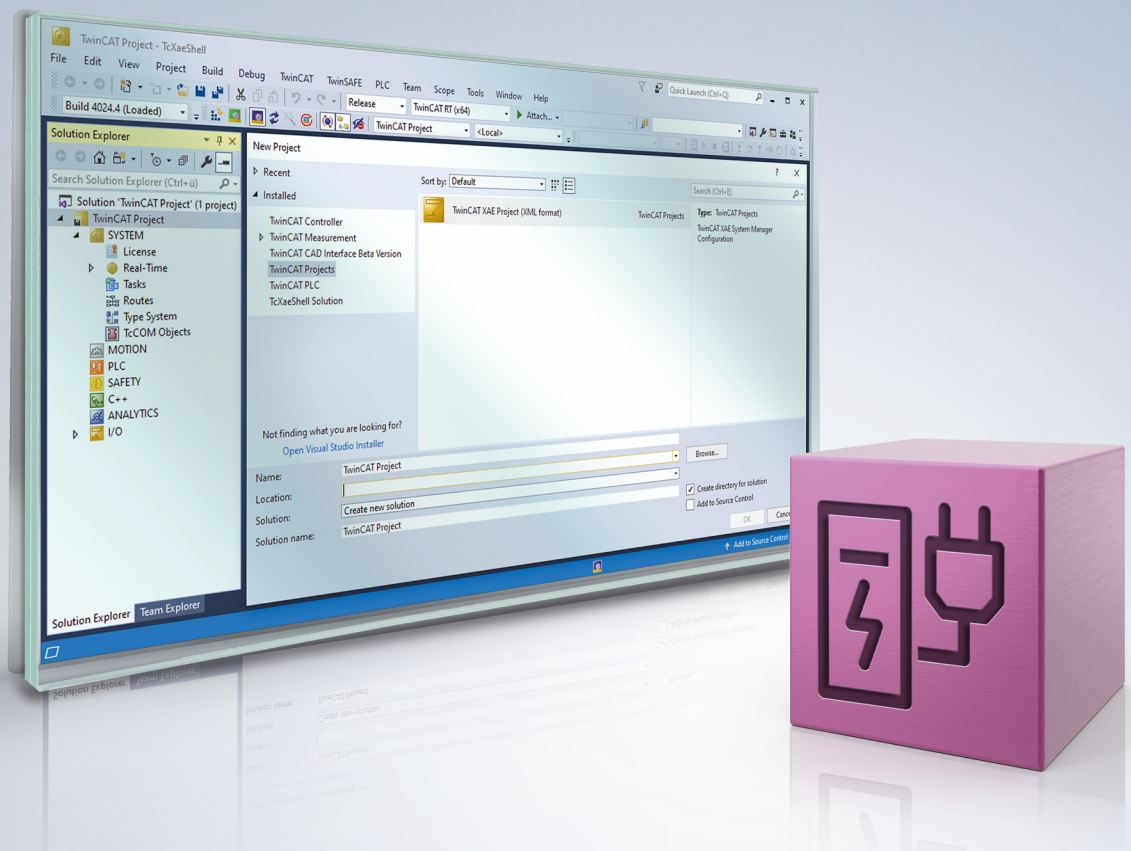
BECKHOFF New Automation Technology

Handbuch | DE

TF6771

TwinCAT 3 | IoT OCPP



31.10.2024 | Version: 1.1.0

Inhaltsverzeichnis

1 Vorwort

1.1 Hinweise zur Dokumentation

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs- und Automatisierungstechnik, das mit den geltenden nationalen Normen vertraut ist.
Zur Installation und Inbetriebnahme der Komponenten ist die Beachtung der Dokumentation und der nachfolgenden Hinweise und Erklärungen unbedingt notwendig.
Das Fachpersonal ist verpflichtet, stets die aktuell gültige Dokumentation zu verwenden.

Das Fachpersonal hat sicherzustellen, dass die Anwendung bzw. der Einsatz der beschriebenen Produkte alle Sicherheitsanforderungen, einschließlich sämtlicher anwendbaren Gesetze, Vorschriften, Bestimmungen und Normen erfüllt.

Disclaimer

Diese Dokumentation wurde sorgfältig erstellt. Die beschriebenen Produkte werden jedoch ständig weiterentwickelt.
Wir behalten uns das Recht vor, die Dokumentation jederzeit und ohne Ankündigung zu überarbeiten und zu ändern.
Aus den Angaben, Abbildungen und Beschreibungen in dieser Dokumentation können keine Ansprüche auf Änderung bereits gelieferter Produkte geltend gemacht werden.

Marken

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® und XPlanar® sind eingetragene und lizenzierte Marken der Beckhoff Automation GmbH.
Die Verwendung anderer in dieser Dokumentation enthaltenen Marken oder Kennzeichen durch Dritte kann zu einer Verletzung von Rechten der Inhaber der entsprechenden Bezeichnungen führen.

Patente

Die EtherCAT-Technologie ist patentrechtlich geschützt, insbesondere durch folgende Anmeldungen und Patente:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
mit den entsprechenden Anmeldungen und Eintragungen in verschiedenen anderen Ländern.

EtherCAT® ist eine eingetragene Marke und patentierte Technologie lizenziert durch die Beckhoff Automation GmbH, Deutschland

Copyright

1.2 Zu Ihrer Sicherheit

Sicherheitsbestimmungen

Lesen Sie die folgenden Erklärungen zu Ihrer Sicherheit.
Beachten und befolgen Sie stets produktspezifische Sicherheitshinweise, die Sie gegebenenfalls an den entsprechenden Stellen in diesem Dokument vorfinden.

Haftungsausschluss

Die gesamten Komponenten werden je nach Anwendungsbestimmungen in bestimmten Hard- und Software-Konfigurationen ausgeliefert. Änderungen der Hard- oder Software-Konfiguration, die über die dokumentierten Möglichkeiten hinausgehen, sind unzulässig und bewirken den Haftungsausschluss der Beckhoff Automation GmbH & Co. KG.

Qualifikation des Personals

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs-, Automatisierungs- und Antriebstechnik, das mit den geltenden Normen vertraut ist.

Signalwörter

Im Folgenden werden die Signalwörter eingeordnet, die in der Dokumentation verwendet werden. Um Personen- und Sachschäden zu vermeiden, lesen und befolgen Sie die Sicherheits- und Warnhinweise.

Warnungen vor Personenschäden

⚠ GEFAHR

Es besteht eine Gefährdung mit hohem Risikograd, die den Tod oder eine schwere Verletzung zur Folge hat.

⚠ WARNUNG

Es besteht eine Gefährdung mit mittlerem Risikograd, die den Tod oder eine schwere Verletzung zur Folge haben kann.

⚠ VORSICHT

Es besteht eine Gefährdung mit geringem Risikograd, die eine mittelschwere oder leichte Verletzung zur Folge haben kann.

Warnung vor Umwelt- oder Sachschäden

HINWEIS

Es besteht eine mögliche Schädigung für Umwelt, Geräte oder Daten.

Information zum Umgang mit dem Produkt

Diese Information beinhaltet z. B.:
Handlungsempfehlungen, Hilfestellungen oder weiterführende Informationen zum Produkt.

1.3 Hinweise zur Informationssicherheit

Die Produkte der Beckhoff Automation GmbH & Co. KG (Beckhoff) sind, sofern sie online zu erreichen sind, mit Security-Funktionen ausgestattet, die den sicheren Betrieb von Anlagen, Systemen, Maschinen und Netzwerken unterstützen. Trotz der Security-Funktionen sind die Erstellung, Implementierung und ständige Aktualisierung eines ganzheitlichen Security-Konzepts für den Betrieb notwendig, um die jeweilige Anlage, das System, die Maschine und die Netzwerke gegen Cyber-Bedrohungen zu schützen. Die von Beckhoff verkauften Produkte bilden dabei nur einen Teil des gesamtheitlichen Security-Konzepts. Der Kunde ist dafür verantwortlich, dass unbefugte Zugriffe durch Dritte auf seine Anlagen, Systeme, Maschinen und Netzwerke verhindert werden. Letztere sollten nur mit dem Unternehmensnetzwerk oder dem Internet verbunden werden, wenn entsprechende Schutzmaßnahmen eingerichtet wurden.

Zusätzlich sollten die Empfehlungen von Beckhoff zu entsprechenden Schutzmaßnahmen beachtet werden. Weiterführende Informationen über Informationssicherheit und Industrial Security finden Sie in unserem https://www.beckhoff.de/secguide.

Die Produkte und Lösungen von Beckhoff werden ständig weiterentwickelt. Dies betrifft auch die Security-Funktionen. Aufgrund der stetigen Weiterentwicklung empfiehlt Beckhoff ausdrücklich, die Produkte ständig auf dem aktuellen Stand zu halten und nach Bereitstellung von Updates diese auf die Produkte aufzuspielen. Die Verwendung veralteter oder nicht mehr unterstützter Produktversionen kann das Risiko von Cyber-Bedrohungen erhöhen.

Um stets über Hinweise zur Informationssicherheit zu Produkten von Beckhoff informiert zu sein, abonnieren Sie den RSS Feed unter https://www.beckhoff.de/secinfo.

1.4 Ausgabestände der Dokumentation

Version	Änderung
1.1.0	**Neu:** Troubleshooting [▶ 215]

2 Übersicht

Mit der TwinCAT Function TF6771 IoT OCPP kann TwinCAT sowohl als OCPP-Client als auch als OCPP-Server agieren. Die Use-Cases werden im weiteren Verlauf näher erläutert.

Produktkomponenten

Die Funktion TF6771 IoT OCPP besteht aus den folgenden Komponenten, die ab TwinCAT-Version 3.1.4026.x verwendet werden können:

- **Treiber:** TcIotOcpp.tmx (in der Installation von TF6771.IotOCPP.XAE enthalten)
- **SPS-Bibliothek:** Tc3_OCPP (in der Installation von TF6771.IotOCPP.XAE enthalten)

TF6771 IoT OCPP wird nur auf dem Engineering installiert. Sobald ein Projekt auf das Zielsystem gespielt wird, wird der TMX-Treiber mit den Projektdateien rüberkopiert.

