Enabling Configurable Data Transmission With Moxa's Advanced IIoT Gateways

Moxa Technical Support Team
support@moxa.com

Contents

Released on February 17, 2023

About Moxa

Moxa is a leading provider of edge connectivity, industrial computing, and network infrastructure solutions for enabling connectivity for the Industrial Internet of Things. With 35 years of industry experience, Moxa has connected more than 82 million devices worldwide and has a distribution and service network that reaches customers in more than 80 countries. Moxa delivers lasting business value by empowering industry with reliable networks and sincere service for industrial communications infrastructures. Information about Moxa's solutions is available at www.moxa.com.

How to Contact Moxa

Tel: 1-714-528-6777
Fax: 1-714-528-6778

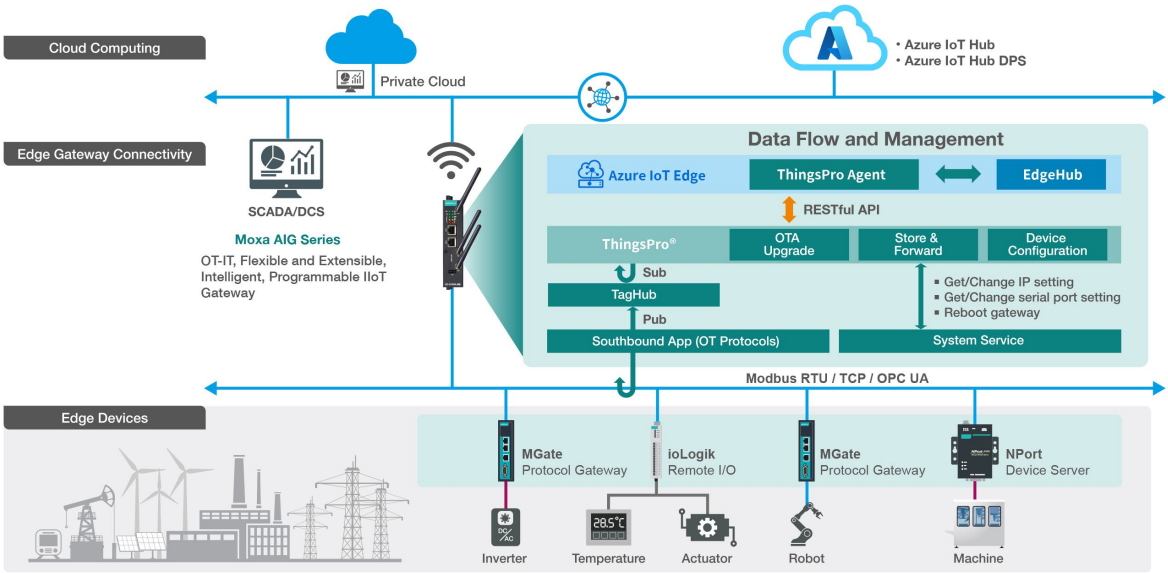
1 Overview

With industrial IoT (IIoT) applications all the rage, the gateways that sit between the "things" and the Internet have become extremely important. A gateway acquires data from sensors and actuators in the field, and then transfers the data to the cloud, enabling smooth data flow from field to cloud. To ensure the stability of the data flow, all aspects of the system, including data acquisition, message payloads, and publishing messages at requested intervals for cloud applications, must be properly configured. Although many systems use a "fixed configuration" for data transmission, this option lacks flexibility and the capability to efficiently adapt to a variety of IIoT applications. This is particularly true for complex OT-IT protocol conversions, where in data exchange and data processing must be completed before data is sent to the cloud.

That's where our AIG family of advanced IIoT gateways come in. The AIG gateways support configurable data transmission, using **message groups** and **custom payload**, to make it easier for users to overcome any issues that arise with respect to data exchange and the adaptation of the payload format for cloud applications.

