robotron®

RoboGate Edge - Anwenderdokumentation

version 2024.04

Inhalt

1. Einleitung . . . 1
   1.1. RoboGate Edge UI . . . 1
   1.2. Module . . . 2
      1.2.1. Übersicht . . . 2
      1.2.2. Topics . . . 2
      1.2.3. Nachrichten . . . 4
   1.3. Varianten . . . 5
      1.3.1. RoboGate Device . . . 5
      1.3.2. RoboGate Container . . . 5
2. Inbetriebnahme . . . 7
   2.1. RoboGate Devices . . . 7
      2.1.1. Technische Daten . . . 7
      2.1.2. Herstellung der Netzwerkverbindung & Erstinbetriebnahme . . . 7
   2.2. RoboGate Container . . . 7
      2.2.1. Voraussetzung . . . 7
      2.2.2. Installation des Images . . . 8
3. Input-Module . . . 10
   3.1. Modbus . . . 10
      3.1.1. Konfigurationsparameter . . . 10
         Server . . . 10
         Slaves . . . 11
         Setup Groups . . . 11
         Poll Groups . . . 12
         Subscription Groups . . . 13
      3.1.2. Konfiguration in der UI . . . 15
         Server konfigurieren . . . 15
         Group konfigurieren . . . 15
         Datenfeld löschen . . . 17
         Group löschen . . . 17
         Server löschen . . . 18
         Konfiguration zurücksetzen . . . 18
   3.2. OPC UA . . . 19
      3.2.1. Konfigurationsparameter . . . 19
         Server Connection . . . 19
         Authentication . . . 20
         Subscriptions . . . 21
         Method Mapping . . . 21
         Method Error Code Mapping . . . 22

Poll Groups . . . 22
Subscription Groups . . . 23
Alarms & Conditions . . . 24
3.2.2. Konfiguration in der UI . . . 25
Server konfigurieren . . . 25
Subscription konfigurieren . . . 25
Method Mapping konfigurieren . . . 27
Error Code Mapping konfigurieren . . . 27
Group konfigurieren . . . 28
Alarms & Conditions konfigurieren . . . 29
Node Browser-Konfiguration bearbeiten / löschen . . . 30
Datenfeld löschen . . . 30
Group löschen . . . 31
Konfiguration zurücksetzen . . . 31
3.3. RFC 1006 . . . 32
3.3.1. Konfigurationsparameter . . . 32
PLC . . . 32
Poll Groups . . . 32
Subscription Groups . . . 34
3.3.2. Konfiguration in der UI . . . 35
PLC konfigurieren . . . 35
Group konfigurieren . . . 36
Datenfeld löschen . . . 39
Group löschen . . . 39
PLC löschen . . . 40
Konfiguration zurücksetzen . . . 40
4. Processing-Module . . . 42
4.1. File Logger . . . 42
4.2. Stream Processor . . . 42
4.2.1. Konfigurationsschema . . . 42
Input . . . 43
Processing . . . 46
Output . . . 49
4.2.2. Konfiguration zurücksetzen . . . 51
4.2.3. Aggregator . . . 52
Funktionsbeschreibung und Anwendung . . . 52
Beispielkonfiguration . . . 54
4.2.4. Dedup . . . 55
Funktionsbeschreibung und Anwendung . . . 55
Beispielkonfiguration . . . 56
Konfiguration in JSON . . . 56

4.2.5. Edge Method . . . 57
Funktionsbeschreibung und Anwendung . . . 57
Beispiel: Aktualisieren eines OPC UA Nodes . . . 57
Häufig verwendete Methoden . . . 57
4.2.6. Expression . . . 64
Funktionsbeschreibung und Anwendung . . . 64
Ausdrücke . . . 65
Benutzung von regulären Ausdrücken (Regex) . . . 66
Konstanten . . . 66
Umrechnung von Zeiten . . . 66
Beispielkonfiguration . . . 67
4.2.7. Join . . . 67
Funktionsbeschreibung und Anwendung . . . 67
Beispielkonfiguration . . . 68
4.2.8. Resend . . . 69
Funktionsbeschreibung und Anwendung . . . 69
Beispielkonfiguration . . . 70
4.2.9. Router . . . 70
Funktionsbeschreibung und Anwendung . . . 70
Beispielkonfiguration . . . 71
Konfiguration in JSON . . . 73
4.2.10. RuleEngine . . . 74
Funktionsbeschreibung und Anwendung . . . 74
Bezeichnungen und Syntax . . . 75
Beispielkonfiguration . . . 77
4.2.11. SNCFilter . . . 84
Funktionsbeschreibung und Anwendung . . . 84
Anmerkungen . . . 86
Beispielkonfiguration . . . 86
Konfiguration in JSON . . . 87
4.2.12. Timer . . . 87
Funktionsbeschreibung und Anwendung . . . 87
Beispielkonfiguration . . . 90
4.3. Template Enricher . . . 91
4.3.1. Konfigurationsparameter . . . 91
4.3.2. Konfiguration in der UI . . . 92
Settings . . . 92
Header . . . 93
Body . . . 94
Library . . . 94
Modules . . . 95

Environment . . . 95
Test . . . 96
Message Output . . . 97
Konfiguration zurücksetzen . . . 98
4.3.3. Konfiguration in JSON . . . 98
5. Output-Module . . . 100
5.1. Azure IoT Hub . . . 100
5.1.1. Konfigurationsparameter . . . 100
Connection . . . 100
Permissions . . . 101
Nachrichten / Topics . . . 101
5.1.2. Device Provisioning (DPS) . . . 102
5.1.3. Konfiguration in der UI . . . 104
Connection konfigurieren . . . 104
Topic löschen . . . 105
Konfiguration zurücksetzen . . . 106
5.2. MQTT . . . 106
5.2.1. Konfigurationsparameter . . . 106
Connection . . . 106
Nachrichten / Topics . . . 107
5.2.2. Konfiguration in der UI . . . 108
Connection konfigurieren . . . 108
Topics löschen . . . 110
Konfiguration zurücksetzen . . . 111
5.3. OPC UA Server . . . 111
5.3.1. Settings . . . 111
Parameter . . . 112
5.3.2. Adress Space Configuration . . . 113
Parameter . . . 114
5.3.3. Matching Rules Overview . . . 115
Parameter . . . 116
5.3.4. Nodes löschen . . . 116
5.3.5. Konfiguration zurücksetzen . . . 116
5.3.6. Module State . . . 117
5.4. Splunk . . . 117
5.4.1. Konfigurationsparameter . . . 118
Connection . . . 118
Settings . . . 118
Nachrichten / Topics . . . 119
5.4.2. Konfiguration in der UI . . . 119
Connection konfigurieren . . . 119

Nachrichten konfigurieren . . . 120
Topic löschen . . . 122
Konfiguration zurücksetzen . . . 122
6. Management-Module . . . 124
6.1. Control Panel . . . 124
6.1.1. Status . . . 125
6.1.2. Network . . . 126
6.1.3. Network Configuration . . . 126
6.1.4. Software Update . . . 127
6.2. ControlCenter . . . 128
6.2.1. Konfigurationsparameter . . . 128
6.2.2. Konfiguration in der UI . . . 129
6.3. System Management . . . 130
6.3.1. Log . . . 130
6.3.2. Edge Insights . . . 131
Oberfläche . . . 132
Settings (Internal Topics) . . . 132
Sonstige Anmerkungen . . . 132
6.3.3. User Settings . . . 132
6.3.4. Device Configuration . . . 133
6.3.5. Metrics . . . 134
6.3.6. Modules Status . . . 134
6.3.7. System Signals . . . 135
6.4. About us . . . 137
6.4.1. Software Versions . . . 137
6.4.2. Software Licenses . . . 138

1. Einleitung

Das RoboGate Edge ist Robotrons innovative IoT-Fieldgateway-Lösung.

Angepasst auf moderne Industriestandards adressiert das RoboGate Edge Szenarien und Anwendungsgebiete der Maschinen- und Energieeffizienz sowie der Zustandsüberwachung in der Produktion.

Das RoboGate Edge dient dem Empfang von Maschinendaten aus Steuerungen durch Unterstützung verschiedener Protokolle in Verbindung mit Ethernet-basierter Kommunikation. Es ermöglicht eine Vorverarbeitung von Daten und stellt diese zur Weiterverarbeitung auf einem Zielsystem zur Verfügung. Für die steuerungsspezifischen Kommunikationsprotokolle (z.B. RFC 1006 und OPC UA) sind verschiedene Module implementier- und konfigurierbar. Die Definition des Zielsystems erfolgt in der Modulkonfiguration. Eine einfache Möglichkeit zur Konfiguration steht über die RoboGate Edge UI zur Verfügung.

1.1. RoboGate Edge UI

Die Administration eines RoboGate Edge kann über die RoboGate Edge UI, eine Web Browser-Anwendung, erfolgen. Diese kann mit den aktuellen Versionen der Browser Google Chrome und Microsoft Edge verwendet werden. Der Internet Explorer und Mozilla Firefox werden nicht unterstützt.

Das Dashboard (Abbildung 1) ist die Startseite der RoboGate Edge UI und dient dem Überblick aller vorhandenen Module. Durch einen Klick auf die jeweilige Kachel im Dashboard erfolgt die Weiterleitung zum ausgewählten Modul. Die Auswahl kann ebenso über die linke, erweiterbare Menüleiste erfolgen.

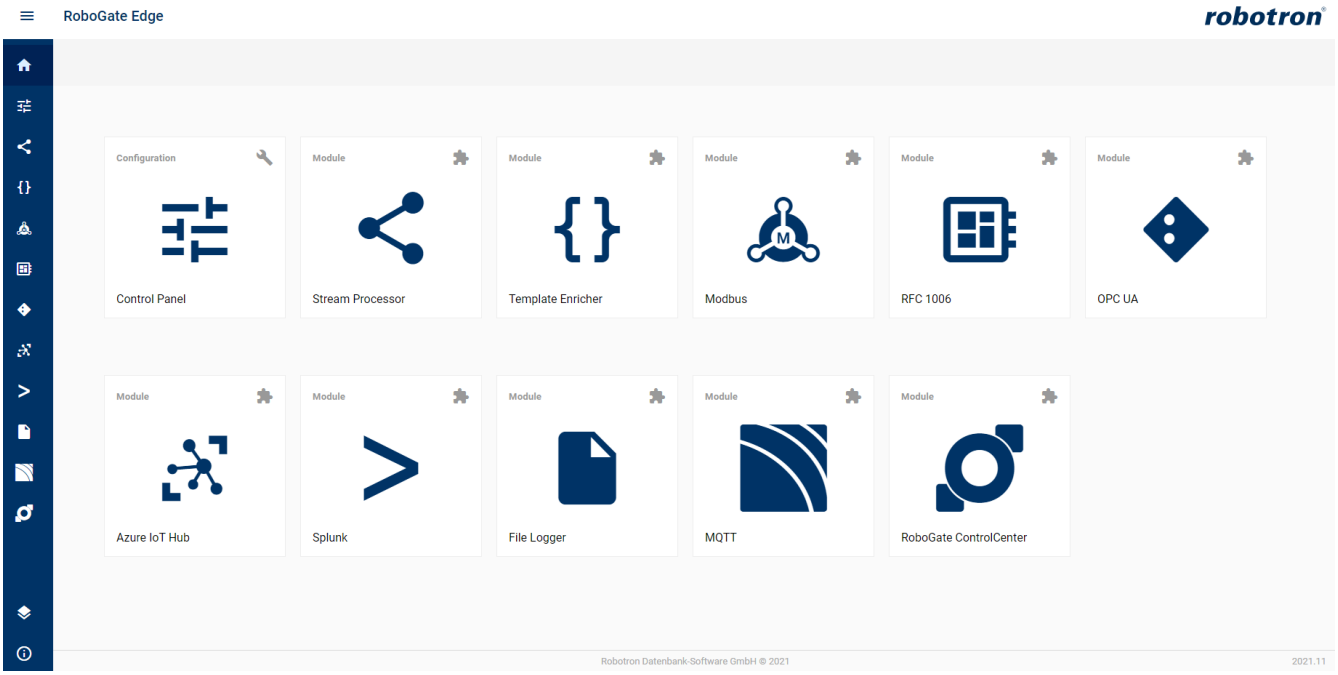


Abbildung 1. RoboGate Edge UI - Dashboard

1.2. Module

RoboGate Edge verfügt über die nachfolgend aufgelisteten Module.

RoboGate Edge Connect bezeichnet das standardmäßig ausgelieferte Bundle an Modulen. Das Hinzufügen weiterer Module oder Entfernen von nicht benötigten Modulen ist auf Anfrage möglich.

1.2.1. Übersicht

Modul	Beschreibung	Edge Connect
Modbus	Schreiben und Auslesen von Slave Modbus Modulen	✓
OPC UA	OPC UA Client zum Auslesen eines OPC UA Servers	✓
OPC UA Server	Konfiguration eines OPC UA Servers bzw. Node-Tree aus EdgeMessages	✗
RFC 1006	Auslesen einer SPS über ISO on TCP	✓
File Logger	Logging von Telemetrie-Daten auf ein File-System	✓
Stream Processor	Datenverarbeitung	✓
Template Enricher	Datentransformation	✓
Azure IoT Hub	Übertragen von Daten an den Azure IoT Hub	✓
MQTT	Übertragen und Empfangen von Daten	✓
Splunk	Übertragen von Daten an einen Splunk-Server	✓
ControlCenter	Verbindung zum ControlCenter	✓
Control Panel	Überblick des Status und des Netzwerks	✓
System Management	Überblick über Logs und Metriken konfigurierter Module	✓
About Us	Informationen über die Software des RoboGate Edge	✓

1.2.2. Topics

Die Module verwenden Topics, um Daten zu erfassen und die Verbindung zwischen **Dateneingabe (Input)** und **Datenausgabe (Output)** abzubilden (Abbildung 2):

- Input-Module veröffentlichen Daten unter einem spezifischen Topic.
- Output-Module können diese Topics abonnieren, um die Daten zu erhalten.

Das Veröffentlichen bzw. Abonnieren der Topics muss in den jeweiligen Modulen des RoboGate Edge konfiguriert werden.

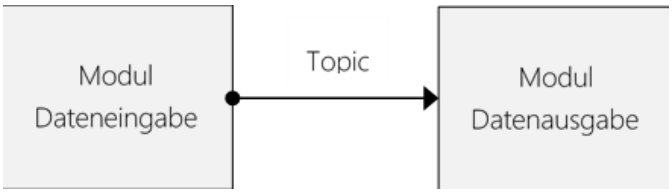


Abbildung 2. Topic-Konzept

▼ **Beispiel: Modbus → Azure IoT Hub**

Um im Azure IoT-Hub Daten vom Modbus-Modul zu abonnieren, muss zunächst das Topic im Modbus-Modul konfiguriert werden. In diesem Beispiel veröffentlicht das **Input-Modul Modbus** Daten unter dem Output Topic *ModbusTCPPollGroup1* (Abbildung 3). Das **Output-Modul Azure IoT Hub** kann nun dieses Topic abonnieren (Abbildung 4). Durch diese Konfiguration erhält das Azure IoT Hub-Modul alle Nachrichten, die unter dem Topic *ModbusTCPPollGroup1* im Modbus-Modul gespeichert wurden.

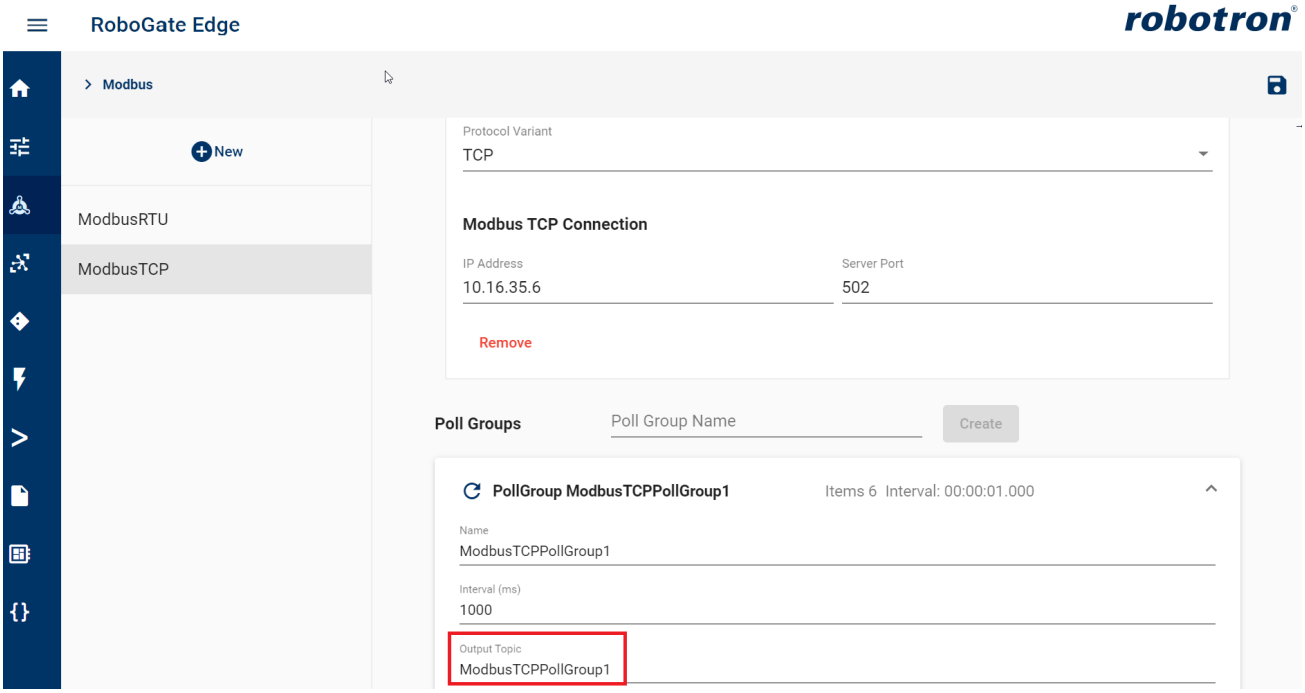


Abbildung 3. Beispielkonfiguration des Modbus Topic

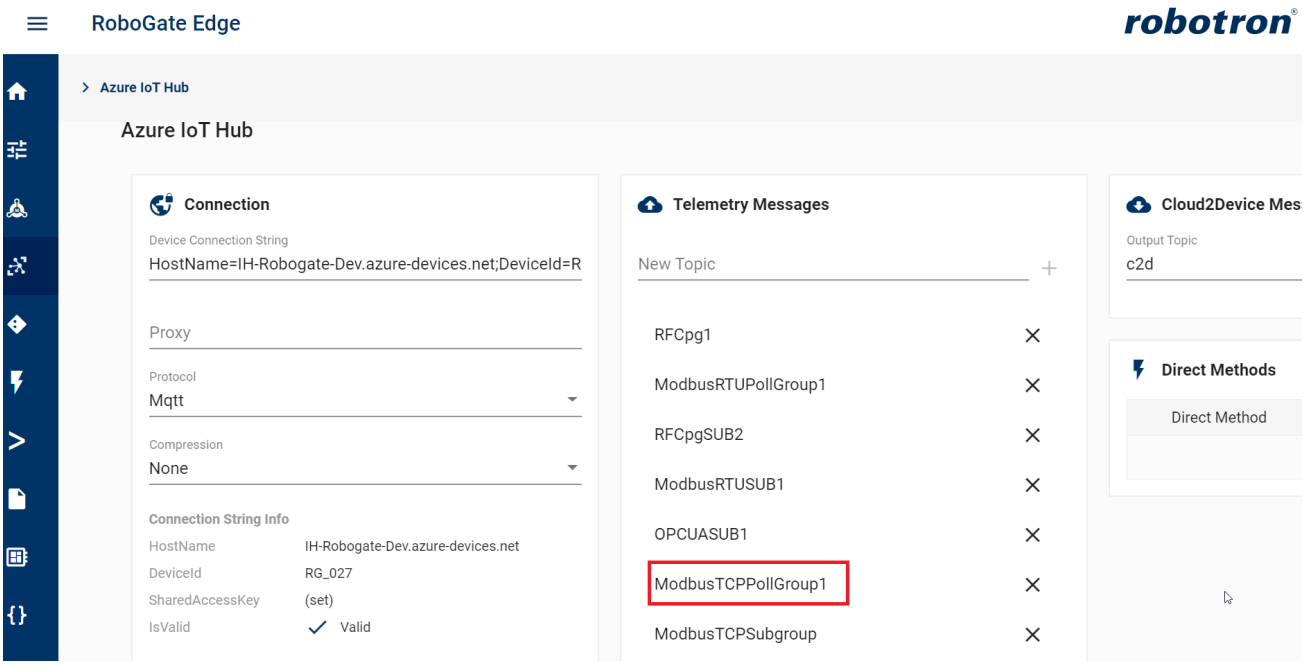


Abbildung 4. Beispielkonfiguration des Azure IoT Hub Topic

1.2.3. Nachrichten

Jede Nachricht beinhaltet Metadaten, welche in Analysesystemen die Zuordnung zur Datenquelle ermöglichen.

- **Source:** Name der Datenquelle
- **Scope:** Name des Auslesebereichs der Datenquelle
- **Source Type:** Typ der Datenquelle

Die Nachrichten-Payloads (nutzbare Nachrichteninhalte) werden als JSON-Datei serialisiert übertragen. Es stehen folgende Payload-Typen zur Verfügung:

- **Model:** Nachricht mit Werten von **mehreren** Datenpunkten (Tabelle 1).
- **Event:** Nachricht mit **einem** Datenpunkt (Tabelle 2).

Tabelle 1. Property und Wert für Model

Property	Wert
timestamp	<UTC ISO Timestamp>
<datenpunktname1>	<datenpunktwert1>
<datenpunktnameN>	<datenpunktwertN>

Tabelle 2. Property und Wert für Event

Property	Wert
timestamp	<UTC ISO Timestamp>
<eventName>	<wert>

1.3. Varianten

Das RoboGate Edge ist eine Software-Anwendung und in zwei Varianten verfügbar:

- auf einem physischen Gerät, dem RoboGate Device
- in einem virtuellen, isolierten Container, dem RoboGate Container

Beide Varianten können alle Module bereitstellen, ausgenommen das Control Panel-Modul, das nur für das RoboGate Edge auf dem RoboGate Device vorgesehen ist.

1.3.1. RoboGate Device

RoboGate Devices sind Hardwaregeräte mit vorinstalliertem RoboGate Edge verfügbar. Robotron bietet ausgewählte Hardwarevarianten von Turck und MOXA, in die das RoboGate Edge voll integriert ist. Für diese Hardwarevarianten ist keine gesonderte Sofwareverwaltung und Systemkonfigurationsverwaltung notwendig.

Alternativ ist auch ein eigenes Hardwaregerät verwendbar. Für diesen Fall steht ein Debian-Paket und ein MSI-Installationspaket zur Verfügung. Es ist jedoch zu beachten, dass die Betriebssysteme von eigenen Geräten nicht verwaltet werden können (z.B. bei einem Software Update).

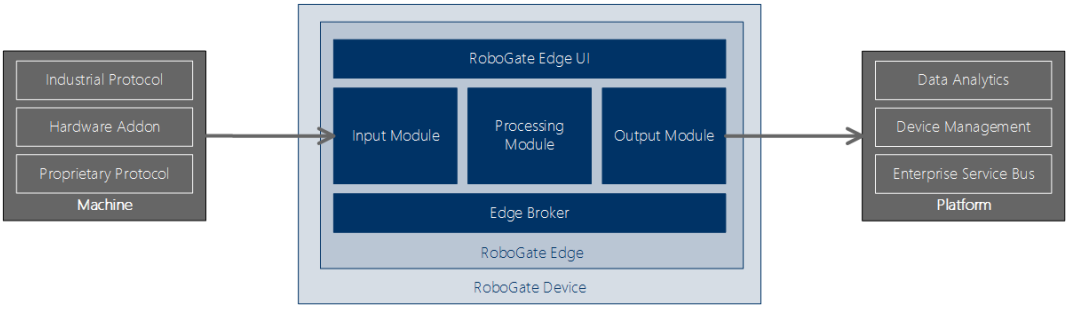


Abbildung 5. RoboGate Edge auf einem RoboGate Device

1.3.2. RoboGate Container

In einigen Anwendungsfällen ist es nicht zweckmäßig, gesonderte Hardware zu verbauen. Für diese Fälle lassen sich auch isolierte und virtuelle Container-Instanzen des RoboGate Edge auf einer zentralen Hardware starten. Voraussetzung dafür ist eine OCI-basierte Containerplattform wie Docker oder Podman.

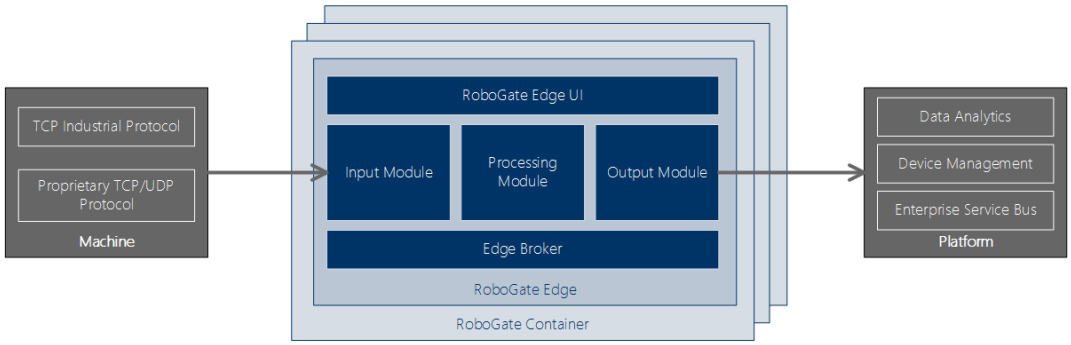


Abbildung 6. RoboGate Edge als Container-Instanz

2. Inbetriebnahme

2.1. RoboGate Devices

2.1.1. Technische Daten

> Siehe Technische Dokumentation „RoboGate Hardware Quickstart“

2.1.2. Herstellung der Netzwerkverbindung & Erstinbetriebnahme

Schnittstellen Ethernet0 und Enternet1

1. Schließen Sie das Netzwerkkabel am RoboGate Device an Ethernet0 oder Ethernet1 an und verbinden Sie den Router/Switch.
2. Starten Sie Ihren Browser und geben Sie folgende Adresse ein:
   a. ohne automatisches DNS: http:///[IP] des RG (auslesen im Router oder beim Administrator erfragen)
   b. mit automatischem DNS: http:///[RG-xxxxxxxxxxxxxxxx] (x-Seriennummer vom EdgeDevice, seitlicher Aufdruck)
3. Geben Sie Benutzername und Passwort in die Maske ein. Standardwerte bei erster Anmeldung bei Bedarf ändern.

Standard-Login:

- Nutzername: rgadmin
- Kennwort: $rgpasswd123

Jetzt steht Ihnen die RoboGate Edge UI als Web-Anwendung zur weiteren Konfiguration zur Verfügung.

2.2. RoboGate Container

Das RoboGate Edge basiert auf dem .NET core Framework, wird virtualisiert in einer Containerumgebung (z.B. Docker) betrieben und als Image bereitgestellt. Die Konfiguration des Software RoboGates kann in der RoboGate Edge UI durchgeführt werden.

2.2.1. Voraussetzung

Die Zielplattform (Host) muss zum Ausführen des RoboGate Edge folgende softwaretechnische Voraussetzungen erfüllen:

- Docker (oder eine andere containerisierte Umgebung): zum Ausführen Docker-basierter Container
- (optional) Docker Compose: zum einfachen Orchestrieren mehrerer Docker-basierter Container

Weiterführende Informationen, welche den Einstieg in Docker [1] bzw. Docker Compose [2] erleichtern, sind der Website der Docker Inc. zu entnehmen.

Hinweis: Die Installation ist abhängig vom Betriebssystem der Zielumgebung (Host) sowie von weiteren Einsatzbedingungen, wie Desktop, Server oder Cluster.

Die Container Images sind, je nach Bedarf, kompatibel mit Linux Betriebssystemen und Intel bzw. ARM Prozessorarchitekturen. Die Mindestanforderungen an die Hardware sind abhängig von:

- der Anzahl der Container
- der Menge und Frequenz der auszulesenden Datenpunkte.

Je mehr Container gestartet werden müssen, desto höher ist die Last des Systems während der Startphase. Je mehr Datenpunkte je Zeiteinheit gelesen werden, desto höher ist die Systemlast zur Laufzeit.

Beispiel für ca. 10 Instanzen:

- OS: Ubuntu Server 18.04
- CPU: 2x 1Ghz
- RAM: 4 GB

2.2.2. Installation des Images

Die RoboGate Edge Software wird als Image bereitgestellt:

1. Es wird eine Datei bereitgestellt: `docker load -f <dateiname>`
2. Es wird eine URL bereitgestellt: `docker pull <imagename>`
    - Falls Credentials mitgeliefert werden: `docker login <registry_url> -u <username> -p <password>`

Auf Basis des Images kann je nach Bedarf mit dem jeweiligen CLI `docker` oder `docker-compose` ein oder mehrere Container gestartet werden. Bei der Konfiguration der Container muss folgendes beachtet werden:

- TTY aktiviert
- Stdin aktiviert
- Portmapping: Um auf die RoboGate Edge UI bzw. API zuzugreifen muss ein Portmapping angelegt werden. Alternativ können die Container auch mit dem Host Network verbunden werden.
- Ports müssen für den Zugriff auf lokale Ports gemappt werden
- Konfigurationsverzeichnis (relevant für volume mount): /app/config
- Umgebungsvariablen:
    - HTTP_PORT: HTTP Port der RoboGate Edge API (Achtung nur innerhalb des Containers!) Default: 80

- HTTPS_PORT: HTTPS Port der RoboGate Edge API (Achtung nur innerhalb des Containers!) Default: RoboGate Edge

Beispiel:

```
docker run -dit -p "80:80" -p "443:443" <imagename>
```

Die RoboGate Edge UI ist nun unter https://localhost verfügbar.

3. Input-Module

3.1. Modbus

Das Modbus-Modul ermöglicht das Schreiben und Auslesen von Slave Modbus-Modulen.

Das Modul agiert als Master und stellt dafür den Zugriff auf Modbus Slaves in den Protokollversionen TCP und RTU für lesenden und schreibenden Zugriff bereit. Für die ausgelesenen bzw. geschriebenen Daten stellt das Modul für verschiedene Datentypen automatische Konvertierungsfunktionen bereit.

Es können mehrere auszulesende Server, sowie mehrere Slaves konfiguriert werden, die in dem konfigurierbaren Intervall mit verschiedenen Optionen (Subscription/Poll Group) abgefragt werden.

Folgende Modbus-*Lese*-Funktionen werden unterstützt:

- *Read Coils* (Function Code 01)
- *Read Inputs* (Function Code 02)
- *Read Holding Registers* (Function Code 03)
- *Read Input Registers* (Function Code 04)

Folgende Modbus-*Schreib*-Funktion wird unterstützt:

- *Write Holding-Registers* (Function Code 06)

Es stehen zwei Übertragungsprotokolle zur Verfügung:

- TCP
- RTU

3.1.1. Konfigurationsparameter

Server

Die Server-Konfiguration für das Modbus-Modul kann in Abhängigkeit des schon bestehenden Systems angepasst oder die Standardkonfiguration beibehalten werden. Die nachfolgenden zwei Tabellen zeigen alle Konfigurationsparameter für die TCP- und RTU-Verbindung.

Tabelle 3. Modbus TCP Server-Konfiguration

Parameter	Beschreibung
Server Name	Beschreibt den Namen des auszulesenden Modbus-Masters. Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar.
Protocol Variant	Auswahl des Protokolls TCP

Parameter	Beschreibung
IP Adress	Eingabe der IP-Adresse des Modbus TCP-Masters mit Format xxx.xxx.xxx.xxx
Server Port	Der Standard Modbus TCP-Port ist „502“. Der Port kann entsprechend der Serverkonfiguration angepasst werden.

Tabelle 4. Modbus RTU Server-Konfiguration

Parameter	Beschreibung
Server Name	Beschreibt den Namen des auszulesenden Modbus-Masters. Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar.
Protocol Variant	Auswahl des Protokolls RTU
Port Name	Name der Datenquelle im Format /*/* Bei RoboGate Devices powered by Turck ist dies standardmäßig /dev/ttyO1.
Baudrate	Serielle Schnittstelle: Datenrate (Vorgabe durch Modbus-Slave)
Data Bits	Serielle Schnittstelle: Anzahl Datenbits (Vorgabe durch Modbus-Slave)
Stop Bits	Serielle Schnittstelle: Anzahl Stoppbits (Vorgabe durch Modbus-Slave)
Parity	Serielle Schnittstelle: Paritätseinstellung (Vorgabe durch Modbus-Slave)

Slaves

Innerhalb der Verbindungsvarianten der Server ist es möglich, auf mehrere Slaves zuzugreifen. Besonders bei Modbus RTU Servern können damit mehrere Slaves mit gleichen Verbindungsparametern angesprochen werden. Die Unterscheidung der verschiedenen Slaves findet über die Slave ID statt. Sie sollte daher innerhalb einer Gruppe für jeden Slave einmalig sein. Jede Slave ID-Gruppe kann eigene Poll, Subscription oder Setup Groups enthalten.

Parameter	Beschreibung
Slave Name	Name des Modbus Slave
Slave ID	Die ID (0 - 255) des auszulesenden Modbus Slave.

Setup Groups

Mit der Konfiguration von Setups können Modbus Register geschrieben werden. Die Konfiguration der Setups kann im Modbus TCP und im Modbus RTU durchgeführt werden. Beim Start oder bei Konfigurationsänderung werden die angegebenen Werte einmalig in die Datenregister geschrieben. Solange nicht alle Werte geschrieben sind, werden alle anderen Gruppen innerhalb eines Slaves angehalten. Der Vorgang wird nur einmal ausgeführt.

Tabelle 5. Setup-Konfiguration

Parameter	Beschreibung
Setup Name	Name des Setups
Parameter Name	Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar.
Write Address	Adresse des schreibenden Modbus Registers
Type	Standarddatentypen (UInt16, UInt32, UInt64, Int16, Int32, Int64) sind vorkonfiguriert. Die Auswahl erfolgt per Dropdownmenü.
Value	Zu schreibender Wert.
Little Endian	Speicherstruktur Umschaltung (wenn angehakt= true wird Little Endian verwendet, ansonsten BigEndian)

Poll Groups

Jede Poll Group erzeugt eine Nachricht (Payload-Typ Model). Das Model entspricht einer Nachricht mit Werten mehrerer Datenpunkte.

Tabelle 6. Konfiguration von Poll Groups im Modbus Modul

Parameter	Beschreibung
Poll Group Name (Scope)	Name der Pollgroup, bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar. Dieser Name entspricht den Namen des auszulesenden Bereichs. Der Name wird unter Scope in Auswertungen bereitgestellt.
Source	Folgende Source Modi stehen zur Verfügung: • "UseServerName" • "UseSlaveName" • "UseBoth Die Auswahl des Modus gibt an, welcher Name in den SourceHeader geschrieben wird. Bei "UseBoth" wird die Kombination ServerName.SlaveName im SourceHeader verwendet.
Interval (ms)	Intervall in Millisekunden gibt die Taktung an, in der die Werte abgefragt werden.
Output Topic	Name des Output Topic. Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar.
Field Name	Bezeichnung des zu lesenden Wertes. Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar.

Parameter	Beschreibung
Function Code	Folgende Function Codes stehen zur Verfügung: • 01 Read Coils (automatisch Boolean Datentyp) • 02 Read Inputs (automatisch Boolean Datentyp) • 03 Read Holding Registers • 04 Read Input Registers
Start Register	Registeradresse
Type	Standarddatentypen sind vorkonfiguriert. Die Auswahl erfolgt per Dropdownmenü.
Length	Anzahl auszulesender 16bit Register
Endian Mode	Speicherstruktur-Umschaltung. Dabei stehen folgende Optionen zur Verfügung: * bigEndian * littleEndian * bigEndianByteSwapped * littleEndianByteSwapped
Value Adjust Gain/Offset	Wert-Nachverarbeitung: Konstante Skalierung durch Faktor und Summand (Offset)
Test	Aktuellen Wert abrufen.

> Die Registerlänge und der Datentyp müssen konsistent sein. Für die Standard-Datentypen ergibt sich die Länge aufgrund der Vorkonfiguration automatisch. Ausschließlich der Datentyp Raw Hex ist konfigurierbar.

Subscription Groups

Mit der Subscription Group wird in einem eingestellten *Publishing Interval* der Slave Node gepollt. Je nach Konfiguration werden nur Datenpakete generiert, wenn es eine Änderung des Zeitstempels gab.

> Es handelt sich in diesem Fall nicht um eine Subscription im Sinne der OPC UA-Spezifikation. Der Slave wird stattdessen fortlaufend mit den Lastauswirkungen auf Slave-, Master- und Netzwerk-Seite gepollt!

Tabelle 7. Konfiguration von Subscription Groups

Parameter	Beschreibung
Subscription Group Name (Scope)	Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar. Der Name wird unter Scope in den Auswertungen bereitgestellt. Mit dem Klick auf „Create“ wird eine neue Gruppe erzeugt.

Parameter	Beschreibung
Source	Folgende Source Modi stehen zur Verfügung: • "UseServerName" • "UseSlaveName" • "UseBoth Die Auswahl des Modus gibt an, welcher Name in den Source Header geschrieben wird. Bei "UseBoth" wird die Kombination ServerName.SlaveName im Source Header verwendet.
Publishing Interval (ms)	Intervall in Millisekunden gibt die Taktung an, in der die Werte abgefragt werden.
Republishing Interval (ms)	Intervall in Millisekunden gibt die Taktung an, in der die zuletzt gelesenen Werte erneut gesendet werden.
On Change: Submit Full Model	**Aktiv**: Falls der Modbus Server eine Änderung mittteilt wird der geänderte Datenpunkt gemeinsam mit den zuletzt gelesenen Werten der anderen Datenpunkte als Model-Nachricht übermittelt. ACHTUNG: Gleichzeitige Änderungen mehrerer Datenpunkte innerhalb der Subscription Group werden individuell behandelt **Inaktiv**: Falls der Modbus Server eine Änderung mitteilt wird nur der geänderte Datenpunkt als Event-Nachricht übermittelt.
On Republish: Submit Full Model	**Aktiv**: Während des Republishings, werden die Datenpunkte als eine Data-modelnachricht versandt **Inaktiv**: Während des Republishings werden die Datenpunkte jeweils als individuelle Nachricht (Event) versandt
Output Topic	Name des Output Topic. Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar.
Field Name	Bezeichnung des zu lesenden Wertes.
Function Code	Folgende Function Codes stehen zur Verfügung: • 01 Read Coils (automatisch Boolean Datentyp) • 02 Read Inputs (automatisch Boolean Datentyp) • 03 Read Holding Registers • 04 Read Input Registers
Start Register	Registeradresse
Type	Standarddatentypen sind vorkonfiguriert. Die Auswahl erfolgt per Dropdownmenü.
Length	Anzahl auszulesender 16bit Register

Parameter	Beschreibung
Endian Mode	Speicherstruktur-Umschaltung. Dabei stehen folgende Optionen zur Verfügung: * bigEndian * littleEndian * bigEndianByteSwapped * littleEndianByteSwapped
Value Adjust Gain/Offset	Wert-Nachverarbeitung: Konstante Skalierung durch Faktor und Summand (Offset)
Test	Aktuellen Wert abrufen.

3.1.2. Konfiguration in der UI

Nach Auswahl des Moduls Modbus auf der Startseite oder über die linke Menüleiste der RoboGate Edge UI gelangen Sie zur Konfiguration des Modbus-Moduls. Die nachfolgenden Ausführungen erfolgen am Beispiel eines Modbus TCP Servers.

Server konfigurieren

Mit einem Klick auf New in der linken Menüleiste kann ein neuer Modbus Server angelegt und konfiguriert werden (Abbildung 7). Im ersten Schritt ist für den neuen Modbus Server eine Konfiguration Der Server Info und Connection durchzuführen. Dies umfasst auch das Anlegen von Modbus Slaves.

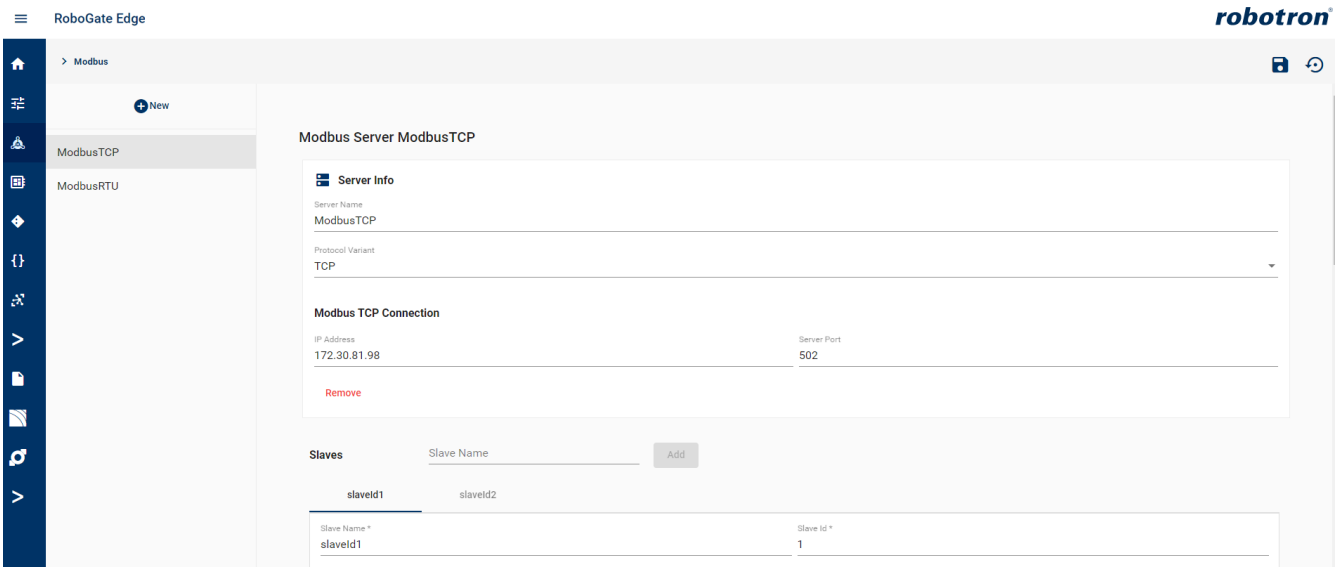


Abbildung 7. Hinzufügen eines neuen Modbus Servers

Wird diese Konfiguration gespeichert , lässt sich im Bereich *Module State* am Ende der Seite erkennen, ob das Modul eine Verbindung zum Modbus Server herstellen konnte.

Nun lassen sich für jeden Slave separat eine oder mehrere Setup, Poll oder Subscription Groups erstellen.

Group konfigurieren

Nach Eingabe eines Namens für die entsprechende Group wird mit einem Klick auf „Create“ eine

neue Gruppe erzeugt. Anschließend lassen sich die Groups mit ihren Konfigurationsparametern konfigurieren. Die nachfolgenden Abbildungen zeigen beispielhafte Konfigurationen für eine Setup, Poll und Subscription Group für den angelegten Modbus TCP Server und Slave.

Um zu prüfen, ob ein Datenblock korrekt konfiguriert wurde, kann dieser über den Pfeil in der Spalte "Test" geprüft werden. Das ausgelesene Ergebnis wird nach einem Klick auf den Pfeil angezeigt bzw. bei Änderungen aktualisiert. Initial werden keine Werte angezeigt.

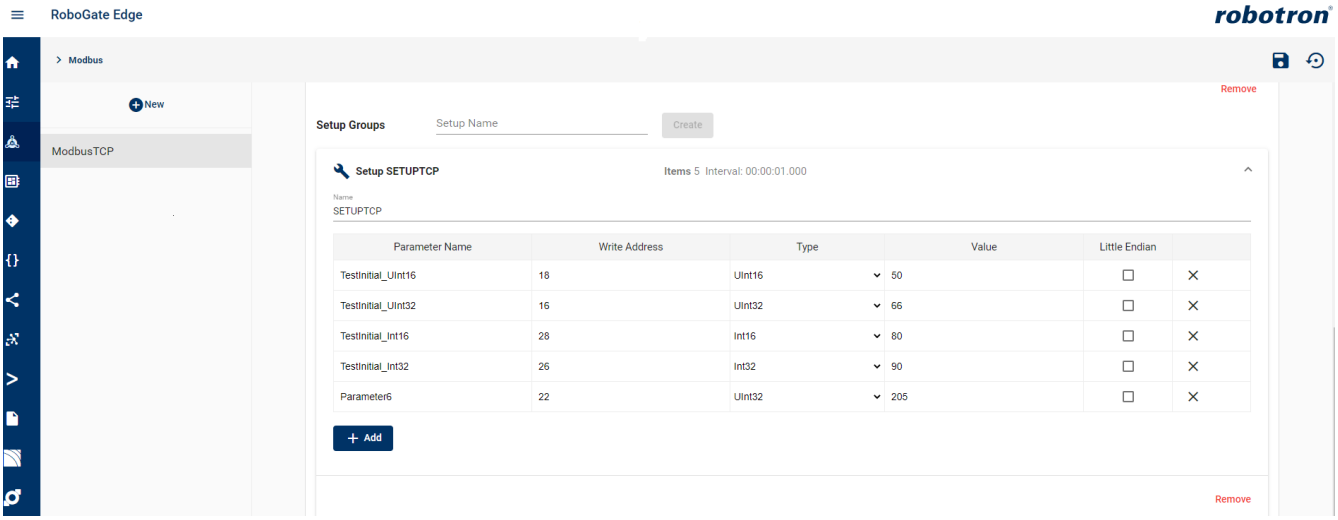


Abbildung 8. Konfiguration der Setup Group

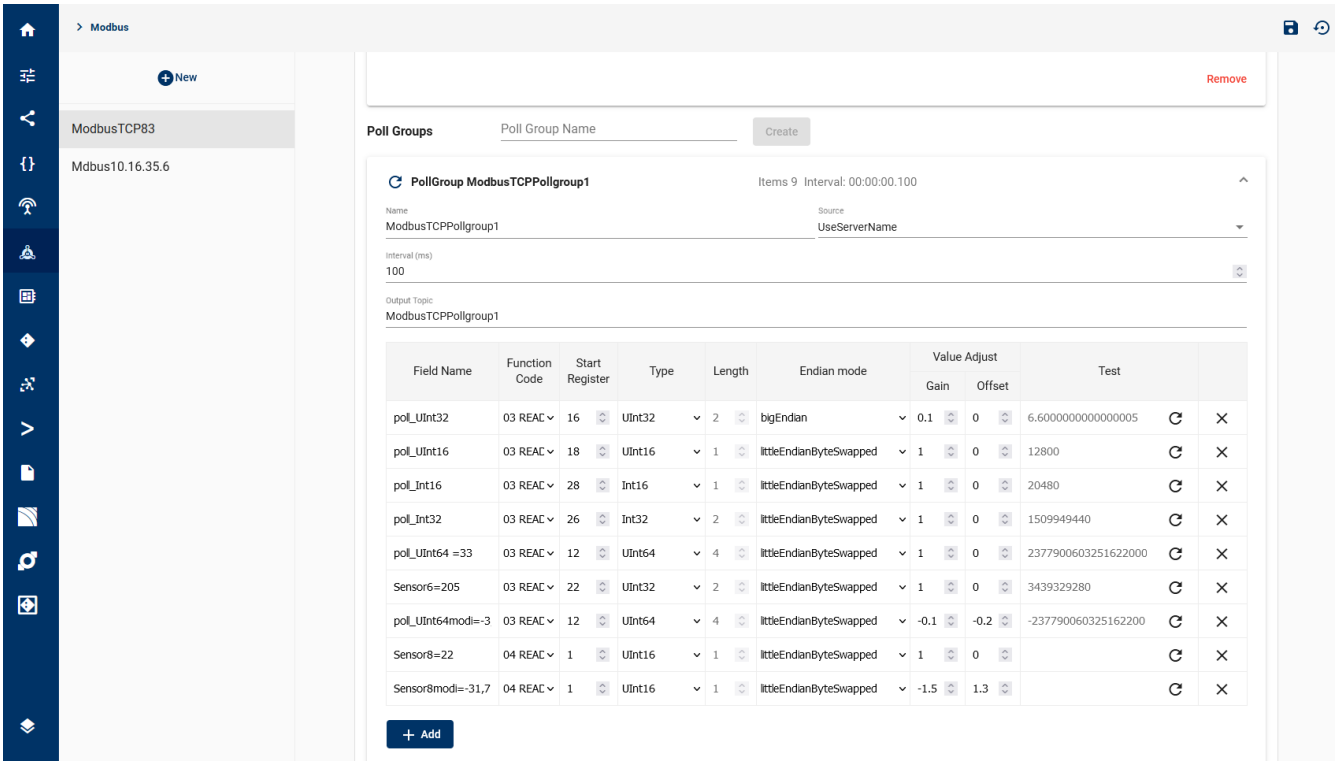


Abbildung 9. Konfiguration der Poll Group

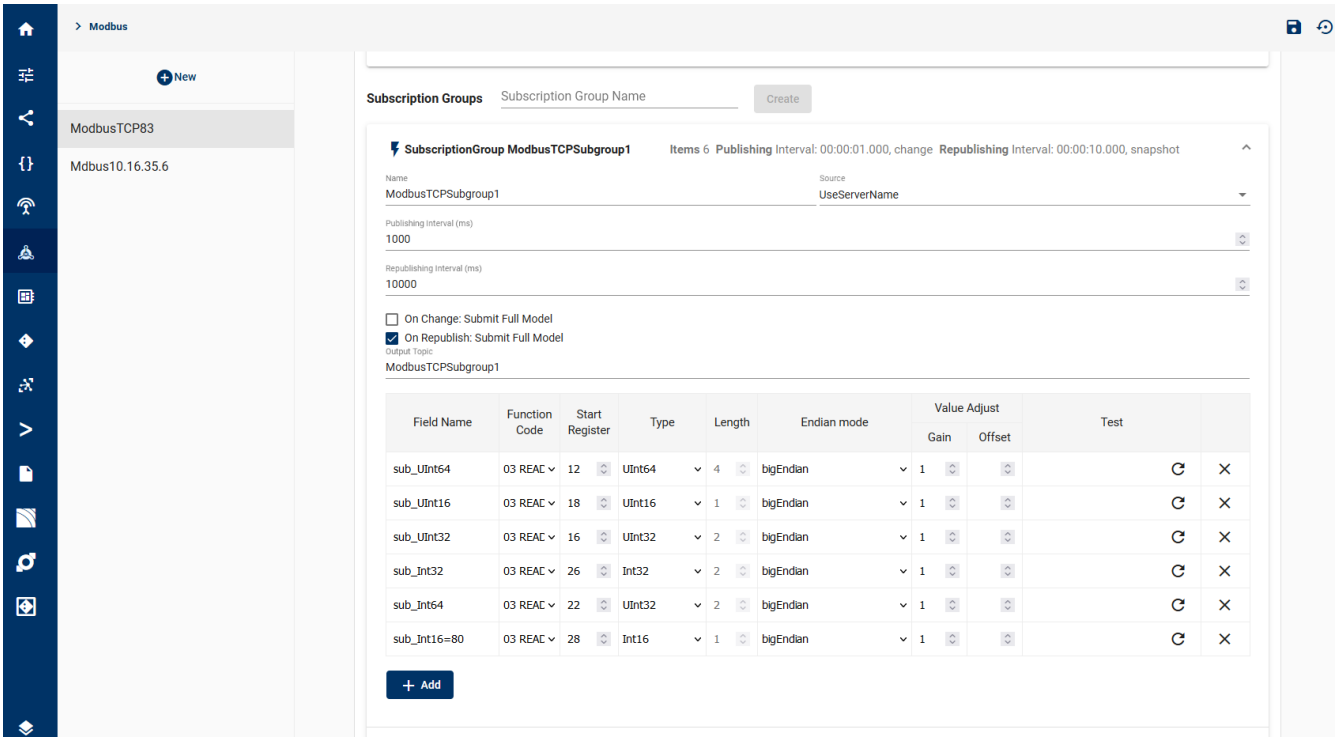


Abbildung 10. Konfiguration der Subscription Group

Nach Fertigstellung der Konfiguration des Modbus Servers kann diese über die Schaltfläche „Save Configuration“ in der horizontalen Menüleiste oben rechts im Modul gespeichert werden ([img-modbus-saveConfig]). Durch das Speichern der gesamten Konfiguration des Modbus-Moduls steht diese im RoboGate Edge zur Verfügung.

Datenfeld löschen

Innerhalb der Konfiguration des Modbu Servers können Datenfelder in den Gruppenkonfigurationen gelöscht werden. Das zu löschende Datenfeld wird über die Entfernen-Schaltfläche gelöscht. Abschließend muss die neue Konfiguration des Modbus Servers über die Schaltfläche „Save Configuration“ bestätigt werden. Die geänderte Konfiguration steht nun im RoboGate Edge zur Verfügung.

Group löschen

Innerhalb der Konfiguration des Modbus Servers können gesamte Gruppen aus der Konfiguration gelöscht werden. Die zu löschende Group wird über die Schaltfläche „Remove“ gelöscht (Abbildung 11). Abschließend muss die neue Konfiguration des Modbus Servers über die Schaltfläche „Save Configuration“ bestätigt werden. Die geänderte Konfiguration steht nun im RoboGate Edge zur Verfügung.

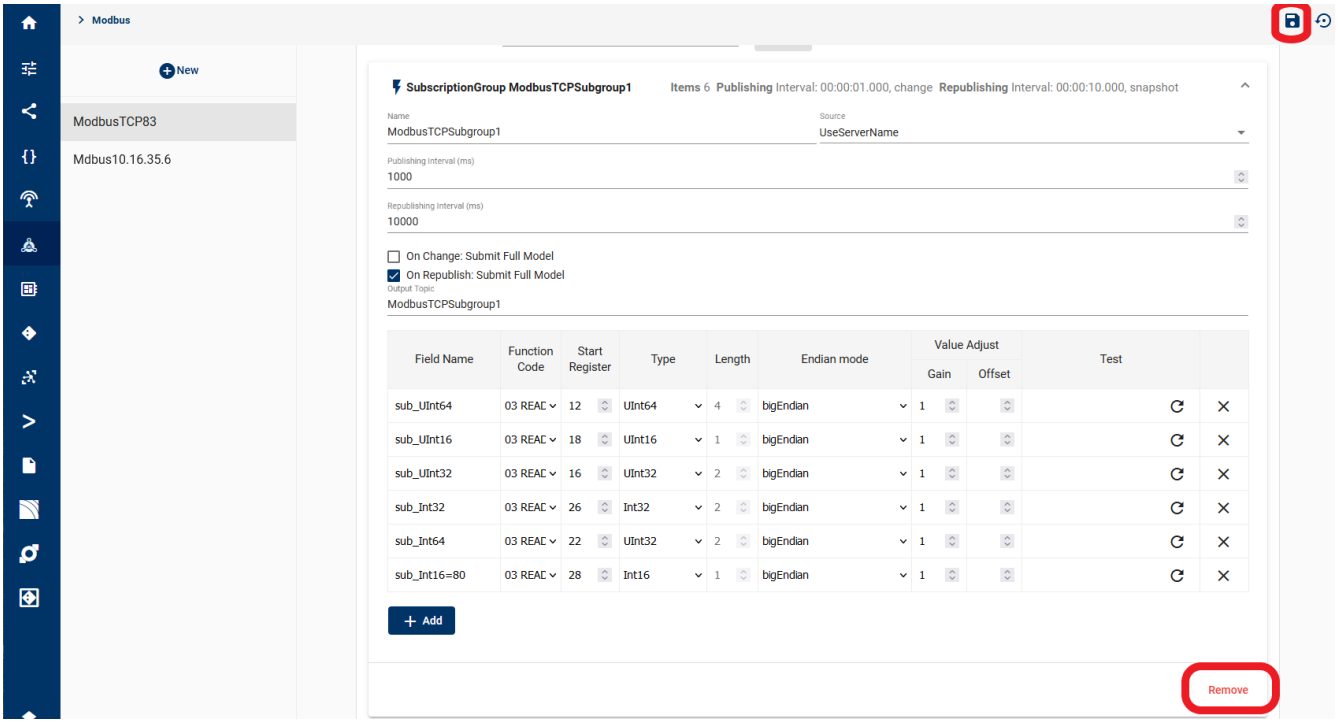


Abbildung 11. Löschen einer Poll Group

Server löschen

Innerhalb der Modbus Server Konfiguration kann ein Modbus Server gelöscht werden. Durch Betätigen der Schaltfläche „Remove“ im Abschnitt zur Server-Konfiguration wird der gewählte Server gelöscht (Abbildung 12). Abschließendes Speichern der Modbus-Konfiguration bestätigt das Löschen.

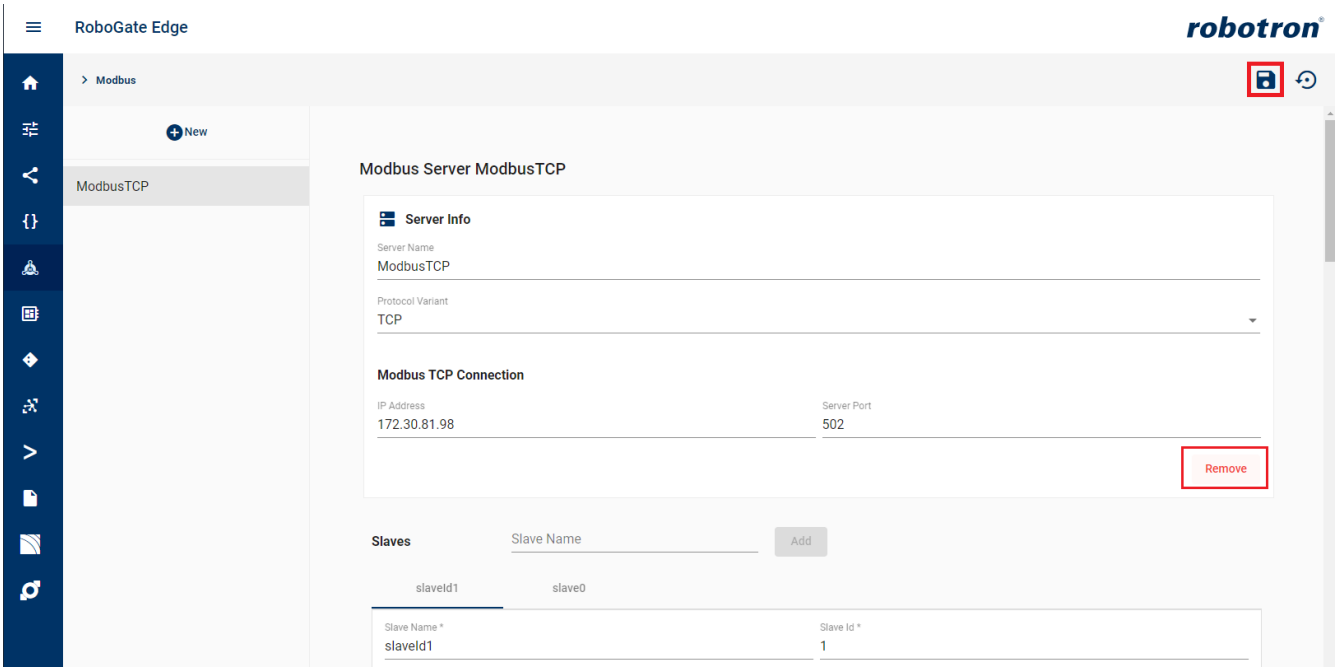


Abbildung 12. Löschen eines Modbus Servers

Konfiguration zurücksetzen

Über die Schaltfläche in der horizontalen Menüleiste lässt sich die komplette Konfiguration des Moduls auf Werkseinstellungen problemlos zurücksetzen. Nach einem Klick auf die Schaltfläche

folgt ein Bestätigungsdialog mit "Yes" und "No" zur Rückfrage ob der Vorgang wirklich ausgeführt werden soll.

> Damit wird auch die gesamte Historie auf dem RoboGate Edge gelöscht!

3.2. OPC UA

Das OPC UA-Modul dient dem Auslesen eines OPC UA Servers.

Nodes können über zwei Varianten ausgelesen werden. Sie können abonniert werden, um sich ändernde Werte auszulesen (*Subscription*) oder sie können zyklisch abgefragt werden (*Poll*). Um das Nachrichtenaufkommen zu minimieren, werden Nodes mit der gleichen Abtastrate in einer Gruppe (*Group*) zusammengefasst. Die Nodes dieser Gruppe werden gleichzeitig abonniert bzw. abgefragt und eine Nachricht als Ergebnis der Abfrage versendet.

3.2.1. Konfigurationsparameter

Server Connection

Die in folgender Tabelle beschriebenen Parameter müssen für den OPC UA Server definiert werden.

Für den OPC UA Server kann eine Security-Konfiguration angelegt und über einen Switch ein- oder ausgeschaltet werden. In Abbildung 15 ist das Ein- bzw. Ausschalten der Security sichtbar.

Tabelle 8. Konfiguration der OPC UA Server Connection

Parameter	Beschreibung
Server Name (Source)	Der Server Name stellt den Namen der Datenquelle dar. Dieser entspricht der Source in Auswertungen.
URL	Verbindung (Connection) zum OPC UA Server über die URL des Servers
Node Blacklist Auto Reset Interval (ms)	Ermöglicht eine automatisch zyklische Bereinigung der Blacklist des ausgewählten Servers. Das Default Interval beträgt 15 Minuten (900.000 Millisekunden).
Security Level	0 – 255
Security Policy	Auswahloptionen: • None • Basic128Rsa15 • Basic256 • Basic256Sha256

Parameter	Beschreibung
Security Mode	Auswahloptionen: • None (kein Security Mode) • Sign (Signieren) • Sign and Encrypt (Signieren und Verschlüsseln)

Weitere Konfigurationsparameter:

- **OptimizeNodeValueReadAccess** - konfigurierbare Abschaltung der Leseoptimierung/des Pollings. Wenn diese Option aktiviert wird, werden mehrere Nodewerte in einer einzigen Anforderung gelesen. Andernfalls wird jeder Node separat über jeweils eine Anfrage angefordert.
- **Use Source Timestamp** - Wenn Use Source Timestamp global aktiviert ist, wird dies bei allen Subscription, Subscription Groups (auch für Submit Full Model) und Poll Groups benutzt. Wenn mehrere Werte in der Nachricht enthalten sind (Poll Group oder Subscription Group mit Submit Full Model) wird der Zeitstempel vom zuletzt ausgelesenen Node als allgemeiner Zeitstempel genutzt.

Authentication

Es wird zwischen drei Authentication-Typen unterschieden:

- Anonymous
- Username and Password
- Certificate

Für Certificate stehen folgende weitere Felder zur Verfügung:

Parameter	Beschreibung
Certificate Subject	Der Aussteller des Zertifikats. Oftmals auch als CN bezeichnet.
Certificate Directory	Der relative Pfad innerhalb des Robogate-Ordners (C:\Program Files\Robotron\Robogate\). Hier muss unter config\certs ein weiterer Ordner erstellt werden, z.B. User, indem sich dann der Ordner "certs" sowie "private" befindet. In den Ordner "certs" werden dann die Zertifikate (.der Datei) abgelegt, im Ordner "private" jeweils die Schlüssel (.pem Datei).

Abbildung 13. Beispiel Konfiguration für Benutzung eines Zertifikats

Subscriptions

Das Auslesen des OPC UA Servers kann über *Subscriptions* erfolgen. Jede Subscription erzeugt eine Nachricht (Event) je Wertänderung der hinterlegten Nodes. Das Event entspricht einer Nachricht mit einem Datenpunkt. Es sind die in folgender Tabelle beschriebenen Parameter zu konfigurieren.