Use-Cases

Mit Verwendung von TF6771 lassen sich vier Kategorien von Use-Cases abbilden. Dazu gehören die Verwendung als OCPP-Client, die Verwendung als OCPP-Server, die Realisierung eines OCPP-Gateways und das Ermöglichen eines OCPP-Retrofits.

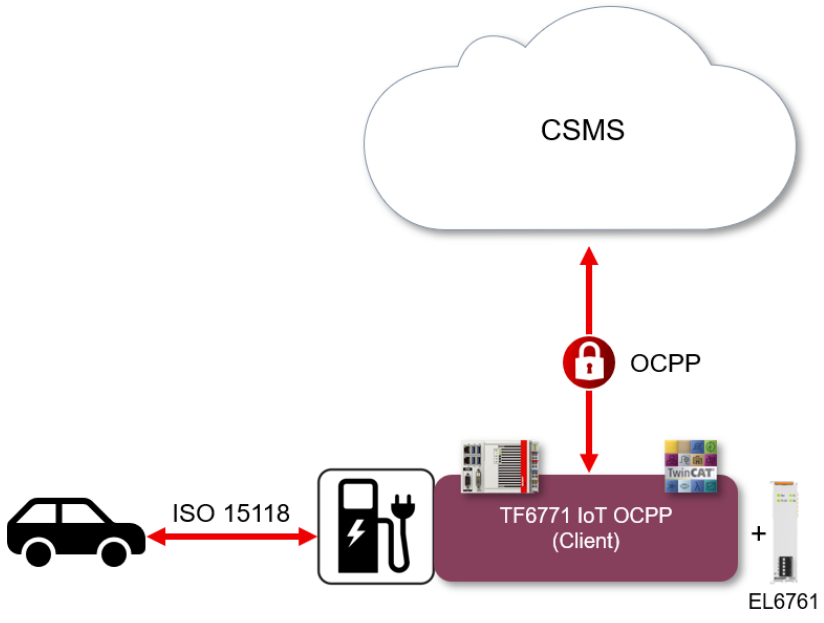
Begrifflichkeiten

Die zwei in dieser Dokumentation betrachteten Versionen des OCPP-Protokolls sehen verschiedene Begriffe vor.

- In Version 1.6 ist die Bezeichnung für die Ladestation **Charge Point**, in Version 2.0.1 hingegen ist die Bezeichnung **Charging Station**.
- In Version 1.6 ist die Bezeichnung für das übergeordnete Management-System **Central System**, in Version 2.0.1 hingegen ist die Bezeichnung **Charging Station Management System (CSMS)**.
- Zur besseren Verständlichkeit werden in der folgenden Beschreibung der Use-Cases und den zugehörigen Grafiken die Begrifflichkeiten aus OCPP-Version 2.0.1 verwendet.

Use-Case 1: TwinCAT Charging Station

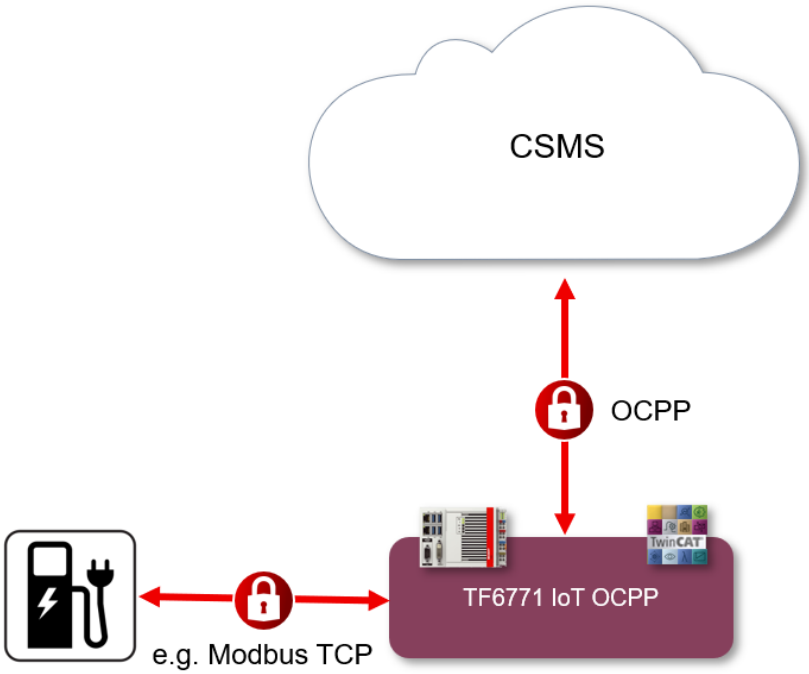
Im ersten Use-Case wird eine Charging Station mit einer Beckhoff-Steuerung automatisiert. Dazu wird neben der Software der TF6771 mit der EtherCAT-Klemme EL6761 noch eine weitere Komponente verwendet. Die folgende Grafik veranschaulicht eine Verwendung in Verbindung mit der EL6761.



Softwareseitig wird die mit TwinCAT automatisierte Charging Station als OCPP-Client mit einem CSMS verbunden. Die Kommunikation in Richtung des zu ladenden Fahrzeugs wird über die EL6761 realisiert.

Use-Case 2: OCPP-Retrofit

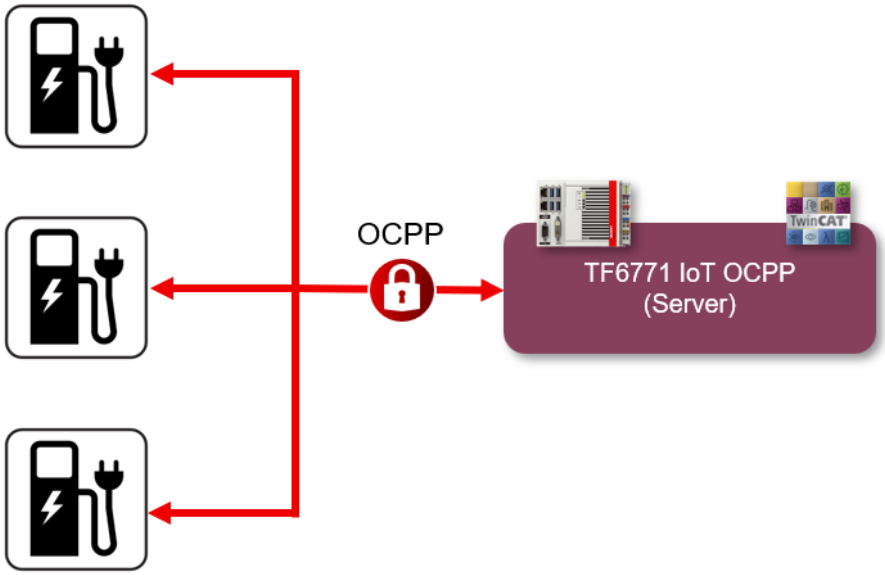
Im zweiten Use-Case werden bestehende Charging Stations ohne OCPP-Unterstützung für OCPP-Kommunikation nachgerüstet.



In bestehender Ladeinfrastruktur kann es Charging Stations geben, die ihre Informationen nicht über OCPP zur Verfügung stellen. Es gibt beispielsweise Charging Stations, die Informationen über Modbus TCP verteilen. Diese Informationen können dann innerhalb von TwinCAT gesammelt werden und stellvertretend für die Charging Station an das CSMS gesendet werden. Dazu agiert TwinCAT als OCPP-Client und gibt sich als die Charging Station aus, die nachgerüstet werden soll.

Use-Case 3: Lokales CSMS

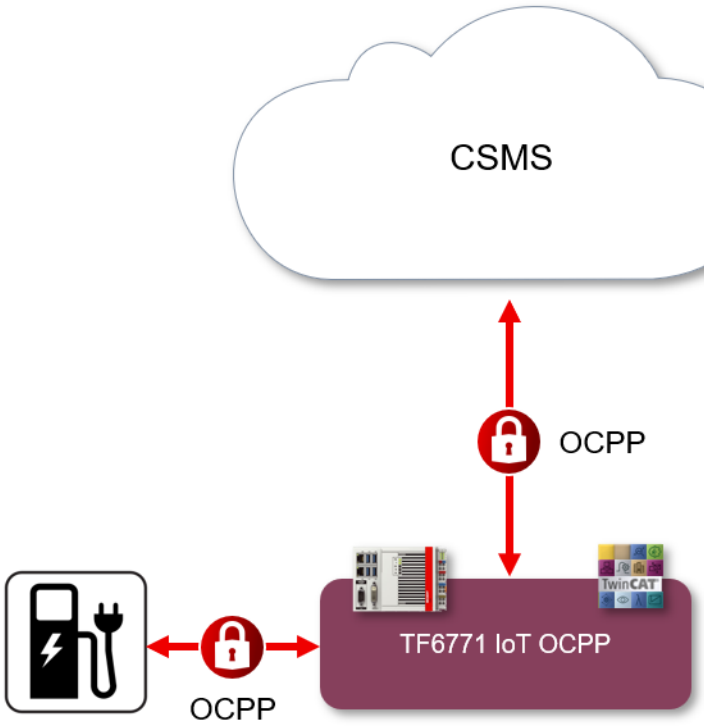
Mit TwinCAT kann ebenso ein lokales CSMS umgesetzt werden. Hier agiert TwinCAT als OCPP-Server, auf den sich eine beliebige Anzahl an Charging Stations verbinden kann.



An dieser Stelle müssen alle Funktionen des CSMS lokal umgesetzt werden. Dazu zählen neben der Festlegung von Ladeprofilen auch die Authentifizierung und die Abrechnung – falls diese Funktionen benötigt werden.

Use-Case 4: OCPP-Gateway

Auch die Umsetzung eines OCPP-Gateways ist möglich. Hier agiert TwinCAT sowohl als OCPP-Client als auch als OCPP-Server.



In so einem Fall wird oft das übergeordnete, in den meisten Fällen cloud-basierte, CSMS für Authentifizierung und Abrechnung genutzt. TwinCAT verhält sich in diesem Fall hauptsächlich als Kommunikationsweiche und leitet Nachrichten von den OCPP-Clients an das übergeordnete CSMS weiter und genauso andersherum. Zusätzlich ist es jedoch möglich, über den lokalen OCPP-Server im OCPP-Gateway ein lokales Lastmanagement umzusetzen. Dies kann unter anderem aufgrund der lokal in der Steuerung vorliegenden Informationen über das Stromnetz an einem Standort von Vorteil sein.

