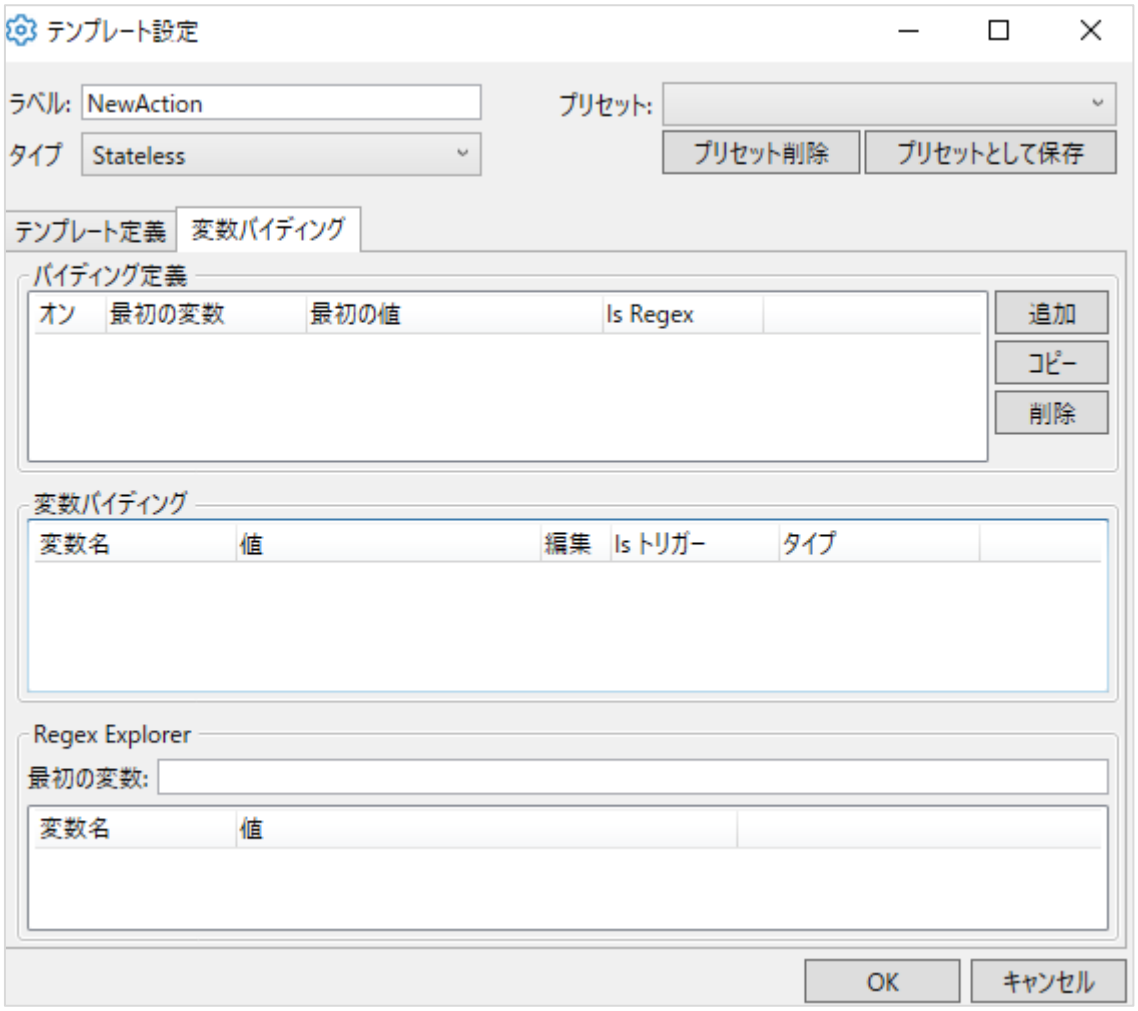
テスト

変数の値を入力して、「実行」ボタンを押すと、スクリプトをテストすることが出来ます。これにより、トリガー条件が満たされたかのように、テスト領域に入力した変数値を使用して、このスクリプトだけが実行されます。すべての変数に値を指定する必要があります。スクリプトに電子メールや SMS メッセージの送信などの副作用がある場合は、その副作用が発生します。



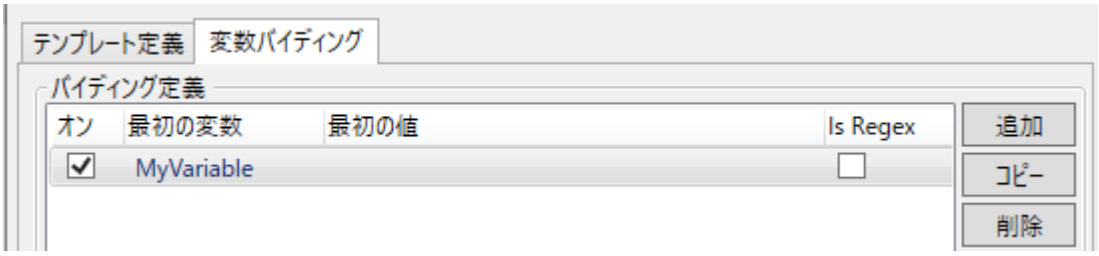
変数バインディング

DataHub ポイントとその他の値はテンプレートにバインドされ、必要に応じて通知を再利用および再構成する簡単な方法を提供します。バインドは、ポイントまたは値をテンプレート変数に割り当てることによって行われます。テンプレート定義セクションでは、データ ポイントと値のプレースホルダーを提供しています。この変数バインディング セクションでは、これらのプレースホルダーに対して 1 つ以上の具体的な値のセットを作成しています。



バイディング定義

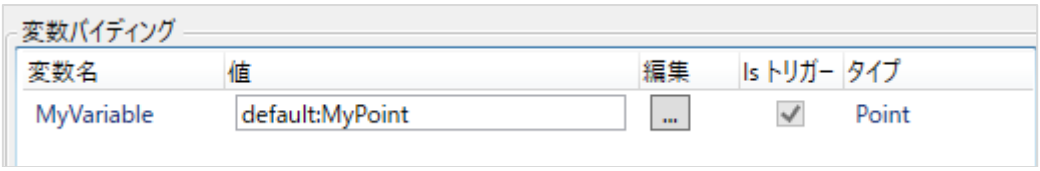
バインディング定義は、アクションテンプレートの単一のインスタンスのためのすべての具体的な値のセットです。つまり、バインディング定義リストの各行は、テンプレート内の変数に対するすべての値を指定し、各行はテンプレートの個別のインスタンスを構成します。たとえば、Tank1 と Tank2 の水位に基づいて通知を生成したい場合、Tank1 と Tank2 のそれぞれについて個別のバインディング定義を作成することになります。テンプレートは、一般的な Tank に対する通知の処理方法を指定し、2 つのバインディング定義は、Tank1 と Tank2 にそれを適用します。



各バインディング定義を識別しやすいように、このリストには最初の変数バインディングのみが表示されます。

変数バインディング

変数は、以前に定義された 各変数のタイプ (DataHub ポイント、整数、倍精度、または文字列) に従って値を入力することによってバインドされます。



バインディング定義の最初の変数は、他の変数とは異なる扱いを受けます。この変数は、単純な文字列ではなく、正規表現として定義することができます。この正規表現は任意の数のデータ ポイントにマッチさせることができ、各マッチは個別のバインディング定義として扱われます。これにより、パターンにマッチするすべてのデータポイントに適用される単一のバインディング定義を作成することができます。正規表現では、.NET Regex 構文を使用します。正規表現がポイント名の任意の部分に一致する場合、比較は成功します。したがって、文字列の完全一致を求める場合は、正規表現が ^ および $ 文字 (それぞれ文字列の開始と終了) で囲まれていることを確認します。

例えば、2 つのタンクがあり、そのレベルがポイント Plant:Tank1.level と Plant:Tank2.level で見える場合、パターン Plant:Tank[0-9]+key.level は、名前が Plant:Tank で、任意の数字のシーケンスと、文字列 .level が続くすべてのポイントにマッチします。このように、この 1 つのバインディングは両方のタンクに適用されます。後で Plant:Tank3.level という名前の 3 番目のタンクを追加すると、通知設定を変更することなく、このバインディングが自動的に適用されます。

最初の変数バインディングが正規表現である場合、その正規表現に関連する他の変数バインディングも作成したい場合があります。例えば、Plant:Tank1.alarm という新しいポイントを作成し、通常運転時には 0、レベルが高すぎる時には 1 という値を持つようにしたいとします。この場合、Input と Output という名前の 2 つの変数を用意します。Input は Plant:Tank1.level、Output は Plant:Tank1.alarm です。通知テンプレートの OnEnter と OnExit スクリプトは、0 または 1 の値を適切に Output 変数に書き込みます。

しかし、正規表現を使用してすべてのタンクにマッチさせる場合、Input が Plant:Tank1.level のとき Output は Plant:Tank1.alarm であり、Input が Plant:Tank2.level のとき Output は Plant:Tank2.alarm というような言い方が必要です。これを実現するには、正規表現のキャプチャを使用します。キャプチャとは、入力文字列のうち、正規表現の一部にマッチする部分のことである。キャプチャは正規表現中の括弧で示される。

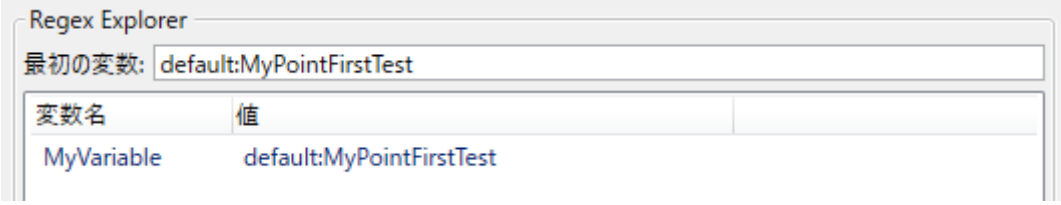
たとえば、タンク表現 Plant:Tank[0-9]+Think.level において、数値の部分表現を特定し、それを使用して Plant:TankN.alarm を定義したいとします（N は任意の数です）。そのためには、式の数値部分を括弧で囲みます。Plant:Tank([0-9]+)¥.level. これにより、Tank の後に続く数字だけを含むキャプチャが作成されます。正規表現では複数のキャプチャが可能で、0 から順番に番号が振られます。

さて、レベルポイントに対応するアラームポイント名を構築するために、Output を Plant:Tank{0}.alarm と定義することができます。{N}構文は、Input 変数の正規表現から N 番目のキャプチャを Output 変数のバインディングのその位置に代入するようにバインディングに指示します。この例では、最初のキャプチャである {0} が必要です。最初の変数のバインディングでは {N} 構文を使用できません。これは、キャプチャを生成する正規表現のバインディングだからです。キャプチャは文字列なので、数値の.NET フォーマット置換修飾子は使用できません。

