ABB MEASUREMENT & ANALYTICS | 2107721MNAA

Data Model Definitions

RMC-100

Contents

List of figures

List of tables

Disclaimer

This document provides examples of data model definitions which support ABB Totalflow products. The examples are available as an aid for customers designing their own custom MQTT implementations, but not as a product that is supported by ABB. ABB is not liable for the incorrect use of any part of the contents. Customers are encouraged to examine the contents before copying or using any of the definitions. ABB expects that customers carefully examine their requirements and create the solutions appropriate to their own environment. It is expected that customers conduct their own testing process and verify that the results meet their requirements and render accurate data.

Additional information

Additional free publications are available for download at www.abb.com/upstream.

Table 0-1: Related documentation

Documents	Document number
MQTT Data Interpretation Code Example Developer Guide	2107649
MQTT Configuration Guide	2106521
Data Model Examples: RMC-100 JSON files	2107732

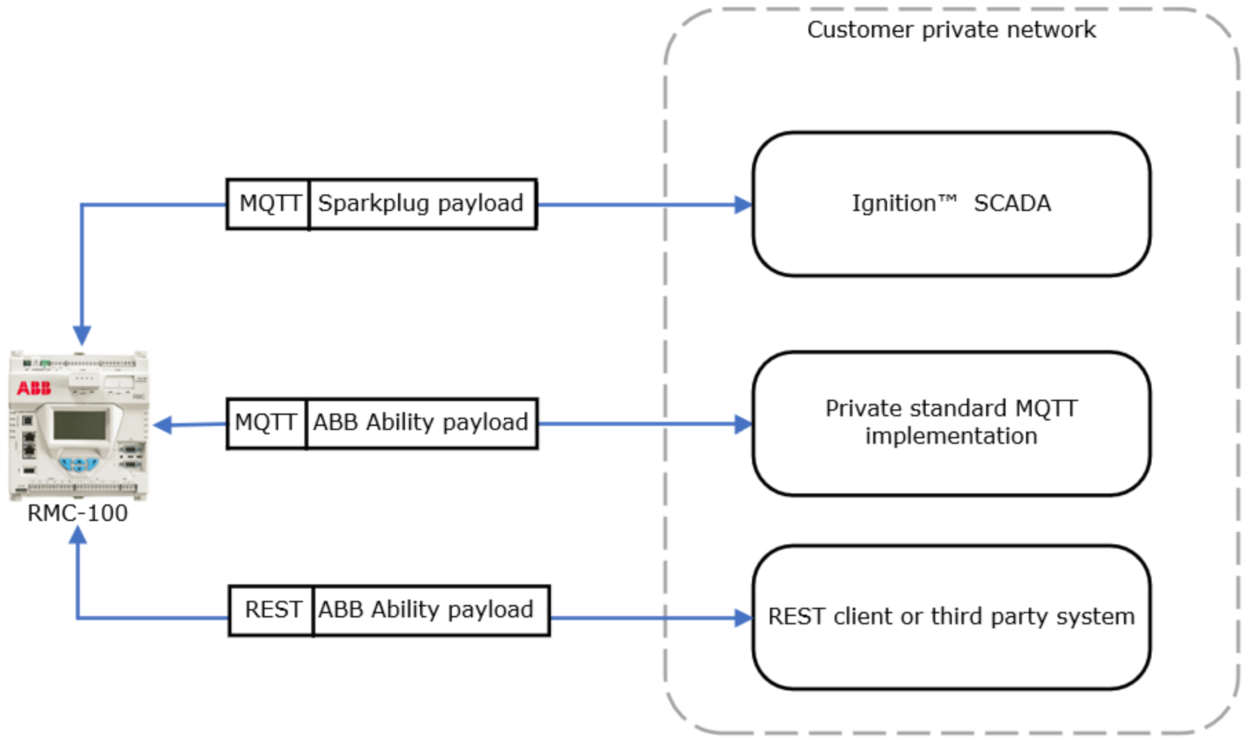
1 Overview

This document describes the payload metadata definitions supported by ABB Totalflow devices. This metadata applies to protocols like MQTT and REST. Devices with MQTT services enabled can publish their data on MQTT brokers. Devices with REST services enabled allow web-based access for configuration of MQTT parameters.

Figure 1-1 shows several scenarios of communication using the MQTT or REST protocols. The protocol packets include the Totalflow device data in their payloads. The metadata is the information required to process or interpret the data by the receiving systems:

- MQTT Sparkplug packets between the device and Ignition™-based implementations encapsulate the sparkplug payload.
- MQTT packets between a device and a standard MQTT implementation encapsulate ABB Ability payload.
- REST packets between the device and a REST client encapsulate ABB Ability payload.

Figure 1-1: Protocol payloads



IMPORTANT NOTE: This document does not include compressibility of payloads.

IMPORTANT NOTE: Totalflow devices publish protocol payload contents in JSON format. This document provides portions of code as examples of the definitions discussed. Color or bold text has been added to bring attention to the elements or concepts discussed. It is not meant to distinguish code properties. Code snippets can be copied and then pasted as "text only".

2 Supported applications

Payloads support device data for the Totalflow applications shown in Table 2-1. The data includes:

— Application register variables and configuration data. Each application, except the Holding Registers application, has pre-defined registers that store application-specific data. The Holding Registers app supports customer selection of the data stored.

— EFM data
  - Daily Record
  - Custom Record
  - Alarms (Definition and Log)
  - Events

— Trend data

Alarm data and trends are automatically published for all supported applications if defined and configured from PCCU.

Table 2-1: Totalflow applications supporting data publishing

Application	Description
Alarm system	Alarm detection, logging, and reporting application (user defines alarms)
Trend system	Data trending application (user defines the variables the device monitors)

Application	Description
Holding Registers	Holding Registers allow the user to custom define how to store values of interest in specific device register ranges. This application is customized per user requirements. The registers are not pre-defined.
AGA3	Orifice gas measurement application
AGA7	Linear gas measurement application
API Liquid SU	Linear liquid measurement
Plunger control	Control of a plunger on a production well
Gas lift	Artificial lift for wells with liquid loading problems
Shutdown System	Shut down a well or site

IMPORTANT NOTE: The Holding Registers app is supported only in private customer implementations and is included in the model definitions. Custom or third-party clients can contain logic to have access to this data. See Additional information for a link to the MQTT Data Interpretation Code Example Developer Guide. This document describes examples of code to process the Holding Registers data in addition to the other supported applications.

3 Type ID definitions

Type definitions have, at a minimum, three required elements:

- Domain: the model definition the Type definition is built on. For example, the model: abb.ability.device. See section 5 Model elements.
- Name (or unique Type ID) of the Type definition
- Version of the Type definition

Definitions may contain additional elements such as names, related models, properties, and variables.

The Type ID identifies the type of information (type definition) the content of the payload belongs to. In the case of Totalflow products and their functions, the following type of information can be indicated in the payload:

- Type of Totalflow device the information belongs to. For example, a Type ID is defined for the RMC-100, µFLO, XFC, etc. See section 3.1 Device Type ID.
- Type of Totalflow application the information belongs to. For example, a Type ID is defined for each AGA3, AGA7, API Liquid, etc. See section 3.3 Application Type ID.

The initial or birth payload carries this Type ID to inform the host or consumer system (broker) about the type the payload content belongs to.