3 Installation

3.1 Systemanforderungen

Technische Daten	Beschreibung
Betriebssystem	Windows 7/10, Windows Embedded Standard 7, TwinCAT/BSD
Zielplattform	PC-Architektur (x86, x64 oder ARM)
TwinCAT-Version	TwinCAT 3.1 Build 4026.3 oder höher
Erforderliches TwinCAT-Setup-Level	TwinCAT 3 XAE, XAR
Erforderliche TwinCAT-Lizenz	TF6771 TC3 IoT OCPP

3.2 Installation

TwinCAT Package Manager

Wenn Sie TwinCAT 3.1 Build 4026 (und höher) auf dem Betriebssystem Microsoft Windows verwenden, können Sie diese Function über den TwinCAT Package Manager installieren, siehe Dokumentation zur Installation.

Normalerweise installieren Sie die Function über den entsprechenden Workload; dennoch können Sie die im Workload enthaltenen Pakete auch einzeln installieren. Diese Dokumentation beschreibt im Folgenden kurz den Installationsvorgang über den Workload.

Kommandozeilenprogramm TcPkg

Über das TcPkg **C**ommand **L**ine **I**nterface (CLI) können Sie sich die verfügbaren Workloads auf dem System anzeigen lassen:

```
tcpkg list -t workload
```

Über das folgende Kommando können Sie den Workload der TF6771 Iot OCPP-Function installieren.

```
tcpkg install TF6771.IotOCPP.XAE
```

TwinCAT Package Manager UI

Über das **U**ser **I**nterface (UI) können Sie sich alle verfügbaren Workloads anzeigen lassen und diese bei Bedarf installieren.
Folgen Sie hierzu den entsprechenden Anweisungen in der Oberfläche.

3.3 Lizenzierung

Die TwinCAT 3 Function ist als Vollversion oder als 7-Tage-Testversion freischaltbar. Beide Lizenztypen sind über die TwinCAT-3-Entwicklungsumgebung (XAE) aktivierbar.

Lizenzierung der Vollversion einer TwinCAT 3 Function

Die Beschreibung der Lizenzierung einer Vollversion finden Sie im Beckhoff Information System in der Dokumentation „TwinCAT-3-Lizenzierung“.

Lizenzierung der 7-Tage-Testversion einer TwinCAT 3 Function

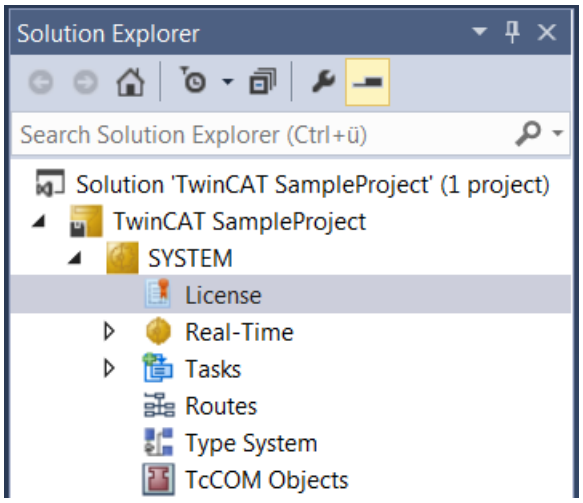
Eine 7-Tage-Testversion kann nicht für einen TwinCAT-3-Lizenz-Dongle freigeschaltet werden.

1. Starten Sie die TwinCAT-3-Entwicklungsumgebung (XAE).
2. Öffnen Sie ein bestehendes TwinCAT-3-Projekt oder legen Sie ein neues Projekt an.

3. Wenn Sie die Lizenz für ein Remote-Gerät aktivieren wollen, stellen Sie das gewünschte Zielsystem ein. Wählen Sie dazu in der Symbolleiste in der Drop-down-Liste **Choose Target System** das Zielsystem aus.

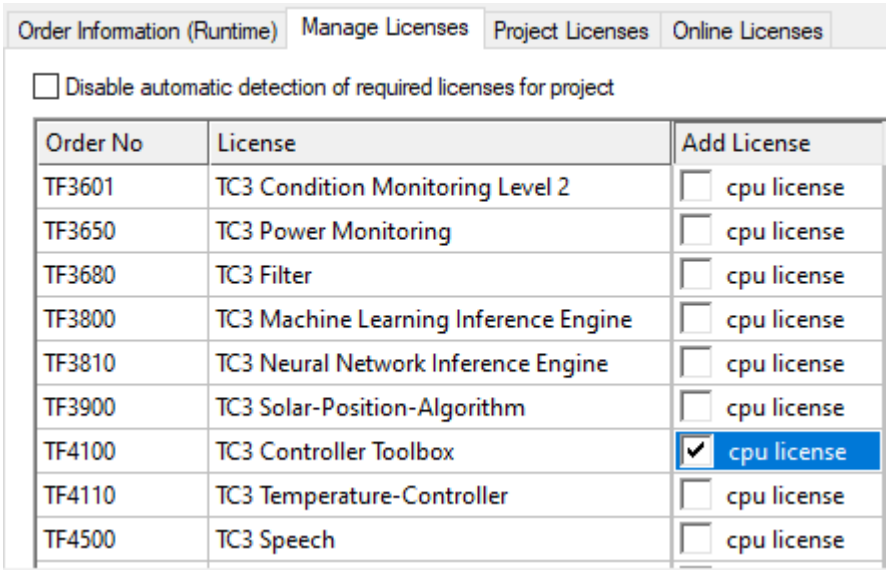
   ⇨ Die Lizenzierungseinstellungen beziehen sich immer auf das eingestellte Zielsystem. Mit der Aktivierung des Projekts auf dem Zielsystem werden automatisch auch die zugehörigen TwinCAT-3-Lizenzen auf dieses System kopiert.

4. Klicken Sie im **Solution Explorer** im Teilbaum **SYSTEM** doppelt auf **License.**



   ⇨ Der TwinCAT-3-Lizenzmanager öffnet sich.

5. Öffnen Sie die Registerkarte **Manage Licenses.** Aktivieren Sie in der Spalte **Add License** das Auswahlkästchen für die Lizenz, die Sie Ihrem Projekt hinzufügen möchten (z. B. „TF4100 TC3 Controller Toolbox“).

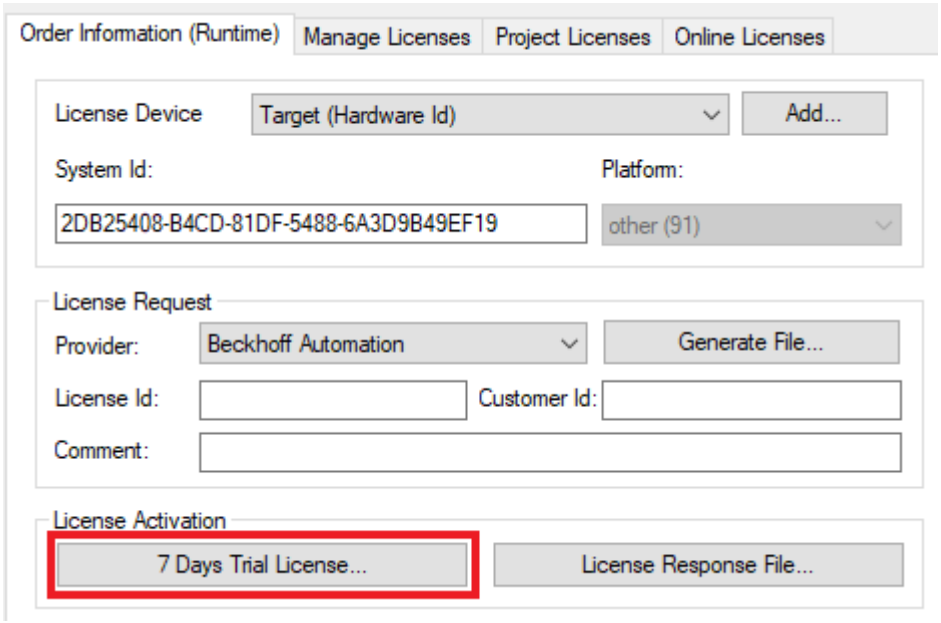


Order No	License	Add License
TF3601	TC3 Condition Monitoring Level 2	cpu license
TF3650	TC3 Power Monitoring	cpu license
TF3680	TC3 Filter	cpu license
TF3800	TC3 Machine Learning Inference Engine	cpu license
TF3810	TC3 Neural Network Inference Engine	cpu license
TF3900	TC3 Solar-Position-Algorithm	cpu license
TF4100	TC3 Controller Toolbox	cpu license
TF4110	TC3 Temperature-Controller	cpu license
TF4500	TC3 Speech	cpu license

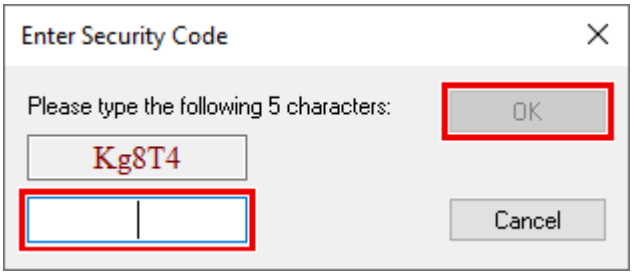
6. Öffnen Sie die Registerkarte **Order Information (Runtime).**

   ⇨ In der tabellarischen Übersicht der Lizenzen wird die zuvor ausgewählte Lizenz mit dem Status „missing“ angezeigt.

7. Klicken Sie auf **7 Days Trial License...**, um die 7-Tage-Testlizenz zu aktivieren.



⇨ Es öffnet sich ein Dialog, der Sie auffordert, den im Dialog angezeigten Sicherheitscode einzugeben.