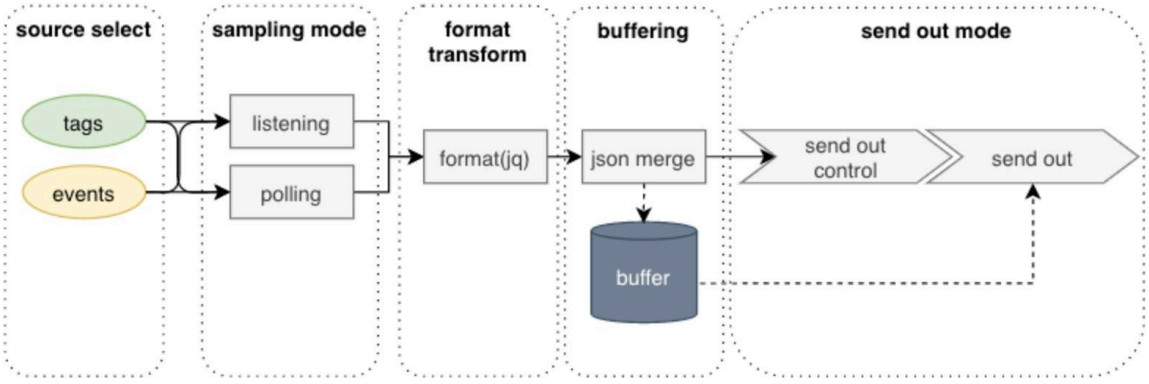
2 Moxa's Advanced IIoT Gateways—The AIG Series

To make configurable data transmission a reality, AIG gateways use the TagHub data repository to store all acquired data. The gateways support a variety of methods for retrieving data from the TagHub repository, making it easy to customize your device-to-cloud message payload.



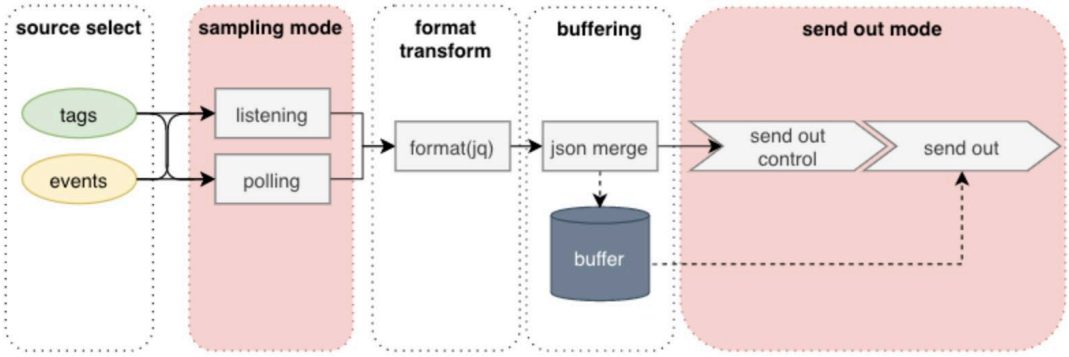
3 Configurable Data Transmission

The AIG gateways use a five-stage configuration strategy that allows users to efficiently adapt the configuration to a variety of IIoT applications. The five stages are referred to as **source select**, **sampling mode**, **format transform**, **buffering**, and **send out mode**. The gateways also provide two outstanding features, **Message Groups** and **Custom Payload**, discussed in detail below.