3.1 Device Type ID

Table 3-1 shows Type IDs for Totalflow devices. The RMC-100 is the device currently supported. Each device type has its own unique Type ID. The device type is shown here in bold.

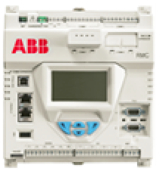
Table 3-1: Totalflow device Type IDs

Type of Information: Device Type	Example	Type ID
Currently supported	RMC-100	abb.totalflow.device.**rmc**.type
Future	µFLO^{G5}	abb.totalflow.device.**uflo**.type
Future	XFC ^{G5}	abb.totalflow.device.**xfc**.type
Future	XRC ^{G5}	abb.totalflow.device.**xrc**.type

Figure 3-1 shows examples of Totalflow devices, like the RMC-100 and µFLO^{G5}, and their respective Type IDs. Each Type ID is generic in that it applies to all instances of the same device type in any installation.

Figure 3-1: Device Type IDs

TypeID
abb.totalflow.device.rmc.type



TypeID
abb.totalflow.device.uflo.type

In installations with several devices of the same type, the Type ID applies to all devices of that type. Specific instances of the same device type must be uniquely identified (see section 4.1 Device Object ID).

3.2 Example: RMC device type definition

Figure 3-2 shows a portion of the Type definition (typeId in the code) for the RMC-100 device in .json format. There are 3 required elements at the top of the definition:

- The top domain or **model** (abb.ability.**device**)
- The Type ID (abb.totalflow.device.**rmc**.type)
- The version (**1.0.0**)
- In this example, the definition also includes the type name: **RMC device definition**.

Additional elements in the definition include related models, variables, and properties:

- The relatedModels section contains other models for the data of the device
- The variables section defines the variable names associated with the device type. As the name indicates, this is information that will change and is refreshed at every publishing cycle as necessary. Each variable has appropriate attributes that characterize the type of information contained.
- The properties section lists properties associated with the device type. These are normally fixed technical or manufacturing details about the device that could be unique, mandatory, vendor-specific, etc.

Figure 3-2 shows a single variable and property definition as examples:

- **dateTime** and its attributes such as displayName, description, etc.

IMPORTANT NOTE: The full definition has all associated device variables and properties. There are several variables defined after dateTime, and then several additional properties after the ipAddress. To simplify, only a portion of the definition is shown in this example. Color or bold style in the code examples is used to highlight discussed elements.

Figure 3-2: Sample Type ID definition for device type: Totalflow RMC-100

```
{
    "model": "abb.ability.device",
    "typeId": "abb.totalflow.device.rmc.type",
    "version": "1.0.0",
    "name": "RMC device definition",
   "isExtensible": false,
    "relatedModels": [
      {
        "type": "abb.totalflow.configuration.rmc.type@1",
        "model": "abb.ability.configuration"
      },
      {
```

```
            "type": "abb.totalflow.structure.rmc.type@1",
            "model": "abb.ability.structure"
         },
         {
            "type": "abb.totalflow.device.register.type@1",
            "model": "abb.ability.register"
         }
      ],
      "description": "Device type definition",
      "tags": [
         "Device"
      ],
      "variables": {
         "dateTime": {
            "displayName": "Date Time",
            "description": "Displays the device's date and time",
            "isMandatory": true,
            "eventEnable": false,
            "cloudAccess": "Read",
            "cloudEnable": true,
            "dataType": "int",
            "dataTypeExt": "uint32"
         },
         ...
"properties": {
         "ipAddress": {
            "displayName": "IP Address",
            "description": "IP Address of the device",
            "isMandatory": true,
            "eventEnable": false,
            "cloudAccess": "Read",
            "cloudEnable": true,
            "dataType": "string"
         },
```


3.3 Application Type ID

Table 3-2 shows the defined Type IDs for the supported Totalflow applications. The type of application is shown here in bold.

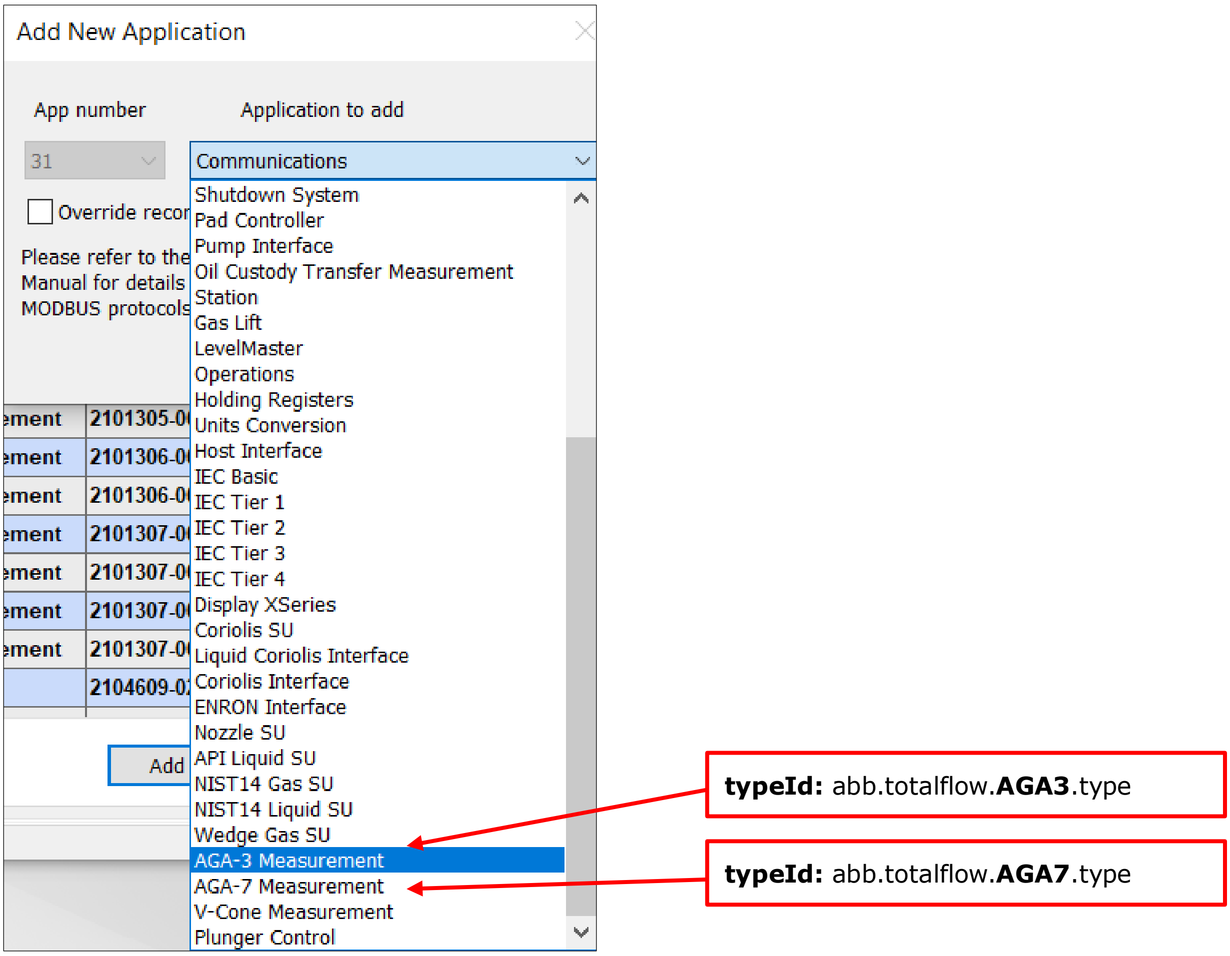
Table 3-2: Type of Totalflow application IDs

