FORCAM FORCE™ EDGE

FORCAM FORCE EDGE

Version 210825

Handbuch

	Dokument: Handbuch - FORCAM FORCE EDGE.docx
	Freigabedatum: 26.10.2021
	Dokumentversion: 3
	Autor: AEgilmez, STernes

FB_8-41 ERSTELLT AM: 17.05.21 GEÄNDERT AM: 26.10.21 VERSION: V1
FREIGEGEBEN DURCH: HEAD OF TTE AUTOR: TECHNICAL WRITER VERTRAULICHKEITSSTUFE: ÖFFENTLICH

Inhaltsverzeichnis

1 Konzept

FORCAM FORCE EDGE bietet produzierenden Unternehmen eine Lösung zur digitalen Anbindung ihres heterogenen Maschinenparks. Mit FORCAM FORCE EDGE lassen sich nahezu alle Maschinen digitalisieren, unabhängig von Alter und technischem Stand. Dadurch unterstützt FORCAM die digitale Transformation einer Fertigungsanlage im Brownfield-Umfeld.

FORCAM liefert damit ein Produkt, welches die Kernanforderung von Industrie 4.0 durch das Gewinnen von digitalen Informationen aus dem Maschinenpark der Produktion löst. FORCAM leistet somit einen maßgeblichen Beitrag zur digitalen Transformation, indem die Kluft zwischen IT (Informationstechnik) und OT (operative Technologie) geschlossen wird.

FORCAM FORCE EDGE verschalt die Vielseitigkeit der Maschinenanbindungen und -signale und liefert diese als standardisierte Events an übergeordnete Systeme. Diese können unter anderem ME- oder MOM-Systeme wie beispielsweise SAP DMC/ME oder MII sein. Damit reduziert FORCAM den Aufwand bei der Digitalisierung und schafft eine standardisierte Schnittstelle zum Maschinenpark. Die Anbindung der Maschinen erfolgt über ein innovatives Plug-in-Konzept für die vereinfachte zukünftige Erweiterung. Aktuell werden alle gängige Maschinenhersteller-spezifische (proprietäre) Protokolle unterstützt (wie z. B. HEIDENHAIN, Siemens S7 oder FANUC & Co.) sowie alle gängige Kommunikationsstandards (wie z. B. MTConnect, OPC-UA oder MQTT). Für nicht netzwerkfähige Maschinen steht der FORCAM I/O Controller als separate Hardware zur Digitalisierung der Maschine zur Verfügung. FORCAM FORCE EDGE wird stetig um Plug-ins erweitert, um den Anspruch zu verwirklichen, jeden Maschinentyp über die Edge-Lösung digital abbildbar zu machen.

Aus den Maschinenanbindungen werden unterschiedlichste Informationen gewonnen. Dazu zählen Informationen über den aktuellen Status der angeschlossenen Maschinen oder deren Sensormesswerte wie z. B. Temperaturen, Drücke oder Energieverbrauch. Gerade im Brownfield-Umfeld ist es wichtig, nicht nur die Signale abzugreifen und weiter zu reichen, sondern diese auch für die Nutzung zu interpretieren. Diese Aufgabe übernimmt der EDGE Composition Layer. Grundlegend wichtig ist zum Beispiel die Interpretation, wann sich eine Maschine tatsächlich in Produktion oder im Stillstand befindet. FORCAM FORCE EDGE ist nicht nur in der Lage, Maschinensignale zu lesen oder zu schreiben. Ein weiterer wesentlicher Bestandteil der Lösung ist der Umgang mit NC-Programmen und der Möglichkeit, diese von und zur Maschine zu übertragen.

Die moderne, klar strukturierte Menüführung von FORCAM FORCE EDGE erlaubt es, mit den vorhanden Steuerungs- und Signalinformationen schnell und effizient Maschinen digital anzubinden.

Die Machine Repository-Komponente ermöglicht die einfache Erstellung und Verwendung von Maschinen-Templates. Diese erlaubt es, für Maschinenanbindungen Vorlagen zu definieren oder aus bestehenden Anbindungen abzuleiten und für die Anbindung von gleichen Maschinentypen zu verwenden. Dadurch wird der individuelle Aufwand für die Anbindung einer Maschine noch einmal deutlich reduziert, was das Digitalisierungsprojekt zeit- und ressourcenschonend umsetzt. Die Template-Struktur sorgt für eine standardisierte Anbindung von gleichen Maschinen und ermöglicht dadurch die Vergleichbarkeit von Maschinen des gleichen Typs.
FORCAM stellt für gängige Maschinen standardmäßige Templates zur Verfügung.

FORCAM FORCE EDGE ist flexibel einsetzbar und kann auf jedes produzierende Unternehmen angewandt werden. Die einzelnen Bausteine der Lösung können in verschiedene Bereiche und Ebenen verortet werden und bringen auf jeder Ebene Vorteile mit sich.

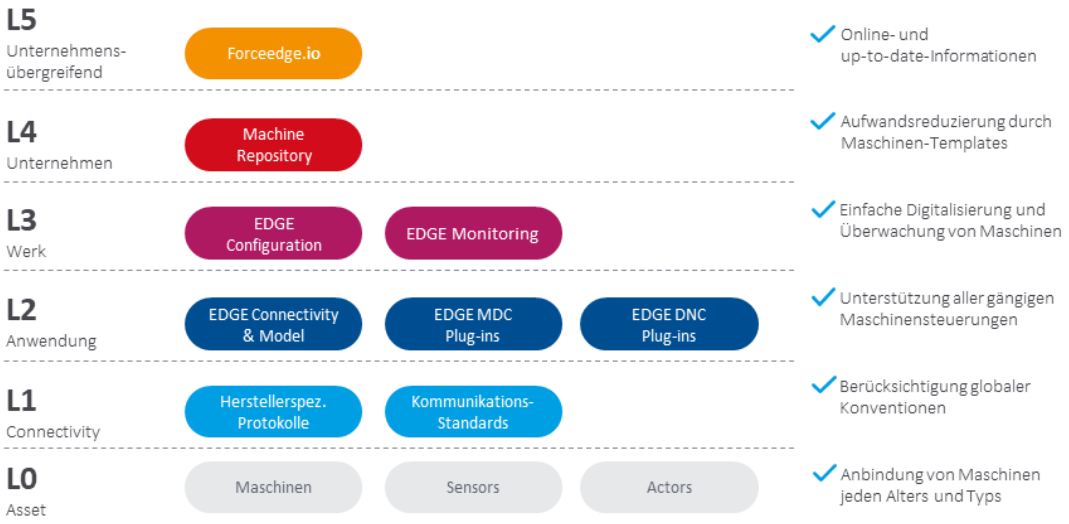


Bild 1: Verortung der Lösungsbausteine von FORCAM FORCE EDGE

Das folgende Bild zeigt die Referenzarchitektur der Open Industry 4.0 Alliance, die auch Grundlage der FORCAM FORCE EDGE Architektur ist. FORCAM leistet damit einen maßgeblichen Beitrag zur Digitalisierung in der Industrie und setzt den Fokus dabei auf den Kundennutzen. Die Vernetzung von Hardware durch intuitive und benutzerfreundliche Software zeichnet FORCAM FORCE EDGE besonders aus.

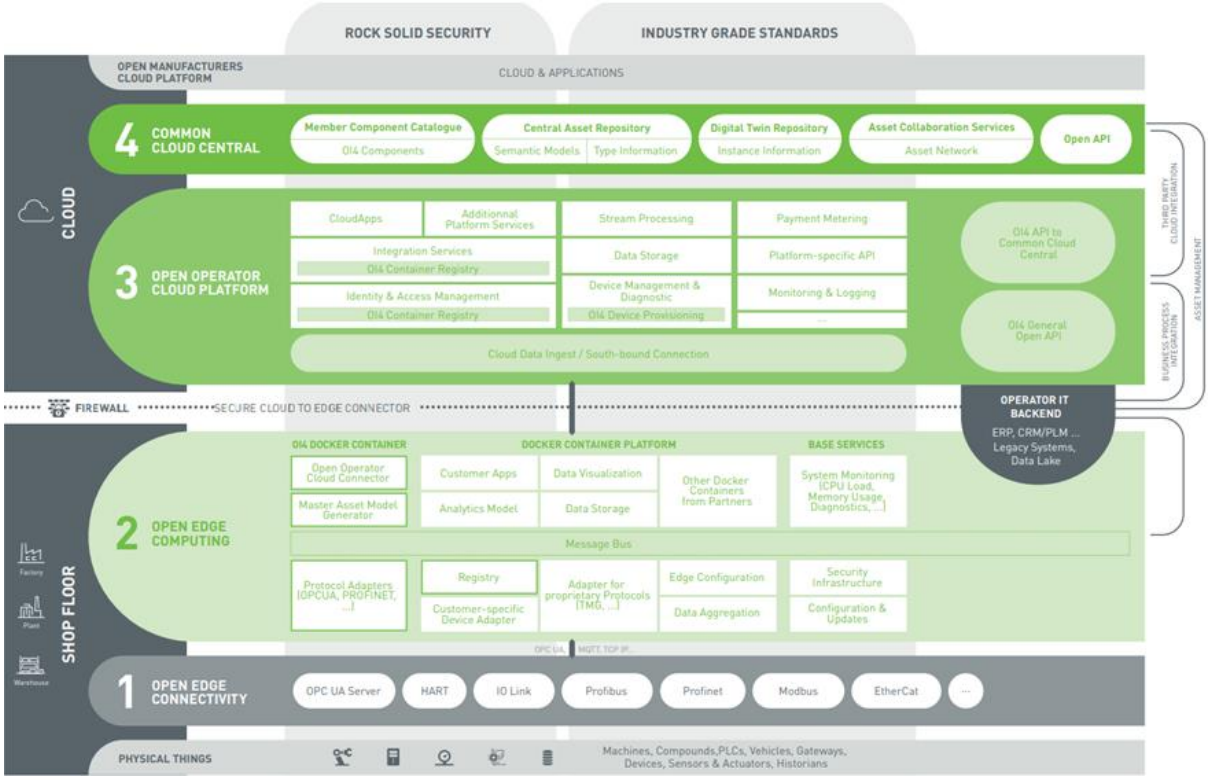


Bild 2: Referenzarchitektur der Open Industry 4.0 Alliance

2 Systemkomponenten

In diesem Kapitel werden die einzelnen Komponenten von FORCAM FORCE EDGE und deren Aufgaben erläutert.

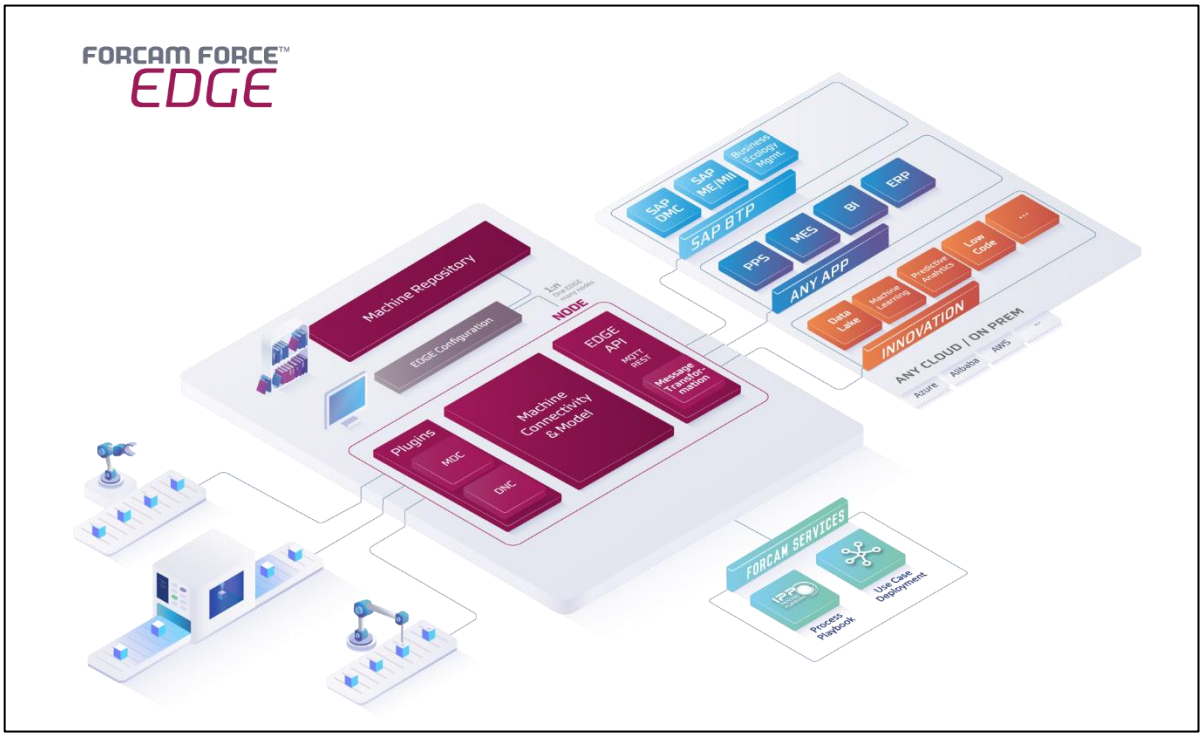


Bild 3: Schematischer Aufbau von FORCAM FORCE EDGE

2.1 Machine Connectivity & Model

Die zentrale Systemkomponente **Machine Connectivity & Model** von FORCAM FORCE EDGE ist das wesentliche Element der Maschinenanbindung und enthält folgende relevanten Subkomponenten:

EDGE MDC Layer

Der EDGE MDC Layer verwaltet die konkrete Anbindung der Maschinen. Die wesentlichen Elemente sind die Auswahl des geeigneten Plug-ins zur Kommunikation mit der Maschinensteuerung, der Konfiguration der Maschinenstammdaten, die Einstellung der Netzwerkverbindung und die Definition der Maschinensignale. Zudem leitet der EDGE MDC Layer Maschinensignale an den EDGE Composition Layer weiter.

EDGE DNC Layer

Der EDGE DNC Layer verwaltet die konkrete Anbindung von Maschinen mit einer NC-Versorgung. Die wesentlichen Elemente sind hier die Auswahl des geeigneten Plug-ins zur Kommunikation mit der Maschinensteuerung, der Konfiguration der Maschinenstammdaten, die Einstellung der Netzwerkverbindung und die Konfiguration der DNC-Versorgung.

EDGE Composition Layer

Der EDGE Composition Layer ermöglicht das Ableiten von logischen Maschinenzuständen. Mit einer einfachen Skriptsprache lassen sich aus Signalkombinationen Statusevents ableiten. Dies ist ein wesentlicher Erfolgsfaktor im Brownfield, da gerade ältere Maschinen bedient werden können, die ansonsten keine verwertbaren Informationen liefern. Die Vereinheitlichung der Reportfähigkeit wird hierzu vom Composition Layer gewährleistet. Dies geschieht über eine Skriptsprache. Zusätzlich werden auch Möglichkeiten für individuelle Events geschaffen. Durch die Skripts gibt es auch die Möglichkeit, auf Ereignisse zu reagieren und Werte in die Steuerungseinheit der Maschine schreiben.

2.2 Plug-ins

Plug-ins im Umfeld von FORCAM FORCE EDGE implementieren die Kommunikationsverbindung mit spezifischen Maschinensteuerungen. Sie erlauben eine direkte Kommunikation mit verschiedenen Maschinensteuerungen, decken aber auch moderne Kommunikationsprotolle wie beispielweise MQTT, OPC UA und viele mehr ab. Das Plug-in-Konzept von FORCAM FORCE EDGE ist erweiterbar, FORCAM baut die Anzahl der unterstützten Plug-ins stetig aus.

Die Plug-ins gliedern sich in Plug-ins zur Maschinendatenerfassung (MDC) und für Distributed Numerical Control (DNC).
MDC-Plug-ins beinhalten sowohl solche zum unidirektionalen Auslesen von Maschinensignalen als auch für eine bidirektionale Signalübertragung, also dem Auslesen und Zurückschreiben von Signalen.
DNC-Plugins werden für das Übertragen und Auslesen von NC-Dateien verwendet. Mit ihrer Hilfe werden NC-Programme an das Dateisystem der Maschine übertragen oder das an der Maschine aktive Programm abgefragt.

Für die gängigsten Steuerungstypen werden Plug-ins in FORCAM FORCE EDGE standardmäßig mitgeliefert. Eine Übersicht der aktuellen FORCAM Plug-ins ist in Abschnitt 8.4 aufgelistet.