Tabelle 9. Konfiguration der Subscription des OPC UA-Moduls

Parameter	Beschreibung
Field Name	Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar. Dieser gibt die Bezeichnung des zu lesenden Wertes an.
Node ID	Die Node ID wird in Abhängigkeit vom Field Name automatisch vergeben. Diese ID kann manuell kopiert werden. Es besteht die Möglichkeit, Nodes des Servers mit dem OPC UA Node Browser zu durchsuchen und auszuwählen.
Output Topic	Die Module zur Dateneingabe veröffentlichen die Daten in einem Topic. Der Name des Topics ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar.

Method Mapping

Über RoboGate Edge können auch OPC UA-Methoden aufgerufen werden. Solche Methoden sind in der Regel Auslöser für eine Aktion, die eine Maschine ausführen soll, beispielsweise einen Prozess zu starten und Parameter zu setzen.

Um mittels des RoboGate Edge eine OPC UA-Methode aufzurufen, muss diese Methode zuvor auf dem OPC UA-Server angelegt worden sein. Weiterhin muss die Object Node ID sowie die ID der Methode bekannt sein. Alternativ kann man auch den Browsing Pfad angeben. Mit diesen Angaben (Tabelle 10) kann das Method Mapping vorgenommen werden. Dies meint das Mapping der OPC UA-Methoden auf die Edge Methods, welche beim Edge Broker des RoboGate Edge registriert werden.

Tabelle 10. Konfiguration des Method Mapping

Parameter	Beschreibung
Method Name	Name der Methode (frei wählbar)
Object Node ID	Node ID des der OPC UA-Methode übergeordneten Objekts
Method Node ID	Node ID der OPC UA-Methode

Wurde eine Methode für das Method Mapping angelegt, ist diese nun im Stream Processor via Edge Method Invocation und im Azure-Modul des RoboGate Edge aufrufbar. Dort muss die Methode wie folgt angegeben werden: `robogate.opcua.<name>`. <name> beschreibt dabei den frei vergebenen Namen der Methode im Method Mapping.

Im Azure-Modul kann die Methode im Bereich "Direct Methods" aufgerufen werden. Hierzu ist ebenfalls ein Name für die Methode (Direct Method) zu definieren und die entsprechend zu referenzierende Methode (robogate.opcua.<name>) als Edge Method anzulegen.

Method Error Code Mapping

Um individuelle konfigurierte Status Codes eines OPC UA Servers in sprechende Fehlernachrichten zu wandeln gibt es das Feature Error Code Mapping.

Tabelle 11. Konfiguration des Error Mapping

Parameter	Beschreibung
Error Code	Der Code ist als ganze Dezimalzahl anzugeben ODER mit dem Präfix "0x" als hexadezimale Repräsentation.
Message	Frei wählbarer Text zur Erläuterung des Fehlers

Poll Groups

Poll Groups sind aktive Abrufe. Jede Poll Group erzeugt eine Nachricht (Model). Das Model entspricht einer Nachricht mit Werten mehrerer Datenpunkte.

Tabelle 12. Konfiguration von Poll Groups im OPC UA Modul

Parameter	Beschreibung
Poll Group Name (Scope)	Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar. Es muss ein Name für die Poll Group eingegeben werden. Dieser Name entspricht den Namen des auszulesenden Bereichs. Der Name wird unter Scope in Auswertungen bereitgestellt. Mit dem Klick auf „Create“ wird eine neue Gruppe erzeugt.
Interval (ms)	Intervall in Millisekunden gibt die Taktung an, in der die Werte abgefragt werden.

Parameter	Beschreibung
Use source timestamp	Benutzung des SourceTimestamps: • On: Der Zeitstempel vom OPCUA Server wird verwendet. • Off: Der Zeitstempel des RoboGates wird verwendet. • Default: Der global definierte Zustand wird verwendet.
Output Topic	An dieser Stelle wird ein vergebenes Output Topic z.B. OPCUAPOLL1 angegeben. Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar.
Field Name	Das ist die Bezeichnung des Feldes im OPC UA Server.
Node ID	Die Node ID wird in Abhängigkeit vom Field Name automatisch vergeben. Diese ID kann manuell kopiert werden. Es besteht die Möglichkeit, Nodes des Servers mit dem OPC UA Node Browser zu durchsuchen und auszuwählen.

Abbildung 14. Individuelle SourceTimestamps

Subscription Groups

Subscription Groups definieren eine Menge von OPC UA Server Nodes, die abonniert werden. Der Zustand jedes Nodes wird in einem Puffer gehalten. Kommt es zu einer Datenwertaktualisierung, wird der entsprechende Wert aktualisiert und anschließend eine Nachricht mit Inhalt aller konfigurierten Nodes gesendet. Jede Subscription Group erzeugt eine Nachricht.

Tabelle 13. Konfiguration von Subscription Groups im OPC UA

Parameter	Beschreibung
Subscription Group Name (Scope)	Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar. Es muss ein Name für die Subscription Group eingegeben werden. Dieser Name entspricht den Namen des auszulesenden Bereichs. Der Name wird unter Scope in Auswertungen bereitgestellt. Mit dem Klick auf „Create“ wird eine neue Gruppe erzeugt.
Publishing Interval (ms)	Intervall in Millisekunden gibt die Taktung an, in der die Werte abgefragt werden.
Republishing Interval (ms)	Intervall in Millisekunden gibt die Taktung an, in der die zuletzt gelesenen Werte erneut gesendet werden.

Parameter	Beschreibung
On Change: Submit Full Model	**Aktiv**: Falls der OPC UA Server eine Änderung mitteilt wird der geänderte Datenpunkt gemeinsam mit den zuletzt gelesenen Werten der anderen Datenpunkte als Model-Nachricht übermittelt. ACHTUNG: Gleichzeitige Änderungen mehrerer Datenpunkte innerhalb der Subscription Group werden individuell behandelt **Inaktiv**: Falls der OPC UA Server eine Änderung mitteilt, wird nur der geänderte Datenpunkt als Event-Nachricht übermittelt.
On Republish: Submit Full Model	**Aktiv**: Während des Republishing werden die Datenpunkte als eine Datamodelnachricht versandt. **Inaktiv**: Während des Republishing werden die Datenpunkte jeweils als individuelle Nachricht (Event) versandt.
Output Topic	An dieser Stelle wird ein vergebenes Output Topic z.B. OPCUASUB1 angegeben. Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar.
Use source timestamp	Benutzung des SourceTimestamps: • On: Der Zeitstempel vom OPCUA Server wird verwendet. • Off: Der Zeitstempel des RoboGates wird verwendet. • Default: Der global definierte Zustand wird verwendet.
Field Name	Bezeichnung des Feldes im OPC UA Server.
NodeID	Die Node-ID wird in Abhängigkeit vom Field Name automatisch vergeben. Diese ID kann manuell kopiert werden. Es besteht die Möglichkeit, Nodes des Servers mit dem OPC UA Node Browser zu durchsuchen und auszuwählen.

Alarms & Conditions

Mit dem OPC UA-Modul können auch Alarme und Konditionen, die im OPC UA Server als Nodes angelegt wurden, aufgerufen werden. Es können pro OPC UA Server mehrere sogenannte Condition Observer konfiguriert werden. Jeder konfigurierte Observer erzeugt dabei auf dem angegeben output Topic eine EdgeMessage sobald ein Event vom OPC UAServer ankommt.

Siehe auch Alarms&Conditions und Conditions in der *OPC UA Online Reference*.

Tabelle 14. Konfiguration von Condition Observern

Parameter	Beschreibung
Condition Observer Name (Scope)	Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar. Es muss ein Name eingegeben werden. Dieser Name entspricht den Namen des auszulesenden Bereichs. Der Name wird unter Scope in Auswertungen bereitgestellt. Mit dem Klick auf „Create“ wird eine neue Gruppe erzeugt.

Parameter	Beschreibung
NodeId	Die Node ID des aufzurufenden Alarms oder der Condidition.
Output Topic	Name des Output Topic. Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar.
Suppress branched Events	Unterdrückt Events, deren Branch ID gesetzt ist.
Field Name	Freier Feldname des Properties in der OutputEdgeMessage.
Attribute Name	Name des Attributs im OPC UA Server

3.2.2. Konfiguration in der UI

Das OPC UA-Modul kann über die Auswahl des Moduls auf der Startseite oder über die linke Menüleiste der RoboGate Edge UI aufgerufen und konfiguriert werden.

Server konfigurieren

Über New kann ein neuer OPC UA Server hinzugefügt und konfiguriert werden. Entsprechend der Anforderungen können die zuvor näher beschriebenen Parameter definiert werden (Abbildung 15).

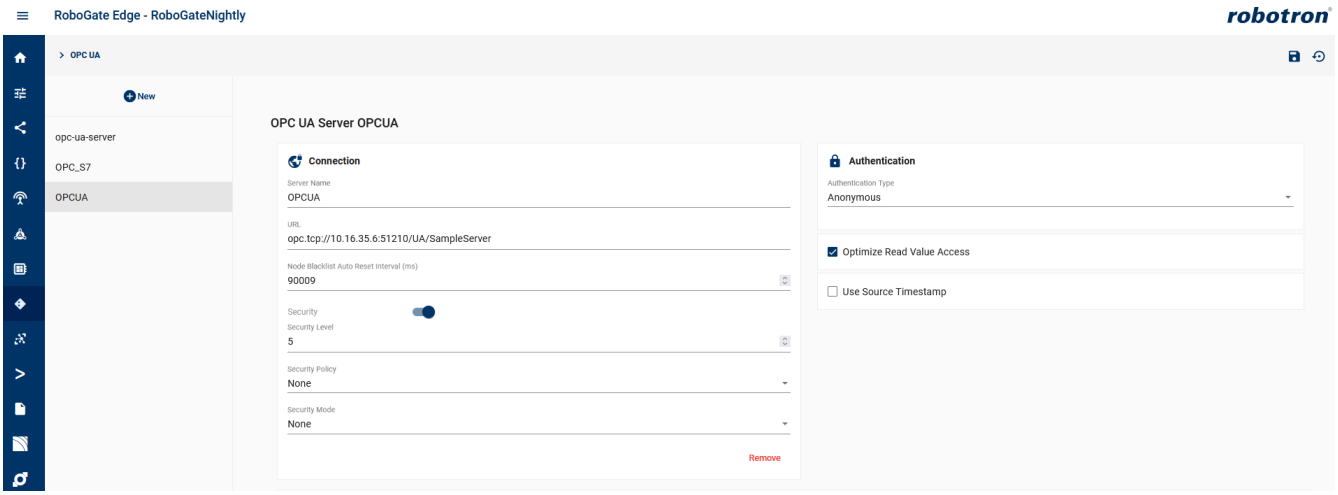


Abbildung 15. Hinzufügen und Konfigurieren eines OPC UA-Servers

Wird diese Konfiguration gespeichert , lässt sich im Bereich *Module State* am Ende der Seite erkennen, ob das Modul eine Verbindung zum OPC UA Server herstellen konnte.

Nun können Subscriptions, Poll Groups und Subscription Groups mit den entsprechenden Datenfeldern (Node IDs) angelegt werden.

Subscription konfigurieren

Um eine Subscription zu konfigurieren, müssen Angaben zum Field Name, zur Node ID und zum Output Topic gemacht werden. Weitere Subscriptions zu Node IDs lassen sich durch einen Klick auf „+“ (Hinzufügen) eintragen. Diese Eingaben können, sofern die Node ID bekannt ist, manuell erfolgen oder mit Hilfe des **OPC UA Node Browser**. Im OPC UA Node Browser werden alle auf dem Server verfügbaren Node IDs angezeigt. Der Node Browser lässt sich über den Button mit dem Lupen-Icon auswählen. Mit einem Klick auf den Pfeil werden die Node IDs aktualisiert und

eine Auswahl aller verfügbaren Node IDs angezeigt (Abbildung 16).

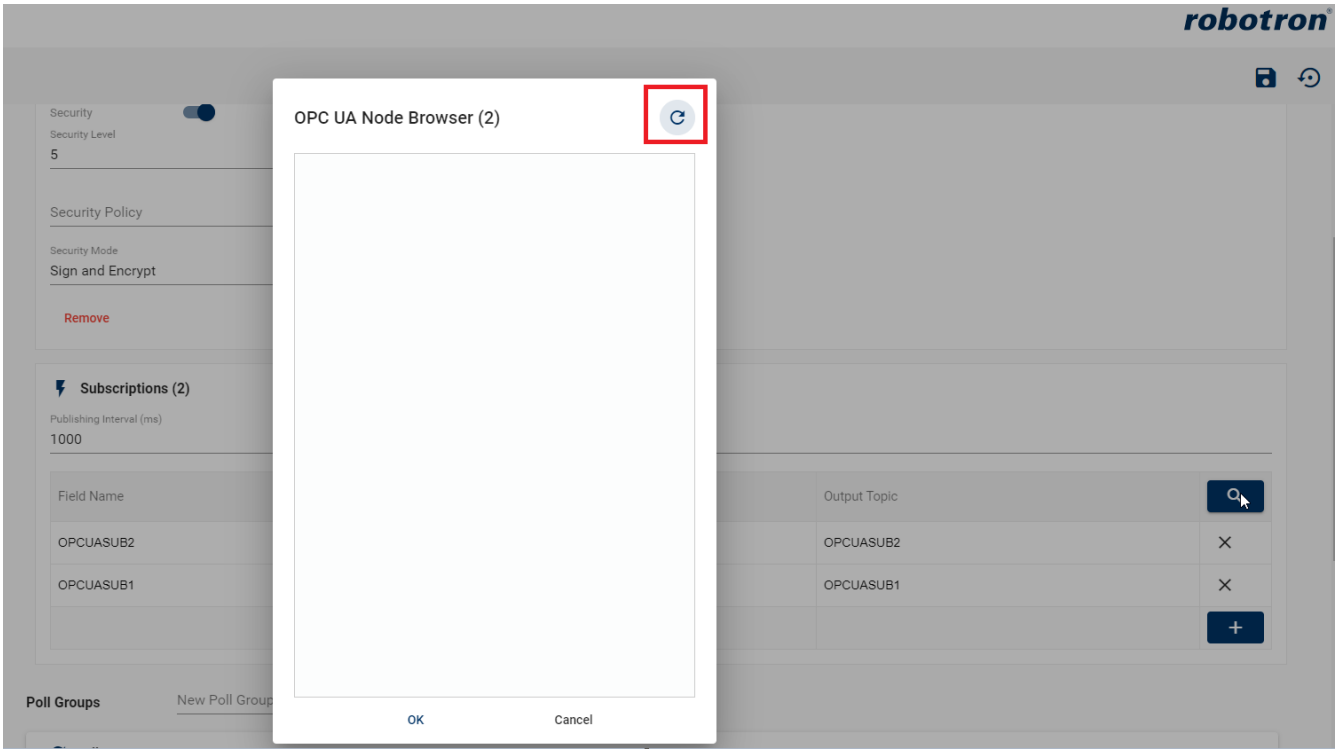


Abbildung 16. Auswahl der NodeID über den OPC UA Node Browser

Für die Subscription gewünschte Nodes können im Node Browser markiert werden. Bei der Auswahl von Node Gruppen kann der Markierungsprozess länger dauern. Nachdem alle Nodes markiert wurden, können diese über den Button „OK“ übernommen werden (Abbildung 17).

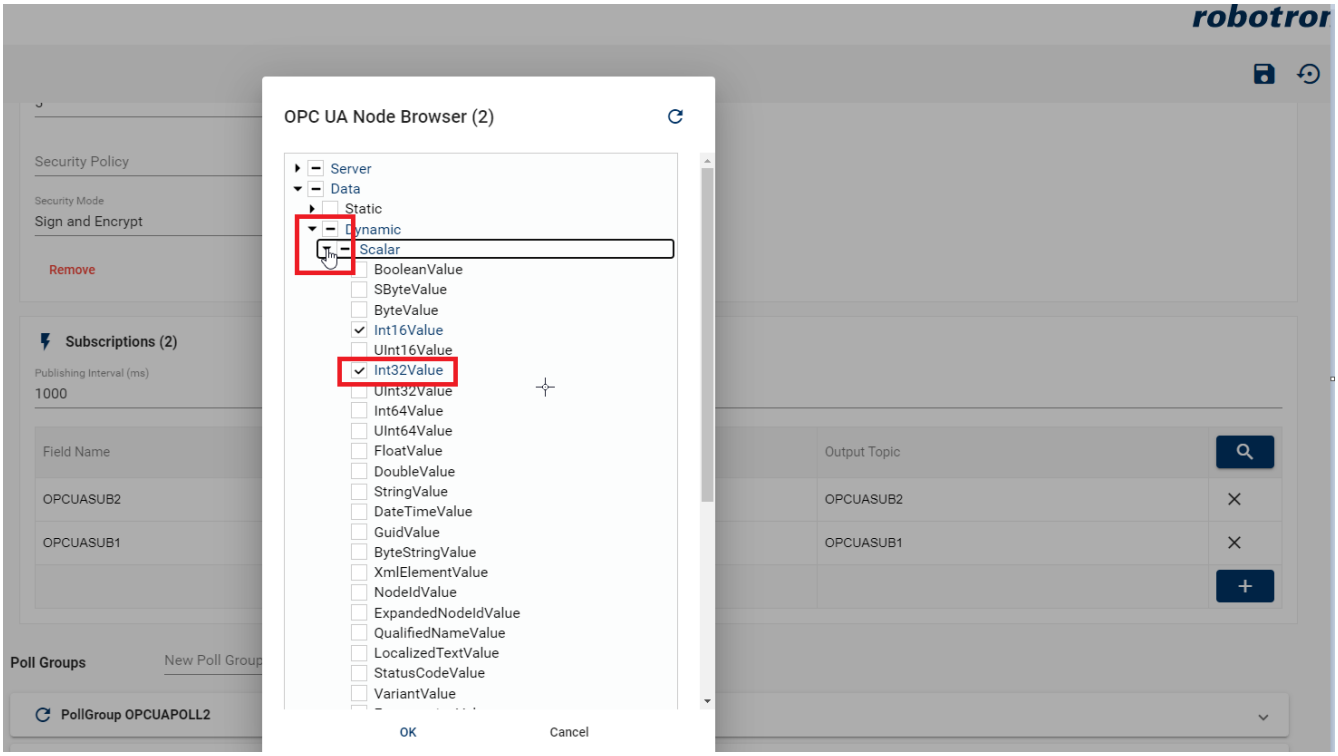


Abbildung 17. Auswahl der Node IDs über den OPC UA Node Browser

Um zu prüfen, ob ein Node korrekt konfiguriert wurde, kann dieser über den Pfeil ↻ in der Spalte "Test" geprüft werden. Das ausgelesene Ergebnis wird nach einem Klick auf den Pfeil angezeigt bzw. bei Änderungen aktualisiert. Initial werden keine Werte angezeigt. Diese Funktion steht für

das Anelgen einer Subscription, Subscription Group und Poll Group zur Verfügung.

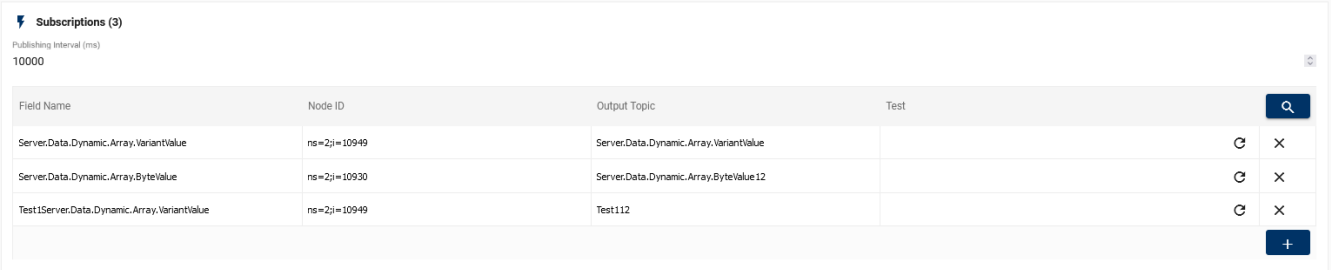


Abbildung 18. Konfiguration der Subscriptions

Nach Fertigstellung der Konfiguration der Subscriptions wird diese über die Speichern-Schaltfläche in der rechten oberen Ecke übernommen.

Method Mapping konfigurieren

Abbildung 19 zeigt die Konfiguration der OPC UA-Methoden mit den in Tabelle 10 beschriebenen Parametern. Diese muss manuell erfolgen.

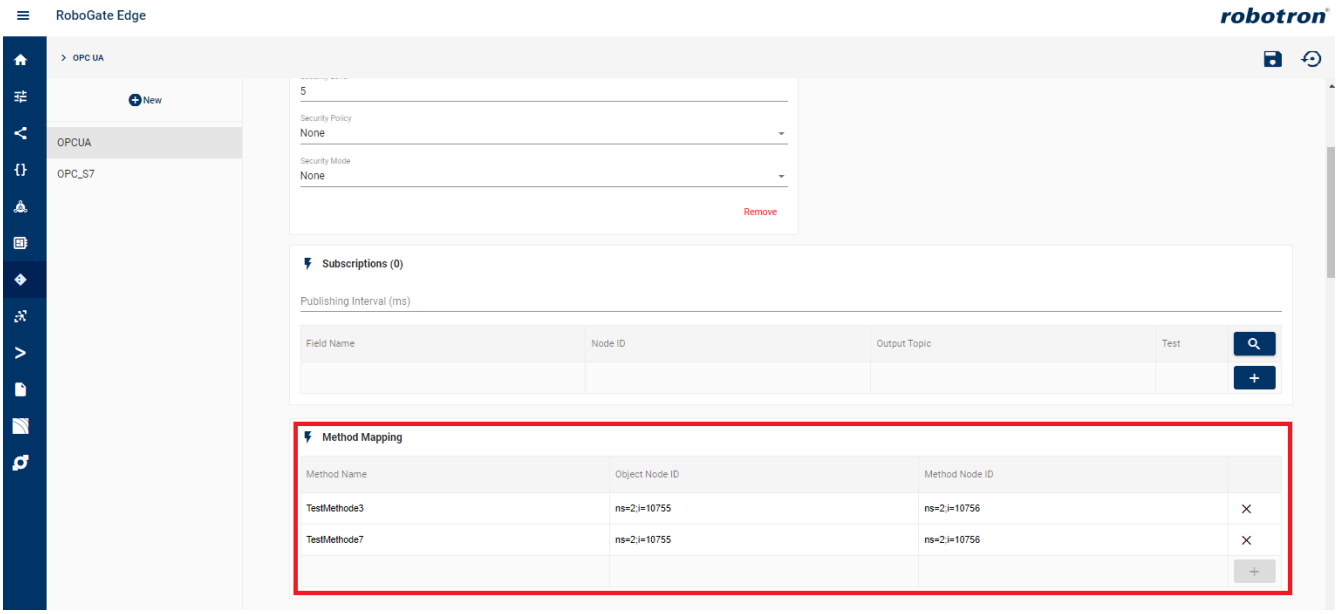


Abbildung 19. Konfiguration von OPC UA-Methoden

Error Code Mapping konfigurieren

Abbildung 20 zeigt eine beispielhafte Konfiguration einer Fehlermeldung durch das Error Code Mapping.

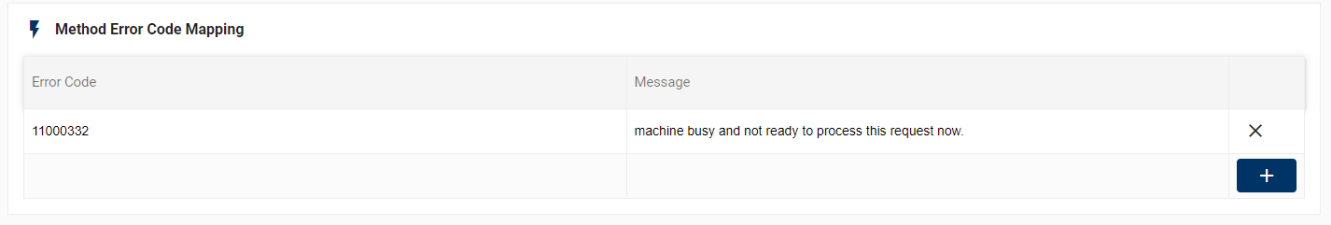


Abbildung 20. Konfiguration von Error Code Mapping

Group konfigurieren

Das Konfigurieren von Poll oder Subscription Groups funktioniert prinzipiell gleich.

- **Poll Groups** erzeugen aktive Datenabfragen.
- **Subscription Groups** reagieren auf Änderungen semantisch zusammenhängender Datenpunkte.

Zuerst erstellen Sie eine neue Poll/Subscription Group, indem Sie einen Namen der Poll/Subscription Group vergeben und auf "Create" klicken. Hierdurch wird eine neue Gruppe erzeugt (Abbildung 21). Anschließend sind Intervall und Output Topic zu definieren und die Node IDs zu konfigurieren (Abbildung 22). Das Browsen nach verfügbaren Node IDs ist identisch zum Vorgang für die Erstellung von Subscriptions.

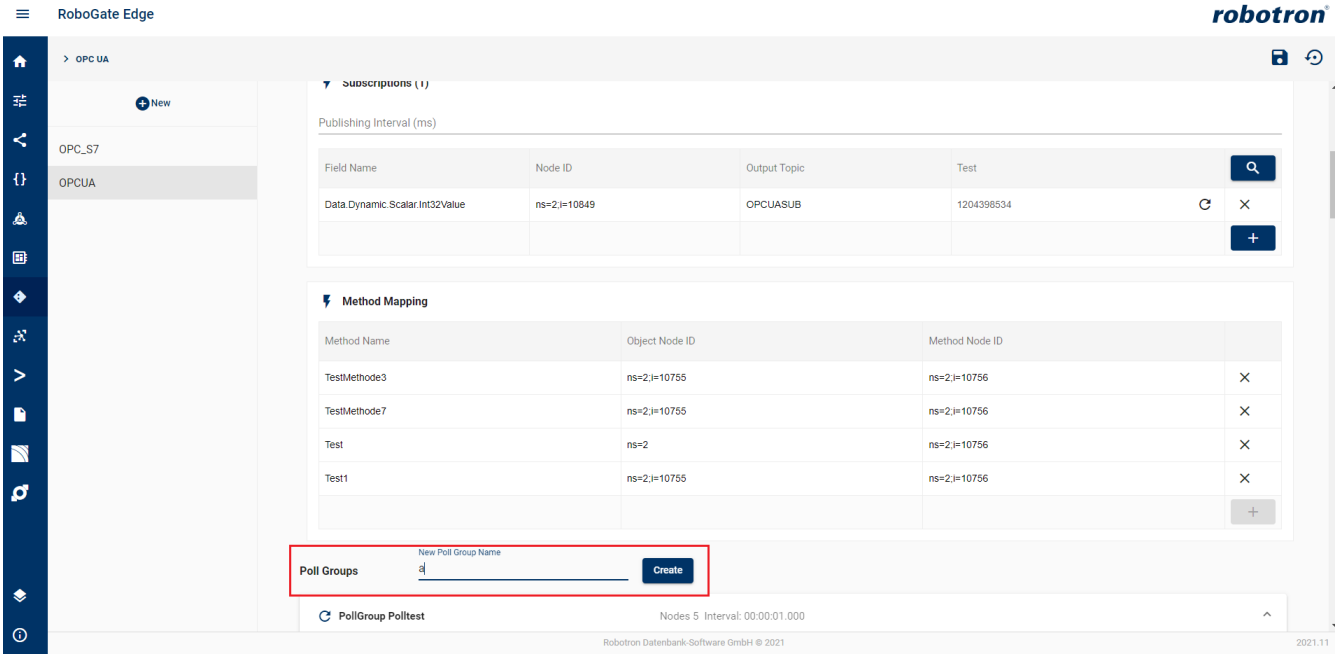


Abbildung 21. Konfiguration einer neuen Poll Group "a"

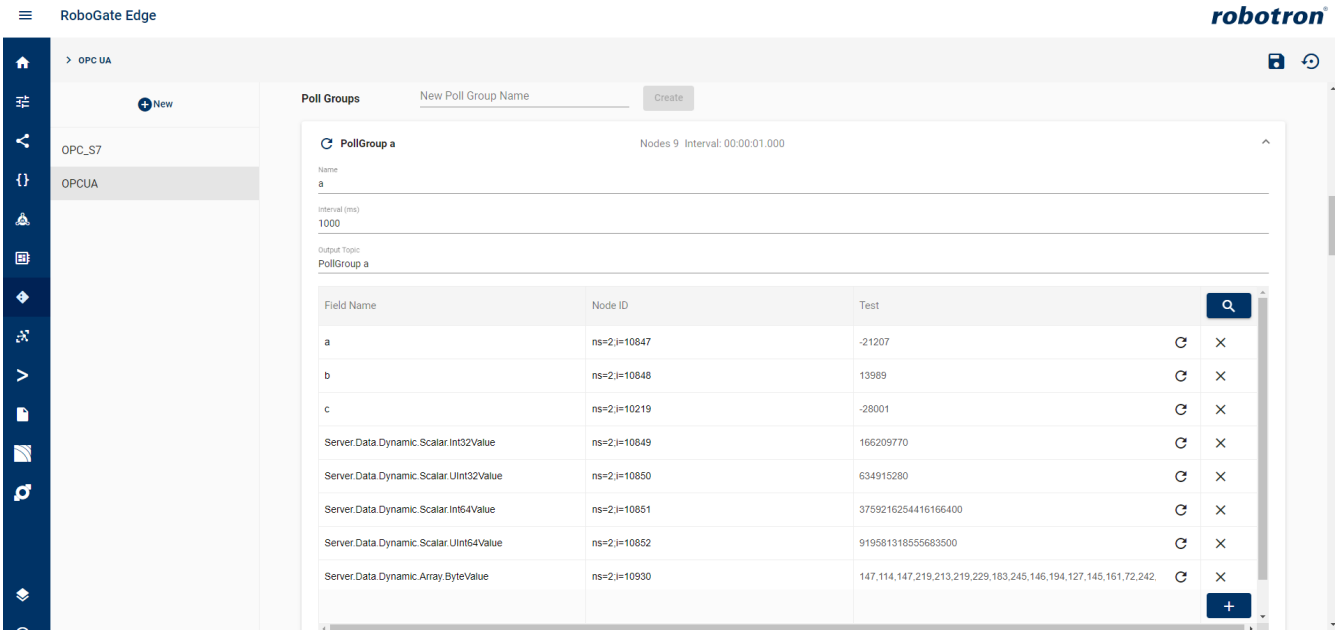


Abbildung 22. Konfiguration der Poll Group "a"

Bei Subscription Groups muss zusätzlich konfiguriert werden, ob Nachrichten *On Change* und/oder *On Resubmit* verschickt werden sollen (Abbildung 23).

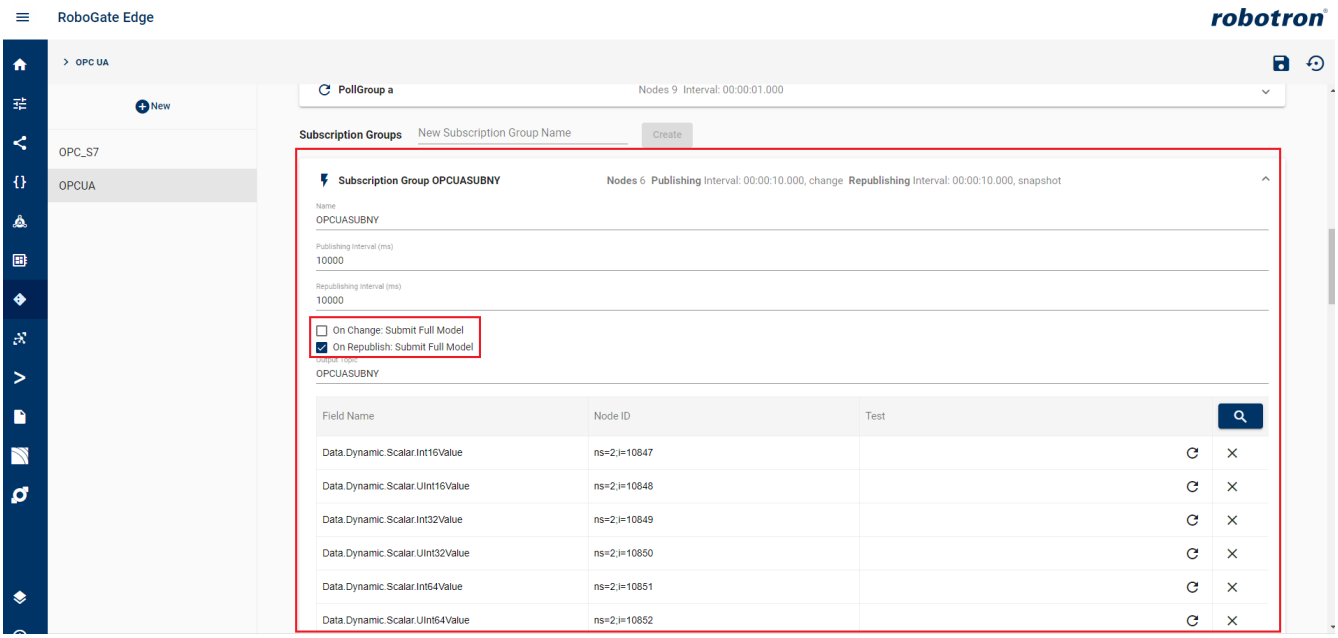


Abbildung 23. Konfiguration einer Subscription Group

Mit Hilfe des gebogenen Pfeils ist es möglich den aktuellen Wert eines Knotens auszugeben. Will man die gesamte Gruppe auslesen, muss der gebogene Pfeil im Header betätigt werden. Der grüne Haken bedeutet dann die Anzahl an korrekt ausgelesenen Werten, das rote Kreuz die Anzahl an fehlerhaften Knoten.

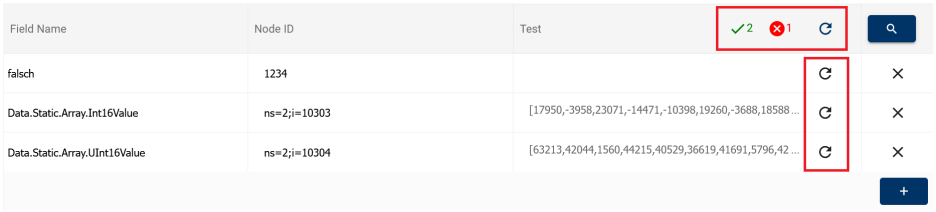


Abbildung 24. Aktuelle Werte der Knoten auslesen

Nach Fertigstellung der Konfiguration des OPC UA-Moduls kann diese über die Schaltfläche „Save“ gespeichert werden. Durch das Speichern der Modulkonfiguration steht diese im RoboGate Edge zur Verfügung.

Alarms & Conditions konfigurieren

Abbildung 25 zeigt eine beispielhafte Konfiguration eines Condition Observers.

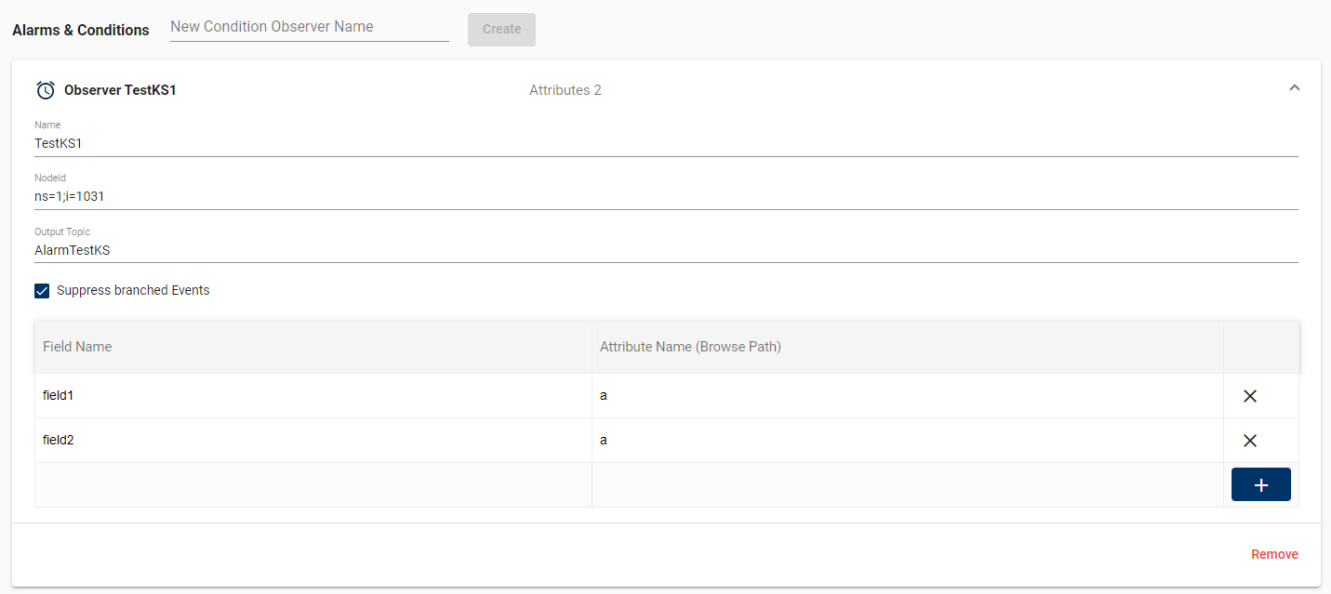


Abbildung 25. Konfiguration eines Condition Observer

Node Browser-Konfiguration bearbeiten / löschen

Innerhalb der Konfiguration des OPC UA Moduls kann die Node-Auswahl im Node Browser für Subscriptions und Poll Groups entfernt werden. Über das Lupen-Icon in der jeweiligen Konfiguration wird die hinterlegte Konfiguration angezeigt und steht zur weiteren Bearbeitung zur Verfügung. Markierungen können entfernt oder hinzugefügt werden. Mit der Bestätigung der Änderung mit „OK“ und dem anschließenden Speichern der Konfiguration werden die Änderungen übernommen. Beim Entfernen aller Markierungen im Browser wird automatisch der gesamte Node Browser aus der Konfiguration entfernt (Abbildung 26).

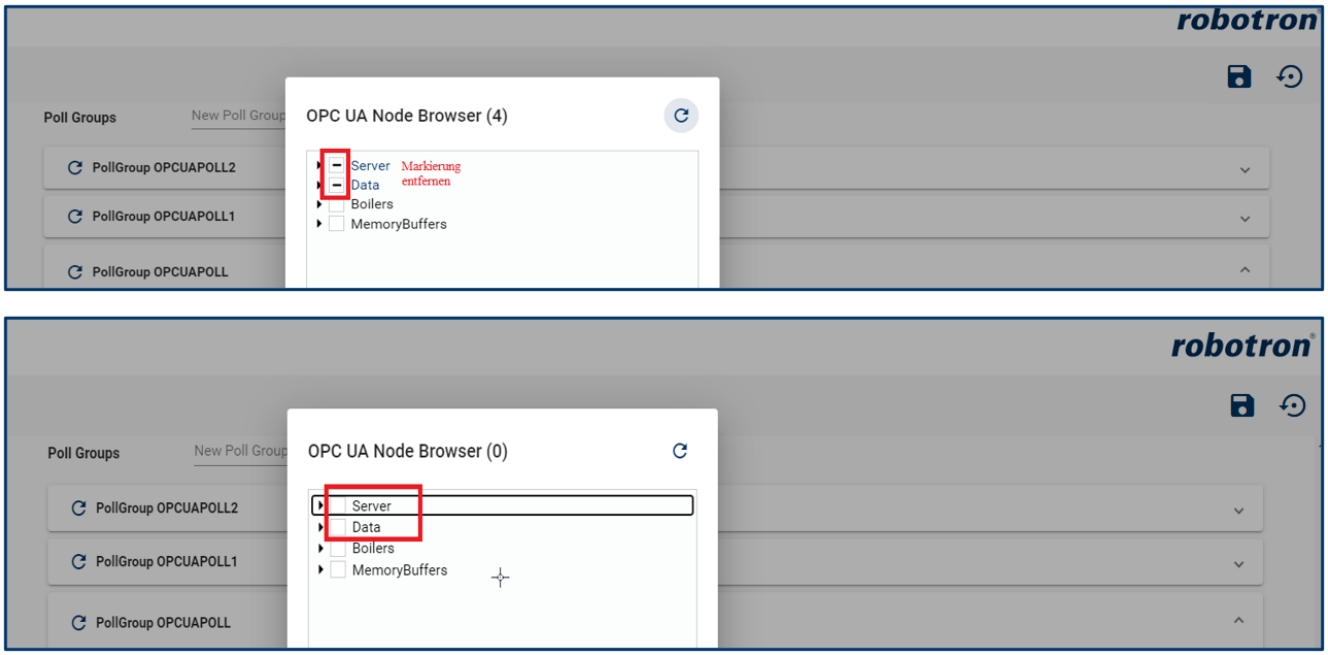


Abbildung 26. Löschen des gesamt ausgewählten Node Browsers

Datenfeld löschen

Innerhalb der Konfiguration des OPC UA-Moduls können Datenfelder der Subscriptions und Poll Groups gelöscht werden. Dazu ist das zu löschende Datenfeld zu markieren und per Mausklick auf

zu entfernen. Durch das Speichern der Konfigurationsänderung steht diese im RoboGate Edge zur Verfügung (Abbildung 27).

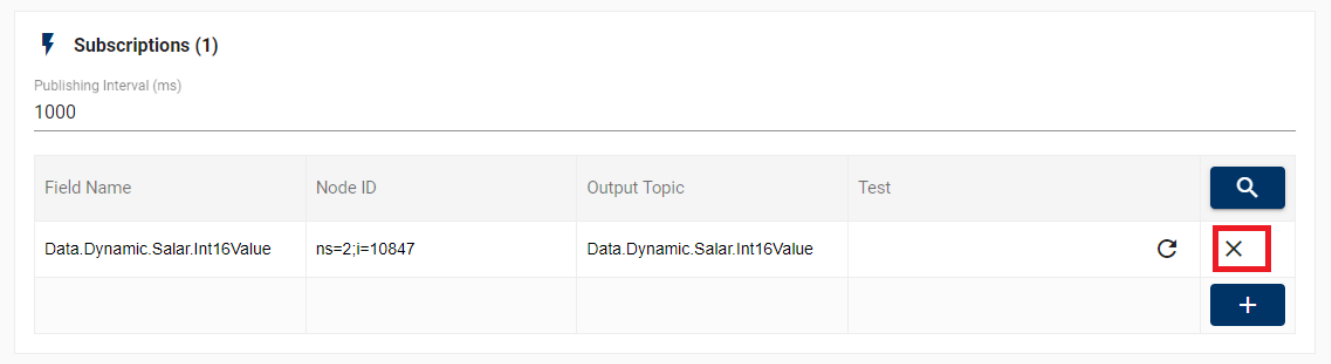


Abbildung 27. Löschen von Nodes

Group löschen

Groups können vollständig gelöscht werden. Durch Betätigen des Löschen-Buttons wird die markierte Poll oder Subscription Group gelöscht (Abbildung 28). Durch das Speichern der gesamten OPC UA-Konfiguration werden die Änderungen übernommen und stehen zur Verfügung.

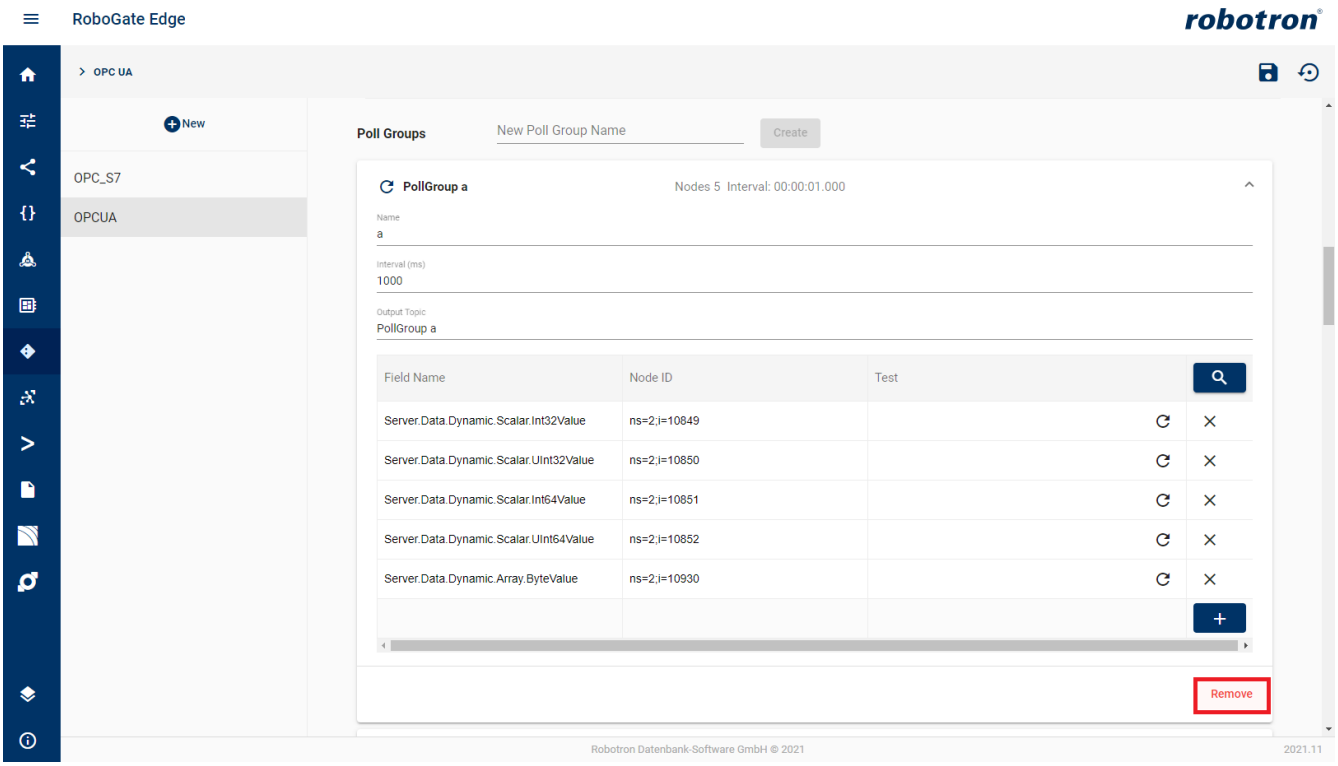


Abbildung 28. Löschen einer Poll Group

Konfiguration zurücksetzen

Über die Schaltfläche in der horizontalen Menüleiste lässt sich die komplette Konfiguration des Moduls auf Werkseinstellungen zurücksetzen. Nach einem Klick auf die Schaltfläche folgt ein Bestätigungsdialog mit "Yes" und "No" zur Rückfrage ob der Vorgang wirklich ausgeführt werden soll.

> Damit wird auch die gesamte Historie auf dem RoboGate Edge gelöscht!

3.3. RFC 1006

Mit dem RFC 1006-Modul können Steuerungen (engl. Programmable Logic Controller (PLC)) zyklisch ausgelesen und die Werte weitergeleitet werden.

Die Wartezeit zwischen den Lesevorgängen ist millisekundengenau einstellbar (Leseintervall). Die ausgelesenen Werte können individuell benannt werden, um eine korrekte Zuordnung dieser zu ermöglichen. Beim Einspielen einer neuen Konfiguration werden laufende Lesevorgänge abgebrochen und neue Lesevorgänge mit der neuen Konfiguration gestartet. Werte aus der gleichen Steuerung mit dem gleichen Leseintervall werden nacheinander mittels Poll Groups gelesen und in einer Nachricht versendet.

3.3.1. Konfigurationsparameter

PLC

Die in folgender Tabelle beschriebenen Parameter müssen für die Steuerung definiert werden.

Tabelle 15. Konfiguration der PLC

Parameter	Beschreibung
PLC Name (Source)	Der Name der Steuerung ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar. Der Name stellt den Namen der Datenquelle dar. Dieser entspricht der Source in den Auswertungen. Die Verbindung (Connection) zum RFC-Modul wird über die URL des Servers hergestellt.
IP Address	IP-Adresse der Steuerung
CPU Type	Eine Auswahl verschiedener CPU-Typen steht zur Auswahl.
Rack	Nummer des Baugruppenträgers. Für Siemenssteuerung gilt das Rack „0“ als Standard für u.a. folgende Typen: S7-1200, S7-1500, S7-300, S7-400
Slot	Position der Steuerung auf Baugruppenträger. Für Siemens-Steuerung gilt der Slot „0“ als Standard für u.a. folgende Typen: S7-1200, S7-1500 Der Slot „2“ ist Standard für u.a. folgende Typen: S7-300, S7-400

> Um den Zugriff auf eine Siemens S7-Steuerung zu ermöglichen, muss die Freigabe „Permit access with PUT/GET communication from remote partner.“ im TIA Portal erfolgen. Diese Einstellung befindet sich in den Projekt-Eigenschaften unter dem Menüpunkt „Protection“.

Poll Groups

Jede Poll Group erzeugt eine Nachricht (Payload-Typ Model). Das Model entspricht einer Nachricht mit Werten mehrerer Datenpunkte.

Tabelle 16. Konfiguration von Poll Groups im RFC 1006-Modul

Parameter	Beschreibung
Poll Group Name (Scope)	Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar. Dieser Name entspricht den Namen des auszulesenden Bereichs. Der Name wird unter Scope in den Auswertungen bereitgestellt.
Interval (ms)	Intervall in Millisekunden gibt die Taktung an, in der die Werte abgefragt werden.
Output Topic	Name des Output Topic z.B. RFCpg1. Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar.
Field Name	Bezeichnung des zu lesenden Wertes. Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar.
Data Type	Dieser Wert bezeichnet den Speicherbereich auf der Steuerung. Folgende Standardtypen stehen zur Verfügung: • Input • Output • Memory • Data Block • Timer • Counter
Data Block	Angabe des auszulesenden DataBlocks.
Start Byte	Erstes, zu lesendes Byte innerhalb des Datenblocks.
Var Type	Datentyp des zu lesenden Wertes. Folgende Typen können ausgelesen werden: • Bit • Byte • Word • DWord • Int • DInt • Real • String • String Ex (Hinweis: für die Siemenssteuerung) • Timer • Counter

Parameter	Beschreibung
Count	Anzahl der zu lesenden Werte. Wenn dieser Wert größer 1 ist, wird automatisch ein Wertfeld (Array) beim Auslesen erzeugt.
Bit	Position des zu lesenden Bits innerhalb des Bytes. Optional, nur notwendig, wenn Var Type einem Bit entspricht. Ansonsten wird der Wert ignoriert.

Subscription Groups

Jede Subscription Group erzeugt eine Nachricht. Mit der Konfiguration von Subscription Groups kann auf Änderung semantisch zusammenhängender Datenpunkte reagiert werden.

Tabelle 17. Konfiguration von Subscription Groups im RFC 1006-Modul

Parameter	Beschreibung
Subscription Group Name (Scope)	Name der Subscription Group. Dieser Name entspricht den Namen des auszulesenden Bereichs. Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar.
Publishing Interval (ms)	Intervall in Millisekunden gibt die Taktung an, in der die Werte abgefragt werden.
Republishing Interval (ms)	Intervall in Millisekunden gibt die Taktung an, in der die zuletzt gelesenen Werte erneut gesendet werden.
On Change: Submit Full Model	**Aktiv**: Falls eine Änderung festgestellt wird, wird der geänderte Datenpunkt gemeinsam mit den zuletzt gelesenen Werten der anderen Datenpunkte als Model-Nachricht übermittelt. **Inaktiv**: Falls eine Änderung festgestellt wird, wird nur der geänderte Datenpunkt als Event-Nachricht übermittelt.
On Republish: Submit Full Model	**Aktiv**: Während des Republishing, werden die Datenpunkte als eine Datamodelnachricht versandt. **Inaktiv**: Während des Republishing werden die Datenpunkte jeweils als individuelle Nachricht (Event) versandt.
Output Topic	Name des Output Topic z.B. RFCpgSUB. Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar.
Field Name	Bezeichnung des zu lesenden Wertes. Der Name ist bestehend aus Buchstaben, Zahlen und ohne Sonderzeichen frei konfigurierbar.

Parameter	Beschreibung
Data Type	Dieser Wert bezeichnet den Speicherbereich auf der Steuerung. Folgende Standardtypen stehen zur Verfügung: • Input • Output • Memory • Data Block • Timer • Counter
Data Block	Angabe des auszulesenden DataBlocks.
Start Byte	Erstes, zu lesendes Byte innerhalb des Datenblocks.
Var Type	Datentyp des zu lesenden Wertes. Folgende Typen können ausgelesen werden: • Bit • Byte • Word • DWord • Int • DInt • Real • String • String Ex • Timer • Counter
Count	Anzahl der zu lesenden Werte. Wenn dieser Wert größer 1 ist, wird automatisch ein Wertfeld (Array) beim Auslesen erzeugt.
Bit	Position des zu lesenden Bits innerhalb des Bytes. Optional, nur notwendig, wenn Var Type einem Bit entspricht. Ansonsten wird der Wert ignoriert.

3.3.2. Konfiguration in der UI

Das RFC 1006-Modul kann über die Auswahl des Moduls auf der Startseite oder über die linke Menüleiste der RoboGate Edge UI aufgerufen und konfiguriert werden.

PLC konfigurieren

Über ⊕ New kann eine Steuerung hinzugefügt und mit den zuvor beschriebenen

Konfigurationsparametern konfiguriert werden (Abbildung 29). Entsprechend der Anforderungen ist die Konfiguration des Moduls vorzunehmen.

Nachdem die allgemeine Konfiguration der Steuerung erfolgt ist (Abbildung 30) und diese gespeichert wurde, kann die Konfiguration der Poll oder Subscription Groups durchgeführt werden. Im Bereich *Module State* am Ende der Seite lässt sich erkennen, ob das Modul eine Verbindung zur Steuerung herstellen konnte.

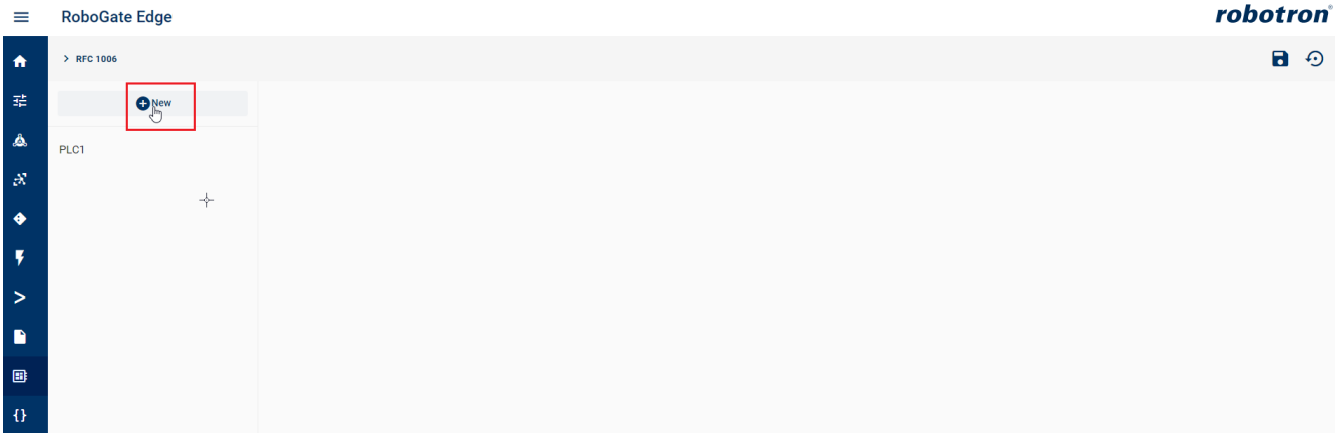


Abbildung 29. Hinzufügen einer Steuerung

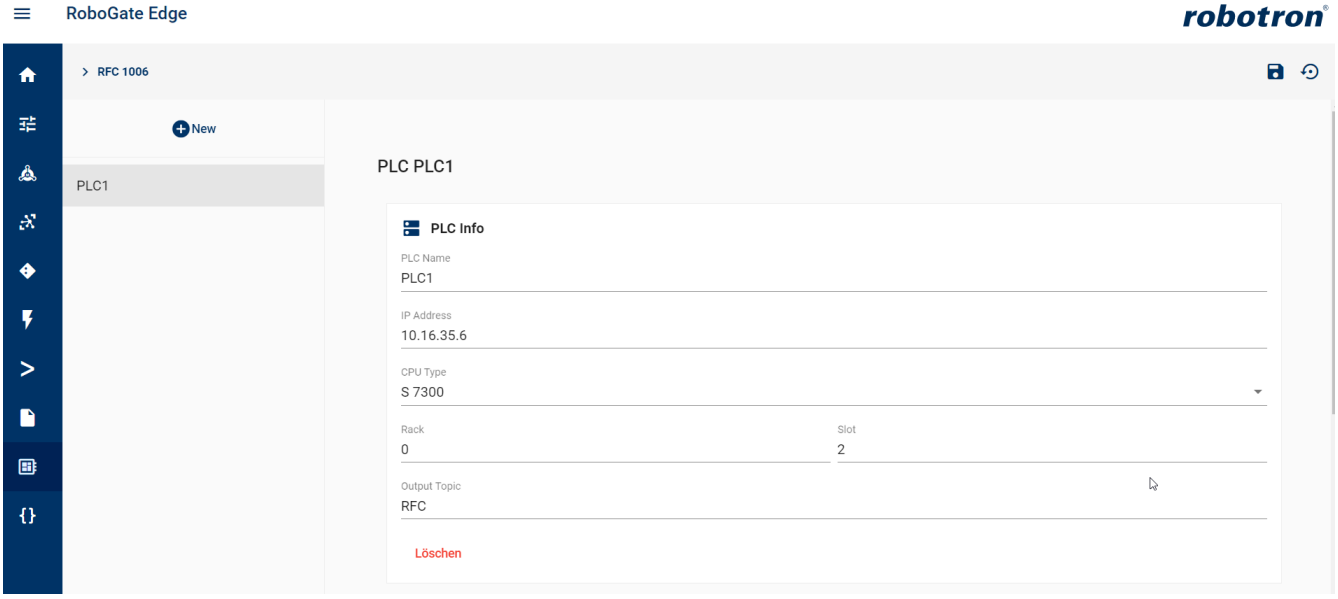


Abbildung 30. Konfiguration der Steuerung

Group konfigurieren

Mit der Konfiguration von **Poll Groups** werden aktive Datenabfragen erzeugt. Die Konfiguration erfordert die Angabe eines Namens der Poll Group, des Intervalls und Output Topics. Mit einem Klick auf „Create“ wird die Konfiguration übernommen und eine Poll Group erzeugt ([img-rfc-newgPollgroup]).

Nun können die auszulesenden Datenblöcke mit ihren Eigenschaften (s. Tabelle 16) konfiguriert werden.

Um zu prüfen, ob der Datenblock in einer Poll oder Subscription Group korrekt konfiguriert wurde, kann dieser in der Spalte "Test" geprüft werden. Das ausgelesene Ergebnis wird nach einem Klick

auf den Pfeil angezeigt bzw. bei Änderungen aktualisiert. Initial werden keine Werte angezeigt.

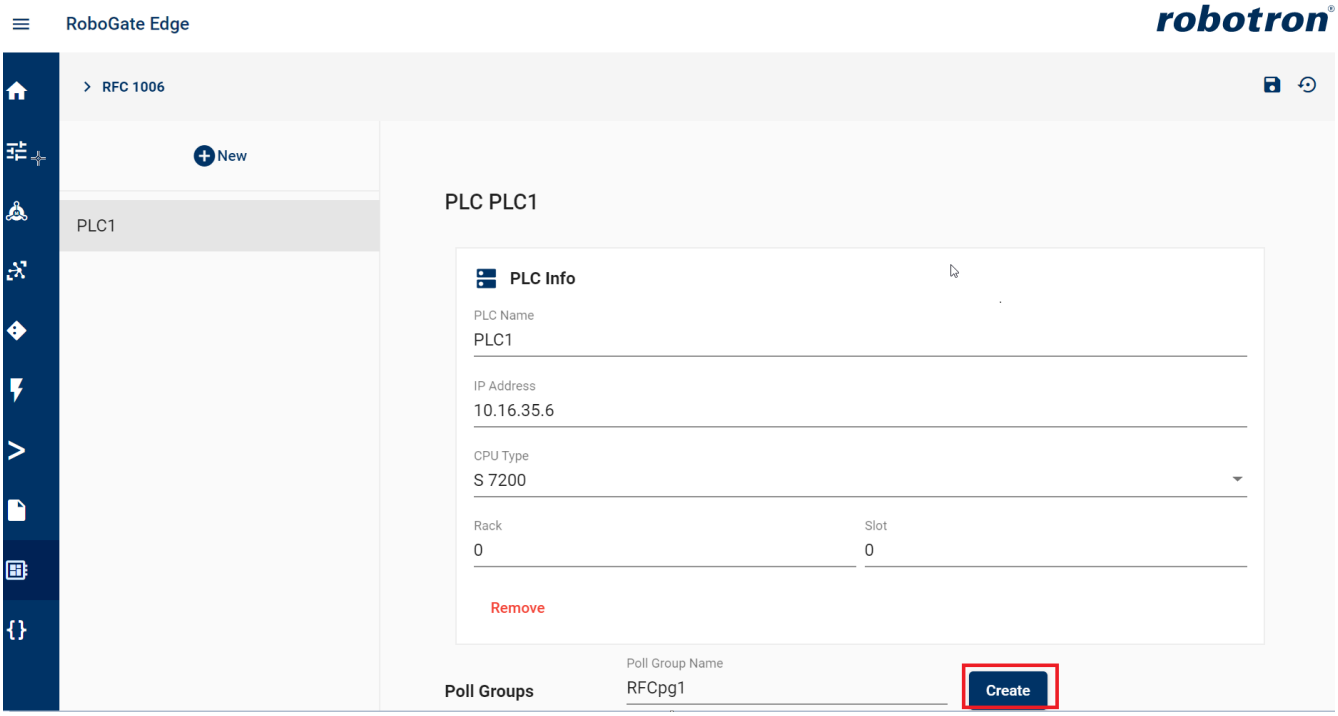


Abbildung 31. Konfiguration einer neuen Poll Group

Nach Fertigstellung der Konfiguration der Poll Group des RFC 1006 Moduls muss diese über die Speicher-Schaltfläche gespeichert werden (Abbildung 32). Durch das Speichern der Modul-Konfiguration steht diese im RoboGate Edge zur Verfügung.