Type of Information: Totalflow app	Example	Type ID
Currently supported	AGA3	abb.totalflow.**AGA3**.type
Currently supported	AGA7	abb.totalflow.**AGA7**.type
Currently supported	API Liquid SU	abb.totalflow.**liquid**.type
Currently supported	Plunger Control	abb.totalflow.**plunger**.type

Type of Information: Totalflow app	Example	Type ID
Currently supported	Gas Lift	abb.totalflow.**gaslift**.type
Currently supported	Shutdown System	abb.totalflow.**shutdown**.type
Currently supported	Holding Registers	abb.totalflow.**holdingregisters**.type

Figure 3-3 shows a portion of the Totalflow applications configurable from PCCU. The applications in the data model are a subset in this list (see section 2 Supported applications). The Type IDs for the AGA3 and AGA7 Measurement applications are displayed.

Figure 3-3: Example of Totalflow applications and their Type IDs



Devices can have multiple instances of the same type of application. The Type ID applies to all the applications of the same type. Specific instances of the same application type must be uniquely identified. See section 4.2 Application Object ID.

3.4 Example: AGA3 application type definition

Figure 3-4 shows a portion of the Type definition for the Totalflow AGA3 Measurement application in .json format. The three required elements are listed at the top of the definition:

- The top domain or model (abb.totalflow.**app**) specific for the Totalflow products
- The Type ID (abb.totalflow.**AGA3**.type)
- The version (**1.0.0**).
- In this example, the definition also includes the type name: **AGA3**.

Additional elements in the definition include related models, variables, and properties:

- The relatedModels section contains other models for the data of the application type (AGA3 has several data categories.) See Table 3-3.
- The properties section lists properties associated with the application type. These are normally fixed information about the application that could be unique, mandatory, etc.

Figure 3-4 shows only one property definition as an example: the **deviceID** and its attributes such as displayName, description, etc.

IMPORTANT NOTE: The full definition has all the variables and properties for the application. Each application instance in the device has its own set of variables and properties. To simplify, only a portion of the definition is shown in this example. Color or bold style in the code examples is used to highlight discussed elements.

Figure 3-4: Sample Type ID definition for application type: Totalflow AGA3

```
{
    "model": "abb.totalflow.app",
    "typeId": "abb.totalflow.AGA3.type",
    "version": "1.0.0",
    "name": "AGA3",
    "relatedModels": [
      {
        "type": "abb.totalflow.AGA3.composition.type@1",
        "model": "abb.totalflow.composition"
      },
      {
        "type": "abb.totalflow.AGA3.lastcalc.type@1",
        "model": "abb.totalflow.lastcalc"
      },
      {
        "type": "abb.totalflow.AGA3.aggregate.type@1",
        "model": "abb.totalflow.aggregate"
      },
      {
        "type": "abb.totalflow.AGA3.auxiliary.type@1",
        "model": "abb.totalflow.auxiliary"
      },
      {
        "type": "abb.totalflow.AGA3.qtr.customlog.type@1",
        "model": "abb.totalflow.qtr.customlog"
      },
      {
        "type": "abb.totalflow.AGA3.qtr.dailylog.type@1",
        "model": "abb.totalflow.qtr.dailylog"
      },
      {
        "type": "abb.totalflow.AGA3.alarmlog.type@1",
        "model": "abb.totalflow.alarmlog"
      },
      {
        "type": "abb.totalflow.AGA3.alarmdefinition.type@1",
        "model": "abb.totalflow.alarmdefinition"
      },
      {
```

```
            "type": "abb.totalflow.AGA3.trendlog.type@1",
            "model": "abb.totalflow.trendlog"
        },
        {
            "type": "abb.totalflow.AGA3.trenddefinition.type@1",
            "model": "abb.totalflow.trenddefinition"
        },
        {
            "type": "abb.totalflow.AGA3.eventlog.type@1",
            "model": "abb.totalflow.eventlog"
        },
        {
            "type": "abb.totalflow.AGA3.register.type@1",
            "model": "abb.totalflow.register"
        }
...
  ],
      "properties": {
        "deviceID": {
            "displayName": "Device/APP ID",
            "isMandatory": true,
            "dataType": "string",
            "eventEnable": false,
            "cloudAccess": "Read",
            "cloudEnable": true,
            "description": "Identification name for application device",
            "tags": [
                "application name",
                "application ID",
                "app name"
             ]
        }
}
```

In the sample code above, the application type definition lists additional or related data models needed to process the data issued by the device. Table 3-3 lists the related models for data generated by the AGA3 application. These constitute the high-level categories the AGA3 data is organized in. Some of these categories correspond to AGA3 application tabs or configuration options as presented in PCCU. The related models apply to all instances in a device.

Table 3-3: AGA3 related models referenced in the AGA3 Type ID definition

TypeName	Encapsulated information
abb.totalflow.AGA3.**composition**.type	All properties and variables for gas composition elements
abb.totalflow.AGA3.**lastcalc**.type	Data structure for last calculated values
abb.totalflow.AGA3.**aggregate**.type	Data structure for variables that are calculated combining several components
abb.totalflow.AGA3.**auxiliary**.type	Data structure for digital input or output values
abb.totalflow.AGA3.**qtr.customlog**.type	Data structure for custom Quality Transaction Record (QTR)

TypeName	Encapsulated information
	Variables defined to be monitored at defined interval
abb.totalflow.AGA3.**alarmdefinition**.type	Data structure for an alarm definition
abb.totalflow.AGA3.**qtr.dailylog**.type	Data structure for daily Quality Transaction Record (QTR)

4 Object ID definitions

The Object ID uniquely identifies a specific instance of a type or category. For example:

- A specific instance of an RMC among several RMCs installed and managed by a customer. See section 4.1 Device Object ID.
- A specific instance of an application among several applications of the same type instantiated in a device. Every payload holds this unique identifier to specify which instance of a device or app the content of the payload belongs to. See section 4.2 Application Object ID.

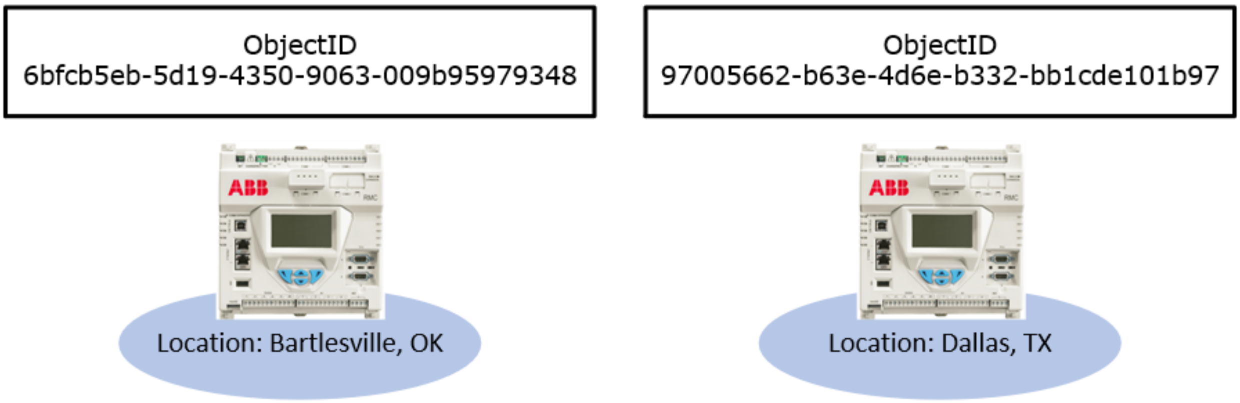
Object IDs are automatically generated by the device based on Universally Unique Identifier (UUID) standards. An Object ID or UUID is a 128-bit number. It is generated using the **libuuid** library and its representation is in the following format: **95ae6669-546c-4946-a824-c1742281e7d8**.
The probablity that a UUID can be duplicated is close enough to zero to be negligible. An Object ID is therefore considered globally unique.