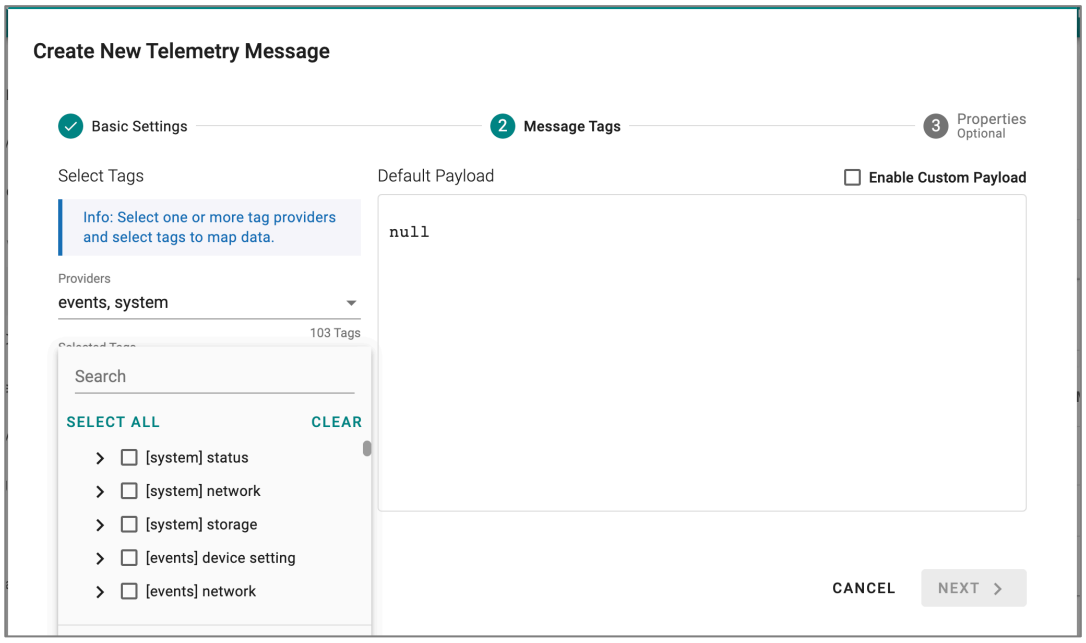


3.1 Message Groups

The ability to specify which tags and events should be included in a telemetry message (source select), using the tags to specify conditions for acquiring data (sampling mode), and sending out the messages based on a given time interval or size (send out mode), allow users to categorize messages into well-defined **Message Groups**.

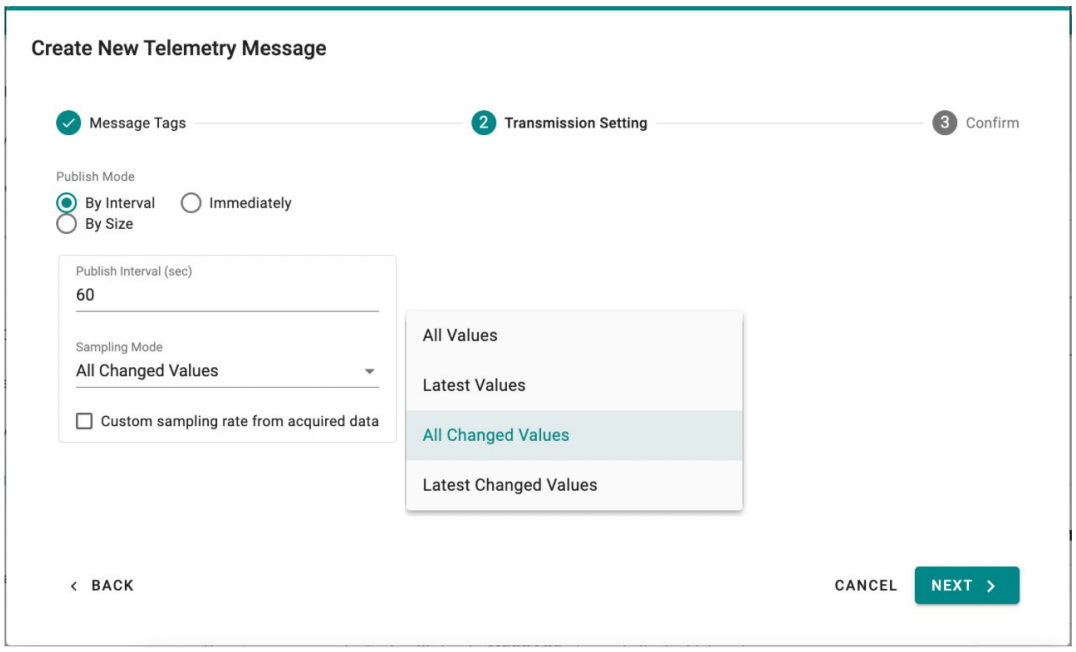


For source selection, system status and events of the gateway are selectable via a dropdown list, and users can also add a telemetry message.