⚠ Das Bereitstellen, Bearbeiten oder Verwalten von NC-Programmen ist keine Funktion von FORCAM FORCE EDGE

2.3 Data Lake

Um einen digitalen Zwilling einer Maschine oder Steuerung zu erhalten, ist es nicht nur wichtig, die Verbindung zur Maschine herzustellen, die Signale zu interpretieren und an andere Anwendungen weiterzugeben, sondern auch die Daten zu speichern. Mit Data Lake werden alle Daten auf der Signalebene, der Interpretationsebene und der Event-Ebene gespeichert, einschließlich Konfigurationsänderungen, Schreibvorgängen und übertragenen NC-Dateien.

Die Daten werden über unsere EDGE API zur Verfügung gestellt. So können die neuesten KI-Algorithmen, Visualisierungstools, aber auch Audit-Anforderungen davon profitieren. Dies ist die Basis für Data-Driven-Manufacturing.

2.4 EDGE API

Über die EDGE API als RESTful API erfolgt der Abruf von Maschinenstammdaten und die Konfiguration der Maschinenanbindungen. Der EDGE API Eventservice dient zur Weitergabe von Maschinendaten in Form von standardisierten Events an übergeordnete Systeme (3rd-Party). Die Anbindung von übergeordneten Systemen kann entweder über HTTP/REST oder MQTT erfolgen. Die Events werden per HTTP im JSON-Format übertragen. Optional kann ein MQTT-Broker als Middleware eingesetzt werden.
Die EDGE API wird mit vorkonfigurierten Standardevents zur Kommunikation mit der MES- oder ERP-Ebene ausgeliefert. Diese lassen sich bei Bedarf weiter individualisieren.

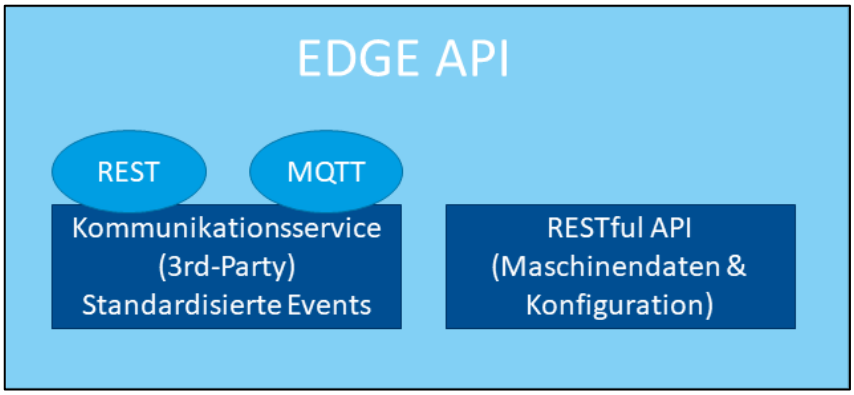


Bild 4: Aufbau der BRIDGE API

2.5 EDGE-Komponenten

2.5.1 EDGE Configuration

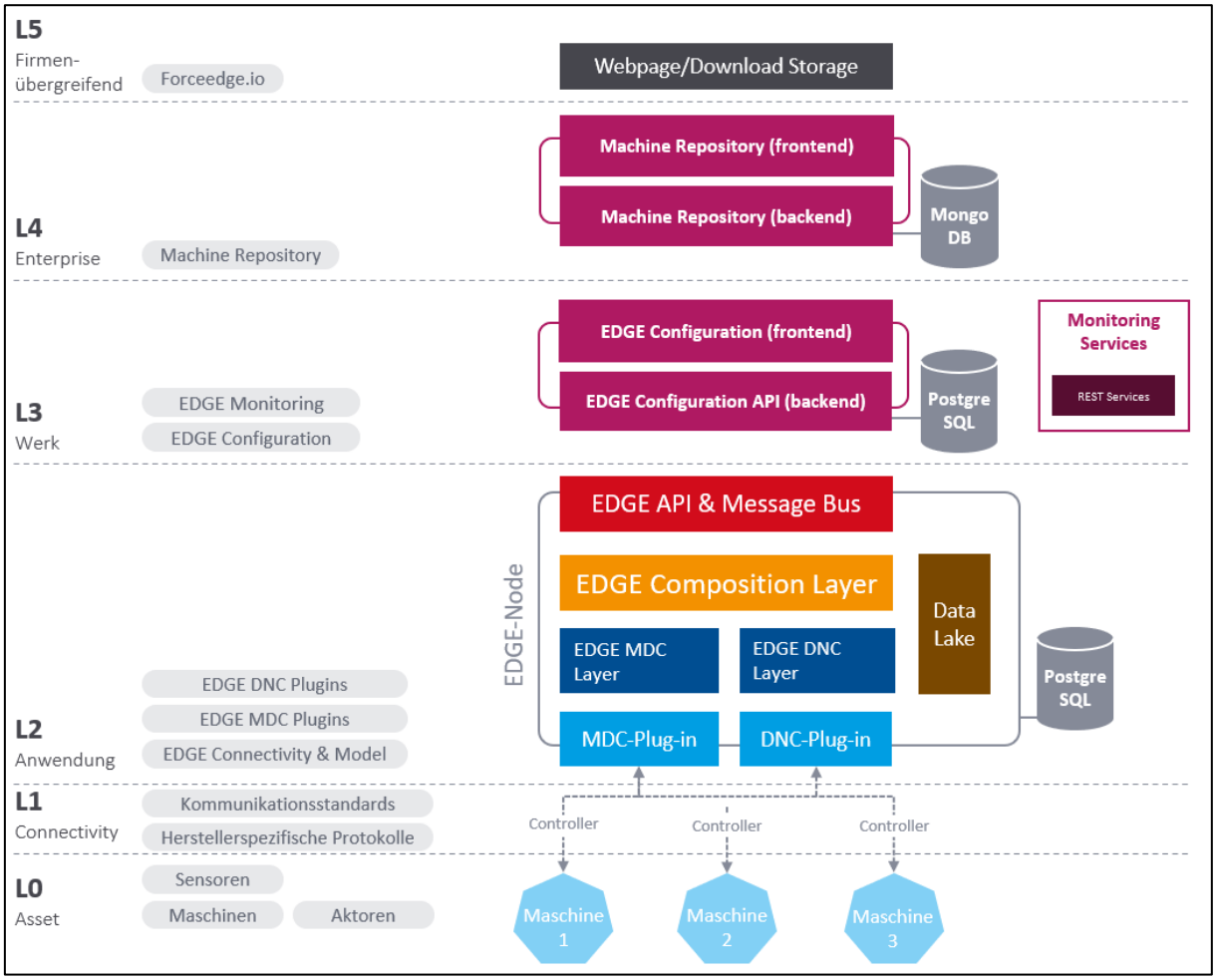
EDGE Configuration ist die Verwaltungsoberfläche für FORCAM FORCE EDGE. Mit ihr lassen sich mehrere EDGE-Knoten verwalten. Ein EDGE-Knoten ist die Bündelung der Signalerfassung von mehreren Maschinen. Je nach Datenmenge werden ein oder mehrere EDGE-Knoten pro Werk eingesetzt. Die Verwaltung der Knoten erfolgt zentral.

2.5.2 Machine Repository

Das Machine Repository erlaubt es, aus bestehenden oder für neue Maschinenanbindungen Templates zu generieren. Mit diesen Templates können Maschinen des gleichen Typs und der gleichen Nutzungsart einheitlich angebunden werden. Das Template enthält alle Konfigurationselemente, welche nicht maschinenindividuell sind. Maschinen- und Maschinenverbindungsspezifische Konfigurationselemente sind beispielsweise IP-Adresse, Maschinenbezeichnung, Equipment-Nummer etc.
Durch das Verwenden eines bestehenden Templates wird der Zeitaufwand zur Anbindung einer Maschine deutlich reduziert.

3 Systemarchitektur

FORCAM FORCE EDGE ist architektonisch in Level (Schichten) unterteilt. Diese orientieren sich nach der betriebswirtschaftlichen Nutzung, was eine hohe Skalierbarkeit der einzelnen Komponenten ermöglicht. So können beispielsweise mehrere Edge-Knoten gehostet werden, um die Maschinen logisch, aber auch Performance-orientiert aufzuteilen.



Level 0 - Assets

Auf der untersten Schicht erfolgt die Anbindung der Maschinen, Sensoren und Aktoren. Hierbei gilt das Werteversprechen, jede Maschine hinsichtlich Typs und Alter zu unterstützen.

Level 1 - Connectivity

Die wachsende Auswahl an Plug-ins ermöglicht die Anbindung vielfältiger Steuerungen mit deren unterschiedlichen Kommunikationsstandards wie OPC UA oder MT Connect sowie herstellerspezifischen Protokollen.

Level 2 - Anwendung

Die Anzahl der möglichen Edge-Knoten ist nicht begrenzt. Ein Knoten umfasst mehrere Schichten bzw. Aufgaben:

- **EDGE MDC Layer** steuert die Anbindung der Maschine mittels **Plug-ins** und leitet die Signale an **EDGE Composition Layer** weiter. Analog dazu werden DNC-fähige Maschinen von **EDGE DNC Layer** angebunden.
- **EDGE Composition Layer** ist zuständig für die Signalinterpretation und das Komponieren der standardisierten Events.
- **EDGE API** ist die Programmierschnittstelle, über welche Maschinen, das Management von Events und die Benachrichtigung von Drittsystemen „northbound“ konfiguriert und durchgeführt werden können.
- **Data Lake** hält alle Daten, Konfigurationsänderungen sowie Schreiboperationen und übertragene NC-Daten fest. Die Daten sind über die EDGE API abrufbar.
- **Postgre SQL** enthält alle Konfigurationen in Bezug auf die Maschinenanbindung und arbeitet als Knoten autark.

Level 3 - Werk

Die Konfigurationskomponente kann 1 bis n Edge-Knoten bedienen. Es ist auch möglich, diese pro Werk, aber auch pro Fertigungslinie bereitzustellen.
Jede Komponente ist für sich eigenständig und ohne eine Verbindung lauffähig (etwa durch zeitweisen Verlust). Dies ermöglicht unterschiedlichste Deployment-Modelle. So muss die Edge-Konfiguration beispielsweise nicht zwingend in FORCAM FORCE EDGE selbst gehostet sein, sondern lediglich eine Verbindung zur jeweiligen API aufgebaut werden.
Grundlegend kommunizieren alle Komponenten über standardisierte Schnittstellen (HTTP/REST).

Level 4 - Enterprise

Das Machine Repository ist eine Komponente von FORCAM FORCE EDGE, welches das Erstellen und Verwalten von Maschinenanbindungsvorlagen erlaubt.

Level 5 - Firmenübergreifend

FORCAM FORCE EDGE bietet über forceedge.io eine frei verfügbare Sammlung von Vorlagen für die Maschinenkonfiguration. Diese können über forceedge.io heruntergeladen und dann in das eigene Machine Repository importiert werden.

4 Deployment

FORCAM FORCE EDGE kann über eine klassische Lizenz oder direkt als Software as a Service (SaaS) erworben und installiert werden.

Bild 5: Möglichkeiten der Installation von FORCAM FORCE EDGE

Lizenzmodell

Bei einer Installation über das klassische Lizenzmodell werden zwei Installer bereitgestellt: **ForceEdgeUI** und **ForceEdgeNode**. Diese werden vom Kunden selbst, oder vom einem FORCAM Servicedienstleister installiert.
ForceEdgeUI beinhaltet die gesamte Benutzeroberfläche inklusive aller Funktionen und wird zuerst installiert. ForceEdgeNode umfasst den Edge-Knoten und kann beliebig oft installiert werden, da die Anzahl der Knoten in FORCAM FORCE EDGE nicht begrenzt sind. Die Anzahl richtet sich nach der erworbenen Lizenz. Hier ist definiert, wie viele Knoten erstellt und wie viele Controller pro Knoten angeschlossen werden können.

SAP DMC

FORCAM FORCE EDGE kann als Solution-Erweiterung von **SAP Digital Manufacturing Cloud** (SAP DMC) auf der SAP Business Technology Platform (SAP BTP) erworben werden.
Bei einem Erwerb als SaaS liefert FORCAM die Hardware, auf der FORCAM FORCE EDGE betrieben wird. Hierzu wird ein Microsoft Azure Stack Edge (ASE) verwendet, was von FORCAM vorkonfiguriert wird. FORCAM benötigt hierfür feste IP-Adressen, die kundenseitig manuell eingetragen oder den Servicedienstleistern von FORCAM mitgeteilt werden müssen. Die Eingliederung des ASE in die bestehende Hardwareumgebung obliegt dem Kunden. FORCAM gewährleistet die Funktion und Verfügbarkeit des ASE und der Softwarekomponenten.

5 Grundlegende Einstellungen

Für allgemeine Einstellungen zu FORCAM FORCE EDGE steht der Bereich **Home** zur Verfügung. Neben der Verwaltung von Benutzern und Rechten können hier ebenfalls aktuelle Dokumente heruntergeladen werden. Dieses Kapitel geht auf die **Benutzerverwaltung, Lizenzierung** und den **Download-Bereich** ein.

ⓘ Die im Profil vorgenommenen Einstellungen bzgl. Sprache und Darkmode werden im Benutzerprofil gespeichert und gelten nur für diesen Benutzer.

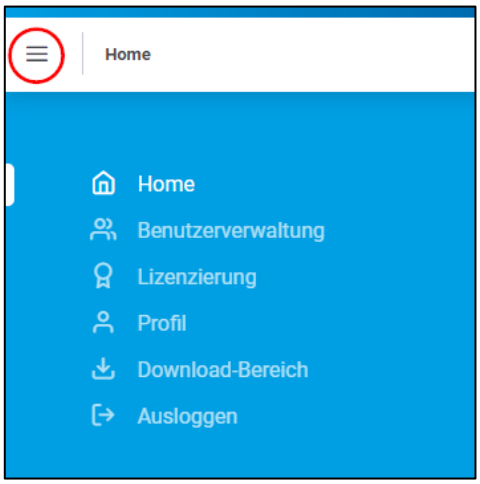


Bild 6: Aufruf des Home-Bereichs

5.1 Benutzerverwaltung

In der Benutzerverwaltung können Benutzer für FORCAM FORCE EDGE erstellt werden. Jedem Benutzer können Rechte zugewiesen werden, die nur die für ihn angemessenen oder beabsichtigten Funktionen enthalten (z. B. Maschine konfigurieren, Knoten neu starten etc.). Vorhandene Benutzeraccounts können auch nachträglich editiert werden.

ⓘ Wurden die Rechte eines eingeloggten Benutzers geändert, werden sie nach einem neuen Login sofort wirksam. Loggt sich der Benutzer nicht erneut ein, kann es bis zu 30 Minuten dauern, bis die Änderung aktiv ist.

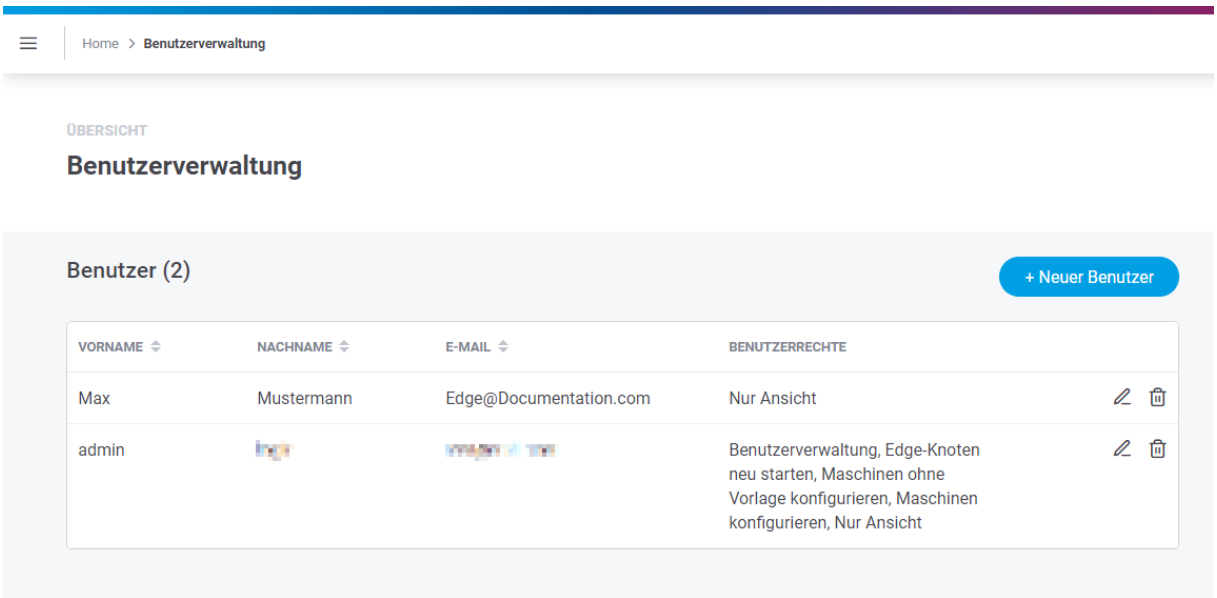


Bild 7: Benutzerverwaltung von FORCAM FORCE EDGE mit 2 Benutzern

Um einen neuen Benutzer zu erstellen:

1. Auf **+ Neuer Benutzer** klicken.
2. Im Folgedialog eine Mail-Adresse, Vor- und Nachnamen eintragen.
3. Gewünschtes Passwort setzen.
   Muss mind. 8 Zeichen lang sein, aus Groß- und Kleinbuchstaben bestehen und mind. eine Zahl und ein Sonderzeichen enthalten.
   Folgende Sonderzeichen sind erlaubt: @$!%*?&
4. Benutzerrechte vergeben (s.u.).
5. Speichern.