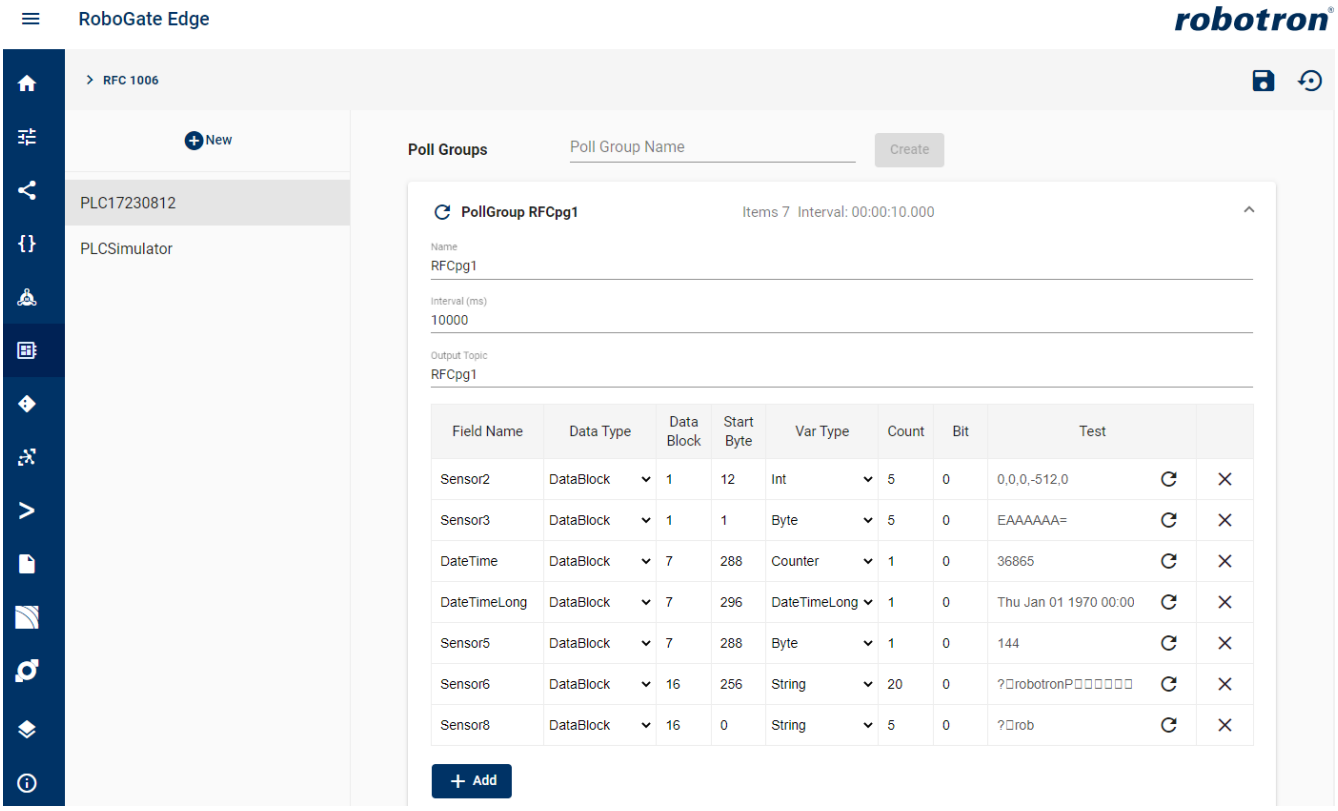


Abbildung 32. Speichern der Konfiguration der Poll Group

Mit der Konfiguration von **Subscription Groups** kann auf Änderung semantisch zusammenhängender Datenpunkte reagiert werden. Die Konfiguration erfordert die Angabe eines

Namens der Subscription Group, welcher über „Create“ übernommen wird und wodurch eine Gruppe erzeugt wird (Abbildung 33) .

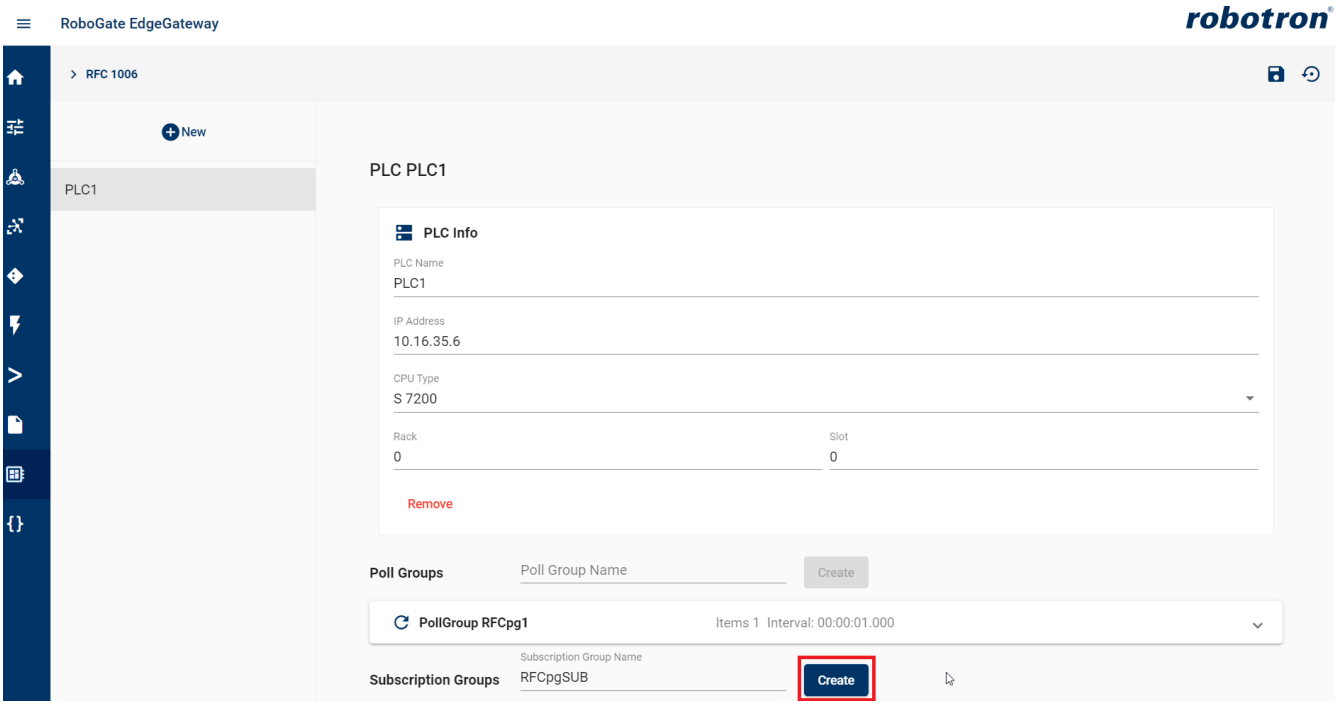


Abbildung 33. Erstellen einer neuen Subscription Group

Die neu erstellte Subscription Group kann anschließend, entsprechend Ihren Anforderungen weiter konfiguriert werden (Abbildung 34). Nach Fertigstellung der Konfiguration wird diese über das Speichern in der rechten oberen Ecke übernommen. Durch das Speichern der gesamten Konfiguration des RFC 1006-Moduls werden die Daten entsprechend des Datenmodells ausgelesen und stehen für die Datenweiterverarbeitung zur Verfügung.

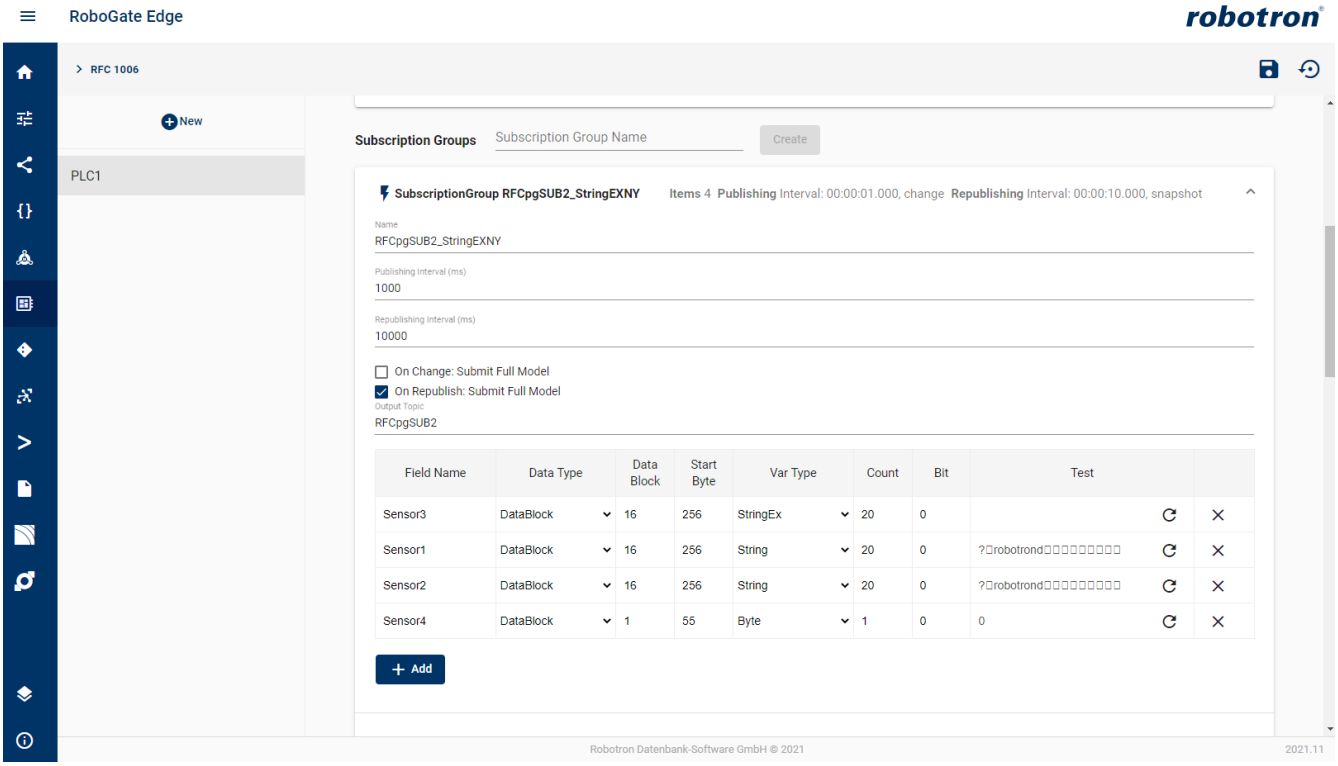


Abbildung 34. Konfiguration der Subscription Group

Datenfeld löschen

Innerhalb der Konfiguration des RFC 1006-Moduls können Datenfelder der Subscription und Poll Groups gelöscht werden. Dazu ist das zu löschende Datenfeld zu markieren und per Klick auf ✕ zu entfernen. Durch das Speichern der Konfigurationsänderung steht diese im RoboGate Edge zur Verfügung (Abbildung 35).

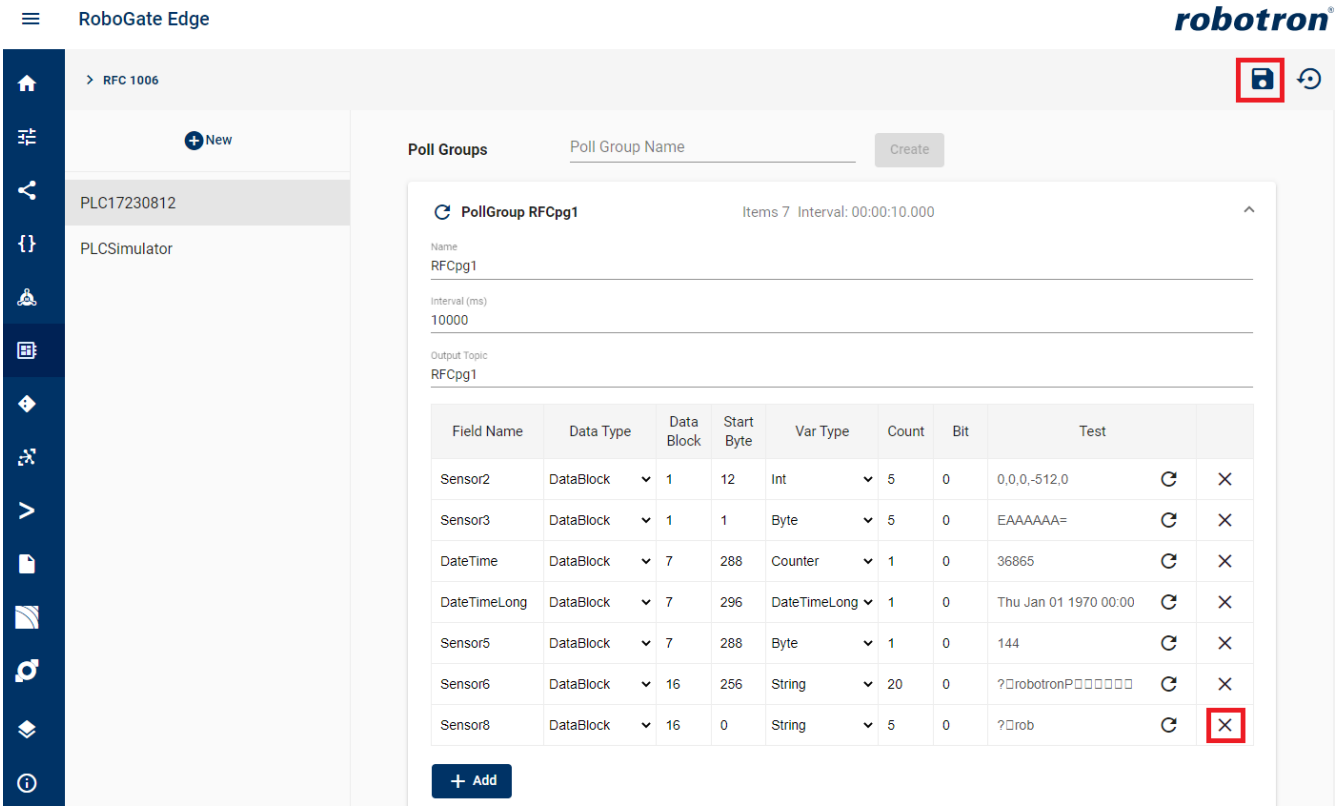


Abbildung 35. Löschen von Datenfeldern

Group löschen

Groups können vollständig gelöscht werden. Durch Betätigen der Schaltfläche "Remove" wird die markierte Group gelöscht (Abbildung 36). Durch das Speichern der gesamten RFC 1006 Konfiguration werden die Änderungen übernommen und stehen zur Verfügung.

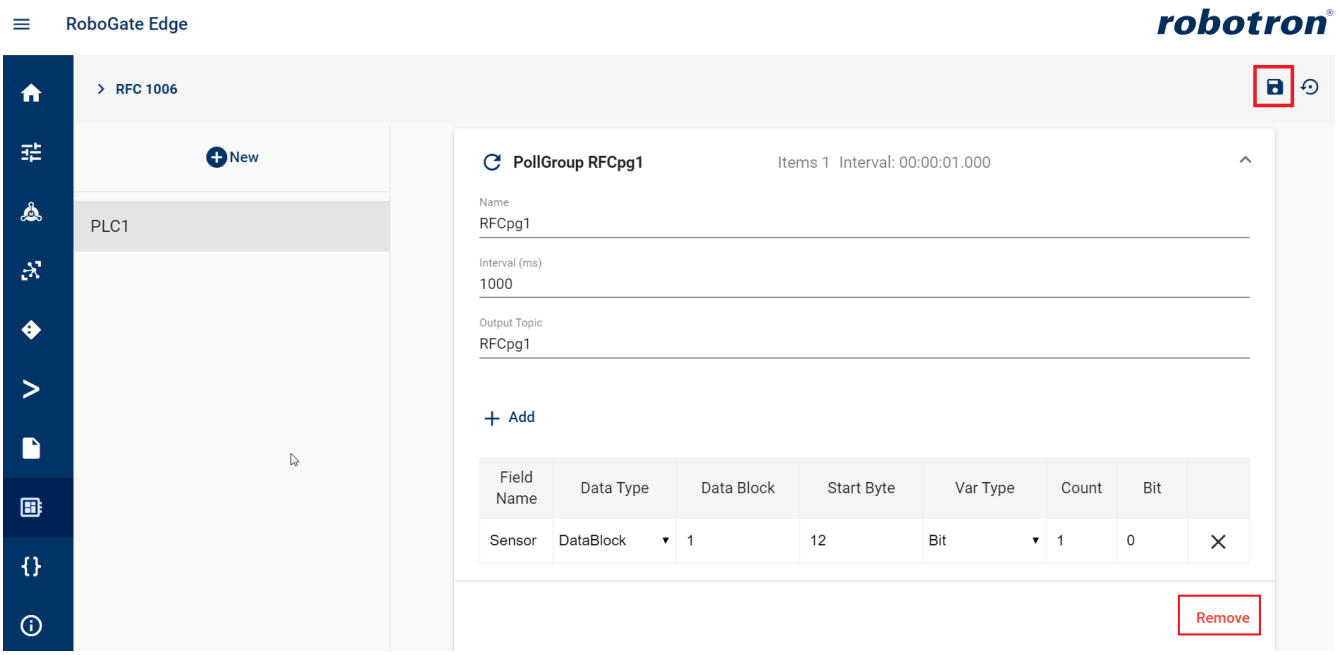


Abbildung 36. Löschen der Poll Group

PLC löschen

Angelegte Steuerungen können vollständig gelöscht werden. Dazu ist innerhalb der „PLC Info“ die Schaltfläche "Remove" zu nutzen (Abbildung 37). Mit dem Speichern wird die Änderung übernommen. Die gelöschte Steuerung wird aus der Übersicht entfernt.

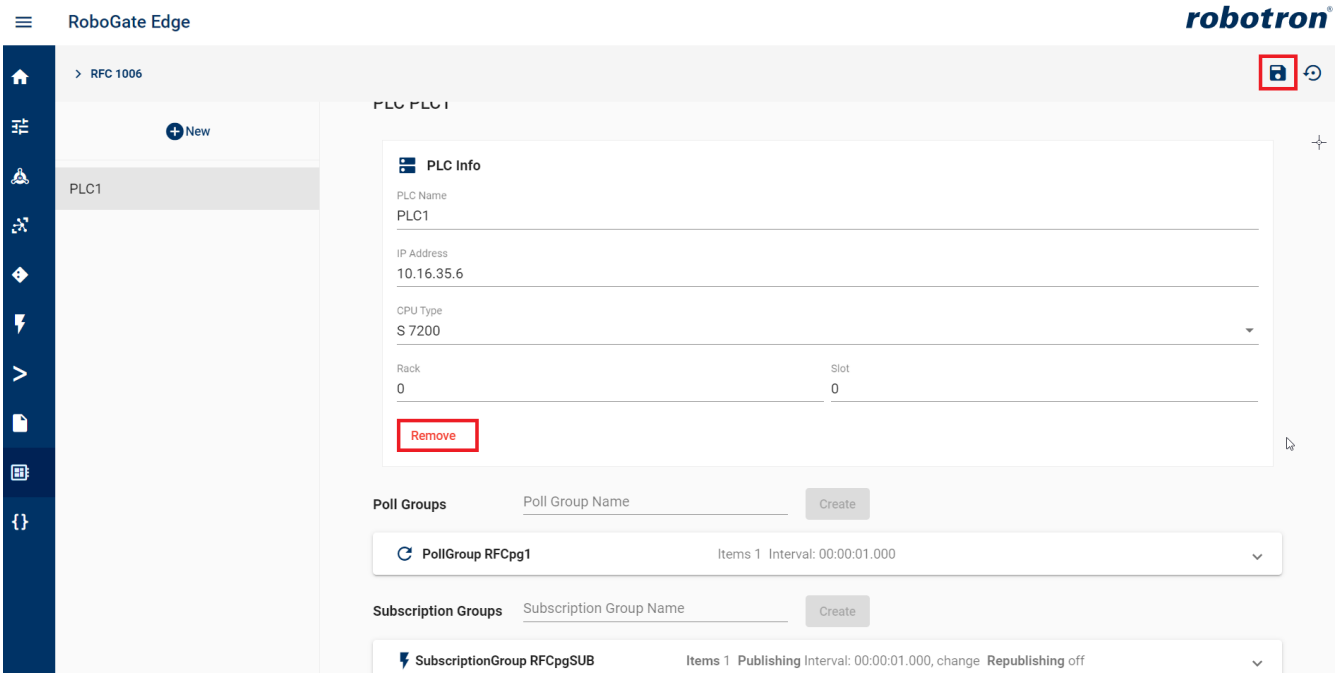


Abbildung 37. Löschen einer angelegten Steuerung

Konfiguration zurücksetzen

Über die Schaltfläche in der horizontalen Menüleiste lässt sich die komplette Konfiguration des Moduls auf Werkseinstellungen zurücksetzen. Nach einem Klick auf die Schaltfläche folgt ein Bestätigungsdialog mit "Yes" und "No" zur Rückfrage ob der Vorgang wirklich ausgeführt werden soll.

> Damit wird auch die gesamte Historie auf dem RoboGate Edge gelöscht!

4. Processing-Module

4.1. File Logger

Unresolved directive in RoboGate_Dokumentation.adoc - include::../modules/processing/file_logger.adoc[leveloffset=+2]

4.2. Stream Processor

Mit dem Stream Processor können einlaufende Nachrichten durch Verschaltung verschiedener Verarbeitungsblöcke ("Processing Blocks") verarbeitet und anschließend zurück an den Edge Broker des RoboGate Edge gegeben werden. Zu den Funktionen gehören z.B. Datenaggregation und mathematische Berechnungen auf Grundlage einlaufender Nachrichten. Mehrere unabhängige Datenströme können unabhängig voneinander und gleichartig verarbeitet werden, siehe dazu die nachfolgende Skizze.

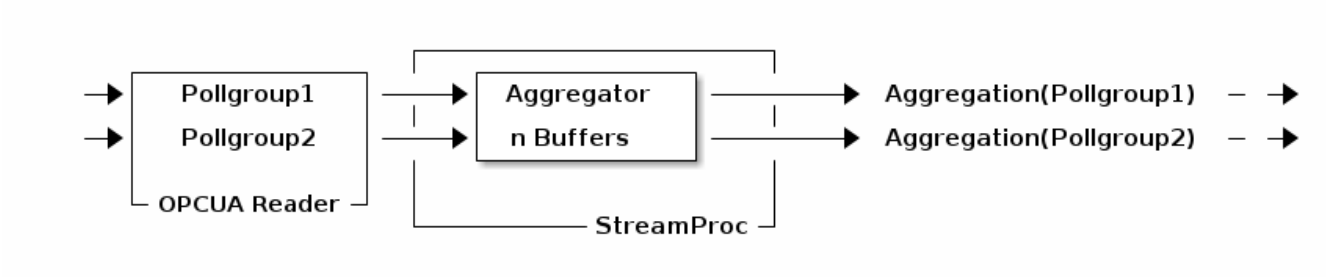


Abbildung 38. Verarbeitung von Nachrichten im Stream Processor

4.2.1. Konfigurationsschema

Die Konfiguration des Stream Processor beinhaltet die Definition folgender Elemente:

- Input in den Stream Processor,
- Verarbeitung (Processing) des Inputs und
- Output aus dem Stream Processor an den Edge Broker.

Für die Konfiguration in der RoboGate Edge UI und Verbindung dieser Elemente bietet der Stream Processor einen **Floweditor** (Abbildung 39) mit folgenden Bestandteilen:

1. Editor-Menüleiste: Zoom, automatische Block-Anordnung auf der Zeichenfläche, Leeren der Zeichenfläche, Löschen eines Blocks
2. Panel "Settings": Konfigurieren von Input Edge Topics und Default Output Topic
3. Panel "Blocks": Sammlung verfügbarer Input-, Processing- und Output-Blöcke
4. Zeichenfläche: Konfiguration, Verbindung und Anordnung von Blöcken
5. Speichern: Speichern der Konfiguration
6. Konfiguration zurücksetzen

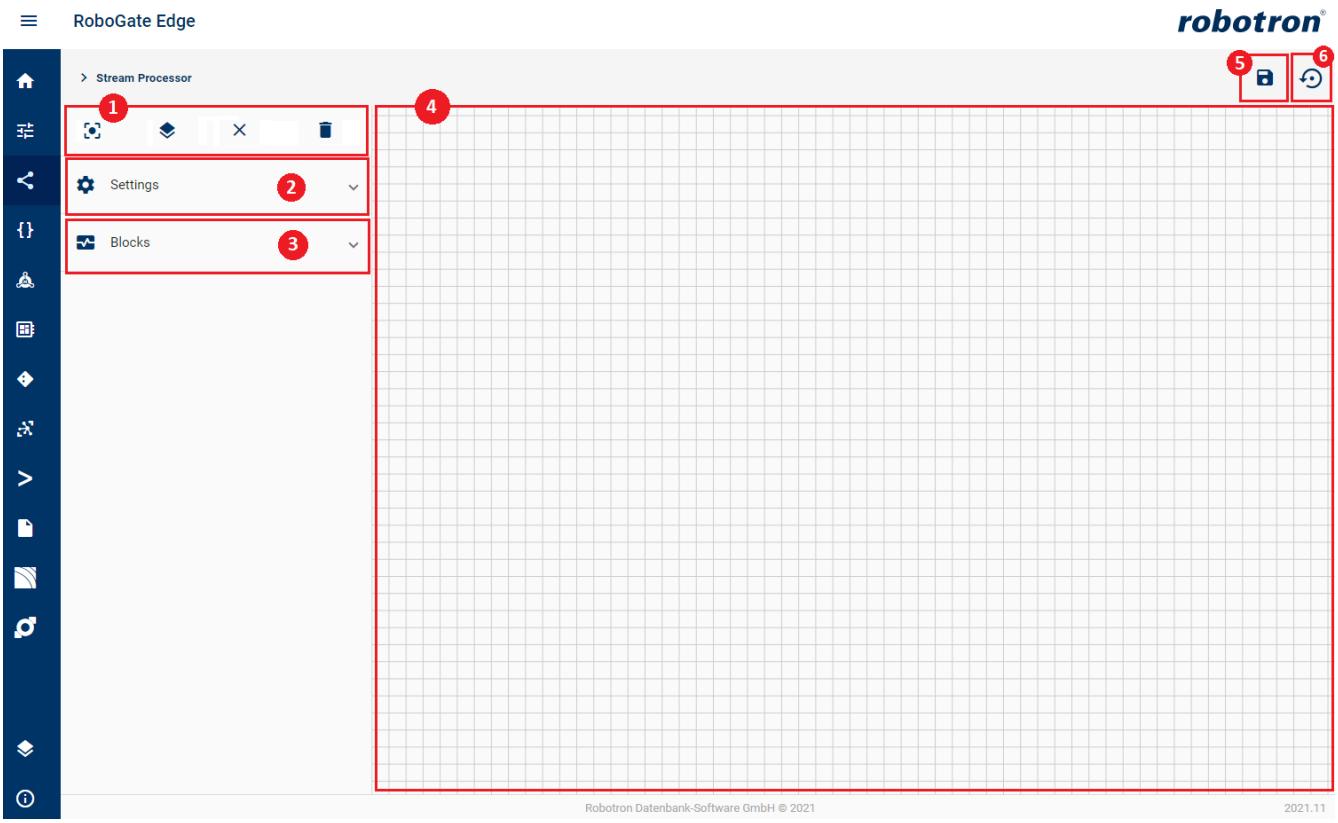


Abbildung 39. Stream Processor Floweditor

Um einen neuen Block anzulegen, klicken Sie das entsprechende Element im Panel "Blocks". Anschließend erscheint dieser Block auf der Zeichenfläche. Sie können den Block via drag&drop frei in der Zeichenfläche verschieben. Um einen Block zu entfernen, klicken Sie entweder auf die Löschschaltfläche im Block in der Zeichenfläche oder wählen Sie den Block an und klicken Sie "Delete Selected" in der Editor-Menüleiste. Um alle Blöcke zu entfernen, klicken Sie auf "Clear" in der Editor-Menüleiste.

Jeder Block hat Ein- und/oder Ausgangsknoten über die er sich verbinden lässt. Über diese Verbindungen werden Nachrichten weitergeleitet. Um Blöcke zu verbinden, klicken Sie auf einen Ausgangsknoten und ziehen eine Linie zu einem Eingangsknoten eines anderen Blocks. Um eine Verbindung zu entfernen, klicken Sie auf den Knoten am Ende der Verbindung und lösen Sie die Verbindung vom Eingangsknoten des Folgeblocks. Wird keine neue Verbindung anglegt, wird die Verbindung automatisch entfernt. Jeder Knotenpunkt kann über mehrere Verbindungen verfügen.

> Die Verbindung der Input-, Processing- und Output-Blöcke ist Voraussetzung für die Verarbeitung von Nachrichten im Stream Processor!

Input

Über die Angabe von **Input Edge Topics** wird definiert, welche der im RoboGate Edge bzw. Edge Broker vorhandenen Topics im Stream Processor verarbeitet werden. Diese Auswahl dient somit als Eingangsfilter in den Stream Processor. Es können mehrere Einträge angelegt werden.

Über einen **Input-Block** lassen sich die konfigurierten Input Edge Topics nach Source und Scope filtern und an einen Processing-Block schicken. Diese Filterung ist durch Reguläre Ausdrücke (RegEx) realisiert. Stimmen die Header-Felder "Source" und "Scope" aus den einlaufenden

Nachrichten mit der Konfiguration des Input-Blocks überein, wird die Nachricht an die nachfolgenden Blöcke gegeben. Es können mehrere Input-Blöcke angelegt werden.

Tabelle 18. Konfigurationsparameter Input-Block

Parameter	Beschreibung
Name	Name des Input-Block
Source (Filter)	RegEx-Filterung des Header-Feld "Source"
Scope (Filter)	RegEx-Filterung des Header-Feld "Scope"

Nach Konfiguration des Inputs in den Stream Processor (Input Edge Topics und Input-Block/Blöcke), müssen die notwendigen Processing-Blöcke konfiguriert und dem Verarbeitungsfluss entsprechend verbunden werden.

Konfiguration in der UI

Im Panel "Settings" können die Input Edge Topics konfiguriert werden. Das Hinzufügen eines Input Edge Topics erfolgt durch die Eingabe des Namen des Topics, das verarbeitet werden soll, im Feld "New Topic". Entfernt werden können Topics mit einem Klick auf das Kreuz ✕.

Beispiel: Der zu verarbeitende Daten-Input wird von der „PollGroup a“ des OPC UA-Moduls bezogen. Abbildung 40 zeigt die nachfolgend beschriebene Konfiguration im OPC UA-Modul:

- Poll Groups: „PollGroup a“ mit Interval 1000ms, Name: “a”, Output Topic: “PollGroup a”
- Field Name: a und b

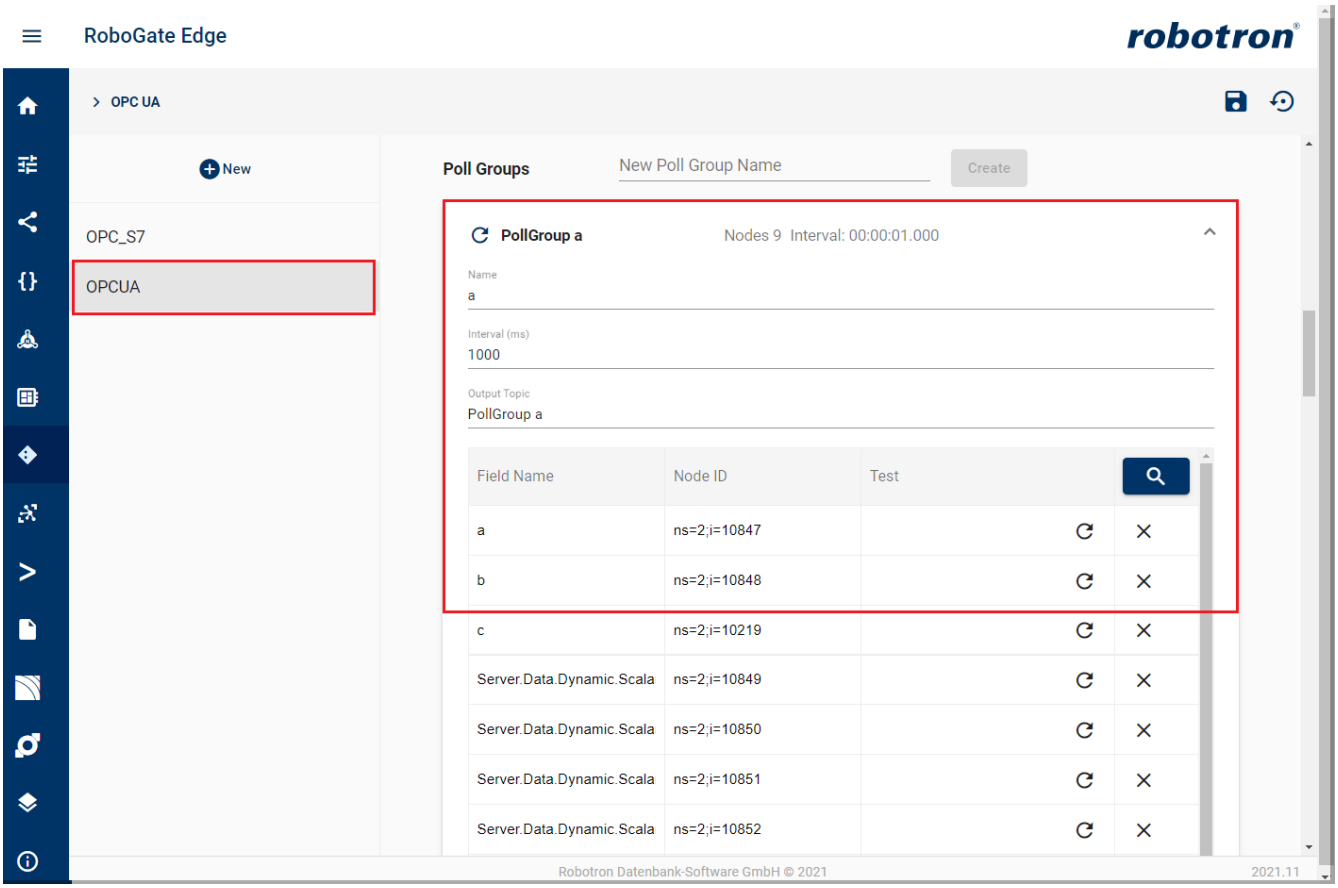


Abbildung 40. Konfiguration der "PollGroup a" im OPC UA-Modul

Die angelegte "PollGroup a" muss nun als Input Edge Topic im Stream Processor angelegt werden. Dies erfolgt durch Eingabe des Namens unter New Topic und einem Klick auf das Hinzufügen-Symbol "+" (Abbildung 41).

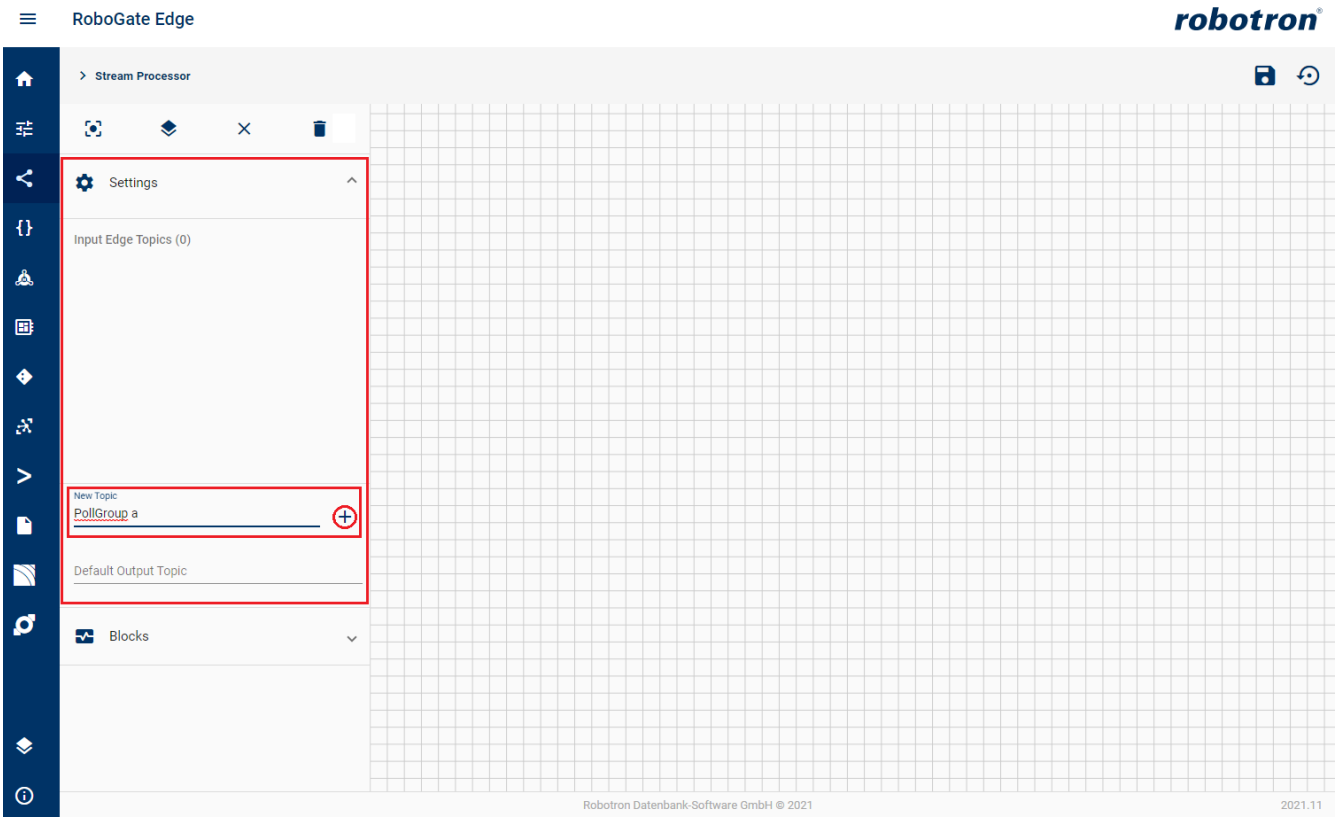


Abbildung 41. Konfiguration des Input Blocks

Anschließend muss ein Input-Block mit dem Filter angelegt werden. Hierzu klicken Sie auf das Element Input in der Auswahlliste des Panels "Blocks" und der Block erscheint in der Zeichenfläche des Stream Processor (Abbildung 42). Mit einem Klick auf den Stift im Input-Block auf der Zeichenfläche öffnet sich das Dialogfenster zum Konfigurieren des Input-Blocks. Im Beispiel erhält der Input Block den Namen "OPCUA" und filtert die Nachrichten nach Source ("OPCUA") und Scope ("^a$") weiter (Abbildung 43)

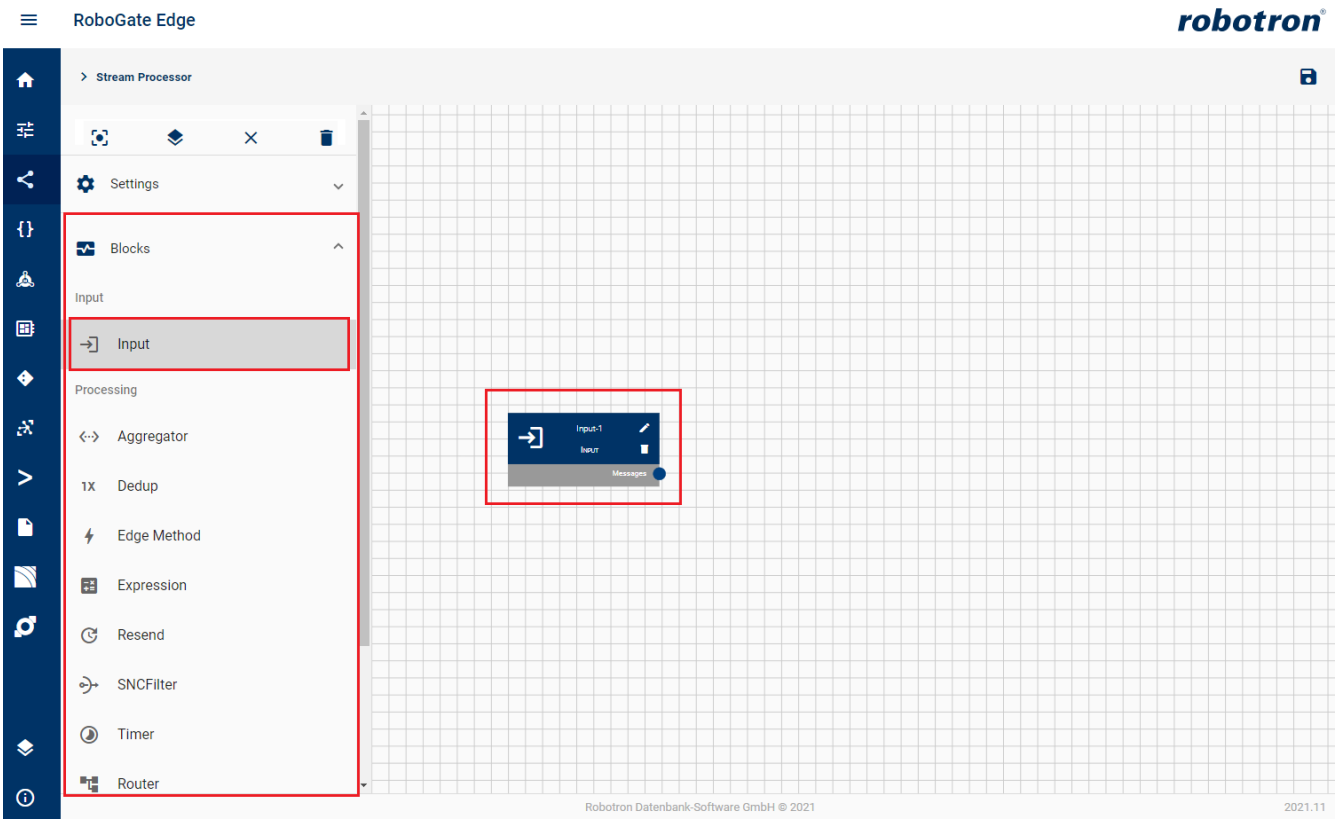


Abbildung 42. Erstellen des Input-Blocks

Abbildung 43. Konfiguration des Input-Blocks

Um den Input zu verarbeiten, müssen nun den Anforderungen entsprechend Processing-Blöcke angelegt werden.

Processing

Die Processing-Blöcke verarbeiten die einlaufenden Nachrichten und können diese anschließend über Output-Blöcke zurück an den Edge Broker des RoboGate Edge geben.

Folgende Typen von Processing-Blöcken stehen zur Verfügung:

- Aggregator
- Dedup
- Edge Method
- Expression
- Join
- Resend
- Router
- RuleEngine
- SNCFilter
- Timer

Die einzelnen Processing-Blöcke und ihre Konfiguration werden separat im Abschnitt StreamProcessor näher beschrieben.

Um Nachrichten mit unterschiedlichen Source- und Scope-Werten (d.h. aus unterschiedlichen Datenquellen) unabhängig voneinander und parallel zueinander zu verarbeiten, bauen die Processing-Blöcke in der Regel für jede Kombination von Source und Scope unabhängige Zustände (z.B. eine Timerinstanz auf dotnet Ebene oder einen Puffer) auf.

Die verfügbaren Typen von Processing-Blöcken mit ihren spezifischen Konfigurationsparametern werden in den nachfolgenden Abschnitten beschrieben.

Konfiguration in der UI

Um einen Processing-Block zu erstellen, wählen Sie das entsprechende Element im Panel "Blocks" unter dem Bereich "Processing" aus. Anschließend erscheint der Block auf der Zeichenfläche des Stream Processor. In diesem Beispiel wird der Aggregator-Block gewählt (Abbildung 44). Mit einem Klick auf den Stift öffnet sich das Dialogfenster zum Konfigurieren des Blocks. Nun lässt sich der Name des Blocks, im Beispiel "agg1", festlegen und die weiteren Parameter konfigurieren.

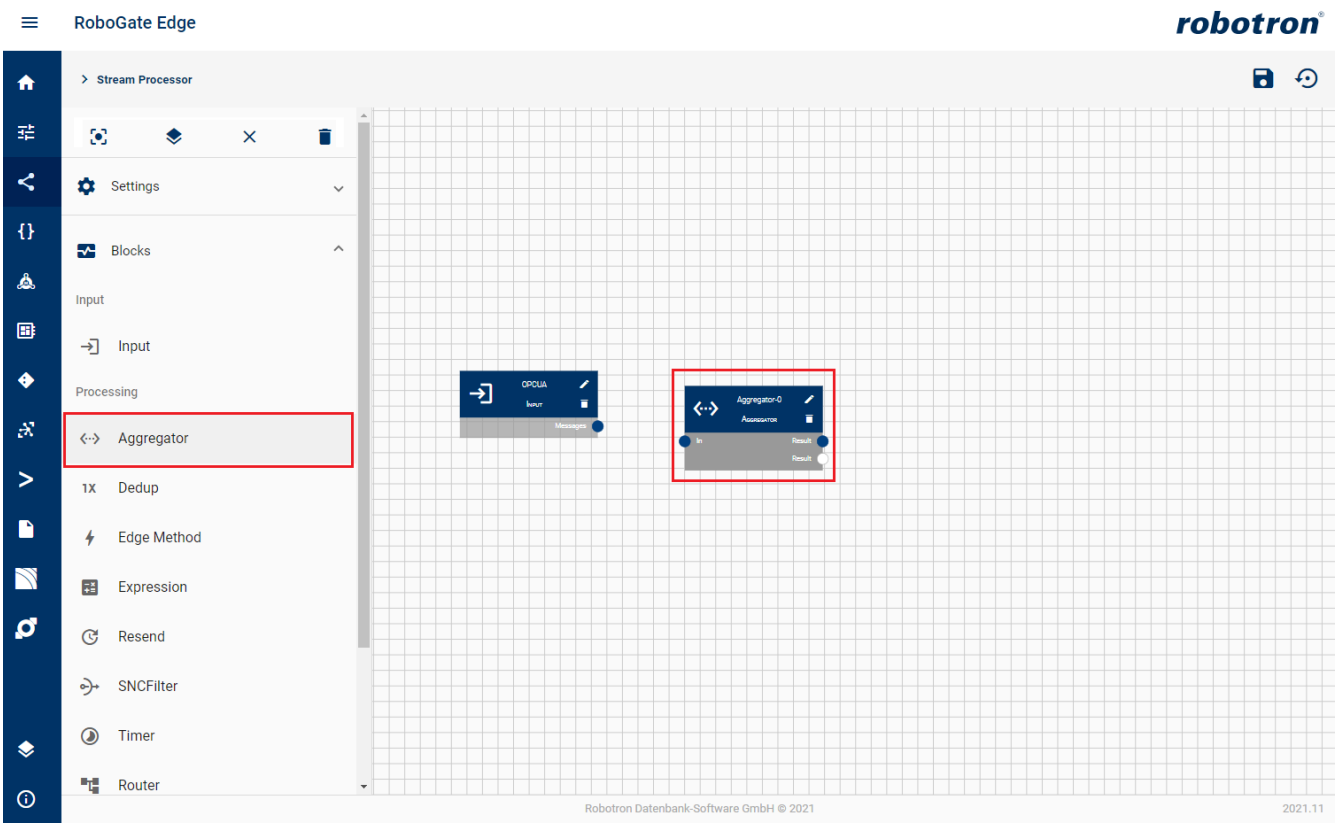


Abbildung 44. Erstellen eines Aggregator-Blocks

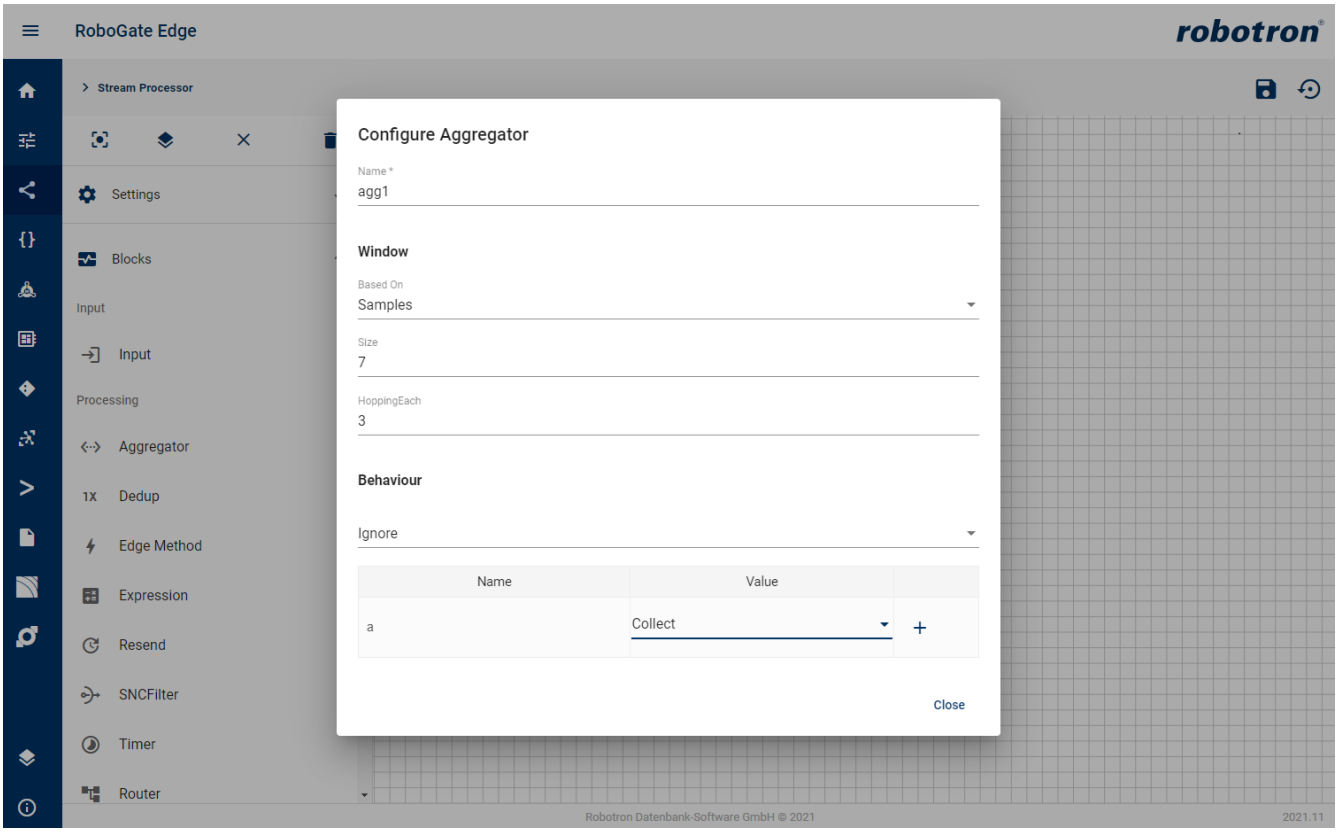


Abbildung 45. Konfigurieren eines Aggregator-Blocks

Damit die Nachrichten des Input Edge Topics weitergeleitet und verarbeitet können, muss eine Verbindung zwischen Input und Verarbeitung (Input Block und Processing Block) geschaffen werden. Hierzu klicken Sie auf den ausgehenden Knotenpunkt "Messages" des Input-Blocks und ziehen eine Verbindung zu einem Eingangsknotenpunkt am Processing-Block (Abbildung 46). Über

diese Verbindung laufen die Nachrichten aus Source und Scope entsprechend des konfigurierten Filters im Input-Block in den Aggregator-Block ein.

Auch Processing-Blöcke können untereinander über Eingangs- und Ausgabeknoten miteinander verbunden werden.

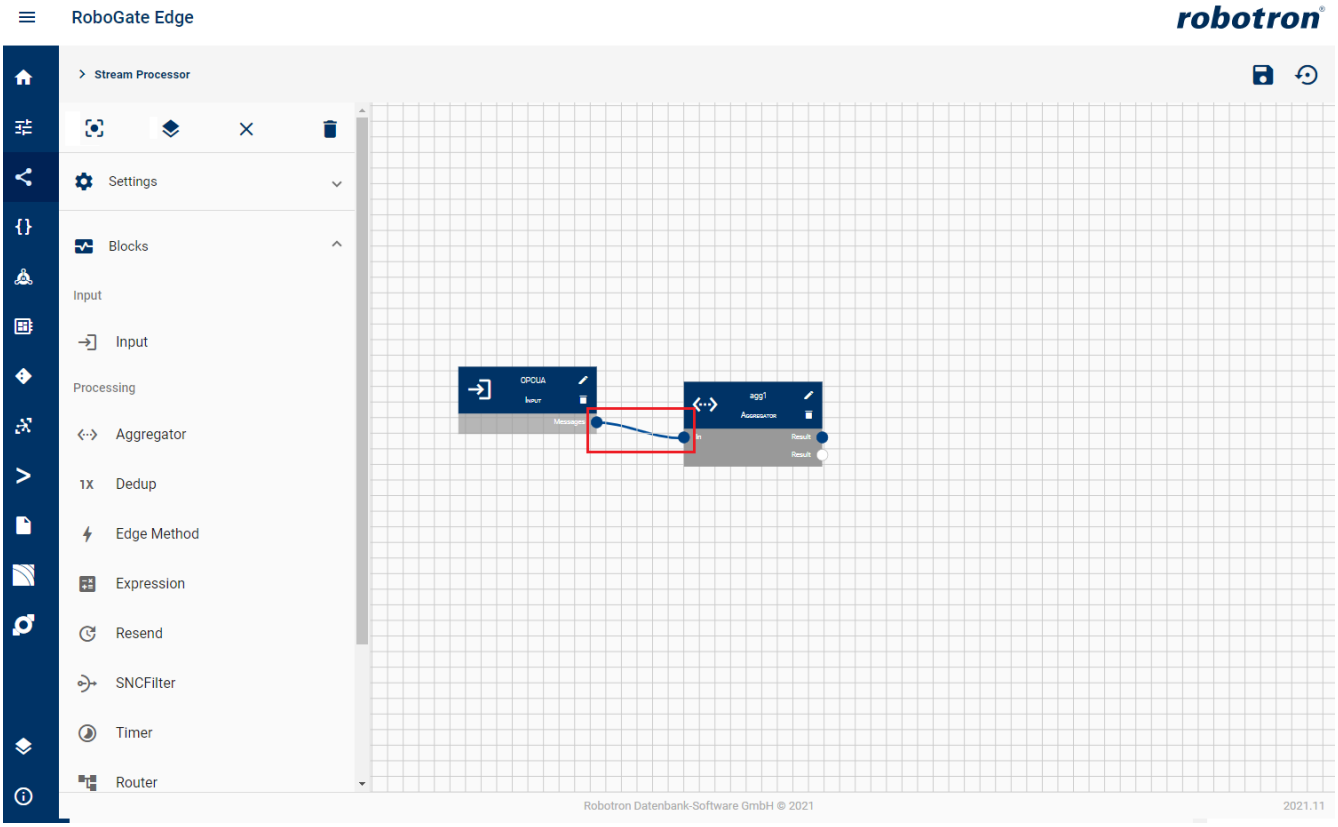


Abbildung 46. Verbindung von Input- und Aggregator-Block

Output

Um die verarbeiteten Nachrichten im Stream Processor in weiteren Modulen des RoboGate Edge zu verarbeiten, müssen diese an den Edge Broker zurückgegeben werden.

Hierzu können ein **Default Output Topic** und/oder mittels **Output Block** weitere Output Topics definiert werden. Das Default Output Topic ist optional. In das Modul einlaufende Nachrichten, die durch keine Filter-Konfiguration passen (in den Input-Blöcken), werden, falls konfiguriert, auf dieses Edge Broker Topic geleitet. Um eine Nachricht aus dem Stream Processor zurück an den Edge Broker zu geben, muss über einen Output-Block ein Output Topic konfiguriert werden. Dieses Topic kann zur Auswertung in einem Datenausgabemodul genutzt werden.

Es können mehrere Output-Blöcke angelegt werden.

Tabelle 19. Konfigurationsparameter Output-Block

Parameter	Beschreibung
Name	Name des konfigurierten Output-Blocks
Edge Topic	Name des Output Topic, um Output-Nachrichten auf die Edge Broker Topic(s) zu mappen

Parameter	Beschreibung
Source	Wenn angegeben, wird Source in der Nachricht überschrieben
Scope	Wenn angegeben, wird Scope in der Nachricht überschrieben

Konfiguration in der UI

Zum Erstellen eines Output-Blocks klicken Sie, analog zum Erstellen eines Input-Blocks auf das Element Output in der Auswahlliste des Panels "Blocks" im Bereich "Output". Daraufhin erscheint der erstellte Output-Block in der Zeichenfläche (Abbildung 47). Mit einem Klick aud den Stift öffnet sich das Dialogfenster zum Konfigurieren des Output-Blocks (Abbildung 48). Im Beispiel erhält der Output-Block den Namen "out" und die Spezifikation der weiteren Parameter.

Anschließend kann der zuvor erstelle Aggregator-Block mit dem Output-Block verbunden werden. Hierzu klicken Sie auf den ausgehenden Knotenpunkt "Result" des Aggregator-Blocks und ziehen die entstehende Verbindungslinie zum Knotenpunkt "Out" des Output-Blocks (Abbildung 49).

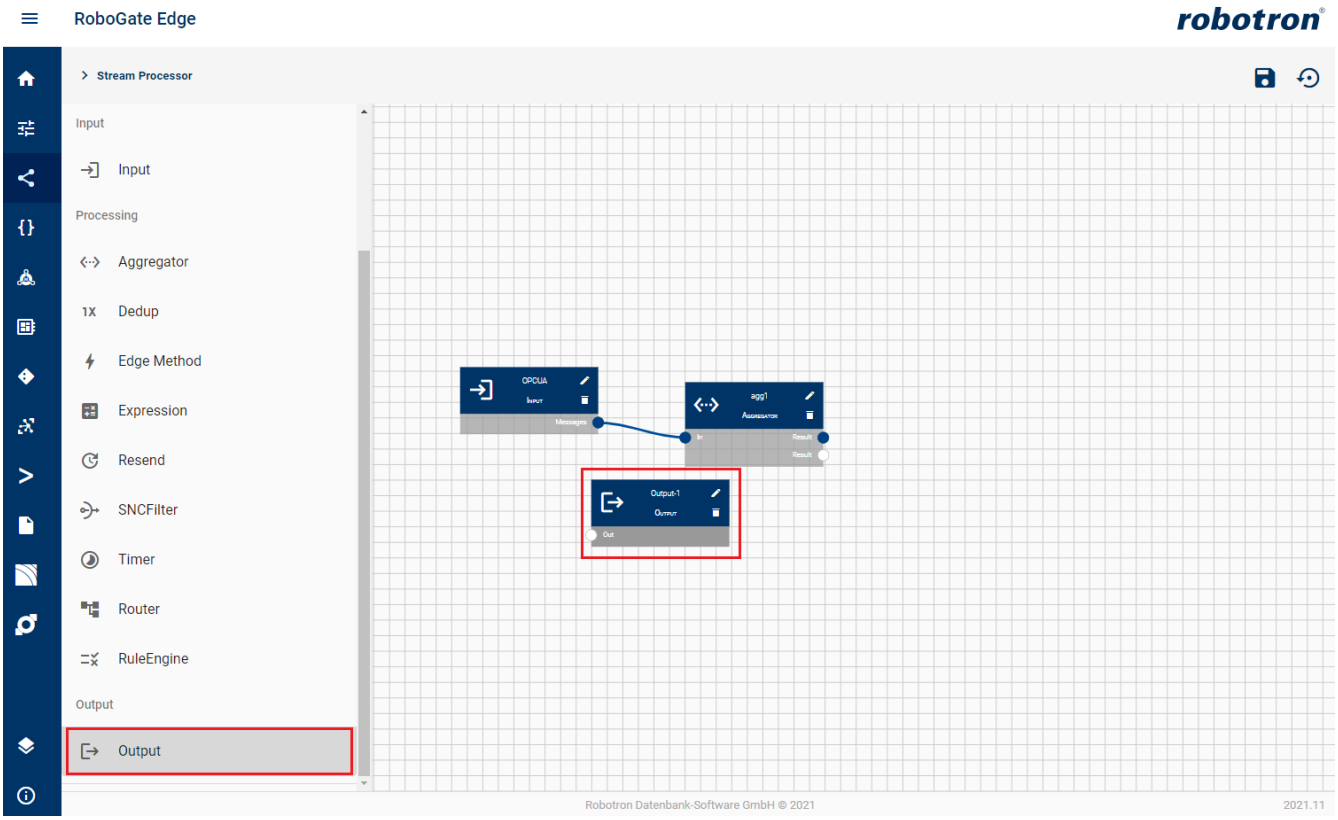


Abbildung 47. Erstellen eines Output-Blocks

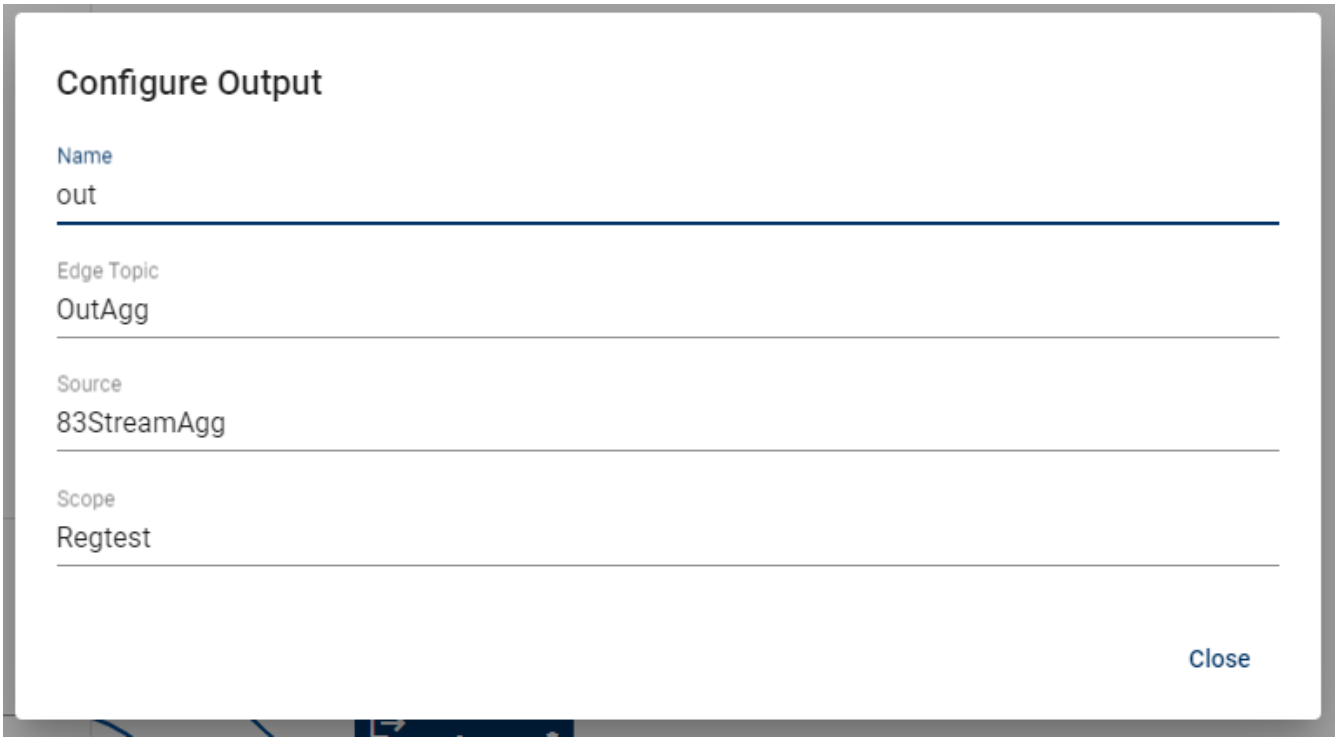


Abbildung 48. Konfiguration des Output-Blocks

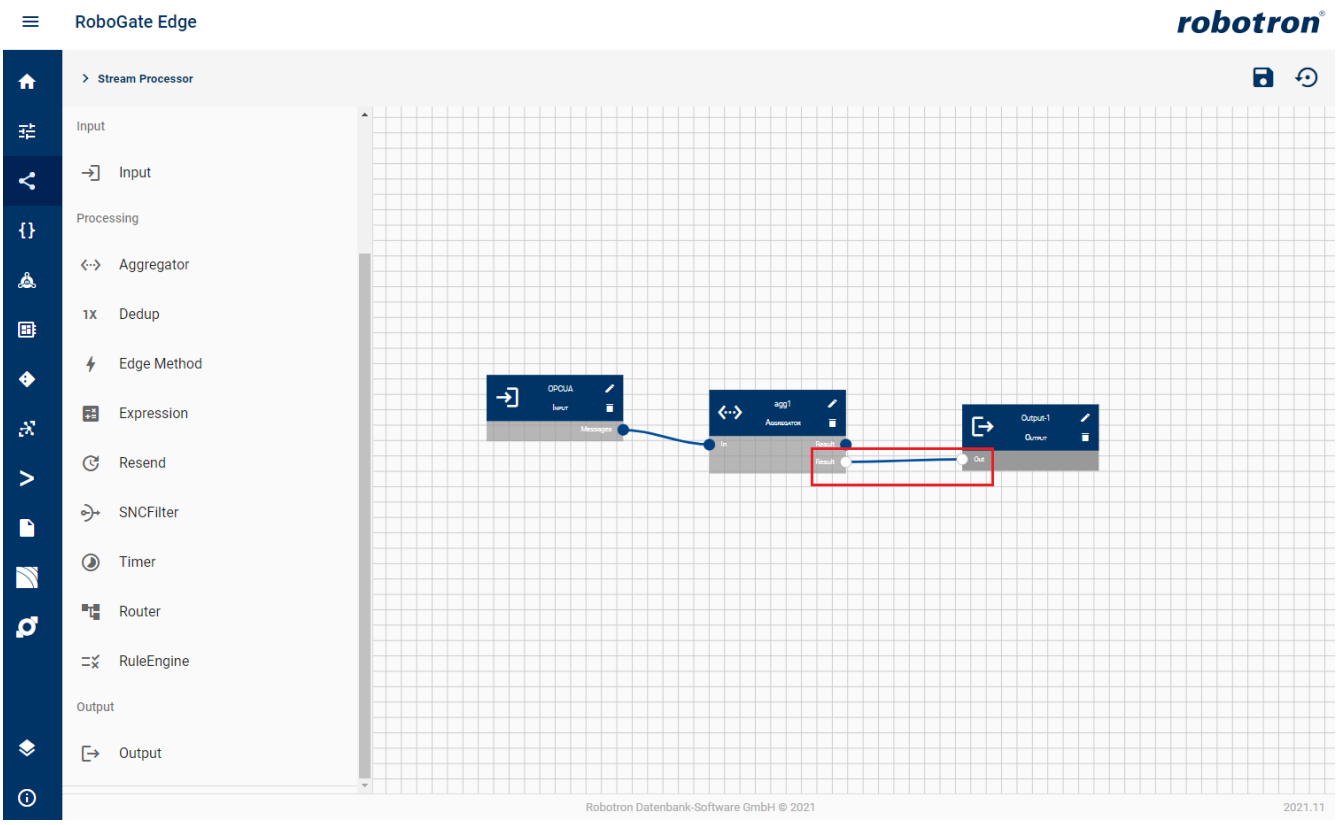


Abbildung 49. Verbindung von Aggregator- und Output-Block

4.2.2. Konfiguration zurücksetzen

Über die Schaltfläche in der horizontalen Menüleiste lässt sich die komplette Konfiguration des Moduls auf Werkseinstellungen zurücksetzen. Nach einem Klick auf die Schaltfläche folgt ein Bestätigungsdialog mit "Yes" und "No" zur Rückfrage ob der Vorgang wirklich ausgeführt werden soll.