4.1 Device Object ID

Each installed and active Totalflow device has a unique Object ID. For example, each RMC installed in a customer implementation will have its own unique Object ID. This guarantees that the RMC's specific data is correctly stored or processed by the end system. If the device is swapped in the field, its serial number will change, but it can still be mapped to the same Object ID.

Figure 4-1 shows how a unique Object ID identifies an RMC at a specific location. This unique ID makes it possible to differentiate the data between the two RMCs. If the same location has multiple RMCs, each RMC will have to have a unique Object ID as well.

Figure 4-1: Object IDs for two RMC-100s at different locations



Example: RMC instance Object ID definition (Figure 4-2) shows a portion of the Object ID (objectId in the code) defintion for a specific RMC in .json format. The 3 required elements in the definition are shown on top:

- The unique objectId (**6bfcb5eb-5d19-4350-9063-009b95979348**)
- The domain or model (abb.ability.**device**)
- In this example, the definition also includes a generic name: **Sample**.

Additional elements in the definition include variables and properties and their actual values:

- The variables section includes the variables associated with the device type. This is information that will change and is refreshed at every publishing cycle as necessary. The value of the variable is also displayed.
- The properties section lists properties associated with the device type. These are normally fixed technical or manufacturing details about the device that could be unique, mandatory, vendor-specific, etc.

Figure 4-2 shows some of the device variables and properties as examples:

- **dateTime** and its value, with **availableMemory** and its value
- **ipAddress** and its value, with **stationId** and its value

IMPORTANT NOTE: The full definition has the value of all device variables and properties. To simplify, not all of these are shown in the example.

Figure 4-2: Sample RMC instance Object ID definition and data

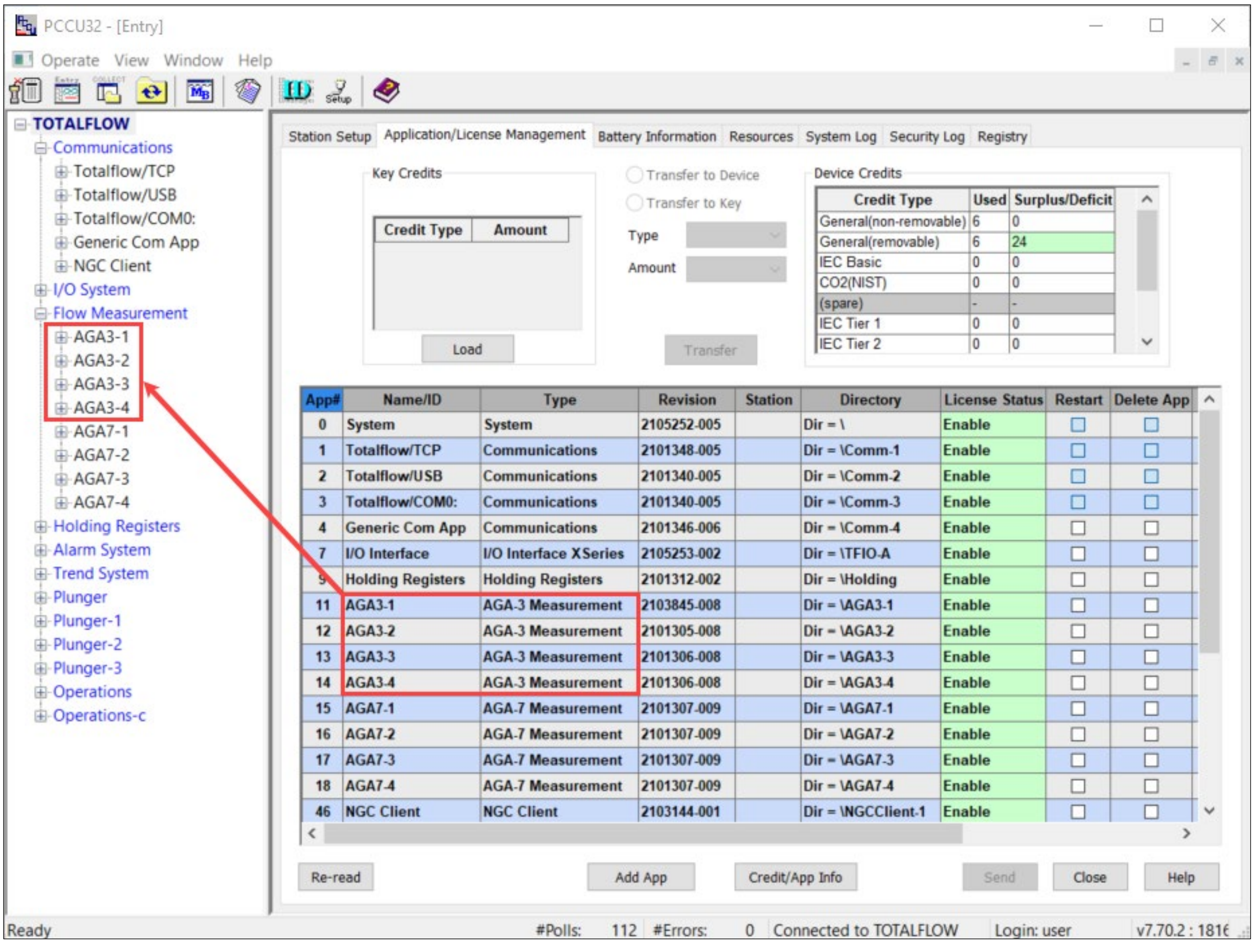

```
{
"objectId": "6bfcb5eb-5d19-4350-9063-009b95979348",
 "model": "abb.ability.device",
 "type": "abb.totalflow.device.rmc.type@1",
  "name": "Sample",
  "variables": {
         "dateTime": {
            "value": "1610116276",
            },
            "availableMemory": {
              "value": "1610116276",
              "units": "Bytes"
             }
           },
   "properties": {
            "ipAddress": {
               "value": "192.168.1.43"
          }
            "stationID":  {
              "value": "TOTALFLOW"
           }
         },
          "deviceStatus": "Connected"
      }
```


4.2 Application Object ID

Each application instance in a device has a unique Object ID. For example, if there are several tubes or AGA3 measurement instances enabled and active on a device, each has its own Object ID to guarantee that the data from each instance is differentiated from the other instances and properly published, stored, and processed by the end system.

Figure 4-3 shows a device with multiple AGA3 measurement app instances. Each of these instances is assigned a unique Object ID.