.NET 正規表現の使用と構文の説明については、こちらを参照してください。例については、通知の作成の章の正規表現を参照してください。

Regex Explorer

Regex Explorer を使用して、正規表現がどのように実装されているかを確認できます。

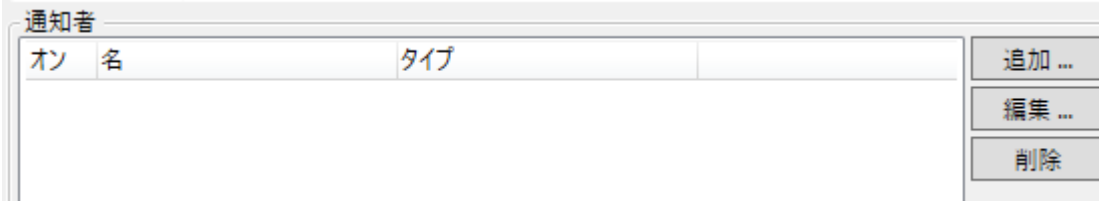


「最初の変数」入力フィールドに、データポイントの名前を入力します。正規表現の基準を満たす場合は、「値」列に表示されます。バインディング正義内の他のすべての変数が表示され、

キャプチャ置換が表示されるため、正規表現とそのキャプチャの両方が意図したものであることを確認できます。

2．3．2．1．通信者

通知テンプレートを追加するには、「追加...」ボタンを押します。

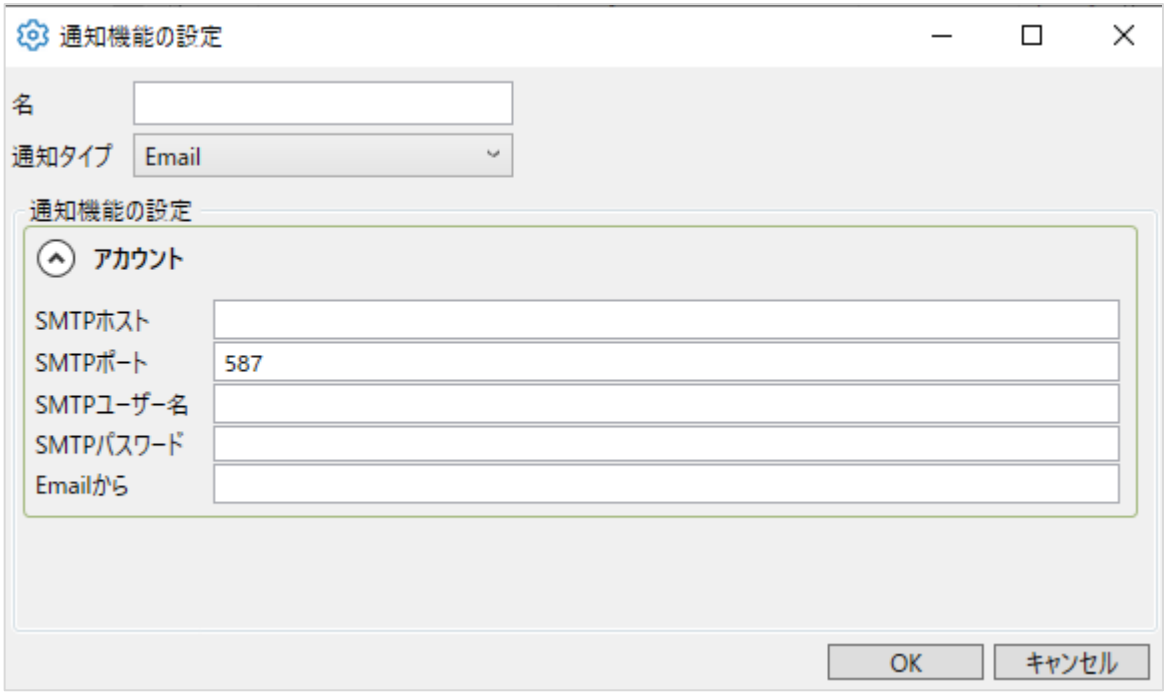


追加：通知テンプレートを設定します。

編集：通知テンプレートの設定を編集します。

削除：通知テンプレートの設定を削除します。

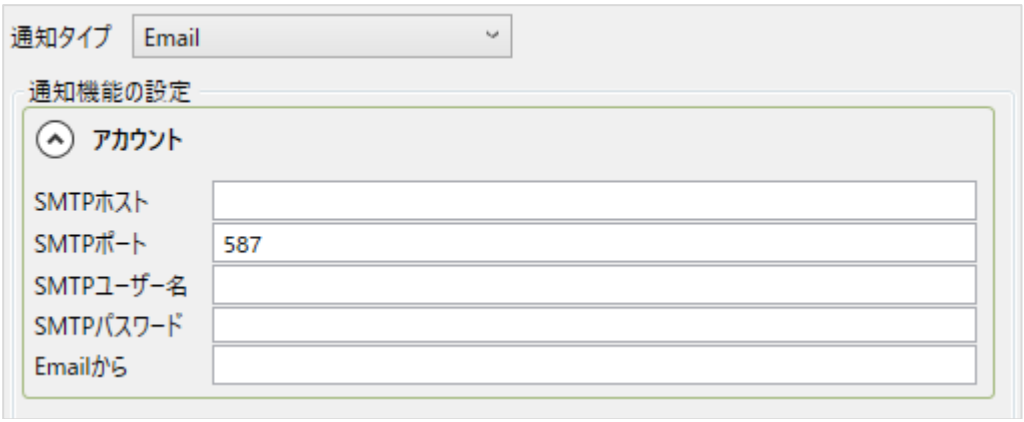
通知設定は、［通知機能の設定］画面で設定します。



この通知設定に名前を「**名**」に入力します。この名前は、スクリプトの識別子として使用できるように、文字、数字、アンダースコア（_）文字のみで構成され、スペースやその他の文字を含まないテキスト文字列である必要があります。

次に、「**通信タイプ**」を選択し、必要な詳細を入力します。

E-メール



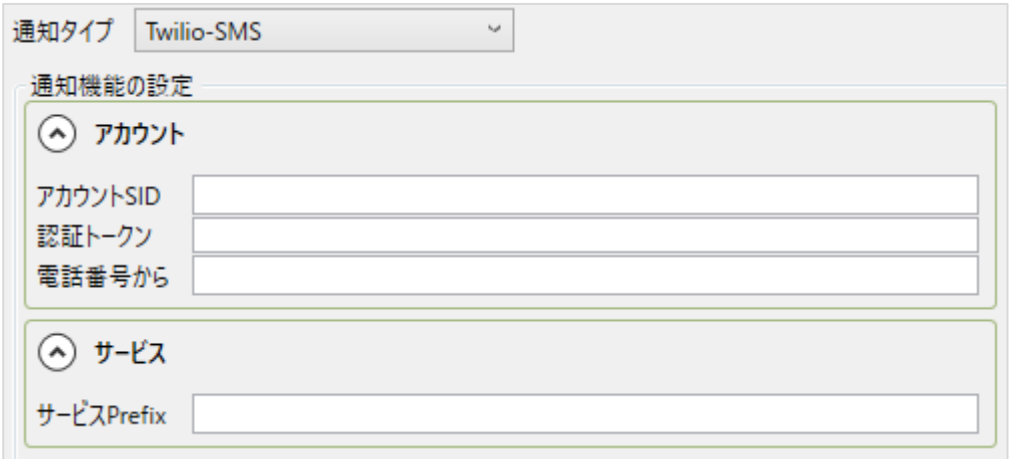
「**SMTP ホスト**」、「**SMTP ポート**」、「**SMTP ユーザー名**」、「**SMTP パスワード**」:
電子メールの送信元の SMTP アカウントの入力情報。

Email から：電子メールの送信元の電子メールアドレス。

HTTP

このオプションを使用すると、SMS ゲートウェイなどの HTTP サーバーにデータを送信できます。
※只今検証中です。

Twilio-SMS



アカウント

「**アカウント SID**」、「**認証トークン**」、「**電話番号から**」: テキストメッセージの送信元アカウントのサービス ID、認証トークン、電話番号。この情報は、Twilio アカウントから入手できます。Twilio アカウントが、Skkynet のよって管理されている場合は、この情報を Skkynet から受け取っているはずです。

サービス

サービス Prefix（接頭辞）：使用しているサービスの Twilio サービス プレフィックス。Twilio-SMS および Twilio-WhatsApp のデフォルトは空白です。より多くの Twilio サービスが利用可能になると、このサービス プレフィックスを使用して、メッセージを他のソーシャルネットワークや配信メカニズムに送信できます。

2．4．OPC UA A&C

概　要

OPC UA A&C サーバーとクライアントに接続し、OPC A&E と OPC A&C を変換します。レガシーソフトウェアを最新の OPC UA A＆C クライアントおよびサーバーに接続することができます。

特　徴

- OPC UA A&C サーバーとクライアントを提供します。
- OPC UA A&C と OPC A&E 間を変換します。
- トンネリング経由で OPC UA A＆C のサーバとクライアントを接続します。
- DataHub がサポートしている他のプロトコルを介して、OPC UA A&C データを利用が可能です。
- OPC UA A&C データをデータベースやヒストリアンに保存します。

マニュアル：

- OPC UA：こちら（メーカ英文マニュアル）
- OPC A&C から OPC A&E 変換ガイドライン：こちら（メーカ英文マニュアル）

2．5．構成インポート

概　要

実際のシステムでは、PLC の何千、何万ものポイントを設定する必要があります。この「構成インポート機能」を使って、CSV 形式の設定ファイルからポイントを一括設定することができます。

特　徴

- ファイルをインポートするには、リモート設定（Remote Config）機能を使用します。
- 対応機能：OPC DA、OPC UA、Modbus、MQTTクライアント、ブリッジ、外部ヒストリアン、ヒストリアン