> Damit wird auch die gesamte Historie auf dem RoboGate Edge Ebene gelöscht!

4.2.3. Aggregator

Funktionsbeschreibung und Anwendung

Der Aggregator-Block sammelt einlaufende Nachrichten. Die in Nachrichten enthaltenen Felder können entsprechend der Konfiguration verschiedenartig behandelt werden. Die Nachrichten werden in sich optional überlappenden Gruppen (Windows) basierend auf Zeit oder Anzahl zusammengefasst und verarbeitet.

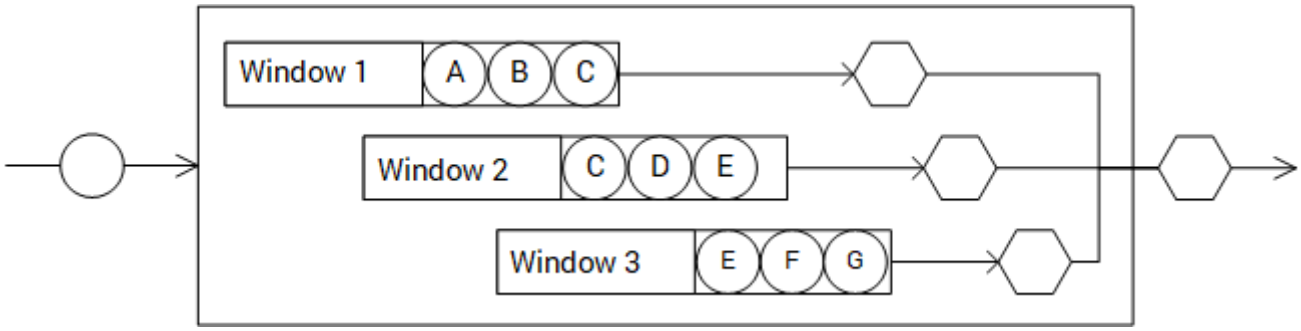


Abbildung 50. Schematische Darstellung Funktionsweise Aggregator-Blocks

Ein Anwendungsbeispiel ist eine Minimum/Maximum-Aggregation, um das aufkommende Datenvolumen zu reduzieren. Beispielsweise laufen Temperaturdaten der Außentemperatur im Sekundentakt ein, aber statt 60 Momentan-Werte zu übertragen, können für jede Minute nur der Minimum- und Maximum-Wert übertragen werden.

Die in der folgenden Tabelle erläuterten Parameter sind für die Konfiguration notwendig.

Tabelle 20. Konfigurationsparameter Aggregator-Block

Parameter	Beschreibung
Name	Name der Blockinstanz.
Into	Name der Blockinstanz des Folgeblocks an welchen Ausgabenachrichten weitergeleitet werden sollen.

Parameter	Beschreibung
Window - Based On	Auswahl des Window-Typ • **Sample**-basiertes Window: nutzt die Sample-Anzahl im Puffer, unabhängig von der Zeit. Bei einem (voll) gefülltem Windowpuffer wird mit jedem Einlaufen einer neuen Nachricht eine alte gelöscht. Bei einem gefüllten Window liegt deshalb immer eine gleiche Anzahl von Samples für nachfolgende Berechnungen vor. Das Triggern erfolgt ausschließlich anhand einlaufender Nachrichten. • **TimeMs**-basiertes Window: nutzt die Zeit in Millisekunden (timestamp) als Grundlage für das Verwerfen alter Samples aus dem Puffer. Es kann passieren, dass ganz unterschiedliche Sample-Anzahlen für eine Berechnung vorliegen. Das Fenster und die Aggregatsbildung agieren getriggert durch die Systemzeit und unabhängig von einlaufenden Nachrichten. **Die Eigenschaft timestamp ist für den Windowtyp TimeMs zwingend erforderlich!**
Window – Size	Die Windowgröße bestimmt den vorhandenen Puffer je nach Window-Typ sample-basiert (Wert als Anzahl) oder zeitbasiert (Wert in Millisekunden).
Window - HoppingEach	Häufigkeit einer Berechnung und Ausgabe je nach Window-Typ sample-basiert (Wert als Anzahl) oder zeitbasiert (Wert in Millisekunden). Ein hoher Wert führt seltener zu einer Ausführung und Ausgabe als ein kleiner.
Behaviour	Die verarbeitende Funktion kann für jedes Feld der einlaufenden Nachricht angegeben werden. Sollte eine ausgehende Nachricht keinen Zeitstempel (Key timestamp) besitzen, so wird dieser mit der aktuellen Zeit erzeugt.
Default	Default-Wert, der für alle Felder gilt, die nicht explizit angegeben sind.
Name	Feldname (Variablenname) im JSON der einlaufenden Nachricht

Parameter	Beschreibung
Value	Funktionsauswahl für das jeweilige Feld. Folgende Werttypen stehen zur Verfügung: • **ignore**: Daten werden verworfen und bei der Ausgabe nicht beachtet • **collect**: Sammeln der Values der einlaufenden Nachrichten des zu bewertenden „Zeitraumes“ in einem Array • **min**: minimaler Datenwert (erfordert numerische Values) • **max**: maximaler Datenwert (erfordert numerische Values) • **first**: zuerst eingelaufener Datenwert • **last**: zuletzt eingelaufener Datenwert • **sum**: Summe der eingelaufen Datenwerte (erfordert numerische Values) • **avg**: mathematisch Durchschnitt (erfordert numerische Values) • **median**: Median (erfordert numerische Values) • **stddev**: Standardabweichung (erfordert numerische Values) • **default**: legt Default-Funktion für alle nicht explizit genannten Felder fest

Beispielkonfiguration

Folgende Beispielkonfiguration ist in Abbildung 51 dargestellt:

- „a“ ist explizit konfiguriert und wird in einem Array gesammelt (Value: "Collect") und ausgegeben.
- „b“ wird explizit ignoriert (Value: "Ignore") und entfällt für die Ausgabenachricht.
- Da der Timestamp nicht konfiguriert ist, erhält die Default-Funktion die Konfiguration *first*.

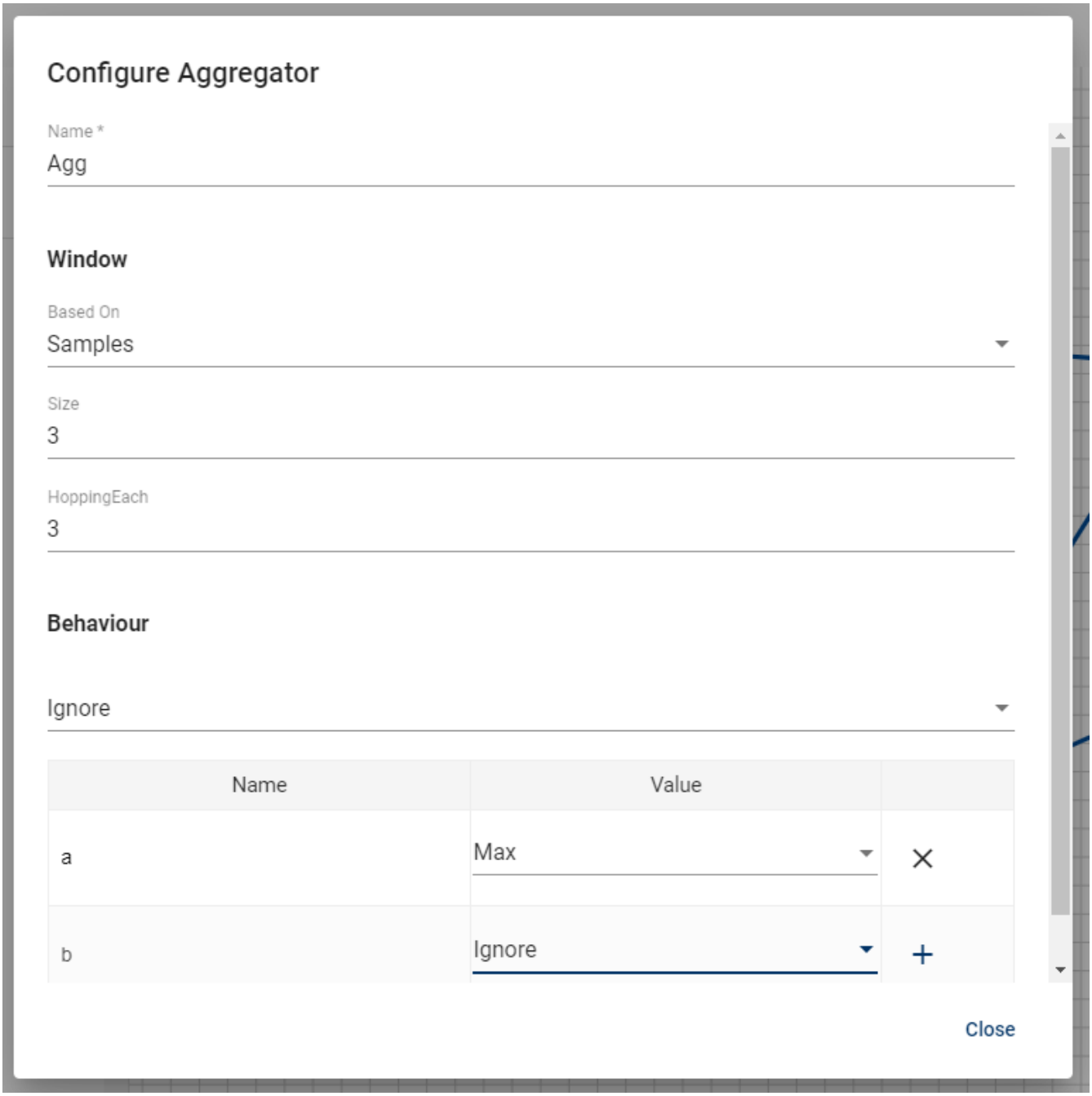


Abbildung 51. Konfiguration des Aggregator-Block

4.2.4. Dedup

Funktionsbeschreibung und Anwendung

Der Deduplicate-Block (Dedup) verwirft Nachrichten, die gleiche Informationen zur vorherigen Nachricht aufweisen. Die Prüfung kann mittels des Parameter `Watch` maskiert, d.h. eingeschränkt werden.

Der Block wird verwendet, um redundante Daten/Nachrichten auszufiltern.

Die in der folgenden Tabelle erläuterten Parameter sind in Abbildung 52 zu finden.

Tabelle 21. Konfigurationsparameter Dedup-Block

Parameter	Beschreibung
Name	Name der Blockinstanz.
Into	Name der Blockinstanz des Folgeblocks an welchen Ausgabenachrichten weitergeleitet werden sollen.
Overwrite Header Scope with	string, Name für neue Scope Bezeichnung (optional). Dient dem Überschreiben des Header-Feldes Scope. Die Option ist inaktiv, wenn der String nicht vorhanden oder leer ist.
Watch	string oder string Array, Watch ist der Ausdruck für die Auswahl des Prüfkriteriums in der Nachricht. Wenn aufeinanderfolgende Nachrichten an der Stelle/den Stellen „x“ gleich sind, wird die Nachricht verworfen. Für den Zugriff auf Nachrichtenkeys mit einem Punkt „.“, bspw. Key Data.Value, ist der Zugriff über die Syntax ['Data.Value'] erforderlich.

Beispielkonfiguration

Die erläuterten Parameter sind in Abbildung 52 beispielhaft angewendet.

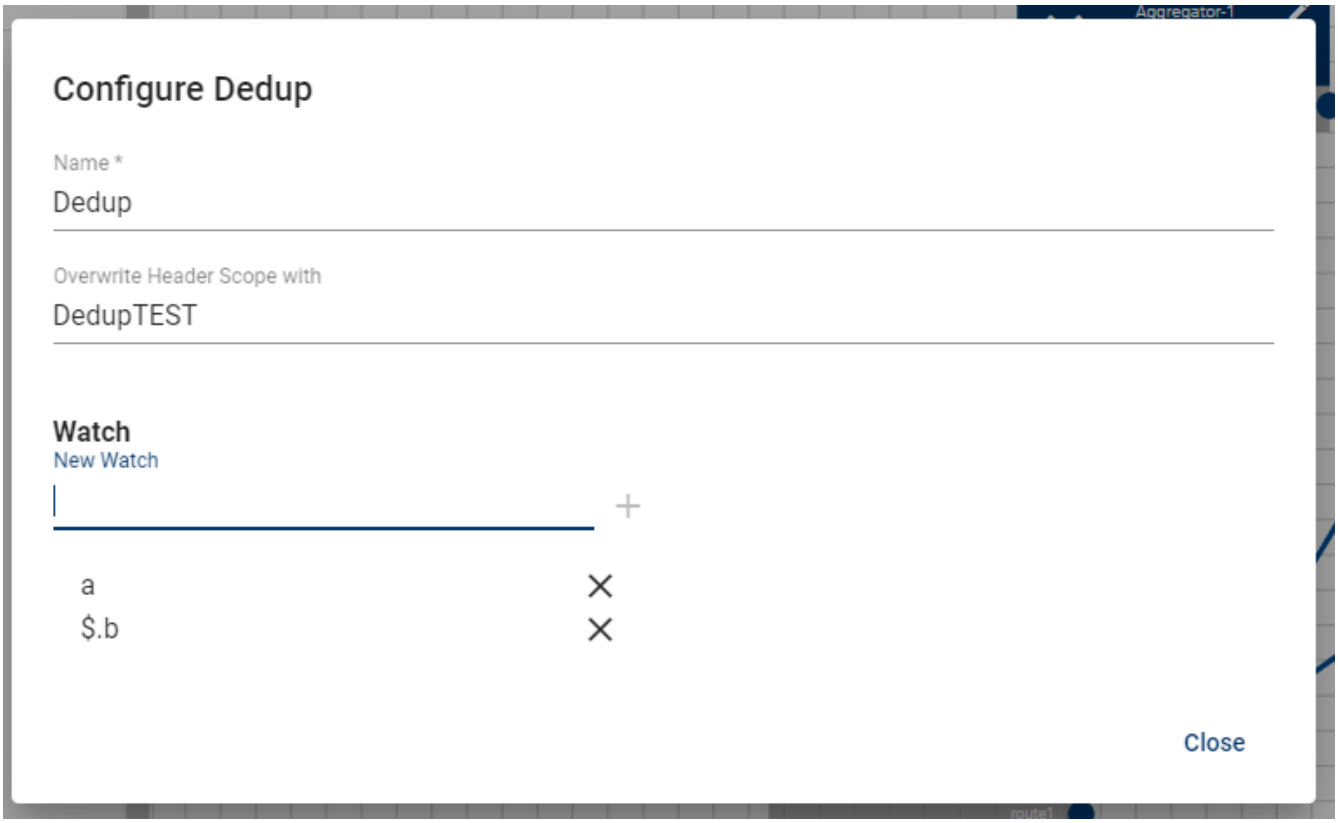


Abbildung 52. Konfiguration des Dedup-Block

Konfiguration in JSON

```
"Dedup_Block": {
    "type": "Dedup",
    "Watch": [
```

```
      "statuscode"
    ],
    "Settings": {
      "OverwriteHeaderScopeWith": ""
    },
    "into": [
      "outDedup"
    ]
}
```

4.2.5. Edge Method

Funktionsbeschreibung und Anwendung

Der Edge Method-Block (Edge Method Invocation) ruft eine Edge Method auf und leitet deren Ergebnis in Abhängigkeit des Erfolgs an einen weiteren Processing- oder Output-Block weiter.

Tabelle 22. Konfigurationsparameter Edge Method-Block

Parameter	Beschreibung
Name	Name der Blockinstanz.
Method	Name der Methode, die aufgerufen werden soll.
Parameter	Liste von konstanten Parametern zum Aufrufen der Methode

Beispiel: Aktualisieren eines OPC UA Nodes

Im Beispiel wird ein OPC UA Node gesetzt, sobald eine Nachricht den Block auslöst.

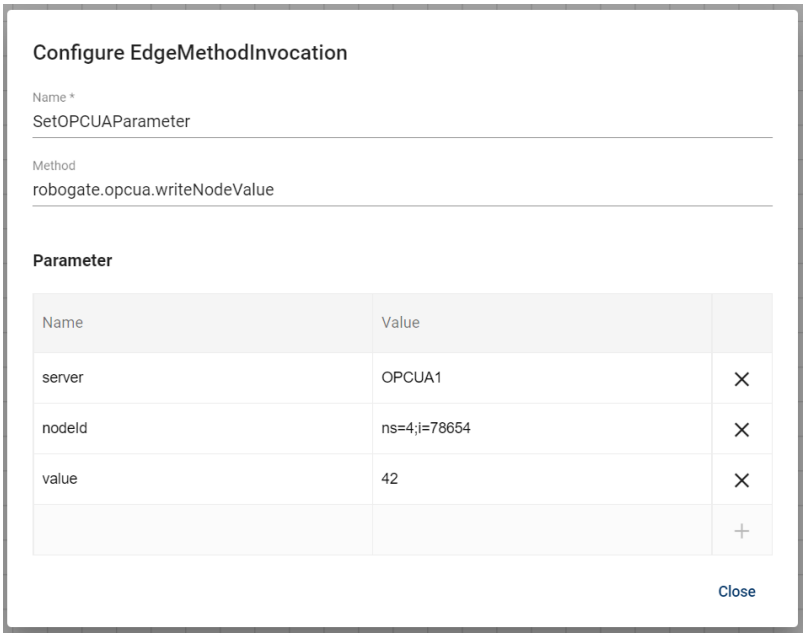


Abbildung 53. Konfiguration des Edge Method-Block

Häufig verwendete Methoden

Im Folgenden werden Methoden des RoboGate Edge beschrieben, welche sich für Verwendung

zusammen mit dem Edge Method-Block eignen.

robogate.opcua.writeNodeValue

Methode des OPC UA-Moduls. Setzt einen Node eines konfigurierten OPC UA Servers auf einen konstanten Wert.

Parameter	Beschreibung
server	Name des konfigurierten Servers
nodeId	Node ID des zu setzenden Nodes
value	Wert des Nodes. Dieser kann auch dynamisch aus einem Topic erfolgen. Hierfür muss vor die Variable ein $-Zeichen gesetzet werden. Zum Beispiel $TOPIC_NAME.

robogate.opcua.readNodeValue

Methode des OPC UA-Moduls. Liest den Wert eines Nodes eines konfigurierten OPC UA Servers.

Parameter	Beschreibung
server	Name des konfigurierten Servers
nodeId	Node ID des zu setzenden Nodes
value	Wert des Nodes

Antwortnachricht

Property	Beschreibung
val	Ausgelesener Wert

robogate.rfc1006.readSignal

Methode des RFC1006-Moduls. Liest einen Wert aus einem Speicherbereich einer konfigurierten PLC.

Parameter	Beschreibung
plc	Name der konfigurierten PLC
dataType	Speichertyp auf PLC • DataBlock • Memory • Timer • Counter • Input • Output

Parameter	Beschreibung
dataBlock	Name des DataBlocks (Bei DataType DataBlock)
startByte	StartByte des auszulesendes Wertes im Speicherbereich
count	Anzahl der zu lesenden Bytes des Speicherbereiches
bit	Zu lesendes Bit innerhalb des zu lesenden Bytes (bei VarType = Bit)
varType	Datentyp in den der gelesene Wert konvertiert werden soll • Bit (boolean) • Byte • Word • DWord • Int • DInt • Real • String • StringEx • Timer • Counter • DateTime • DateTimeLong

Antwortnachricht

Property	Beschreibung
value	Ausgelesener Wert

robogate.rfc1006.writeSignal

Methode des RFC1006-Moduls. Schreibt einen Wert in einen Speicherbereich einer konfigurierten PLC.

Parameter	Beschreibung
plc	Name der konfigurierten PLC

Parameter	Beschreibung
dataType	Speichertyp auf PLC • DataBlock • Memory • Timer • Counter • Input • Output
dataBlock	ID des DataBlocks
startByte	Start Byte des zu lesendes Wertes im Speicherbereich
varType	Datentyp in den der gelesene Wert konvertiert werden soll • Bit (boolean) • Byte • Word • DWord • Int • DInt • Real • String • StringEx • Timer • Counter • DateTime • DateTimeLong
value	Zu schreibender Wert

modbus-testread

Methode des Modbus-Moduls. Liest einen Wert aus einem konfigurierten Modbus Slave.

Parameter	Beschreibung
serverInstance	Name des konfigurierten Modbus
slaveId	ID des auszulesenden Slaves im Modbus

functionCode	Modbus Function Code • 1: Read Coil (Boolean) • 2: Read Input (Boolean) • 3: Read Holding Register (2 byte) • 4: Read Input Register (2 byte)
registerStartAddress	Startadresse des zu lesenden Speicherbereichs
registerLength	Länge des zu lesenden Speicherbereichs in Words (2 byte) bei FunctionCodes 3 und 4
type	Datentyp in den der augelesene Wert konvertiert werden soll • UInt16 (16 bit Ganzzahl ohne Vorzeichen, Länge: 1 Word) • Int16 (16 bit Ganzzahl mit Vorzeichen, Länge: 1 Word) • UInt32 (32 bit Ganzzahl ohne Vorzeichen, Länge: 2 Word) • Int32 (32 bit Ganzzahl mit Vorzeichen, Länge: 2 Word) • UInt64 (64 bit Ganzzahl ohne Vorzeichen, Länge: 4 Word) • Int64 (64 bit Ganzzahl mit Vorzeichen, Länge: 4 Word) • Float (Einfache Fließkommazahl, Länge: 2 Word) • Double (Genauere Fließkommazahl, Länge: 4 Word) • RawHex (Rohdaten des gelesenen Speicherbereichs als Hexadezimalstring) Bei FunctionCodes 1 und 2: Bool, Länge 1
valueGain	Konstanter Faktor des gelesenen Wertes, um richtigen Wert zu erhalten (Default: 1.0)
valueOffset	Konstante Verschiebung des gelesenen Wertes, um richtigen Wert zu erhalten (Default: 0.0)
isLittleEndian	Wenn true, interpretiert beim Lesen des Wertes das erste gelesende Byte als kleinstwertiges Byte. (Default: false ⇒ big-endian)

Antwortnachricht

Property	Beschreibung
resultCode	Ergebnis Code ("Ok" oder "Error")
data.Variable	Ausgelesener Wert

robogate.set_led

Diese Methode ist nur bei RoboGate Devices powered by Turck verfügbar. Konfiguriert das Blinkverhalten einer HMI LED am Gerät.

Parameter	Beschreibung
name	Name der LED • red, green, blue : Obere LED • ered, egreen: Mittlere LED • bred, bgreen: Untere LED
modus	Blink Modus • off: Dauerhaft aus • on: Dauerhaft an • flash: Blinkt in der Frequenz von frequence • heartbeat: Blinkt im Rhythmus eines Herzschlags

robogate.opcua.triggerReadGroup

Diese Methode ermöglicht ein erneuetes Auslesen einer OPC-UA Pollgruppe. Dies kann zum Beispiel verwendet werden, um nur bei bestimmten Zuständen eine OPC-UA Gruppe auszulesen. So lässt sich unter anderem an der Rechnerauslastung sparen.

Parameter	Beschreibung
server	Name des konfigurierten Servers
name	Name der zu triggerenden OPC UA Gruppe

Achtung: das Modul selbst triggert nur die angegebene Gruppe. Selbst gibt das Modul nur Statusinformationen aus ob die Aktion funktioniert hat. Siehe auch die Abbildung.

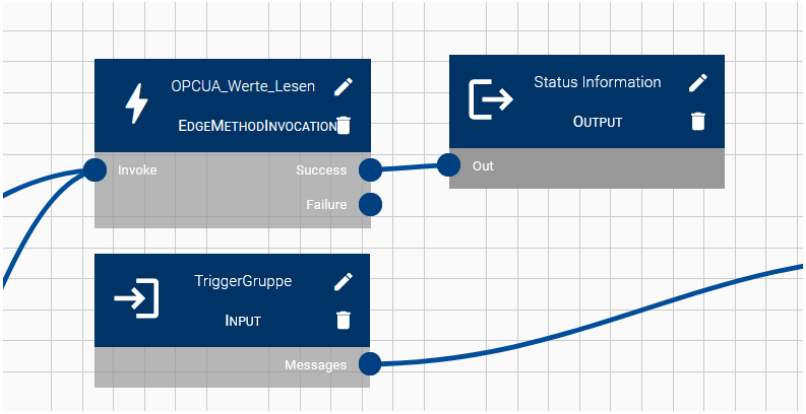


Abbildung 54. OPC UA Gruppen Trigger in typischer Verwendung

Wenn die Gruppe ausschließlich auf den Trigger reagieren soll und nicht auf übliches Polling, muss das Polling der jeweiligen OPCUA Gruppe auf 0 Sekunden gestellt werden.

robogate.opcua.callMethod

Mit der callMethode können OPC-UA Methoden aufgerufen werden. Folgende Parameter sind dabei mit anzugeben:

Parameter	Beschreibung
server	Name des konfigurierten Servers
objectNodeid	NodeId zum verwendenten Objekt
methodNodeid	NodeId zur benutzten Methode
inputParameter	Json mit den Eingabeparametern ("<PARAMETER NAME>": <VALUE>, "<PARAMETER NAME>": <VALUE>, ...)

robogate.opcua.<OpuUA Methode>

Diese Methode ähnelt der callMethod Methode, hat allerdings den Vorteil, dass objectNodeid und methodNodeid nicht in der EdgeMethode mit angegeben werden müssen. Diese wurden hier bereits im EdgeMapping des OPC-UA Modules definiert. Der Methodenname zum Aufrufen wird ebenfalls im OPC-UA Modul definiert. Der Inputparameter (im Beispiel x) muss hier dem Namen auf dem OPC-UA Server entsprechen. Siehe dazu die nächste Abbildung.

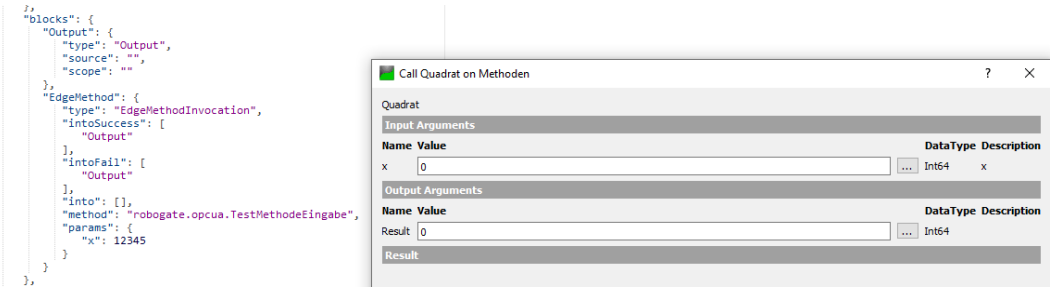


Abbildung 55. OPC UA Methoden Aufruf und das Korrespondant in OPC UA Expert

4.2.6. Expression

Funktionsbeschreibung und Anwendung

Der Expression Evaluator-Block (ExpEval) kann Berechnungen und sonstige mathematische oder logische Verknüpfungen auf Grundlage einlaufender Nachrichten durchführen. Der Processing-Block hat Zugriff auf einlaufende Nachrichten-Payloads.

Die folgenden Skizze veranschaulicht die Funktionsweise.

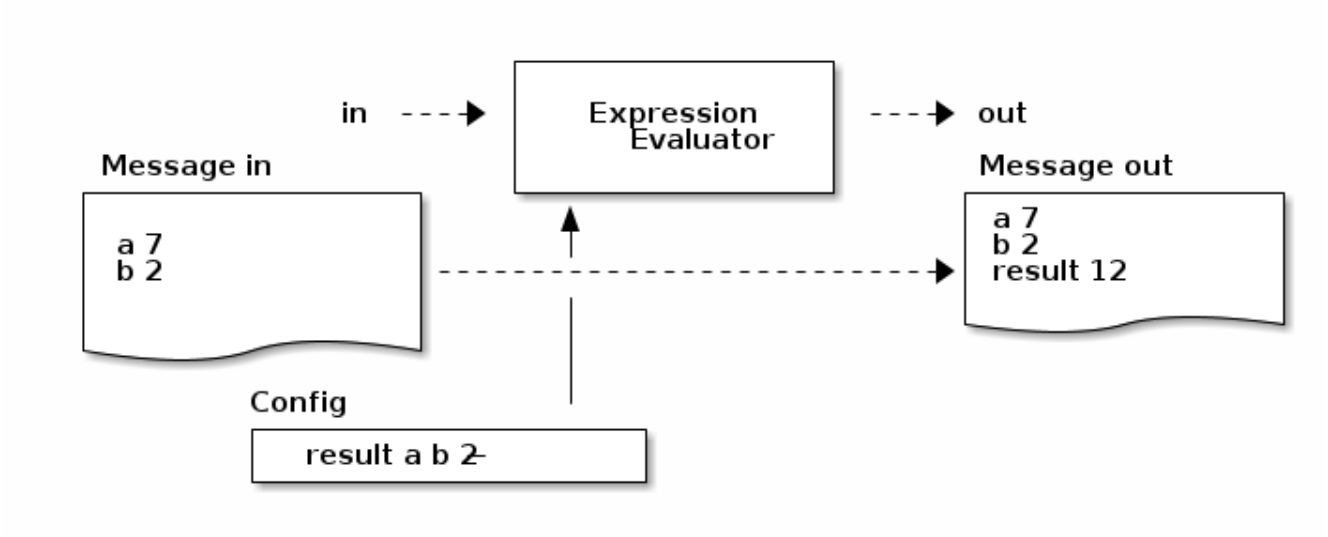


Abbildung 56. Funktionsweise Expression Evaluator

Im Beispiel der Skizze wird laut Konfiguration die Berechnung eines Wertes `result` durchgeführt. Dafür wird auf die einlaufenden Datenfelder `a` und `b` zugegriffen. Nach Verarbeitung wird die einlaufende Nachricht mit dem Wert `result` erweitert und ausgegeben.

In der folgenden Tabelle werden die Konfigurationsparameter erläutert.

Tabelle 23. Konfigurationsparameter Expression-Block

Parameter	Beschreibung
Name	Name der Blockinstanz.
Into	Name der Blockinstanz des Folgeblocks an welchen Ausgabenachrichten weitergeleitet werden sollen.
Expressions - Name	Name des Ergebnis (Key) für einen angegebenen Ausdruck.
Value	string, Auszuwertender Ausdruck. Die Syntax ist unter Section 4.2.6.2 beschrieben.
Preserve input properties	optional, boolean, default=false, Bei Wert *true* bleiben die einlaufenden Nachrichten-Properties in den ausgehenden Nachrichten erhalten.
Results in object	optional, string, Wenn an dieser Stelle etwas angegeben wird, werden Ergebnisse in einem Unterobjekt der Ausgangsnachricht eingefügt.

Ausdrücke

Folgende Operatoren und Ausdrücke stehen zur Verfügung und können ausgewertet werden:

- mathematische Verknüpfungen mit den Operatoren `+, -, *, /`
- Modulo mit Operator `%`
- mathematische Funktionen aus .NET Math. Bspw. `Round(..), Abs(..), Pow(..), Sqrt(..), ..`
- binäre Verknüpfungen mit den Operatoren `| , &, ^, ~`
- logische Verknüpfungen mit den Operatoren `||, &&, or, and, !, not`
- mathematische Vergleiche mit den Operatoren `<, >, <=, >=, ==, !=`

Neben numerischen Konstanten (wie `12` oder `7.91`) oder booleschen `true`/`false` kann auf die Properties der einlaufenden Nachricht per JSON Key zugegriffen werden. Die folgende Tabelle zeigt einige Beispiele.

Tabelle 24. Beispiel-Expressions

Expression (JSON Value)	Variables (aus einlaufender Nachricht)	Ergebnis (JSON Value)
"1.1*6"		6.6
"20 % 10"		0
"not true"		false
"true && true"		true
"true || false"		true
"not(2>8)"		true
"foo*bar"	foo:20; bar:4	80
"foo-bar"	foo:20; bar:4	16
"foo/bar"	foo:20; bar:4	5
"foo+bar"	foo:20; bar:4	24
"foo%bar"	foo:20; bar:4	0
"foo-(bar * 100)"	foo:20; bar:4	-380

Zu beachten ist, dass dieser Processing-Block über keinen Zustand verfügt. Jede Berechnung erfolgt anhand der Konfiguration und der einlaufenden Nachrichten!

.Net Math Operations Syntax: Auf die Funktionen aus dotnet math kann folgendermaßen zugegriffen werden:

- Potenzieren: `y = x^a` ⇒ "y": "Pow(x,a)"
- Runden: `y = x (auf zwei Nachkommastellen)` ⇒ "y": "Round(x,2)"
- Quadratwurzel: `y = sqrt(x)` ⇒ "y": "Sqrt(x)"
- Abrunden auf Ganzzahl: x ⇒ "y": "Floor(x)"

- Aufrunden auf Ganzzahl: x ⇒ "y": "Ceiling(x)"
- Betrag einer Zahl: x ⇒ "y": "Abs(x)"

Weitere Details sind in der Dokumentation der Math Klasse .Net Core zu finden: https://docs.microsoft.com/de-de/dotnet/api/system.math.sqrt?view=netcore-3.1

Benutzung von regulären Ausdrücken (Regex)

Mit Hilfe regulärer Ausdrücke können komplexere Suchmuster erstellt werden. Beispielsweise nicht nur eine Zahl (z.B. 42), sondern auf ein ganzes Intervall (z.B. 00-99). Die Syntax für Regular Expression ist: `RegEx(<value>,'<pattern>')`. Dabei gelten folgende Sonderregelen:

- Mit Hochkomma (') um den Value kann ein statischer Wert eingetragen werden
- Ohne Hochkomma (') wird der Wert aus den Parametern (Header & Body & Kontext) geholt

Für eine Expression die einen leeren String enthält, lautet die Ausgabe immer `False`. Beispiel:

```
"HeaderScopeContainsSample" : "RegEx(header_scope, '.Sample.')"
```

Konstanten

Der Zugriff auf folgende Konstanten ist möglich:

- `PI` ⇒ Kreiszahl π
- `EULER` ⇒ Eulerscher Zahl e, Basis des natürlichen Logarithmus
- `ONE` ⇒ ganze Zahl 1

Umrechnung von Zeiten

Für die Konvertierung von Zeiten stehen mehrere Funktionen zur Verfügung:

Funktion	Beschreibung
DateTimeNowUtc()	Gibt die aktuelle Zeit des Robogates im ISO Format aus.
ToDateTime(value,format)	Berechne aus Unixepochtime, einem String oder OLE-Zeitstempel ein Objekt im Datumsformat, Datetime genannt.
DateTimeToString(value:datetime,format:optional string)	Umkehrfunktion zu ToDateTime wo ein Objekt im Datumsformat zu einem String berechnet wird.
DateTimeToUnixEpoch(value:datetime)	Berechne aus der Datetime die Anzahl der Sekunden seit dem 1.1.1970 0:00 UTC.
DateTimeToNumber(value:datetime)	Berechnet die Anzahl der Tage vor oder nach Mitternacht, dem 30. Dezember 1899. Die Bruchkomponente enstpricht der Zeit an diesem Tag dividiert durch 24.

Das Format ist hier die übliche Schreibweise mit den Identifikatoren. Beispiel: yyyy-MM-ddTHH:mm:ss.FFFFFFFK bzw. hier ein konkretes Datum wäre 2023-02-08T17:05:54.355023Z. Für genauere Details siehe https://learn.microsoft.com/de-de/dotnet/standard/base-types/custom-date-and-time-format-strings.

Beispiel für Zeitenumrechnung

Im folgendem Beispiel wird die Differenz zwischen einem Zeitstring und der aktuellen Zeit berechnet:

```
"Restzeit": "DateTimeToNumber(ToDateTime('2023-02-08T17:05:54.355023Z')) - DateTimeToNumber(DateTimeNowUtc())"
```

Aus dem String 2023-02-08T17:05:54.355023Z wird zunächst die Datetime berechnet. Anschließend wird die Datetime in eine Zahl formatiert und von der aktuellen Zeit im Zahlenformat (DateTimeNowUtc) abgezogen. Dies kann zum Beispiel benutzt werden, um zu prüfen ob der angegebene Zeitpunkt in der Zukunft (Restzeit>0) oder in der Vergangenheit liegt (Restzeit<0).

Beispielkonfiguration

Die Berechnung im Beispiel in Abbildung 57, ergibt sich aus dem Feld „a“ einer einlaufenden Nachricht und Addition mit einem konstanten Wert 1. Das Ergebnis wird in dem Feld „aPlus1“ abgelegt. Im Weiteren sollen alle vorhandenen Felder auf der Eingangsseite erhalten bleiben (→ *Preserve input properties: true*). Das Ergebnis soll direkt in das Top Level JSON-Objekt eingefügt werden (→ *Results in object: null*).

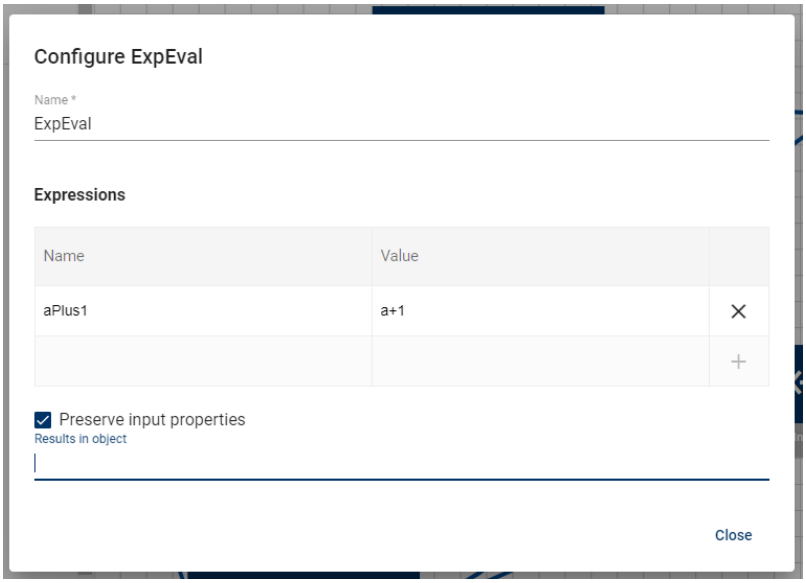


Abbildung 57. Konfiguration des Expression-Block

4.2.7. Join

Funktionsbeschreibung und Anwendung

Der Join-Block fügt die Nachrichten mehrerer Kanäle (Topics) zusammen. Im Block erfolgt keine getrennte Verarbeitung anhand Source und Scope Werten.

Tabelle 25. Konfigurationsparameter Join-Block

Parameter	Beschreibung
Main	Primärer Datenstrom
Meta	Metadatenströme mit konfigurierbarer Anzahl. Speichert die zuletzt eingetroffene Nachricht durch Überschreiben. Initial ist eine leere Nachricht gespeichert.
Modus	Modusauswahl für die Behandlung von JSON-Arrays. Wert ist optional, default = "Replace". Beispiel Arrays: A_Array: [0,1,2,3], B_Array: [2,3,4]. A-Array geht in Main, B_Array geht in ersten Metainput. • Concat: Arrays verketten. Ergebnis: [0,1,2,3,2,3,4] • Union: Arrays vereinigen, wobei bereits vorhandene Elemente übersprungen werden. Ergebnis: [0,1,2,3,4] • Replace: Ersetzen aller Array-Elemente. Ergebnis: [2,3,4] • Merge: Array-Elemente zusammenführen, abgestimmt auf den Index. Das Main-Array gilt als Grundlage, die Meta-Arrays werden anschließend Index für Index ersetzt. Falls die Länge beider Gleich ist, entspricht dies einem Replace. Ergebnis: [2,3,4,3]

Trifft eine Nachricht im Main Input ein, werden die aktuell gespeicherten Nachrichten der Meta Inputs nacheinander auf die Main-Nachricht gepatcht. D.h. Nachrichten an höheren Meta Inputs überschreiben ggf. Werte aus niedrigen Meta Inputs.

Eine neue Nachricht an einem Meta Input aktualisiert die derzeit gespeicherte, triggert aber keine neue Result-Nachricht. Der timestamp wird nicht gespeichert. Die neue Nachricht kann auch leer sein. Der timestamp aus der Nachricht am Main Input bleibt für die resultierende Nachricht erhalten.

Beispielkonfiguration

Im Beispiel werden Nachrichten über die konfigurierten Topics aus dem Modbus- und OPC-Modul (Input-1, Input-2, Input-4) im Join-Block zusammengeführt und als Metadaten dem Main Input aus dem RuleEngine-Block angehangen.

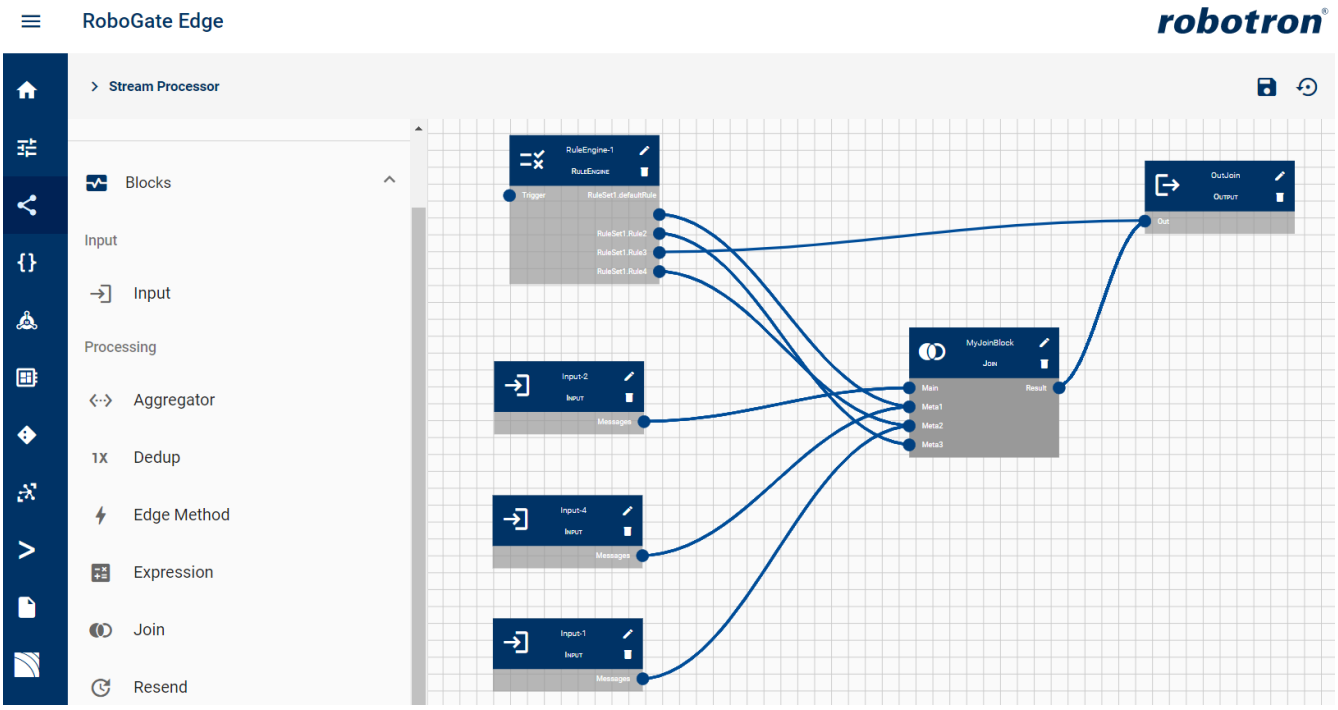


Abbildung 58. Konfiguration der Stream Processor

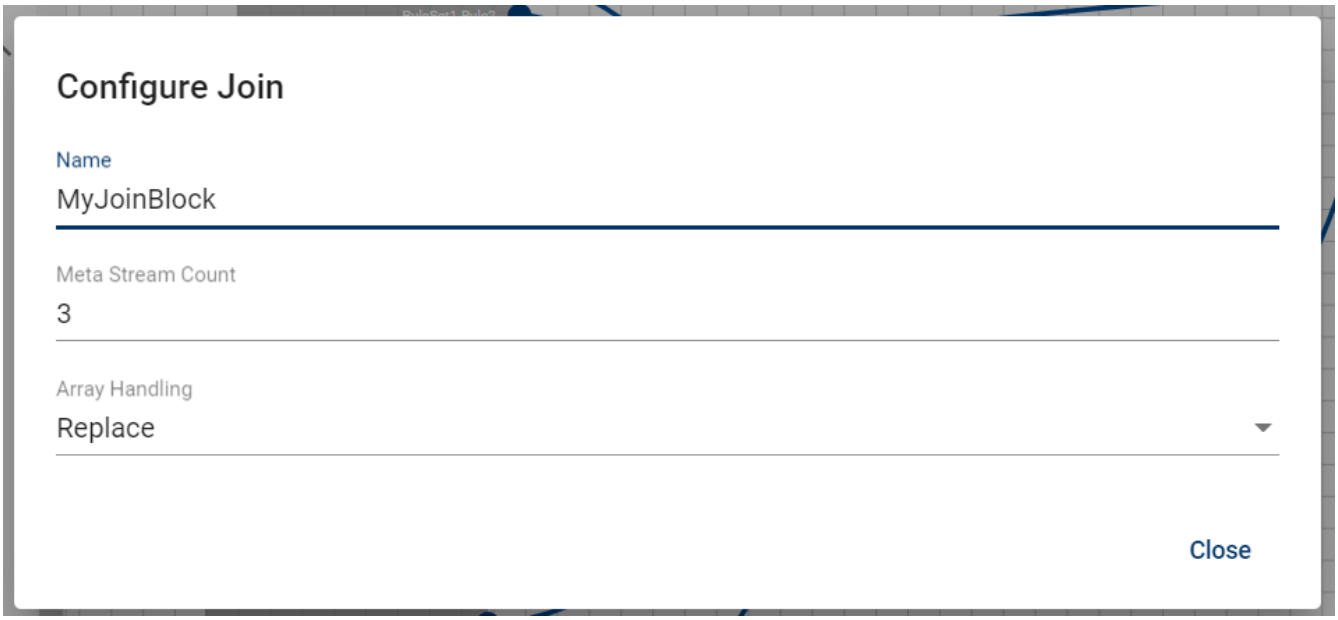


Abbildung 59. Konfiguration des Join-Block in der UI

4.2.8. Resend

Funktionsbeschreibung und Anwendung

Der Resend-Block speichert jede einlaufende Nachricht in einem Cache und sendet diese in einem Intervall erneut aus. Jede einlaufende Nachricht wird dabei sofort wieder am Ausgang ausgesendet und der Timer für das erneute Aussenden neugestartet.

Der Block kann verwendet werden, um eine Datenübertragung in einem Zeitintervall zu erzwingen.

Beispielsweise wird ein Temperaturwert per Subscription gelesen und nur gesendet, wenn sich dieser ändert. In der Zielplattform möchte man jedoch mindestens einmal pro Tag ein Wert vom

EdgeGateway erhalten, da sonst z.B. ein Alarm ausgelöst wird. Der Resend-Block kann nun vor der Übergabe an die Zielplattform platziert werden, um mindestens alle 24h erneut den letzten Wert zu übertragen.

In der folgenden Tabelle werden die Konfigurationsparameter erläutert.

Tabelle 26. Konfigurationsparameter Resend-Block

Parameter	Beschreibung
Name	Name der Blockinstanz.
Into	Name der Blockinstanz des Folgeblocks an welchen Ausgabenachrichten weitergeleitet werden sollen.
ResendInterval	Millisekunden-Intervall, in dem die letzte eingelaufene Nachricht zyklisch erneut gesendet wird.
Upate Timestamp On Resend	default=false, true = Beim Resend wird der Zeitstempel mit der aktuellen Zeit überschrieben.

Beispielkonfiguration

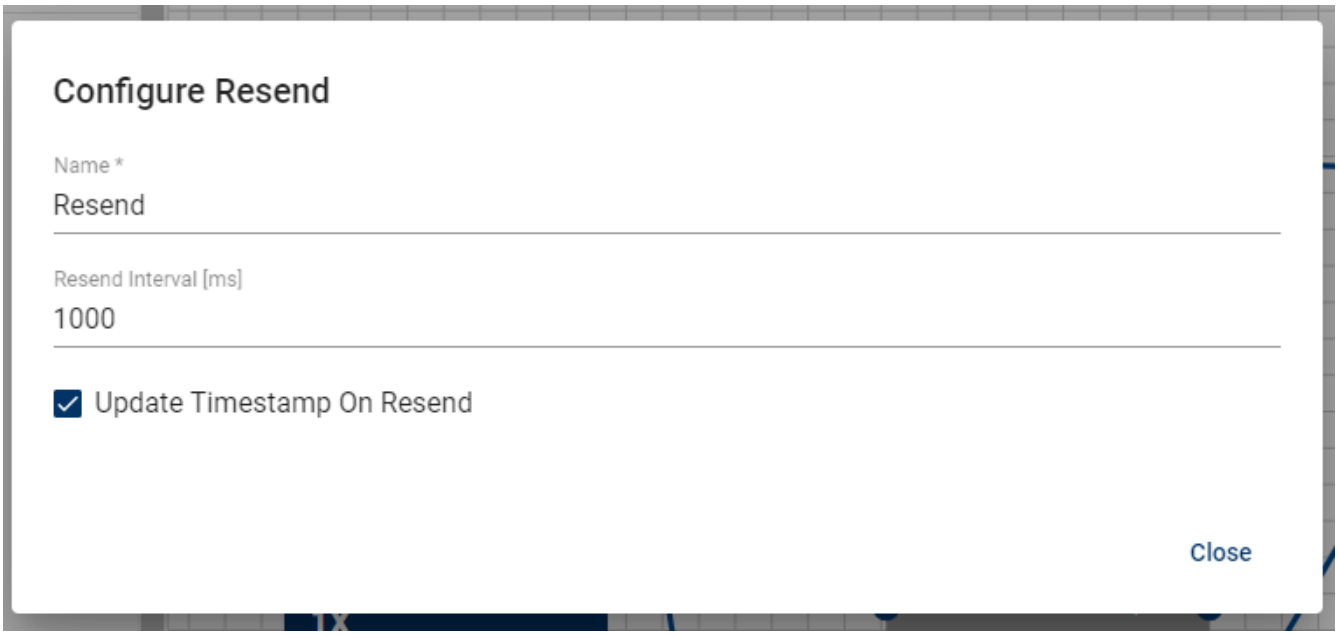


Abbildung 60. Konfiguration des Resend-Block

4.2.9. Router

Funktionsbeschreibung und Anwendung

Mit dem Router-Block können einlaufende Nachrichten bedingungsbezogen in bestimmte Processing-Blöcke weitergeleitet werden.

Hierfür gilt:

- Jede Bedingung muss einen booleschen Zustand zurückgeben.
- Jede Bedingung, die keinen booleschen Zustand zurückgibt, ist automatisch "false".

- Sind alle Bedingungen einer Route erfüllt, wird die Nachricht an alle mit dem Block verbundenen Nachfolgerblöcke weitergeleitet.
- Es werden immer alle konfigurierten Routes überprüft.
- Wenn keine Route erfüllt wird, wird die Nachricht das defaultOutputTopic des stream processor weitergeleitet.

Folgende Ausdrücke sind in den Bedingungen zulässig:

- Regular Expression
  - Syntax: RegEx(<value>,'<pattern>')
  - Hochkomma (''): Für Value kann ein statischer Wert eingetragen werden
  - Ohne Hochkomma (''): Value aus Parametern (Header & Body) wird geholt.
- alle Ausdrücke, die auch im Expression Evaluator Block angewendet werden können

Der Zugriff auf Nachrichteninhalte, wie die Eigenschaften des Headers und des Bodys, kann über die Bedingungen umgesetzt werden.

Die Tabelle `conditions` wurde durch den Condition-Builder ersetzt.

Der Condition Builder ermöglicht die Konfiguration über den Visual Editor. Es können folgende Werte konfiguriert werden:

- Source (header, body, context)
- Property
- Operator
- Comparison value

Im Text Editor werden die konfigurierten `conditions` angezeigt.

In der folgenden Tabelle werden die Konfigurationsparameter erläutert. Die Route-Blöcke enthalten die Routes mit ihren Bedingungen (conditions). Werden mehrere Bedingungen angegeben, so werden diese per Und-Logik verbunden. Das Ergebnis einer Bedingung muss immer einen boolschen Wert ergeben.

Tabelle 27. Konfigurationsparameter Router-Block

Parameter	Beschreibung
Name	Name der Blockinstanz
Route Name	Name des Route-Blocks. Wurden mehrere Route-Blöcke angelegt, erscheinen diese als wählbare Reiter unter der Eingabezeile.
Conditions	Sind alle Bedingungen einer Route erfüllt, wird die Nachricht an alle mit dem Block verbundenen Nachfolgerblöcke weitergeleitet.

Beispielkonfiguration

Die Schrittweise Konfiguration des Stream Processors mit Block Routing wird am folgenden

Beispiel erläutert:

1. Aggregate-Blöcke 1 bis 3 mit den gleichen Parametern anlegen (Abbildung 61)
   - Samples mit Size „1” und HoppingEach „1“
   - Behavior: ignore default and collect für „b”
2. Neuen Processing Block vom Type Router mit Name "Routing" anlegen (Abbildung 62)
   - Routes anlegen:
     - Route Name: b<10000
     - Conditions:
       - (header_scope == 'a$' || body_b < 10000)

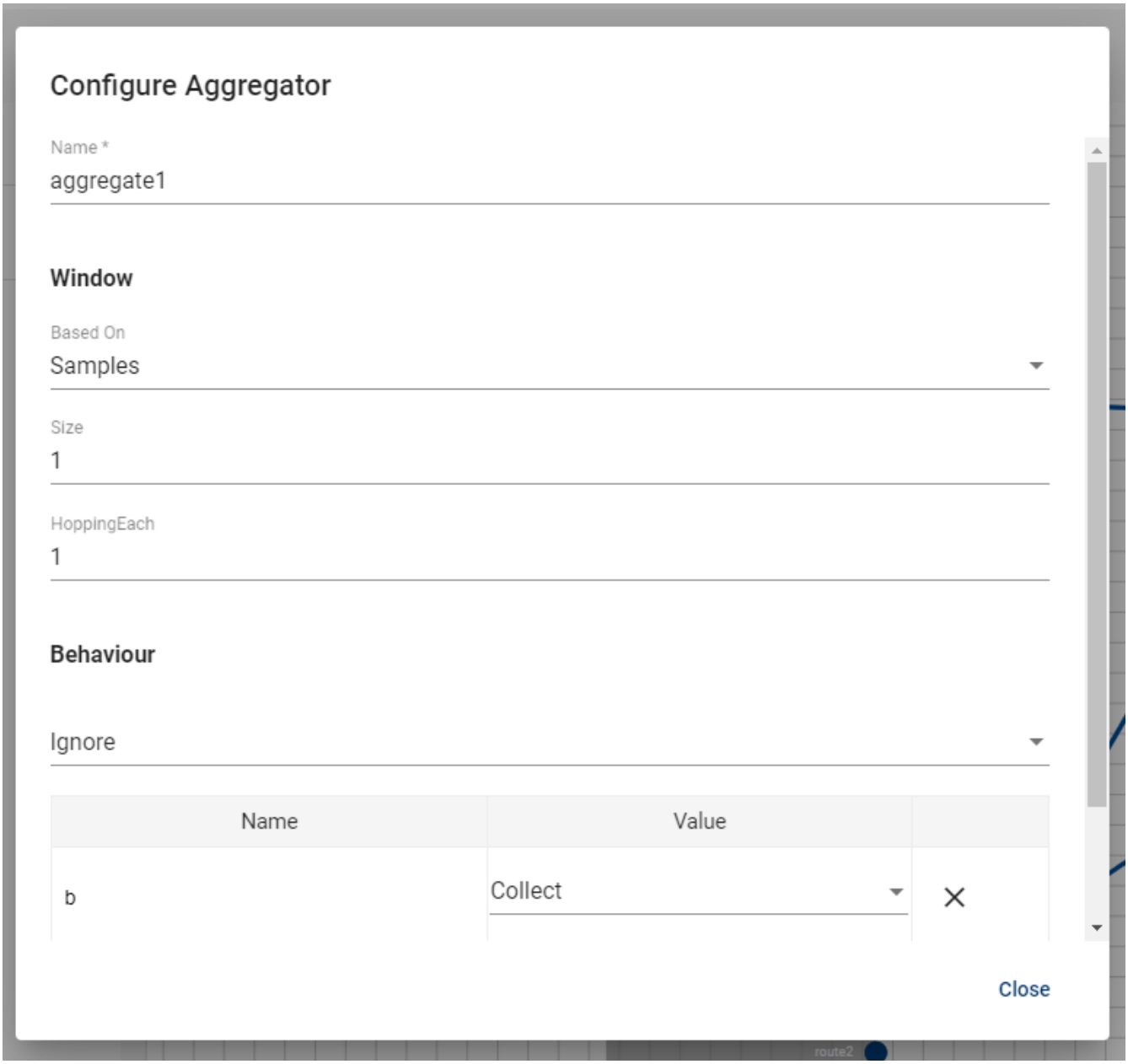


Abbildung 61. Konfiguration des Aggregator-Blocks "aggregate1"

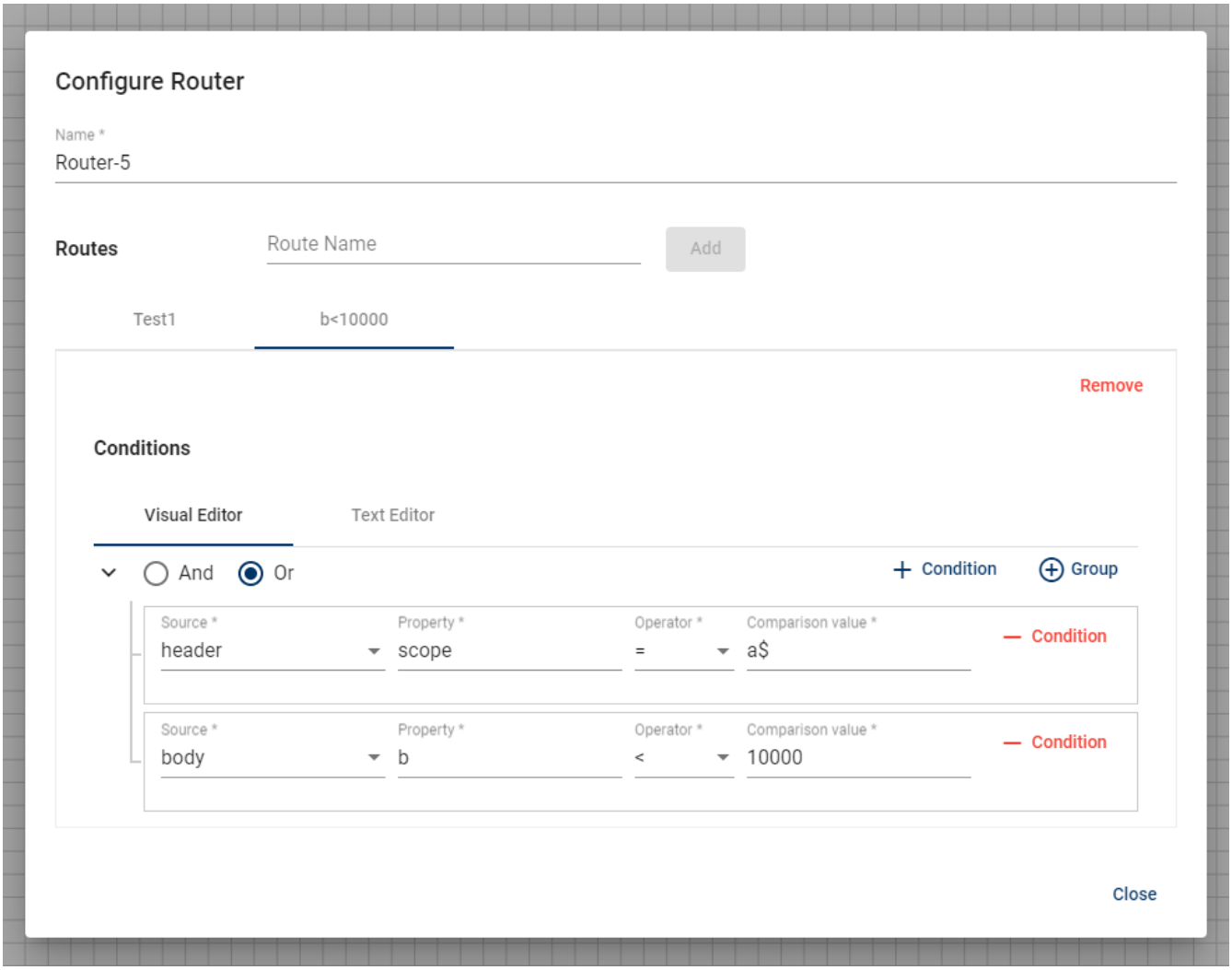


Abbildung 62. Konfiguration des Route-Block "Router-5"

Konfiguration in JSON

```
{
    "Router": {
        "type": "Router",
        "routes": {
            "intoAll": {
                "into": [
                    "Block1",
                    "Block2",
                    "Block_Dedup"
                ],
                "condition": "(true)"
            }
        }
    }
}
```

4.2.10. RuleEngine

Funktionsbeschreibung und Anwendung

Der RuleEngine-Block dient der Beschreibung von Zustandsübergängen anhand einlaufender Nachrichten in Abhängigkeit von Vorbedingungen. Dabei können Form und Inhalt der Ausgangsnachricht frei definiert werden.