Figure 4-3: Device with multiple AGA3 instances as shown in PCCU



[Figure 4-4](#) shows a portion of the Object ID definition for an instance of the AGA3 application in .json format. The key elements include:

- The objectId (**6a819bc3-75c5-4b4c-9811-e8bd9bbcf959)**
- The domain or model (abb.**totalflow**.**app**) specific for the Totalflow applications
- A generic name for the category of the Object ID (**Application**). Additional information such as the time stamps for the data are also included.

Additional elements in the definition include variables and properties and the actual values:

- The variables section includes the variables associated with the AGA3 application. This is information that will change and is refreshed at every publishing cycle as necessary. The value of the variable is also displayed.
- The properties section lists properties associated with the AGA3 application. These are normally fixed information about the application that could be unique, mandatory, application-specific, etc.

Figure 4-4 shows some variables and properties and their actual values as examples:

- The **staticPressure** and its value
- The **deviceID** and its value: the AGA3 instance name as saved in the device. In this case, it is the default name for the first AGA3 instance: AGA3-1 as configured in PCCU. If other names have been used to identify the instance, the user-defined name should display.

Figure 4-4: Example AGA3 instance Object ID definition and data

{
"**objectId**": "6a819bc3-75c5-4b4c-9811-e8bd9bbcf959",
"**model**": "abb.totalflow.app",
"**type**": "abb.totalflow.AGA3.type@1",
"**name**": "Application",

"deviceTimestamp": "1610116276",
"deviceUtcTimestamp": "1610096476",
"**variables**": {
"**StaticPressure**": {
"value": "14.72999954223628",
"units": PSIA,
},
"**differentialPressure**": {
"value": 0,
"units": "inH2O",
},
"**volumeFlow**": {
"value": "0",
"units": {
"value ": 1
}
}
...
},
"**properties**": {
"**deviceID**": {
"value": " AGA3-1",
},
"**tubeDescription**": {
"value": "Totalflow",
},
"**enhancedMode**": {
"value": "0",
}
}
}

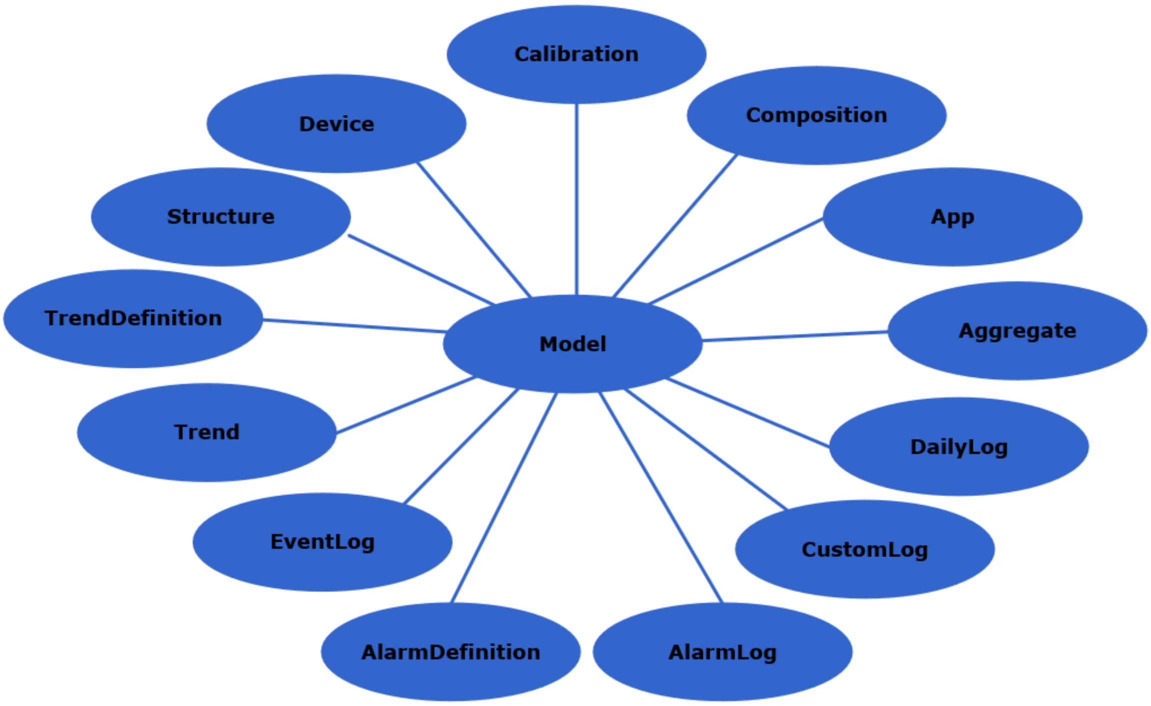
5 Model elements

Every Type ID and Object ID definition includes the model element. The model indicates that the content in the payload belongs to an abstract feature or category independent of the type and instance. For example:

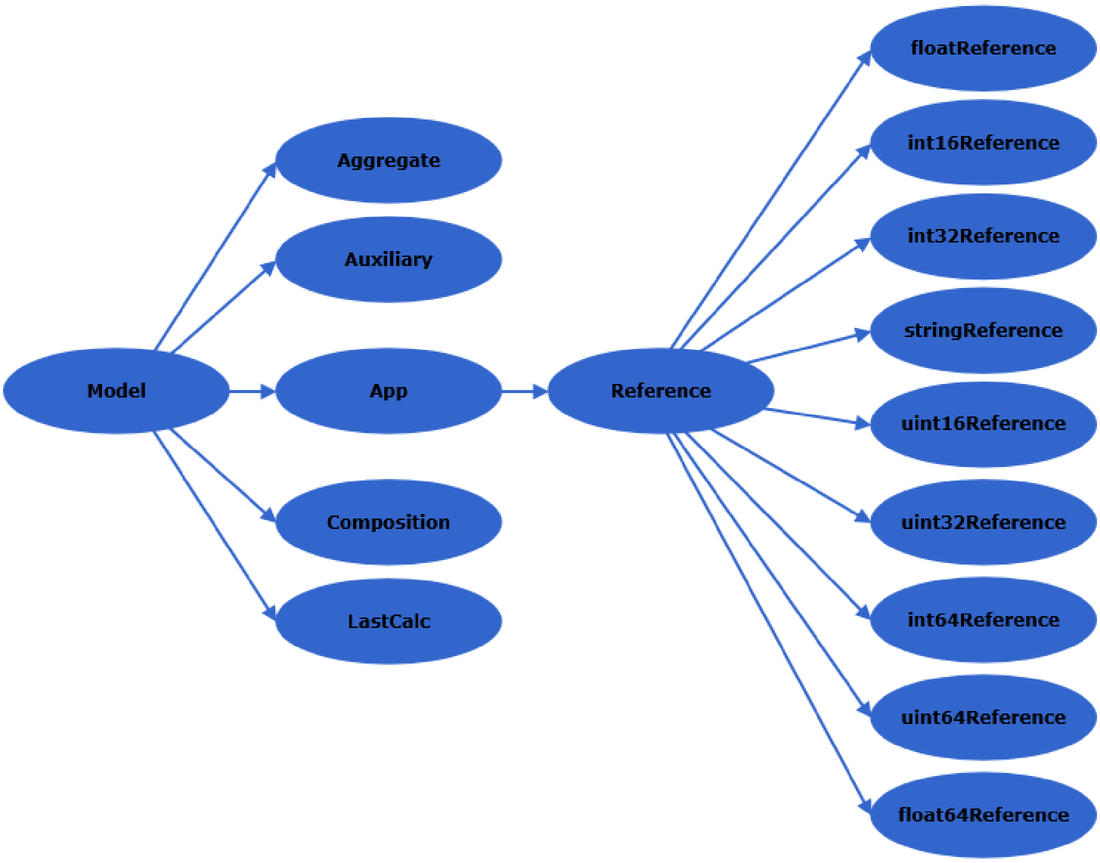
- The model: abb.ability.device indicates that the payload content belongs to a device from any type or instance.
- The model: abb.totalflow.qtr.dailylog indicates that the payload content belongs to daily logs from any type or instance.