8. Geben Sie den Code genauso ein, wie er angezeigt wird, und bestätigen Sie ihn.
9. Bestätigen Sie den nachfolgenden Dialog, der Sie auf die erfolgreiche Aktivierung hinweist.
   ⇨ In der tabellarischen Übersicht der Lizenzen gibt der Lizenzstatus nun das Ablaufdatum der Lizenz an.
10. Starten Sie das TwinCAT-System neu.

⇨ Die 7-Tage-Testversion ist freigeschaltet.

4 Technische Einführung

4.1 OCPP

Das Open Charge Point Protocol (OCPP) ist ein standardisiertes Kommunikationsprotokoll, das zwischen Ladestationen für Elektrofahrzeuge (EVSEs) und einem zentralen Management-System (CSMS) eingesetzt wird. Es ist ein offener und herstellerunabhängiger Standard, der eine breite Interoperabilität zwischen verschiedenen Herstellern von Ladestationen und den zugehörigen Management-System ermöglicht.

Technisch wird beim OCPP-Protokoll eine WebSockets-Verbindung zwischen OCPP-Client und OCPP-Server aufgebaut. Es ist vom Protokoll vorgesehen, dass ein OCPP-Request sowohl vom Client als auch vom Server ausgehen kann.

4.2 Begriffe

Im Folgenden werden einige für das Verständnis dieser Dokumentation wichtige Begriffe in Kürze erklärt. Für ein genaueres Verständnis des OCPP-Protokolls und die Erläuterung weiterer Begriffe aus dem OCPP-Protokoll muss die OCPP-Spezifikation zu Rate gezogen werden.

In der ersten Tabelle werden die wichtigsten Begriffe für das Grundverständnis der Komponenten erläutert:

Begriff	Erklärung
Charging Station (2.0.1)	Die Charging Station ist das physische System, an dem ein Elektrofahrzeug aufgeladen werden kann. Eine Charging Station hat einen oder mehrere EVSEs. Eine Charging Station hat in OCPP-Version 1.6 den Namen Charge Point.
Central System (1.6)	Bezeichnung für das zentrale Management-System. Dieser Begriff wird in Version 2.0.1 der OCPP-Spezifikation durch CSMS ersetzt.
Charge Point (1.6)	Der Charge Point ist das physische System, an dem ein Elektrofahrzeug aufgeladen werden kann. Ein Charge Point hat einen oder mehrere Connectors. Ein Charge Point hat in OCPP-Version 2.0.1 den Namen Charging Station.
Connector	Ein Connector ist laut OCPP-Spezifikation eine unabhängig betriebene und verwaltete Steckdose an einem Charge Point.
CSMS (Charging Station Management System) (2.0.1)	Bezeichnung für das zentrale Management-System. Dieser Begriff ersetzt den Begriff Central System aus Version 1.6 der OCPP-Spezifikation.
EVSE (2.0.1)	Eine EVSE wird als ein unabhängig betriebener und verwalteter Teil der Charging Station betrachtet, die jeweils ein Fahrzeug mit Energie versorgen kann.

In der zweiten Tabelle sind weitere Begriffserläuterungen zum OCPP-Protokoll zu finden:

Begriff	Erklärung
Charging Profile	Generisches Charging Profile, das für verschiedene Arten von Profilen verwendet wird. Enthält Informationen über das Profil und enthält die Charging Schedule.
Charging Schedule	Teil eines Charging Profiles. Definiert einen Block von Grenzwerten für die Ladeleistung oder den Ladestrom. Kann eine Startzeit und eine Länge enthalten.
Charging Schedule Period	Spezifische Zeitperiode in einer Charging Schedule. Eine Charging Schedule enthält eine oder mehrere Charging Schedule Periods.
Configuration Key	Konfigurierbare Einstellung in einem Charge Point/einer Charging Station.
Local Authorization List	Die Local Authorization List ermöglicht es dem Charge Point/der Charging Station, Ladevorgänge zu autorisieren, selbst wenn sie temporär nicht mit dem Central System/CSMS kommunizieren kann.
MeterValue	Container für einen oder mehrere SampledValues zu einem bestimmten Zeitpunkt.
SampledValue	Einzelner Messwert, der verschiedene Daten enthalten kann.
Transaction (1.6)	Der Teil des Ladevorgangs, der beginnt, wenn alle relevanten Vorbedingungen (z. B. Autorisierung, eingesteckter Stecker) erfüllt sind. Er endet in dem Moment, in dem die Stromzapfsäule diesen Zustand unwiderruflich verlässt.
Transaction (2.0.1)	Eine Transaktion in OCPP ist ein Teil des gesamten Ladevorgangs eines Elektrofahrzeugs, der auf der Grundlage konfigurierbarer Parameter beginnt und endet. Diese konfigurierbaren Parameter beziehen sich auf Momente im Ladevorgang, wie z.B. das Anschließen des E-Fahrzeugs oder die Autorisierung des Fahrers.
Unknown Offline Authorization	Der Charge Point/die Charging Station hat die Möglichkeit, in einer Offline-Situation unbekannte Benutzer zuzulassen und ihre Autorisierung im Nachhinein zu prüfen.

4.3 Unterstützte Funktionen

Unterstützte Versionen und Datenformate

Das OCPP-Protokoll in der Version 1.6 sieht entweder ein SOAP-basiertes Datenformat oder ein JSON-basiertes Datenformat vor. In Version 2.0.1 des Protokolls wird nur noch die JSON-Variante unterstützt. Die folgende Tabelle zeigt die Unterstützung in der vorliegenden Implementierung der TF6771.

OCPP-Version	Unterstützung?
OCPP1.6S (SOAP)	Nein
OCPP1.6J (JSON)	Ja
OCPP2.0.1J (JSON)	Folgt

4.4 Quick Start

Das folgende Kapitel ermöglicht einen Schnelleinstieg in die Function TwinCAT 3 IoT OCPP. In dieser Anleitung setzen Sie einen OCPP-Client und einen OCPP-Server auf und lassen diese in vereinfachter Form miteinander kommunizieren. Weitergehende Funktionalitäten können aus den weiteren Beispielprojekten entnommen werden.

Beispielhafte Umsetzung

In diesem Quick Start wird aus Vereinfachungsgründen auf Security verzichtet. In realen Applikationen sollte die sichere Implementierung immer ein zentraler Bestandteil der Betrachtungen sein.

Initialisierung OCPP-Client

Bei der Initialisierung kann entweder der Funktionsbaustein direkt genutzt werden oder ein bereits bestehendes TcCOM-Objekt verknüpft werden. In diesem Getting Started wird die Initialisierung über den Funktionsbaustein durchgeführt.

```
    // Client param
    stClientParam      : ST_OCPP1_ClientParam :=(sHost:='127.0.0.1', nPort:=8080, sIdentity:='TestId
ent', sPath:='ocpp', eDebugLevel:=E_OCPP_DebugLevel.MessageLogFile, eTraceLevel:=TcTraceLevel.tlVerb
ose);
    fbClient           : FB_OCPP1_Client(stClientParam);
    eAction            : E_OCPP1_Action;
    hMessageId         : T_OCPP_MessageId;
```

Der OCPP-Client verbindet sich in diesem Fall mit dem ebenfalls auf demselben Gerät unter Port 8080 laufenden OCPP-Server. Zu Demonstrationszwecken werden sowohl das Logfile angelegt als auch das Trace-Level hochgestellt. Dies ist im Regelbetrieb nicht unbedingt notwendig. Einzelheiten zum Logging finden Sie im Kapitel Logging [▶ 20].

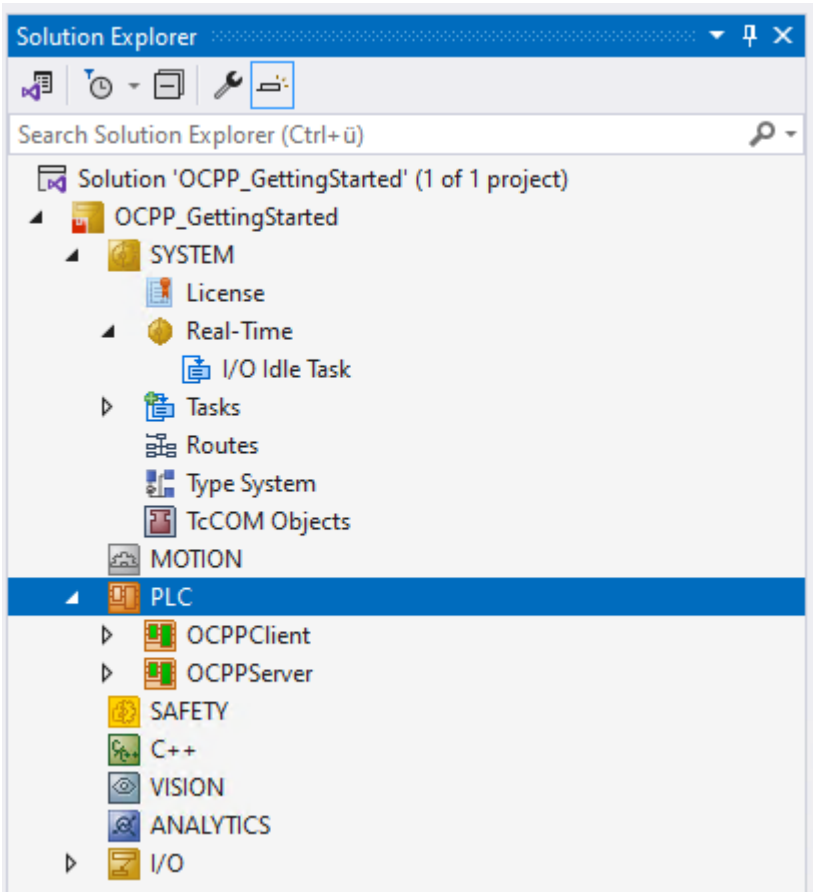
Initialisierung OCPP-Server

```
    // Server param
    stServerParam       : ST_OCPP1_ServerParam :=
                           (nPort:=8080,
                            eAuthMode:=E_OCPP1_AuthenticationMode.None,
                            eEncryptionMode := E_OCPP_EncryptionMode.None,
                            eDebugLevel:=E_OCPP_DebugLevel.MessageLogFile,
                            eTraceLevel:=TcTraceLevel.tlVerbose);
    fbOcppServer        : FB_OCPP1_Server(stServerParam);
    eAction             : E_OCPP1_Action;
    hMessageId          : T_OCPP_MessageId;
    hStationId          : UDINT;
```

Der vorliegende OCPP-Server läuft ebenfalls auf dem lokalen Gerät, auf Port 8080. Auch hier werden zu Demonstrationszwecken Logfile und Trace-Level eingestellt.