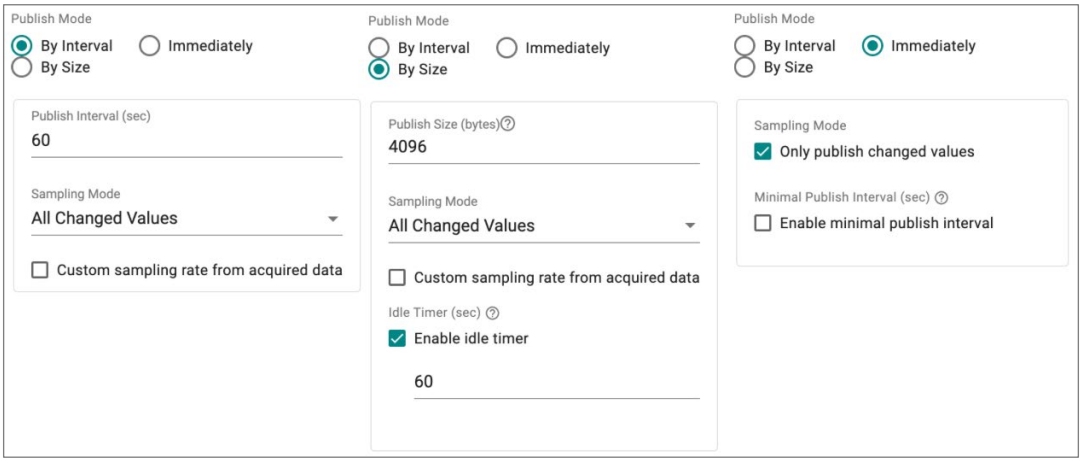


Sampling mode has four options. All or some of the data is buffered when it is received depending on the option chosen, and then only data that is buffered is published after an entire message group is received.

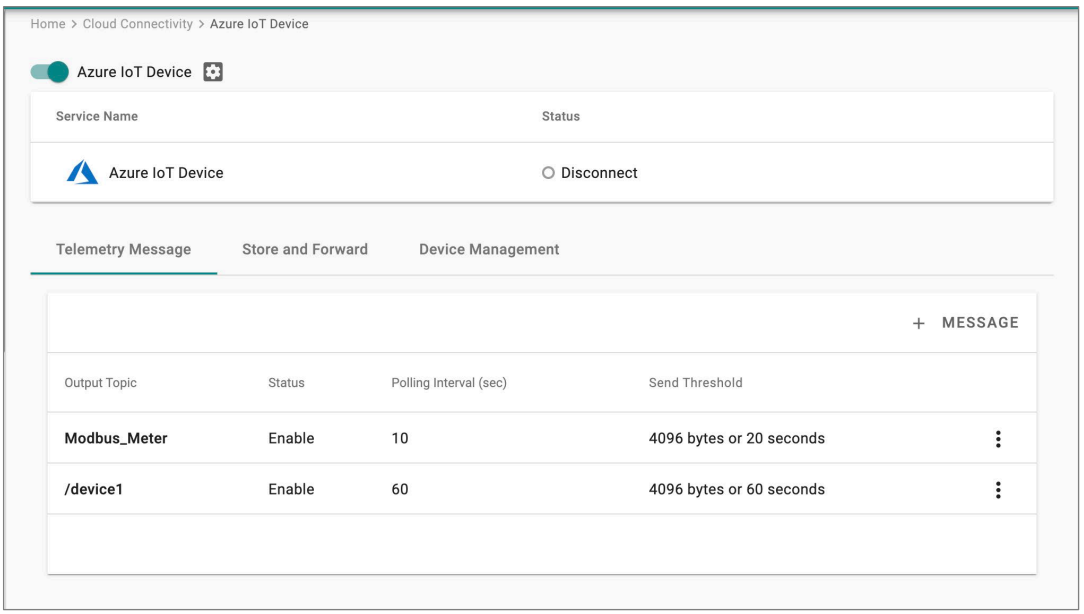
No.	Mode	Description
1	All Values	All of the data is buffered and then published.
2	Latest Values	Only data received in the last sampling period is buffered and then published.
3	All Changed Values	All data values that have changed compared with the latest values are buffered and then published.
4	Latest Changed Values	Only data that is received in the last sampling period and that has changed compared with the latest values is buffered and then published.