- Es existiert ein **Kontext** in dem Informationen abgelegt werden können, die bei der Auswertung der Bedingungen oder in der Ausgangsnachricht verwendet werden können. Der Kontext kann initial mit Informationen gefüllt werden.
- Es gibt eine Menge von "**RuleSets**", welche (Vor-)Bedingungen beinhalten.
- Jedes RuleSets kann eine Menge von "**Rules**" enthalten, die ebenfalls über Bedingungen verfügen.
- Weiterhin können ein "defaultRuleSet" sowie in RuleSets "defaultRules" angelegt werden, die keine Bedingungen enthalten.

Jede **Bedingung** muss einen boolschen Zustand zurückgeben. Jede Bedingung, die keinen boolschen Zustand zurückgibt, ist automatisch "Falsch" (false). In den Bedingungen kann auf Inhalte des Kontextes und der eingehenden Nachricht (header & body & context) zugegriffen werden. Damit eine "Rule" zutrifft, müssen alle Bedingungen von dieser erfüllt sein.

Die Tabelle `conditions` wurde durch den Condition-Builder ersetzt.

Der Condition Builder ermöglicht die Konfiguration über den Visual Editor. Es können folgende Werte konfiguriert werden:

- Source (header, body, context)
- Property
- Operator
- Comparison value

Im Text Editor werden die konfigurierten `conditions` angezeigt.

Zuerst werden die Bedingungen der definierten RuleSets der Reihe nach, von RuleSet-1 ausgehend, geprüft. Sobald ein RuleSet die Bedingungen erfüllt, wird dieses RuleSet für die weitere Verarbeitung der Nachricht ausgewählt. Erfüllt kein RuleSet die Bedingung, wird das defaultRuleSet gewählt, sofern es angegeben ist, andernfalls erfolgt ein Abbruch.

Anschließend werden im ausgewählten RuleSet die Rules geprüft. Die Reihenfolge der Prüfung ist abhängig von der definierten Reihenfolge der Rules in der Baumstruktur. Entsprechend kann eine Überprüfung der weiteren im RuleSet enthaltenen "Rules" übersprungen, die „zutreffendste“ oder die letzte zutreffende "Rule" verwendet werden. Erfüllt keine Rule die Bedingungen erfolgt ein Abbruch. Folglich wird im RuleEngine-Block nur ein RuleSet und darin nur eine Rule verarbeitet.

Das Ergebnis einer "Rule" wird in dem Kontext abgelegt, der Inhalt des Kontext wird gepatched. Es können gezielt Inhalte aus dem Kontext nach dem Patchen entfernt werden.

- Wird das angegebene Message Template nicht gefunden, wird das Message Template "default" genommen.
- Die Ausgangsnachricht wird anhand des Message Templates erzeugt.
- Ist in dem Message Template kein Header definiert, werden die Informationen "source" und "scope" aus der Eingangsnachricht übernommen.
- Ist in dem Message Template Header kein "source" und/oder "scope" definiert, werden die Informationen aus der Eingangsnachricht übernommen.
- Ist in dem Message Template kein Body definiert, enthält die Ausgangsnachricht keine Informationen im Body.
- Nach dem Erfüllen einer "Rule" wird eine Ausgangsnachricht an alle zugewiesenen Input bzw. Output Topics gesendet.

Bezeichnungen und Syntax

Die in den folgenden Tabellen erläuterten Parameter sind in einer beispielhaften Konfiguration und deren Abbildungen im folgenden Absatz zu finden.

Tabelle 28. Konfigurationsparameter RuleEngine-Block

Parameter	Beschreibung
Name	Name der Blockinstanz
Context	Optionaler Bereich in dem Variablen mit oder ohne initialen Inhalt initialisiert werden. Jedes Contextelement kann in den Bedingungen verwendet werden.
RuleSets	Zusammenstellung von Bedingungspfaden/Zustandsüberwachungen. Es können beliebig viele erstellt werden. Die angelegten RuleSets erscheinen als Reiter unterhalb der Eingabemöglichkeit. Die Namen "RuleSets-n" können nicht geändert werden, n steht hierbei für eine fortlaufende Zahl. Hier kann auch ein „DefaultRuleSet“ definiert werden.
Message Templates	Beliebig viele Vorlagen für die Ausgabe von Nachrichten und Daten. Es können hier alle Elemente aus dem Nachrichtenbereich Header und Body bearbeitet werden.

Tabelle 29. Konfigurationsparameter des Context im RuleEngine-Block

Parameter	Beschreibung
Name	optional, Name des Kontextelementes
Value	optional, Inhalt des Kontextelementes

Tabelle 30. Konfigurationsparameter der RuleSets im RuleEngine-Block

Parameter	Beschreibung
DefaultRuleSet	Zu befüllen wie ein normales RuleSet mit Rules – jedoch ohne Bedingungen. In diesem RuleSet können wiederum Rules angelegt werden, die diverse Bedingungen enthalten.
Conditions	optional, globale Bedingung(en) damit in diesem RuleSet die Rules auf Übereinstimmung geprüft werden.
Rules	Es kann zwischen den Reitern per Pfeiltasten gewechselt werden. Zusammenstellung von Bedingungspfaden/ Zustandsüberwachungen zur Verfeinerung der Bedingungspfade. Es können beliebig viele erstellt werden. Die angelegten Rules erscheinen als Reiter unterhalb der Eingabemöglichkeit. Die Namen können nicht geändert werden – Rules-n, n steht hierbei für eine fortlaufende Zahl. Hier kann auch ein „DefaultRule“ definiert werden.
Rules – Context	Beliebige Anzahl von Contextelementen
Rules – Context – Name	Name des Contextelementes
Rules – Context – Value	Inhalt des Contextelementes
Rules – Obsolete context	Hier können Elemente des Context entfernt werden.
Rules – Message Template	Optionale Eingabe eines Namens für das Message Template zur Erstellung der Output-Nachricht

Tabelle 31. Konfigurationsparameter der Message Templates im RuleEngine-Block

Parameter	Beschreibung
Header – Name	optional, Name des zu erstellenden keys
Header – Value	optional, sofern kein Name/ key definiert wurde, Wert aus dem Context, Body, Header oder selbstdefiniert
Body – Name	optional, Name des zu erstellenden keys
Body – Value	optional, sofern kein Name definiert wurde, Wert aus dem Context, Body, Header oder selbstdefiniert

Verwendung von Elementen aus Header, Body und Context:

- Groß- und Kleinschreibung ist zu beachten
- Um das Element source aus dem Nachrichtenbereich Header zu erhalten, muss header_ für den

zu verwendenden Bereich und source für das Element eingegeben werden. → header_scope

Verwendung einer Zeichenkette/ String

- Eine Zeichenkette wird mit einem vorangehenden „ ‘ „ und einem abschließenden „ ‘ „ gekennzeichnet. Bsp.: ‘Zeit verstrichen‘
- Werden die Zeichen nicht verwendet, so wird die Eingabe wie eine Variable verwendet.

Beispielkonfiguration

Die schrittweise Konfiguration im Stream Processor wird am folgenden Beispiel dargestellt. Es werden Daten eines OPC UA Servers verarbeitet. Die Reihenfolge sollte wie beschrieben erfolgen, da so alle angezeigten Auswahlmöglichkeiten im Zuge der Konfiguration zur Verfügung stehen. Auswahlwahlmöglichkeiten stehen immer erst nach einem Speichervorgang zur Verfügung.

Voraussetzung: Im Modul OPC UA wurde ein OPC UA Server mit dem Output Topic „OpcuaOut1“ konfiguriert (Abbildung 63).

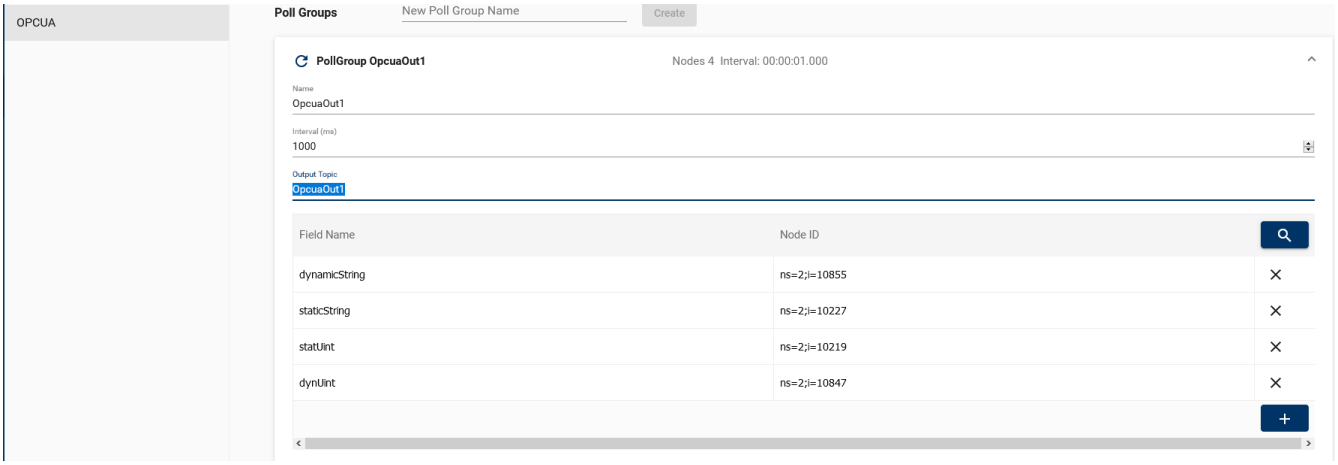


Abbildung 63. Pollgroup im OPC UA-Modul

Nun werden die grundlegenden Elemente im Stream Processor angelegt bzw. konfiguriert (Abbildung 64):

- Topic "OpcuaOut1" unter "Settings" als Input Edge Topic hinzufügen
- Input-Block "switch1" anlegen und Source (OPCUA) und Scope (OpcuaOut1$) konfigurieren
- RuleEngine-Block anlegen mit Name "UsingRules"
- Output-Block "OUTBlock" anlegen

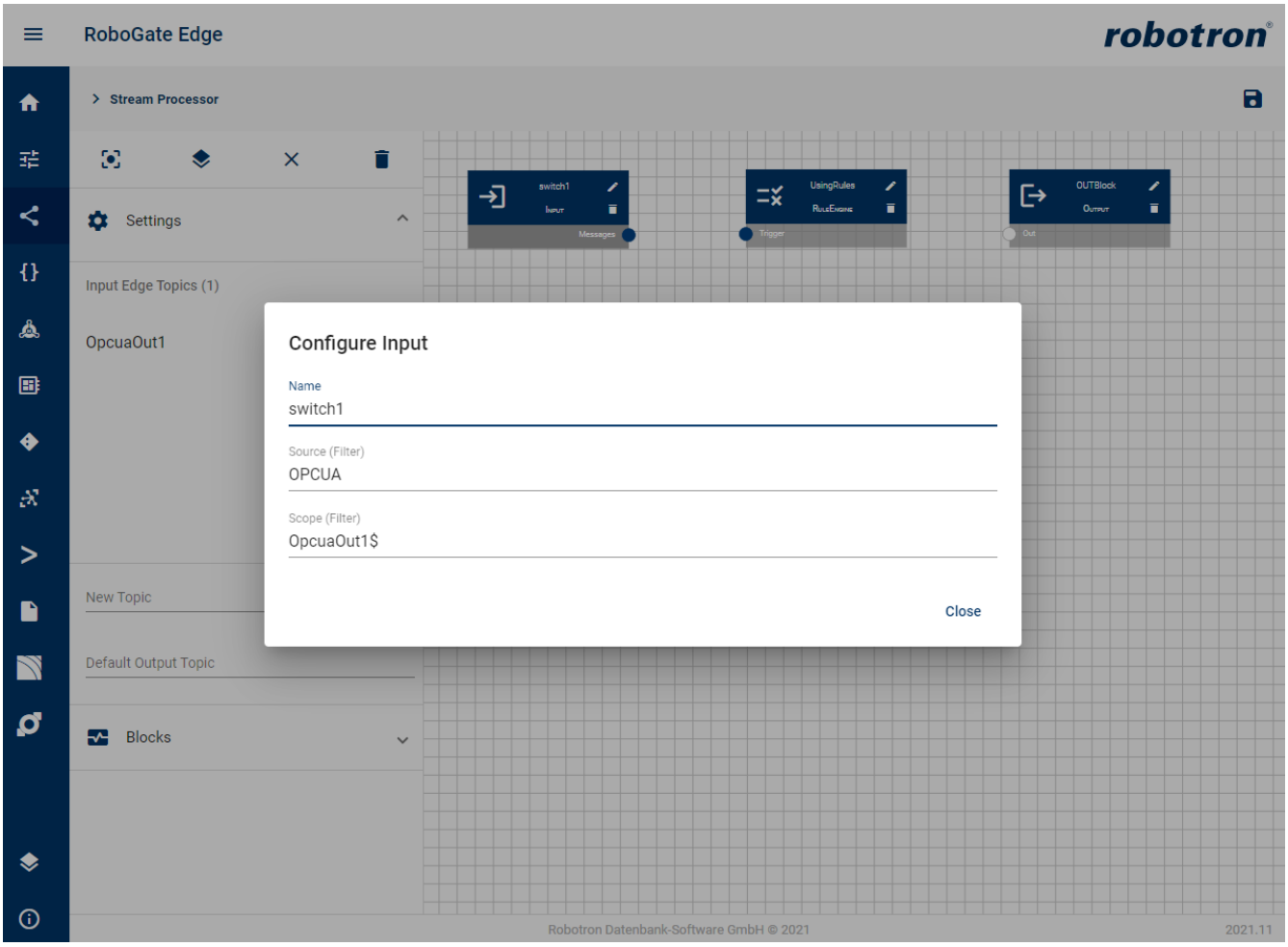


Abbildung 64. Grundeinstellungen im Stream Processor

Die Konfiguration der RuleEngine erfolgt unter Context, RuleSets und Message Templates.

Im Tab RuleSets bzw. Message Templates steht eine Baumstruktur zur Verfügung. Um die Reihenfolge der Elemente in der Baumstruktur zu verändern, lassen sich diese via drag & drop verschieben.

Neue Unterelemente können über ein "+" im Baum neben dem übergeordneten Element angelegt werden.

Die Unterelemente können über über die Schaltfläche entfernt werden.

Um die Konfiguration zu übernehmen, muss diese über die Schaltfläche gespeichert werden.

Weiter in der Beispielkonfiguration RuleEngine-Block:

Es folgt die Definition der **Message Templates** "default" und "Rule1" im RuleEngine-Block:

- default: Änderung des Inhaltes des Headers von Source mit einer Zeichenfolge und des Inhaltes Zusatzinformation mit einer Zeichenfolge im Body (Abbildung 65)
- Rule1: Änderung des Inhaltes des Headers von Source mit einer Zeichenfolge und Scope mit dem Inhalt eines Elementes vom Context. Der Body wird mit Werten aus dem ursprünglichen Body - statUint und dynUint – befüllt und mit während der Abarbeitung der Rules generierten Werten – Kontext1 und Zusatz (Abbildung 66).

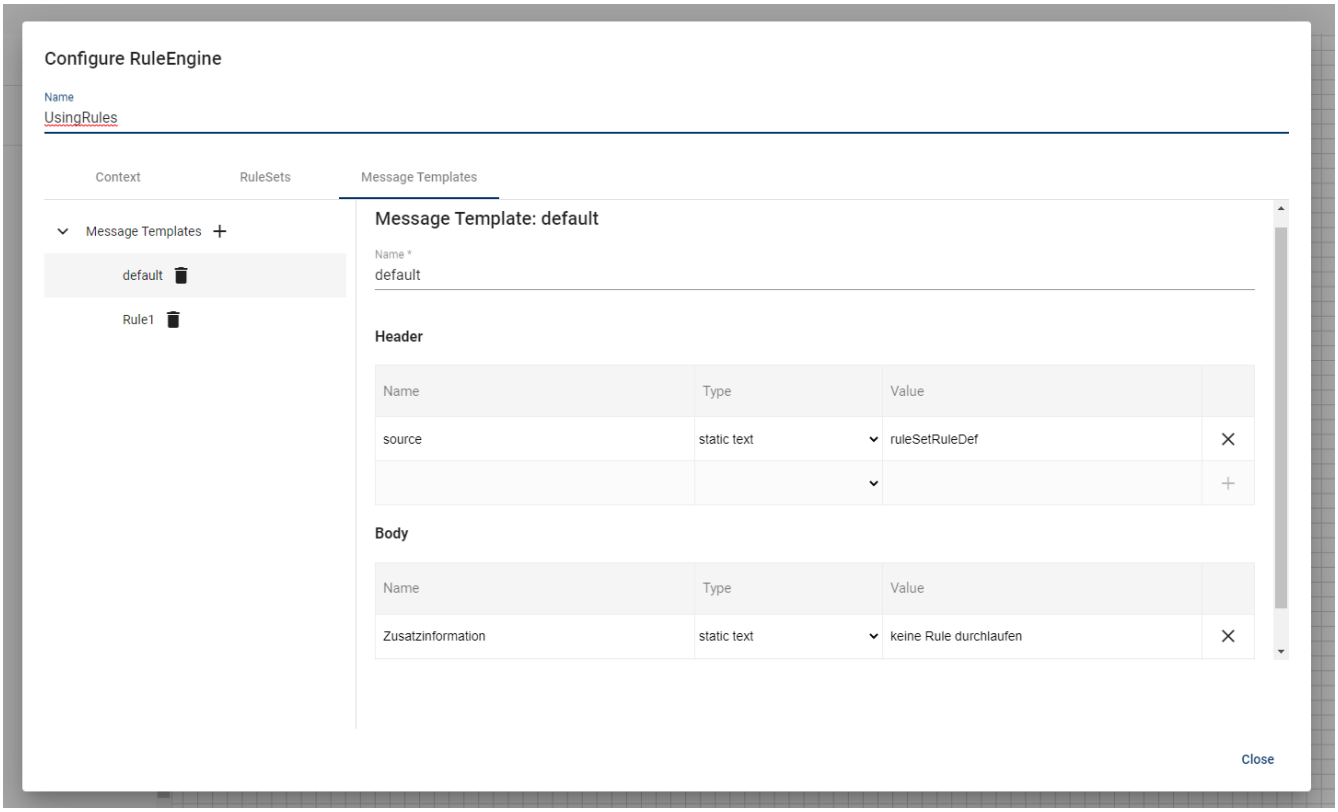


Abbildung 65. Einstellungen in Message Template "default"

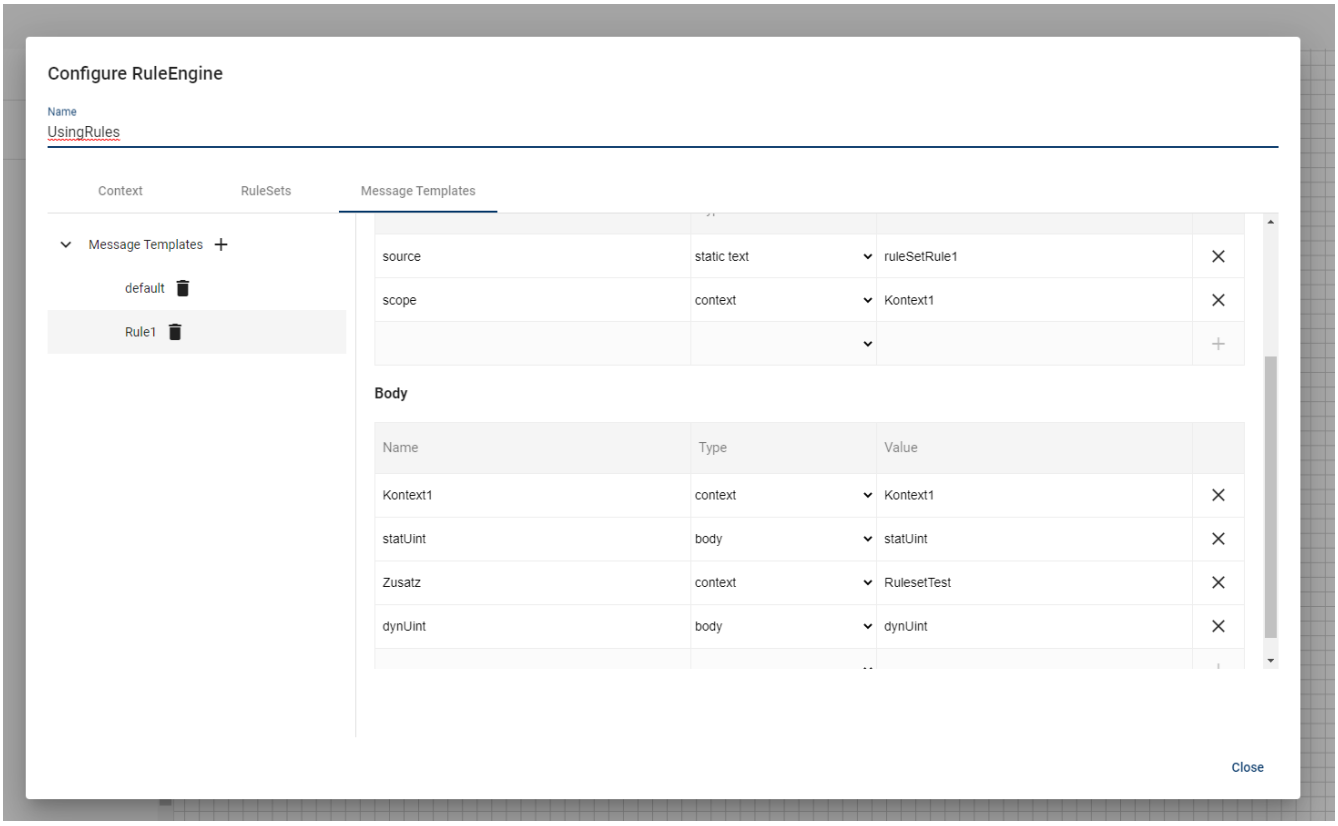


Abbildung 66. Einstellungen in Message Template "Rule1"

Anschließend wird der **Context** des Block "UsingRules" konfiguriert, hier optional die Eingabe von Kontext mit Inhalten(Abbildung 67).

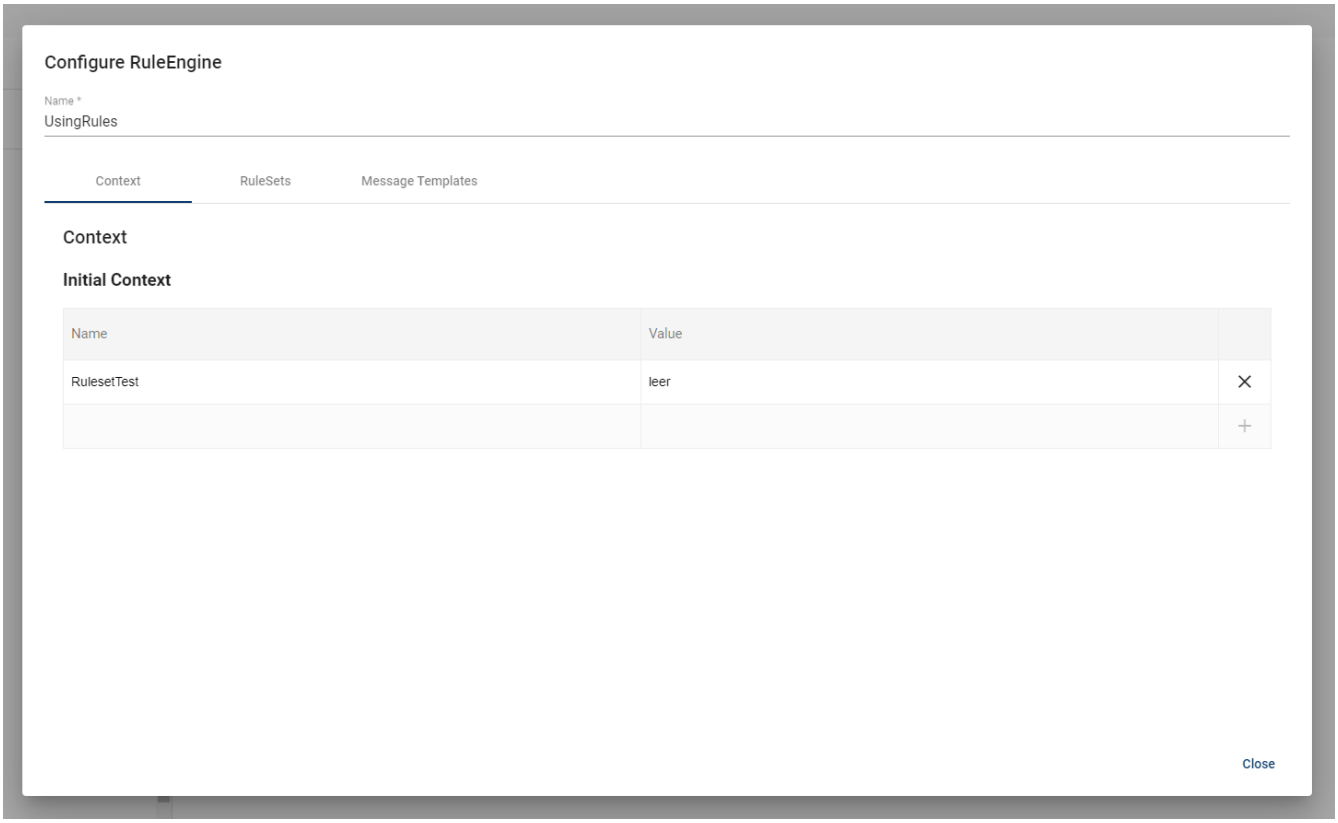


Abbildung 67. befüllter Context

Es folgt die Konfiguration der **RuleSets**:

- DefaultRule (Abbildung 68)
  - RuleSets: Add Default → DefaultRuleSet erstellt
  - DefaultRuleSet:Add Default-→DefaultRule erstellt
  - Context und Into wurden leer gelassen
  - Im Obsolete Context werden die Elemente Kontext2 und Kontext3 entfernt
  - Message Template "default" angeben

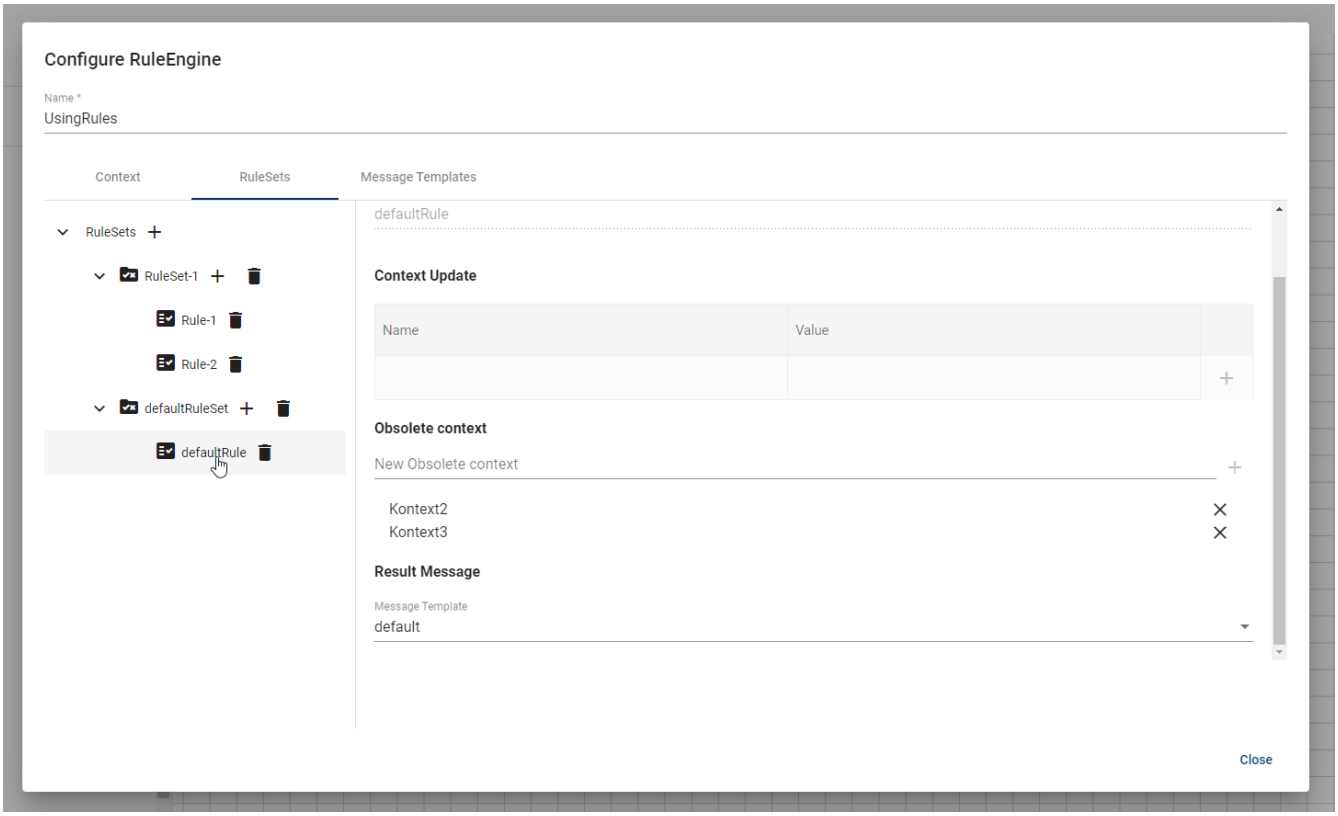


Abbildung 68. DefaultRule ausgefüllt

Ist das RuleSet-1 erstellt, müssen die **Rules** mit ihren Bedingungen konfiguriert werden.

- Rule Set-1-→Add Rule → Rule-1 erstellt
  - Konfiguration der Conditions im Visual Editor
    - Source: body
    - Property: dynUint
    - Operator: >=
    - Comparison value: 30000
  - Im Context werden zwei Elemente mit einem neuen Wert beschrieben:
    - RulesetTest mit der Zeichenfolge 'RuleSet1 Rule1'
    - Kontext1 mit der Zeichenfolge 'dynUint >= 35000'
  - Obsolete Context wird leer gelassen
  - Message Template mit Rule1 angeben

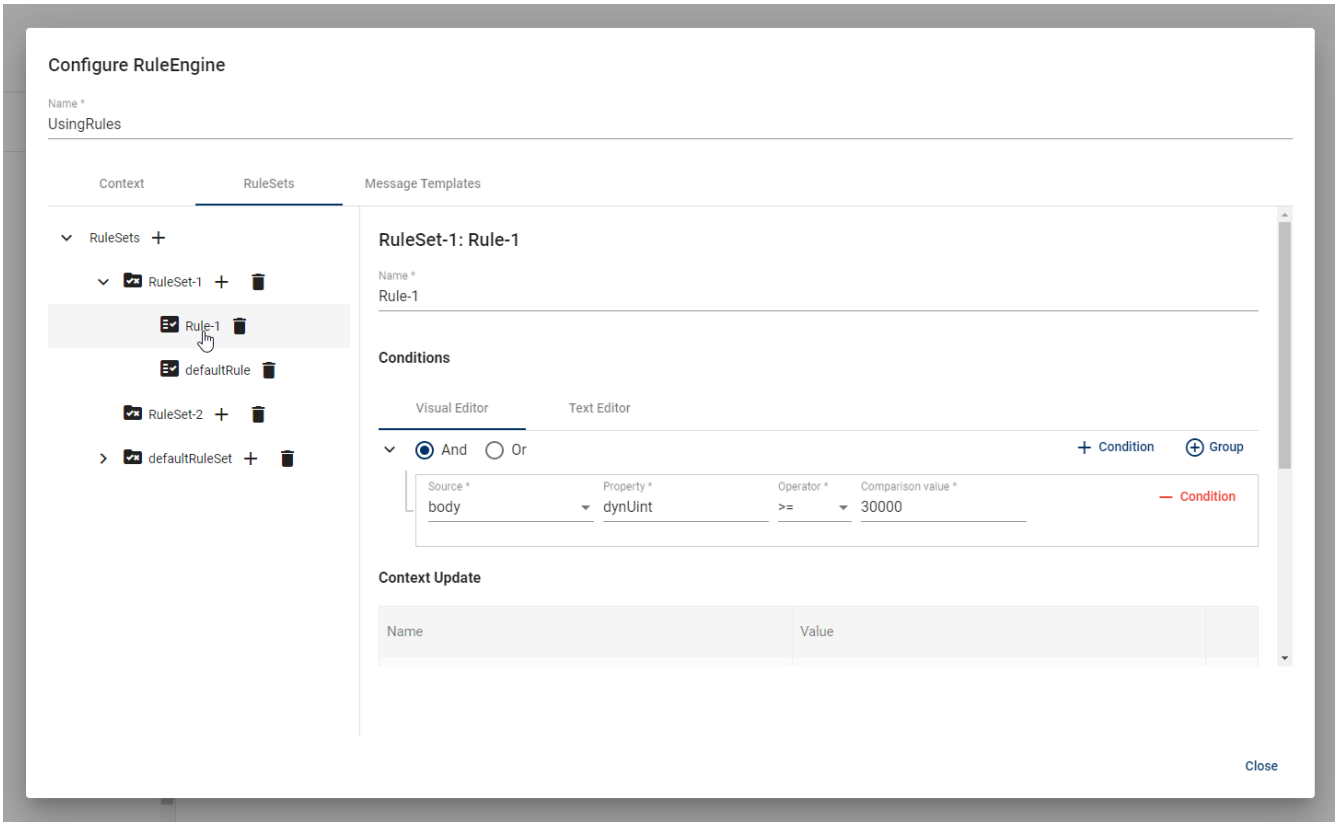


Abbildung 69. Ruleset-1 Rule-1 Conditions ausgefüllt

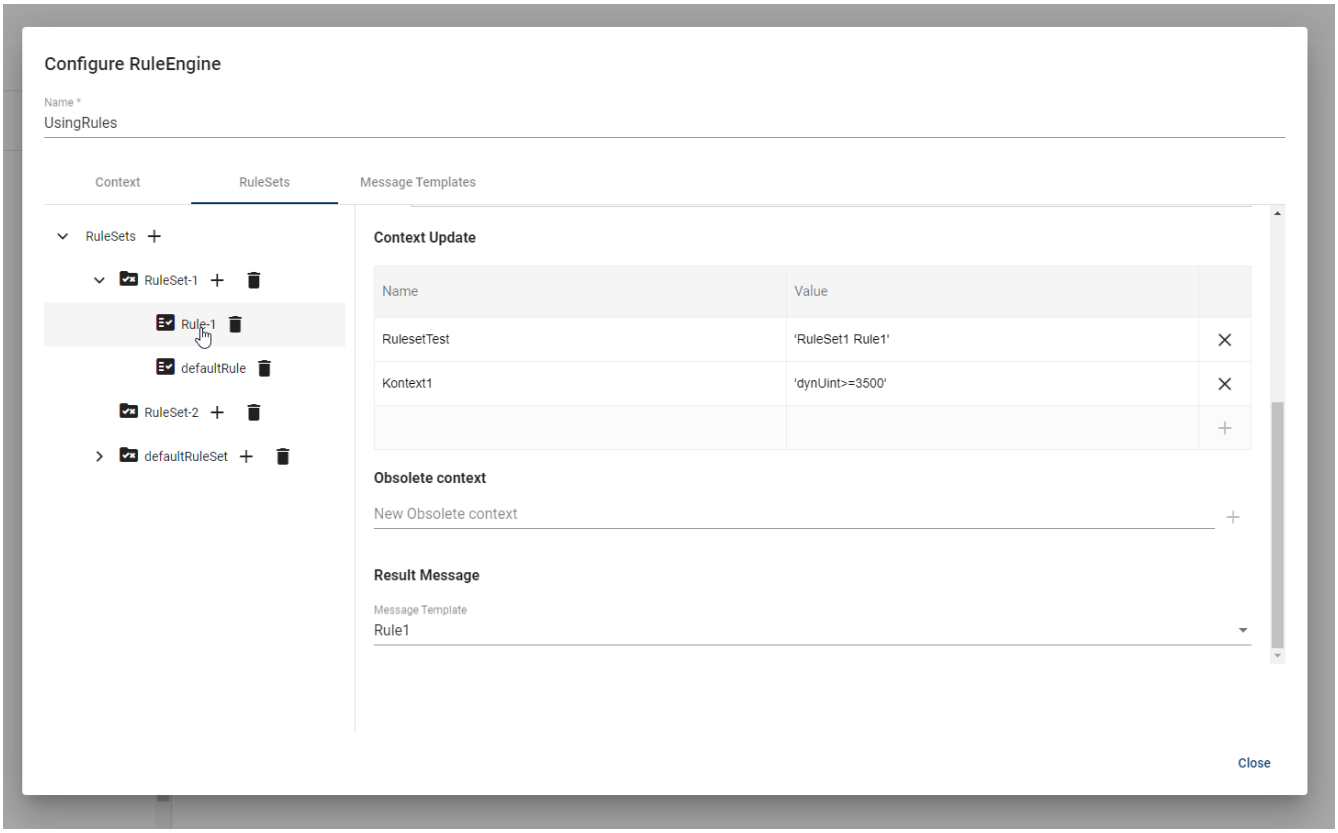


Abbildung 70. Ruleset-1 Rule-1 Context und Message Template ausgefüllt

- Rule Set-1-→Add Rule → Rule-2 erstellt (Abbildung 71)
  - Konfiguration der Conditions im Visual Editor
    - Source: body

    - Property: dynUint
    - Operator: ⇐
    - Comparison value: 10000

- Im Context werden zwei Elemente mit einem neuen Wert beschrieben:
    - RulesetTest mit der Zeichenfolge 'RuleSet1 Rule2'
    - Kontext1 mit der Zeichenfolge 'dynUint < = 10000'
- Obsolete Context wird leer gelassen
- Message Template mit Rule1 angeben

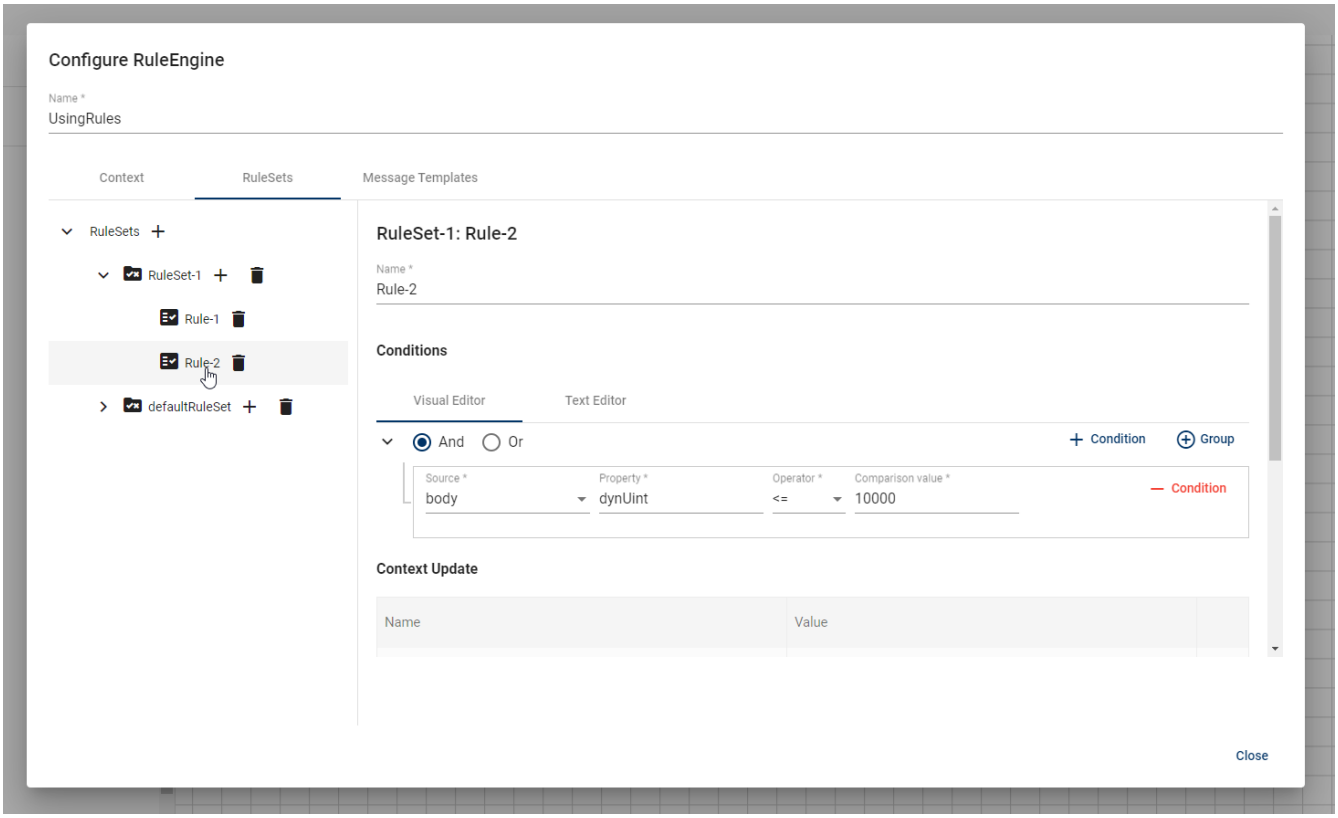


Abbildung 71. Ruleset-1 Rule-2 Conditions ausgefüllt

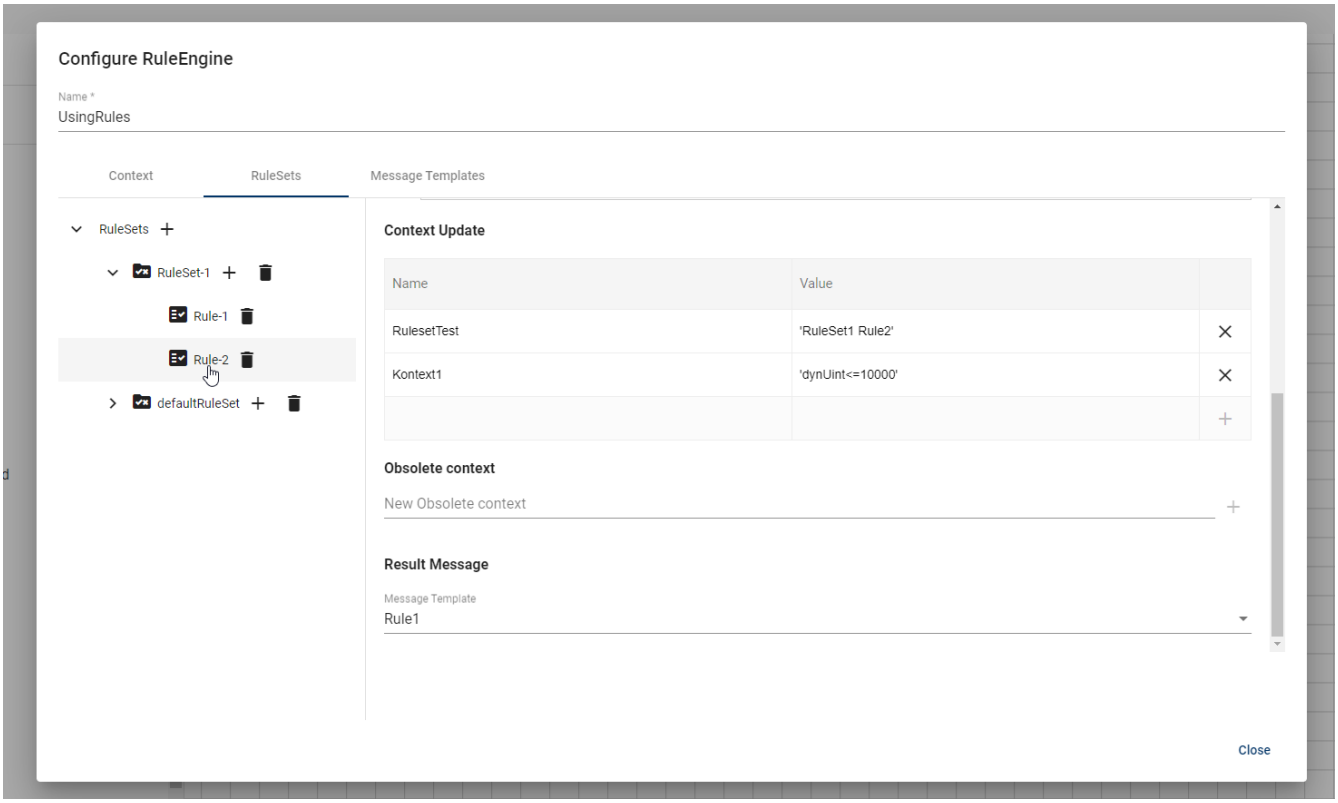


Abbildung 72. Ruleset-1 Rule-2 Context und Message Template ausgefüllt

Abschließend müssen die konfigurierten Blöcke verbunden werden (Abbildung 73).

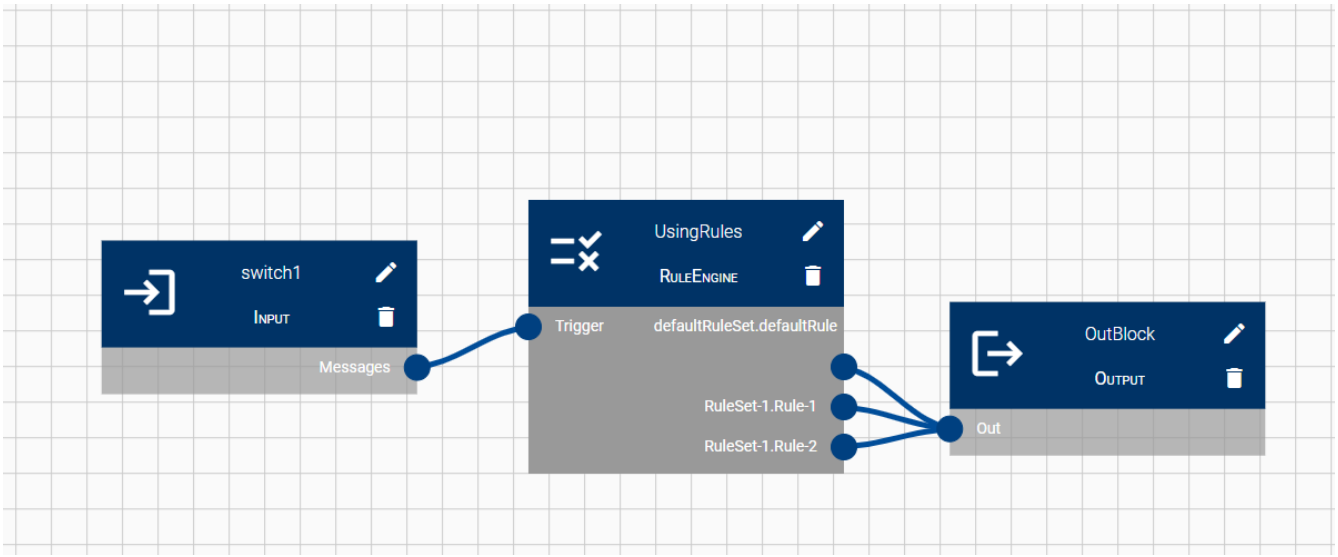


Abbildung 73. Verbindung der konfigurierten Blöcke

Mit dieser Konfiguration kann nun über den MQTT oder über andere Schnittstellen die fertige Nachricht ausgegeben werden.

4.2.11. SNCFilter

Funktionsbeschreibung und Anwendung

Der Significant-Numeric-Change-Filter (SNCFilter) dient der Reduzierung der Datenlast durch schwankende Sensor- oder Analogwerte. Der Filter sendet nur eine Nachricht, wenn:

- Der Daten Key der zu filternden Variablen unter Thresholds konfiguriert ist
- UND der zugehörige Value ggü. der zuletzt weitergeleiteten Nachricht eine Änderung betraglich größer/gleich des konfigurierten Thresholds hat.

Beim erstmaligen Eintreffen eines neuen Parameters, d.h. der Parameter ist im internen Cache noch nicht vorhanden, wird dieser angelegt und zukünftig immer durchgeleitet.

Folgende Parameter sind relevant für die Filterfunktion:

- Value: aktueller Wert der Variable in der Eingangsnachricht
- ValueCache: letzter durchgereichter Wert der Variable
- Threshold: konfigurierte Schwelle

Abbildung 74 veranschaulicht das Verhalten des Filters.

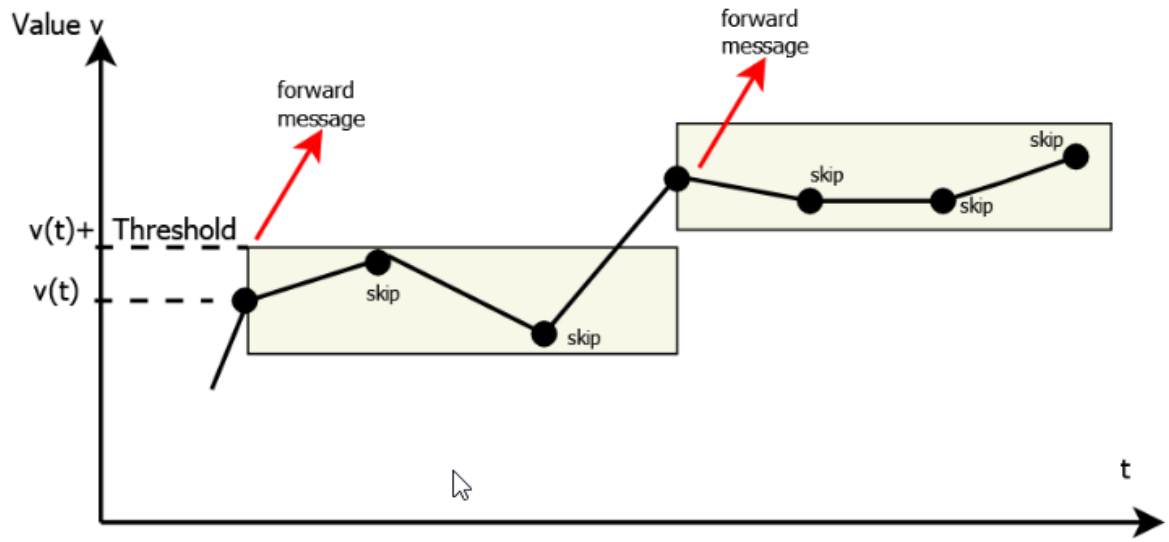


Abbildung 74. Skizze SNC-Filterverhalten

Die in folgender Tabelle erläuterten Parameter sind Abbildung 75 zu finden.

Tabelle 32. Konfigurationsparameter SNCFilter-Block

Parameter	Beschreibung
Name	Name der Blockinstanz.
Into	Name der Blockinstanz des Folgeblocks an welchen Ausgabenachrichten weitergeleitet werden sollen.
Thresholds - Name	Name des Daten Key (Variable) in der eingehenden Nachricht. Zugriff nur auf die oberste Objektebene im JSON. Die Variable muss einen numerischen Wert (Ganz- oder Fließkomma Typen) haben.

Parameter	Beschreibung
Thresholds - Value	double oder integer; Numerische Schwelle (einseitig) der Variable. Genauigkeitsverlust: Die Variable kann ein Bereich von knapp 2*Threshold überschreiten, ohne ein Nachrichten-Update aus Ausgang.
Overwrite Header Scope with	string; optional. Name für neue Scope Bezeichnung, zum Überschreiben des Header-Feldes Scope. Option ist inaktiv, wenn String nicht vorhanden oder leer ist.
Issue Full Model	boolean, default=true; *IssueFullModel=false*: Sendet jede Änderung einer Variablen separat in einer Nachricht (mit timestamp) an den Broker. Es entstehen ggf. mehrere Nachrichten aus einer einlaufenden Nachricht, wenn mehrere konfigurierte Schwellen überschritten wurden. *IssueFullModel: true*: Sendet die gesamte einlaufende Nachricht mit allen Parametern, sobald eine Schwelle überschritten wurde.

Anmerkungen

- Bei einem Schwellwert von 0 wird jede Änderung weitergeleitet. Vergleichbar also der Abwesenheit eines SNC-Filters.
- Es werden nur geänderte Werte weitergeleitet, wenn diese sich auch im SNC-Filter befinden. Beispiel: die Gruppe Auto enthält die Knoten A,B,C,D. Die Knoten A und B sollen durch den SNC-Filter laufen und nur bei größeren Änderungen Werte ausgeben. C und D laufen zwar auch in den SNC-Filter, werden aber nicht erwähnt. Falls sich jetzt in C oder D ein Wert ändert, passiert nichts. Die gesamte Gruppe Auto wird nur durchgereicht, falls sich ein Knoten im SNC-Filter ändert (voraussgesetzt Issue Full Model ist auf true).
- Falls Boolsche Werte (true/false) weitergeleitet werden sollen, sollte der einzustellende Wert 1e-8 betragen. Boolsche Werte selbst lassen sich natürlich nicht "runden". Allerdings kann es vorkommen, dass innerhalb einer Datengruppe Boolsche Werte und Fließkommazahlen vorkommen und alle Werte weitergeleitet werden sollen.

Beispielkonfiguration

Folgende Abbildung zeigt eine beispielhafte Konfiguration des SNCFilter.

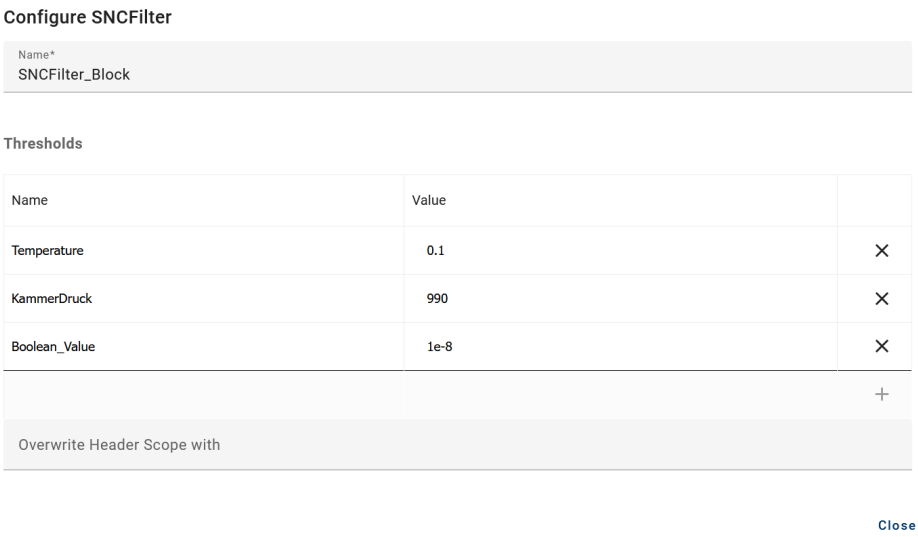


Abbildung 75. Konfiguration des SNCFilter-Block

Konfiguration in JSON

```
{
    "SNCFilter_Block": {
        "type": "SNCFilter",
        "into": [
            "Next_Blog"
        ],
        "Thresholds": {
            "Temperature": 0.1,
            "KammerDruck": 990,
            "Boolean_Value": 1e-8
        }
    }
}
```

4.2.12. Timer

Funktionsbeschreibung und Anwendung

Der Timer-Block kann einlaufende Nachrichten verzögern. Im Detail besitzt der Block die folgenden Events:

a) **Start** bzw. Neustart des Timers.
Startet den Timer oder startet einen laufenden Timer von vorn. Es wird jedes mal eine Action (falls konfiguriert) `OnStartEvent` ausgelöst. Der Event Trigger ist über die Bedingungen in `OnStartEvent.conditions` festzulegen.

b) **Cancel** Timer. Ein laufender Timer wird aufgrund einer Nachricht gestoppt und die Action `OnCancelEvent` ausgeführt.
Der Event Trigger ist über die Bedingungen in `OnCancelEvent.conditions` festzulegen.

c) Timer **Expired**. Ein Timer hat die konfigurierte Zeitspanne erreicht.

Das Event wird nur durch den Timer getriggert und kann nicht per Bedingung konfiguriert werden. Das Event `OnExpiredEvent` wird ausgelöst, wenn die entsprechende Action ausgeführt wird.

Die `conditions` wurden durch den Condition-Builder ersetzt.
Dieser steht in folgenden Events zur Verfügung:

- `OnStartEvent`
- `OnCancelEvent`

Der Condition Builder ermöglicht die Konfiguration über den Visual Editor. Es können folgende Werte konfiguriert werden:

- Source (header, body, context)
- Property
- Operator
- Comparison value

Im Text Editor werden die konfigurierten `conditions` angezeigt.

Die angegebenen `conditions` müssen alle erfüllt sein (true), um den Trigger/die Aktion zu erhalten/auszulösen. Der Zugriff auf die Nachrichten Properties der Timer Start Nachricht über `body_<BODY_KEY>` und `header_<HEADER_KEY>` ist ebenso wie in den Blöcken RuleEngine und Router möglich.

Als Aktion auf ein Event stehen die folgenden Möglichkeiten über das Property `Type` zur Wahl:

- `SendMessage` ⇒ Sende eine vordefinierte Nachricht. Der Nachrichteninhalt ist unter Data zu konfigurieren.
- `SendStartMessage` ⇒ Sende die (komplette) Nachricht aus, die zuletzt zum Start des Timers geführt hat. Der `timestamp` der Nachricht wird mit dem Aussenden aktualisiert.

> Wird ein Timer bspw. mit den identischen Bedingungen für Start und Cancel konfiguriert und treffen diese mit einem Nachrichteneingang zu, so wird der Timer gestartet und sofort wieder gestoppt und die zugehörigen Aktionen ausgeführt.

Die folgende Tabelle erläutertet die Konfigurationsparameter des Timer-Blocks.

Tabelle 33. Konfigurationsparameter Timer Block

Parameter	Beschreibung
Name	Name der Blockinstanz.
Into	Name der Blockinstanz des Folgeblocks an welchen Ausgabenachrichten weitergeleitet werden sollen. mit Folgeblöcken verbunden und Ausgabenachrichten an die entsprechend angegeben Blockinstanzen (Namen) weitergeleitet werden.

Parameter	Beschreibung
Timespan(ms)	Zeitspanne für den Timer in Millisekunden
Konfiguration Timer	Folgende Events sind im Block vorhanden • OnStartEvent • OnCancelEvent • OnExpiredEvent Die konfigurierbaren Actions *OnStartEvent*, *OnCancelEvent* und *OnExpiredEvent* sind nur bei Bedarf zu konfigurieren und können ggf. weggelassen werden. Bei Auftreten eines der Events und einer konfigurierten Aktion wird der aktuelle Timestamp im Message Body hinzugefügt (und ggf. überschrieben). Bei ausgehenden Nachrichten kann das Event per Header Property timer-was: {started, canceled, expired} für weitere Prüfung herangezogen werden.
Conditions	Alle in diesem Objekt angegebene Ausdrücke müssen mit *true* ausgewertet werden, um das Event auszuführen.
Name	Nutzerdefinierter Name der Bedingung. Der Wert kann frei gewählt werden, es bestehen keine Abhängigkeiten zu anderen Konfigurationsparametern.
Value	string, Ausdruck für die Prüfung der Regel. Für den Zugriff auf die Properties der zugehörigen Start-Nachricht ist Folgendes zu beachten: Der Zugriff auf die Nachrichten-Properties der Timer Start-Nachricht ist über body_<BODY_KEY> und header_<HEADER_KEY> möglich.
TimerType	Als Aktion auf ein Event stehen folgende Möglichkeiten zur Wahl: • SendMessage ⇒ Sendet eine vordefinierte Nachricht. Der Nachrichteninhalt ist unter Data zu konfigurieren, in dem Fall ist das Konfigurationsobjekt Data notwendig. • SendStartMessage ⇒ Sendet die Nachricht aus, die zuletzt zum Start des Timers geführt hat. Der Timestamp der Nachricht wird mit dem Aussenden aktualisiert. Für diesen Mode wird das Konfigurationsobjekt Data ignoriert/nicht benutzt. Hinweis: Wird ein Timer bspw. mit den identischen Bedinungen für Start und Cancel konfiguriert und treffen diese mit einem Nachrichteneingang zu, so wird der Timer gestart und sofort wieder gestoppt und die zugehörigen Aktionen ausgeführt.

Parameter	Beschreibung
Header	Nachrichten-Body der auszusendenden Nachricht bei diesem Event.
Body	Nachrichten-Header der auszusendenden Nachricht bei diesem Event.

In der folgenden Darstellung ist ein beispielhaftes Szenario aufgezeigt. Es läuft die Nachricht **[1]** ein. Diese wird vom Block ignoriert, da die Timer condition (`state >=10`) nicht erfüllt ist. Nachricht **[2]** folgt einlaufend und triggert den Start, da hier `state >=10`. Der Timer startet und sendet eine Nachricht **[3]**, die vielleicht für Debugging Zwecke konfiguriert ist, sofort heraus. Der Timer erreicht die konfigurierte Zeit von 20 Sekunden und sendet die Nachricht **[4]**, welche der Nachricht **[2]** mit aktuellem Zeitstempel entspricht, am Ausgang heraus.

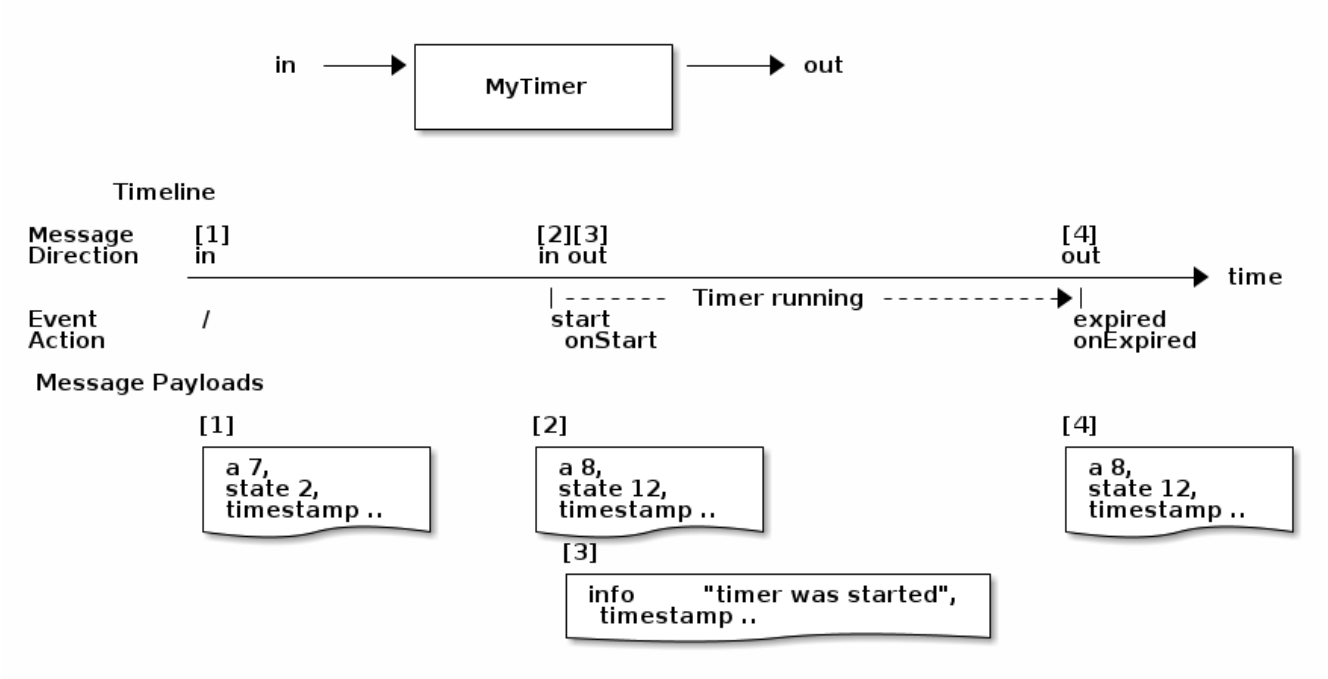


Abbildung 76. Timer Funktionsweise

Beispielkonfiguration

Folgende Abbildung zeigt eine beispielhafte Konfiguration des Timer-Blocks in der RoboGate Edge UI.