Figure 5-1 shows the models across various types and instances.

Figure 5-1: Models defined for ABB Totalflow devices



Some applications, like API Liquid SU and Holding Registers, also support a Reference model which is managed by the References field in the payload. Figure 5-2 shows a reference model for the Holding Registers app. The figure shows the model accessibility from other models.



6 Variable and property elements

Type ID and Object ID definitions include variable and property elements. The variable and property are the equivalent to the tag and register in the app or device.

- The variables category includes characteristics that are expected to change frequently during device operation. For example: static pressure on a tube.
- The properties category includes characteristics that are fixed or that do not change frequently. For example: a device's serial number.

Type definitions include variable and property attribute definitions. Attributes are fixed or pre-defined characteristics that describe each variable or property. In other words, attributes are the variable and property metadata. For example, in Figure 6-1 the variables for the AGA7 application type include the static pressure variable (**staticPressure** in the sample code). The attributes of this variable include **name**, **description**, **data type**, **supported units**, **default value** or **actual value**, **minimum** and **maximum values**, etc.

IMPORTANT NOTE: The full definition of an application has all the attributes for all variables and properties. To simplify, only a portion of the definition is shown in this example. Color or bold style in the code examples is used to highlight discussed elements.

Figure 6-1: Example of attributes for a specific variable (variable definition): staticPressure

```
{
"model": "abb.totalflow.app",
"typeId": "abb.totalflow.AGA7.type",
"name": "AGA7 App",
"version": "1.0.0",
  "variables": {
    "staticPressure": {
        "displayName": "Static Pressure",
        "description": "Static Pressure",
        "eventEnable": false,
        "cloudAccess": "Read",
        "cloudEnable": true,
        "isMandatory": true,
        "dataType": "float",
        "group": "staticPressure",
        "high": {
            "$ref": "#/properties/staticPressureHigh"
        },
        "low": {
            "$ref": "#/properties/staticPressureLow"
         },
        "usedMasured":  true,
        "defaultValue": {
            "value ": 14.73,
             "typeOnly":  false
        },
        "units": "PSIA"
         "tags": [
             "pressure",
```

"base",
"absolute"
]
}
}
}

Figure 6-2 shows the actual value of the staticPressure variable which is published. The payload contains the updated values to publish for the specific application instance (uniquely identified by the Object ID).

The payload only publishes the data updated during a given interval. It does not publish any metadata information. In this example, the device has measured a static pressure of 25 PSIA. This value was not the same as the default (**defaultValue**: **4.73 PSIA** defined in Figure 6-1). Therefore, because there was a change, the payload must carry the new value for publishing. Note that any other variable changes will also be included in the payload. The time stamp must be included.

Figure 6-2: Example of publishable data in the payload: staticPressure value

{
"objectId": "970055662-b63e-4d6e-bb1cde101b97",
"model": "abb.totalflow.app",
"type": "abb.totalflow.AGA7.type@1",
"name": "Application",
"deviceTimestamp": 1610116276,
"deviceUtcTimestamp": 1610096476,
"variables": {
"staticPressure": {
"value": "25",
"units": PSIA,
},
"volumeFlow": {
"value": "85327160",
"units": {
"value ": 5
}
}
}

7 Payload and Data Model for different use cases

The following sections provide examples of the data model and payloads for different applications.

7.1 App register payloads

The data model contains metadata for payloads with the traditional register-based tags/variable/property data. It supports publishing whenever there is a change in the register values.

Figure 7-1 shows an example of register values.

Figure 7-1: Totalflow device register payload

{
"objectId": "d8acc909-e167-44c6-99e1-2b6980760388",

"model": "abb.totalflow.app",
"type": "abb.totalflow.liquid.type@1",
"name": "Application",
"deviceTimestamp": "1599083323",
"deviceUtcTimestamp": "1599063523",
"variables": {
"flowRate": {
"value": "95362992",
"units": {
"value": "bbl/hr"
}
},
"flowingPressure": {
"value": "35.270000457763672",
"units": {
"value": "psig"

"pulseCount": {
"value": "2852",
"units": Counts/Flow Period
}
},
"properties": {
"deviceID": {
"value": "LQ-1"
},
"tubeDescription": {
"value": "Totalflow"
},
"ehnancedMode": {
"value": "0"
}
}
}

See Data Model Examples: RMC-100 JSON files for an example of the of the **API Liquid SU application type definition**. The definitions are required to identify the metadata and to interpret the data carried in the payload for this application.

7.2 Daily EFM Record payloads

The data model supports the payloads with QTR (Quality Transaction Record) data. Daily QTRs are user-configured to be issued by the device once a day. Figure 7-2 shows an example of a payload containing this type of data. Each log has its own unique identifier: **efmLoguuid**. The QTR varies depending on the application type. This example is for the AGA3 Measurement application.

Figure 7-2: Daily Quality Transaction Record (QTR) data example

{

"objectId": "750f7eb5-abf1-4df0-b9dfe26277046e65",
"model": "abb.totalflow.**qtr.dailylog**",
"type": "abb.totalflow.AGA3. **qtr.dailylog**.type@1",
"name": "AGA3 Daily Log",
"**efmLoguuid**": "3283750a-daa5-4718-bfb2-cf2d774ef721",
"sequence": "24",
"periodTime": {
  "units": "seconds",
  "value": "44751",
},
"**variables**": {
  "differentialPressure": {
    "units": {
      "defaultValue": "InH2O",
    },
  "logAverage": {
    "value": 17.969967,
  },
  "maximum": {
    "value": 47.276909,
  }
  "minimum": {
    "value": 0
  }
  },
  "volume": {
    "units": "SCF"
    "value": "288.295563",
    },
    "contractHour": {
      "value": 0,
    }
    "alarms": {
      "value": 2064,
  }
}

See Data Model Examples: RMC-100 JSON files for an example of the of the **AGA3 application daily QTR log type definition**. The definitions are required to identify the metadata and to interpret the data carried in the payload for this application.

7.3 Custom EFM Record payloads

The data model supports the payloads with custom QTR (Quality Transaction Record) data. Custom QTRs are issued by the device at user-defined time intervals. Figure 7-3 shows an example of a payload containing this type of data. The QTR varies depending on the application type. This example is for the AGA7 Measurement application.