ⓘ Ein Benutzer kann mit denselben Daten kein weiteres Mal angelegt werden.

Tabelle 1: Benutzerrechte in FORCAM FORCE EDGE

Benutzerrecht	Beschreibung
Benutzerverwaltung	Der Benutzer kann die Benutzerverwaltung aufrufen, neue Benutzer erstellen und Rechte vergeben/entfernen.
Edge-Knoten neu starten	Der Benutzer kann einen Edge-Knoten neu starten.
Maschinen ohne Vorlage konfigurieren	Der Benutzer kann Maschinen ohne MR-Templates anlegen.
Vorlagenbasierte Maschinenwerte ändern	Der Benutzer kann MR-Templates anpassen (Signale und Skript).
Maschinen konfigurieren	Der Benutzer kann Maschinen nur mit MR-Templates anlegen. Änderungen können nicht vorgenommen werden.
Nur Ansicht	Der Benutzer kann den Edge-Knoten nur einsehen. Änderungen können nicht vorgenommen werden.

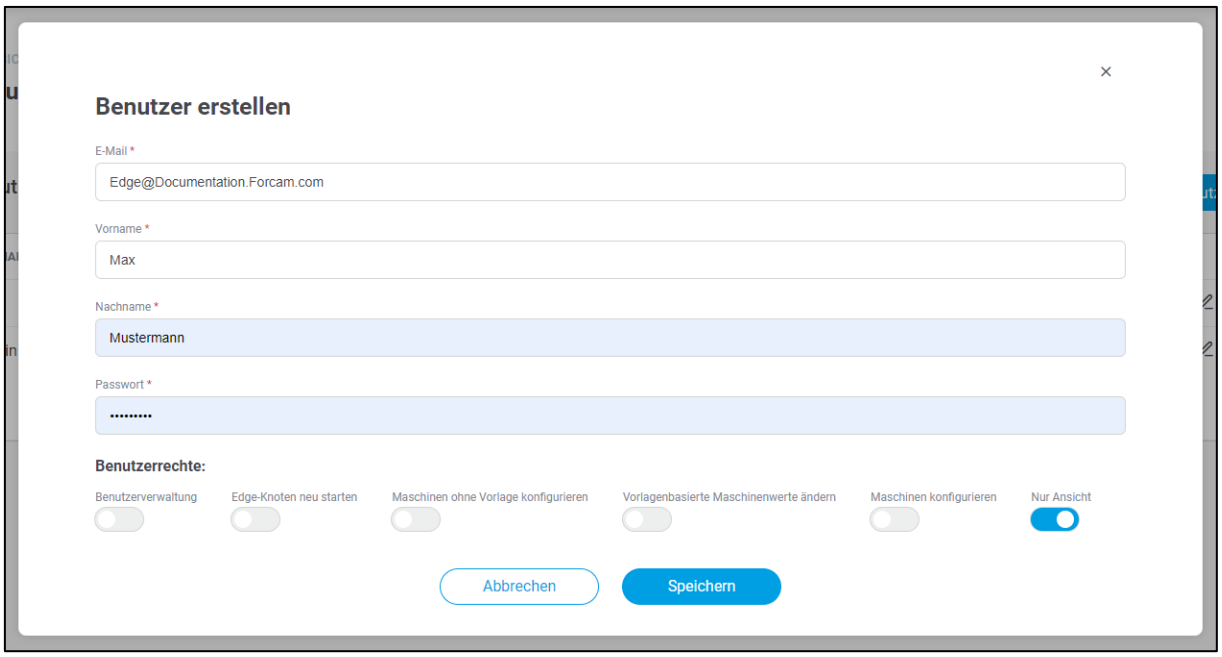


Bild 8: Dialog zu Erstellen eines neuen Benutzers

5.2 Lizenzierung

Unter **Lizenzierung** können Lizenzen eingespielt und eingesehen werden.

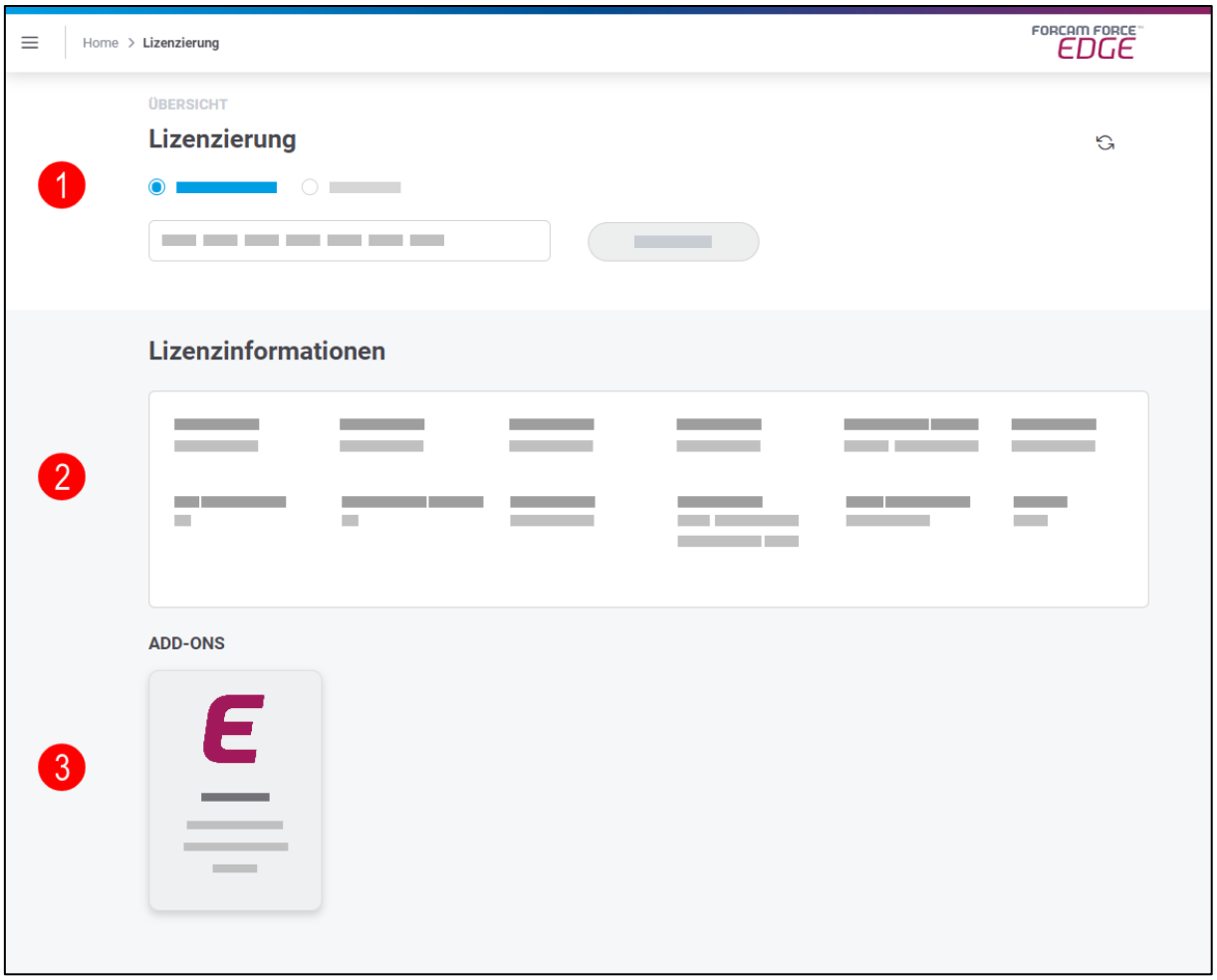


Bild 9: Lizenzierung und Übersicht

(1) Eine neue Lizenz kann als Datei hochgeladen oder direkt als Schlüssel eingetragen werden.
(2) Die Lizenzinformationen umfassen Typ und Status der Lizenz, Anzahl der lizenzierten Knoten und Maschinen, Wartung, Gültigkeit und weitere Daten.
(3) Alle gebuchten Add-ons werden hier aufgelistet.

5.3 Download-Bereich

Unter dem Reiter **Allgemein** kann die aktuelle Dokumentation von FORCAM FORCE EDGE in mehreren Sprachen heruntergeladen werden. Zur Verfügung stehen dabei das Handbuch und eine Produktbeschreibung. Das Handbuch ist das vorliegende Dokument mit genauen Anweisungen zur Konfiguration. Die Produktbeschreibung ist ein kürzeres Dokument, das lediglich Funktion und Nutzen der Anwendung beschreibt und die Leistung aufführt und abgrenzt.

In den Reitern **MDC Plug-ins** und **DNC Plug-ins** stellt FORCAM eine selbstentwickelte Anwendung bereit. Sie wird benötigt, um über das entsprechende Plug-in mit einer Maschine zu kommunizieren. Die Anwendung steht dabei softwareseitig zwischen dem EDGE MDC Layer und der Maschine und ermöglicht die (bi-)direktionale Kommunikation.

6 EDGE-Konfiguration

Die Konfiguration eines Edge-Knotens sowie einer Maschine findet vollständig in der EDGE Configuration von FORCAM FORCE EDGE statt. Die benutzerfreundliche Oberfläche führt durch alle relevanten Einstellungen und zeigt in der Übersicht alle Knoten und die Status an.

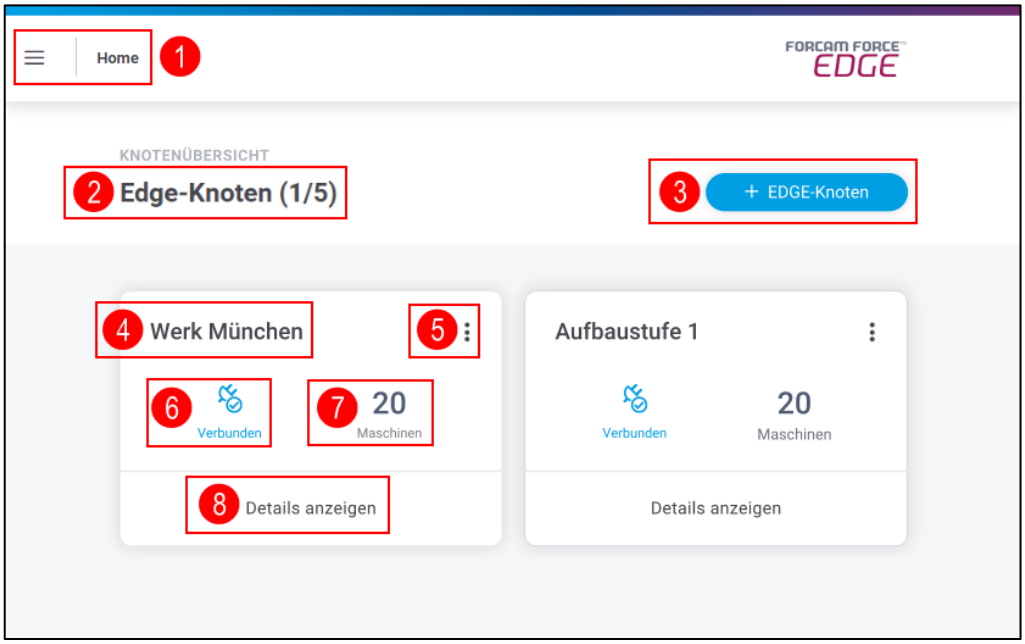


Bild 10: Einstiegs- und Übersichtsseite von FORCAM FORCE EDGE

(1) Einstellungsmenü von FORCAM FORCE EDGE
- Lizensierung
- Benutzerverwaltung
- Sprache

(2) Gibt an, wie viele Edge-Knoten konfiguriert sind (erste Zahl) und wie viele Knoten gemäß Lizenz konfiguriert werden können (zweite Zahl)

(3) Fügt einen neuen Edge-Knoten hinzu

(4) Name des Knotens

(5) Einstellungsmenü des Knotens:
- Editieren
- Event-Konfiguration
- Löschen

(6) Status der angebundenen Maschinen

(7) Anzahl der angebundenen Maschinen

(8) Weitere Detailinformationen des Knotens:
- Liste aller angebundenen Maschinen und deren Status
- Möglichkeit zum Hinzufügen einer neuen Maschine
- Monitoring angebundener Maschinen

ⓘ Änderungen in der Benutzerverwaltung bzgl. Benutzerrechte können bis zu 30 Minuten benötigen, bis sie im gesamten System greifen.

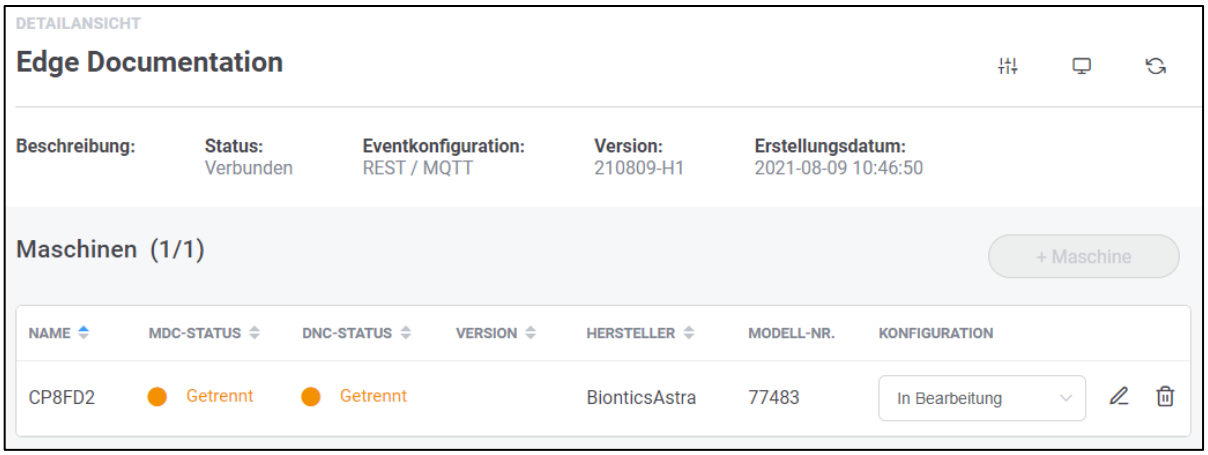


Bild 11: Maschinenübersicht als Folgeseite nach Klicken auf „Details anzeigen“

Die **TEMPLATE-VERSION** zeigt die aktuelle Ausführung an (siehe Abschnitt 4.2.1).

Unter **KONFIGURATION** kann manuell bestimmt werden, welchen Status die Konfiguration haben soll, um etwa Mitarbeitenden einen Überblick zu verschaffen oder diese hinzuzuziehen:

- In Bearbeitung:
  Die Konfiguration ist noch nicht abgeschlossen und soll zu einer anderen Zeit fortgesetzt werden.
- In Validierung:
  Die Konfiguration der Maschine soll auf Fehler und Konsistenz hin überprüft werden.
- Abgeschlossen:
  Die Konfiguration ist vollständig abgeschlossen. Nur in diesem Status kann der Lernzyklus des MR stattfinden und aus der Konfiguration ein Template generiert werden.

6.1 Edge-Knoten hinzufügen

In FORCAM FORCE EDGE können Knoten in wenigen Schritten hinzugefügt werden. Ein Edge-Knoten entspricht einer Instanz einer Anbindungsvariante. Pro Werk kann es mehrere Knoten geben. Sie werden logisch gebündelt, so dass die Last von Maschinen sinnvoll aufgeteilt wird.

ⓘ Wird ein konfigurierter Edge-Knoten aus der Oberfläche entfernt, bleibt seine Konfiguration erhalten. Wird der Knoten wieder unter denselben Daten angelegt, übernimmt er die vorher konfigurierten Daten automatisch.