Grundaufbau Beispielprojekt

In diesem Getting Started wird ein TwinCAT-Projekt verwendet in dem sowohl OCPP-Client als auch OCPP-Server in einem separaten SPS-Projekt betrieben werden. Die Aktionen gehen vom OCPP-Client aus. In einem realen Szenario werden selbstverständlich auch Server-seitig Aktionen ausgelöst.



Implementierung OCPP-Client

Das Beispiel umfasst auf Seite des Clients zwei verschiedene Abläufe. Zum einen kann eine einfache StatusNotification abgesendet werden:

```
// Send StatusNotification
IF bSendStatus THEN
    bSendStatus:=FALSE;
    eError:=E_OCPP1_ChargePointError.HighTemperature;
    eStatus:=E_OCPP1_ChargePointStatus.SuspendedEV;
    eAction:=E_OCPP1_Action.StatusNotification;
END_IF
```

Über die Parameter eError und eStatus können die mit der StatusNotification versendeten Werte angepasst werden. Zum anderen wird ein abgespeckter Ablauf zum simulierten Laden umgesetzt:

```
// Charging sample
CASE nState OF
    0:
        IF bStartCharging THEN
            bStartCharging:=FALSE;
            eAction:=E_OCPP1_Action.Authorize;
            nState:=nState+1;
        END_IF
    1:
        IF bAuthorized THEN
            bAuthorized:=FALSE;
            nMeterStart:=nMeterStop;
            eAction:=E_OCPP1_Action.StartTransaction;
            nState:=nState+1;
        END_IF
    2:
        IF bChargingStarted THEN
            //10 seconds charging simulation
            timerCharge(IN:=TRUE);
            IF timerCharge.Q THEN
                bChargingStarted:=FALSE;
                timerCharge(IN:=FALSE);
                nMeterStop:=nMeterStart+10;
                eAction:=E_OCPP1_Action.StopTransaction;
                nState:=nState+1;
```

```
            END_IF
        END_IF
    3:
        IF bChargingStopped THEN
            bChargingStopped:=FALSE;
            nState:=0;
        END_IF
END_CASE
```

Wenn das Charging Sample über bStartCharging gestartet wird, ruft der Client zunächst die Authorize-Methode auf. Im Anschluss an ein erfolgreiches Authorize wird dann die StartTransaction-Methode ausgeführt. Nach erfolgreichem StartTransaction wird dann 10 Sekunden lang „geladen“ und anschließend ein StopTransaction ausgeführt.

Die spezifische OCPP-Kommunikation wird über eine Case-Anweisung umgesetzt. In dieser Case-Anweisung werden die einzelnen OCPP-Kommandos implementiert:

```
// Different OCPP commands implemented for OCPP client CASE eAction OF
    E_OCPP1_Action.None:
        IF NOT fbClient.bError AND fbClient.bValid THEN
            fbClient.PollRequest(hMessageId=> hMessageId, eAction=> eAction );
        ELSE
            // Implement error handling here
        END_IF
    E_OCPP1_Action.Authorize:
        IF fbClient.SendAuthorize(sIdTag:=sIdTag, eStatus => stIdTagInfo.eStatus) THEN
            IF stIdTagInfo.eStatus = E_OCPP1_AuthorizationStatus.Accepted THEN
                bAuthorized := TRUE;
                eAction    := E_OCPP1_Action.None;
            END_IF
        END_IF
    E_OCPP1_Action.StartTransaction:
        IF fbClient.SendStartTransaction(sIdTag:=sIdTag, nConnectorId:=nConnectorId, nMeterStart:=nM
eterStart, eStatus => stIdTagInfo.eStatus, nTransactionId => nTransactionId) THEN
            IF stIdTagInfo.eStatus = E_OCPP1_AuthorizationStatus.Accepted THEN
                bChargingStarted := TRUE;
                eAction    := E_OCPP1_Action.None;
            END_IF
        END_IF
    E_OCPP1_Action.StopTransaction:
        IF fbClient.SendStopTransaction(nConnectorId:=nConnectorId, nMeterStop:=nMeterStop, eStatus
=> stIdTagInfo.eStatus) THEN
            bChargingStopped := TRUE;
            eAction    := E_OCPP1_Action.None;
        END_IF
    E_OCPP1_Action.StatusNotification:
        IF fbClient.SendStatusNotification(nConnectorId:=nConnectorId, eError, eStatus) THEN
            eAction    := E_OCPP1_Action.None;
        END_IF
    // Implement here more functionalities like:
    // E_OCPP1_Action.TriggerMessage:
ELSE
    eAction := E_OCPP1_Action.None;
    // not implemented...
END_CASE
```

Hier sind nur die für das Beispiel verwendeten Funktionen implementiert. Weitere Funktionen würden eine Erweiterung der Case-Anweisung bedeuten.

Implementierung OCPP-Server

Auf Seite des OCPP-Servers werden die vom Client gesendeten Befehle empfangen und die passende Antwort geschickt:

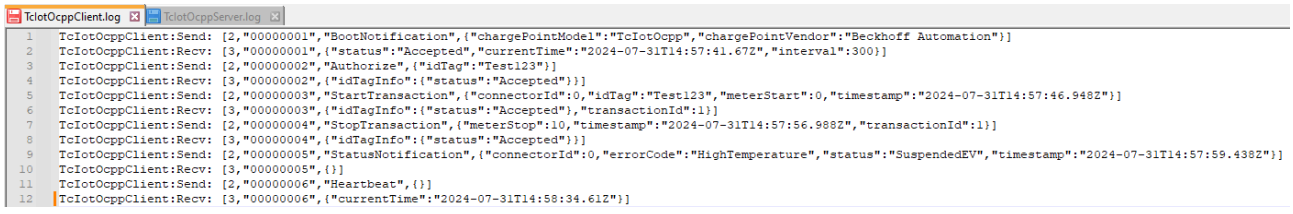
```
CASE eAction OF
    E_OCPP1_Action.None:
        IF NOT fbOcppServer.bError AND fbOcppServer.bValid THEN
            fbOcppServer.PollRequest(eAction => eAction, hMessageId => hMessageId, hStationId => hSt
ationId);
        ELSE
            // Implement error handling here
        END_IF
    E_OCPP1_Action.BootNotification:
        IF fbOcppServer.RecvBootNotification(hStationId, hMessageId=>hMessageId,
            sModel                => stBootNotification.sModel,
            sVendor                => stBootNotification.sVendor,
            sChargeBoxSerial    => stBootNotification.sChargeBoxSerial,
```

```
            sChargePointSerial    => stBootNotification.sChargePointSerial,
            sFirmwareVersion    => stBootNotification.sFirmwareVersion,
            sMeterSerial        => stBootNotification.sMeterSerial,
            sMeterType            => stBootNotification.sMeterType)
        THEN
            fbOcppServer.RespBootNotification(hStationId, hMessageId, E_OCPP1_RegistrationStatus.Acc
epted);
            eAction := E_OCPP1_Action.None;
        END_IF
    E_OCPP1_Action.Heartbeat:
        IF fbOcppServer.RecvHeartbeat(hStationId) THEN
            fbOcppServer.RespHeartbeat(hStationId:=hStationId,hMessageId:=hMessageId);
            eAction := E_OCPP1_Action.None;
        END_IF
    E_OCPP1_Action.Authorize:
        IF fbOcppServer.RecvAuthorize(hStationId, sIdTag => sIdTag) THEN
            IF sIdTag='Test123' THEN
                fbOcppServer.RespAuthorize(hStationId:=hStationId, hMessageId:=hMessageId, E_OCPP1_A
uthorizationStatus.Accepted);
                eAction := E_OCPP1_Action.None;
            END_IF
        END_IF
    E_OCPP1_Action.StartTransaction:
        IF fbOcppServer.RecvStartTransaction(hStationId, hMessageId => hMessageId, sIdTag => sIdTag,
 nConnectorId => nConnectorId, nMeterStart => nMeterStart, nTimestamp => nTimestamp) THEN
            nTransactionId:=nTransactionId+1;
            fbOcppServer.RespStartTransaction(hStationId, hMessageId, E_OCPP1_AuthorizationStatus.Ac
cepted, nTransactionId:=nTransactionId);
            eAction := E_OCPP1_Action.None;
        END_IF
    E_OCPP1_Action.StopTransaction:
        IF fbOcppServer.RecvStopTransaction(hStationId, hMessageId => hMessageId, sIdTag => sIdTag,
nTransactionId => nTransactionIdRecv, nConnectorId => nConnectorId, nMeterStop => nMeterStop, nTimes
tamp => nTimestamp, eReason => eReason) THEN
            fbOcppServer.RespStopTransaction(hStationId, hMessageId, E_OCPP1_AuthorizationStatus.Acc
epted);
            eAction := E_OCPP1_Action.None;
        END_IF
    E_OCPP1_Action.StatusNotification:
        IF fbOcppServer.RecvStatusNotification(hStationId, hMessageId => hMessageId, nConnectorId =>
 nConnectorIdStatus, eError => eError, eStatus => eStatus) THEN
            fbOcppServer.RespStatusNotification(hStationId, hMessageId);
            eAction := E_OCPP1_Action.None;
        END_IF
END_CASE
```

Hier sind im Vergleich zum Client-Sample noch zwei Funktionen mehr implementiert. Der Heartbeat wird vom Client intern abgeschickt und kann über ein Property am Funktionsbaustein konfiguriert werden. Die BootNotification wird automatisch beim Aufstarten des Clients versendet und muss dementsprechend ebenfalls im Server verarbeitet werden.

OCPP-Nachrichten in Server und Client können auch in die andere Richtung implementiert werden. Der Server kann Nachrichten in Richtung des Clients schicken. Andersrum kann dann der Client diese Nachrichten vom Server empfangen.