マニュアル

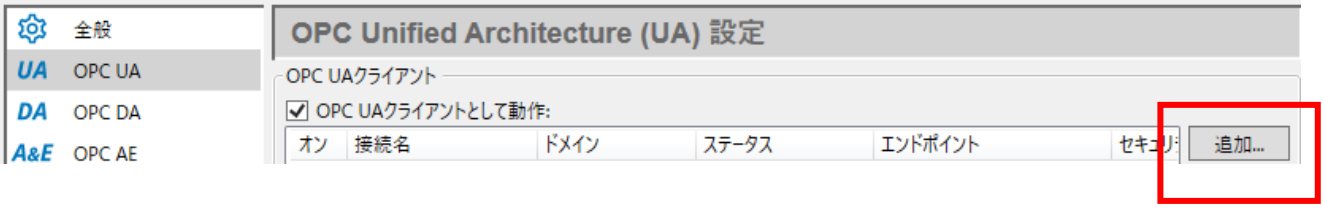
構成のインポート：こちら（メーカ英文マニュアル）

リモート設定：こちら（メーカ英文マニュアル）

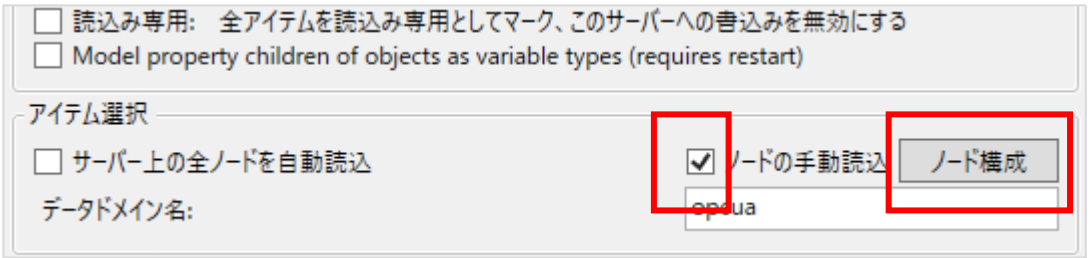
2．5．1．設定ファイルをインポートする

① 「リモート設定ツール」で DataHub に接続する。

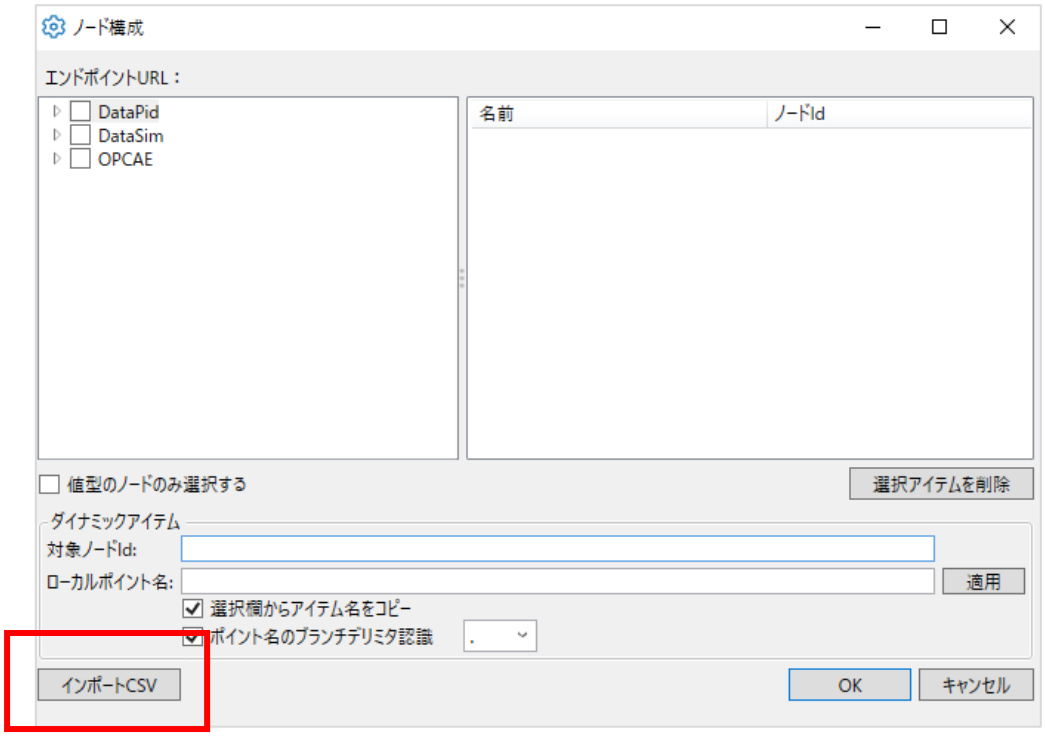
② OPC UA オプションで、「追加」ボタンを押します。

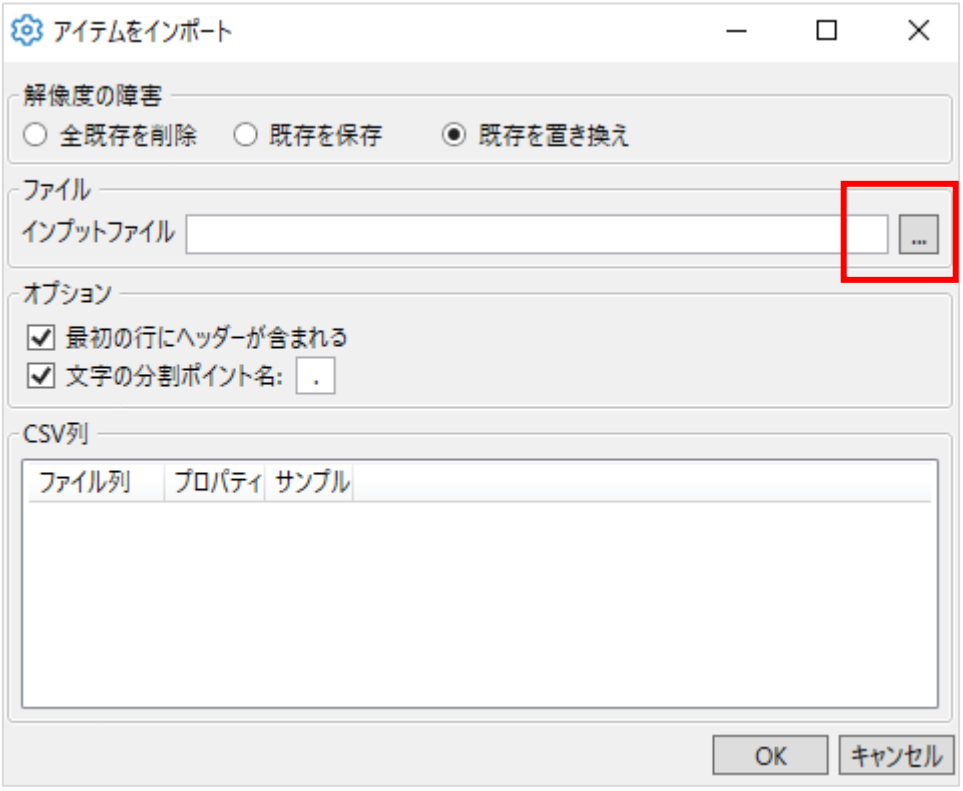


③ 「OPC UA データアクセスサーバー構成」画面にて、「ノードの手動読込」にチェックを入れ、「ノード構成」ボタンを押します。



④ 「ノード構成」画面の左下にある、「インポート CSV」ボタンを押して、設定ファイルを指定します。


ノード構成
エンドポイントURL：
DataPid
DataSim
OPCAE
名前
ノードId
値型のノードのみ選択する
選択アイテムを削除
ダイナミックアイテム
対象ノードId:
ローカルポイント名:
適用
選択欄からアイテム名をコピー
ポイント名のブランチデリミタ認識
インポートCSV
OK
キャンセル


アイテムをインポート
解像度の障害
全既存を削除
既存を保存
既存を置き換え
ファイル
インプットファイル
...
オプション
最初の行にヘッダーが含まれる
文字の分割ポイント名:
.
CSV列
ファイル列
プロパティ
サンプル
OK
キャンセル

2．5．2．設定ファイル

OPC UA の設定ファイルには、「NodeId」と「PointName」プロパティを記載し、CSV ファイル形式で保存します。

NodeId	PointName
ns=2;s=DataSim:Ramp	DataSim.Ramp
ns=2;s=DataSim:Offset	DataSim.Offset
ns=2;s=DataSim:Sine	DataSim.Sine
ns=2;s=DataSim:Square	DataSim.Square

設定可能なプロパティ

・プロパティリスト

各プロトコルで設定可能なプロパティのリストを記載します。

OPC UA

プロパティ	タイプ	レンジ	説明
NodeId	string	any	OPC UA サーバから提供される NodeId
PointName	string	unqualified point	NodeId に対して作成する DataHub 内のデータポイント名

OPC DA

プロパティ	タイプ	レンジ	説明
ItemID	string	any	OPC DA サーバから提供される ItemID
PointName	string	unqualified point	ItemID に対して作成する DataHub 内のデータポイント名

Bridging

プロパティ	タイプ	レンジ	説明
Disabled	boolean	0 or 1	「1」に設定すると、このブリッジが無効であることを示す。
Source	string	qualified point	送信元ポイントのフルネーム
Destination	string	qualified	デスティネーションポイントのフルネーム

プロパティ	タイプ	レンジ	説明
		point	
Forward	boolean	0 or 1	「1」に設定すると、方向へブリッジします
ForceConsistency	boolean	0 or 1	「1」に設定すると、順方向に強制的に整合させることができます
Inverse	boolean	0 or 1	「1」に設定すると、逆方向のブリッジを行う ForceConsistency が「1」の場合、「1」にすることはできません
DirectCopy	boolean	0 or 1	「1」に設定すると、変換無しで直接コピーされます
LinearTransform	boolean	0 or 1	「1」に設定すると、線形変換が行われます
Multiply	double	number	線形変換の乗算器
Add	double	number	線形変換の加算器
LinearRangeMap	boolean	0 or 1	線形範囲マッピングに「1」を設定します
SrcMin	double	number	線形範囲マッピングのための、ソース最小値
SrcMax	double	number	線形範囲マッピングのための、ソース最大値
DstMin	double	number	線形範囲マッピングのための、ディスティネーション最小値
DstMax	double	number	線形範囲マッピングのための、ディスティネーション最大値
ClampMin	boolean	0 or 1	「1」に設定すると、線形レンジマッピングのディスティネーション値が最小値でクランプされます
ClampMax	boolean	0 or 1	「1」に設定すると、線形レンジマッピングのディスティネーション値が最大値でクランプされます

MQTT Client

プロパティ	タイプ	レンジ	説明
PointName	string	qualified point	MQTT サーバにプッシュするポイントのフルネーム

Modbus

プロパティ	タイプ	レンジ	説明
Point or Range	String	point, range	このエントリが単一のポイントまたは

プロパティ	タイプ	レンジ	説明
			ポイントの範囲のどちらであるかを示すポイントまたは範囲
Point Name	string	unqualified point	ポイント名またはポイント式
Block	string	MB_AI, MB_AO, MB_DI, MB_DO	このアドレスの I/O ブロック
Address	string	address expression	ブロックへのオフセットとしてのこの項目の数値アドレス
Type	string	type expression	D データ型とフラグ（Modbus のドキュメントを参照）
OneBasedAddress	boolean	0 or 1	アドレスが 1 ベースであれば「1」、そうでなければ「0」を設定
OneBasedBitAddress	boolean	0 or 1	ワード内のビット番号が 1 ベースであれば 1、そうでなければ 0
Allow Write	boolean	0 or 1	DataHub がこのアドレスに書き込み可能な場合は 1、そうでない場合は 0
Deadband	double	any	変換後の値に適用される絶対デッドバンド。このデッドバンド以下の変化は無視されます。
Transform	string	Direct, Linear, Range	適用する変換の種類
Multiply	double	number	線形変換の乗算器
Add	double	number	線形変換の乗算器
Modbus Min	double	number	線形範囲マッピングの Modbus 最小値
Modbus Max	double	number	線形範囲マッピングの Modbus 最大値
Point Min	double	number	線形範囲マッピングのための、ポイントの最小値
Point Max	double	number	線形範囲マッピングのための、ポイントの最大値
Clamp Min	boolean	0 or 1	「1」 に設定すると、線形範囲マッピングの最小値で Modbus 値をクランプ

プロパティ	タイプ	レンジ	説明
			する
Clamp Max	boolean	0 or 1	「1」 に設定すると、線形範囲マッピングの最大値で Modbus 値をクランプする
Convert	boolean	0 or 1	ポイント タイプが Modbus タイプと異なる場合は、「1」 に設定します (たとえば、Modbus 整数を倍精度ポイントに変換する)
Converted Type	string	type expression	フラグを持たないデータ型 (例 I2 or R8).
Range Count	integer	> 0	Point または Range が range に設定されている場合の項目数。
Range Offset	integer	>= 0	Point または Range が range に設定されている場合の項目数。