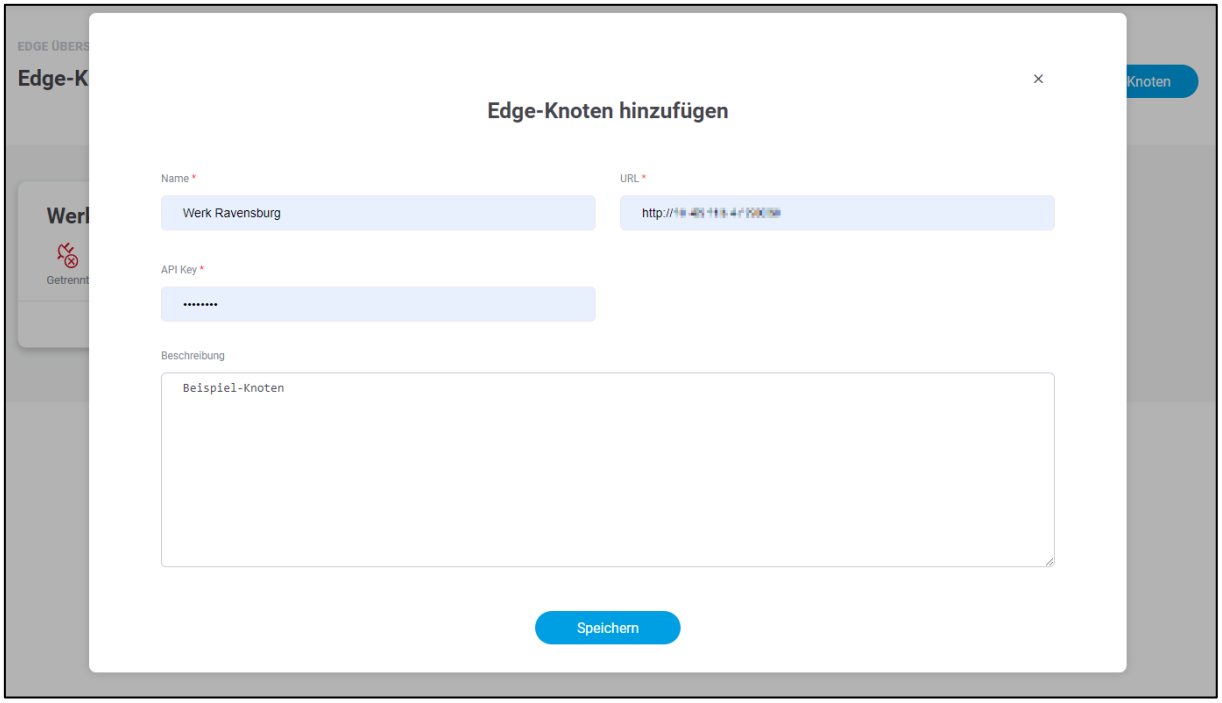


Bild 12: Dialog zum Hinzufügen eines neuen Knotens

Um einen neuen Edge-Knoten hinzuzufügen:

1. In der Knotenübersicht auf **+ Edge-Knoten** klicken.
2. Im Folgedialog alle obligatorischen Felder ausfüllen:
   - Name:
     Erscheint in der Knoten-Übersicht als Titel des Knotens
   - URL:
     Bestehend aus http + IP-Adresse + Host. Pro URL kann nur ein Edge-Knoten erstellt werden.
   - API Key:
     Passwort, das bei der initialen Installation des Knotens vergeben wurde
3. Optional: Beschreibung hinzufügen.
4. Speichern.

6.2 Maschine hinzufügen

Der Dialog zum Hinzufügen einer Maschine leitet durch acht Schritte durch, die für eine Anbindung nötig sind. Hier werden unter anderem MDC-/DNC-Steuerungen konfiguriert und Maschinensignale definiert.

ⓘ Negative Werte sind in der Maschinenkonfiguration nicht erlaubt.

ⓣ Ist ein Schritt abgeschlossen, wird er in der oberen Leiste blau markiert.
Durch Klicken auf einen abgeschlossenen Schritt kehrt man zu diesem zurück.
Beim Editieren einer bereits konfigurierten Maschine kann jede Konfigurationsseite direkt ausgewählt und aufgerufen werden.

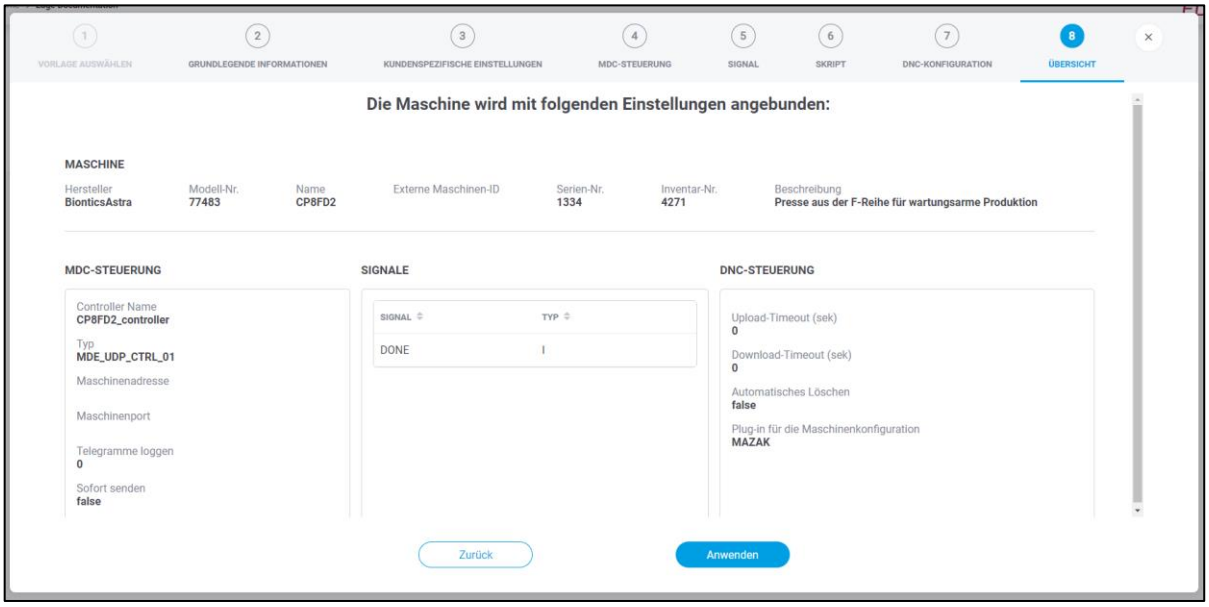


Bild 13: Dialog zur Konfiguration einer Maschine in FORCAM FORCE EDGE

Um eine Maschine hinzuzufügen:

1. In der Edge-Übersicht beim gewünschten Knoten auf **Details anzeigen** klicken.
2. In der Folgeseite auf **+ Maschine** klicken.

➔ Der Folgedialog führt durch die nachfolgenden acht Schritte zur Konfiguration einer Maschine.

6.2.1 ① Vorlage auswählen

Mehrere Maschinen vom selben Typ müssen in FORCAM FORCE EDGE nicht jedes Mal komplett neu konfiguriert werden: Eine einmal konfigurierte Maschine kann als Template im Machine Repository erfasst werden und wird dann bei der nächsten Maschinenanbindung in dieser Maske angeboten. Wird das Template hier ausgewählt, werden alle Einstellungen automatisch für diese Maschine übernommen und alle nicht maschinenspezifischen Konfigurationsfelder sind vorausgefüllt. Lediglich maschinenspezifische (z. B. Seriennummer) und verbindungsspezifische Informationen (z. B. IP-Adresse oder Port der Maschine bzw. Steuerung) müssen noch angepasst werden.

Die Template-**VERSION** gibt an, in welcher Ausführung es sich befindet. Wird ein Template überarbeitet, wird die Versionsnummer automatisch hochgezählt und die frühere Version überschrieben. Version 0 bedeutet, dass im entsprechenden Template kein Skript konfiguriert ist.

ⓘ Dieser Schritt ist nur verfügbar, wenn die MR-Komponente verwendet wird. Ist kein Template konfiguriert oder MR nicht in Verwendung, beginnt die Maschinenanbindung mit Schritt ②.

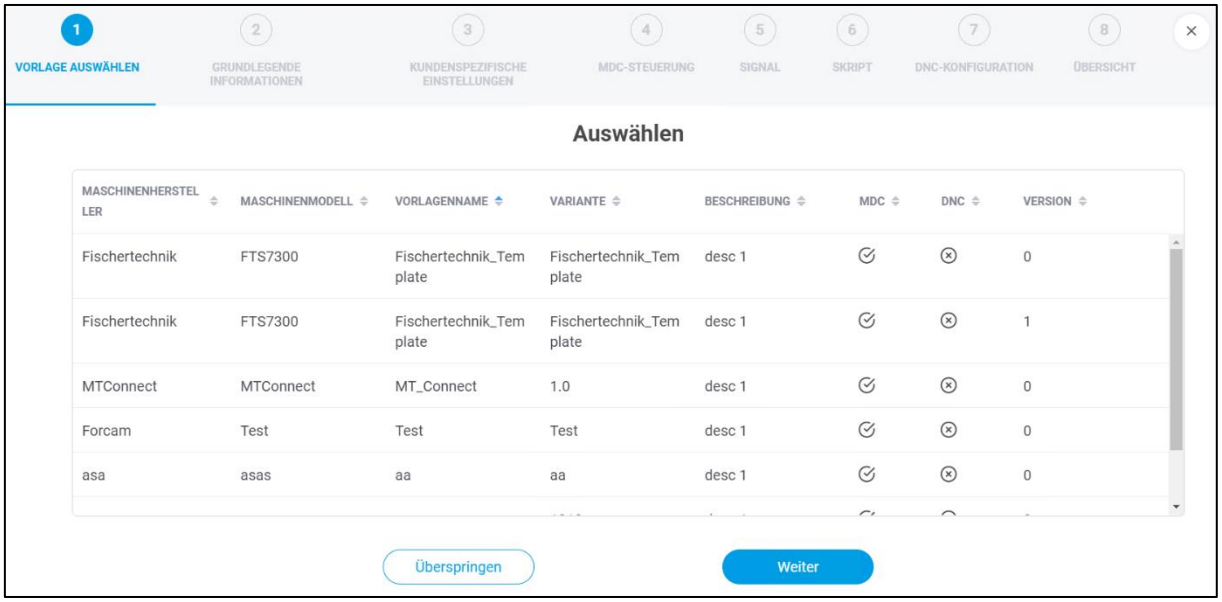


Bild 14: Maschine hinzufügen - Vorlage auswählen

1. Gewünschte Maschinenanbindung in der Liste auswählen.
2. Auf **Weiter** klicken.

ⓘ Wird beim nachträglichen Editieren einer bereits konfigurierten Maschine ein Template ausgewählt, überschreibt es die vorhandene Konfiguration nach einem Klick auf **Weiter**.
Aus diesem Grund muss das Template durch die Checkbox im unteren Bereich bestätigt werden. Die Checkbox wird erst sichtbar, wenn ein Template ausgewählt wurde.

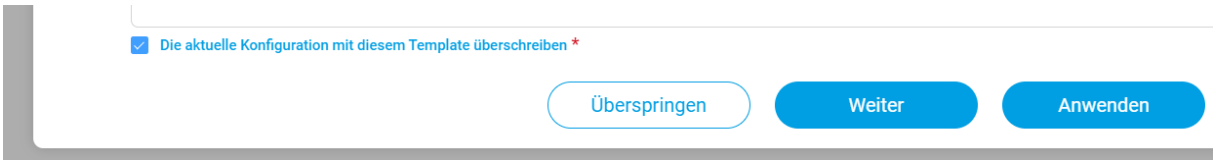


Bild 15: Bestätigung des Überschreibens der Konfiguration durch ein Template

6.2.2 ② Grundlegende Informationen

Basisinformationen der zu konfigurierenden Maschine wie z. B. Name oder Seriennummer. Hier wird zudem bestimmt, ob eine MDC- oder eine DNC-Steuerung konfiguriert werden soll – oder beides. Mit dem MDC-Controller werden Signale von der Maschine abgeholt und weitergegeben oder auf diese geschrieben. Über die DNC-Steuerung werden NC-Dateien an die Maschine übertragen.

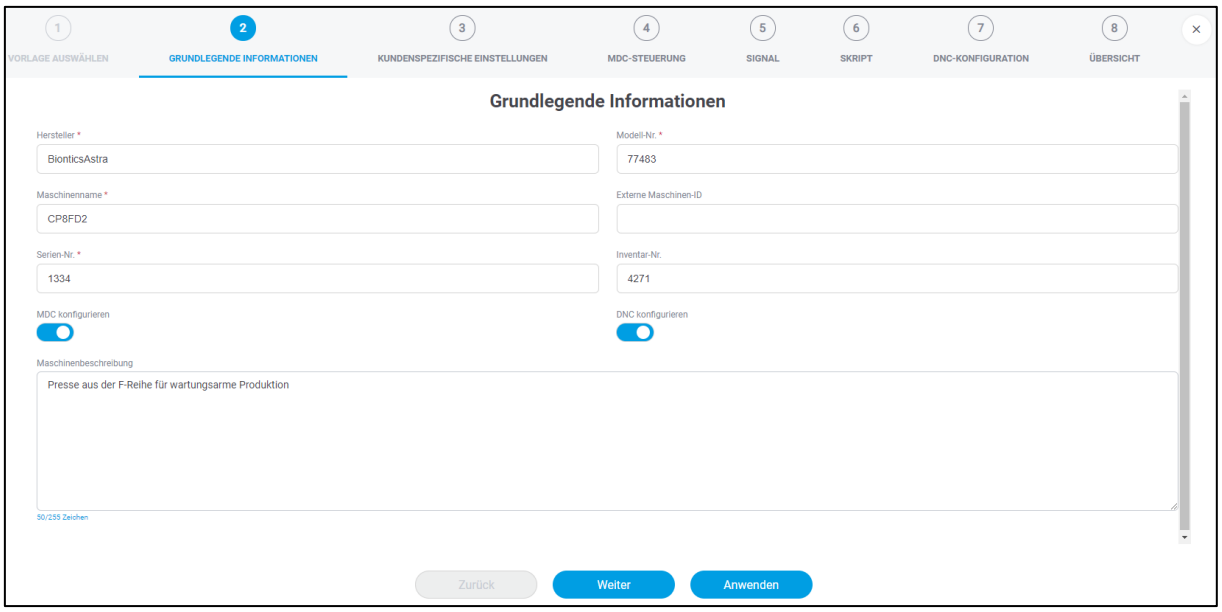


Bild 16: Maschine hinzufügen - Grundlegende Informationen

1. **Maschinenname**, **Hersteller**, **Modell-Nr**. und **Serien-Nr**. eintragen.
2. Optional: Weitere Informationen wie gewünscht eintragen.
3. Schalter **MDC konfigurieren** und/oder **DNC konfigurieren** aktivieren.
4. Auf **Weiter** klicken.

6.2.3 ③ Kundenspezifische Einstellungen

Möglichkeit zum Hinterlegen individueller, werkspezifischer Informationen an eine Maschine mit dem Zweck, Maschinendaten zusätzlich anzureichern. Diese Daten können später von der API abgerufen werden, um einem Drittsystem mehr Informationen bereitzustellen.

Beispiel: Name = Ort, Wert = Halle 2.
Hier wird den Maschinendaten ein zusätzlicher örtlicher Aspekt verliehen, um die Maschine bei einer eventuellen Störung genau zu lokalisieren.

ⓘ Dieser Schritt ist optional.