Das Logfile sieht nach einmaliger Durchführung des Client-Samples (Senden einer StatusNotification und Ausführen des Charging Samples) wie folgt aus:

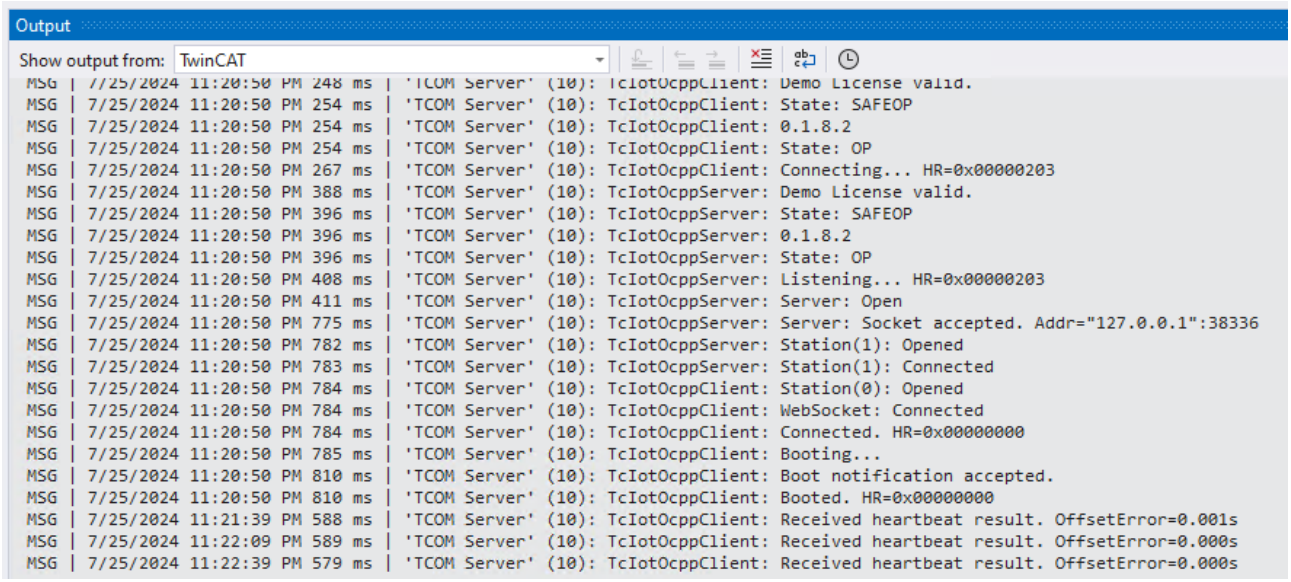


4.5 Logging

Um eine erweiterte Diagnose zu ermöglichen, gibt es verschiedene Logging-Funktionalitäten.

TwinCAT Output

Die erste zu berücksichtigende Stelle ist der TwinCAT Output. Hier werden je nach Trace-Level Informationen reingeschrieben. Im Fehlerfall kann zunächst dort nach einer dem Fehlverhalten entsprechenden Meldung geschaut werden:



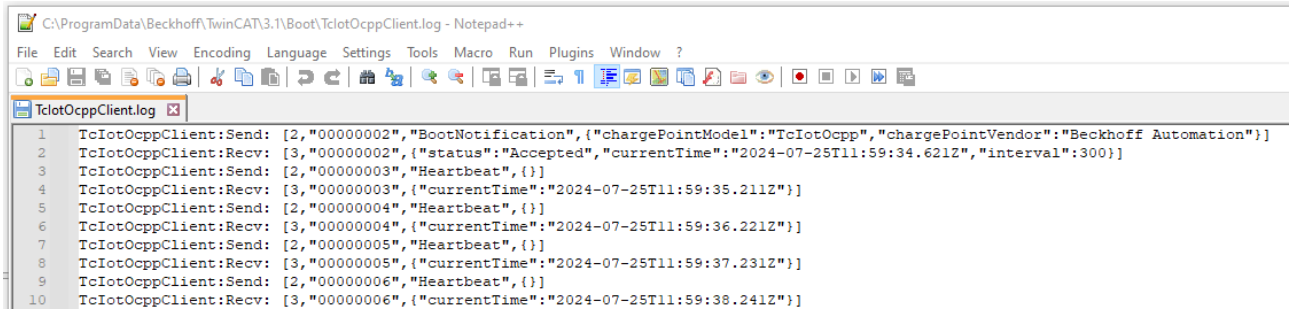
Das Trace-Level kann über die Strukturen ST_OCPP1_Client_Param [▶ 201] und ST_OCPP1_Server_Param [▶ 203] angepasst werden. Dies geht nicht nur zum Start der Applikation, sondern auch zur Laufzeit. Dazu muss der Wert geändert und die jeweilige Init-Methode von Client oder Server aufgerufen werden. Die folgende Tabelle zeigt die verschiedenen TcTraceLevel und ihre Auswirkungen:

TcTraceLevel	Auswirkung
tlAlways	Es werden keine Nachrichten über den TwinCAT Output ausgegeben.
tlError	Es werden nur Fehlermeldungen ausgegeben.
tlWarning	Es werden Warnungen und Fehlermeldungen ausgegeben.
tlInfo	Es werden einmalige Zustandsinformationen, Warnungen und Fehlermeldungen ausgegeben.
tlVerbose	Es werden alle Nachrichten ausgegeben. Dazu gehören zyklische Informationen, einmalige Zustandsinformationen, Warnungen und Fehlermeldungen.

Logfile

Die zweite Diagnosemöglichkeit ist das Schreiben eines Logfiles. Hier gibt es nur zwei verschiedene Optionen, das Schreiben des Logfiles ist entweder aktiviert oder deaktiviert. Das Trace-Level kann über die Strukturen ST_OCPP1_Client_Param [▶ 201] und ST_OCPP1_Server_Param [▶ 203] angepasst werden.

Im Logfile werden die OCPP-Nachrichten geloggt. Der folgende zeigt ein Beispiel für ein aktiviertes Logfile im OCPP-Client.



Die Logfiles werden im Boot-Verzeichnis von TwinCAT abgelegt:

C:\ProgramData\Beckhoff\TwinCAT\3.1\Boot

Nutzung des Logfiles nur für Debugging-Zwecke

Das Logfile sollte nur für Debugging-Zwecke eingesetzt werden. In der Datei werden alle OCPP-Nachrichten geloggt, sodass über längere Zeiträume große Dateigrößen des Logfiles entstehen können.

5 SPS-API

5.1 POUs

5.1.1 FB_OCPP1_Client

Dieser Baustein repräsentiert einen OCPP-Client, der mit einem OCPP-Server verbunden werden kann. Dabei gibt es zwei Kommunikationsrichtungen.

Beim Senden von Anfragen an den Server werden die Send-Methoden verwendet. In diesen Methoden wird die Antwort des Servers direkt verarbeitet und in den Ausgabeparametern der Methoden hinterlegt.

Wird hingegen eine Anfrage vom Server durch eine der Receive-Methoden empfangen, muss die Antwort mittels der passenden Response-Methode gesendet werden.

Der Client sendet intern einen Heartbeat, dessen Zeitintervall über das Property HeartbeatInterval angepasst werden kann. Erhält der Client keine Antwort vom Server auf einen Heartbeat, wird anschließend ein Reconnect zum Server versucht. Bei allen anderen Nachrichtentypen wird bei fehlender Antwort ein Timeout-Fehler am Client-Baustein ausgegeben.

Syntax

```
FUNCTION BLOCK FB_OCPP1_Client
VAR_OUTPUT
    bValid              : BOOL;
    bBusy               : BOOL;
    bError              : BOOL;
    eErrorResult        : HRESULT;
    eErrorAction        : E_OCPP1_Action;
END_VAR
```

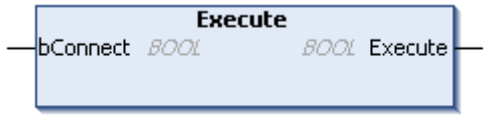
Ausgänge

Name	Typ	Beschreibung
bValid	BOOL	Die Schnittstelle zum Treiber im Hintergrund besteht.
bBusy	BOOL	Ist TRUE, solange der Baustein mit einer Bearbeitung beschäftigt ist.
bError	BOOL	Wird TRUE, sobald eine Fehlersituation auftritt.
eErrorResult	HRESULT	Zuletzt am Baustein anliegender Fehler.
eErrorAction	E_OCPP1_Action [▶ 190]	OCPP-Befehl, bei dem der Fehler aufgetreten ist.

Eigenschaften

Name	Typ	Zugriff	Beschreibung
HeartbeatInterval	TIME	Get, Set	Internes HeartbeatInterval vom Client. Zusätzlich kann auch manuell ein weiterer Heartbeat über die SendHeartbeat [▶ 70]-Methode gesendet werden. Beim internen HeartbeatInterval ist zu beachten, dass der Server in seiner BootNotification.conf mit einem vorgesehenen HeartbeatInterval für den Client antwortet. Dieses vorgesehene HeartbeatInterval wird dann intern gesetzt. Zur Laufzeit kann das HeartbeatInterval dann aber beliebig verändert werden.
IsConnected	BOOL	Get	Status der Verbindung zum OCPP-Server.
IsPending	BOOL	Get	Warten auf Fertigstellung der Anfrage.

5.1.1.1 Execute



Diese Methode kann verwendet werden, um eine Verbindung herzustellen oder eine Verbindung zu trennen. Es ist kein dauerhafter Aufruf erforderlich, es werden nur Änderungen von bConnect an den Treiber übertragen.

Bei einer bestehenden Verbindung wird über FALSE die Verbindung getrennt, bei einer nicht bestehenden Verbindung wird diese über TRUE hergestellt.