External Historian

プロパティ	タイプ	レンジ	説明
NodeId	string	qualified point	ポイントのフルネーム

Historian

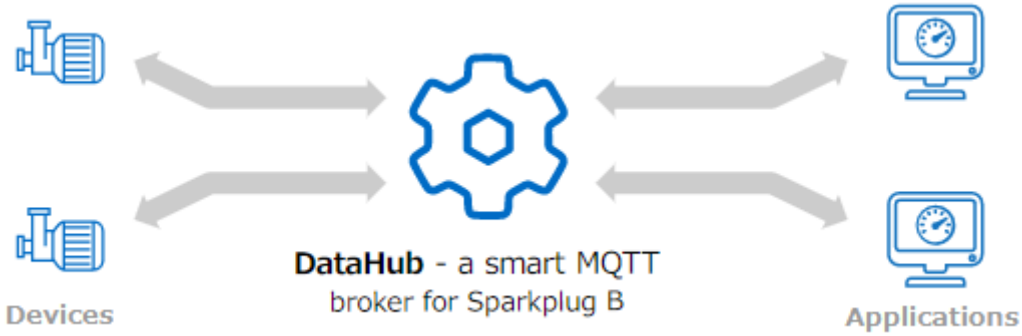
プロパティ	タイプ	レンジ	説明
NodeId	string	qualified point	Th ポイントのフルネーム

2．6．MQTT の強化

2．6．1．Sparkplug B

概　要

Sparkplug Bは、データ送受信方法を定義するMQTTの仕様です。ベンダー間の相互運用性の問題を解決するために導入されました。ネットワークのエッジにあるデバイスやセンサーは、Sparkplug Bを使用して、SCADAシステム、ヒストリアン、分析プログラムなどのアプリケーションと通信することができます。



DataHub プログラムは、Sparkplug B に対応したスマート MQTT ブローカーです。他の MQTT ブリーカーとは異なり、DataHub インスタンスは単にデータを渡すのではなく、Sparkplug B メッセージペイロードを解釈し、次の利点を提供します。

●エラーへの対応

DataHub プログラムは、デバイスから順不同または失われたメッセージを識別できます。その場合、DataHub プログラムはデバイスを切断し、再接続を許可します。これにより、デバイスはスタートアップ (BIRTH) メッセージを再送信し、すべての受信アプリケーションを再同期し、1 つのバージョンの真実を維持します。

●デバイスへの書き込みの失敗を解決

DataHub プログラムは、タイマーを使用して、デバイスへのすべての書き込み要求をチェックし、デバイス上でデータ値が実際に変更されたかどうかを確認できます。そうでない場合、DataHub プログラムはデバイスを強制的に切断し、BIRTH メッセージを再送信させることができます。これにより、そのデバイスをリッスンしているすべてのアプリケーションが再同期され、単一のバージョンの真実が維持されます。

●データ品質情報を追加

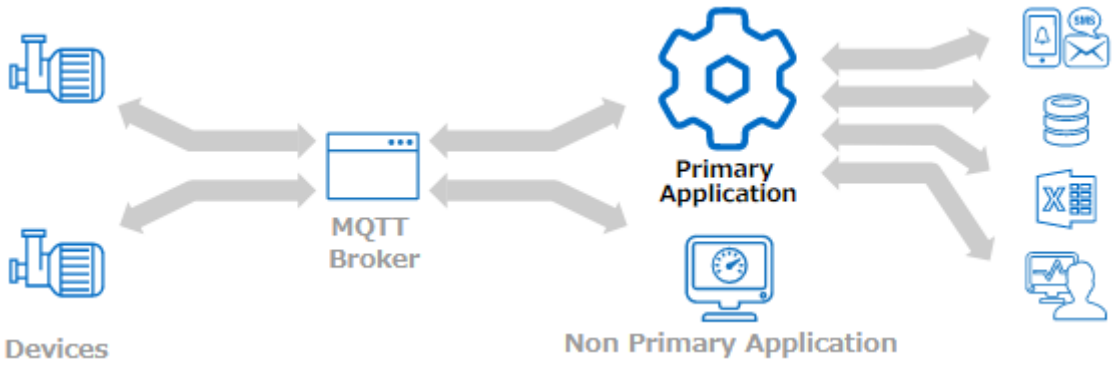
Sparkplug B を他のプロトコルに変換する場合、DataHub プログラムは品質情報を追加できます。たとえば、スパークプラグ B データを OPC に変換する場合、DataHub プログラムは OPC データ品質を追加できます。データ品質を BIRTH メッセージまたは DATA メッセージの場合は Good に設定し、DEATH (シャットダウン) メッセージの場合は Not Connected に設定できます。

●Sparkplug B 対応の汎用性の高いクライアント

DataHub プログラムは、任意の Sparkplug B クライアント タイプ (EoN デバイス、プライマリ アプリケーション、または非プライマリ アプリケーション) として構成することが可能です。

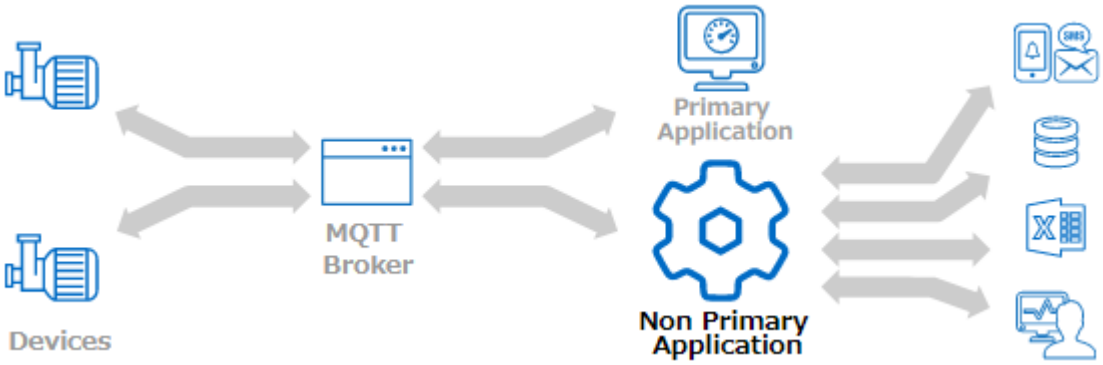
主な用途

●Primary Application：



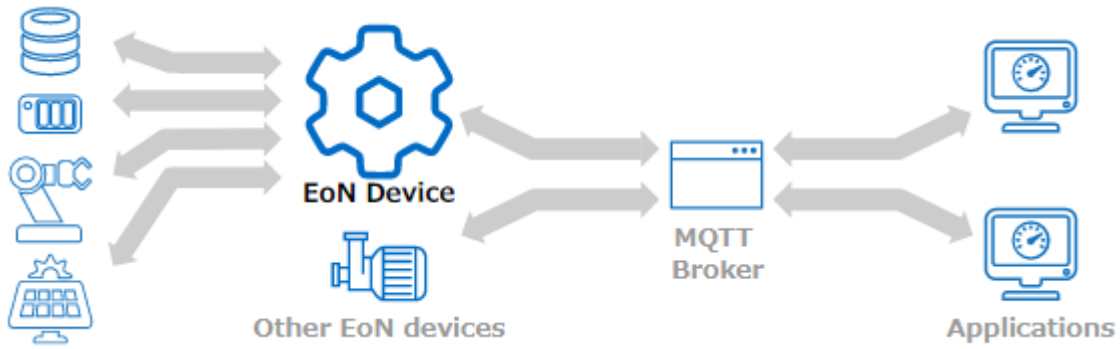
Sparkplug B ベースのシステムを構築する場合は、DataHub プログラムをプライマリアプリケーションとして使用することができます。WebView HMI、データロギング、ヒストリアン、Excel、アラーム、電子メール通知などと提供します。

●Non Primary Application：



同様に、すでにプライマリアプリケーションがある場合、DataHub プログラムは非プライマリアプリケーションとして Sparkplug B システムに接続し、WebView、データロギング、ヒストリアンと Excel の接続、アラーム、電子メール通知などのデータハブ機能を提供できます。

●EoN Device：
スパークプラグ以外のデータをスパークプラグ B システムにフィードする必要がある場合は、DataHub プログラムをエッジ オブ ネットワーク (EoN) デバイスとして接続することが可能です。これにより、OPC DA や UA、Modbus TCP、データベースデータ、カスタムプログラムなどのソースから Sparkplug B アプリケーションにデータを送信することが可能になります。



ホワイトペーパー：よりスマートな MQTT へ
https://www.ibress.com/support/white-papers/DataHub-WP-for-mqtt-smarter-is-better-20220329.pdf

マニュアル：MQTT クライアント　こちら（メーカ英文マニュアル）

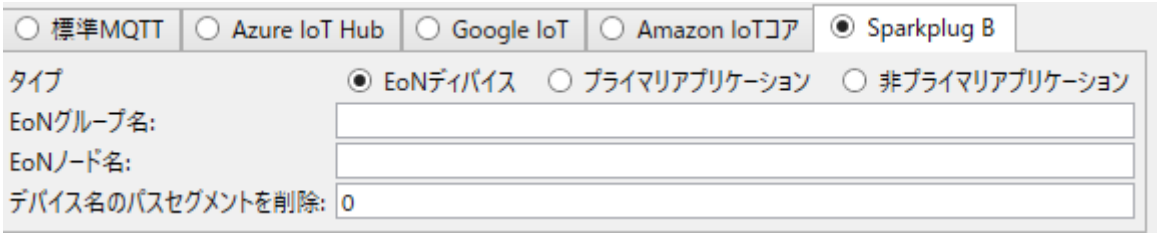
マニュアル：Sparkplug B クライアント　こちら（メーカ英文マニュアル）

マニュアル：MQTT Broker、Sparkplug B サーバ　こちら（メーカ英文マニュアル）

2．6．1．1．Sparkplug B クライアント

DataHub のプログラムは、Sparkplug B クライアントタイプの 「EoN Device」、「Primary Application」、「Non Primary Application」として構成することが出来ます。

［注意］Sparkplug B の識別文字列には、プラス記号（+）、フォワードスラッシュ（/）、ナンバー記号（#）以外の有効な UFT-8 英数字を含めることができます。



●EoN Device

この設定により、DataHub インスタンスは、Edge of Network デバイスとして機能するようになります。このモードでは、DataHub インスタンスは、データソースとして動作し、サポートされているあらゆるプロトコルと Sparkplug B 間のゲートウェイとして効果的に機能します。