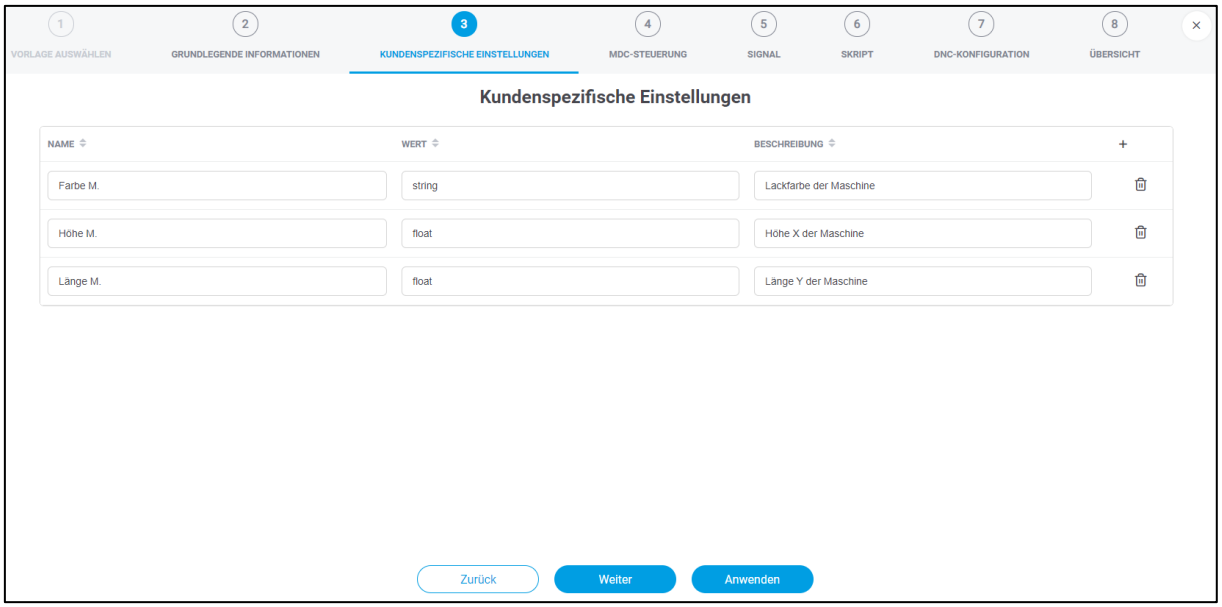


Bild 17: Maschine hinzufügen - Kundenspezifische Einstellungen

1. Auf das **+** Icon klicken.
2. Gewünschte Parameter eintragen.
3. Auf **Weiter** klicken.

6.2.4 ④ MDC-Steuerung

Möglichkeit zur Konfiguration einer MDC-Steuerung. Bestimmt die Art und Weise, wie mit der Maschine verbunden werden soll.
FORCAM unterstützt alle gängigen Steuerungen auf dem Markt und ist bemüht, die Verfügbarkeit stetig auszubauen. Eine Übersicht der aktuellen FORCAM Plug-ins ist in Abschnitt 8.4 aufgelistet.

Der Bus-Typ ist ein bestimmtes Kommunikationsprotokoll des Steuerungstyps. Bei vielen Steuerungen gibt es lediglich ein Protokoll und somit nur einen Bus-Typ zur Auswahl (z. B. Bus-Typ **FORCAM IO Box connection** für den Controller **FORCAM IO Box**). Bei der Steuerung Siemens S7 sind beispielsweise mehrere Protokolle möglich, weswegen mehrere Bus-Typen verfügbar sind.

ⓘ Dieser Schritt ist nur verfügbar, wenn in Schritt ② **MDC konfigurieren** aktiviert wurde.

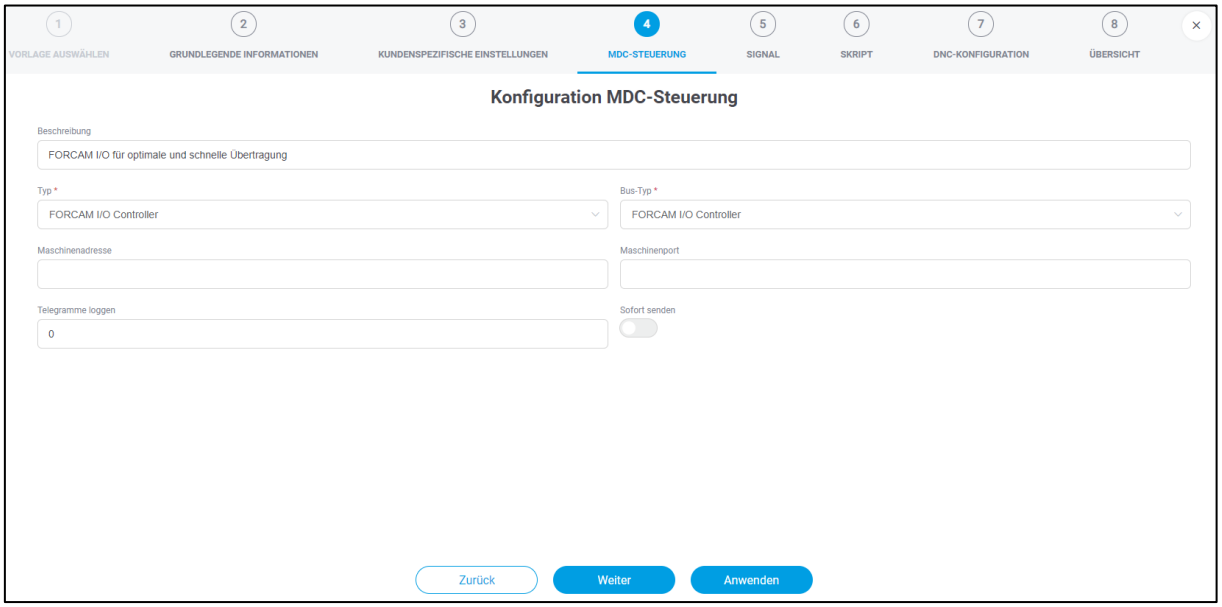


Bild 18: Maschine hinzufügen - MDC-Steuerung

1. Optional: Beschreibung der Steuerung eintragen.
2. Steuerungstyp auswählen.
   ➔ Je nach ausgewähltem Typ erscheinen zusätzliche Konfigurationsparameter.
3. Bus-Typ auswählen.
   Die Auswahlmöglichkeit richtet sich nach dem zuvor ausgewählten Steuerungstyp.
4. Steuerungsspezifische Konfiguration anpassen.
5. Auf **Weiter** klicken.

ⓘ Die Funktion **Telegramme loggen** dient zum Ausloggen der UDP-Telegramme im DCU-Log. In dem Eingabefeld wird die Anzahl der auszuloggenden Zeichen jedes Telegramms angegeben.

6.2.5 ⑤ Signal

Legt fest, welche Signale aus der Steuerung ausgelesen werden. Um beispielsweise die Interpretation „Maschine läuft“ bzw. „Maschine läuft nicht“ zuzulassen, kann ein binäres Signal konfiguriert werden. Wird 1 gemeldet, ist die Maschine in Produktion, bei 0 ist sie entsprechend im Stillstand. Das Signal heißt in diesem Beispiel „DONE“.
Darüber hinaus können unterschiedliche Datentypen verwendet werden, um diverse Signale wie z. B. Temperatur abzubilden. Es können auch Signale festgelegt werden, die auf die Steuerung geschrieben werden können, z. B. eine Auftragsnummer oder Vorgabezeiten. Mit Data Lake werden alle Daten festgehalten und gespeichert. Pro Signal kann die Data Lake Speicherung an & abgeschaltet werden. Es können Einheiten auf einzelne Signale erfasst werden (z.B. Grad Celsius oder Liter die Minute), und zudem können Skalier-Faktoren konfiguriert werden.

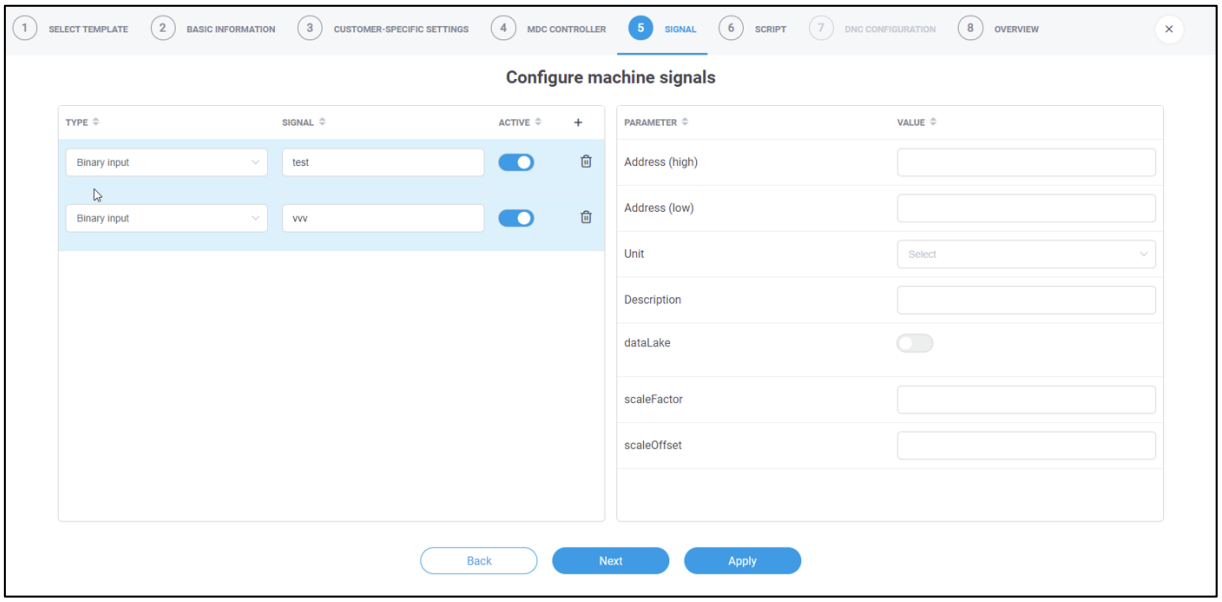


Bild 19: Maschine hinzufügen - Signal

1. Auf das **+** Icon klicken.
2. **Name**, **Typ** und **Beschreibung** eintragen.
3. Am linken Zeilenrand das Drop-down-Menü aufklappen.
   Der Name ist entsprechend dem ausgewählten Typ vorausgefüllt.
4. Gewünschten Wert eintragen.
5. Rechts pro Signal die unterschiedlichen gewünschten Parameterangaben und Werte eingeben.
6. Auf **Weiter** klicken.

6.2.6 ⑥ Skript

Legt fest, wie die Maschinensignale interpretiert werden sollen.
Je nach Steuerung können durch ein Skript unterschiedliche Signale ausgelesen werden, z. B. „Produktion im Zustand Teilautomatik“ oder „Stillstand wegen mechanischem Fehler“.
Im linken Bereich der Maske werden alle in Schritt ⑤ konfigurierten Signale aufgelistet.
Der zentrale Bereich in der Mitte ist das Eingabefeld für ein Skript, wo die tatsächliche Maschinenlogik definiert wird.

ⓘ Für eine Auflistung von Formelelementen und Operatoren in Skripten, siehe das Handbuch **Skriptsprache**.

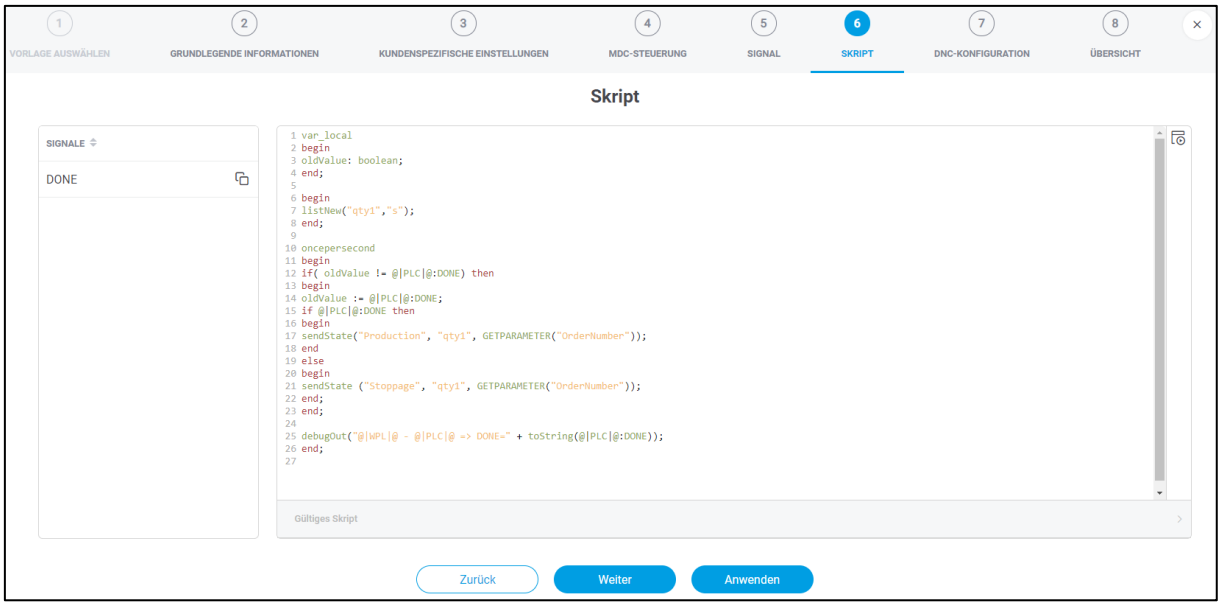


Bild 20: Maschine hinzufügen - Skript

Um dem Beispiel vom Signal „DONE“ aus Schritt ⑤ zu folgen, ist das Skript in Bild 20 folgendermaßen aufgebaut:
Kommt von der Maschine das Bit 1 für „DONE“, wird ein Event **sendStateProduction** versendet, was letztendlich dem Status **Produktion** entspricht. Ist kein Bit-Flag auf 1, wird **sendStateStoppage** versendet, also **Stillstand**. Weitere Skriptbeispiele sind in Abschnitt 8.6 verfügbar.

```
10 oncepersecond
11 begin
12 if( oldValue != @|PLC|@:DONE) then
13 begin
14 oldValue := @|PLC|@:DONE;
15 if @|PLC|@:DONE then
16 begin
17 sendState("Production", "qty1", GETPARAMETER("OrderNumber"));
18 end
19 else
20 begin
21 sendState ("Stoppage", "qty1", GETPARAMETER("OrderNumber"));
22 end;
23 end;
24
25 debugOut("@|WPL|@ - @|PLC|@ => DONE=" + toString(@|PLC|@:DONE));
26 end;
27
```

Bild 21: Auszug aus dem Skriptbeispiel für das Signal „DONE“

Um ein Skript zu konfigurieren:

1. Im linken Bereich das Signal auswählen, dem ein Skript hinterlegt werden soll.
2. Gewünschtes Skript in das zentrale Eingabefeld einfügen.
3. Optional: Skript über das Abspielen-Icon am rechten oberen Rand ausführen, um die Validität zu überprüfen.
4. Auf **Weiter** klicken.

6.2.7 ⑦ DNC-Konfiguration

Möglichkeit zur Konfiguration einer DNC-Steuerung. Bestimmt die Art und Weise, wie eine NC-Datei an die Maschine übertragen werden soll.
FORCAM unterstützt alle gängigen Steuerungen auf dem Markt und ist bemüht, die Verfügbarkeit stetig auszubauen. Eine Übersicht der aktuellen FORCAM Plug-ins ist in Abschnitt 8.4 aufgelistet.

ⓘ Dieser Schritt ist nur verfügbar, wenn in Schritt ② **DNC konfigurieren** aktiviert wurde.

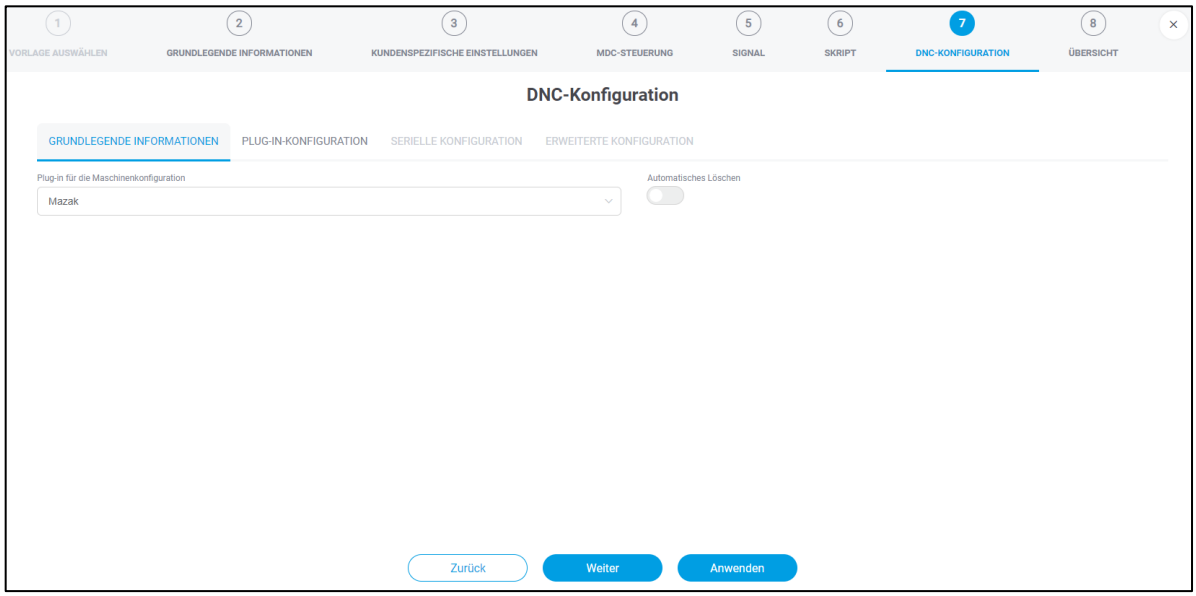


Bild 22: Maschine hinzufügen - DNC-Konfiguration

1. Optional: **Upload**- und **Download-Timeout** eintragen.
2. **Plug-in für die Maschinenkonfiguration** auswählen.

➔ Je nach ausgewähltem Plug-in erscheinen zusätzliche Konfigurationsparameter in weiteren Reitern.