Syntax

```
METHOD Execute : BOOL
VAR_INPUT
    bConnect : BOOL;
END_VAR
```

Rückgabewert

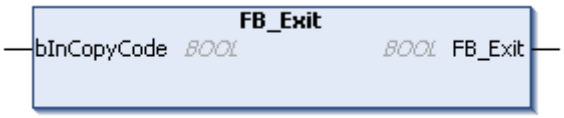
Name	Typ	Beschreibung
Execute	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
bConnect	BOOL	Bei einer bestehenden Verbindung wird über FALSE die Verbindung getrennt, bei einer nicht bestehenden Verbindung wird diese über TRUE hergestellt.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.2 FB_Exit



Syntax

```
METHOD FB_Exit : BOOL
VAR_INPUT
    bInCopyCode : BOOL;
END_VAR
```

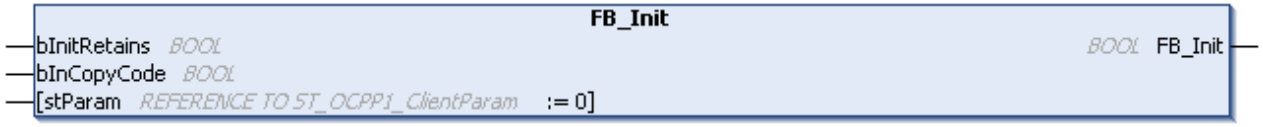
Rückgabewert

Name	Typ	Beschreibung
FB_Exit	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
bInCopyCode	BOOL	Wenn TRUE, wird die Exit-Methode zum Verlassen einer Instanz aufgerufen, die anschließend kopiert wird (Online Change).

5.1.1.3 FB_Init



Syntax

```
METHOD FB_Init : BOOL
VAR_INPUT
    bInitRetains : BOOL;
    bInCopyCode  : BOOL;
    stParam      : REFERENCE TO ST_OCPP1_Client_Param REF=0;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
FB_Init	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
bInitRetains	BOOL	Wenn TRUE, dann werden die Retain-Variablen initialisiert (Warm Start/Cold Start).
bInCopyCode	BOOL	Wenn TRUE, wird die Exit-Methode zum Verlassen einer Instanz aufgerufen, die anschließend kopiert wird (Online Change).
stParam	REFERENCE TO ST_OCPP1_Client_Param [▶ 201]	Parameter für die WebSockets-Verbindung des OCPP-Clients.

5.1.1.4 Init

Diese Methode wird beim Initialisieren des Funktionsbausteins in der Methode FB_Init [▶ 24] aufgerufen und spezifiziert die Parameter der WebSockets-Verbindung. Wenn die Verbindungsparameter im Nachhinein geändert werden sollen, kann diese Methode während des Laufens der Applikation erneut aufgerufen werden.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    stParam : REFERENCE TO ST_OCPP1_Client_Param;
END_VAR
```

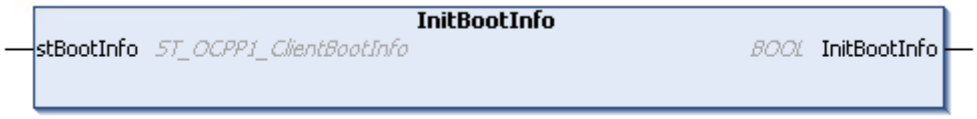
Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
stParam	REFERENCE TO ST_OCPP1_Client_Param [▶ 201]	Parameter für die WebSockets-Verbindung des OCPP-Clients.

5.1.1.5 InitBootInfo



Mit dieser Methode kann die beim Herstellen einer OCPP-Verbindung vom OCPP-Client versendete Boot Notification eingestellt werden. Wenn diese Methode nach dem Starten der Applikation nicht aufgerufen wird, werden Standardwerte verwendet. Genauere Informationen sind ST_OCPP1_Client_BootInfo [▶ 199] zu entnehmen.

Syntax

```
METHOD InitBootInfo : BOOL
VAR_IN_OUT CONSTANT
    stBootInfo : ST_OCPP1_Client_BootInfo;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
InitBootInfo	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ein-/Ausgänge

Name	Typ	Beschreibung
stBootInfo	ST_OCPP1_Client_BootInfo [▶ 199]	BootInfo für den OCPP-Client.

5.1.1.6 InitOptions

Mit dieser Methode können verschiedene Optionen für den OCPP-Client eingestellt werden. Wenn diese Methode nach dem Starten der Applikation nicht aufgerufen wird, werden Standardwerte verwendet. Genauere Informationen sind ST_OCPP1_Client_Options [▶ 200] zu entnehmen.

Syntax

```
METHOD InitOptions : BOOL
VAR_IN_OUT CONSTANT
    stOptions : ST_OCPP1_Client_Options;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
InitOptions	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ein-/Ausgänge

Name	Typ	Beschreibung
stOptions	ST_OCPP1_Client_Options [▶ 200]	Optionen für den OCPP-Client.

5.1.1.7 InitSettings

Mit dieser Methode können verschiedene Einstellungen für den OCPP-Client getroffen werden. Wenn diese Methode nach dem Starten der Applikation nicht aufgerufen wird, werden Standardwerte verwendet. Genauere Informationen sind ST_OCPP1_Client_Settings [▶ 202] zu entnehmen.

Syntax

```
METHOD InitSettings : BOOL
VAR_IN_OUT CONSTANT
    stSettings : ST_OCPP1_Client_Settings;
END_VAR
```

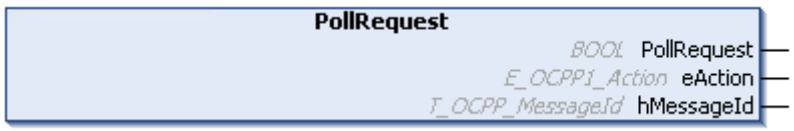
Rückgabewert

Name	Typ	Beschreibung
InitSettings	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ein-/Ausgänge

Name	Typ	Beschreibung
stSettings	ST_OCPP1_Client_Settings [▶ 202]	Einstellungen für den OCPP-Client.

5.1.1.8 PollRequest



Diese Methode muss aufgerufen werden, um eingehende Requests des OCPP-Servers zu empfangen. Die Ausgänge der Methode liefern sowohl Message-ID der empfangenen Nachricht als auch die Information, um welchen OCPP-Request es sich handelt zurück.

Syntax

```
METHOD PollRequest : BOOL
VAR_OUTPUT
    eAction    : E_OCPP1_Action;
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
PollRequest	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

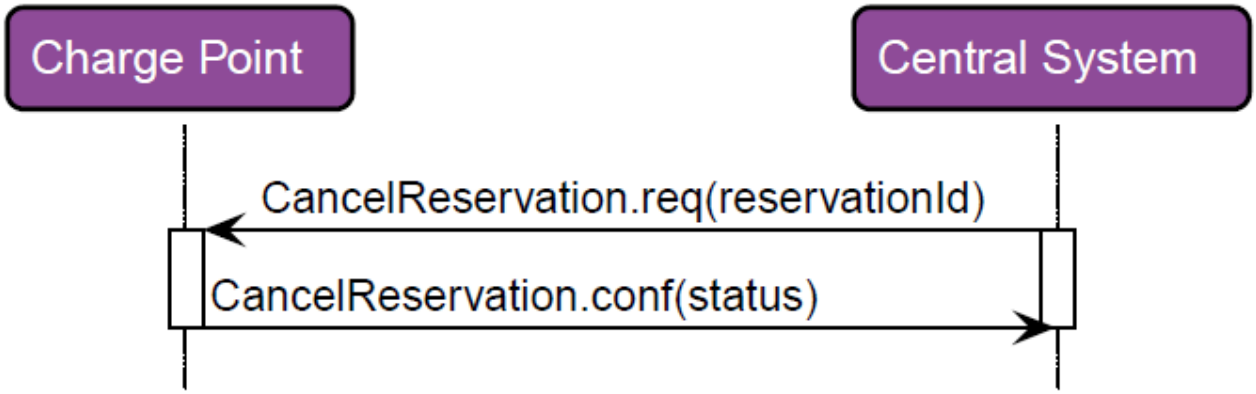
Ausgänge

Name	Typ	Beschreibung
eAction	E_OCPP1_Action [▶ 190]	Art des vom Server empfangenen OCPP-Requests.
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.9 RecvCancelReservation

Mit dieser Methode empfängt ein OCPP-Client einen Cancel Reservation-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespCancelReservation [▶ 48] aufgerufen werden.



Syntax

```
METHOD RecvCancelReservation : BOOL
VAR_OUTPUT
    hMessageId     : T_OCPP_MessageId;
    nReservationId : UDINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvCancelReservation	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

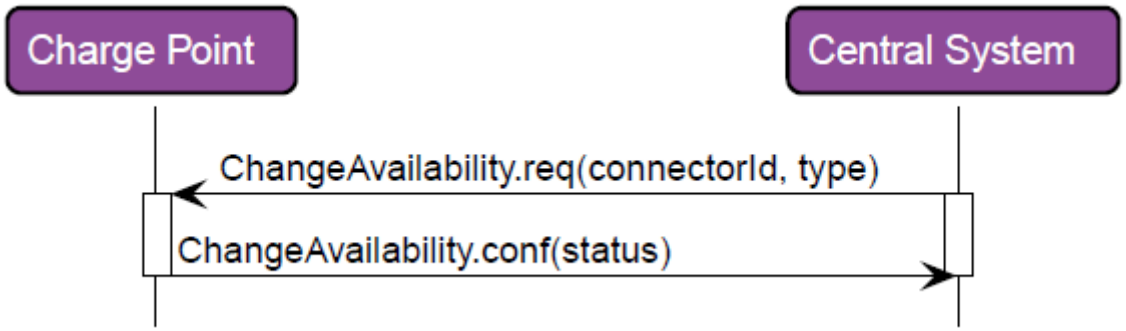
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nReservationId	UDINT	ID der zu stornierenden Reservierung.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.10 RecvChangeAvailability

Mit dieser Methode empfängt ein OCPP-Client einen Change Availability-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespChangeAvailability [▶ 49] aufgerufen werden.