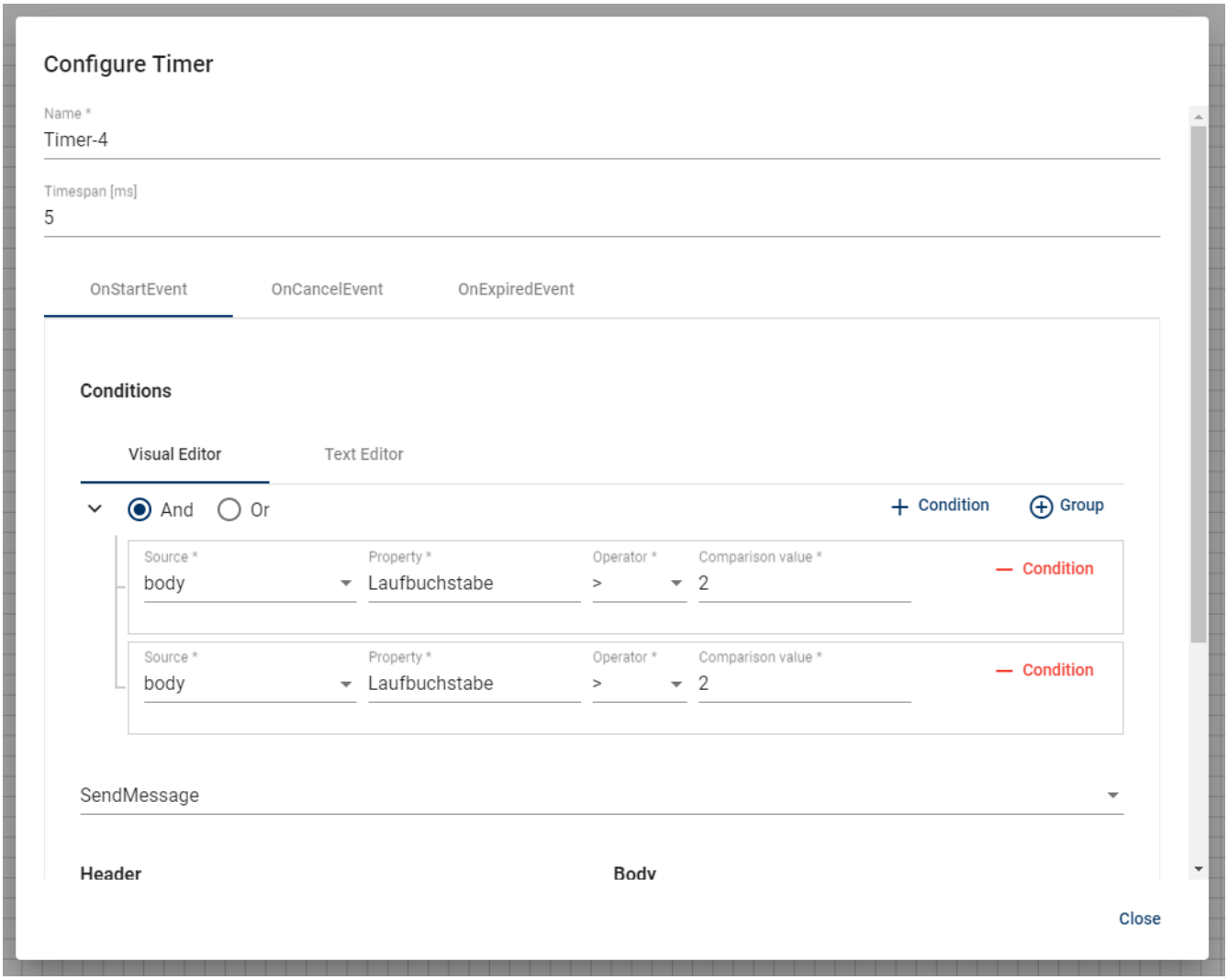


Abbildung 77. Konfiguration des Timer-Block VisualEditor

4.3. Template Enricher

Das Modul Template Enricher ist ein Tool zur Anpassung der Daten und ihrer Datenstruktur von empfangenen Nachrichten. Die einheitliche Datenstruktur dient der Unterstützung der Datenweiterverarbeitung. Es wird die Template-Sprache Scriban [3] unterstützt.

4.3.1. Konfigurationsparameter

Die Konfiguration des Template Enricher beinhaltet die nachfolgend beschriebenen Parameter.

Tabelle 34. Konfigurationsparameter des Template Enricher

Parameter	Beschreibung
Settings	• Input Settings: zu verarbeitendes Output Topic eines anderen Moduls. • Output Settings: neu definiertes Output Topic vom Template Enricher.
Headers	Anpassung der Metadaten (Scope, Source) der empfangenen Nachricht.

Parameter	Beschreibung
Body	Definition der Inhalte des neuen Nachrichten-Bodys.
Library	Verwaltung von wiederverwendbaren Template-Teilen.
Modules	Sammlung wiederverwendbarer Templates.
Environment	Festlegung eines statischen Objekts (JSON) auf das der Body zugreifen kann.
Test	Die Konfiguration des Template Enrichers kann getestet werden.
Message Output	Template, um Nachrichten anhand von Bedingungen an definierte Topics/Module weiterzugeben.

4.3.2. Konfiguration in der UI

Das Template Enricher-Modul kann über die Auswahl des Moduls auf der Startseite oder über die linke Menüleiste der RoboGate Edge UI aufgerufen und konfiguriert werden.

Settings

Für die Verwendung des Moduls muss mindestens ein in einem anderen Modul definiertes Output Topic als Input (z.B. aus dem Modul OPC UA). Diesem Topic sind die zu manipulierenden Nachrichten zugeordnet. Weiterhin muss mindestens ein neues Output Topic definiert werden, welchem die durch den Template Enricher manipulierten Nachrichten zugeordnet werden. Dieses neue Output Topic kann dann beispielsweise zur Datenausgabe in Azure bzw. Splunk hinzugefügt werden (Abbildung 78).

Die erläuterten Konfigurationen zu Input und Output sind unter *Settings* vorzunehmen. Die Eingabe von Input und Output Topic wird durch Klicken auf + übernommen. Mit einem Klick auf ✕ kann das Topic der entsprechenden Zeile gelöscht werden. Mit dem Speichern werden die Änderungen übernommen. Alle Eingaben und Änderungen werden erst mit dem Speichern wirksam.

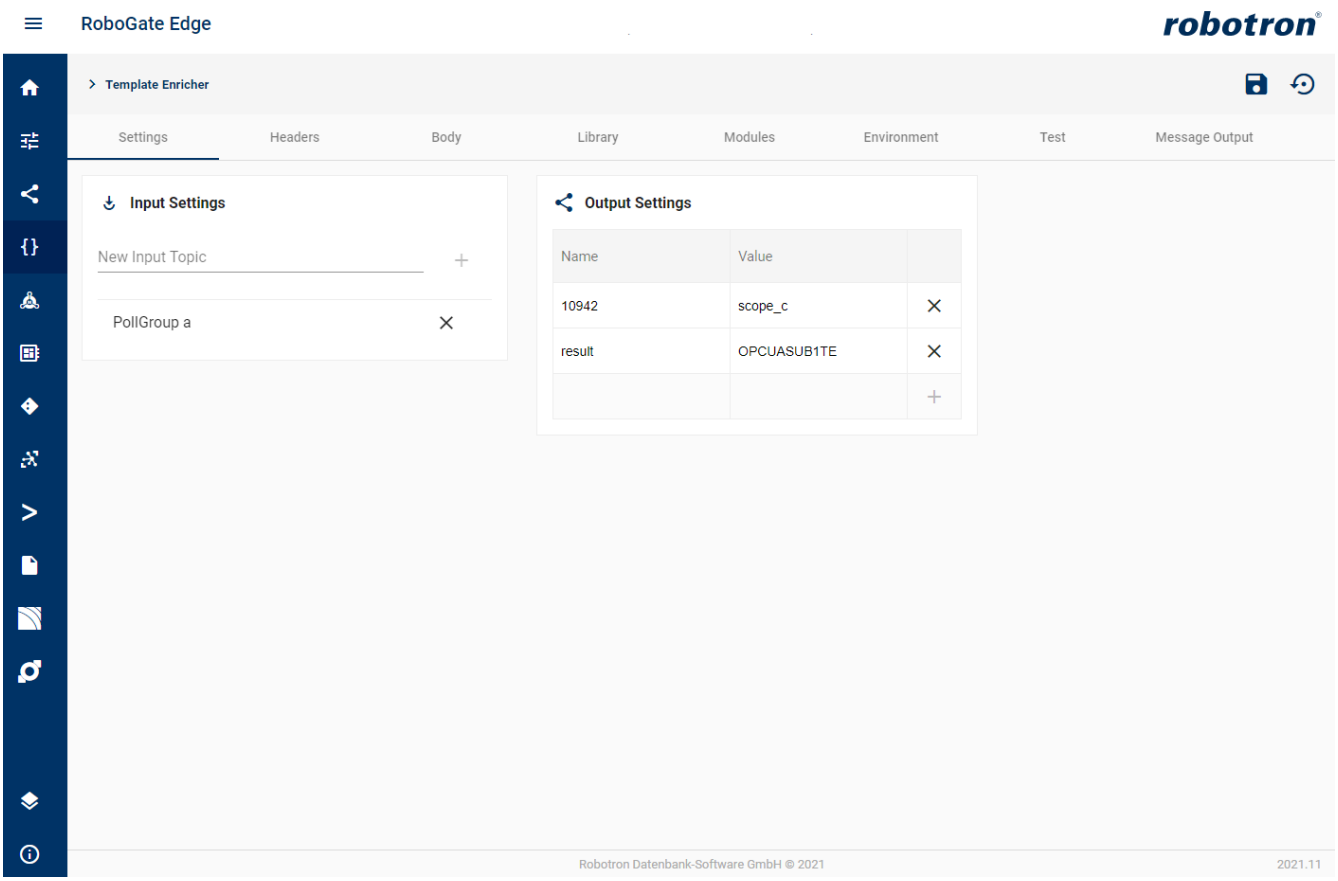


Abbildung 78. Konfiguration Template Enricher - Settings

Header

Im Reiter *Header* können Metadaten der Nachricht, wie z.B. Source und Scope, angepasst werden. Existiert der Key Value bereits, wird er mit dem angegebenen Wert überschrieben, existiert der Key Value noch nicht, wird er dem Header der Nachricht hinzugefügt.

Abbildung 80 und Abbildung 79 zeigen eine Beispielkonfiguration an. Im Beispiel wird die Source "PollGroup a" (siehe Abbildung 78) mit Bezeichnung "TE_a" überschrieben.

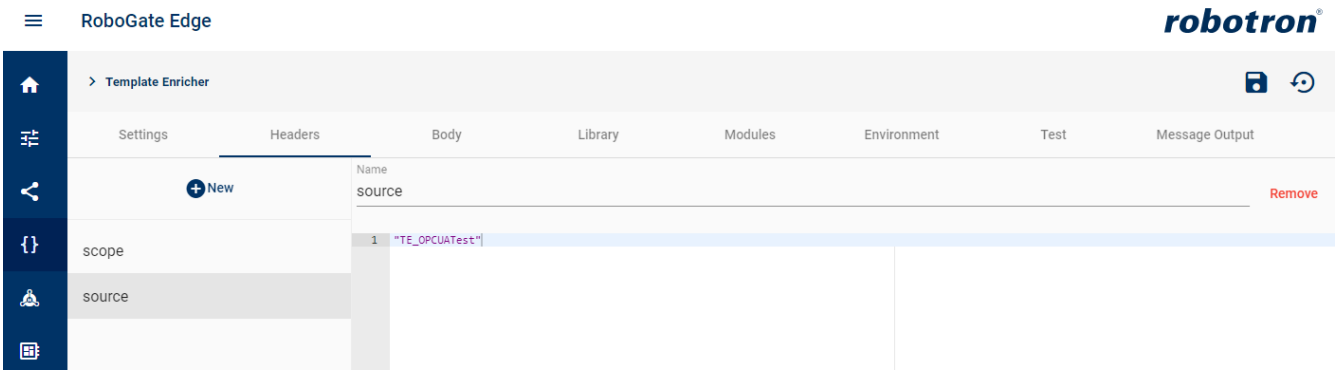


Abbildung 79. Beispielkonfiguration Header Source

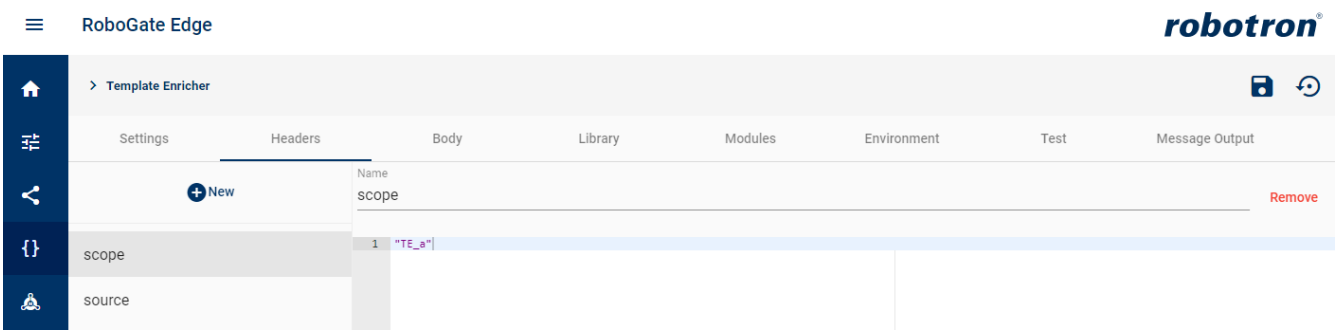


Abbildung 80. Beispielkonfiguration Header Scope

Body

Um die Rohdatenstruktur anzupassen, ist im Reiter *Body* der Quelltext zur Umwandlung einzufügen.

Existiert der Key Value bereits, wird er mit dem angegebenen Wert überschrieben, existiert der Key Value noch nicht, wird er dem Header der Nachricht hinzugefügt.

Im Body kann auf Objekte im Reiter *Environment* verwiesen werden. Dies bietet sich beispielsweise an, um maschinenspezifische statische Daten in den Body zu übernehmen. (Abbildung 81)

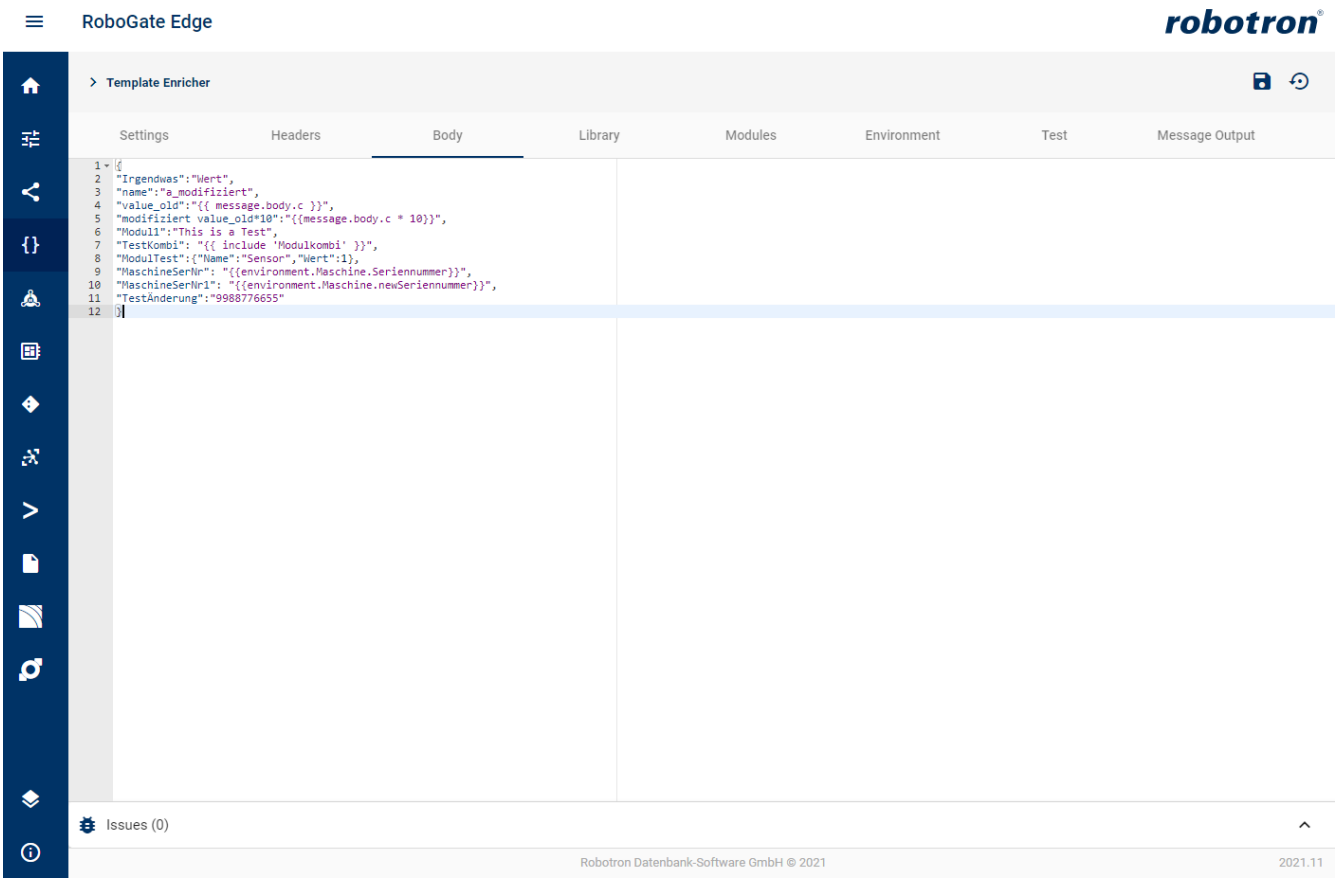


Abbildung 81. Konfiguration Template Enricher - Body

Library

In dem Reiter „Library“ können wiederverwendbare Templateabschnitte, welche in jeder Template-Definition verfügbar sein müssen, verwaltet werden.

Modules

Module sind eine Sammlung wiederverwendbarer Templates. Diese dienen dazu ein großes Template ggf. in kleinere Templates zu zerlegen, um die Lesbarkeit/Wartbarkeit zu erhöhen. Aus jedem Template (Body, andere Module) kann auf Module zugegriffen werden. Wenn ein Modul referenziert wird, wird der Inhalt von diesem an dieser Stelle eingefügt. Der Zugriff erfolgt via "include" Directive im Template. Die Schreibweise der "include" Directive sieht wie folgt aus:

```
{{ include 'Test' }}
```

An der Stelle 'Test' steht der Name des Modules, welches an dieser Stelle "eingefügt" werden soll. Abbildung 82 zeigt eine Beispielkonfiguration.

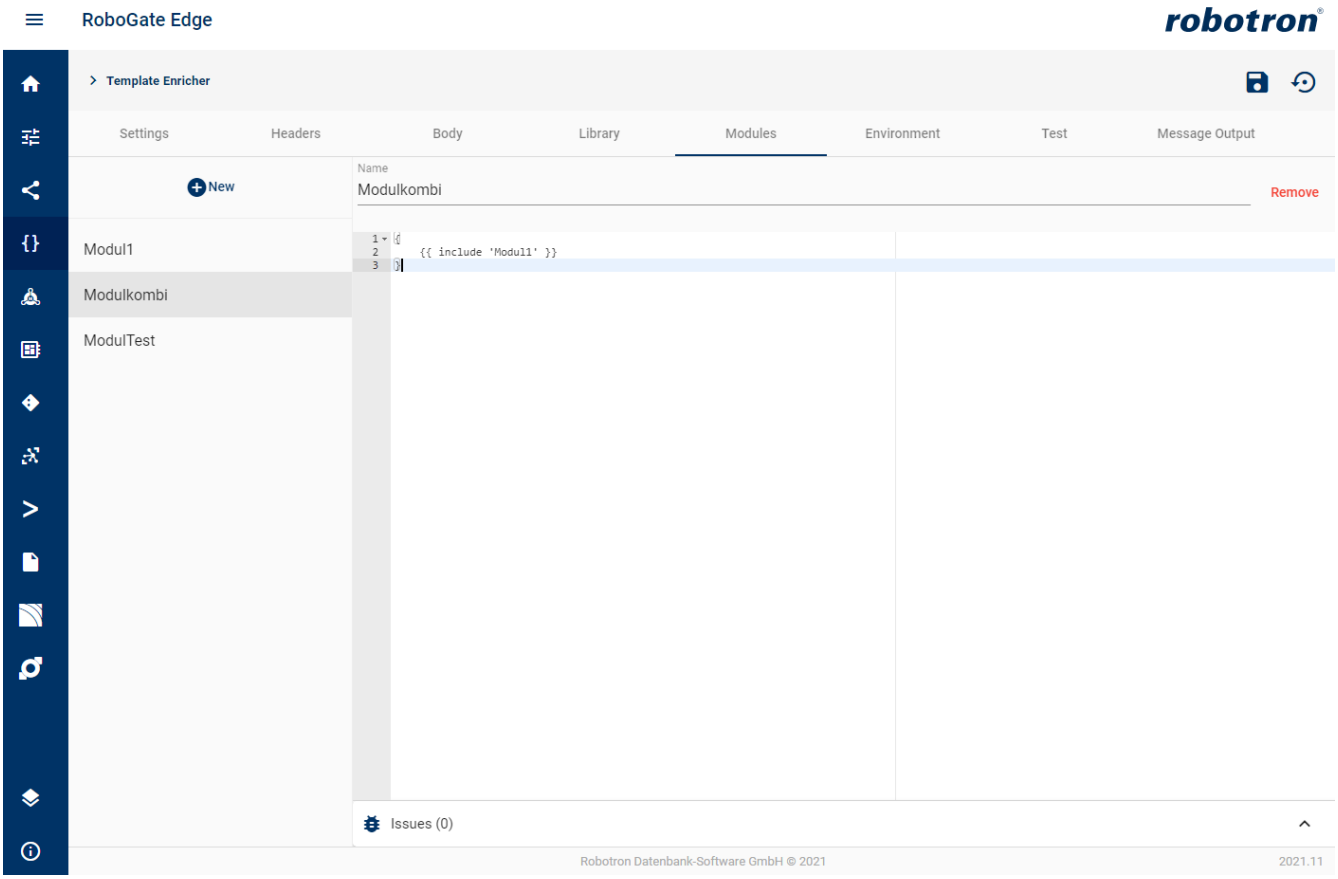


Abbildung 82. Beispielkonfiguration für Module

Environment

Im Bereich *Environment* kann ein statisches Objekt (JSON) festgelegt werden, auf welches aus dem Body zugegriffen werden kann. Abbildung 83 zeigt eine Beispielkonfiguration.

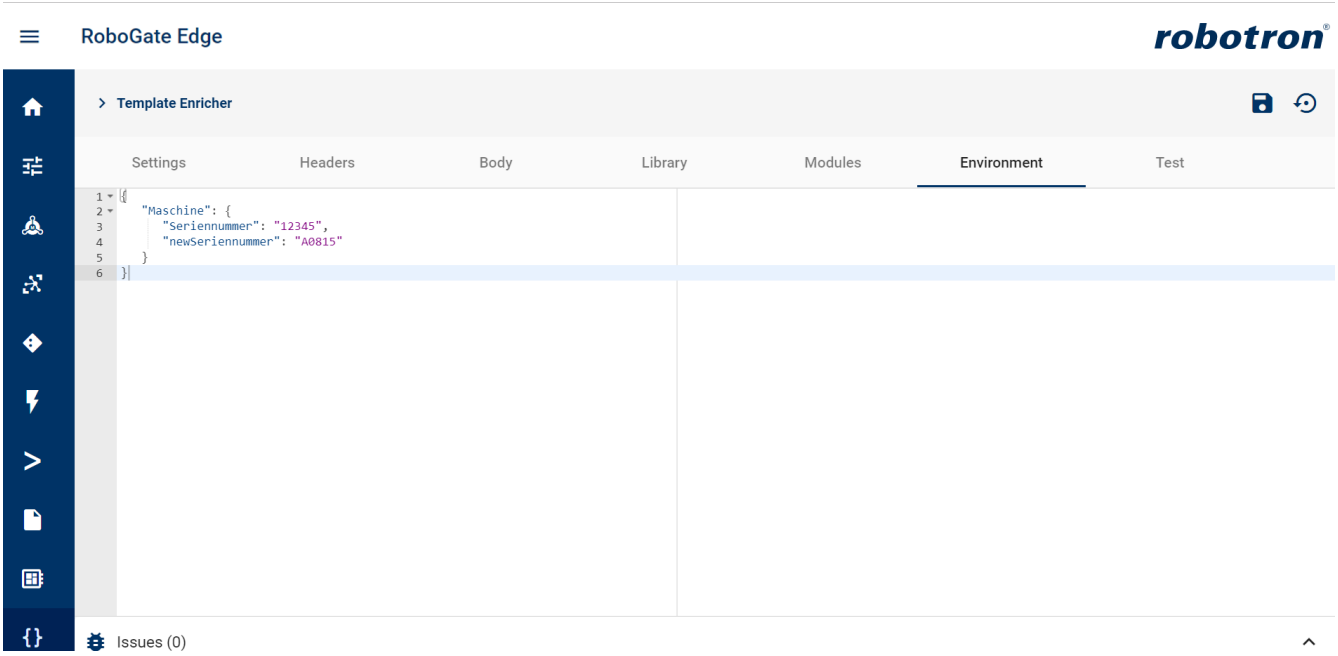


Abbildung 83. Beispielkonfiguration Environment

Test

Im Reiter *Test* kann die Funktion des konfigurierten Template Enricher getestet werden. Hierzu wird im Bereich "Ingest" eine erwartete Nachricht definiert, auf die die konfigurierten Templates angewendet werden. Nach einem Klick auf die Schaltfläche "Run" wird im Ergebnis (Result) die transformierte Nachricht zurückgegeben. Abbildung 84 zeigt ein Beispiel.

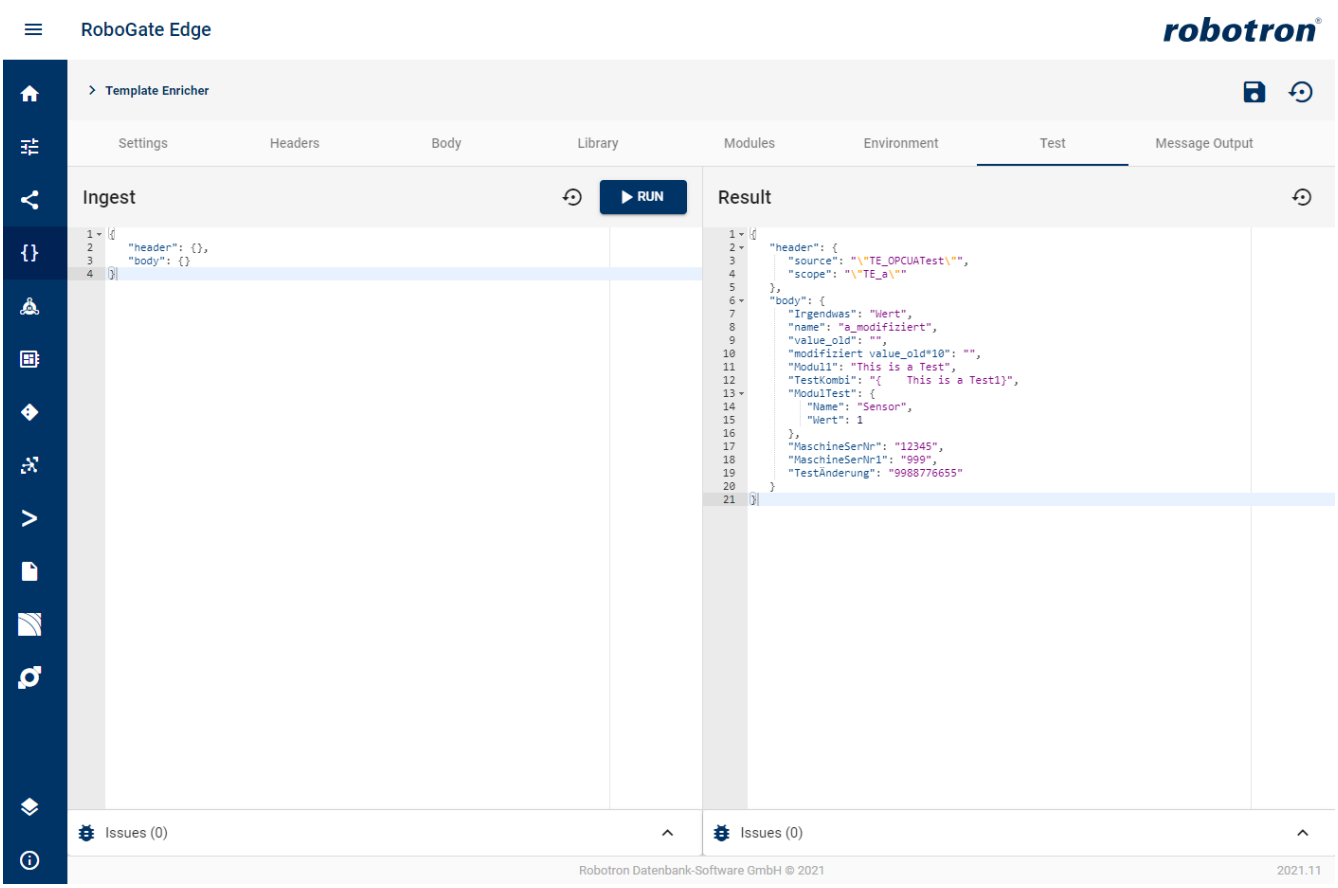


Abbildung 84. Test mit einem "valid“" json-Format

Ist beispielsweise der Body des Template Enricher nicht in einem validen JSON-Format, wird eine

Fehlermeldung ausgegeben (Abbildung 85).

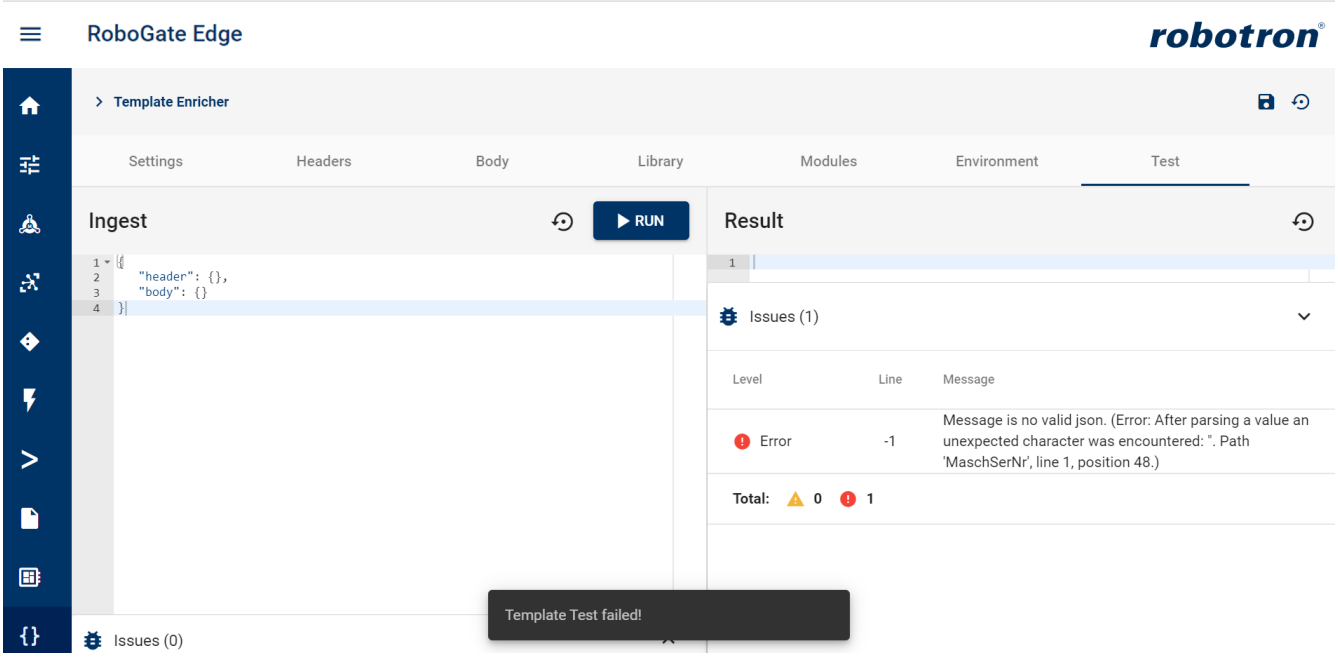


Abbildung 85. Test mit keinem "valid" json-Format

Message Output

Um die Ergebnisse des Template Enrichers auf Basis von Bedingungen an unterschiedliche Module des RoboGate Edge zu schicken, kann ein Message Output Template definiert werden, welches den Message Output als Ergebnis hat. Durch die Konfiguration des Mappings zwischen Message Output und Topic, kann die Nachricht richtig weitergeleitet werden.

Der errechnete Message Output wird auf der Ebene von Header und Body ausgegeben. Ist das Message Output Template leer, wird ein Default/Fallback Message Output (result) gewählt. Das Output Topic "result" und weitere Output Topics sind entsprechend im Bereich *Settings* unter Output Settings anzugeben.

Zeilenumbrüche und Einrückungen außerhalb der Scriban-Klammerung verbleiben nicht nicht im endgültigen Ergebnis.

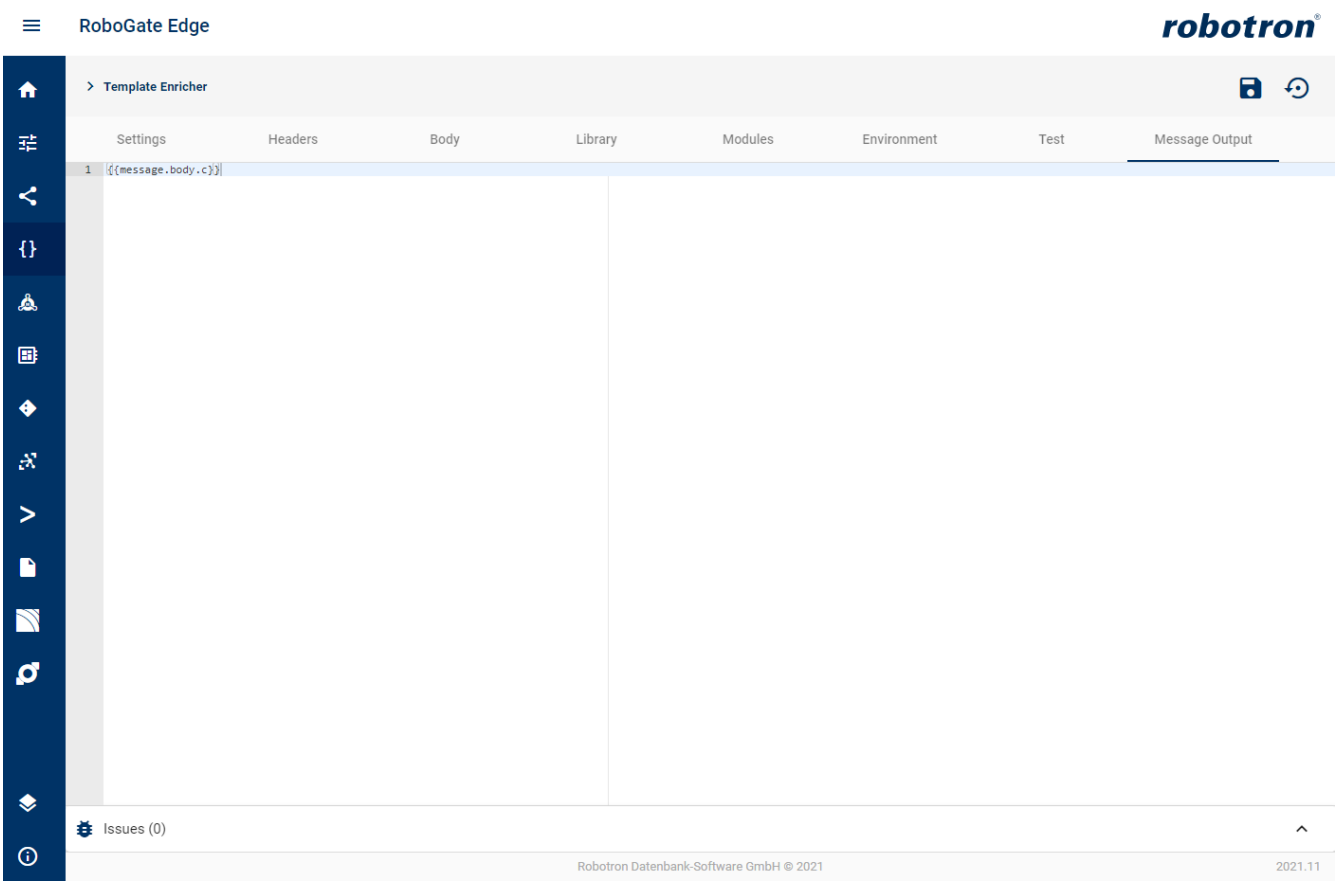


Abbildung 86. Beispielkonfiguration Message Output

Konfiguration zurücksetzen

Über die Schaltfläche in der horizontalen Menüleiste lässt sich die komplette Konfiguration des Moduls auf Werkseinstellungen zurücksetzen. Nach einem Klick auf die Schaltfläche folgt ein Bestätigungsdialog mit "Yes" und "No" zur Rückfrage ob der Vorgang wirklich ausgeführt werden soll.

> Damit wird auch die gesamte Historie auf dem RoboGate Edge Ebene gelöscht!

4.3.3. Konfiguration in JSON

```
{
    "templates": {
        "body": "{{ if message.headers.type == \"Started\" }} \n
        {{ include 'Started' }}\n
        {{ end }}\n
        {{ if message.headers.type == \"Completed\" }} \n
         {{ include 'Completed' }}\n
         {{ end }}\n
         {{ if message.headers.type == \"Changed\" }} \n
         {{ end }}"
        "headers": {},
        "modules": {
```

```
            "Started": "{\n  \"name\": \"{{ message.body.name }}\",\n  \"parameters\":
{\"Konstant\":123}",
            "Completed": "{\n  \"name\": \"{{ message.body.name }}\",\n
\"parameters\": {\"Konstant\":456}",
            "Changed": "{\n  \"name\": \"{{ message.body.name }}\",\n  \"parameters\":
{\"Konstant\":789}"
             },
        "library": ""
    },
    "$messageInputs": {
        "raw_notify": "raw"
    },
    "$messageOutputs": {
        "result": "mdb_notify"
    },
    "environment": {}
}
```

5. Output-Module

5.1. Azure IoT Hub

Das Azure IoT Hub-Modul dient dem Senden von Nachrichten an einen Azure IoT Hub.

Eingehende Nachrichten werden über das Modul gesammelt und in einen Zwischenspeicher geschrieben. Alle 2,5 Sekunden werden die enthaltenen Nachrichten gebündelt zum IoT Hub gesendet und der Zwischenspeicher geleert. Außerdem besteht die Möglichkeit einer Datenkomprimierung. Vor dem Senden wird die Batchgröße überprüft. Ist diese größer als 256 KB, wird der Batch gesplittet. Diese gesplitteten Batches werden aller 0,5 Sekunden gesendet. Das Timeout für das Senden ist auf 5 Sekunden gesetzt.

Des Weiteren besteht die Möglichkeit Funktionen für eine Cloud2Device-Kommunikation zu konfigurieren.

5.1.1. Konfigurationsparameter

Connection

Für die Verbindung mit dem Azure IoT Hub sind die in nachfolgender Tabelle beschriebenen Parameter zu konfigurieren.

Tabelle 35. Konfiguration der Connection des Azure IoT Hub-Modul

Parameter	Beschreibung
Device Connection String	Zugangspfad sowie die Authentifizierung des Gerätes am entsprechenden IoT-Hub. Zu finden ist dieser im angelegten Device des IoT Hubs im Azure Portal [4] unter Menü (links oben) → *Alle Dienste* → *IoT Hub*. Die Auswahl IoT Hub führt zu *Explore IoT Devices*. Dort muss das entsprechende Gerät gewählt und der Connection String (Primary Key) kopiert werden.
Proxy	Die Verwendung eines Proxys bei der Verbindung zwischen Gerät und Azure IoT-Hub ist optional.
Protocol	Bei der Datenüberragung zum Azure IoT Hub werden folgende Protokolle unterstützt: • MQTT (Standard) • HTTP • AMQP
Compression	Eine Kompression von Telemetry-Nachrichten vor dem Senden ist mittels Deflate oder GZip möglich. • Deflate: Datenkomprimierung mit Deflate-Algorithmus • GZip: Datenkomprimierung mit GZip-Algorithmus

Die gruppierte Anzeige für die *Connection String Info* zeigt Metainformationen aus dem Device Connection String an. Wichtig ist hier auf die Anzeige „isValid“ zu achten.

Bei der Verwendung des HTTP-Protokolls ist keine Cloud2Device-Kommunikation (siehe Section 5.1.1.2) möglich. Das bedeutet:

- Keine TWIN-Updates
- keine Cloud2Device Messages
- keine Direct Method Calls

Permissions

Es besteht die Möglichkeit, eine Cloud2Device-Kommunikation mit dem Azure IoT Hub mit folgenden Konfigurationsoptionen zu erlauben:

Receive Cloud Messages

Erlaubt das Empfangen von Cloud Messages.

Receive Configuration from Desired Properties

Erlaubt die Verarbeitung von Desired Property Updates des Device Twins. Bei dieser Option steht das HTTP-Protokoll nicht zur Verfügung.

Submit Message Headers as Azure Message Properties

Edge Message Headers werden als Message Properties übermittelt

Send ModuleStatus to Reported Properties

Erlaubt das Senden des Modulstatus in die Reported Properties des Device Twins. Bei dieser Option steht das HTTP-Protokoll nicht zur Verfügung.

Nachrichten / Topics

Topics sind logische Instanzen, in welchen zusammengehörende Nachrichten gesammelt und verarbeitet werden. Es können Topics für verschiedene Datenkategorien/-verwendungszwecke erstellt werden, welche in Tabelle 36 erläutert sind.

Um die Nachrichten eines Input-Moduls, konkreter Input Topics eines Input-Moduls, zu sammeln und zu verarbeiten, muss der Name des gewünschten Input Topics im Azure IoT Hub-Modul als "New Topic" oder "Output Topic" definiert werden.

Tabelle 36. Topics im Azure IoT Hub

Kategorie	Beschreibung
Telemetry Messages	Neue Topics können durch Eingabe des Topic Namens und einem Klick auf „+“ hinzufügt werden. Im Anschluss muss das Dateneingabe-Modul definiert werden, dessen Daten ausgewertet werden sollen.
Cloud2Device Messages (optional)	Das Output Topic legt fest, unter welchen Topic ankommende Cloud Messages publiziert werden.

Kategorie	Beschreibung
Direct Methods	Die Methoden können über Azure aufgerufen werden. Der Name der Direct Method kann in der lokalen RoboGate Edge UI erfasst werden. Es erfolgt ein Mapping zwischen Azure Direct Method und Edge Method.
Blob Upload	Der IoT Hub bietet die Möglichkeit, größere Datenmengen in einen verbunden Azure Blob Storage zu speichern. Eingehende Nachrichten werden als Datei im Blob Storage abgelegt. Name und Speicherort kann mit einem Pattern festgelegt werden.

Für den Blob Upload ist ein bestimmtes Pattern anzugeben. Dabei kann sich an nachfolgender Übersicht orientiert werden.

Default Pattern: {prefix}{ticks}
Folder Separator: /

Tabelle 37. Übersicht Pattern

Variable	Content Default
{year}	Current UTC Year (4 Digits)
{month}	Current UTC Month (2 Digits)
{day}	Current UTC Day (2 Digits)
{hour}	Current UTC 24h Hour (2 Digits)
{minute}	Current UTC Minute (2 Digits)
{second}	Current UTC Second (2 Digits)
{ticks}	Count of 100ns Intervals since 0001-01-01 00:00:00 UTC (64-bit number)
{prefix}	Edge Message Header prefix
{name}	Edge Message Header name
{ext}	Edge Message Header ext (default blob)
{hub}	IoT Hub Name
{host}	IoT Hub DeviceId
{Source}	Edge Message Header Source (default any)
{scope}	Edge Message Header scope (default any)

5.1.2. Device Provisioning (DPS)

Um eine Bereitstellung ohne Konfiguration des IoT-Hubs bzw. ohne werkseitige Konfiguration zu erreichen, bietet Azure den *Device Provisioning Service* an. DPS ermöglicht eine skalierbare und

sichere Bereitstellung von mehreren RoboGate Edge.

Damit der DPS verwendet werden kann, muss der "Device Connection String" in der Connection-Konfiguration leer sein. Die Verbindungszeichenfolge wird dann vom DPS bezogen. Damit diese bereit gestellt werden kann, müssen mindestens zwei Einstellungen der folgenden DPS Konfiguration übergeben werden:

- Scope
- EnrollmentType (Art der Bereitstellung)

Tabelle 38. Konfiguration Device Provisioning Service (DPS)

Kategorie	**Beschreibung**
Scope ID	Legt den DPS ID-Bereich fest. Diese Einstellung muss zwingend übergeben werden. Dieser muss von Azure bezogen werden. (ID-Bereich)
Mode	EnrollmentType: • **Group:** Gilt für eine Gruppenbreitstellung. Dabei muss zwingend der PrimaryKeyBase64 mitgegeben werden, da dieser als Authentifikation verwendet wird. Dieser symmetrische Schlüssel wird von Azure erzeugt und kann von dort bezogen werden. • **Individual:** Gilt für eine individuelle Bereitstellung. Sollte hier eine Breitstellung ohne Angabe des "PrimaryKeyBase64" erfolgen muss dieser Schlüssel vorher vom Gerät abgeleitet werden (Seriennummer /Geräte ID). Das kann mit Hilfe der Edge-Methode "robogate.dps.hash" erfolgen. Der dort erzeugte Base64-Schlüssel muss dann in Azure eingetragen werden. Außerdem muss die Geräte-ID in Azure angeben werden, da bei Abrufen des IOT-Hubs der Primary Kay mit der Geräte ID verknüpft wird. Die Geräte ID ist mit "robogate.security.GetClientId" abrufbar.
ID	Die ID ist folgendermaßen anzugeben: • **Group**, optional, individuell wählbar • **Individual**, optional (Geräte ID)
Symmetric Key (base64)	Ist ein Schlüssel zur Authentifzierung des Gerätes. Je nach Registrierungs-Breitstellung-Konfiguration kann dieser Key weggelassen werden, da er automatisch erzeugt wird (siehe Enrollment Type: Individual).
Advanced settings	Endpunkt (URL) für den DPS. Wenn dieser nicht übergeben wird, findet der default-Endpoint Verwendung.

5.1.3. Konfiguration in der UI

Nach Auswahl des Moduls Azure IoT Hub auf der Startseite oder über die linke Menüleiste der RoboGate Edge UI kann mit einem Klick auf „New“ eine Verbindung zu einem Azure IoT Hub konfiguriert werden.

Connection konfigurieren

Im ersten Schritt ist die Verbindung hinsichtlich der zuvor beschriebenen Parameter zu konfigurieren. Neue Topics für die Auswertung der Daten aus den Input-Modulen können per Klick auf „+“ hinzugefügt werden. Beispielhaft ist in Abbildung 87 das Output-Topic „*Modbus_1*“ eingegeben.

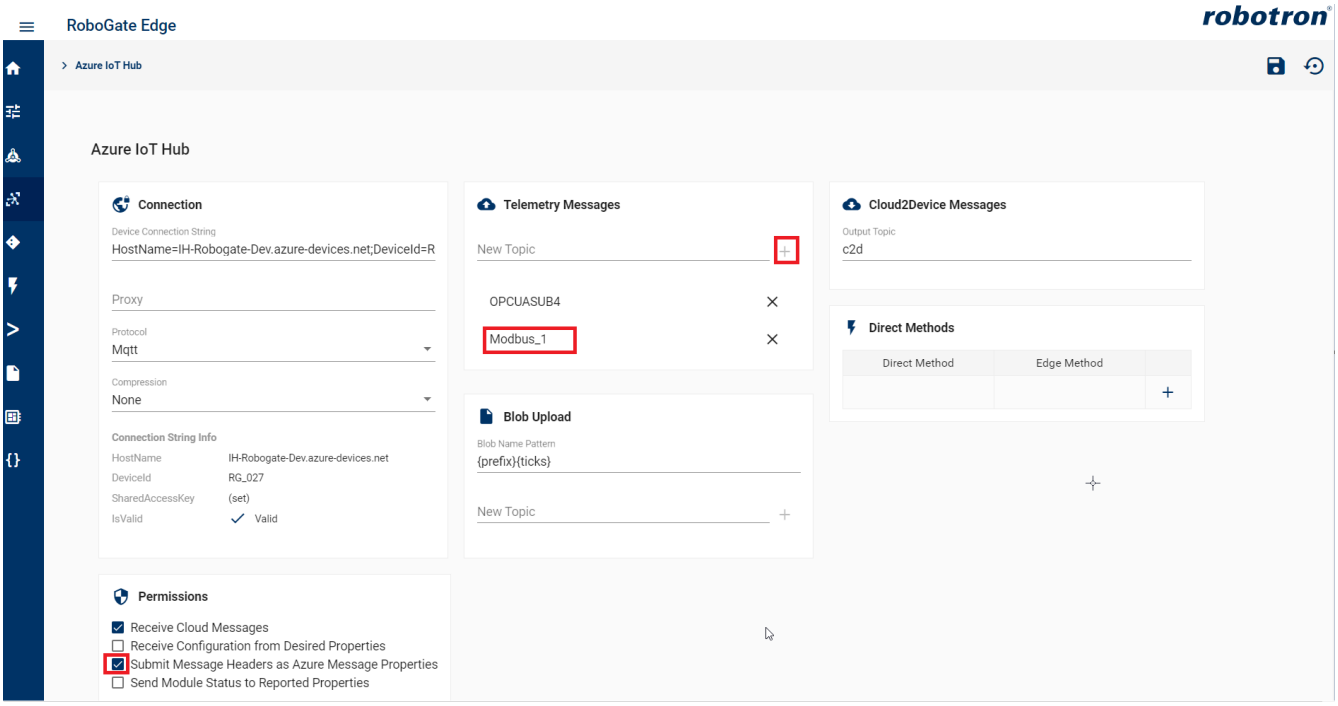


Abbildung 87. Hinzufügen eines neuen Topics Modbus_1

Nach Fertigstellung der Konfiguration des Azure IoT Hub kann diese gespeichert werden (Abbildung 88). Durch das Speichern der Modul-Konfiguration werden die Daten der ausgewählten Topics an Microsoft Azure gesendet und stehen zur Auswertung im IoT Hub bereit.

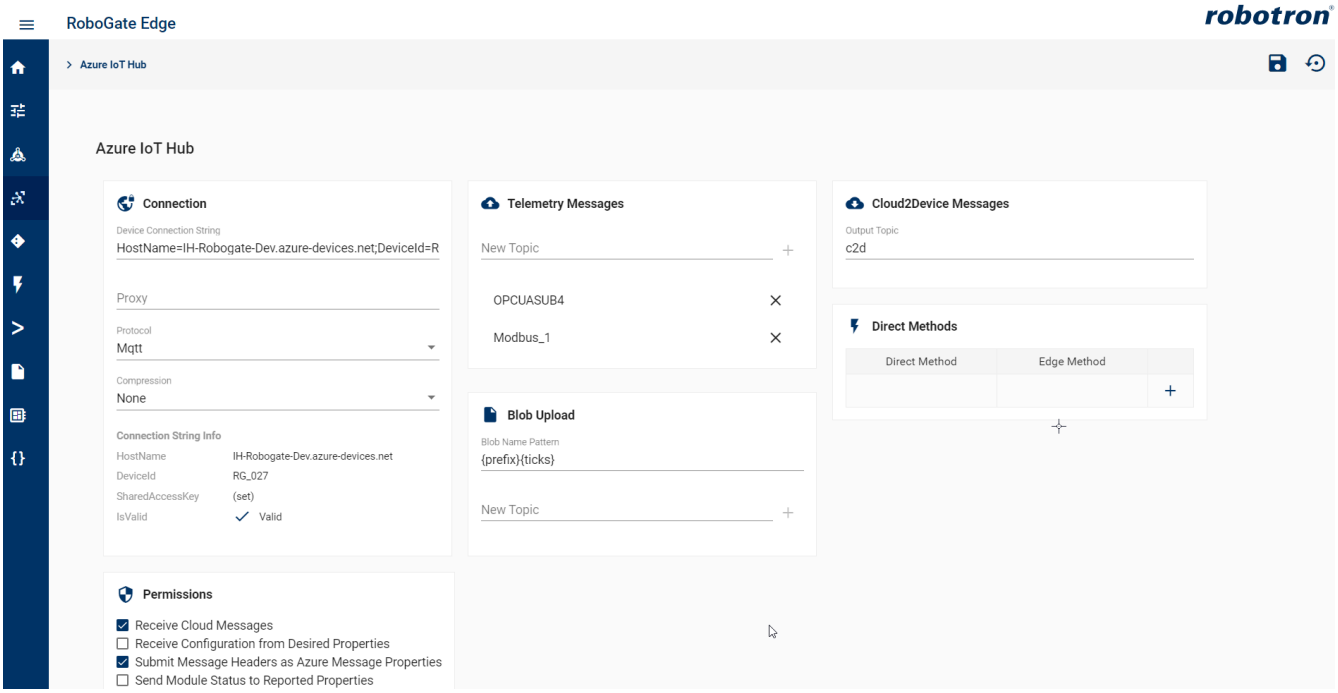


Abbildung 88. Speichern der Konfiguration des Azure IoT Hub-Moduls

Im Bereich "Module State" lässt sich erkennen, ob das Modul eine Verbindung zu Azure herstellen konnte.

Topic löschen

Topics können aus der Konfiguration des Azure IoT Hub-Moduls entfernt werden. Dazu muss das Kreuz ✕ hinter dem zu löschenden Topic geklickt werden. Dadurch wird die markierte Zeile gelöscht (Abbildung 89). Durch das anschließende Klicken auf den Speicher-Button 💾 wird die geänderte Konfiguration übernommen.

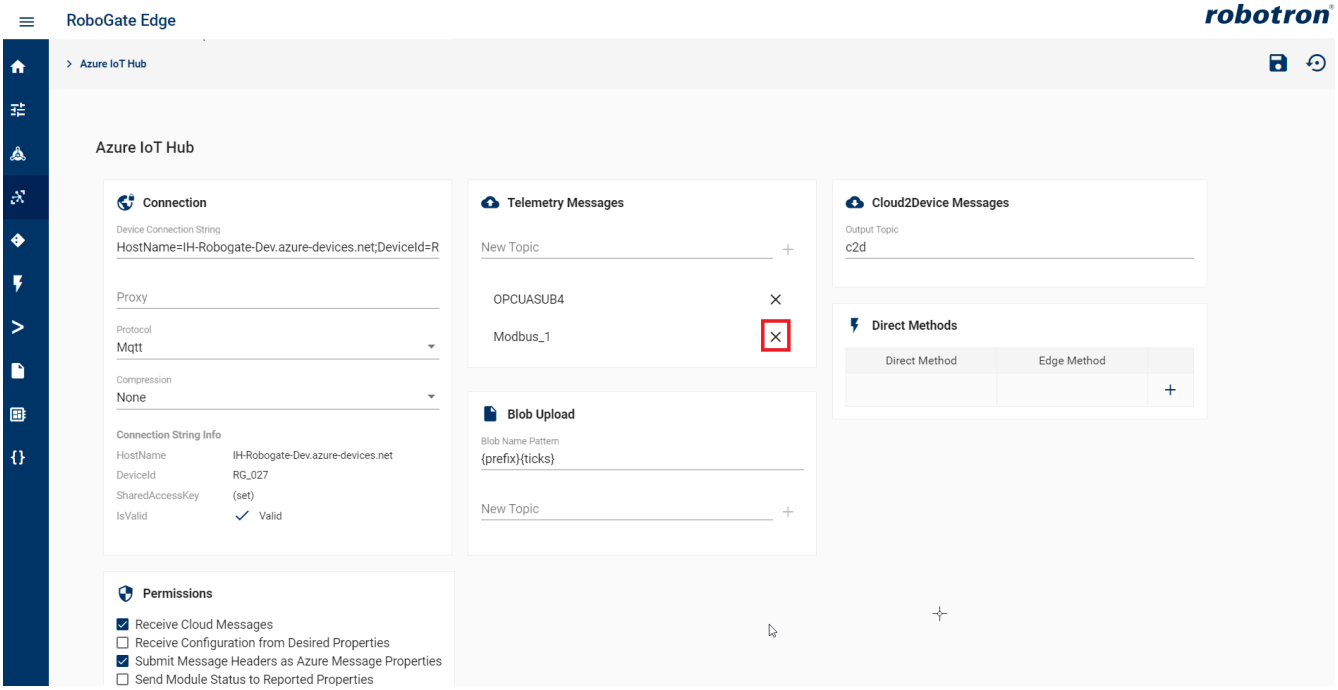


Abbildung 89. Löschen eines Azure IoT Hub-Topics

Konfiguration zurücksetzen

Über die Schaltfläche in der horizontalen Menüleiste lässt sich die komplette Konfiguration des Moduls auf Werkseinstellungen zurücksetzen. Nach einem Klick auf die Schaltfläche folgt ein Bestätigungsdialog mit "Yes" und "No" zur Rückfrage ob der Vorgang wirklich ausgeführt werden soll.

> Damit wird auch die gesamte Historie auf dem RoboGate Edge gelöscht!

5.2. MQTT

Das MQTT-Modul dient der Kommunikation mit einem MQTT Broker.

Folgende Funktionen werden unterstützt:

- Nachrichtenversand an den Broker (Publish)
- Nachrichtenempfang vom Broker (Subscribe)
- Optional TLS Verschlüsselung
- QoS und Retain Flag individuell pro Nachricht

5.2.1. Konfigurationsparameter

Connection

Die MQTT-Konfiguration beinhaltet die Definition der Zugangsdaten zu einem MQTT Server. Diese setzen sich zusammen aus den Angaben der IP, Benutzername und Passwort. Eine zusätzliche Verschlüsselung (TLS) ist möglich. Tabelle 39 enthält die Erläuterungen der einzelnen Konfigurationsparameter.

Tabelle 39. Konfiguration Connection MQTT Modul

Parameter	Beschreibung
Enabled/Disabled-Switch	Die Verbindung des Moduls lässt sich aktivieren oder deaktivieren. Ist das Modul inaktiv werden einlaufende Nachrichten verworfen.
Hostname	Broker Hostname oder IP-Adresse
Client ID	Client ID für die Client Zuordnung am MQTT Broker (s. MQTT-Spezifikation)
Clean Session	Wenn aktiviert, werden keine Client spezifischen Informationen (Subscribed Topics etc.) beim Broker gespeichert (s. MQTT-Spezifikation). Muss aktiviert sein, wenn Client-ID leer ist.
Username	Login Daten für den Zugriff auf den Broker
Passwort	Login Daten für den Zugriff auf den Broker
Port	Broker Port (TCP/IP). Wenn das Feld leer gelassen ist, werden die Standard Ports 1883 (ohne TLS) und 8883 (mit TLS) verwendet.

Parameter	Beschreibung
Queue Size	Größe der Queue, d.h. Anzahl der Nachrichten, die vorgehalten werden sollen.
TLS	Wenn aktiv, wird eine verschlüsselte Verbindung zum Broker aufgebaut.
Allow untrused Certificates	Allow untrusted Certificates: Verbindung zum Broker wird auch bei nicht vertrauenswürdigem Zertifikat herstellen.

Nachrichten / Topics

Im Bereich MQTT Publishing wird definiert, welche Topics und Nachrichtenformate im Modul verarbeitet werden.

MQTT Server Topic

Tabelle 40. Konfiguration MQTT Publishing

Parameter	Beschreibung
Topic Name	Eingabe des Ziel Topic am Broker. Es kann auf definierte Platzhalter zugegriffen werden (siehe Tabelle 41).
Default Message Retain Flag	Wenn aktiv, wird das Retain Flag für die an den Broker übertragenen Nachrichten standardmäßig gesetzt. Die Einstellung wird überschrieben mittels Edge Message Header "retainFlag" (true/false) bzw. individuell für jede Nachricht (allgemeine MQTT Spezifikation).

Um das Topic dynamisch zur Laufzeit anpassen zu können, kann auf die Platzhalter in der folgenden Tabelle zugegriffen werden. Der Platzhalter aus Spalte 1 wird mit geschweiften Klammern umschlossen (Whitespaces bedacht setzen), um verarbeitet zu werden.

Beispiel:

Ein Wert `{year}/{month}/{header:source}/data` würde exemplarisch zu einem Laufzeit Topic extrahiert `2020/06/MaschineNameXY/data`.

Tabelle 41. Übersicht Topic Platzhalter

Platzhalter	Beschreibung	Beispiel
"year"	Kalenderjahr, vierstellig	"2020"
"month"	Monat, zweistellig ggf. mit führender '0'	"06"
"day"	Tag im Monat, zweistellig ggf. mit führender '0'	"02"
"hour"	Stunde des Tages, zweistellig ggf. mit führender '0'	"05"

Platzhalter	Beschreibung	Beispiel
"minute"	Minute der Stunde, zweistellig ggf. mit führender '0'	"59"
"second"	Sekunde der Minute, zweistellig ggf. mit führender '0'	"59"
"ticks"	siehe: Microsoft Docs	
"ext"	Headers ext Feld	
"name"	Headers name Feld	
"prefix"	Headers prefix Feld	
"source"	Headers Source Feld, sonst "any"	"192.12.1.11"
"scope"	Headers Scope Feld, sonst "any"	"pollgroup2"
"hostname"	Hostname des RoboGate Edge	"RG-231889000100401"
"header: <name>"	Message Header Zugriff auf den Key name, sonst "any". Beispielsweise "{header:foo}".	beliebiger String

Edge Topics

Die Edge Topics bezeichnen die internen Nachrichtenkanäle. Die empfangenen MQTT-Nachrichten lassen sich durch die Definition der Topics in anderen RoboGate Edge-Modulen weiterverarbeiten.

Tabelle 42. Konfiguration Edge Topics

Parameter	Beschreibung
New Topic	Auswahl der abonnierten Topics, die in den Input-Modulen definiert wurden, die an den MQTT Broker übertragen werden sollen.

MQTT Subscribing

Das RoboGate Edge hat die Möglichkeit, Nachrichten von einem MQTT Broker zu abonnieren. Einer *MQTT Topic Expression* (Quelle) kann ein *Output Edge Topic* (Ziel) zugewiesen werden. Von dort aus sind die Nachrichten für andere Module zugreifbar. Die Topic Expression ist nach MQTT-Spezifikation einzutragen.

5.2.2. Konfiguration in der UI

Nach Auswahl des Moduls MQTT auf der Startseite oder über die linke Menüleiste der RoboGate Edge UI kann das MQTT-Modul aufgerufen und konfiguriert werden.

Connection konfigurieren

Im ersten Schritt ist die Verbindung mit den oben beschriebenen Parameter zu konfigurieren. Anschließend können die gewünschten Topics im MQTT Publishing und MQTT Subscribing angelegt

werden. Hierzu geben Sie die Namen der Topics, die das MQTT-Modul verarbeiten soll, bei den Input Edge Topics ein und definieren entsprechend Ihrer Anforderungen einen Topic Namen für den MQTT Server.

Nach Fertigstellung der Konfiguration des MQTT-Moduls kann diese gespeichert werden (Abbildung 90). Durch das Speichern der Modul-Konfiguration werden die Daten der ausgewählten Topics an den konfigurierten Client gesendet und stehen zur Auswertung z.B. im MQTT Explorer bereit (Abbildung 91).