EoN グループ名：EoN デバイス間で共有可能な識別子。類似したデバイスとグループ化する際によく使用されます。

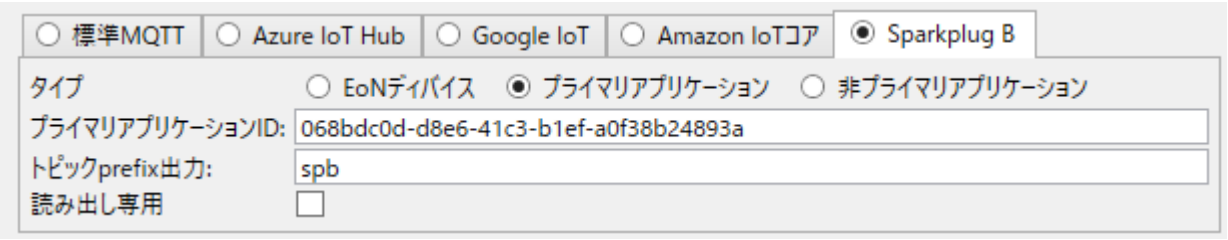
※［注］EoN グループ名と EoN ノード名（下記）の組み合わせで各 EoN デバイスを識別し、同じブローカーに接続しているすべての Sparkplug B クライアント間でユニークである必要があります。

EoN ノード名：グループ内でこのデバイスを識別するための識別子。

デバイス名のパスセグメントを削除：DataHub MQTT ブローカーがデータ階層を Sparkplug B Edge of Network デバイスとして公開する場合、データポイント名の最初のブランチ名を Sparkplug B の「デバイス」名として使用します。このブランチの子ブランチとリーフは、デバイスメトリクスとして公開されます。場合によっては、デバイス名として最適に使用される分岐の上に 1 つ以上の追加分岐レベルを含む階層がある場合があります。このオプションを使用すると、EoN デバイス名を表す階層内のブランチに到達するためにスキップするデータポイント階層内のブランチを変更せずに使用するには、この値をゼロに設定します。

●Primary Application

この設定により、DataHub インスタンスをプライマリアプリケーションとして機能させることができます。プライマリアプリケーションは、現在の接続ステータスについて他の Sparkplug クライアントに通知します。MQTT ブローカに接続するプライマリアプリケーションは 1 つだけにする必要があります。

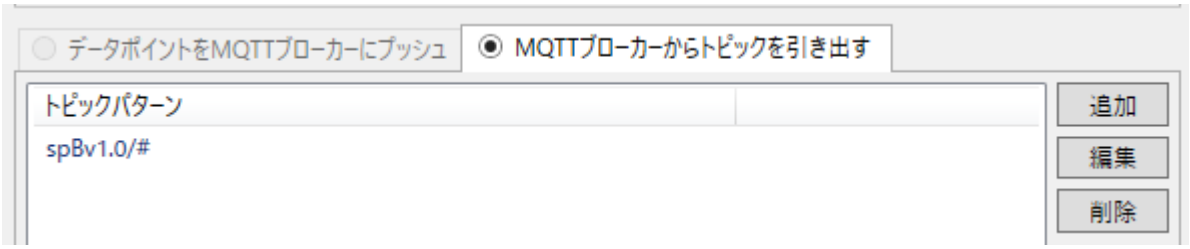


プライマリアプリケーション ID：このクライアント接続の識別子。MQTT ブローカへのすべての接続の中で固有でなければなりません。

トピック Prefix 出力：DataHub ドメイン名（が存在する場合）の後ろ、グループ名またはノード名の前に、各データポイントの名前に追加される識別子文字列。通常の Sparkplug B 識別子の文字制限に加えて、この識別子にはコロン（:）文字を含めることはできません。

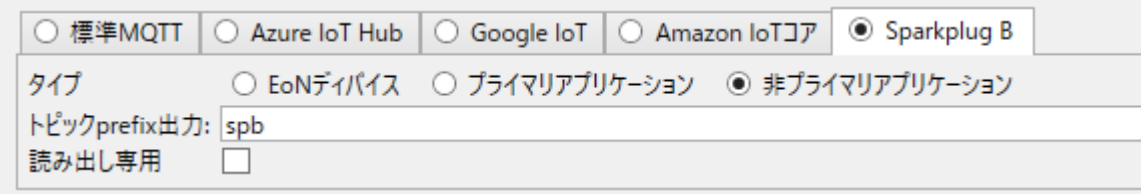
読み出し専用：このボックスにチェックすると、このアプリケーションが EoN デバイスとクライアントに値を書き戻さなくなります。

MQTT ブローカからトピックをプルするためのトピックパターンは、常に設定する必要があります。



●Non Primary Application

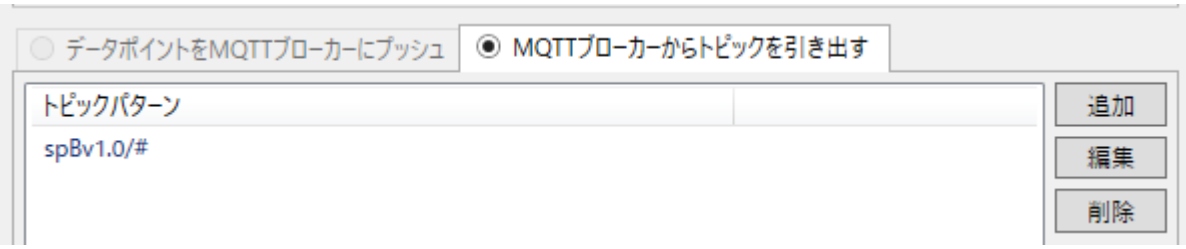
この設定により、DataHub インスタンスが非プライマリアプリケーションとして機能するようになります。非プライマリアプリケーションは、現在の接続状態について他の Sparkplug クライアントに通知しません。非プライマリアプリケーションをいくつでもブローカに接続することが可能です。



トピック Prefix 出力：DataHub ドメイン名（が存在する場合）の後ろ、グループ名またはノード名の前に、各データポイントの名前に追加される識別子文字列。通常の Sparkplug B 識別子の文字制限に加えて、この識別子にはコロン（:）文字を含めることはできません。

読み出し専用：このボックスにチェックすると、このアプリケーションが EoN デバイスとクライアントに値を書き戻さなくなります。

MQTT ブローカからトピックをプルするためのトピックパターンは、常に設定する必要があります。



接続設定が終了した後、MQTT データ交換 オプションを設定します。

2．6．1．2．Sparkplug B サーバ

DataHub の Sparkplug B smart broker は、Sparkplug B メッセージを解釈し、プライマリまたは非プライマリアプリケーションとして動作します。

Sparkplug B スマートブローカーとして動作：ここにチェックを入れると、MQTT ブローカが Sparkplug B ブローカとして機能します。

Sparkplug B の追加処理には、次のものが含まれます。

- プライマリーまたはノンプライマリーアプリケーションとして動作します。EoN デバイスからすべての情報を収集し、DataHub ポイントとして公開します。スマートブローカーが読み

取り専用に設定されていない場合、これらのデータポイントへの変更は EoN デバイスに送信されます。

- すべての EoN デバイスの接続状態を監視し、EoN デバイスが切断されると、関連するデータポイントの品質を「未接続」に変更します。
- プライマリーまたは非プライマリーのアプリケーションが接続するたびに、現在接続されているすべての EoN デバイスに対して Sparkplug B BIRTH メッセージを合成します。これにより、接続順序に関係なく、アプリケーションは常に EoN デバイスの情報を得ることができるようになります。
- EoN デバイスからのメッセージのシーケンス番号を監視します。シーケンス外のメッセージが到着した場合、スマートブローカーは EoN デバイスの接続を強制的に切断します。これにより、EoN デバイスは再接続し、データを再同期します。
- EoN デバイスへの書き込みを監視します。デバイスへの書き込みが発生し、5 秒以内にそのメトリックについてデバイスから後続のデータが到着しない場合、スマートブローカーは EoN デバイスを強制的に切断します。これにより、EoN デバイスは再接続し、そのデータを再同期することになります。

タイプ：DataHub Smart Broker は、Sparkplug B アプリケーションとしても同時に機能します。システムにプライマリアプリケーションがすでにある場合は、ここで 非プライマリアプリケーション を選択します。それ以外の場合は、プライマリアプリケーション を選択します。

Sparkplug アプリケーション ID：クライアント接続の識別子。MQTT Broker へのすべての接続に対して固有である必要があります。

（注）Sparkplug B の識別子文字列には、プラス記号(+)、フォワードスラッシュ(/)、ナンバーサイン(#)を除く、有効な UTF-8 英数字を含めることができます。

データポイント prefixc：各データポイント名に追加される識別子文字列です。

DataHub ドメイン名がある場合はその後ろ、グループ名またはノード名の前に追加されます。データドメイン名(Place all points in this data domain で指定)が空白の場合、またはここで指定した データポイント prefixc がコロン(:)文字で終わる場合、この値は Sparkplug B データから派生したデータポイントのデータドメイン名として使われます。Sparkplug B の STATE ポイントは、このオプションの影響を受けず、プレーンテキストの MQTT メッセージとして処理されます。

故障メッセージを検出：これにより、DataHub インスタンスは、EoN（Edge of Network）デバイスからのメッセージの順番が狂っていたり、失われたりしていることを確認できます。この

場合、DataHub インスタンスはデバイスを切断し、再接続することで BIRTH（起動）メッセージを再送信し、すべての受信アプリケーションを再同期させます。

EoN 書き込みの障害を検出：EoN（Edge of Network）デバイスへの書き込みが失敗した場合、デバイスは必ずしも正しい電流値を再送信しません。そのため、接続されたアプリケーションは書き込みが失敗したことを知らず、誤った値が反映されてしまいます。このオプションを選択すると、タイマーを使用して、書き込み要求が送信されたが、その後の DATA メッセージが EoN デバイスから受信されなかったことを確認します。この場合、DataHub インスタンスは EoN デバイスを強制的に切断し、BIRTH メッセージを再送信させ、アプリケーションを再同期させます。

EoN BIRTH メッセージを放出：このオプションを使用すると、新しいアプリケーションが接続したときに、DataHub インスタンスが EoN（Edge of Network）デバイスの代わりに BIRTH メッセージを合成することができます。これにより、新規に接続したアプリケーションは正しく同期し、その後、EoN デバイスから DATA メッセージを受信して処理します。このオプションは、EoN デバイスと同期できない非プライマリー・アプリケーションでも機能します。