The "send out mode" is used to define how a message group is formed. The three options for "send out mode" are **interval** (time), **size**, and **immediately**. Enable **Custom sampling rate from acquired data** to specify the sampling frequency.

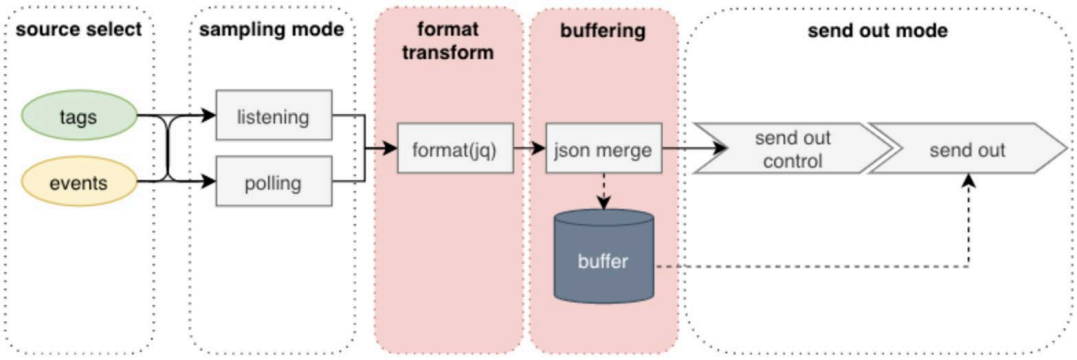


In addition to message groups, the AIG gateways alllow users to create messages for specific purposes. For example, a device's status could be monitored frequently within a short time period, whereas environmental data (temperature, for example) could be sampled over a longer time period.



3.2 Custom Payload

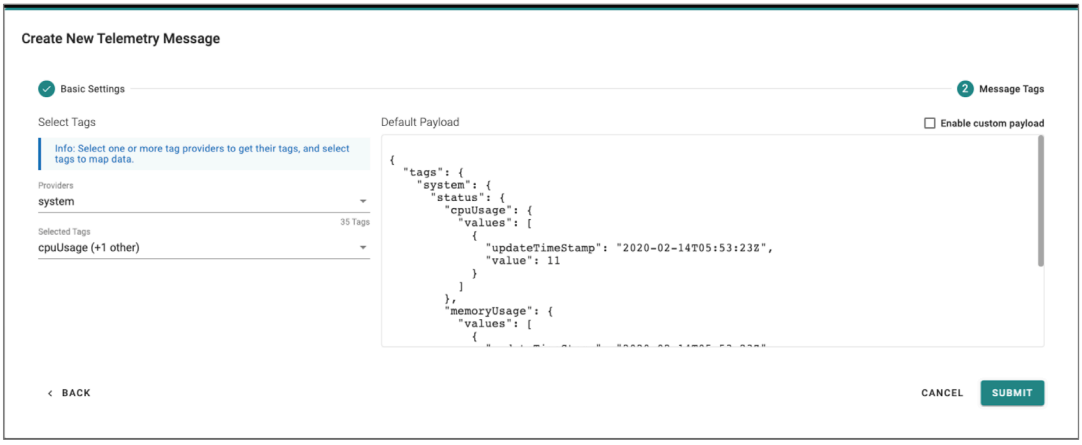
Once the tags and events are selected, the AIG gateways support using a jq filter[1] to transform the default payload to your desired payload schema. This operation is called **Custom Payload**. The tags can also be merged into buffered messages using the transformed payload. This action is called a **JSON merge**.