3. Optional: Automatisches Löschen bestimmen.
   Ist der Schalter aktiviert, wird die NC-Datei nach dem Lesen aus der Maschine automatisch gelöscht.
4. Weitere Konfigurationsparameter abhängig vom ausgewählten Plug-in in den übrigen Reitern eintragen.
5. Auf **Weiter** klicken.

6.2.8 ⑧ Übersicht

Fasst die bisherige Konfiguration aus allen Schritten zusammen und listet alle definierten Signale auf. Nach der Bestätigung wird die Maschine mit der angegebenen Konfiguration abgebildet und somit digitalisiert. Die konfigurierte Maschine erscheint unter dem definierten Namen in der Übersicht (siehe Bild 11).

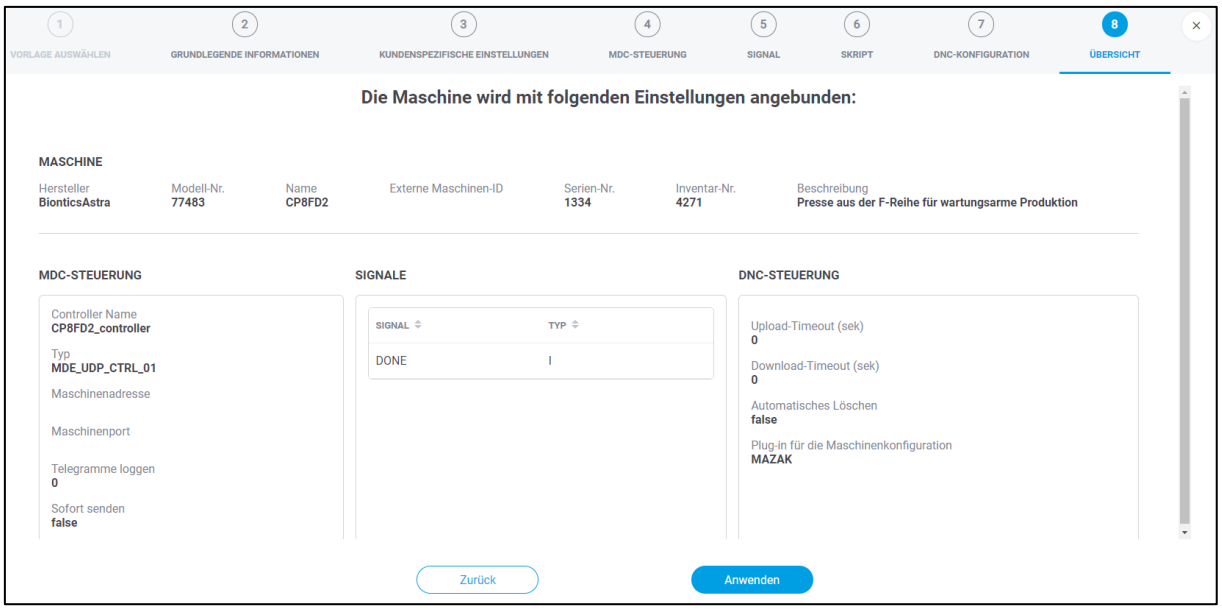


Bild 23: Maschine hinzufügen - Übersicht

6.3 Eventkonfiguration

In der Eventkonfiguration wird festgelegt, wie die Signale an ein übergeordnetes System geschickt werden. Payload und Endpunkt sind standardmäßig vordefiniert, können aber individualisiert werden.

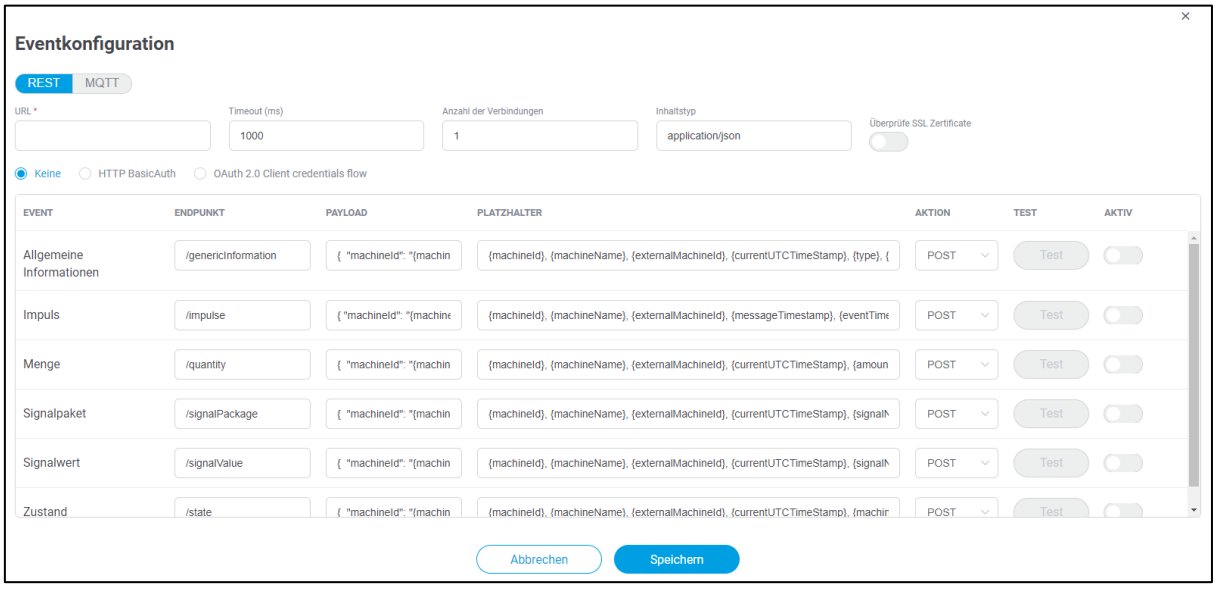


Bild 24: Eventkonfiguration in FORCAM FORCE EDGE

Events werden in einem Skript verwendet, um daraus ausgehende Ereignisse zu generieren. Hierfür stehen Skriptfunktionen zur Verfügung, die je nach Typ ein entsprechendes Event erzeugen.
Im Beispiel aus Abschnitt 6.2.6 etwa wurde die Skriptfunktion **sendStateProduction** verwendet, was den Zustand **Produktion** als Event an ein Drittsystem verschickt.
Für jeden Event-Typ gibt es ein standardisiertes **Event**. Der Event-Typ **Menge** verschickt beispielsweise die von der Maschine produzierte Menge. Alle verfügbaren Events sind in Abschnitt 6.3 aufgelistet.

Im JSON-body unter **PAYLOAD** wird festgelegt, wie die Nachricht an das übergeordnete System aussehen soll. Die Platzhalter (Wildcards) werden schließlich durch die entsprechenden, vorhandenen Signale ersetzt. Beispiel einer Event-Struktur:

```
{
        machineId: $machineId$
        machineName: $machineName$
        externalMachineId: $externalMachineId$
        reference: $reference$
        timeStamp: $currentUTCTimeStamp$
        signalName: $signalName$
        value: $value$
        unit: $unit$
}
```

Skriptfunktionen ermöglichen es Events, **Platzhalter** (Wildcards) zu verwenden, mit deren Hilfe unterschiedliche Informationen weitergegeben werden können. Hierüber kann beispielsweise die Maschinen-ID oder der Zeitstempel in UTC aufgelöst werden. Abschnitt 8.7 listet alle verfügbaren Skriptfunktionen auf und erklärt diese.

Ist der Schalter unter **AKTIV** aktiviert, wird das entsprechende Event verschickt. Nicht aktivierte Events werden nicht verschickt.
Ein aktives Event kann durch Klicken auf **TEST** auch verprobt werden. Im Folgedialog können Werte wie **machineId** (Maschinen-ID) oder **value** (Wert) eingetragen werden, um das Signal exemplarisch zu generieren und auszuführen, ohne einen Einfluss auf die tatsächliche Maschinenanbindung zu haben. Auf diese Weise können Events vorab getestet werden, ohne sie in der Live-Umgebung ausführen zu müssen.

6.3.1 Signale & Events von Edge zum übergeordneten System

Für die Versorgung von Signalen und Events von einem Edge-Knoten zu einer 3rd-Party-Applikation gibt es zwei technische Möglichkeiten.

ⓘ Die Versorgung kann jeweils im Edge-Knoten selbst konfiguriert werden.

HTTP/REST

Zur Versorgung des Fremdsystems kann ein beliebiger dort bereitgestellter REST-Endpunkt bedient werden. Dabei werden die HTTP Methoden POST und PUT unterstützt.
Als HTTP-Authentifizierungsmethoden sind folgende Standards implementiert:

- Basic Authentification: Authentifizierung nach RFC 2617 durch Eingabe von Benutzername und Passwort (siehe https://datatracker.ietf.org/doc/html/rfc2617).
- Client credential flow: Authentifizierung nach OAuth 2.0 RFC 6749 über dem System bekannte Client ID und Client Secret. (siehe https://auth0.com/docs/flows/client-credentials-flow). Diese Art der Authentifizierung erfolgt ohne Benutzereingriff, d.h. im Hintergrund.

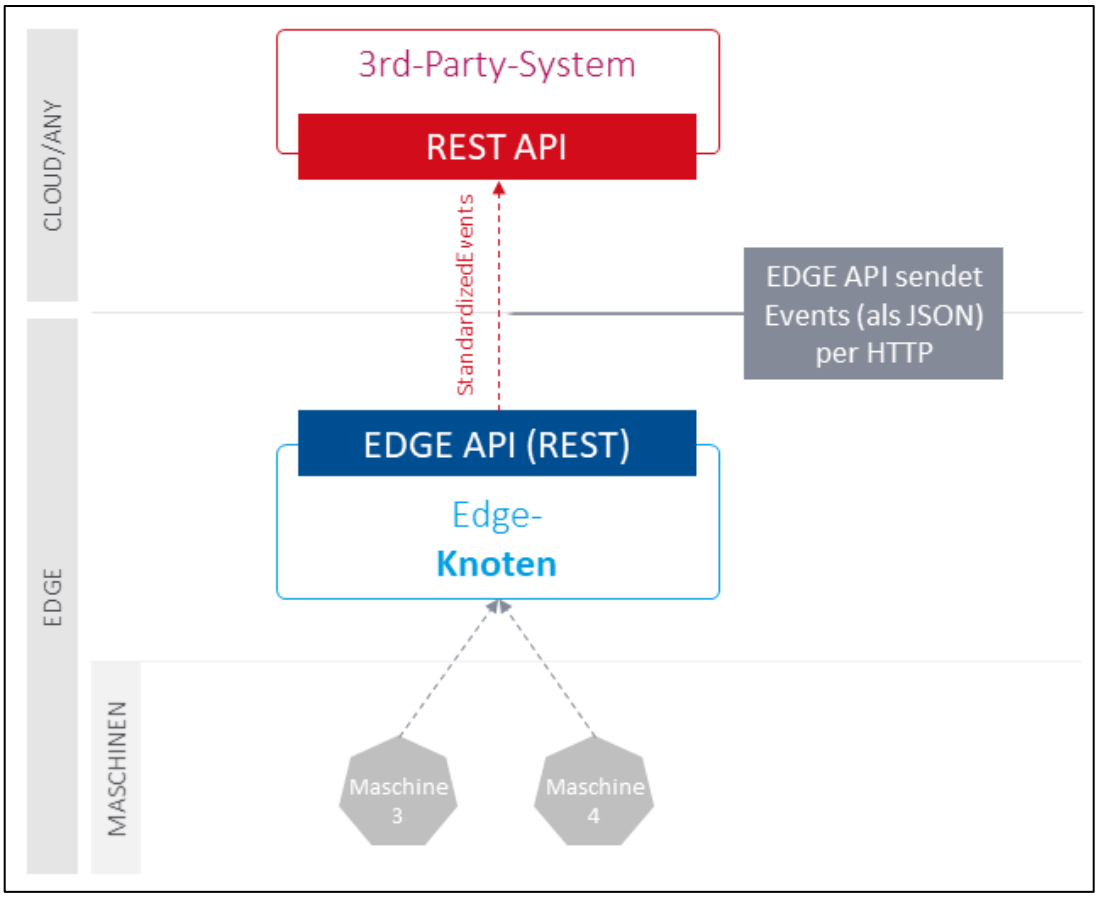


Bild 25: Kommunikation mit übergeordneten Systemen über HTTP/REST

MQTT-Messaging

Zur Versorgung kann auch ein beliebiger MQTT-Broker bedient werden, sofern durch Kunden oder Partner bereitgestellt.

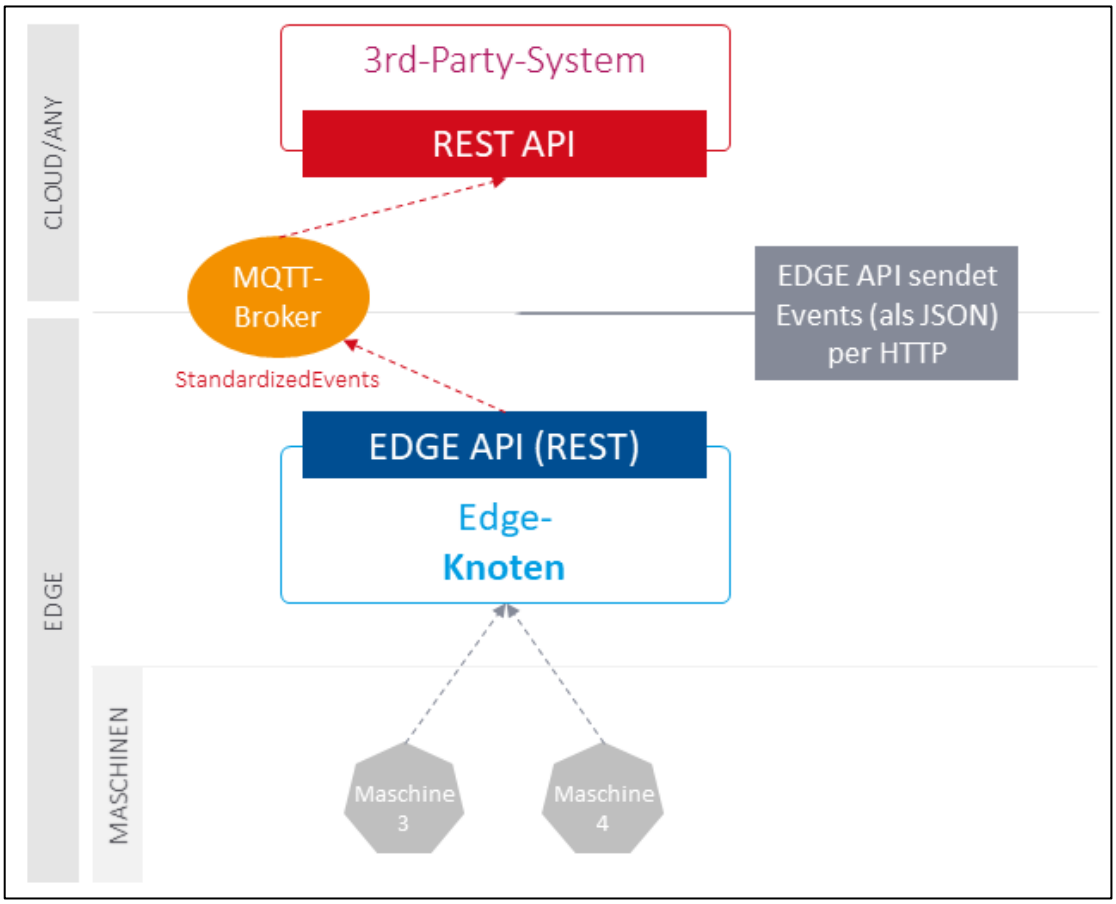


Bild 26: Kommunikation mit übergeordneten Systemen über MQTT-Broker

6.3.2 Daten & Dokumente vom übergeordneten System zu Edge

Über die EDGE API kann FORCAM FORCE EDGE mit Daten und Dokumenten versorgt werden.
Technisch ist dies ausschließlich über **HTTP/REST** möglich.
Folgende Schnittstellen werden bereitgestellt:

Tabelle 2: Schnittstellen für die Übertragung von Daten und Dokumenten

Schnittstelle	Beschreibung
Übertragen von Prozess- und Referenzparametern	Diese Business-Parameter können im EDGE Composition Layer verwendet werden, um damit unter anderem standardisierte Events anzureichern (z. B. Auftragsnummer oder Taktzeit).
Übertragung von Signalparametern	Es können Parameterwerte für spezifische Signale übertragen werden. Diese werden direkt auf die Maschinensteuerung geschrieben.
Übertragung von Dokumenten	Es können NC-Programme übertragen werden, welche ebenfalls auf die Maschinensteuerung übertragen werden.