読込み専用：このボックスをチェックすると、Smart Broker から EoN デバイスやクライアントへの値の書き込みを禁止します。

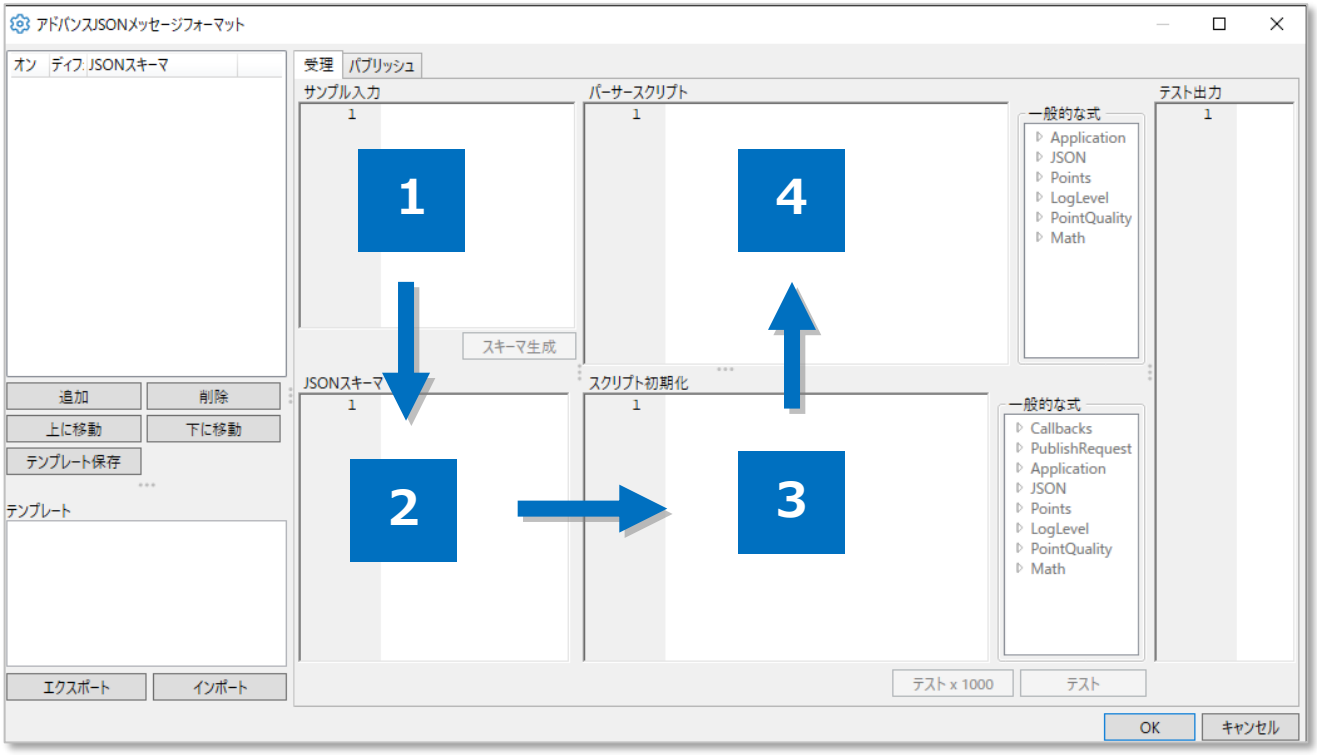
2．6．2．MQTT Advanced Parser Tutorial

こちら（メーカ英文マニュアル）

DataHub の MQTT クライアントと MQTT Broker の両機能に、MQTT メッセージの送受信に対して、任意の数の異なる JSON フォーマットを受け入れて使用することができる Advanced オプションが追加されました。これにより、MQTT クライアント間だけでなく、他のすべての DataHub プロトコルで MQTT データを統合することができます。

このセクションでは、MQTT Advanced Parser インターフェイスの紹介と、その使用方法を説明する簡単な Work-through（ウォークスルー）について説明します。詳細については、MQTT Advanced Parser Reference をご参照ください。

アドバンス JSON メッセージ フォーマット 画面は、4 つのメインパネルで構成されており、通常、順番に使用します：



1．「サンプル入力」パネルは、MQTT メッセージのサンプルを入力するところです。これは、デバイスが生成できる最大のメッセージコンテンツで、完全なサンプルである必要があります。デバイスメーカーがサンプル・メッセージ形式を提供していない場合は、DataHub MQTT Broker ですべてのメッセージをプレーンテキストとして扱うように設定し、デバイスを接続することで判断できます。メッセージは DataHub のデータブラウザのデータポイントに配置されるので、ここからコピーすることができます。

2．「JSON スキーマ」パネルは、デバイスが生成するメッセージを記述した JSON スキーマです。デバイスの製造元がスキーマを提供している場合は、ここに入力することができます。そうでない場合は、サンプルメッセージ（上記）を入力し、「スキーマ生成」ボタンを押すと、そのメッセージからスキーマが作成されます。必要に応じて、スキーマを編集して、不足するフィールドを削除したりすることができます。スキーマには、すべてのメッセージに含まれることが保証されているすべてのフィールドをリストアップした “required” メンバーを含める必要があります。“required” メンバーが設定されていない場合、有効な JSON すべてのスキーマと一致することになり、異なるデバイスのスキーマを区別することができなくなります。

3．「スクリプト初期化」パネルは、スキーマを使用する最初のメッセージがトピック毎に受信された時に 1 回だけ実行されます。これは、トピック毎に 1 回実行され、解析プロセスをガイドする関数

を作成するために使用されます。デバイスが高速でデータを生成している場合は、デバイスからのメッセージを解析する時の CPU 負荷を最小限に抑えるために、できるだけ多くのコードをここに配置してください。このスクリプトは空にすることもできます。

4．「パーサースクリプト」パネルは、スキーマに一致する MQTT トピックから受信したすべてのメッセージに対して実行されます。メッセージからデータを抽出し、DataHub データポイント値に書き込みます、最小限のパーサースクリプトは、次の 1 行で構成されます。

```
app.ProcessJson();
```

2．6．2．1．サンプル例①

このデバイス出力例では、次のような内容で MQTT メッセージを生成している可能性があります。

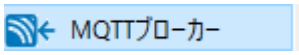
```
{"devicename": "Pump3","manufacturer": "Acme Co.","speed": 47.33,
"gpm": 37.4,"status": true,"tstamp": 1633677890,"qcomms": 64,
"activate": false,"setpoint": 47}
```

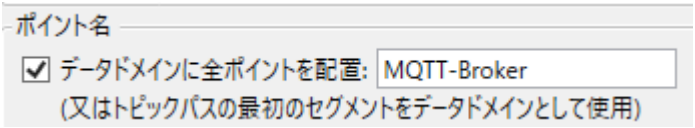
または、読み易くするには、次のようにします。

```
{
   "devicename": "Pump3",
   "manufacturer": "Acme Co.",
   "speed": 47.33,
   "gpm": 37.4,
   "status": true,
   "tstamp": 1633677890,
   "qcomms": 64,
   "activate": false,
   "setpoint": 47
}
```

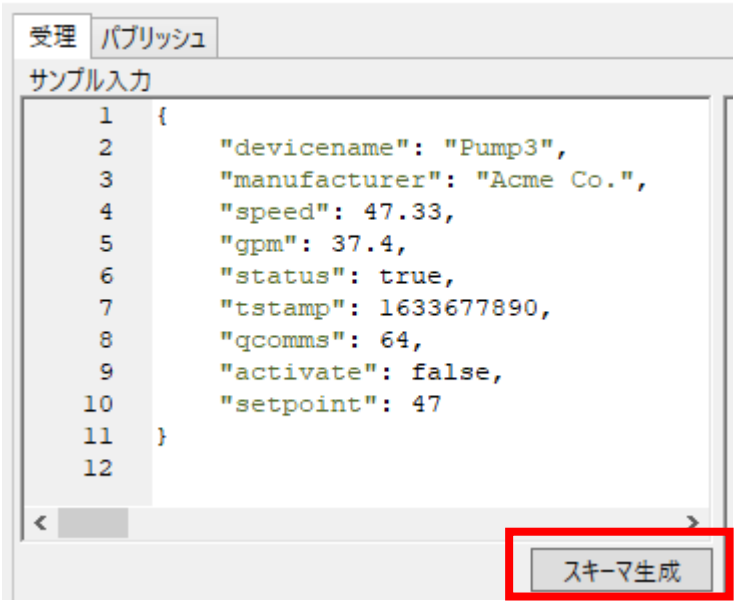
各行は、MQTT トピックとその値を表しており、DataHub のポイントとして DataHub インスタンスに入れることができます。 この例では、**manufacturer** トピックを DataHub ポイントとして表示する必要がないとします。 そして、**tstamp** トピックを他のいくつかのポイントのタイムスタンプとして機能させ、**qcomms** トピックをそれらの品質として機能させたいとします。

この例では、最初の部分でメッセージを入力してスキーマを生成します。 次に、単純なスクリプトで出力を変更し、結果をテストして、データブラウザ で確認します。

1. DataHub プロパティ画面から、MQTT Broker を選択します。 

2. このデモでは、「ポイント名」で「データドメインに全ポイントを配置」にチェックを入れ、「MQTT-Broker」と入力し、「適用」ボタンをクリックします。


3. 「メッセージフォーマット」の「アドバンス」にチェックを入れ、「編集」ボタンをクリックします。

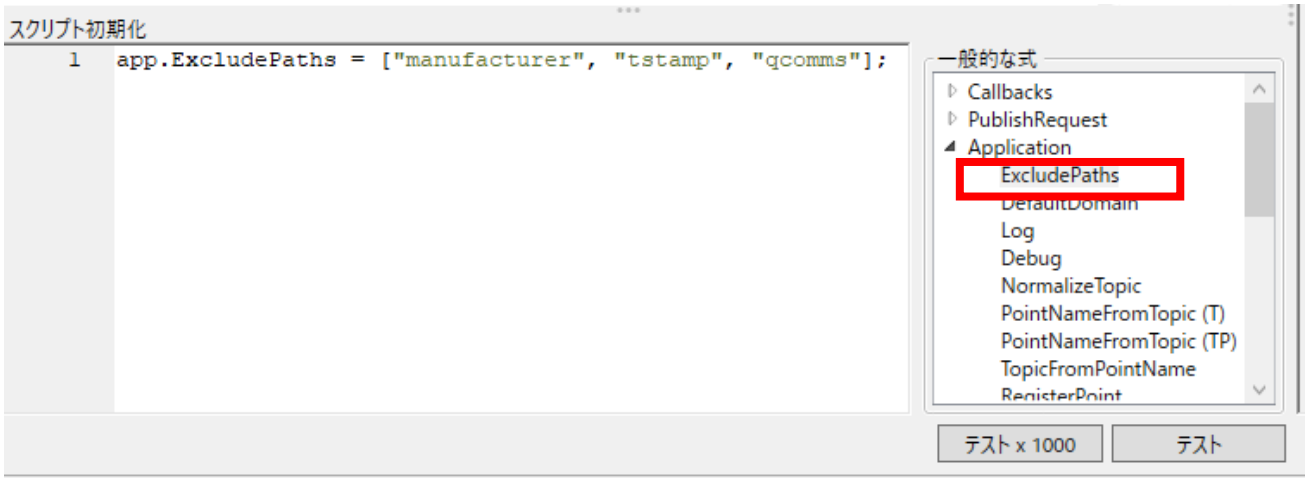

4. 左上のパネルの「追加」ボタンを選択し、新しい JSON スキーマを作成します。スキーマには、これらのメッセージを生成するデバイスの種類を識別する名前を付けます。
5. リストから新しく作成したスキーマを選択します。
6. 「サンプル入力」パネルに、上記のデバイス出力例をコピーします。


7.「スキーマ生成」ボタンをクリックします。「JSON スキーマ」パネルに JSON スキーマが生成され、表示されます。

8.「初期化スクリプト」パネルの右側にある、「一般的な式」 →「Application」グループを開き、「ExcludePaths」をダブルクリックします。これにより、app.ExcludePaths 関数が「初期化スクリプト」パネルに配置されます。