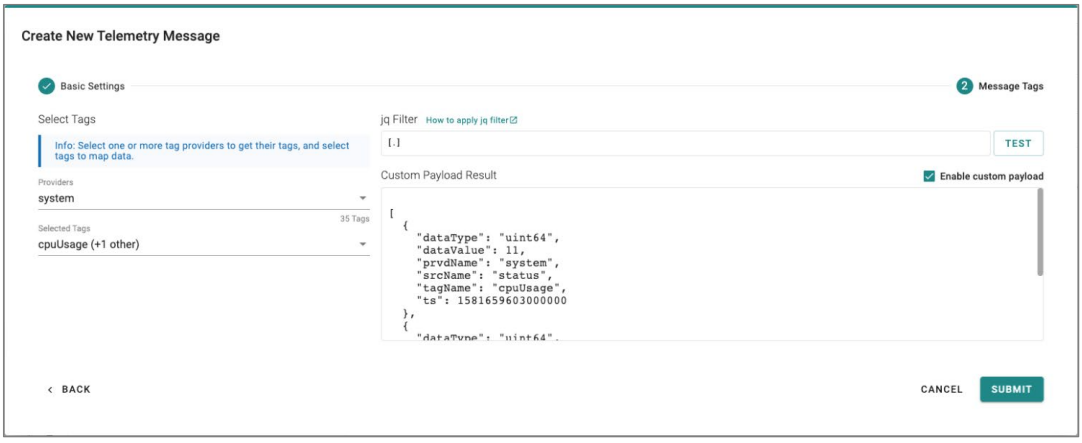
The AIG offers a web GUI to apply the jq filter, making it possible to transform the default payload and display the output. After the tags or events are selected, a default

[1] *jq filter is a lightweight and flexible command-line JSON processor. Click here for detailed information.*

payload will be shown on the right side.



Select the **Enable custom payload** checkbox to display a text field for entering the jq filter. You can enter **[.]** to display the output within an array, as shown below.



The AIG gateways provide several keys that can be used to customize your format.

No.	Key	Description
1	prvdName	Provider name of tag
2	srcName	Source name of tag
3	tagName	Tag name
4	dataValue	Tag value
5	ts	Time stamp when the tag value is collected
6	dataType	Data type of tag value. (e.g.: int64)

For example, let us consider a tag that contains the following keys and values:

```
{
  "dataType": "int32",
  "dataValue": 11,
  "prvdName": "modbus_tcp_master",
  "srcName": "A1",
  "tagName": "status",
  "ts": 1581659603000000
}
```

Apply the jq filter:

```
{
    device: .srcName,
    timestamp: (now|todateiso8601),
    tag: [
        {
            name: .tagName,
            value: .dataValue
        }
    ]
}
```

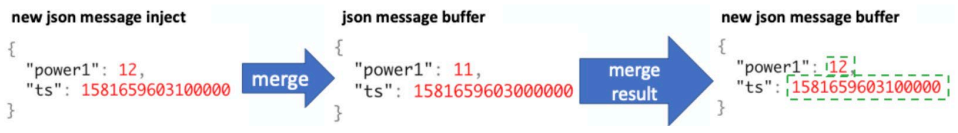
The transformed result will look like this:

```
{
  "device": "A1",
  "tag": [
    {
      "name": "status",
      "value": 11
    }
  ],
  "timestamp": "2020-11-10T03:58:16Z"
}
```

For **JSON merge**, one of two modes can be used to merge tags into buffered messages.

- **Object Override**
  This mode is used to only record the last value.
  The payload is formatted by the following jq filter:

```
{
  (.tagName): .dataValue,
  ts: .ts
}
```



- **Array Append**

  This mode creates an array of tag data, with the latest data appended to data that was already received.

  The payload is formatted by the following jq filter:

```
{
  (.tagName): [
    {
      value: .dataValue,
      ts: .ts
    }
  ]
}
```