6.3.3 Event konfigurieren

1. In der Detailansicht einer konfigurierten Maschine (siehe Bild 11) im rechten oberen Bereich auf das Eventkonfiguration-Icon klicken.
2. Im Folgedialog bestimmen, ob **REST** oder **MQTT** verwendet werden soll.
3. In der oberen Leiste **URL** und optional weitere Angaben wie Timeout etc. eintragen. Bestimmt, wohin die Events geschickt werden sollen.
4. SSL-Zertifikatsprüfung bestimmen.
   Ist der Schalter **Überprüfe SSL-Zertifikat** aktiv, kann FORCAM FORCE EDGE über REST auch mit einer Anwendung ohne gültiges bzw. unsigniertes Sicherheitszertifikat verbunden werden.
5. Gewünschte Authentifizierung auswählen und Anmeldedaten eintragen.
6. Events wie gewünscht konfigurieren.
7. Speichern.

7 Monitoring

FORCAM FORCE EDGE hat die Option integriert, einzelne Komponenten über die Monitoring-Seite zu überwachen. Die Seite gibt Aufschluss darüber, ob die jeweilige Komponente fehlerfrei läuft, oder ob Störungen vorliegen.
Das Monitoring ist über das Icon im rechten oberen Bereich in der Maschinenübersicht aufrufbar (siehe Bild 11).

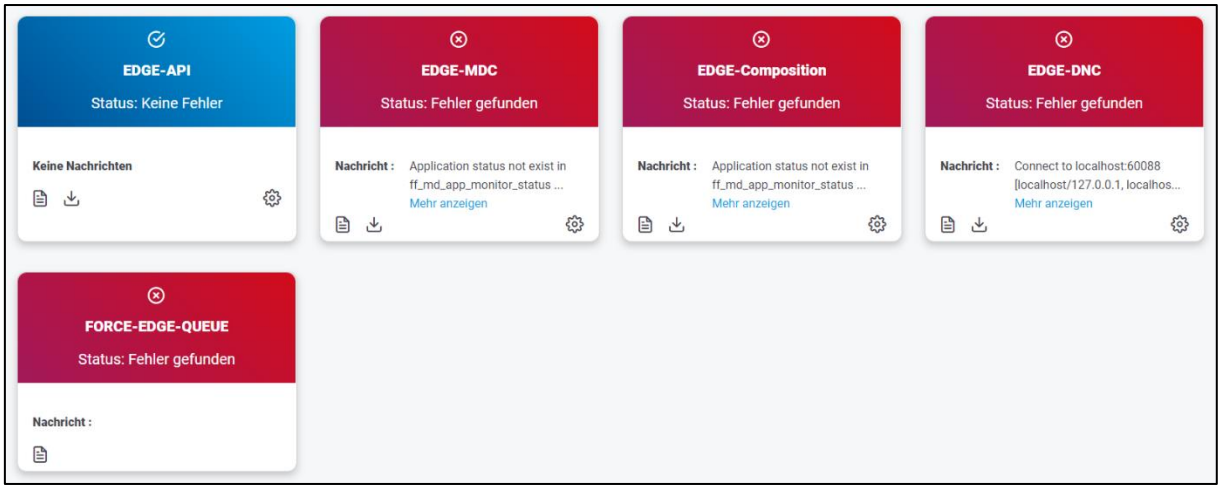


Bild 27: Monitoring in FORCAM FORCE EDGE

In jeder Komponente können Fehlermeldungen und Logs gezielt abgerufen werden.

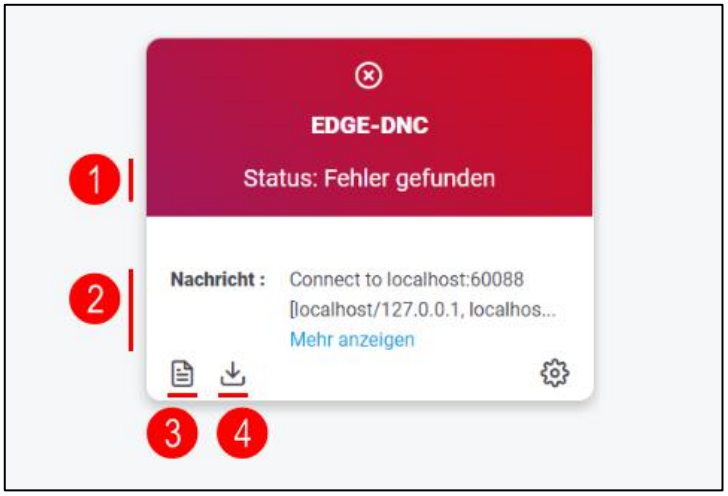


Bild 28: Komponente „EDGE-DNC“ in der Monitoring-Seite

(1) Aktueller Status der Komponente
(2) Nachricht bei einem Fehler.
Durch Klicken auf **mehr…** wird die vollständige Fehlermeldung in einem Pop-up angezeigt.
(3) Zeigt die jeweils letzte Warn- und Fehlermeldung der Komponente an
(4) Erlaubt das Herunterladen einer Logdatei von einem bestimmten Tag

8 Anhang

8.1 Änderungstabelle

Tabelle 3: Änderungen in Release-Version 210809 in Dok.-Version 2

Datum	Edge-Version	Dok.-Version	Änderung
11.08.21	210809	2	Landing Page (Bild 3) ausgetauscht
11.08.21	210809	2	Liste unterstützter Plug-ins in 8.4 aktualisiert: Für MDC Weihenstephan und WUT hinzugefügt, für DNC Heidenhain, WUT und COM hinzugefügt und Soflex und Comportserver entfernt
11.08.21	210809	2	Hinweis in 6.2.6 zum Handbuch Skriptsprache hinzugefügt
11.08.21	210809	2	Skript 4 in 8.6 als Beispiel für Signal Packages hinzugefügt
18.08.21	210514	2	Kapitel 5 Grundlegende Einstellungen hinzugefügt
19.08.21	210514	2	Pflichtangaben bei Schritt 1 in 6.2.2 angepasst
19.08.21	210514	2	Hinweis in 6.2 hinzugefügt, dass negative Werte nicht erlaubt sind
24.08.21	210622	2	Tipp in 6.2 um den Satz ergänzt, dass Konfigurationsseiten beim Editieren direkt aufgerufen werden können
25.08.21	210717	2	FORCAM FORCE EDGE ist nun auf Französisch verfügbar.
25.08.21	210717	2	Schritt über URL und optionale Angaben bei 6.3.3 auf Position 3 verschoben. Schritt über SSL-Zertifikatsprüfung in Position 4 hinzugefügt. Schritt über Authentifizierung auf Position 5 verschoben
26.08.21	210802	2	Hinweis in 6.2.4 zum Loggen von Telegrammen hinzugefügt
26.08.21	210802	2	Hinweis in 5.1 über das Anlegen eines Benutzers mit denselben Daten hinzugefügt
26.08.21	210802	2	Hinweis in 6.2.1 über das Überschreiben der Konfiguration nach Auswahl eines Templates hinzugefügt. Zusätzlich Screenshot hinzugefügt
27.08.21	210809	2	Node-RED in 8.4 als unterstütztes Plug-in (MDC) hinzugefügt
30.08.21	210825	2	Kapitel 4 Deployment hinzugefügt
01.09.21	210825	2	Screenshots unter 6 aktualisiert
01.09.21	210825	2	Skript-Beispiele in 8.6 in einzelne Unterkapitel unterteilt
14.09.21	210825	2	Screenshots unter 7 aktualisiert
25.10.21	210825	3	Informationen zu Data Lake hinzugefügt. Kapitel 2.3, 3, und 6.2.5

8.2 Dokument-Konventionen

Tabelle 4: Verwendete Schriftarten, Formatierungen und Zeichen

Konvention	Beschreibung
Fettschrift	Die Bezeichnungen von Schaltflächen und Optionen sind fettgeschrieben.
Kursivschrift	Hervorgehobene Wörter sind kursivgeschrieben.
Icons	Bei einer Funktion, die über ein Icon dargestellt ist, wird auf das Icon als Objekt referiert.
Handlungsresultat	Handlungsresultate sind durch ➔ gekennzeichnet.
Voraussetzungen	Voraussetzungen sind durch ✓ gekennzeichnet.
Warnungen	Warnungen sind durch ⚠ gekennzeichnet.
Hinweis	Hinweise sind durch ⓘ gekennzeichnet.
Tipps	Tipps sind durch ⓣ gekennzeichnet.

8.3 Abkürzungen und Begriffe

Tabelle 5: Verwendete Abkürzungen und Begriffe

Abkürzung	Erklärung
Brownfield	Eine Fabrik oder Fertigungsanlage, die bereits gebaut und schon seit einiger Zeit in Betrieb ist. Der Brownfield-Ansatz in Zusammenhang mit der Industrie 4.0 meint die digitale Transformation einer vorhandenen Fertigungsanlage.
CP	Communication Processor
DNC	Distributed Numerical Control: NC-Anlagen, die mit einem Computer verbunden sind. Die Einzelanlagen können zentral mit NC-Programmen versorgt und koordiniert werden.
IT	Informationstechnik
Maschine	In FORCAM FORCE EDGE ist die Maschine eine Teilanlage nach ISA 95. Sind keine weiteren Teilanlagen (d.h. zusätzliche physikalische Steuerungen) vorhanden, sprechen wir von einer Anlage.
MDC	Machine Data Connection (Maschinendatenerfassung)
MQTT	Message Queuing Telemetry Transport: offenes Netzwerkprotokoll für Machine-to-Machine-Kommunikation (M2M), das die Übertragung von Telemetriedaten in Form von Nachrichten zwischen Geräten ermöglicht, trotz hoher Verzögerungen oder beschränkter Netzwerke
MR	Machine Repository
Northbound	Eine northbound-Schnittstelle kommuniziert in einer bestimmten Netzwerk-Komponente mit einem höher eingestuften Element.
OT	Operative Technologie
POST	POST ist eine Methode, die von HTTP unterstützt wird und darstellt, dass ein Webserver die im Body der Nachricht enthaltenen Daten annimmt, die angefordert werden.
PUT	Die PUT-Methode wird verwendet, um eine auf dem Server verfügbare Ressource zu aktualisieren. Typischerweise ersetzt sie alles, was an der Ziel-URL existiert, durch etwas anderes.
REST	Representational State Transfer: Programmierparadigma für verteilte Systeme (Zusammenschluss unabhängiger Computer, die sich für den Benutzer als ein einziges System präsentieren)
RESTful API	API für den Datenaustausch auf Basis von HTTP-Anfragen mittels GET, PUT, POST und DELETE, der den Anforderungen bzw. Beschränkungen der REST Architektur unterliegt.
Signal	Aus der Maschinensteuerung ausgelesene Werte wie z. B. Temperatur, Druck oder bestimmte Status.
Southbound	Äquivalent zur northbound-Schnittstelle kommuniziert eine southbound-Schnittstelle mit darunterliegenden Komponenten.
SPS	Speicherprogrammierbare Steuerung
UTC	Coordinated Universal Time (koordinierte Weltzeit)
Wildcard	Platzhalter für andere Zeichen

8.4 Liste unterstützter Plug-ins

MDC-Plug-ins

Tabelle 6: Liste aller unterstützter Maschinenanbindungsvarianten

Name	Lesen	Schreiben	Übertragungsart Polling/Eventbasiert
AUDI SPS	X	X	X/
CSV File Exchange	X		X/
Euromap 63	X		X/
Euromap 77 (via OPC UA)	X	X	/X
FANUC	X	X	X/
FORCAM I/O Controller	X	X	/X
FORCAM I/O Controller (Hardware)	X		
Heidenhain	X	X	X/
MAKINO Pro 3/Pro 6	X		
Mazak	X		
MCIS RPC (SINUMERIK 810D/840D/840D)	X		X/X
Modbus	X		
MQTT	X	X	/X
MT Connect	X		X/
Node-RED	X	X	/X
OKUMA	X		
Omron	X		
OPC Classic	X	X	X/
OPC UA	X	X	/X
OPC XML	X		X/
Rockwell/Allen Bradley	X	X	X/

Name	Lesen	Schreiben	Übertragungsart Polling/Eventbasiert
Siemens S5 mit CP	X		
Siemens S5 ohne CP	X		
Siemens S7 mit CP	X	X	X/
Siemens S7 ohne CP	X	X	X/
SQL Database Exchange	X		X/
Weihenstephan	X		X/
Wiesemann & Theis (WUT)	X		X/

DNC-Plug-ins

Tabelle 7: Liste aller unterstützter NC-Maschinen-Anbindungsvarianten

Name	Lesen	Schreiben
COM	X	X
Heidenhain	X	X
Mazak-DNC	X	X
RPC Plug-in	X	X
FTP Plug-in	X	X
FANUC	X	X
File Handler (File Copy)	X	X
File Handler Server	X	X
MOXA-Box	X	X
Wiesemann & Theis (WUT)		

8.5 Standardisierte Events

Tabelle 8: Events und deren Funktion in FORCAM FORCE EDGE

Event-Typ	Werte	Funktion
Allgemeine Information	– Machine Id – Machine Name – External Machine Id – Reference (any) – Timestamp – Type (any) – Value (any)	Beliebige Information
Impuls	– Machine Id – Machine Name – External Machine Id – Reference (any) – Timestamp – Count	Z. B. Hub, Schuss etc.
Menge	– Machine Id – Machine Name – External Machine Id – Reference (any) – Timestamp – Amount – Unit (optional) – QualityDetail (optional)	Produzierte Menge
Signalpaket	– Machine Id – Machine Name – External Machine Id – Reference (any) – Timestamp – ARRAY [SignalName, Value, TimeStampUTC, Unit (optional)]	Sammlung von Signalen (z. B. Seriennummer, Druck und Temperatur)
Signalwert	– Machine Id – Machine Name – External Machine Id – Reference (any) – Timestamp – SignalName – Value – Unit (optional)	Temperatur, Druck etc.

Event-Typ	Werte	Funktion
Zustand	– Machine Id – Machine Name – External Machine Id – Reference (any) – Timestamp – State (Production or Downtime) – StatusCodes (optional list of statuses)	Zustand der Maschine (läuft, läuft nicht)

8.6 Skript-Beispiele

8.6.1 Maschinenstatus und Temperatur

Durch das folgende Skript wird der Status der Maschine versendet (Produktion oder Stillstand). Zusätzlich wird die Temperatur ausgegeben. Sobald sich die Temperatur ändert, wird die aktualisierte Temperatur versendet.

```
var_local

begin
        oldState: boolean;
        oldTemperature: string;
end;

oncepersecond
begin

        if( oldState!= @|PLC|@:DONE) then
        begin
                oldState := @|PLC|@:DONE;
                if @|PLC|@:DONE then
                begin
                        sendStateProduction()
                end
                else
                begin
                        sendStateStoppage();
                end;
        end;

        if( oldTemperature != toString(@|PLC|@:TEMP)) then
        begin
                oldTemperature := toString(@|PLC|@:TEMP);
                sendSignalValue("TEMPERATURE", toString(@|PLC|@:TEMP), "Degrees");
        end;

end;
```

8.6.2 Temperatur und Luftfeuchtigkeit

Durch das folgende Skript werden die aktuelle Temperatur und Luftfeuchtigkeit versendet. Dies passiert im Intervall von 30 Sekunden und sobald eine Änderung dieser Werte stattfindet.

```
var_local
begin
        oldTemperature : string;
        oldHumidity : string;
        seconds: number;
end;

oncepersecond
begin

        seconds := seconds + 1;

        if (seconds > 30) then
        begin
                seconds := 0;
                oldTemperature := "";
                oldHumidity := "";
        end;

        if (oldTemperature != @|PLC|@:TEMP ) then
        begin
                oldTemperature := @|PLC|@:TEMP;
                sendSignalValue("TEMP", toString(@|PLC|@:TEMP), "Degree");
        end;

        if (oldHumidity != @|PLC|@:HUMIDITY ) then
        begin
                oldHumidity := @|PLC|@:HUMIDITY;
                sendSignalValue("HUMIDITY", toString(@|PLC|@:HUMIDITY), "Degree");
        end;
end;
```

8.6.3 Kransteuerung

Dieses Skript erfasst eine Kransteuerung mit den Knöpfen Schwarz, Grün und Rot.