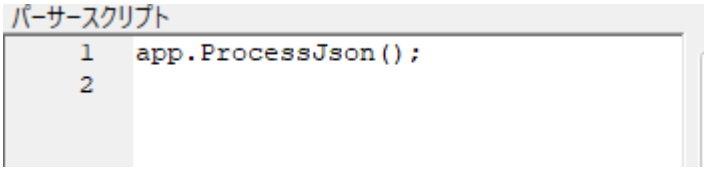
app.ExcludePaths = [];

9. 除外するトピックの名前を、文字列で追加します。



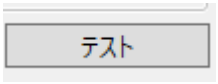
tstamp と **qcomms** を除外したのは、DataHub エンジンによってこれらをポイントとして扱わないようにするためです。次のサンプル例②でその理由を説明します。

10. 「パーサースクリプト」パネルには、app.ProcessJson()関数という 1 行があることに注目してください。

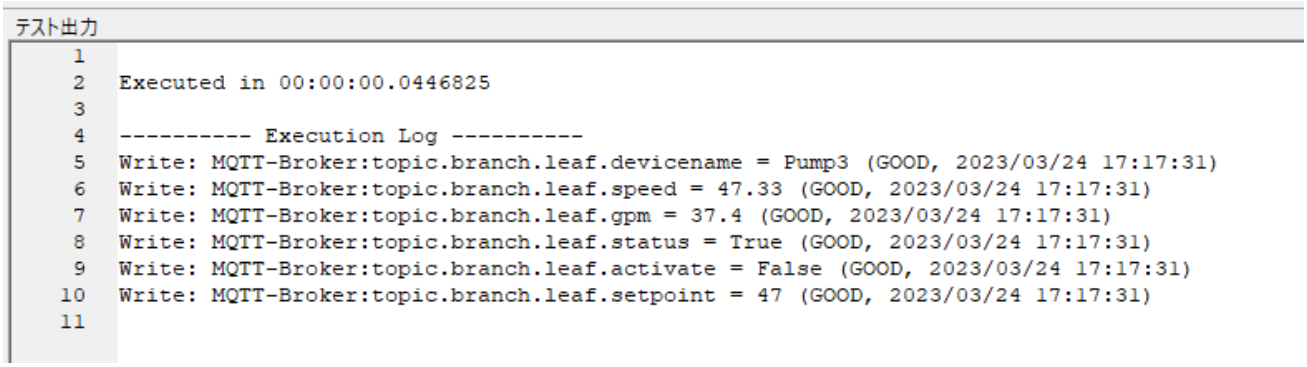


これはデフォルトの項目で、JSON 出力を処理するために必要な最小限のものです。今はこのままにしておきます。

11. 「初期化スクリプト」パネルの下部にある「テスト」ボタンをクリックします。



「テスト出力」パネルに以下のようなものが表示されます。



このテスト機能では、DataHub インスタンスに送信される内容を確認することができます。スクリプトのエラーをチェックしたり、データポイントの形式や内容が正しいことを確認したりするために使用します。このように表示されない場合は、作業を確認し、必要な変更を加えてから、再度テストを行ってください。

12. プリパティ画面の「適用」ボタンをクリックし、変更を適用して保存します。
13. 設定をテストするには、DataHub インスタンスに同じフォーマットの MQTT メッセージを送信する必要があります。任意の MQTT テストクライアントを使用することができます。DataHub インスタンスに接続し、メッセージを送信します。
14. DataHub のプロパティ画面の「データブラウザ」ボタンをクリックし、データポイントやデータを確認することができます。

このサンプル例①では、サンプル・メッセージの入力、スキーマの生成、簡単なスクリプトによる出力の変更、テスト、テンプレートの保存と適用、データブラウザにて確認する方法について説明しました。

2．6．2．2．サンプル例②

デフォルトでは、データポイントは、メッセージが配信されたトピックに由来するパスで書き込まれます。この例のデバイスでは、デバイス名が JSON ペイロードの一部です。デバイス名をポイント名の一部としてデータポイントを公開することをお勧めします。これを行うには、パーサースクリプトの一部としてデバイス名を抽出し、app.PointNameModifier コールバックを使用して、パーサーが生成するポイント名を変更します。

1. DataHub MQTT Broker プロパティのメッセージフォーマットの「アドバンス」を選択し、「編集」ボタンをクリックし、「アドバンス JSON メッセージフォーマット」画面を開きます。
2. パーサースクリプトでは、デバイス名を抽出してローカル変数に格納する必要があります。これは、デバイス名がメッセージごとに異なり、メッセージが到着するまでわからないため、初期化スクリプトではなく、パーサースクリプトに追加する必要があります。

```
var devicename = input.Json["devicename"];
```

パーサースクリプト

```
1  ver devicename = input.Json["devicename"];
2  app.ProcessJson();
3
```

3. 次のコードを初期化スクリプトに追加します。

```
app.PointNameModifier = function (topic, jsonPath)
{
  var pointName = app.PointNameFromTopic (topic, devicename +
                                  "/" + jsonPath);
  return pointName;
};
```

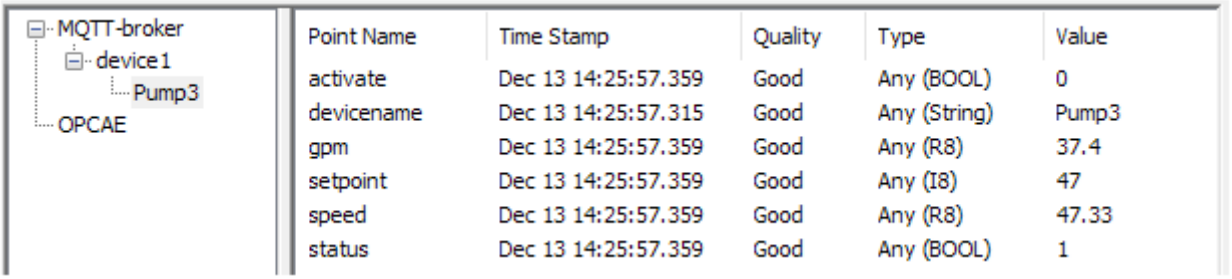
Callbacks、Application、JSON などのコードテンプレートは、「一般的な式（Common Formulas）」にあることを忘れないでください。ダブルクリックすると、作業スペースに入力されます。

4. 「テスト」ボタンをクリックします。「テスト出力」パネルには、次のような表示となります。

```
---------- Execution Log ----------
Write: broker:topic.branch.leaf.Pump3.devicename = Pump3 (GOOD, 2021-12-13 11:56:03 AM)
Write: broker:topic.branch.leaf.Pump3.speed = 47.33 (GOOD, 2021-12-13 11:56:03 AM)
Write: broker:topic.branch.leaf.Pump3.gpm = 37.4 (GOOD, 2021-12-13 11:56:03 AM)
Write: broker:topic.branch.leaf.Pump3.status = True (GOOD, 2021-12-13 11:56:03 AM)
Write: broker:topic.branch.leaf.Pump3.activate = False (GOOD, 2021-12-13 11:56:03 AM)
Write: broker:topic.branch.leaf.Pump3.setpoint = 47 (GOOD, 2021-12-13 11:56:03 AM)
```

topic.branch.leaf.テキストは、テスト出力のプレースホルダーです。このテキストは、MQTT クライアントによって指定された実際のパスに置き換えられます。

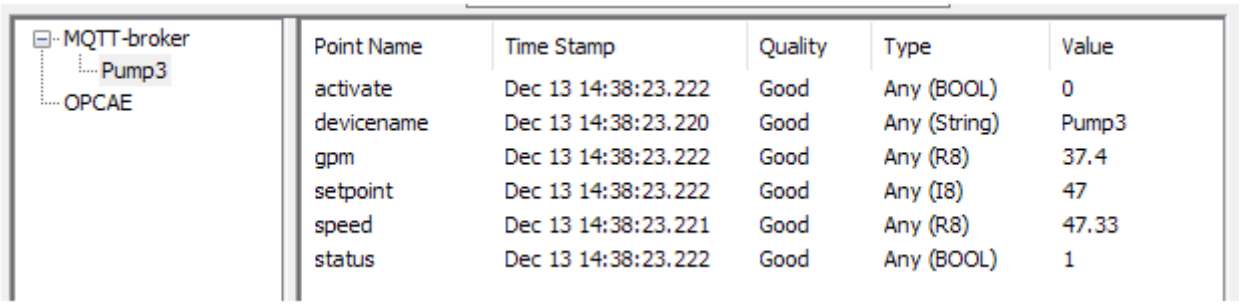
5. 「OK」をクリックし、「アドバンス JSON メッセージフォーマット」画面を閉じ、「適用」をクリックします。
6. DataHub インスタンスを再起動し、未使用のデータポイントを消去します。
7. MQTT デモクライアントを DataHub インスタンスに再接続し、サンプル・メッセージを再送信します。データブラウザの表示には、”MQTT-Broker”ドメインの “device1” ブランチ内のデータポイントが表示されるはずです。



8. これは、私たちが望んでいることとは異なるかもしれません。このデバイスを識別するには devicename フィールドで十分なので、トピック名で作成された追加のパスは必要ありません。そこで、app.PointNameModifier を変更し、トピックから最後のパス要素を削除することで、ポイント名を計算する前にトピックのレベルを 1 つ削除することができます。:

```
app.PointNameModifier = function (topic, jsonPath)
{
  topic = topic.Delete(-1,1) + "/" + devicename;
  var pointName = app.PointNameFromTopic (topic, jsonPath);
  return pointName;
};
```

9. DataHub インスタンスを再起動して未使用のデータポイントを消去し、MQTT デモクライアントを再接続してサンプル・メッセージを再度送信します。すると、次のようなデータポイントが表示されるはずです：



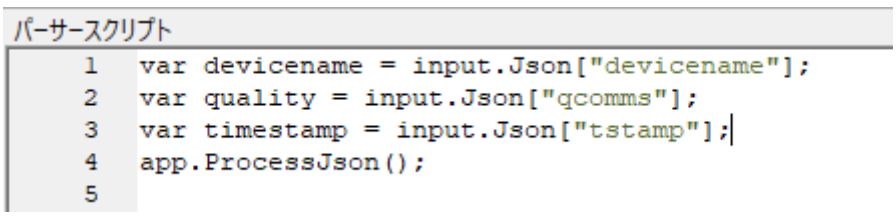
さて、システムのデフォルトの動作は、品質値 192（GOOD）と現在のシステム時刻のタイムスタンプを挿入することです。しかし、これらをデバイス自体から来る品質とタイムスタンプに変更することができます。この例では、**qcomms** と **tstamp** の値がその情報を含んでいます。ここでは、それを抽出してデータポイントに適用する方法を説明します。

1. 「アドバンス JSON メッセージ フォーマット」画面を再度開き、編集を続けます。
2. 「初期化スクリプト」に、app.ProcessJson が提供されたときの標準的な動作をオーバーライドする ValueProcessor コールバックを追加します。これを使用して、独自のタイムスタンプと品質で、データポイントの値を自分で書き込むことにします。:

```
app.ValueProcessor = function (topic, jsonPath, obj)
{
  app.WritePoint (topic, jsonPath, obj, quality,
                  JsonExtensions.GetDate(timestamp));
};
```