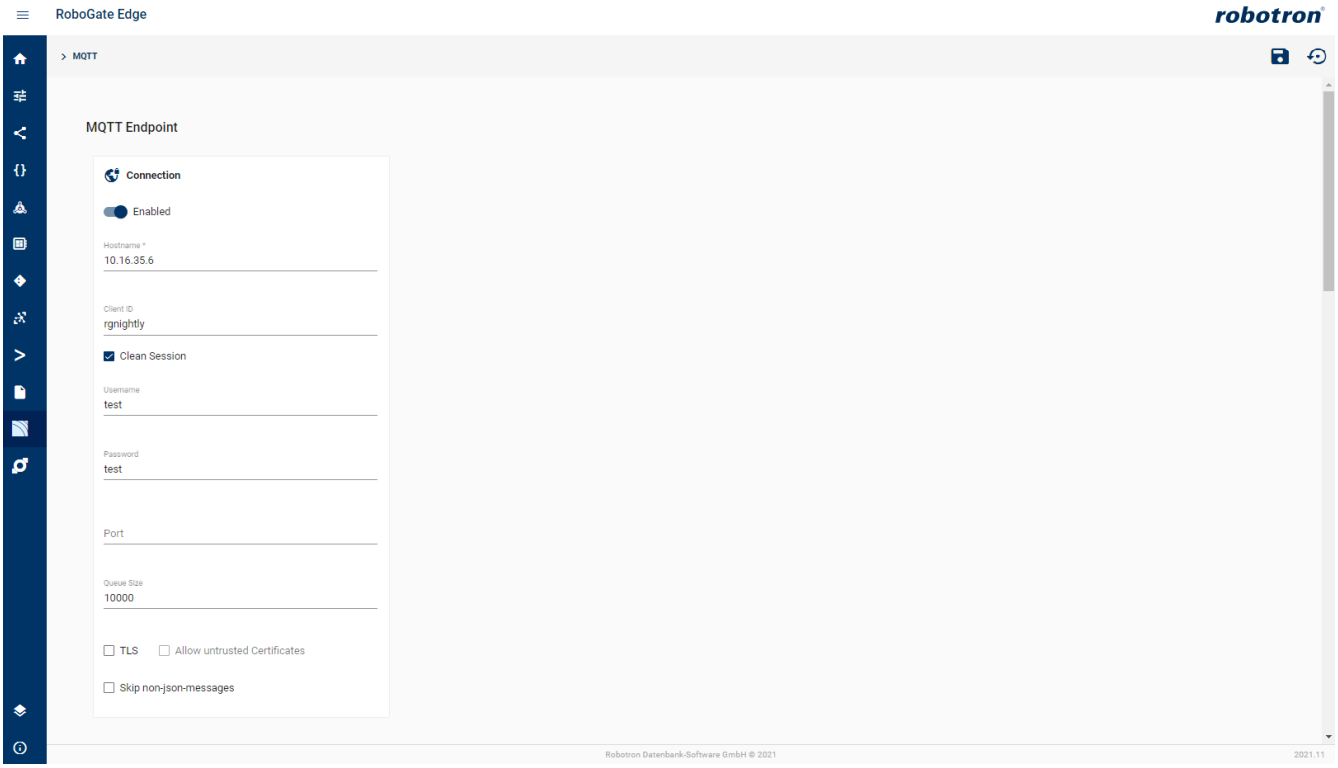


Abbildung 90. Konfiguration des MQTT Moduls

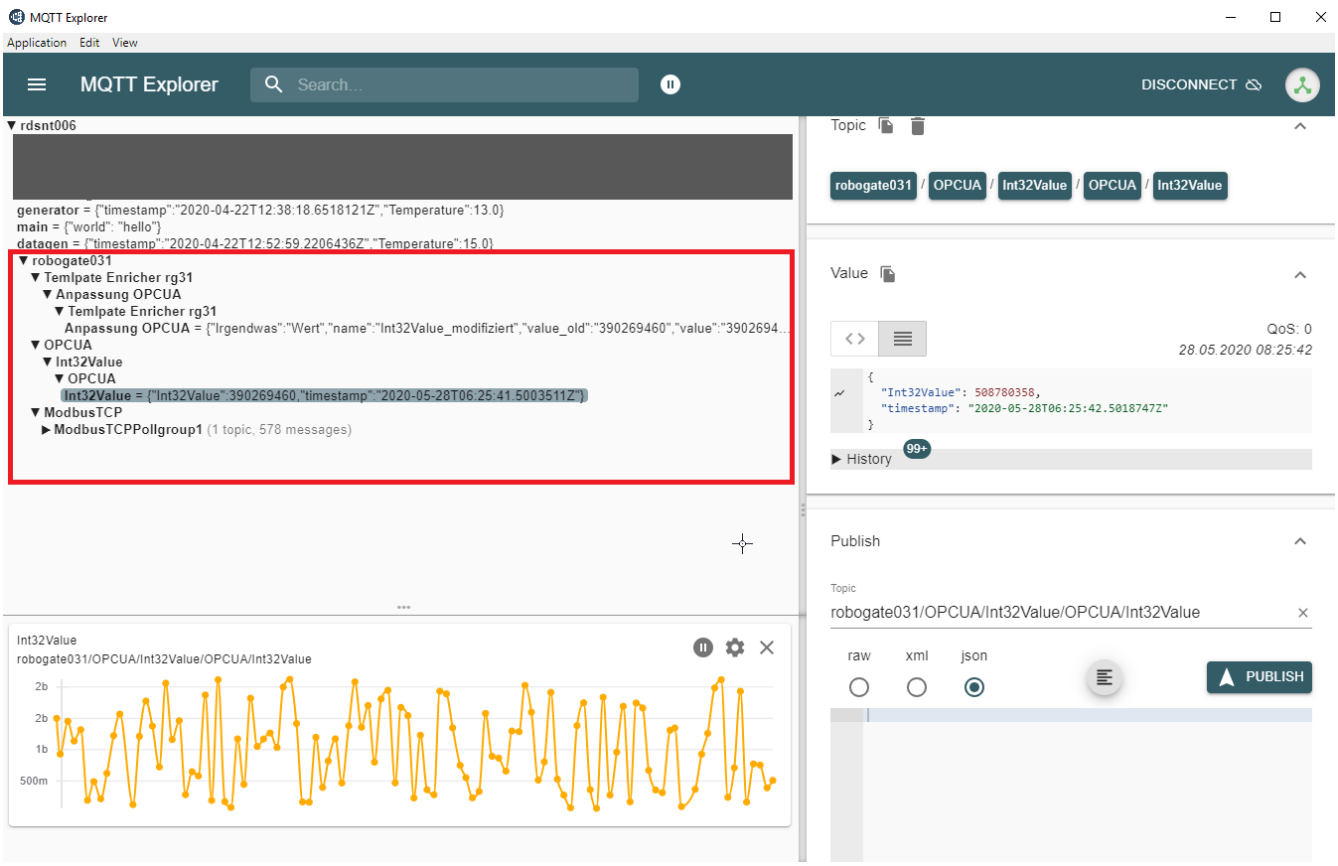


Abbildung 91. Auswertung der gesendeten Daten im MQTT Explorer

Im Bereich "Module State" lässt sich erkennen, ob das Modul eine Verbindung zum MQTT Server herstellen konnte.

Topics löschen

Topics können aus der Konfiguration des MQTT-Moduls entfernt werden. Dazu muss das ✕ hinter dem zu löschenden Topic geklickt werden. Dadurch wird die markierte Zeile entfernt. Durch das anschließende Klicken auf den Speicher-Button (Diskette) wird die geänderte Konfiguration übernommen (Abbildung 92).

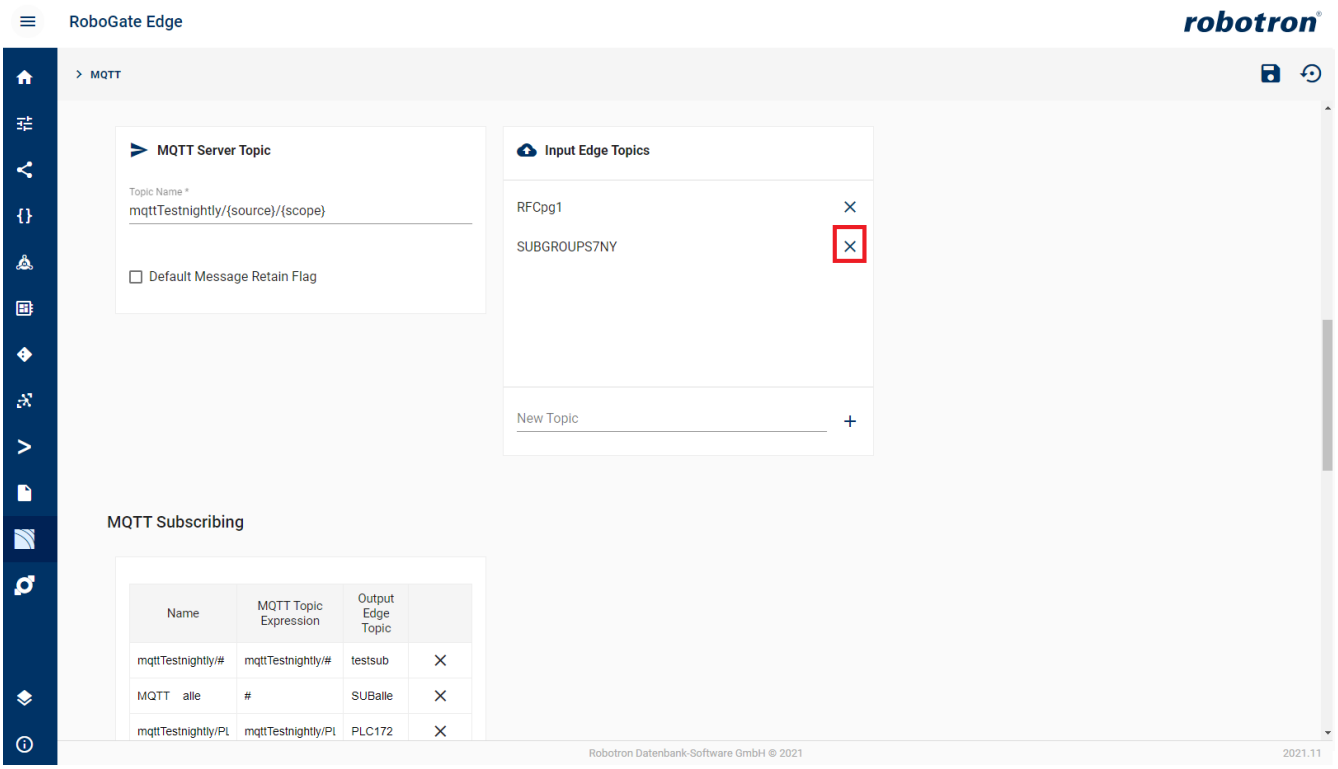


Abbildung 92. Löschen eines Topics im MQTT Modul

Konfiguration zurücksetzen

Über die Schaltfläche in der horizontalen Menüleiste lässt sich die komplette Konfiguration des Moduls auf Werkseinstellungen zurücksetzen. Nach einem Klick auf die Schaltfläche folgt ein Bestätigungsdialog mit "Yes" und "No" zur Rückfrage ob der Vorgang wirklich ausgeführt werden soll.

> Damit wird auch die gesamte Historie auf dem RoboGate Edge gelöscht!

5.3. OPC UA Server

Das Modul stellt einen OPC UA Server bereit. Dieser OPC UA Server bzw. der OPC UA Node Tree wird aus EdgeMessages, d.h. ein- bzw. ausgehenden Nachrichten anderer Module des RoboGate Edge, befüllt. Der konfigurierte Node Tree kann dann über einen OPC UA Client (z. B. UA Expert) und Node Browsing die definierten Nodes darstellen.

5.3.1. Settings

Im nachfolgenden Beispiel werden Daten des RFC-Moduls als EdgeMessages an das OPC UA Server-Modul gesendet, und schließlich ein Node Tree aus diesen Nachrichten befüllt.

Zunächst kann dem OPC UA Server ein Computer und Application Name im Bereich "General" vergeben werden (Abbildung 93).

Um die Daten des RFC-Moduls in den OPC UA Server einzubringen, wird die Pollgroup "RFCpg1" als Input Edge Topic im Bereich "Settings" hinzugefügt.

Je nach Bedarf, können Sicherheitseinstellungen vorgenommen werden.

Um die Konfiguration zu übernehmen, muss diese über die Schaltfläche gespeichert werden.

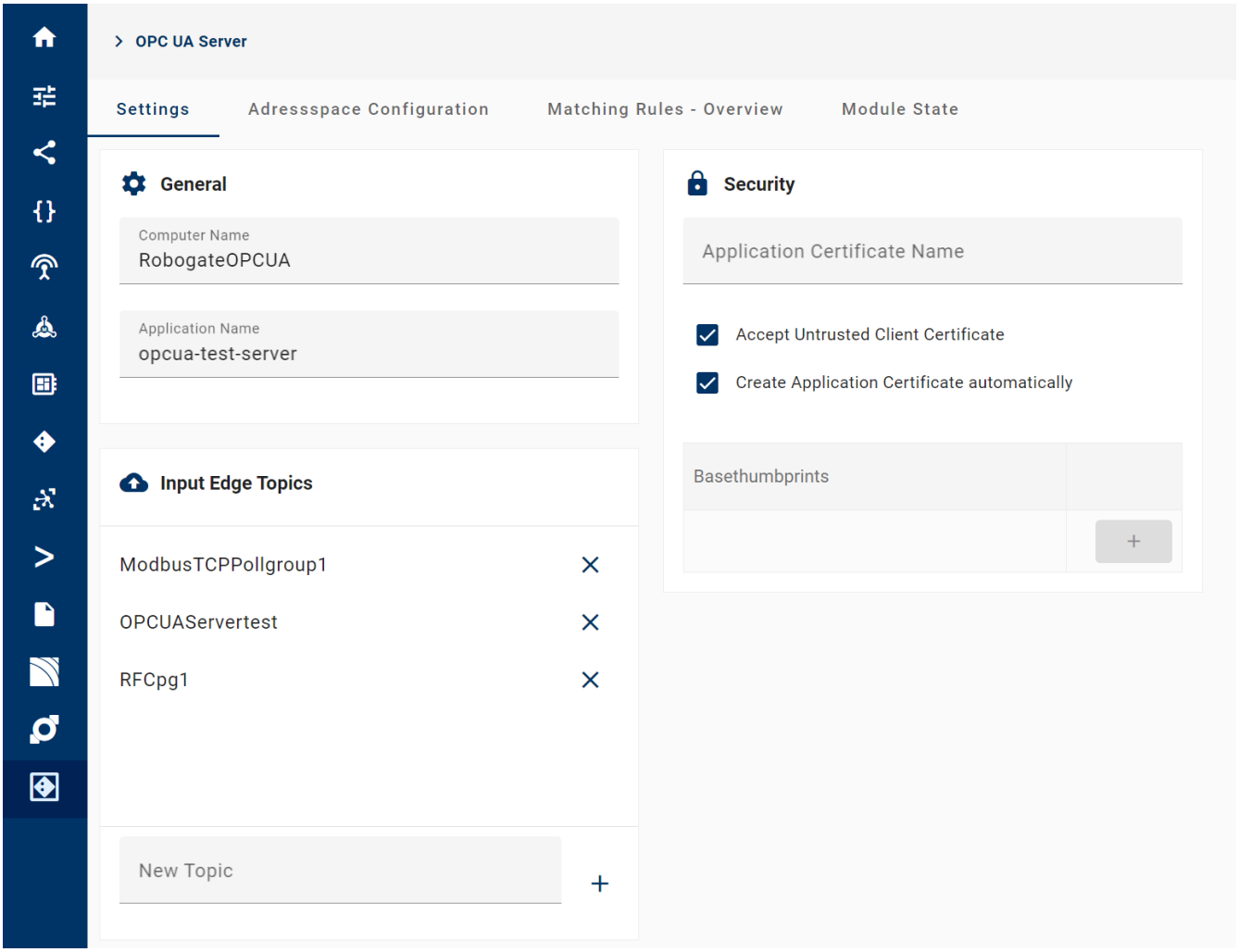


Abbildung 93. OPC UA Server - Settings

Parameter

Zur Erstellung des Zertifikats des OPC UA Servers können der Computer Name und Application Name definiert werden.

Tabelle 43. General

Parameter	Beschreibung
Computer Name	optional, Name unter dem der Endpunkt des OPCUA Servers über einen Client erreichbar ist, muss ein gültiger DNS-Name sein default wird der hostname des RoboGate Edge verwendet
Application Name	Identifikationsname des OPC UA Servers

Die Verbindung zum OPC UA Server ist über eine Client-Autorisierung mittels Application Certificate möglich. Es lässt sich konfigurieren, ob der Server automatisch bei Start ein Zertifikat erstellt und ob und welche Client IDs akzeptiert werden, um sich mit dem Server zu verbinden.

Tabelle 44. Security

Parameter	Beschreibung
Application Certificate Name	optional, Name des Applikations-Zertifikats
Accept Untrusted Client Certificate	Wenn aktiviert, akzeptiert der Server automatisch die ID des sich verbindenden Clients
Create Application Certificate automatically	Wenn aktiviert, erzeugt sich der konfigurierte OPC UA Server ein selbst signiertes Applikations-Zertifikat, mit dem konfigurierten Computer und Application Name und Endpunkt-Informationen.

Im Bereich Basethumbprints lassen sich die IDs der OPC UA Clients (thumbprints), die zugelassen werden, hinzufügen. Jedes valide Client-Zertifikat wird automatisch in diese Trust List aufgenommen. Weiterhin muss definiert werden, aus welchen EdgeMessages der OPC UA Server befüllt wird (Input Edge Topics).

5.3.2. Adress Space Configuration

Im Bereich "Adress Space Configuration" kann der Node Tree des OPC UA Server konfiguriert werden.

Mit einem Klick auf die Hinzufügen-Schaltfläche + lässt sich ein neuer Folder anlegen. Im Beispiel in Abbildung 94 wurde der Folder "Datablock_RFC_N" mit den jeweiligen Namensgebungen angelegt. Anschließend wurden weitere Folder, u.a. der Folder "30_string_number", unter diesem angelegt.

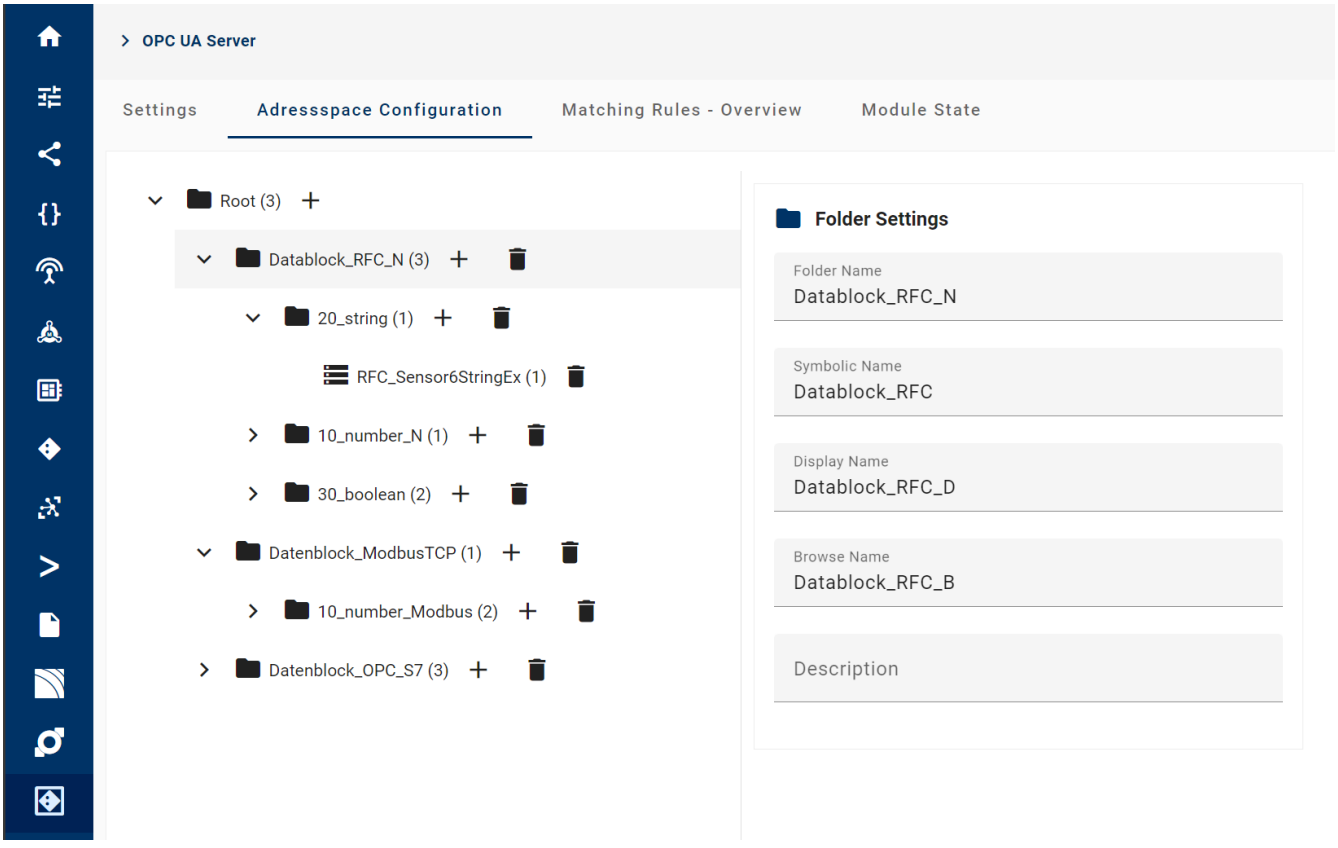


Abbildung 94. OPC UA Server - Adressspace Configuration - Folder Settings

Unter jedem Folder (Ausnahme Root-Folder) lässt sich über + ein Node hinzufügen und konfigurieren. Unter den Node Settings lassen sich analog zu den Folder Settings die Namen des

Nodes und zusätzlich der Datentyp und ggf. ein Initialwert konfigurieren (Abbildung 95).

Im Bereich Matching Rules wird definiert, aus welchen EdgeMessages die Daten des ausgewählten Nodes stammen. Hierzu müssen diese EdgeMessages zuvor, in unserem Beispiel im RFC-Modul, angelegt worden sein bzw. anschließend noch angelegt werden (Abbildung 96). Source bezieht sich auf den angelegtem RFC Server, Scope auf die darunter eingerichtete Pollgroup.

Um die Konfiguration zu übernehmen, muss diese über die Schaltfläche gespeichert werden.

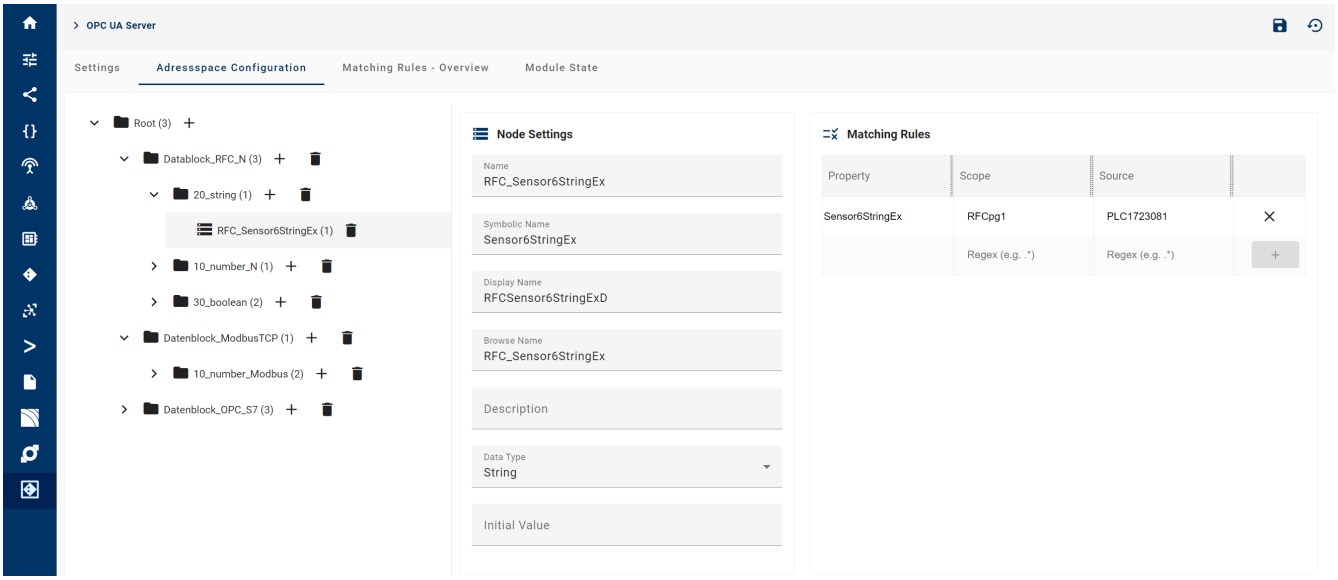


Abbildung 95. OPC UA Server - Adressspace Configuration - Node Settings

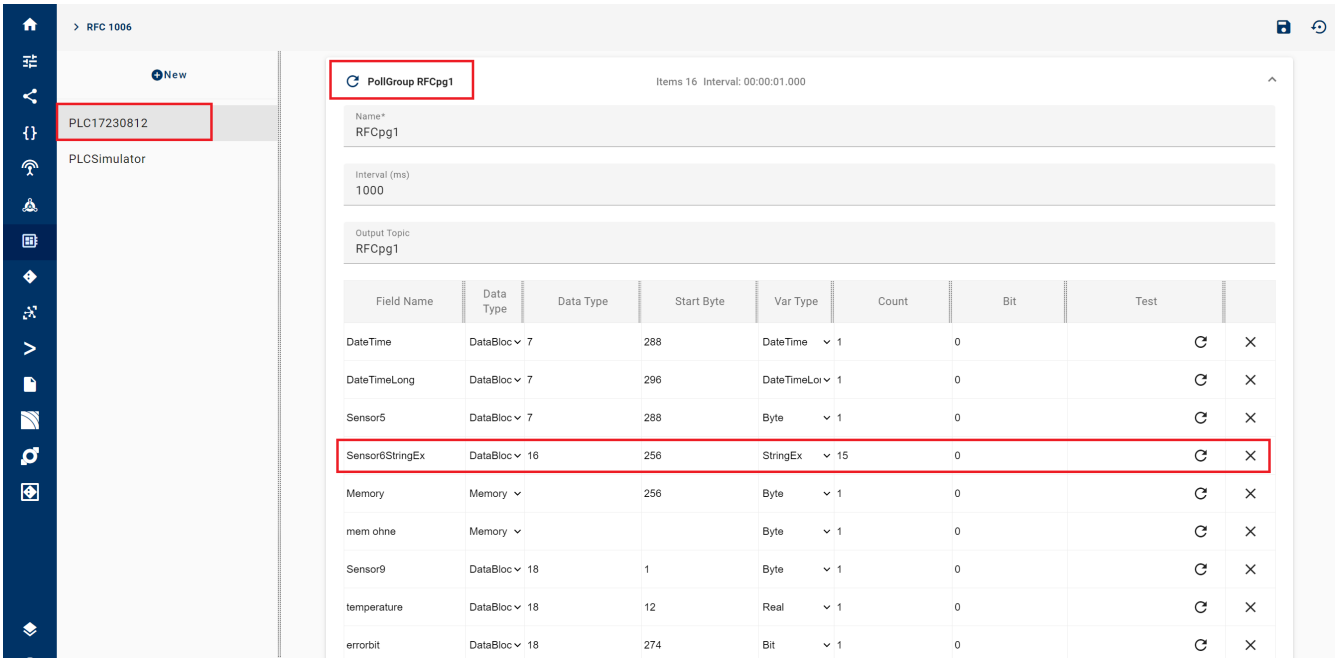


Abbildung 96. Konfiguration des RFC-Moduls zur Einbindung in den OPC UA Server

Parameter

Zur Erstellung des Node Tree müssen Folders und die darin enthaltenen Nodes mit folgenden Parametern konfiguriert werden. Die Folders ermöglichen dabei eine Strukturierung des Node Tree.

Tabelle 45. Folder Settings

Parameter	Beschreibung
Folder Name	Name/Key des Folder-Objekt
Symbolic Name	optional, eindeutiger Name des Folder (default wird der definierte Name des Folder verwendet)
Display Name	optional, beliebiger Name des Folder der im Node Browsing angezeigt wird (default wird der Browse Name verwendet)
Browse Name	optional, eindeutiger Name, der im Node Browsing angezeigt wird (default wird der Symbolic Name verwendet)
Description Name	optional, Freitext zur Beschreibung des Folder

Tabelle 46. Node Settings

Parameter	Beschreibung
Name	Name/Key des Node-Objekt
Symbolic Name	optional, eindeutiger Name des Node, entspricht dem Target Node in den Matching Rules (default wird der definierte Name des Node verwendet)
Display Name	optional, beliebiger Name des Node der im Node Browsing angezeigt wird (default wird der Browse Name verwendet)
Browse Name	optional, eindeutiger Name, der im Node Browsing angezeigt wird (default wird der Symbolic Name verwendet)
Description	optional, Freitext zur Beschreibung des Node
Data Type	Definition bzw. Auswahl des Datentyps des konfigurierten Node: String, Number, Byte, Boolean
Initial Value	Initialer Wert des Node der bei Start des Server in den Node geschrieben wird

5.3.3. Matching Rules Overview

Im Bereich "Matching Rules - Overview" wird eine Liste der konfigurierten Nodes inkl. Source und Scope aufgeführt (Abbildung 97). Auch hier lassen sich analog zum Vorgehen unter Adress Space Configuration Rules konfigurieren, wobei darauf zu achten ist, den korrekten Target Node anzugeben, da dieser in dieser Ansicht nicht vorausgewählt ist.

Die Abarbeitung der Regeln erfolgt top down. Sind zwei Regeln für den gleichen Target Node konfiguriert, so wird am Ende der letzte Wert in den entrpechenden Node geschrieben. Um diese Reihenfolge zu verändern, lassen sich die Regeln via drag & drop verschieben.

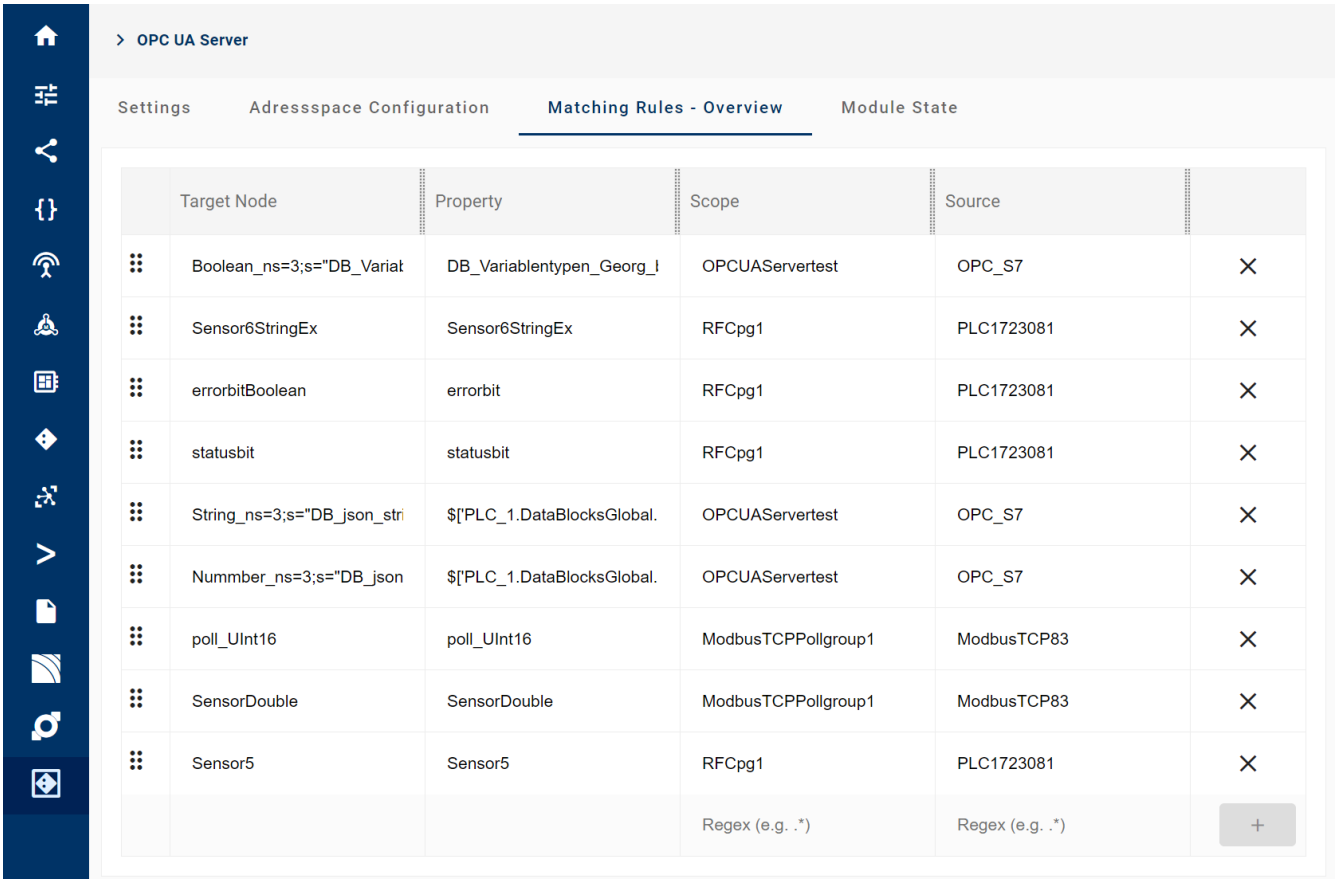


Abbildung 97. OPC UA Server - Matching Rules - Overview

Parameter

Das OPC UA Server-Modul extrahiert anhand einer Liste von Regeln - den Matching Rules - einen Wert dieser EdgeMessages und weist diesen einem konfigurierten Node zu.

Tabelle 47. Matching Rules

Parameter	Beschreibung
Property	Key des Property (z.B. Datenfeld) im Body der EdgeMessage
Scope	Regex-Filterung der die Information im Header-Feld "Scope" der EdgeMessage abgleicht
Source	Regex-Filterung der die Information im Header-Feld "Source" der EdgeMessage abgleicht

5.3.4. Nodes löschen

Angelegte Nodes können über die Schaltfläche 🗑 im Node Tree oder via der Schaltfläche ✕ in der Auflistung im Bereich Matching Rules glöscht werden.

5.3.5. Konfiguration zurücksetzen

Über die Schaltfläche ⟲ in der horizontalen Menüleiste lässt sich die komplette Konfiguration des Moduls auf Werkseinstellungen zurücksetzen. Nach einem Klick auf die Schaltfläche folgt ein Bestätigungsdialog mit "Yes" und "No" zur Rückfrage ob der Vorgang wirklich ausgeführt werden soll.

> Damit wird auch die gesamte Historie auf dem RoboGate Edge gelöscht!

5.3.6. Module State

Im Module State Fenster wird der aktuelle Verbindungsstatus des OPCUA Server Moduls gezeigt.

Abbildung 98. OPC UA Server - Module State

5.4. Splunk

Das Splunk-Modul ermöglicht das Übertragen von Daten an einen Splunk-Server.

Das Splunk-Modul nimmt Nachrichten eines Message Brokers entgegen und schreibt diese in einen Zwischenspeicher. Die entstehende Queue wird zyklisch geleert und der Inhalt per HTTP-Post-Event an den Splunk Event Collector gesendet.[5]

Als Splunk Timestamp wird der Zeitstempel des Datennachricht-Payloads verwendet. Oder der Zeitstempel des Datennachricht-Payloads wird durch die Splunk Indizierungszeit ersetzt. In das Splunk-Feld „Source“ werden die Inhalte der Source Metadaten geschrieben. Die Übertragung der Scope Metadaten erfolgt in das zusätzlich angelegte Splunk Feld „Scope“.

5.4.1. Konfigurationsparameter

Für die Nutzung des Splunk Moduls sind die im Folgenden beschriebenen Paramerter erforderlich.

Connection

Tabelle 48. Konfiguration Connection Splunk Modul

Parameter	Beschreibung
URL	URL des HTTP Event Collectors (Standardport: 8088)
Proxy	Angabe eines HTTP Proxy für die Verbindung zwischen RoboGate Edge und Splunk; optional.
Access Token	Wird im verwendeten Splunk-System erstellt. Dieser ist für die Eingabe in der HTTP Event Collector Instanz erforderlich (Erstellung s. u.)
Host	Als Ereignishost kann der Hostname, die IP-Adresse oder der vollqualifizierte Domänenname des Netzwerkhosts, von dem das Ereignis stammt, genutzt werden. Mit dem Host können Daten im Splunk gezielt gesucht und ausgewertet werden, die aus einer bestimmten Quelle stammen.

Erstellen des Access Tokens in Splunk:

1. Anmeldung im Splunk und Öffnen der „Einstellungen“
2. Auswahl „Dateneingaben“
3. Auswahl „HTTP-Ereignissammlung“
4. Button „NeuToken“ in der oberen rechten Ecke
    a. Bearbeitung des Reiters „Quelle auswählen“ und „Weiter“
    b. Bearbeitung des Reiters Eingabeeinstellungen und „Weiter“
    c. Bearbeitung des Reiters Prüfen und „Weiter“
    d. Bearbeitung des Reiters Senden

Der Token steht nach der Erstellung zur Eingabe in der RoboGate Edge UI zur Verfügung.

Settings

Tabelle 49. Konfiguration Splunk-Modul - Settings

Parameter	Beschreibung
Send Timestamp	Wenn der Switch ausgeschalten ist (sendTimestamp = false) wird die Zeitstempel-Information aus den Nachrichten vor dem Senden nach Splunk verworfen. Splunk setzt eine Event-Zeit, die dem Zeitpunkt des Einlaufens der Nachricht am Splunk Server entspricht.

Parameter	Beschreibung
Source Type	Source Type ist eines der Standardfelder, die Splunk allen eingehenden Daten zuweist. Es gibt die Datenstruktur eines Ereignisses an (_json).
Enable persistent Queue	Queue-Persistierung wird eingeschaltet. Bei Verbindungsproblemen werden nicht übertragene Nachrichten zyklisch persistiert und gehen somit bei einem zwischenzeitlichen Neustart der Anwendung/des Geräts nicht verloren.

Nachrichten / Topics

Für die Nachrichtenübertragung ist einer der beiden folgenden Modi zu wählen.

Tabelle 50. Konfiguration Splunk-Modul - Messages

Parameter	Beschreibung
Telemetry Messages	Nachrichten werden ohne Umformung an den Splunk Server übermittelt, nach dem Schema: `{<VariablenNameA>: <VaraiblenWertA>, <VariablenNameB>: <VariablenWertB>, ⋯ }`

Event Messages	Einlaufende Nachrichten werden in einzelne Event Nachrichten umgewandelt. Folgendes Schema gilt:
	`{ "name": <VariablenNameA>, "value": <VaraiblenWertA> } UND`
	`{ "name": <VariablenNameB>, "value": <VaraiblenWertB> }`
	(bezogen auf das obige Bsp. Telemetry Messages)

5.4.2. Konfiguration in der UI

Das Splunk-Modul kann über die Auswahl des Moduls auf der Startseite oder über die linke Menüleiste der RoboGate Edge UI aufgerufen und konfiguriert werden.

Connection konfigurieren

Entsprechend Ihrer Anforderungen ist die Konfiguration des Moduls vorzunehmen. Die Parameter sind in oben näher beschrieben. Nach der Eingabe der URL, ggf. Proxy, des Access Token und Host, können Topics hinzugefügt werden.

Nach Fertigstellung der Konfiguration wird diese über die Speichern-Schaltfläche in der rechten oberen Ecke übernommen. Durch das Speichern der Konfiguration steht diese im RoboGate Edge zur Verfügung (Abbildung 99).

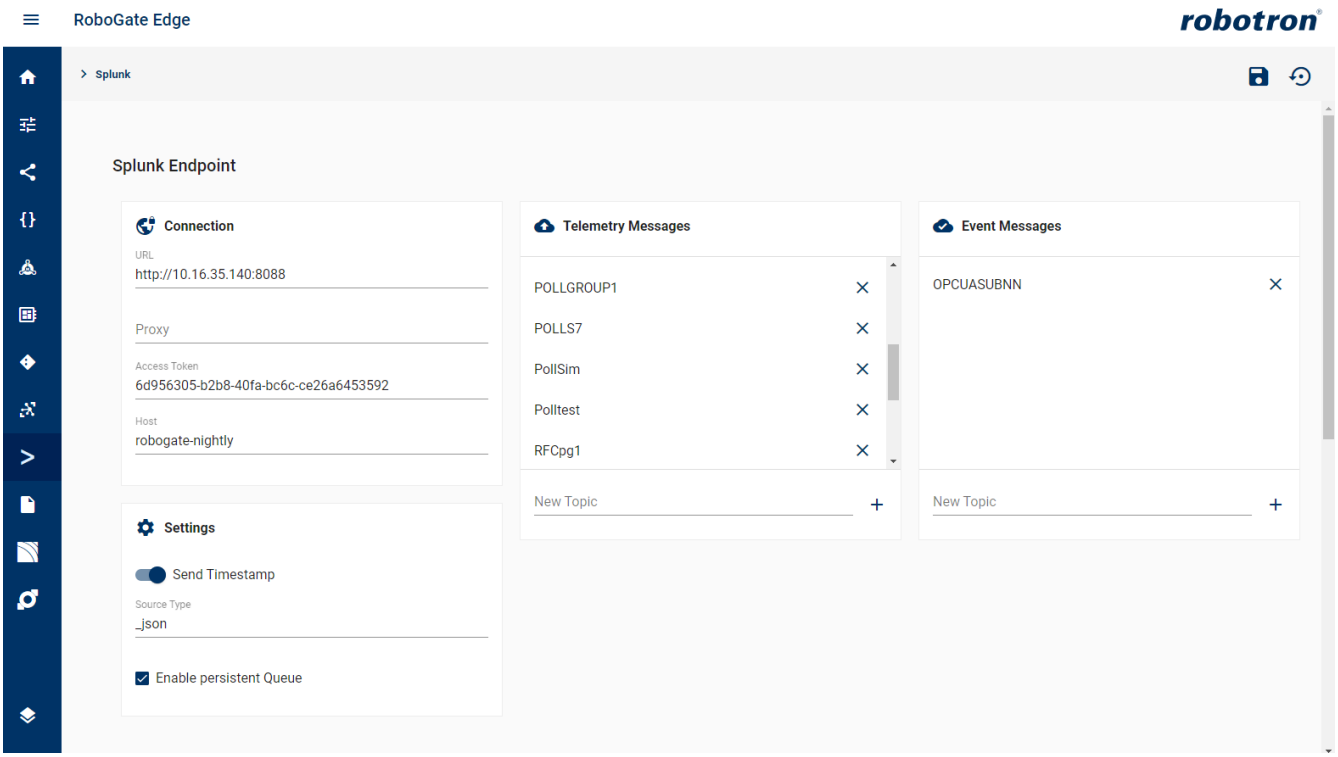


Abbildung 99. Konfiguration Splunk

Nachrichten konfigurieren

Unter Telemetry Messages und Event Messages können Output Topics der anderen RoboGate Edge Module per Klick auf „+“ hinzugefügt werden. Diese enthalten die im Splunk auszuwertenden Daten. Durch das Speichern der gesamten Konfiguration des Splunk-Moduls werden die Daten der ausgewählten Topics an Splunk gesendet und stehen zur Auswertung in Splunk zur Verfügung.

Abbildung 100 zeigt die Auswertung für Telemetry Messages am Beispiel Topic „ModbusRTUPollGroup1“ und Abbildung 101 die Auswertung für Event Messages am Beispiel Topic „OPCUAPOLL1“.

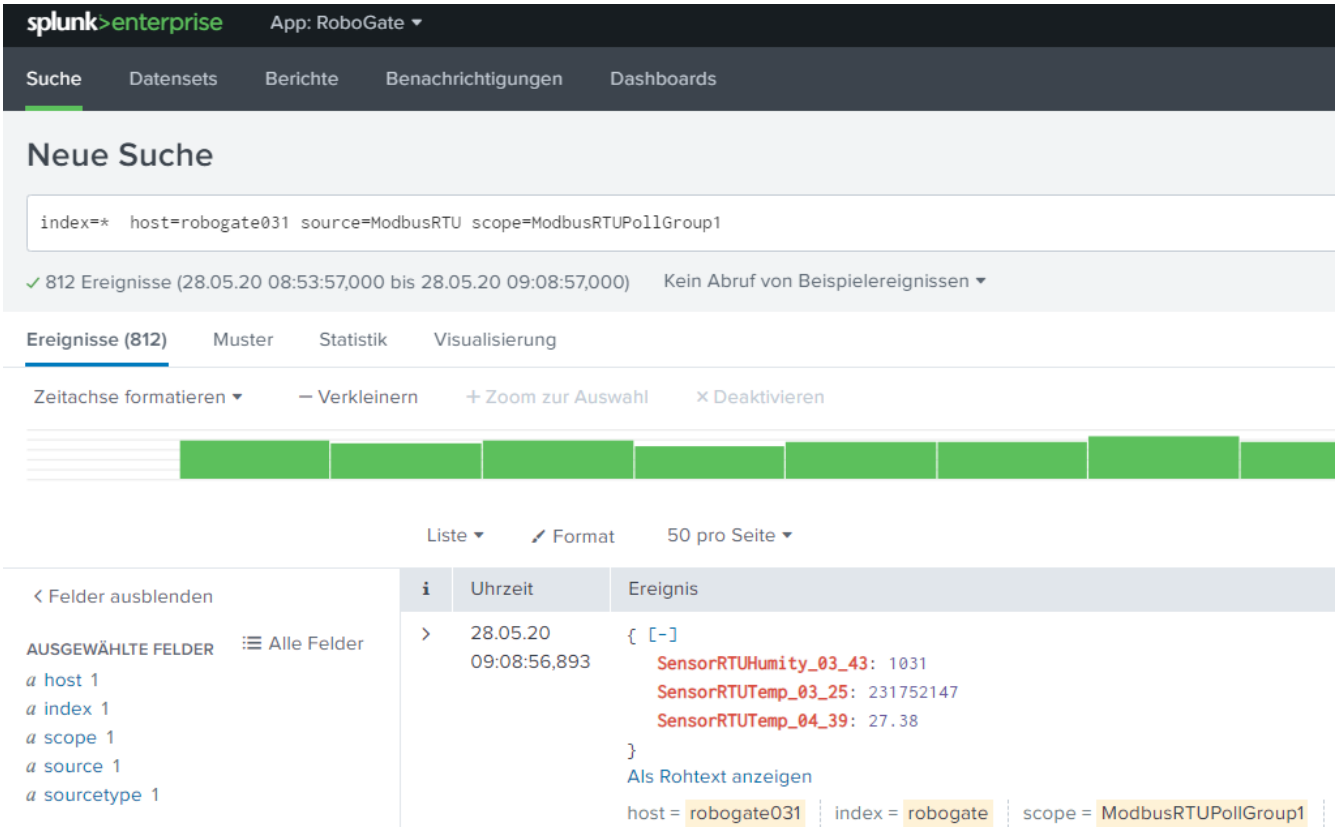


Abbildung 100. Splunk-Auswertung Message-Typ: Telemetry

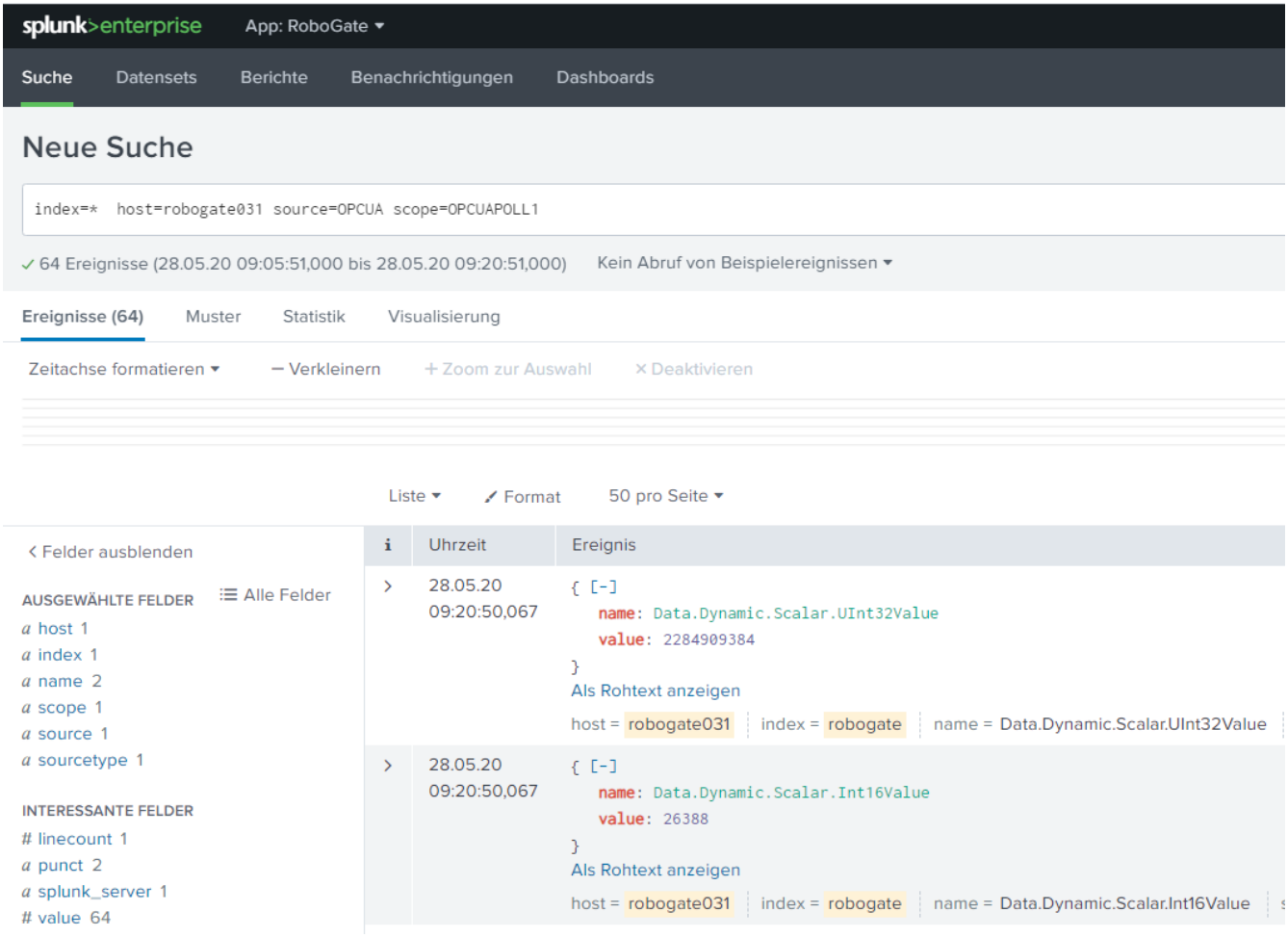


Abbildung 101. Splunk-Auswertung Message-Typ: Event

Im Bereich "Module State" (Abbildung 102) lässt sich erkennen, ob das Modul eine Verbindung zu

Splunk herstellen konnte.

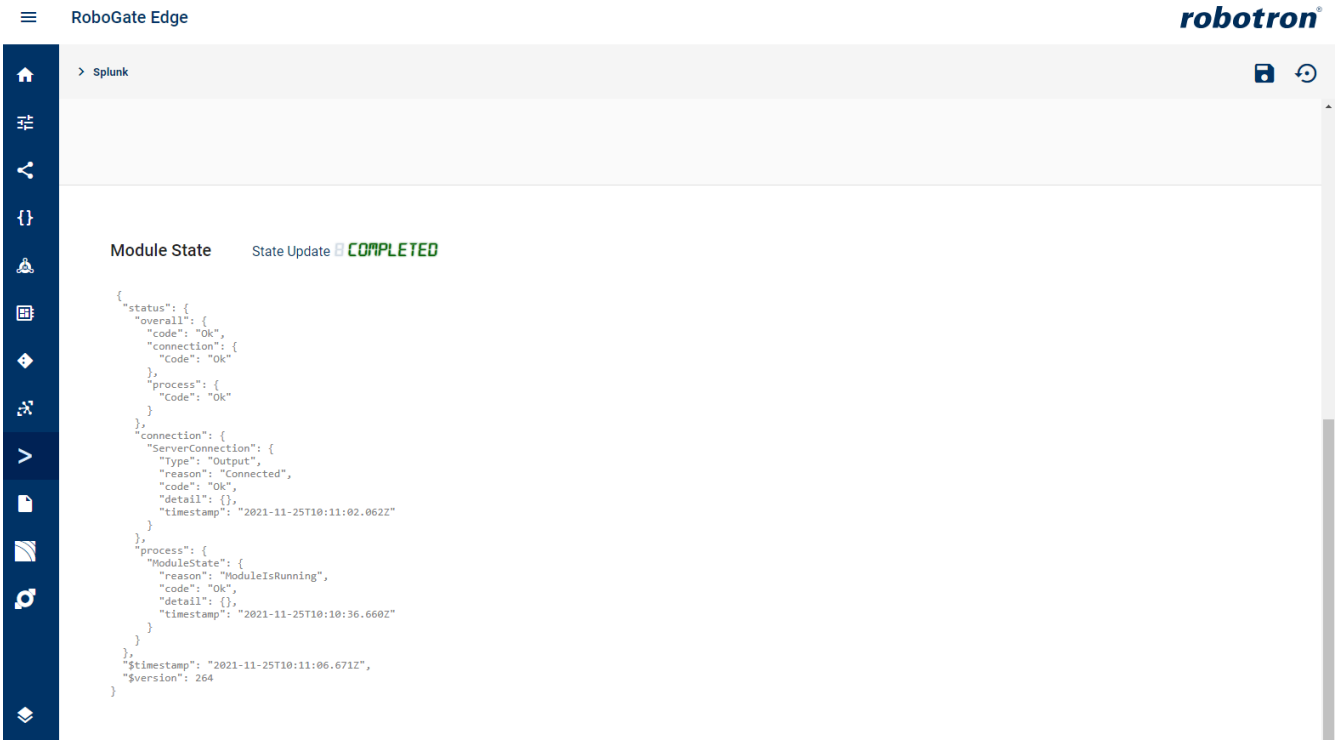


Abbildung 102. Splunk-Modul - Status

Topic löschen

Innerhalb der Konfiguration des Splunk-Moduls können Topics gelöscht werden. Dazu ist das zu löschende Topic per Mausklick auf „x“ zu entfernen. Durch das Speichern der Konfigurationsänderung steht diese im RoboGate Edge zur Verfügung (Abbildung 103).

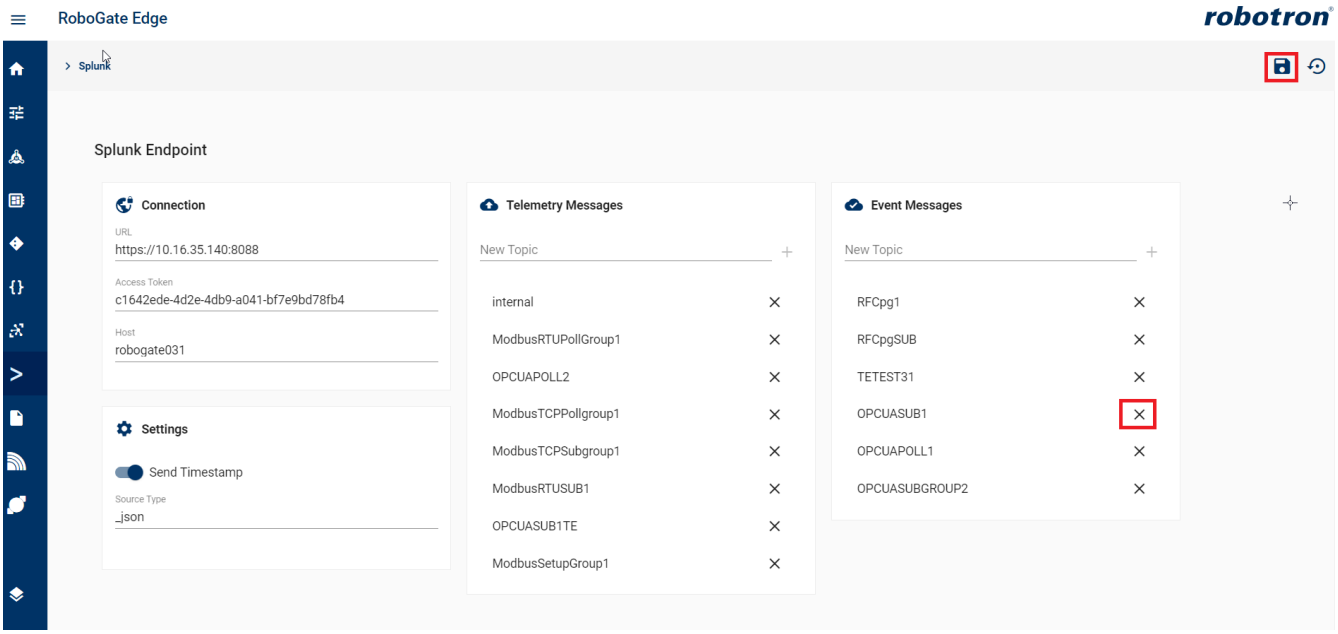


Abbildung 103. Löschen eines Topics

Konfiguration zurücksetzen

Über die Schaltfläche in der horizontalen Menüleiste lässt sich die komplette Konfiguration des Moduls auf Werkseinstellungen zurücksetzen. Nach einem Klick auf die Schaltfläche folgt ein

Bestätigungsdialog mit "Yes" und "No" zur Rückfrage ob der Vorgang wirklich ausgeführt werden soll.

> Damit wird auch die gesamte Historie auf dem RoboGate Edge gelöscht!

6. Management-Module

6.1. Control Panel

Das Control Panel gibt einen Überblick über den Status und das Netzwerk des RoboGate Edge.

Hier kann ein Neustart des RoboGate Edge mit einem Klick auf "Reboot System" (Abbildung 104) durchgeführt werden. Es öffnet sich ein Dialogfenster, in dem eine von drei Neustartoptionen gewählt und über die Schaltfläche "Ok" bestätigt werden kann (Abbildung 105):

- Normal: Das Betriebssystem wird neu gestartet. Ausführbar auf RoboGate Devices powered by Turck und RoboGate Devices powered by Moxa.
- Power: Das Betriebssystem und die Hardware werden neu gestartet. Gleicht einem kurzzeitigen Trennen der Stromversorgung vor dem Start. Ausführbar auf RoboGate Devices powered by Turck.
- Force: Neustart des Betriebssystems ohne geregeltes Herunterfahren. Ausführbar auf RoboGate Devices powered by Turck.

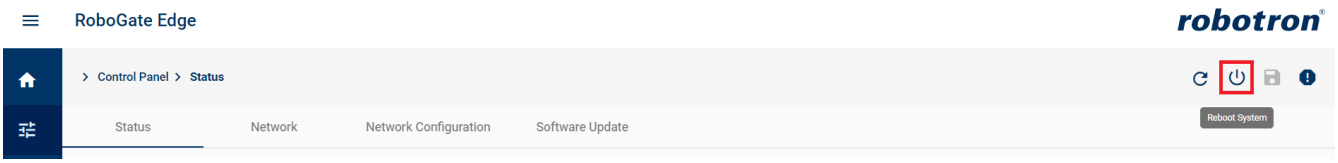


Abbildung 104. Reboot System im Control Panel

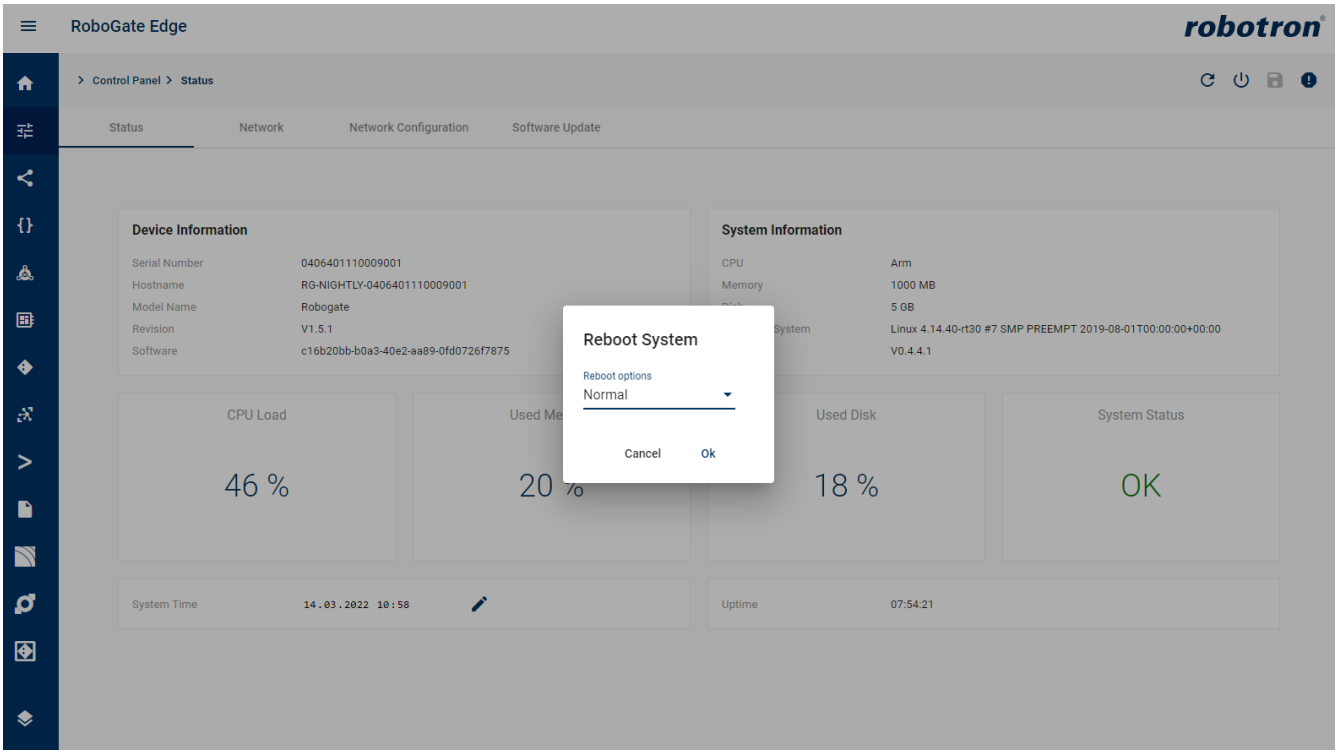


Abbildung 105. Reboot System-Optionen im Control Panel

Weiterhin können die Systeminformationen (z.B. Modulversionen, Error Logs, Konfiguration) über "Download Support packages" (Abbildung 106) heruntergeladen werden, um diese Robotron bzw. Ihrer Kontaktperson bei Robotron zu Support-Zwecken bereitzustellen. Es folgt die Meldung

"Creating Information Package... Please wait...". Abschließend wird eine .zip-Datei mit den Systemdaten heruntergeladen.

> Dieser Vorgang kann bis zu 5 min dauern!

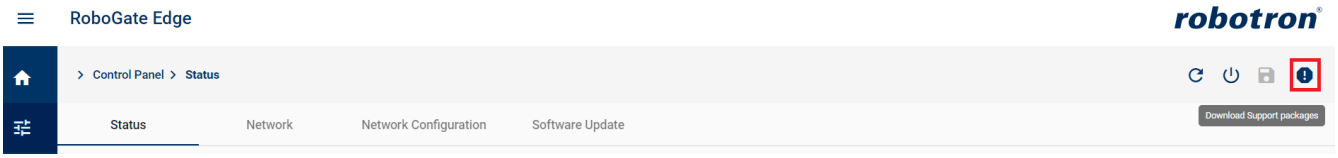


Abbildung 106. Quick Support Button im Control Panel

6.1.1. Status

Über den Reiter „Status“ werden Geräteinformationen (Device Information) und Systeminformationen (System Information) angezeigt. Dazu gehören u.a. die CPU-Last als Average Load der letzten Minute, genutzter RAM, belegter Speicher und eine Gesamteinschätzung darüber, in welchem Zustand sich das System befindet. In Abbildung 107 ist ein RoboGate Edge mit dem System Status „OK“ abgebildet. Die Informationen werden minütlich aktualisiert. Des Weiteren werden die Systemzeit und die Betriebszeit angezeigt. Genaue Informationen über die Device und System Informationen befinden sich in den folgenden Tabellen.