- Der schwarze Knopf schaltet die Maschine an und aus
- Der rote Knopf löst einen Notfall aus
- Der grüne Knopf sendet einen Impuls für Stückzahlen und zählt diesen anschließend hoch

```
var_local
begin
        // GENERAL LOGIC VARIABLES
        seconds:  number;
        // MACHINE STATE
        state: number;
        stateOld: number;
        // MACHINE STATUS REASON
        status_reason: string;
        status_reasonOld: string;
        // PIECE COUNT VARIABLES
        counter: number;
        counterOld: number;
        counterSend: number;
end;

begin
        //DEFINE LISTS START
        ListNew("STATUSCODES", "S");
        //DEFINE LISTS END
end;

begin
        // INITIALIZE SCRIPT VARIABLES START
        if not initialized and not offline(@|PLC|@) then
        begin
                status_reason := " ";
                status_reasonOld := " ";
                counter := @|PLC|@:Good_count;
                counterOld:= counter;
                ListClear("STATUSCODES");
                // Set initialized to perform initializing once
                initialized := true;
        end
        else if initialized then
        begin
                counter := @|PLC|@:Good_count;
                ListClear("STATUSCODES");
        end;
        // INITIALIZE SCRIPT VARIABLES END

        // ACTIONS ONCE PER SECOND START
        oncePerSecond
        begin
                seconds := seconds + 1;
        end;
        // ACTIONS ONCE PER SECOND END

        // DEFINITION STATE / STATUS_REASON START
        if offline(@|PLC|@) then
        begin
                state := "1";
                status_reason :='NOT_CONNECTED';
                seconds := 0;
        end
        else if not @|PLC|@:Emergency_ON then
        begin
                state := "1";
                status_reason :='EMERGENCY_ON';
                seconds := 0;
        end
        else if not @|PLC|@:Machine_ON then
```

```
begin
        state := "2";
        status_reason := 'PRODUCTION'
        seconds := 0;
end
else
begin
        if seconds > karenzZeit then
        begin
                state := "1";
                status_reason := 'UNDEFINED_STOPPAGE'
                seconds := 0;
        end
end;
// DEFINITION state END

// DEFINITION COUNTER START
if counter >= counterOld then                   // Part counter on PLC is incremented
begin
        counterSend := counter - counterOld;
        counterOld := counter;
end
else if counter < counterOld then               // Part counter on PLC changes to negative
begin
        counterSend := 32768 - counterOld;
        counterOld := counter;
end;
// DEFINITION COUNTER END

// SEND state status_reason START
if state <> stateOld or status_reason <> status_reasonOld then
begin
        if state == 2 then
        begin
                ListAdd("STATUSCODES", status_reason);
                sendStateProduction("STATUSCODES");
        end
        else
        begin
                if == 1 then
                begin
                        ListAdd("STATUSCODES", status_reason);
                        sendStateStoppage("STATUSCODES");
                end
        end;
        debugOut("@|PLC|@" + "Send state: " + state);
        stateOld := state;
        status_reasonOld := status_reason;
end;
// SEND state status_reason END


// SEND STROKES / QUANTITY START
if counterSend > 0 and packetNo <> packetNoOld then
begin
        debugOut("@|PLC|@" + "Send quantity: " + toString(counterSend));
        SendQuantity(counterSend);
        counterSend := 0;
end;
// SEND STROKES / QUANTITY END

// LOGGING SIGNALS WHEN CHANGED START
logstring := "@|PLC|@ Signals: " + " offline: "             + toString(offline(@|PLC|@))
                                 + " State: "               + toString(state)
                                 + " Status Reason: "       + toString(status_reason)
                                 + " Machine_ON: "          + tostring(@|PLC|@:Machine_ON)
                                 + " Emergency_ON: "        + tostring(@|PLC|@:Emergency_ON )
                                 + " COUNTER: "             + tostring(@|PLC|@:Good_count)
                                 + " seconds: "             + toString(seconds);

if logString <> logstringOld then
begin
        debugOut(logString);
```

```
                logstringOld := logString;
        end;
        // LOGGING SIGNALS WHEN CHANGED END
end;
```

8.6.4 Signalpakete

Das folgende Skript ist ein Beispiel für Signal Packages (Signalpakete):

```
//
// Task: Send machine state / status_reason / quantities to runtime
// Created: 2021-05-12
// Version: 1.0
// Author:  FORCAM MDC
//
// ---------------------------------------------------------------
//
// Incoming signals
// Reg1 = holding register  1
//
//
// Outgoing information
// //state         = machine state
// //STATUSCODES   = Status reason
// Reg1SEND   = just display holding register
////-------------------------------------------------------------

// VARIABLES
var_local
begin
// GENERAL  VARIABLES
    seconds: number;
    logstring: string;
    logstringOld: string;
//  SIGNAL VARIABLES
    H1Old: number;
    H2Old: number;
    H3Old: number;
    H4Old: number;
    H5Old: number;
    H6Old: number;
    H7Old: number;
    H8Old: number;
    H9Old: number;
    H10Old: number;
 // SCRIPT INIZIALIZING VARIABLES
    initialized: boolean;
end;

begin
     if not initialized and not offline(@|PLC|@) then
    begin
    //DEFINE LISTS START (S=strin B=boolean N=number)
        ListNew("Signals", "S");
        ListNew("Values", "S");
//        ListNew("Timestamps", "S");
    //DEFINE LISTS END
    end;

// INITIALIZE SCRIPT & VARIABLES START
    if not initialized and not offline(@|PLC|@) then
    begin
          H1Old := 0;
          H2Old := 0;
          H3Old := 0;
          H4Old := 0;
          H5Old := 0;
          H6Old := 0;
          H7Old := 0;
```

```
          H8Old := 0;
          H9Old := 0;
          H10Old := 0;
          ListClear("Signals");
          ListClear("Values");
 //         ListClear("Timestamps");
//    set initialized to perform initializing once
      initialized := true;
    end
    else if initialized then


// ACTIONS ONCE PER SECOND START
   oncePerSecond
   begin
     seconds:= seconds + 1;

// ACTIONS ONCE PER SECOND END
// send one package for all 10 holding registers  for now always
// Reg1Content Start
    if( H1Old <> @|PLC|@:H1 ) then
   begin

 //fill lists
      H1Old := @|PLC|@:H1;
       ListAdd("Signals", "H1");
      ListAdd("Values", toString(@|PLC|@:H1));
      H2Old := @|PLC|@:H2;
      ListAdd("Signals", "H2");
      ListAdd("Values", toString(@|PLC|@:H2));
      H3Old := @|PLC|@:H3;
      ListAdd("Signals", "H3");
      ListAdd("Values", toString(@|PLC|@:H3));
      H4Old := @|PLC|@:H4;
      ListAdd("Signals", "H4");
      ListAdd("Values", toString(@|PLC|@:H4));
 //     H5Old := @|PLC|@:H5;
 //     ListAdd("Signals", "H5");
 //     ListAdd("Values", toString(@|PLC|@:H5));
 //     H6Old := @|PLC|@:Reg6;
 //     ListAdd("Signals", "H6");
 //     ListAdd("Values", toString(@|PLC|@:H6));
 //     H7Old := @|PLC|@:H7;
 //     ListAdd("Signals", "H7");
 //     ListAdd("Values", toString(@|PLC|@:H7));
 //     H8Old := @|PLC|@:H8;
 //     ListAdd("Signals", "H8");
 //     ListAdd("Values", toString(@|PLC|@:Reg8));
 //     H9Old := @|PLC|@:H9;
 //     ListAdd("Signals", "H9");
 //     ListAdd("Values", toString(@|PLC|@:Reg9));
 //   H10Old := @|PLC|@:H10;
 //    ListAdd("Signals", "H10");
 //     ListAdd("Values", toString(@|PLC|@:H10));
//      sendSignalValue("HoldingReg1", toString(@|PLC|@:H1));
 //     sendSignalValue("HoldingReg2", toString(@|PLC|@:H2));
 //     sendSignalValue("HoldingReg3", toString(@|PLC|@:H3));
//      sendSignalValue("HoldingReg4", toString(@|PLC|@:H4));
 //     sendSignalValue("HoldingReg10", toString(@|PLC|@:H10));
//  send Signal Package with lists
                                                SendSignalPackage("Signals", "Values")
       //initialize list
      begin
         ListClear("Signals");
         ListClear("Values");
      end;

// SENDING Holding Register  END

// LOGGING SIGNALS WHEN CHANGED START
    logstring := "@|PLC|@ Signals: "
                                        + " Reg 1: "                + tostring(@|PLC|@:H1)
                                        + " Reg 2: "                + tostring(@|PLC|@:H2)
```

```
                                    + " Reg 3: "                + tostring(@|PLC|@:H3)
                                    + " Reg 4: "                + tostring(@|PLC|@:H4)
   //                                 + " Reg 5: "                + tostring(@|PLC|@:H5)
   //                                 + " Reg 6: "                + tostring(@|PLC|@:H6)
   //                                 + " Reg 7: "                + tostring(@|PLC|@:H7)
   //                                 + " Reg 8: "                + tostring(@|PLC|@:H8)
   //                                 + " Reg 9: "                + tostring(@|PLC|@:H9)
   //                                 + " Reg 10: "               + tostring(@|PLC|@:H10)
                                                                       ;
    if logString <> logstringOld then
    begin
      debugOut(logString);
      logstringOld := logString;
    end;
// LOGGING SIGNALS WHEN CHANGED END
end;
end;
end;
```

8.7 Skriptfunktionen

Verwendung	Skriptfunktion Parameter in [..] sind optional	Beschreibung	Output-Event
Standard	SendImpulse(ImpulseCount, [Reference])	Sendet Impulse.	Impulse
Standard	SendQuantity(Quantity, [Unit], [QualityDetail], [Reference])	Sendet Menge.	Quantity
Custom	SendState(State, [StatusCodesListName], [Reference])	Sendet Status.	State
Standard	SendStateProduction([StatusCodesListName], [Reference])	Sendet Produktionsstatus.	State
Standard	SendStateStoppage([StatusCodesListName], [Reference])	Sendet den Zustand Stopp.	State
Standard	SendSignalValue(SignalName, Value, [Unit], [Reference], [CustomerSpecificSetting], [Timestamp])	Sendet den Wert eines Signals. Liste für Timestamp muss mit dem Datentyp "Long" (L) angelegt werden	SignalValue
Standard	SendSignalPackage(SignalNamesListName, ValuesListName, [UnitsListName], [Reference], [CustomerSpecificSetting], [TimestampsListName])	Sendet Signalwerte als Paket. Liste für Timestamp muss mit dem Datentyp "Long" (L) angelegt werden	SignalPackage
Custom	SendGenericInformation(ParamName, ParamValue, [Reference])	Sendet generische Informationen.	GenericInformation
Helfer	ListNew(ListName, DataType)	Erstellt eine neue Liste mit dem Namen ListName und Listenelementen vom Datentyp DataType (S für String, B für Boolean, N für Number).	-
Helfer	ListAdd(ListName, Value)	Fügt der Liste ein Element hinzu.	-
Helfer	ListClear(ListName)	Leert die Liste.	-
Helfer	ListDelete(ListName)	Löscht die Liste.	-

Verwendung	Skriptfunktion Parameter in [..] sind optional	Beschreibung	Output-Event
Helfer	GetMachineStatus()	Gibt den Maschinenstatus an.	-
Helfer	GetMachineData(ParameterName)	Gibt Maschinendaten für den angegebenen Parameter an.	-
Helfer	SetParameter(ParameterName, ParameterValue)	Setzt einen neuen Wert für den angegebenen Parameter.	-
Helfer	GetParameter(ParameterName)	Ruft den Wert für den angegebenen Parameter ab.	-
Helfer	DeleteParameter(ParameterName)	Löscht den Parameter.	-
Helfer	DeleteAllParameters()	Löscht alle Parameter.	-
Helfer	OFFLINE	Merker, ob der Controller offline ist oder nicht.	-
Helfer	CONTROLLERERROR1 / 2	Fehlernummer 1 und 2 für den Controller.	-
Helfer	IPADDRESS	Die IP-Adresse des DACQ.	-
Helfer	HOSTNAME	Hostname des DACQ.	-
Helfer	OFFLINESTRING	Controller-Offline-Fehlerstring.	-
Helfer	SQRT(args)	Wurzelfunktion MATH.	-
Helfer	SIN(args)	Sinusfunktion MATH.	-
Helfer	COS(args)	Kosinusfunktion MATH.	-
Helfer	TAN(args)	Tangensfunktion MATH	-

Verwendung	Skriptfunktion Parameter in [..] sind optional	Beschreibung	Output-Event
Helfer	S5TIMETONUMBER(args)	Siemens S5 Zeit (Integer) zu Zahl (Integer).	-
Helfer	NUMBERTOS5TIME(args)	Zahl (Integer) zu Siemens S5 Zeit (Integer).	-
Helfer	RISINGEDGE(args)	Prüft, ob der zuletzt geprüfte Wert falsch war und ob dieser wahr ist.	-
Helfer	FALLINGEDGE(args)	Prüft, ob der zuletzt geprüfte Wert wahr war und ob dieser falsch ist.	-
Helfer	SUBSTRING(str, startIndex[, endIndex])	Sub-string der angegebenen Zeichenkette.	-
Helfer	TONUMBER(str)	String zu Zahl (doppelt), ersetzt Komma zu Punkt im String.	-
Helfer	TOSTRING(str or number[, formatSpecifier])	Formatangabe der Formularbreite. Für leere Zeichenketten wird die Standardformatierung verwendet. Breite ist die Mindestlänge der Ergebniszeichenfolge. Präzision ist die Anzahl der Dezimalstellen. Wenn nicht angegeben, wird 0 verwendet. Wenn die Formatangabe mit 0 beginnt, werden der Ergebniszeichenfolge aufgefüllte Nullen vorangestellt. Wenn die Formatangabe mit X endet, wird die Zahl in hexadezimal umgewandelt, und zwar mit Groß- oder Kleinbuchstaben mit großem oder kleinem x. In diesem Fall werden die Dezimalstellen immer abgeschnitten.	-
Helfer	LENGTH(obj)	Die Länge eines Objekts als String-Wert.	-

Verwendung	Skriptfunktion Parameter in [..] sind optional	Beschreibung	Output-Event
Helfer	FORMATTIME(timeformatStr, timeOffset, [, timeunit])	Formatiert die aktuelle Zeit mit der Zeiteinheit als eines der folgenden: MILLISEKUNDE SEKUNDE MINUTE STUNDE TAG MONAT JAHR MSABSOLUTE (aktuelle Zeit) "R" bei Format wird als Zahl in Millisekunden angegeben, ansonsten wird das Format verwendet und Offset und Zeiteinheit zum Berechnen der Zeit verwendet.	-
Helfer	STDLOG(ignored, logLevel, suffixNumber, logText)	Der erste Parameter wird ignoriert. Die Log-Ebene sollte W = Warnung, C oder F = Fehler und alles andere für die Debug-Ebene sein. Die Suffix-Nummer, falls nicht 0, wird bei Skript-Logger als "(<SuffixNummer>)" am Ende des Log-Textes angehängt.	-
Helfer	FILELOG(filepath, textToAppend)	Erstellt bzw. fügt eine Datei mit einem gegebenen Dateipfad ('/' als Trennzeichen verwenden) mit einem bestimmten anzuhängenden Text an.	-
Helfer	SENDTOFORCAM(str1[, str2]) / SENDTOCLIENT (str1[, str2])	Existiert bereits eine Zeichenkette, wird "clientsend" oder "forcamsend" als erstes Argument verwendet, ansonsten wird die erste Zeichenkette als erstes Argument verwendet, um die Daten der zweiten Zeichenkette an die in Javis.INI konfigurierten TCP-Verbindungen zu senden.	-
Helfer	DEBUGOUT(text)	Loggt den Text auf Debug-Log-Ebene mit Parser-Logger.	-

Verwendung	Skriptfunktion Parameter in [..] sind optional	Beschreibung	Output-Event
Helfer	COPYFILE(inFile, outFile)	Kopiert Daten von in-file nach out-file. Argumente können Dateipfade sein. Bei Erfolg wird auch die zuletzt geänderte out-file als in-file aktualisiert.	-
Helfer	COPYREPLACE(inFile, outFile, searchStr, replaceStr)	Kopiert von in-file nach out-file wie bei Funktion COPYFILE, und ersetzt dabei alle Vorkommen von search-string durch replace-string.	-
Helfer	ATTIME(seconds, obj)	Berechnet das Objekt jeden Tag zu vorgegebenen Zeiten in Sekunden (Sekunden bedeutet Zeitanteil des aktuellen Tages in Sekunden).	-
Helfer	FROMASCII(num)	Sendet eine Zeichenkette zurück, die den numerischen Wert als num hat.	-
Helfer	SLEEP(ms)	Pausiert den aktuellen Thread für eine bestimmte Zeit in ms.	-