Figure 7-3: Custom Quality Transaction Record (QTR) data example


```
{
"model": "abb.totalflow.qtr.customlog",
"type": "abb.totalflow.AGA7.qtr.customlog.type@1",
"name": "AGA7CustomLog",
"objectId": "2a5a8f5c-e65a-11e8-9f32-f2801b9fd1",
"description ": "Displays all variables in AGA7 related to custom log",
"timeStamp": "2011-08-12T20:17:46.384Z",
"efmLoguuid": "2a5a8f5c-e65a-11e8-9f32-f2801f1b9fd1",
"sequence": 14,
"periodTime": {
    "units": "seconds",
    "value": 0
},
 "variables": {
    "staticPressure": {
      "units": " ",
      "value": 0
      },
    "pulseCount": {
      "units": " ",
     "value": 0
    },
    "temperature": {
       "units": " ",
       "value": 0,
    },
    "volume": {
      "units": " " ,
      "value": 0
    },
    "energy": {
      "units": " ",
      "value": 0
    },
    "flowTime": {
      "units": " ",
      "value": 0
    }
  }
}
```

See Data Model Examples: RMC-100 JSON files for an example of the of the **AGA7 application custom QTR log type definition**. The definitions are required to identify the metadata and to interpret the data carried in the payload for this application.

7.4 Alarm payloads

The data model supports the payloads with Alarm data. Alarms are the warnings generated by the devices or the applications instantiated in the devices. The two types of alarm data supported are:

- Alarm Definition data: User-defined conditions that trigger an alarm
- Alarm Log data: Alarms logged when user-defined conditions are met

7.4.1 Alarm log payload

The Alarm log record is created as soon as the alarm condition occurs and is sent from the Totalflow device to the MQTT broker. The log record data also provides the ID associated with the alarm definition that is evaluated to trigger the alarm. Each alarm log has an identifier (**alarmloguuid**) which is globally unique.

Figure 7-4 shows an example of alarm data in the payload. The example shows two alarms, each uniquely identified by its own Object ID. Each alarm also includes the reference to the alarm definition with its own unique Object ID.

Figure 7-4: Example of payload with Alarm log data for the Plunger Control application


```
{
"objectId": "2a5a8f5c-e65a-11e8-9f32-f2801b9fd1",
"model": "abb.totalflow.alarmlog",
"type": "abb.totalflow.plunger.alarmlog.type@1",
"alarmLogs": [
  {
    "alarmLoguuid": "2a5a8f5c-e65a-11e8-9f32-f2801b9fd1",
    "name": "Plunger – Tubing ID",
    "value": 0,
    "deviceTimestamp": 1704425842,
    "deviceUtcTimestamp": 1704447442,
    "seqNum": 119950,
    "state": 1,
    "ackStatus": 1
    "severity": 200,
    "variableName": "tubing"
    "unit": "PSIA"
    "alarmType": “useDefined”
    "alarmdefinitionRef": "3dd919f7-65d3-492c-9878-3996ecaebd5c"
  },
 {
    "alarmLoguuid": "7458361e-14603-4ea7-9dbc-b24eb68f6c31",
    "name": "Plunger- Casing ID"
    "value": 0,
    "deviceTimestamp": 1904427442,
    "deviceUtcTimestamp": 1904447442,
    "seqNum": 129950,
    "state": 1,
    "ackStatus": 1,
```

"severity": 200,
"variableName": "casing"
"unit": "PSIA"
"alarmType": "userDefined"
"**alarmDefinitionRef**": "cbf632dc-7789-4703-8c24-7996b034e017"
}
]
}

See Data Model Examples: RMC-100 JSON files for an example of the **Plunger Control application alarm log type definition**. The definitions are required to identify the metadata and to interpret the data carried in the payload for this application.

7.4.2 Alarm definition payload

An alarm definition is a user-defined condition that triggers an alarm. For each alarm definition, users establish the logic that compares operating device or application variables with user-defined acceptable values. The alarm application evaluates and monitors those values to determine if they fall within or outside accepted or expected values.

Alarms are generated based on this definition. The Totalflow device sends and publishes all alarm definitions on the MQTT broker.

Figure 7-5: Example of payload with Alarm definition data for the AGA3 application

{
"model": "abb.totalflow.alarmdefinition",
"type": "abb.totalflow.AGA3.alarmdefinition.type@1",
"name": "Alarm Definition Type",
"objectId": "6a819bc3-58c5-9811-e8bd9bbcf959",
"deviceTimestamp": 1610116279,
"deviceUtcTimestamp": 1610096479,
"**alarmDefinitions**": [
{
"**alarmDefuuid**": "b3b7407c-4567-4430-b46f-86d883364374",
"name": "AGA3 -SP",
"severity": 100,
"alarmType": "userDefined",
"condition": 1,
"inputVariableRf": {
"variableName": "staticPressure",
"modelId": "abb.totalflow.app"
},
"thresholdRef": {
"const": 10
},
"supressref ": {
"const: 0
},
"filter": {

"type: 0,
"unit": 0,
"value": 2
}
},
{
"alarmDefuuid": "4ccdd53fb-3456-9ddb-5aa01516c164"
"name": "AGA3 – Temp",
"severity": 200,
"alarmType": "userDefined",
"condition": 1,
"inputVariableRf": {
"variableName": "temperature",
"modelId": "abb.totalflow.app"
},
"thresholdRef": {
"const": 0
},
" supressref ": {
"const: "useDefined"
}
"filter: {
"type: 0,
"unit": 0,
"value": 2
}
}
]
}

See Data Model Examples: RMC-100 JSON files for an example of the **AGA3 application alarm definition type**. The definitions are required to identify the metadata and to interpret the data carried in the payload for this application.

7.5 Event payload

The data model supports the payloads with Event data. Events are significant occurrences in the devices or their active applications. Events can be triggered by the system or by user actions. Events are logged and tracked for audit purposes. Each event log has a globally unique identifier: **eventLoguuid**.

Figure 7-6 shows the device or application event information that is included in the payload issued by a Totalflow device when the event occurs.

Figure 7-6: Example of payload with Event data for the Gas lift application

{
"model": "abb.totalflow.eventlog",
"type": "abb.totalflow.gaslift.eventlog.type@1",
"name": "Event Log Type Definition For Gas Lift application",

"objectId": "5fcaba55-b270-4196-9e18-50685c9041f6",
"**eventLogs**": [
{
"deviceTimestamp: 158761736,
"deviceUtcTimestamp ": 1587651936,
"**eventLoguuid**": "77bb5468-0bd7-4fc3-8048-49bfad561671",
"mode": 3,
"event": 0,
"plunger": 0,
"prodRate": {
"value": 0.057476401329040527,
},
"injectRate": {
"value": 0,
},
"criticalRate": {
"value": 469.44125366210938,
},
"injSetPoint": {
"value": 0,
}
}
]
}

See Data Model Examples: RMC-100 JSON files for an example of the **Gas Lift application event log type definition**. The definitions are required to identify the metadata and to interpret the data carried in the payload for this application.

7.6 Trend payloads

The data model supports the payloads with Trend data. Trends are user-selected variables chosen to be monitored and stored by the Totalflow device. The two types of trend data supported are:

— Trend Definition: a snapshot of a user-selected variable set in the device or specific app instance
— Trend log: a snapshot of a set of variables based on the trend definition

7.6.1 Trend log payload

Each trend log has a globally unique ID: **trendLoguuid**. Figure 7-7 shows an example of the trend log payload. Note that the log also has the associated trend definition reference Object ID: **trendDefinitionRef**.