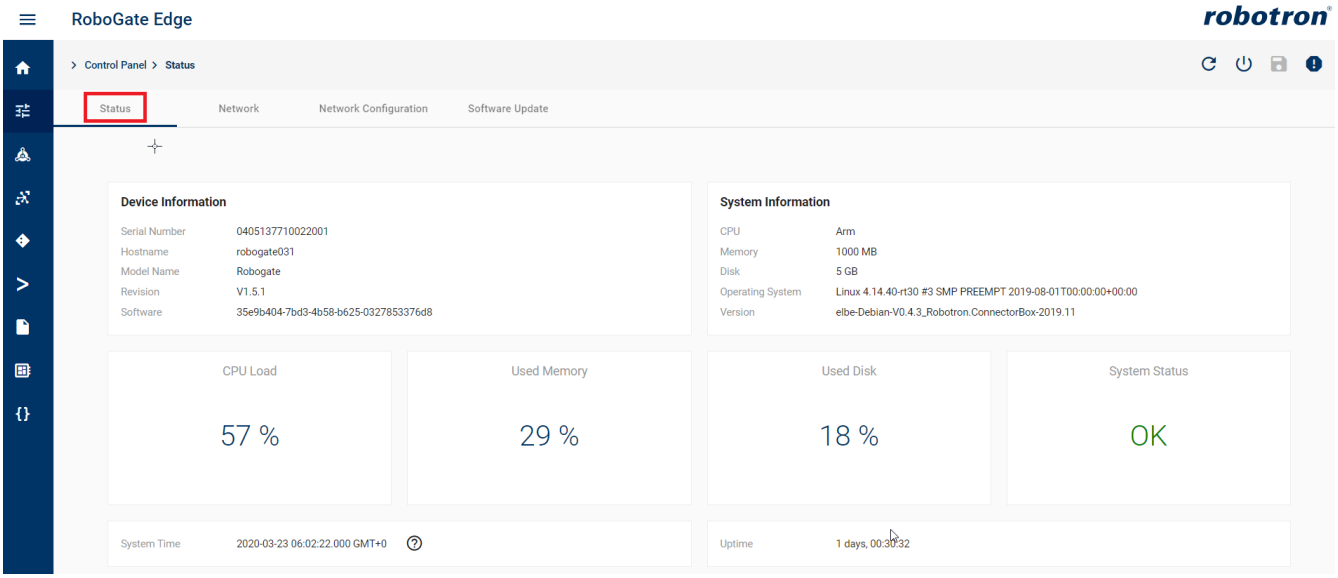


Abbildung 107. Control Panel - Status

Tabelle 51. Device Information

Feld	Beschreibung
Serial Number	Seriennummer des Gerätes
Hostname	Host Name vom Gerät Default: RG-<SerialNumber>
Model Name	Art des RoboGate Edge
Revision	Nummer der Hardware-Revision
Software	ID der Software Version ID Bitte bei Fehlern angeben.

Tabelle 52. System Information

Feld	Beschreibung
CPU	Typ und Version der eingebauten CPU
Memory	Größe des installierten Arbeitsspeichers
Disk	Größe des Speicherplatzes
Operating System	Art des Betriebssystems
Version	Version des Betriebssystems

6.1.2. Network

Im Reiter „Network“ (Abbildung 108) werden die Netzwerkinformationen des RoboGate Edge angezeigt. Auf dieser Seite werden alle Schnittstellen mit den im RoboGate Edge hinterlegten Konfigurationen angezeigt. Die Aktualisierung erfolgt alle 5 Minuten.

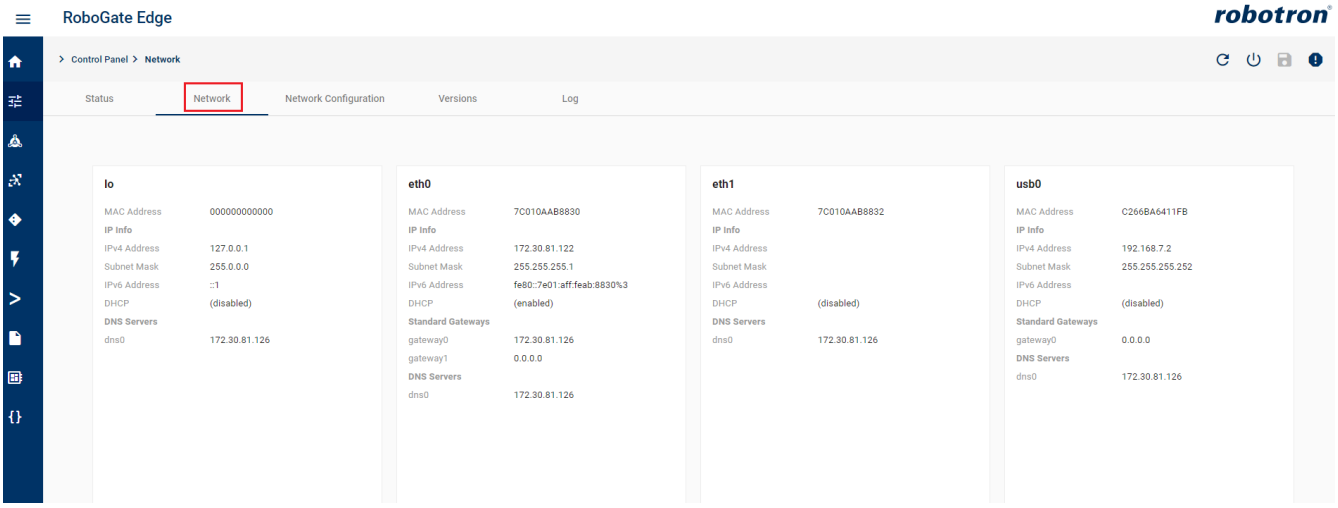


Abbildung 108. Control Panel - Network

6.1.3. Network Configuration

Der Systemadministrator kann das Netzwerk über den Reiter „Netzwerkkonfiguration“ (Abbildung 109) einrichten.

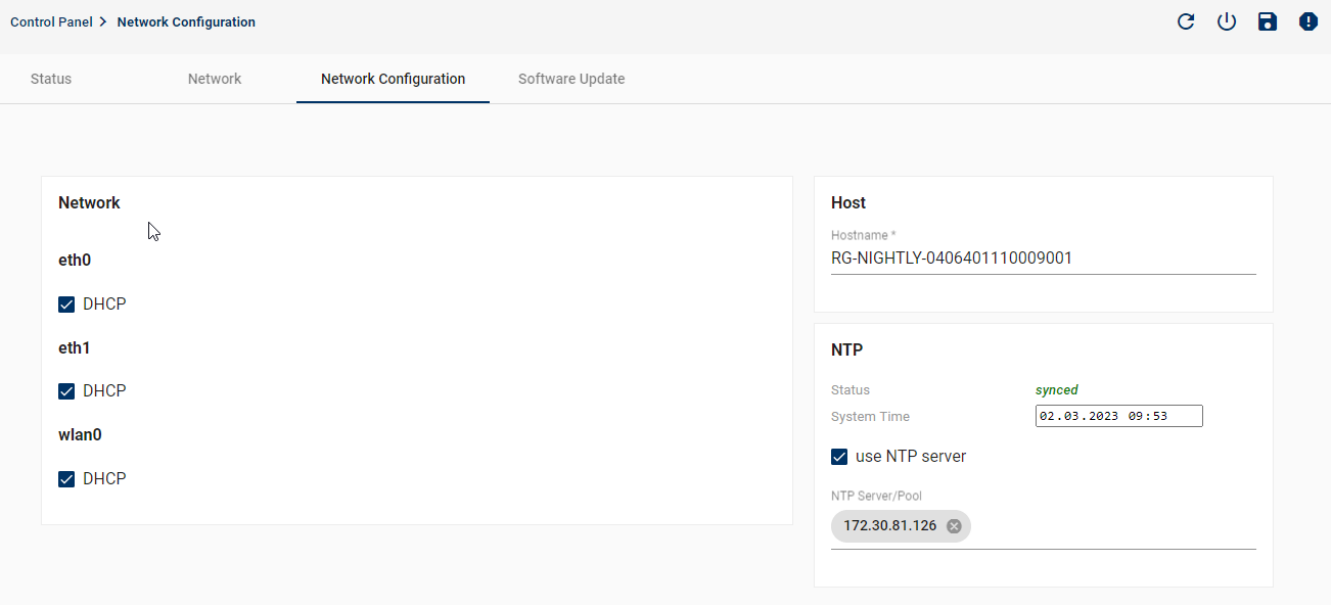


Abbildung 109. Control Panel - Network Configuration

Network: Es können Konfigurationen für die beiden Ethernet-Schnittstellen „eth0“, „eth1“ und wlan0 vorgenommen werden. Die Einstellungen beziehen sich auf:

- IP: Dynamic Host Configuration Protocol (DHCP) oder statische Konfiguration.
  Eine statische Konfiguration erfordert die Eingabe der (eigenen) IP-Adresse und Subnetz-Maske, im Format „xxx.xxx.xxx.xxx/yy“
- DNS-Server
- Standard Gateway

In der Regel werden bei Verwendung von DHCP alle Parameter über den DHCP Server aufgelöst.

Host: Der Hostname entspricht dem DNS-Name des RoboGate Edge.

NTP: Der NTP Server/Pool dient der Zeitsynchronisierung per Network Time Protocol (NTP). Die Nutzung erfordert die Eingabe der IP-Adresse (v4) oder des DNS-Namens. Weitere Informationen finden Sie auf folgender Seite: http://support.ntp.org/servers. Wird das "USE NTP Feld" leer gelassen, ist die Zeitsynchronisierung ausgeschalten und eine Zeit manuell im Reiter „Status“ bei "System Time" zu hinterlegen.

6.1.4. Software Update

Im Reiter "Software Update" lässt sich die Software des RoboGate Edge auf RoboGate Devices aktualisieren. Abbildung 110 zeigt das Eingabefeld für das Software Update.

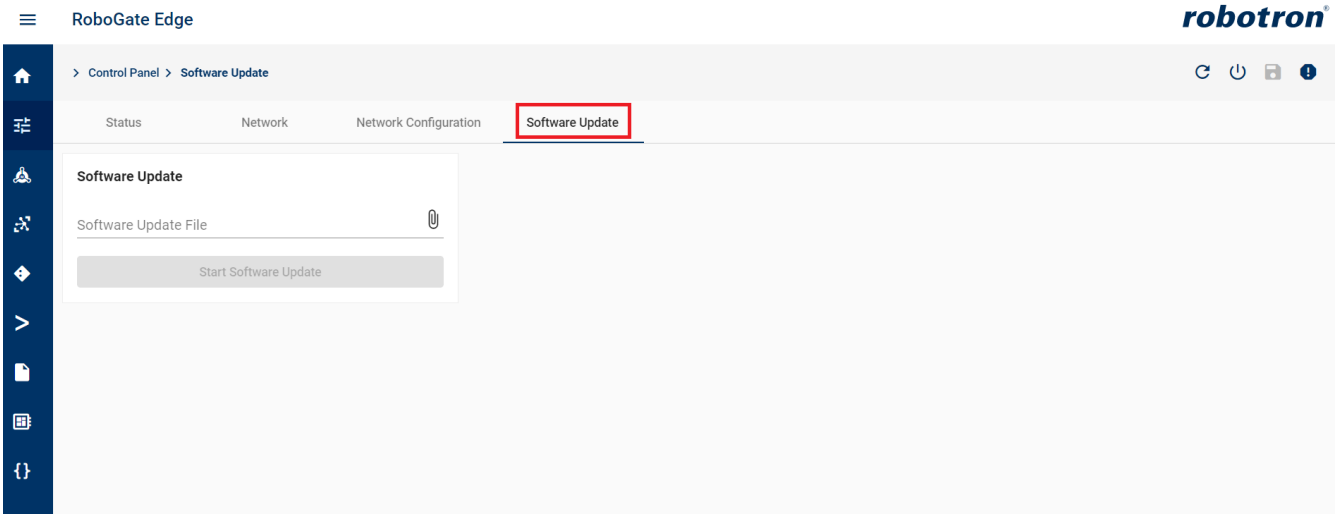


Abbildung 110. Control Panel - Software Update

Um die Software zu aktualisieren, muss über das Eingabefeld eine von Robotron bereitgestellte *.rsb-Datei (RoboGate Software Bundle) ausgewählt werden. Anschließend kann durch den Klick auf *Start Software Update* der Aktualisierungsprozess gestartet werden. Dies kann einige Minuten in Anspruch nehmen.

> **Bitte aktualisieren Sie die Seite während des Update-Prozesses nicht.**

6.2. ControlCenter

Das Modul ControlCenter ist ein Client, der das Management von RoboGate Edges über das **RoboGate ControlCenter** ermöglicht.

Sobald das Modul konfiguriert ist, baut das RoboGate Edge zum Zwecke des Fleet Management zyklisch eine Verbindung zum Management Server auf und synchronisiert sich mit dem zentralen Management. Dies erlaubt neben der Statusüberwachung (Heartbeat) auch die Synchronisation der Konfiguration(en) und zentralisierten Software-Rollouts.

6.2.1. Konfigurationsparameter

In Tabelle 53 werden die Parameter zur Konfiguration der Verbindung (Connection Settings) des ControlCenter erläutert.

Tabelle 53. Konfiguration des ControlCenter-Moduls

Parameter	Beschreibung
Hostname	ConnectionString zum ControlCenter Server. Er besteht aus einem Hostname/URL. Es wird kein Schema angegeben, sondern automatisch der Default benutzt (derzeit https). Optional können auch ein Port und ein Path angegeben werden. Beispiele: • control-center.robotron.de • http://control-center.robotron.de:5000/controlcenter

Parameter	Beschreibung
Proxy	Wenn leer, wird eine Proxy Nutzung für diese Verbindungen deaktiviert. Wenn gesetzt, wird dieser Proxy für alle Verbindungen des Moduls benutzt (ControlCenter Server). Format entspricht dem Schema: Beispiele: proxy http://proxy.intranet.local http://proxy.intranet.local:80
SSL Check	Default: Ja : Das Serverzertifikat wird geprüft. Deaktivieren ist zum Beispiel für den Anwendungsfall SelfSigned Zertifikate im ControlCenter WebServer nutzbar.
Apply Identities	Wenn aktiviert, werden Identities, die auf dem Server konfiguriert wurden und in Heartbeat-Antwort zum Gerät geschickt werden, in die entsprechende Modulkonfiguration (z.B. Azure) geschrieben.
Heartbeat Interval (ms)	Default: 10 sec, Legt das Intervall fest, mit dem sich das RoboGate Edge beim ControlCenter Server meldet. Sollte je nach Anforderungen so hoch wie möglich, so niedrig wie nötig festgelegt werden. Empfehlung: 1-10 Minuten.

6.2.2. Konfiguration in der UI

Die oben genannten Parameter sind im Modul zu definieren (Abbildung 111) und werden nach Speichern wirksam. Im Bereich "Module State" lässt sich erkennen, ob das Modul eine Verbindung zum ControlCenter herstellen konnte.

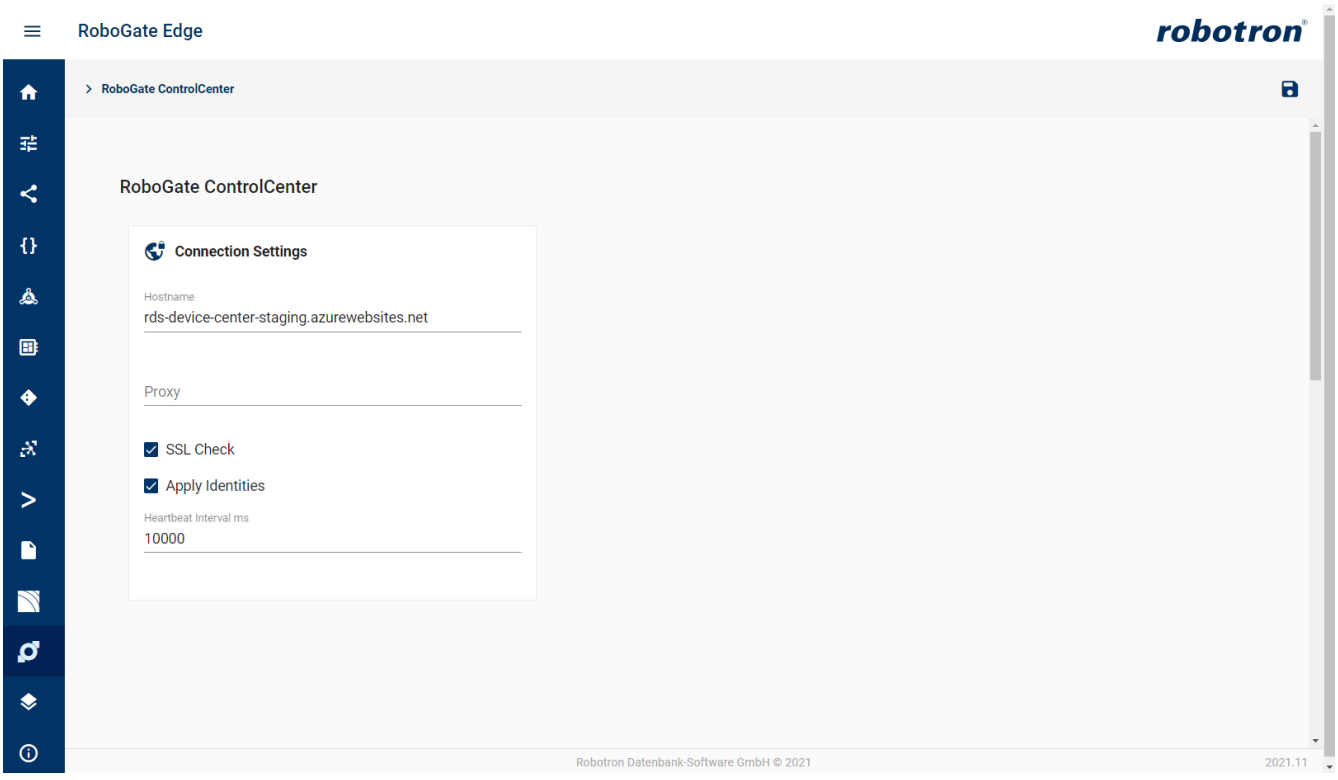


Abbildung 111. RoboGate ControlCenter-Modul

6.3. System Management

6.3.1. Log

Im Reiter „Log“ werden die letzten 500 Systemlogs des RoboGate Edge aufgelistet (Abbildung 112). Durch einen Klick auf eine Logzeile werden weitere Details angezeigt.

Die folgende Tabelle zeigt, welche Informationen zu Logs erfasst werden.

Tabelle 54. System Management - Log-Informationen

Parameter	**Beschreibung**
Time Stamp	Zeitstempel des Logs
Level	Einstufung des logs in „Warning“ oder „Error“
Module	Anzeige der Module, durch die Logs entstanden sind
Message	Kurzbeschreibung des Logs

Es ist auch möglich die Fehlermeldungen oder Warnungen nach Loglevel, Modulherkunft oder Regexausdrücken zu filtern. Genaue Optionen finden sich in der Tabelle:

Tabelle 55. System Management - Filterungsoptionen

Parameter	**Beschreibung**
Number of total logs	Maximale Anzahl an sichtbaren Fehlermeldungen.
Log Level	Welcher Grad von Nachrichten gelistet werden sollen. Zur Option stehen "All(None)", "Error" oder "Warning".
Modules	Auswahl aus welchem Modul die Meldungen stammen. Hier stehen z.B. OPCUA, Streamprocessor zur Verfügung. Abhängig davon aus welchem Modul Meldungen stammen.
Regex	Für eine präzise Filterung mit Hilfe von regulären Ausdrücken (RegEx).

Für die Anwendung des Filters muss hier auf "Apply Filters" geklickt werden. Um die Meldungen weiter zu verarbeiten können diese als JSON oder CSV heruntergeladen werden. Diese Option findet sich unten rechts im Filtermenu.

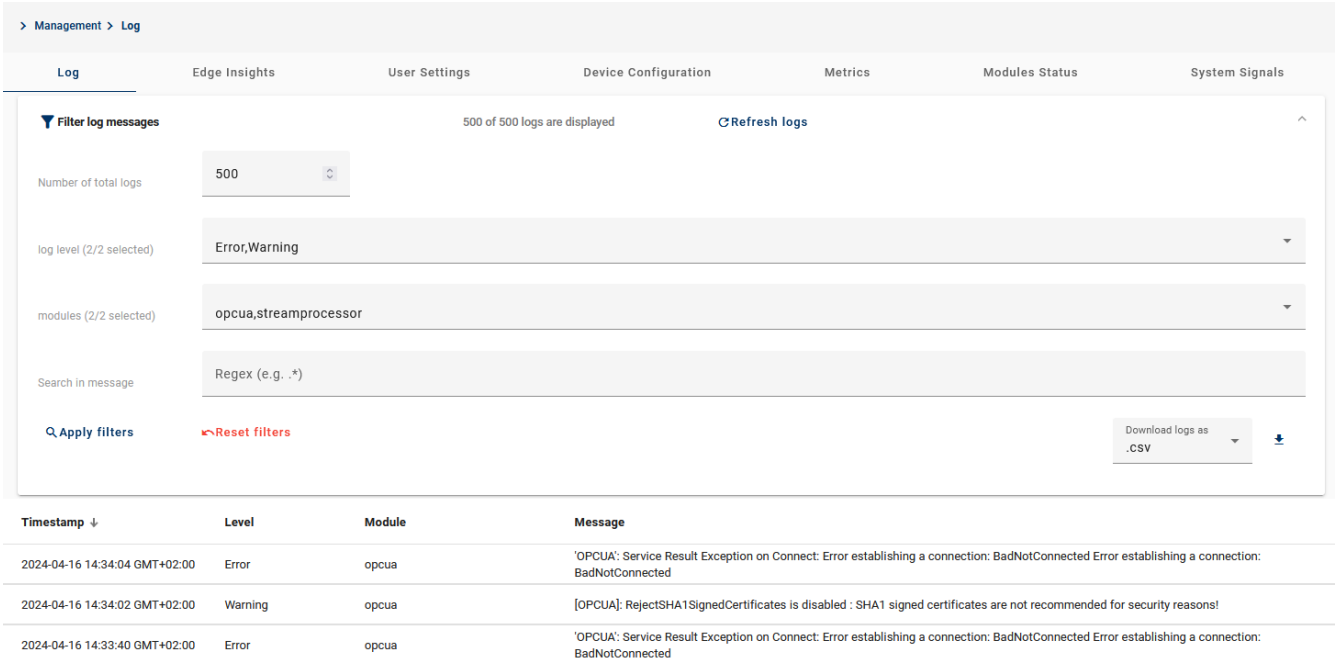


Abbildung 112. System Management - Log

6.3.2. Edge Insights

Edge Insights dient dem Abfragen der internen Topics des Robogates. Hier kann z.B. direkt die Ergebnisse der Quellen ausgeben werden, z.B. OPC UA. Dies kann sehr hilfreich werden, wenn bei komplizierten Verarbeitungen im StreamProcessor Daten nicht korrekt verarbeitet werden. Es bietet sozusagen ein Debugging an.

Zusätzlich zu den individuellen, vom Anwender benutzte Topics, existieren zusätzlich auch unter Settings die Robogate internen Topics die den Status verschiedener Module aufzeigt. Um diese anzuzeigen muss ein Häkchen bei **show internal topics** gesetzt werden.

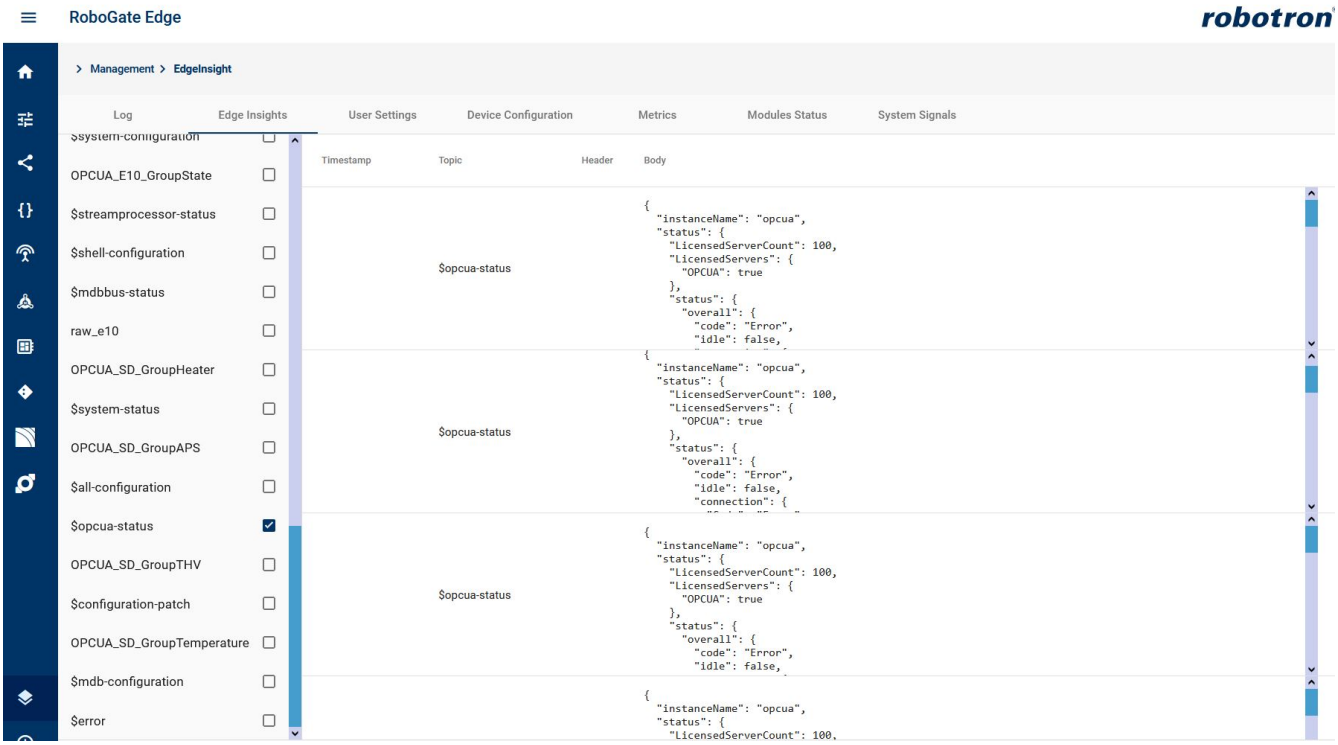


Abbildung 113. Edge Insights

Oberfläche

Die Ausgabe der Topicinformationen erfolgt in einer Tabelle. Siehe hierzu Abbildung 113. Zudem ist es möglich die Anzahl der Einträge pro Seite zu ändern. Hierfür klickt man auf **Item per page** und wählt eine gewünschte Anzahl aus.

Spalten

Feld	Beschreibung
Timestamp	Zeitpunkt wann Nachricht raus geschickt wurde
Topic	Der Name des Topics.
Header	Die Headerinformationen des jeweiligen Topics.
Body	Die Bodyinformationen des Topics.

Settings (Internal Topics)

Alle internen Topics besitzen ein Dollarzeichen vor dem Namen. In der Abbildung 113 wurde beispielsweise das Topic $opcua-status aktiviert.

Sonstige Anmerkungen

- Topics bleiben in der Leiste auch sichtbar, wenn Sie nicht mehr exisitieren, zum Beispiel nach einem Löschen. Will man diese aus der Ansicht entfernen muss man das Robogate neu starten.
- Bei Fehlersuche im Streamprocessor kann es eventuell helfen zusätzliche Outputs einzubauen und diese dann in Edge Insights anzeigen zu lassen. So sind rudimentäre Debuggingmöglichkeiten gegeben.
- Falls die Ansicht geleert werden soll, muss auf einen beliebigen Reiter (z.B. Log) geklickt werden und anschließend wieder auf Edge Insights.

6.3.3. User Settings

In den "User Settings" (Abbildung 114) kann ein Username und ein neues Passwort festgelegt werden. Die Änderungen können nur mit einer HTTPS Verbindung vorgenommen werden.

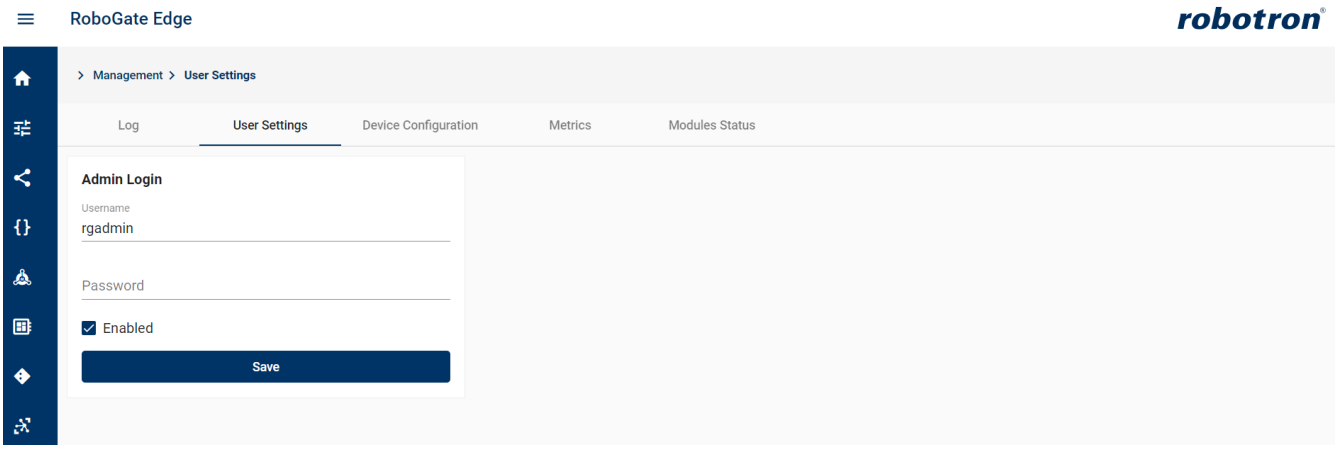


Abbildung 114. System Management - User Settings

6.3.4. Device Configuration

Im Reiter "Device Configuration" wird die gesamte Konfiguration des RoboGate Edge angezeigt. Diese Gesamtkonfiguration lässt sich über die Schaltflächen in der horizontalen Menüleiste als JSON herunterladen oder hochladen und somit importieren (Abbildung 115). Alternativ kann die Konfiguration direkt in der UI bearbeitet werden. Änderungen müssen über die Disketten-Schaltfläche gespeichert werden, um wirksam zu werden.

Weiterhin kann die komplette Konfiguration des RoboGate Edge über die Schaltfläche "Reset configuration" zurückgesetzt werden (Abbildung 116).

> Achtung: Das Löschen der "system"-Konfiguration führt zum Verlust der Netzwerk/Firewall Einstellungen des Gerätes.

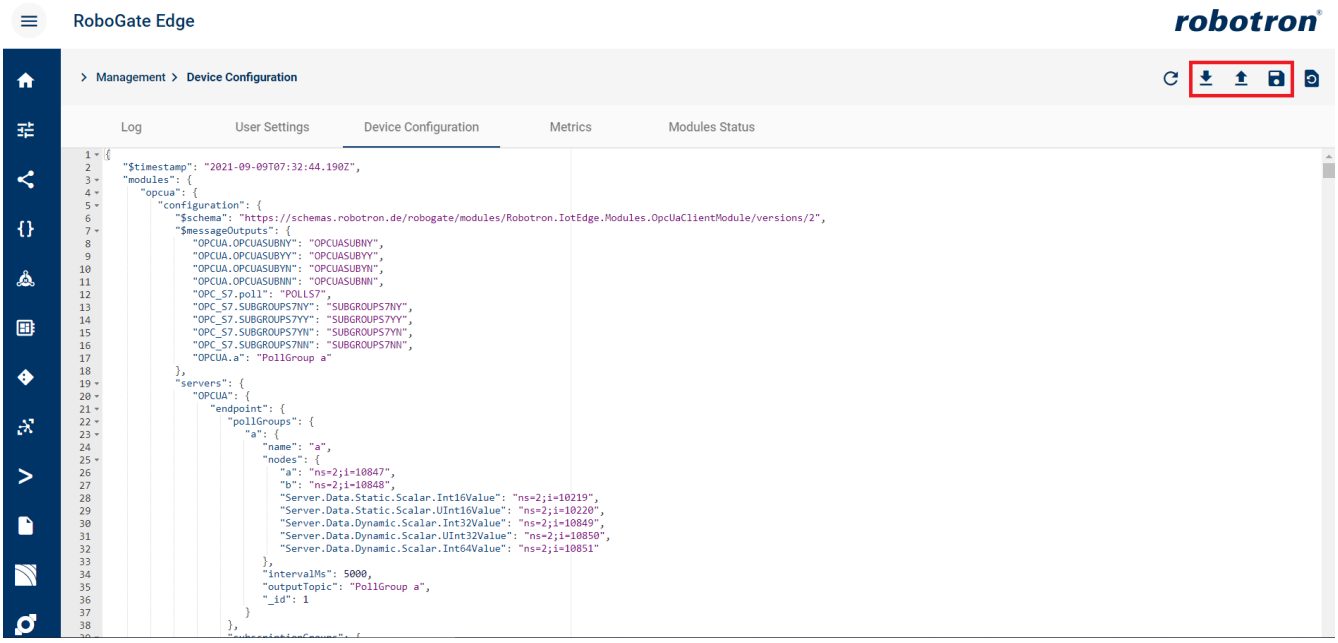


Abbildung 115. System Management - Device Configuration

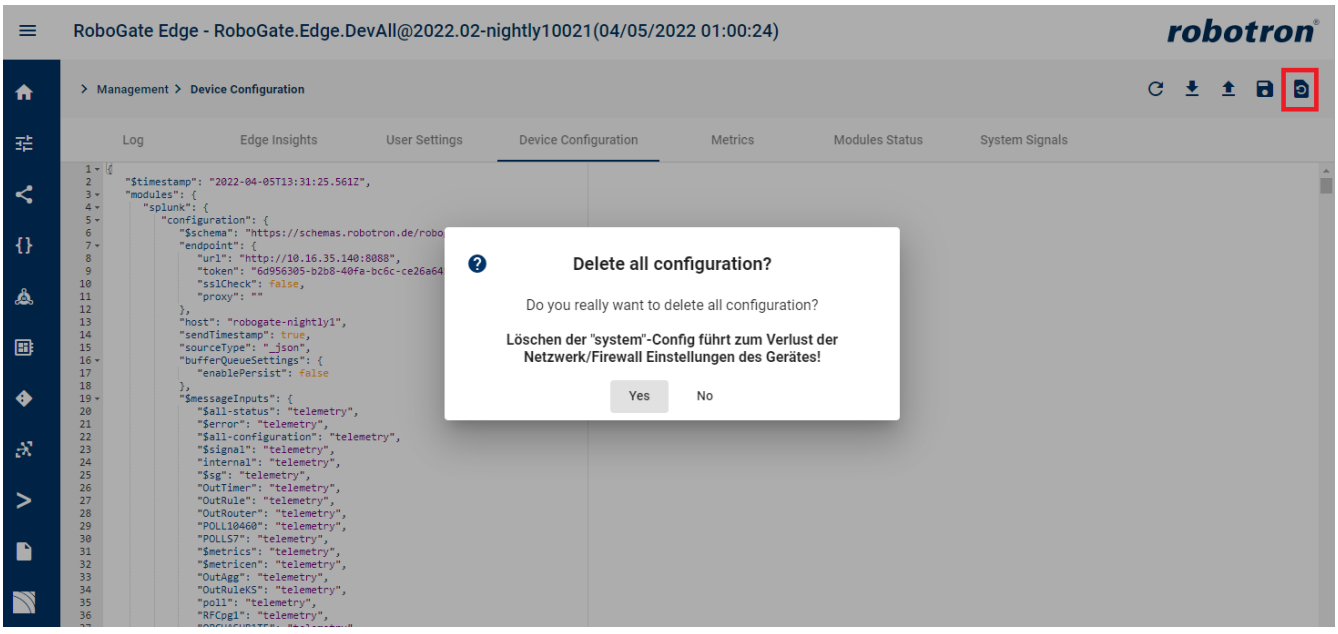
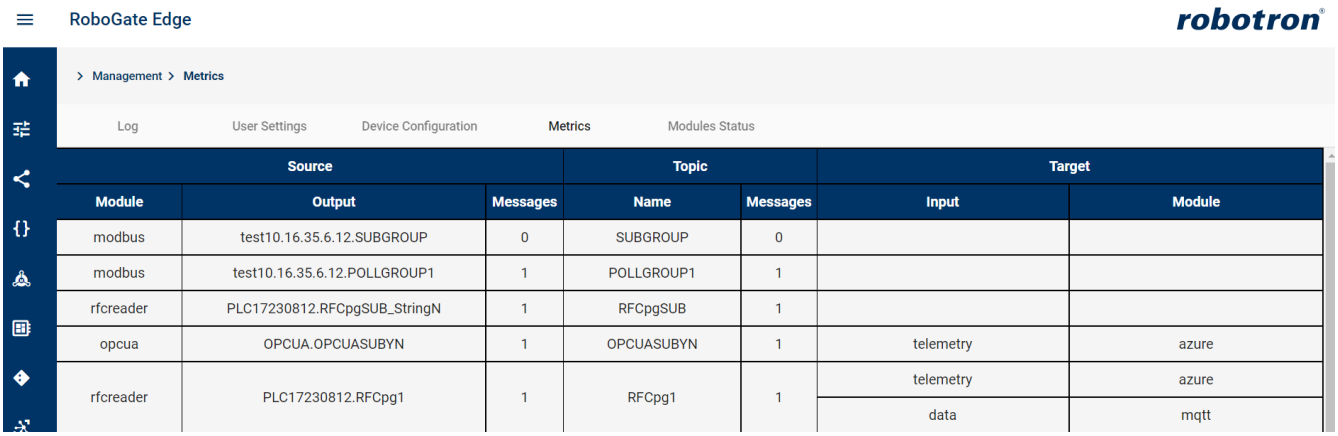


Abbildung 116. System Management - Reset Configuration

6.3.5. Metrics

Der Reiter "Metrics" (Abbildung 117) bietet eine Übersicht über die Nachrichten auf dem Broker pro Topic. Folgende Hinweise dazu sind zu beachten:

- Die Informationen werden nur einmalig geladen.
- Das Topic ist das zentrale Element in der Tabelle. Alle Source- und Targetangaben beziehen sich auf ein Topic.
- Die Anzahl der Nachrichten wird bei den Outputs sowie bei den Topics angezeigt.
- Es wird die korrekte "Verknüpfung" der Konfiguration widergespiegelt.
- Es werden nur Topics angezeigt, über welche mindestens 1 Nachricht gelaufen ist.



Source			Topic		Target	
Module	Output	Messages	Name	Messages	Input	Module
modbus	test10.16.35.6.12.SUBGROUP	0	SUBGROUP	0		
modbus	test10.16.35.6.12.POLLGROUP1	1	POLLGROUP1	1		
rfcreader	PLC17230812.RFCpgSUB_StringN	1	RFCpgSUB	1		
opcua	OPCUA.OPCUASUBYN	1	OPCUASUBYN	1	telemetry	azure
rfcreader	PLC17230812.RFCpg1	1	RFCpg1	1	telemetry	azure
					data	mqtt

Abbildung 117. System Management - Metrics

6.3.6. Modules Status

Der Reiter "Modules Status" (Abbildung 118) liefert einen schnellen Überblick über alle Module, die den Status-Report derzeit unterstützen. Im Status wird pro Modul die Connection und der Process detailliert dargestellt (Abbildung 119). Es gibt folgende Statusformate:

- OK
- Warnung
- Fehler

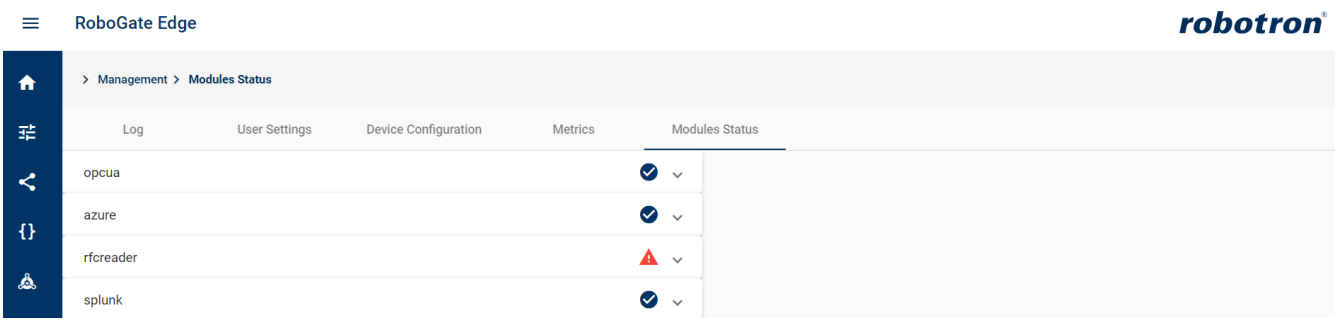


Abbildung 118. System Management - Modules Status

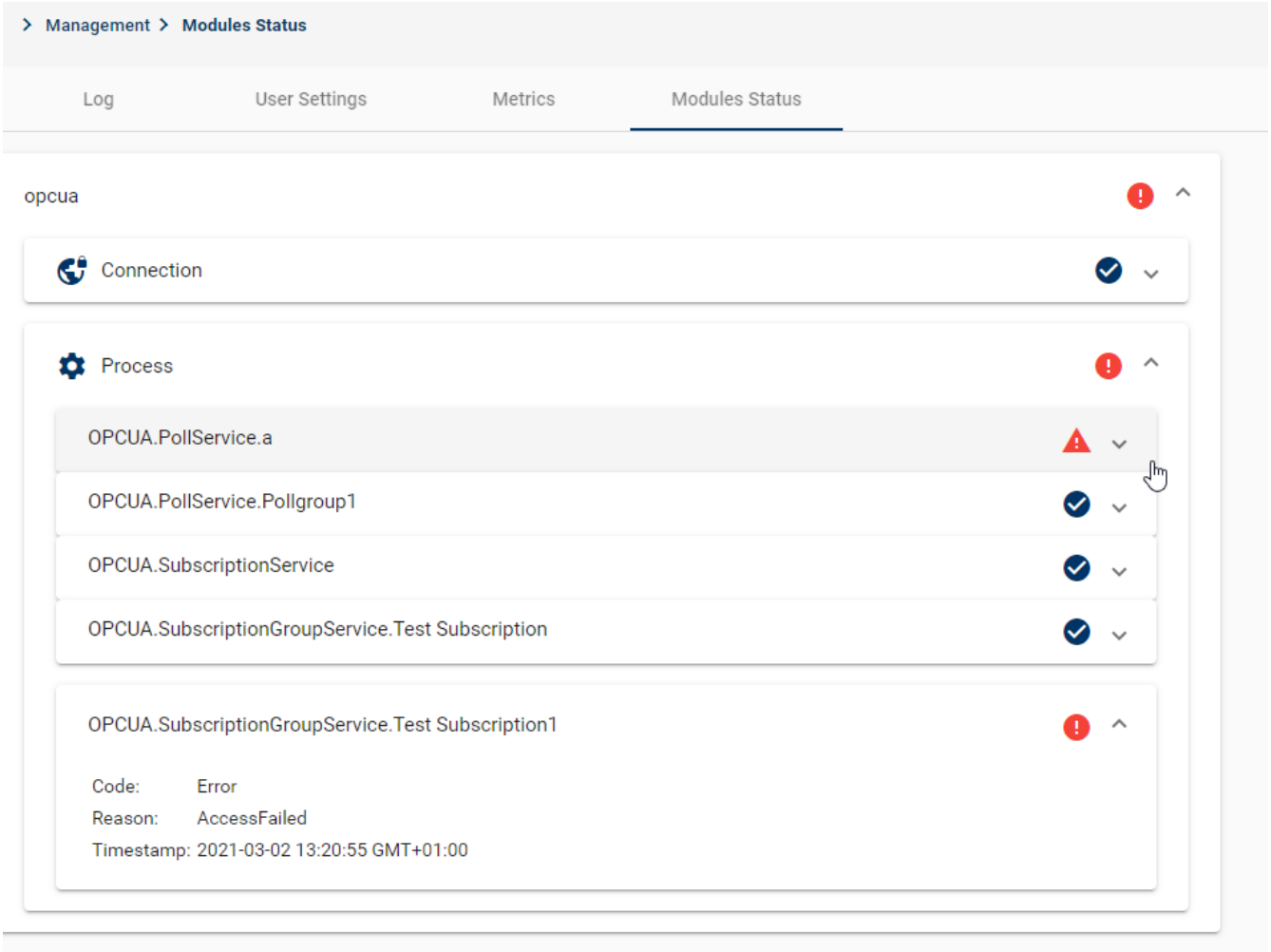


Abbildung 119. System Management Detaillierter Status Überblick

6.3.7. System Signals

Um ein RoboGate Edge in Drittsystemen überwachen zu können, lassen sich Systemstatus und Systemmetriken als Signale, ähnlich wie bei Daterfassungsmodulen, auslesen. Hierfür können Signal Groups mit den in folgender Tabelle beschriebenen Konfigurationsparametern angelegt werden.

Eine Liste der möglichen Metriken findet man unter http://ROBOGATE-ADRESSE/swagger und dort unter GET /metrics. Anschließend auf "Try it out" und "Execute" klicken.

Tabelle 56. Konfigurationsparameter System Signals

Parameter	Beschreibung
Signal Group Name	Bezeichnung der Signal Group.
Output Topic	Name des Output Topic, beginnend mit einem "$", optional. Default wird der Name der Signal Group mit einem vorangestellten "$" verwendet. Alternativ kann dem Topic-Namen, wenn es in einem Ausgabemodul hinzugefügt wird, in diesem Modul ein "$" vorangestellt werden.

Parameter	Beschreibung
Intervall	Intervall in Millisekunden gibt die Taktung an, in der die Werte abgefragt werden.
Signal Name	Name des Signals, eindeutiger string
Signal Type	SelectBox mit Einträgen metric und moduleStatus (source)
Signal Path	Path bzw. sourceKey zum Signal, string

So können beispielsweise alle Statussignale, die in den Modulen unter "Module State" aufgeführt sind, abgefragt werden.

Die folgenden Abbildungen zeigen eine Beispielkonfiguration. In Abbildung 120 ist der abzufragende Status des Azure-Moduls rot markiert, in Abbildung 121 die entsprechende Konfiguration der Abfrage als System Signal, und Abbildung 122 zeigt, wie die konfigurierten Signaldaten in Splunk ankommen.

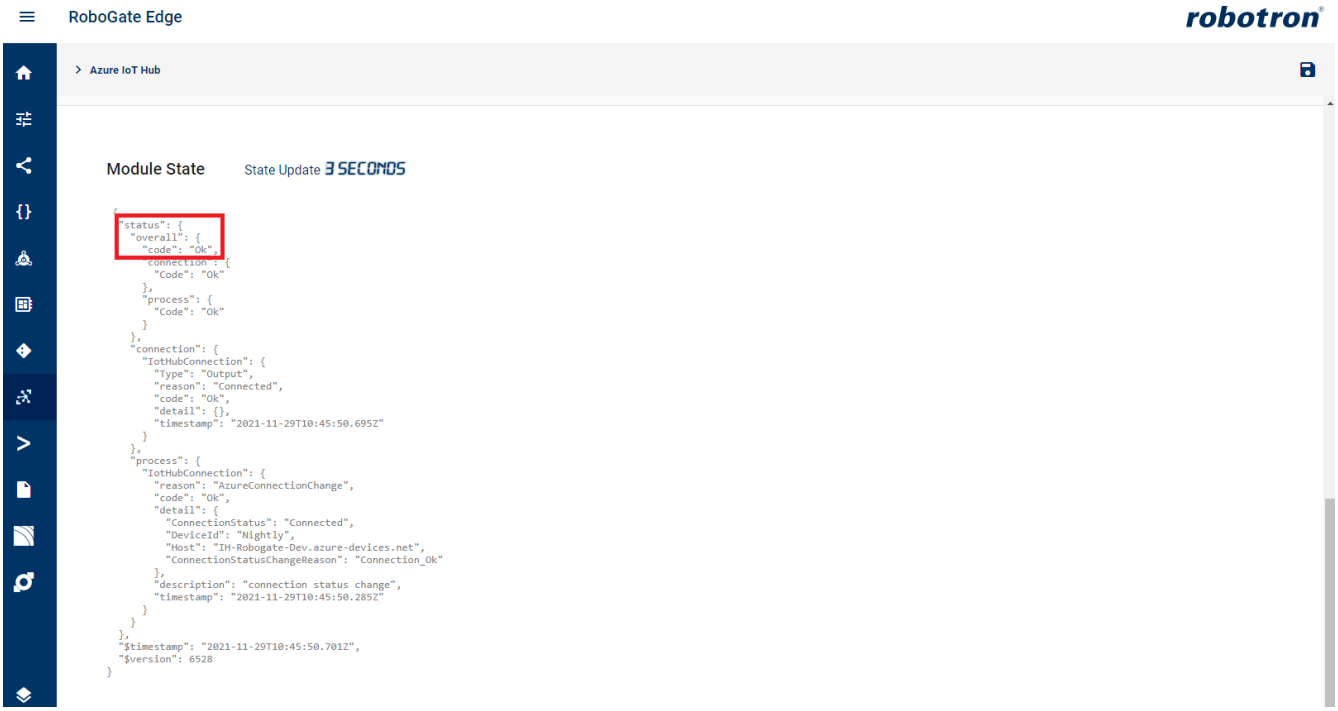


Abbildung 120. System Management - Azure Modulstatus

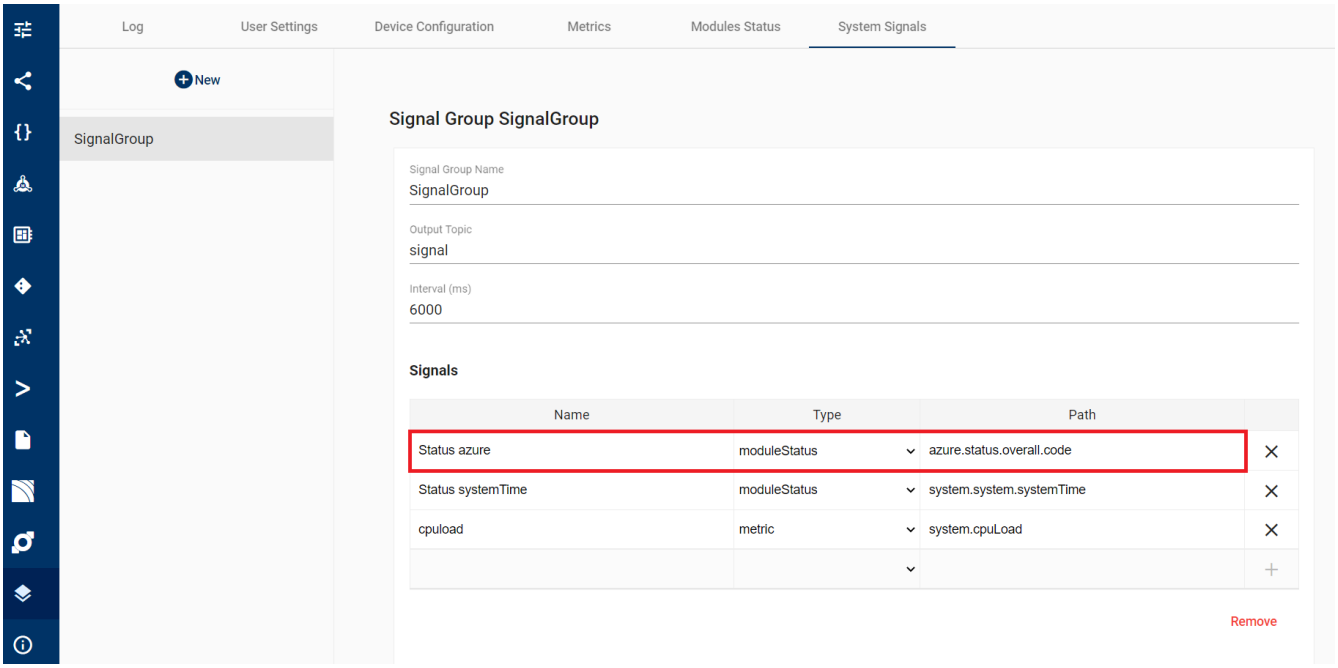


Abbildung 121. System Management - Beispielkonfiguration Signals

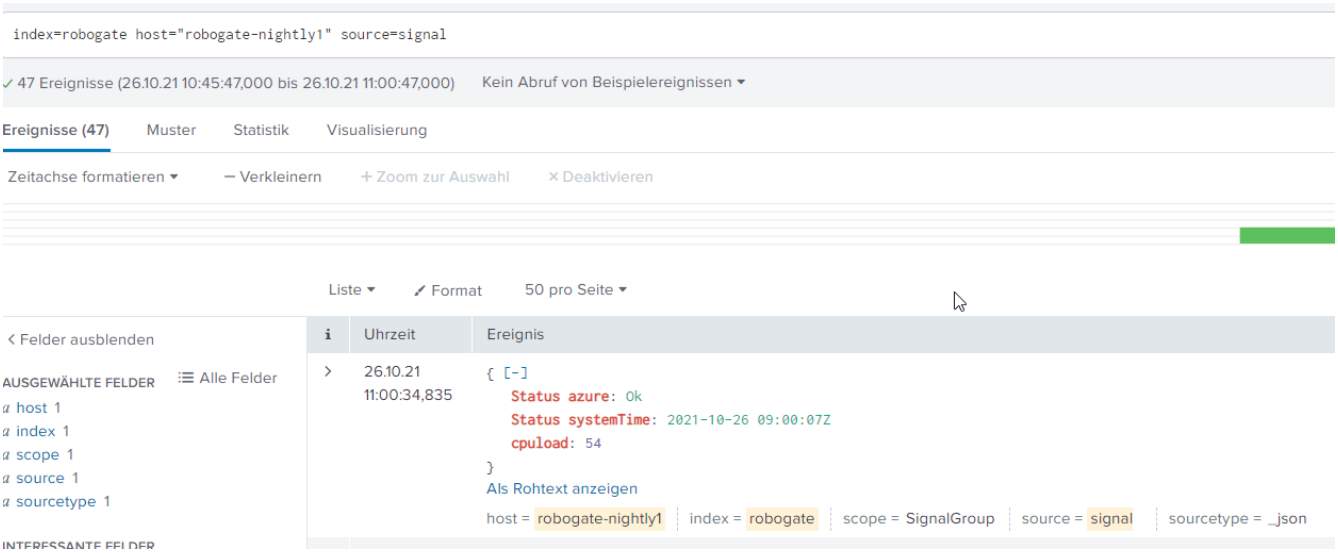


Abbildung 122. Beispiel - Signalmeldungen in Splunk

6.4. About us

Im Bereich About Us wird ein Überblick über die Software des EdgeGateway gegeben.

6.4.1. Software Versions

Der Reiter „Software Versions“ (Abbildung 123) gibt Auskunft über die Versionierung der Benutzeroberfläche, der Module und der zugehörigen Bibliotheken. Über die Suchfunktionalität „Search“ können Versionen bestimmter Komponenten einfach und schnell gefunden werden.

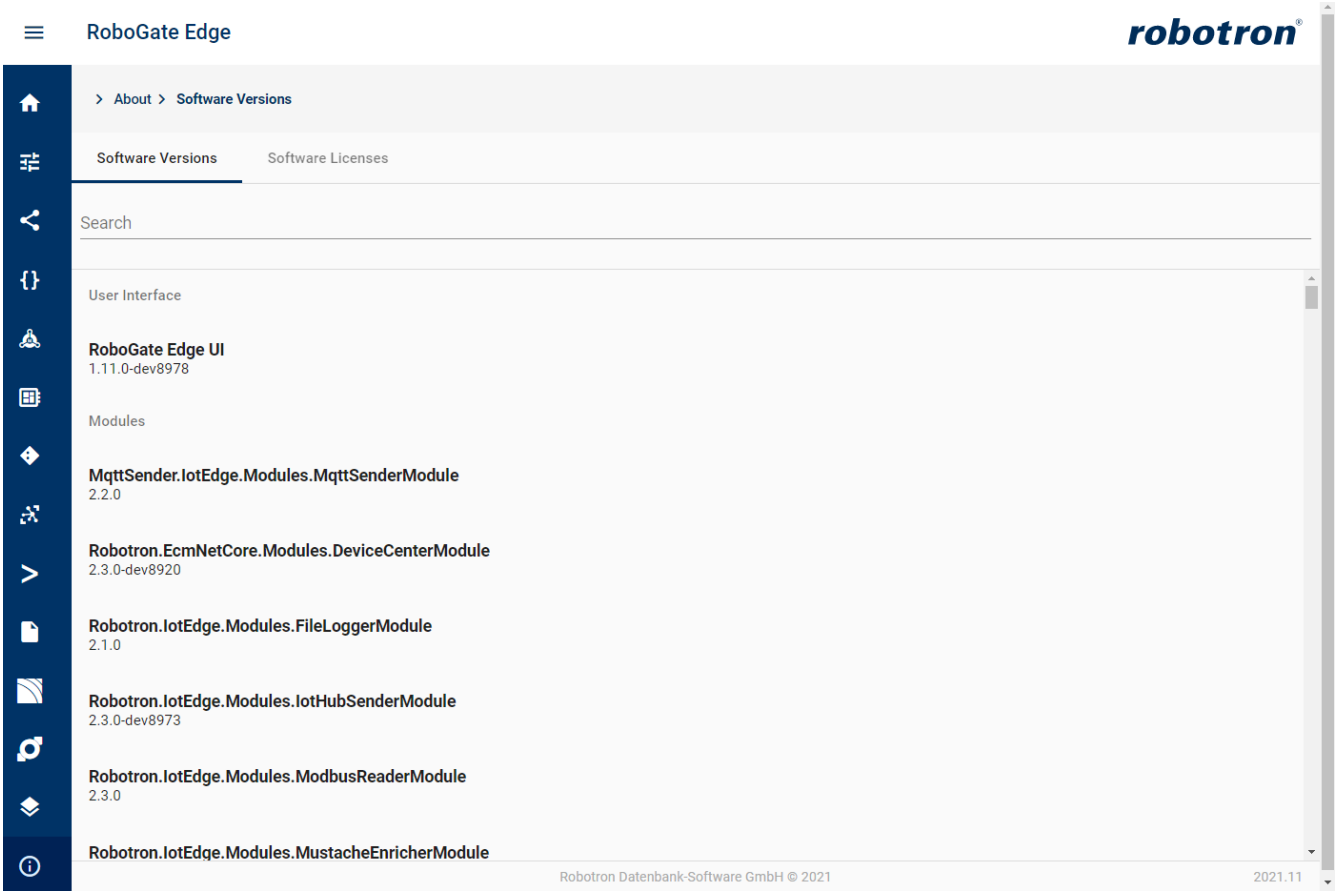


Abbildung 123. About us - Versions

6.4.2. Software Licenses

Der Reiter "Software Licenses" (Abbildung 124) gibt Auskunft über die Open Source Software Lizenzen der verwendeten Systembibliotheken.

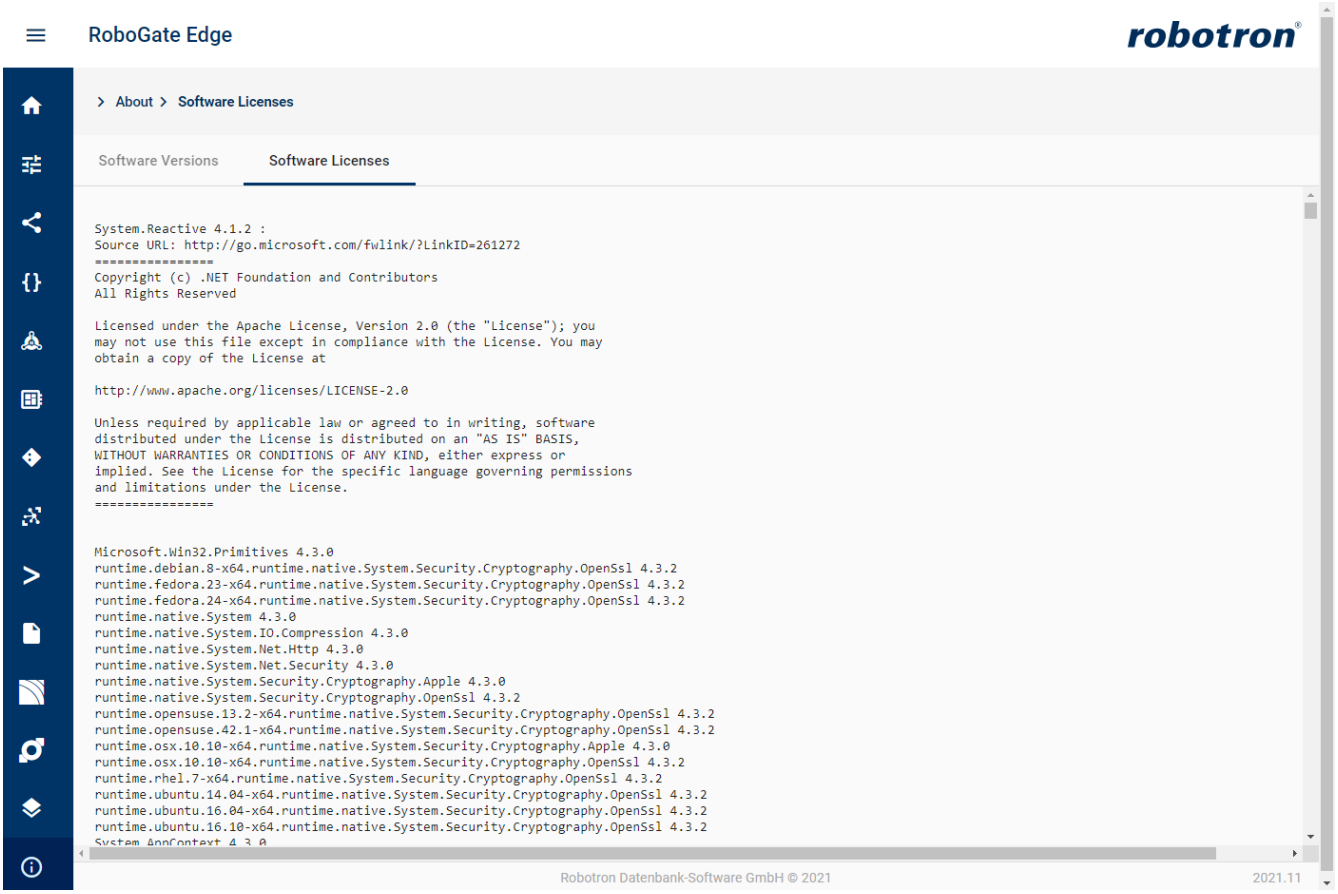


Abbildung 124. About us - Software Licenses

[1] https://www.docker.com/reSources/what-container

[2] https://docs.docker.com/compose/

[3] https://github.com/lunet-io/scriban

[4] https://portal.azure.com

[5] Informationen zur Einrichtung des HTTP Event Collectors sind der Dokumentation der Splunk Inc. zu entnehmen. https://docs.splunk.com/Documentation/Splunk/latest/Data/UsetheHTTPEventCollector