3. 「パーサースクリプト」に以下の行を追加し、デバイスからの値を代入します。

```
var quality = input.Json["qcomms"];
var timestamp = input.Json["tstamp"];
```



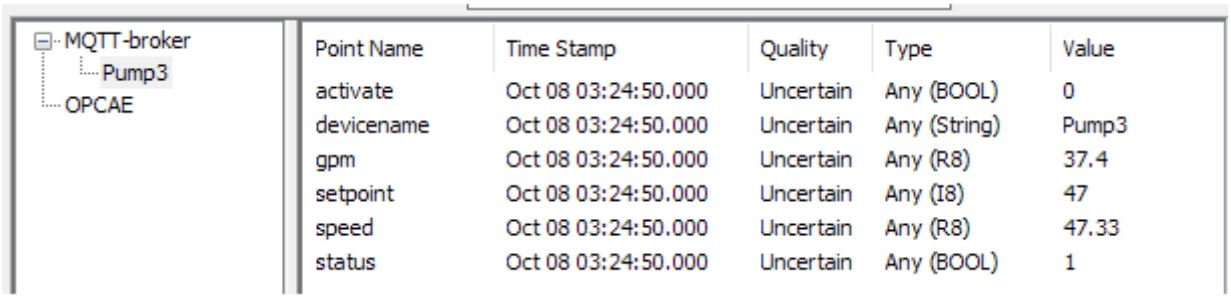
サンプル入力（input.Json）から得られるタイムスタンプと品質の値を、app.WritePoint 関数で指定した **timestamp** と **quality** の変数に代入しています。これらの値はメッセージごとに

変わる可能性があるため、パーサースクリプトで代入する必要があります。これらのコード行はすべて、最後の app.ProcessJson(); 行の上に記述する必要があります。
JsonExtensions.GetDate 関数を使用して、デバイスから送られてくる数値のタイムスタンプを日付/時刻形式に変換しています。

4. Test ボタンをクリックします。テスト出力では、品質（64 = UNCERTAIN）とタイムスタンプ（1633677890 = 2021-10-08 7:24:50 AM）がサンプル入力から来るようになり、DataHub インスタンスによって生成されていないことがわかります。

```
---------- Execution Log ----------
Write: MQTT-Broker:topic.branch.leaf.devicename = Pump3 (UNCERTAIN, 2021-10-08 7:24:50 AM)
Write: MQTT-Broker:topic.branch.leaf.speed = 47.33 (UNCERTAIN, 2021-10-08 7:24:50 AM)
Write: MQTT-Broker:topic.branch.leaf.gpm = 37.4 (UNCERTAIN, 2021-10-08 7:24:50 AM)
Write: MQTT-Broker:topic.branch.leaf.status = True (UNCERTAIN, 2021-10-08 7:24:50 AM)
Write: MQTT-Broker:topic.branch.leaf.activate = False (UNCERTAIN, 2021-10-08 7:24:50 AM)
Write: MQTT-Broker:topic.branch.leaf.setpoint = 47 (UNCERTAIN, 2021-10-08 7:24:50 AM)
```

5. ［OK］をクリックして［Advanced JSON Message Format］ウィンドウを閉じ、［Apply］をクリックします。
6. MQTT デモクライアントを DataHub インスタンスに再接続し、サンプル・メッセージを再送信します。これで、タイムスタンプと品質に関する MQTT テストクライアントの値が DataHub インスタンスに送信されたことが確認できるはずです。



［注意]を参照してください。
タイムスタンプが複数時間異なるのは、Test Output パーサーが GMT0 時間で日付を表示し、DataHub Data Browser がローカル時間で日付を表示するためです。

2．6．2．3．サンプル例③

これまで、デバイスまたは他の MQTT クライアントからデータを受信する方法をご説明しました。次に、データをクライアントに公開する方法についてご説明します。このために、データ ポイント **setpoint** と **activate** を使用します。

1. DataHub MQTT Broker の「アドバンス JSON メッセージ フォーマット」画面を再度開き、編集を続けます。
2. 上部の「パブリッシュ」タブをクリックし、「パブリッシュ」インターフェイスを開きます。
3. 「Add」ボタンをクリックし、「TestPublish1」という名前を入力し、「OK」ボタンをクリックします。
4. 「ASP 仕様」に以下を入力します：

```
{
   "setpoint": <%= point.DblVal %>
}
```

これは“セットポイント”の簡単なパブリッシュフォーマットです。DataHub のデータエンジンが Double でデバイスに書き込むことができます。

5. もう一度、「追加」ボタンをクリックし、「TestPublish2」という名前を入力し、OK をクリックします。
   「ASP 仕様」に以下を入力します：

```
{
   "activate": <%= point.IntVal == 0 ? "false" : "true" %>
}
```

これは、DataHub のデータエンジンに、0 または 1 のブーリアン入力値を、デバイスのデータ形式に合わせて、false または true という文字列に変換するよう指示します。
各仕様を入力すると、クライアントに送信される MQTT 形式のメッセージの例が生成されます。下部のチェックボックスは、これが有効な JSON であるかどうかを示しています。

6. ここで、これらの仕様を呼び出す必要があります。「受信」タブに戻り、「初期化スクリプト」パネルで次の行を追加します。

```
app.TopicInitialization = function (topic, originalTopic, message, json)
{
        app.RegisterPoint(topic, "setpoint", "TestPublish1");
        app.RegisterPoint(topic, ”Activate”, “TestPublish2”);
};
```

app.TopicInitialization では、データポイントとパブリッシュフォーマットを関連付けることができます。app.RegisterPoint 関数は、これら 2 つのポイントを DataHub データエンジンに登録し、それぞれの JSON 出力フォーマットを指定します。

7. 「テスト」ボタンをクリックし、出力を確認します。
8. 「OK」をクリックして「アドバンス JSON メッセージ」画面を閉じ、「適用」をクリックします。
9. DataHub インスタンスを再起動し、未使用のデータポイントを消去します。
10. MQTT デモクライアントを DataHub インスタンスに再接続し、サンプル・メッセージを再送します。
11. ハッシュ(#)文字だけを使用し、すべてのポイントをパブリッシュします。**setpoint** と **activate** ポイントからメッセージを受信するはずです。
12. テストするには、データブラウザを MQTT-Broker ドメインの **device1** ブランチを開きます。
13. **setpoint** をクリックし、新しい値を入力します。「入力」ボタンをクリックすると、その値が MQTT クライアントに送信されます。

これらの例は、MQTT Advanced Parser の機能と柔軟性を簡単に紹介するものです。より詳細な情報については、「MQTT Advanced Parser Reference」セクションを参照してください。

2．6．3．MQTT Advanced Parser リファレンス

詳細は、このリンクの DataHub マニュアル（英文）をご参照ください。

付録

1．資料情報

Cogent DataHub® に関する資料のご紹介

- **Cogent Real-Time Systems社 ホームページ**
- **Cogent DataHub マニュアル（英文）**
- **Cogent DataHub 最新ソフトウェア ダウンロード ページ**

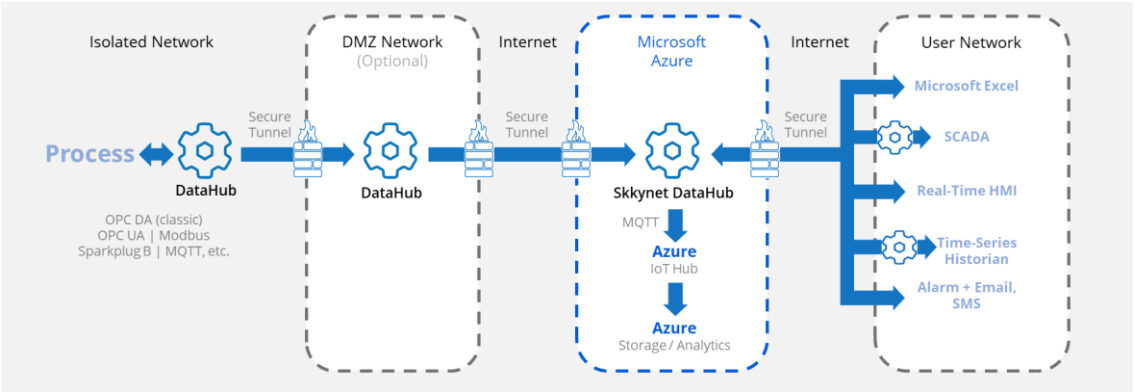
2．ソリューション・サービスのご紹介

- **Skkyent DataHub Service**

  紹介動画（日本語字幕付き）リンク

  Skkynet DataHub Service は、Microsoft Azure のマネージドアプリケーションとして提供される、Cogent DataHubソフトウェアをベースとした産業用IoTサービスです。マシンやアプリケーションや組み込みシステム間において、安全なリアルタイムデータ接続を確立します。

  Skkynet社サイト



3．ケーススタディ

多様な導入事例

Cogent DataHub® V10 ユーザーズ

作成　　株式会社ベルチャイルド

＊ 本書に記載されたURL等は、予告なく変更されることがあります。
＊ 本書のいかなる部分も 株式会社ベルチャイルド の承諾を得ることなく、いかなる方法によっても無断で複写、複製することを禁止します。
＊ 株式会社ベルチャイルドは、いかなる誤りや記載漏れについての責任を負いません、またこの文章に含まれる情報の使用から生じる損害に対する責任を負いません。
＊ DataHub®, WebView™, Cascade™, QuickTrend™, WebView™, UA Tunneller™, DA Tunneller™, DDE Tunneller™, Modbus Tunneller™, UA Logger™, DA Logger™, Modbus Logger™, UA Bridge™, DA Bridge™, Modbus OPC Server™, OPC Gateway™, OPC DataHub™, System Monitor™, Gamma™ , TextLogger™, DataSim™, DataPid™は、Real Innovations International LLCの商標または登録商標であり、Cogent 社のライセンスに基づいて使用されています。
＊ Skkyent® , SkkynetHub™, VINE™ のロゴまたはアイコンはSkkynet Cloud Systems, Inc. の商標であり、Cogent社のライセンスに基づいて使用されています。
＊ Amazon Kinesis は Amazon Web Services の登録商標または商標です。＊ AVEVA Historian、AVEVA Insight は AVEVA の登録商標または商標です。＊ Azure は Microsoft Corporation の登録商標または商標です。＊ Google は Google LLC の商標登録または商標です。＊ InfluxDB Cloud は InfluxDBData Inc. の登録商標または商標です。PI System は OSIsoft LLC の登録商標または商標です。WhatsApp は WhatApp Inc. の登録商標または商標です。OPC は OPC Foundation 商標です。その他の会社名、商品名、製品名は、各社の商標もしくは登録商標です。
＊ なお、本文中では、™、®マークは明記しておりません。
＊ 以下のCogentおよびSkkynetの製品およびサービスは、米国およびその他の国における特許および特許出願によって保護されています。

DataHub® WebView™	U.S. Patent No.: 8,661,092、9,667,689、10,498,796
Skkynet™ および DataHub®	U.S. Patent No.: 9,100,424、9,288,272、9,762,675
Vine™ Add-in	U.S. Patent Nos.: 10,558,744, 10,462,206

本書ガイドに関するお問い合わせ先

株式会社ベルチャイルド
contact@ibress.com