Figure 7-7: Example of trend log payload for the Holding Registers application

{
"objectId": "2648f20f-bdad-4df1-a121-1471620c71dd",
"model": "abb.totalflow.trendlog",
"**trendLoguuid**": "2fc257a7c-a363-4348-913e-a0bcfffa74c8",
"deviceTimestamp": 1619101141,
"deviceUtcTimestamp": 1619081341,
"type": "abb.totalflow.holdingregisters.trendlog.type@1",

"trendDefinitionRef": "022afbf1-a80b-478d-8db5-053bf5d553c6",
"periodTime": {
  "value": 30,
  "unit": "sec"
},
"variables": [
 {
    "name": "AGA7 - Temp",
    "value": 125,
    "unit ": "Deg F",
 },
    "name": "AGA3 - Static Pressure",
    "value": "14.7299995422635"
    "unit": "PSIA"
 }
]
}

See Data Model Examples: RMC-100 JSON files for an example of the **Holding Registers application trend log type definition**. The definitions are required to identify the metadata and to interpret the data carried in the payload for this application.

7.6.2 Trend Definition payload

Each trend definition has a globally unique ID: **trendDefuuid**. Figure 7-8 shows an example of the trend definition payload.

Figure 7-8: Example of trend definition payload for the Plunger Control application

{
 "trendDefuuid": "04035cd2c-a3ea-4afb-aed5-874acda9208",
 "objectId": "6e392e5f-f00e-4438-9790-0bde1f7e1c54",
 "model": "abb.totalflow.trenddefinition",
 "type": "abb.totalflow.plunger.trenddefinition.type@1",
 "interval": {
    "value": 60,
    "unit": "sec"
 },
 "variables": [
  {
    "name": "flowRate",
    "displayName": "Flow Rate",
    "$ref": "6e392e5f-f00e-4438-9790-0bde1f7e1c54/abb.totalflow.app/variables/flowRate",
    "unit": "SCF/Hr"
  }
 ]
}

See Data Model Examples: RMC-100 JSON files for an example of the **Plunger Control application trend definition type**. The definitions are required to identify the metadata and to interpret the data carried in the payload for this application.

8 Keys in the Data Model

As observed in examples provided in this document, definitions in the data models include several "**key**":"**value**" pairs. Table 8-1 lists some of these keys (schema definition).

Table 8-1: Data model keys

Key	Description
relatedModels	Holds array of object for all related models defining additional Type ID data categories
version	Version of the type definition
description	Description of the type definition
isMandatory	Indicates whether a field can be enabled or disabled from the MQTT REST interface. The possible values are: — True — False For example, the Register configuration page lists all the registers or variables that can be published for an application. Some of these registers are required and users cannot disable their publishing. For these registers the **IsMandatory** attribute is set to: True. The page shows these variables with the **Enable** checkbox selected and grayed out to indicate that they are required always. A variable with the **isMandatory** attribute set to: False, allows the user to enable or disable for publishing.
variables	Defines all the variables for a Type and Model
properties	Defines all the properties for a Type and Model
displayName	Defines the name used for display purposes
cloudEnable	Indicates whether the variable or property can be accessed by an ABB cloud application. The possible values are: — True — False
cloudAccess	Indicates whether the variable or property can be updated from an ABB cloud application. Possible values are: — Read — Read/Write
dataType	Defines the data type of a variable or property
dataTypeExt	Defines the data type for a variable or property with respect to C/C++

9 Useful terms

Table 9-1 provides some definitions of terms and acronyms used in data model concepts. For more details on Internet standards and protocols, search online resources.

Table 9-1: Useful terms

Term	Definition
ABB Ability	A set of tools, software processes and data models available for each ABB cloud-based domain-specific solution. The Totalflow web applications are solutions specific for oil and gas upstream production and constitute one of the many ABB solutions offered on the cloud.
UUID	A UUID (Universally Unique Identifier) is a 128-bit number used to uniquely identify some object or entity on the Internet. Depending on the specific mechanisms used, a UUID is either guaranteed to be different or is, at least, extremely likely to be different from any other UUID generated until 3400 AD. ABB Totalflow devices support the automatic standards-based generation of these identifiers to assign Object IDs to unique device or application instances.
Payload	The actual data in a packet or file minus all headers attached for transport and minus all descriptive meta-data. In a network packet, headers are appended to the payload for transport and then discarded at their destination. The

Term	Definition
	payload format depends on the communication protocol used: MQTT or Sparkplug.
EFM	Electronic Flow Measurement
MQTT	Message Queue Telemetry Transport A client-server publish-subscribe messaging protocol for use on top of the TCP/IP protocol. This protocol enables connectivity and integration of field devices into the cloud. Packet payload for the standard MQTT protocol supports the ABB Ability format.
MQTT client	Functionality that performs the client role in MQTT communication. Typically implemented on field devices.
MQTT server	Functionality that performs the server role in MQTT communication. Typically implemented on systems serving as IoT hubs or MQTT brokers.
MQTT-enabled field device	ABB Totalflow devices with embedded capability to connect and communicate with an MQTT broker. These devices support the MQTT client functionality which requests connections to the broker and establishes the communication links for data transfer to and from the broker.
MQTT Service	Service available in some Totalflow devices like the RMC-100. When enabled, the device MQTT client capabilities are enabled. This service is disabled by default. See Additional information for links to the MQTT Configuration Guide for instructions to enable this service and configure parameters successfully.
MQTT Broker	The system with the MQTT server functionality that authenticates and accepts connection requests, establishes communication links, and allows data transfer for MQTT clients. Customers may have their own private brokers as part of their custom implementations.
REST (Architecture)	Representational State Transfer Architecture Web standards-based architecture based on the HTTP Protocol methods. It consists of REST Servers providing REST clients services and access to resources.
REST Service	Service available in some Totalflow devices like the RMC-100. When enabled, the device REST server capabilities are enabled. The device then can be accessed by a REST client such as a web browser. The access is for the configuration of the MQTT parameters. This service is disabled by default. See Additional information for links to the MQTT Configuration Guide for instructions to enable this service and configure parameters successfully.
REST Client	Uses HTTP methods (protocol) to access resources on a REST server. For example, the web browser which accesses the MQTT configuration interface on the RMC-100.
Sparkplug	Communication protocol that enhances the standard MQTT protocol to support field device connection with real-time SCADA or IIoT systems. The Sparkplug packet payload format is different from the format used by standard MQTT. Sparkplug requires specific payload format definitions.

ABB Inc.

Measurement & Analytics

Quotes: US-IAMA.inquiry@us.abb.com

Orders: US-IAMA.order@us.abb.com

Training: US-IAMA.training@us.abb.com

Support: upstream.support@us.abb.com

+1 800 442 3097 (opt. 2)

Additional free publications are available for download at:
www.abb.com/upstream

Main Office - Bartlesville
7051 Industrial Blvd
Bartlesville, OK 74006
Ph: +1 918 338 4888

Kansas Office - Liberal
2705 Centennial Blvd
Liberal, KS 67901
Ph: +1 620 626 4350

Texas Office - Houston
3700 W. Sam Houston
Parkway S., Suite 600
Houston, TX 77042
Ph: +1 713 587 8000

Texas Office – Odessa
8007 East Business 20
Odessa, TX 79765
Ph: +1 432 272 1173

Texas Office – Pleasanton
150 Eagle Ford Road
Pleasanton, TX 78064
Ph: +1 830 569 8062

2107721MNAA