Syntax

```
METHOD RecvChangeAvailability : BOOL
VAR_OUTPUT
    hMessageId   : T_OCPP_MessageId;
    nConnectorId : UDINT;
    eType        : E_OCPP1_AvailabilityType;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvChangeAvailability	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

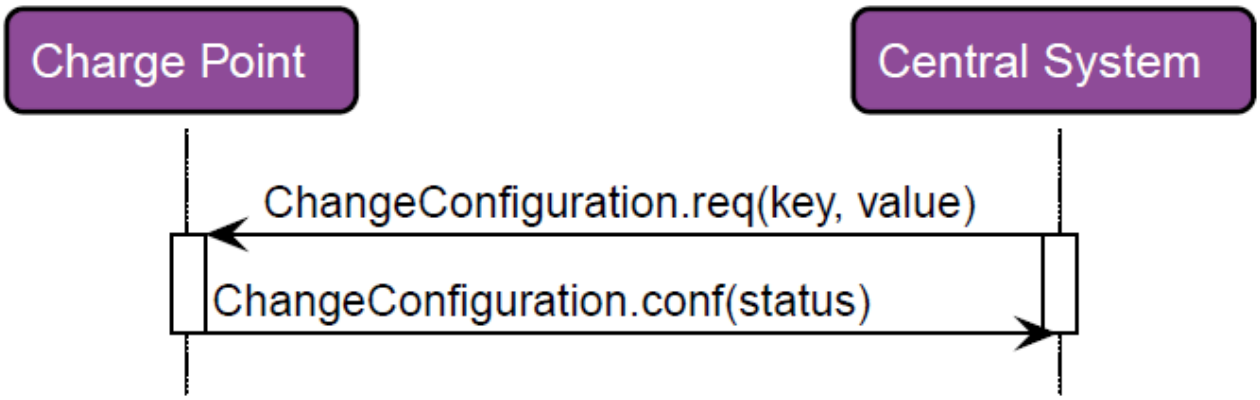
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
eType	E_OCPP1_AvailabilityType [▶ 192]	Typ der Verfügbarkeitsänderung, die der Charge Point durchführen soll.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.11 RecvChangeConfiguration

Mit dieser Methode empfängt ein OCPP-Client einen Change Configuration-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespChangeConfiguration [▶ 50] aufgerufen werden.



Syntax

```
METHOD RecvChangeConfiguration : BOOL
VAR_OUTPUT
    hMessageId : T_OCPP_MessageId;
    sKey       : T_OCPP1_ConfigKey;
    sValue     : T_OCPP1_ConfigValue;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvChangeConfiguration	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

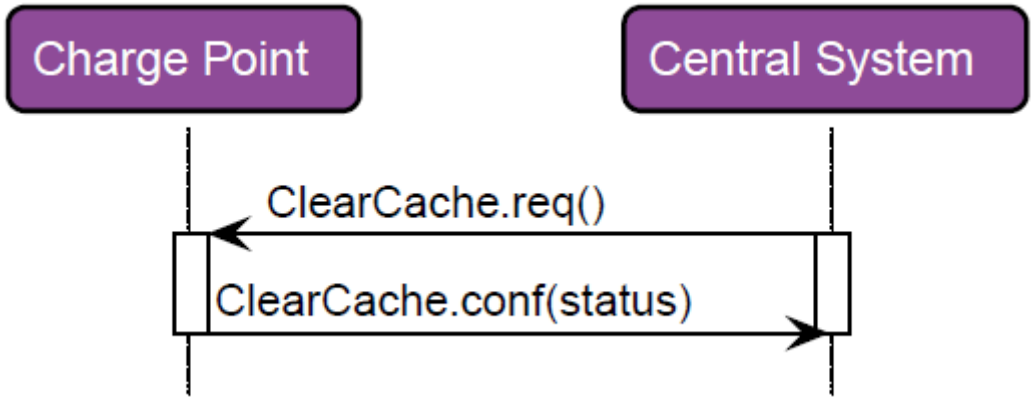
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
sKey	T_OCPP1_ConfigKey [▶ 211]	Name des zu ändernden Configuration Key.
eType	T_OCPP1_ConfigValue [▶ 211]	Neuer Wert für den Configuration Key.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.12 RecvClearCache

Mit dieser Methode empfängt ein OCPP-Client einen Clear Cache-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespClearCache [▶ 51] aufgerufen werden.



Syntax

```
METHOD RecvClearCache : BOOL
VAR_OUTPUT
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

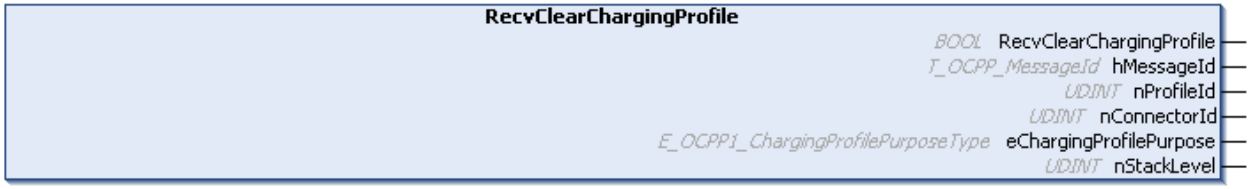
Name	Typ	Beschreibung
RecvClearCache	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ausgänge

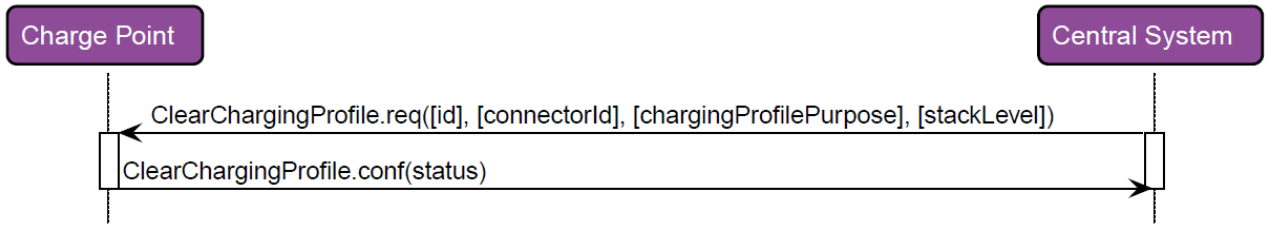
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.13 RecvClearChargingProfile



Mit dieser Methode empfängt ein OCPP-Client einen Clear Charging Profile-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespClearChargingProfile [▶ 52] aufgerufen werden.



Syntax

```
METHOD RecvClearChargingProfile : BOOL
VAR_OUTPUT
    hMessageId              : T_OCPP_MessageId;
    nProfileId              : UDINT;
    nConnectorId            : UDINT;
    eChargingProfilePurpose : E_OCPP1_ChargingProfilePurposeType;
    nStackLevel             : UDINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvClearChargingProfile	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

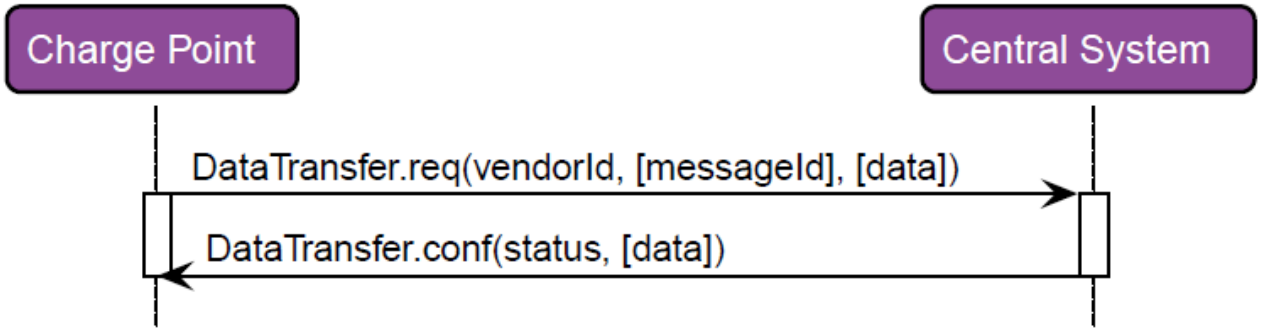
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nProfileId	UDINT	ID des zu löschenden Charging Profile.
nConnectorId	UDINT	ID des Connectors eines Charge Points, für den die Charging Profiles gelöscht werden sollen.
eChargingProfilePurpose	E_OCPP1_ChargingProfilePurposeType [▶ 193]	Spezifiziert den ChargingProfilePurposeType, dessen Charging Profiles gelöscht werden sollen.
nStackLevel	UDINT	Spezifiziert das Stack-Level, für das Charging Profile gelöscht werden sollen, wenn alle anderen Kriterien im Request erfüllt werden.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.14 RecvDataTransfer

Mit dieser Methode empfängt ein OCPP-Client einen Data Transfer-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespDataTransfer [▶ 53] aufgerufen werden.



Syntax

```
METHOD RecvDataTransfer : BOOL
VAR_OUTPUT
    hMessageId : T_OCPP_MessageId;
    sVendorId  : STRING(255);
    sMessageId : STRING(63);
END_VAR
VAR_IN_OUT
    sData      : T_OCPP1_DataTransferData;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvDataTransfer	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
sVendorId	STRING(255)	Identifier für den Hersteller, der die herstellerspezifische Implementierung kennzeichnet.
sMessageId	STRING(63)	Zusätzliches Identifikationsfeld für eine einzelne Nachricht.

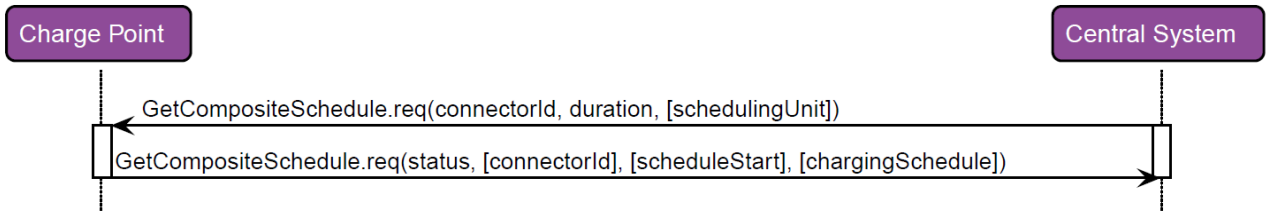
Ein-/Ausgänge

Name	Typ	Beschreibung
sData	T_OCPP1_DataTransferData [▶ 211]	Text ohne spezifizierte Länge und Format.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.15 RecvGetCompositeSchedule

Mit dieser Methode empfängt ein OCPP-Client einen Get Composite Schedule-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespGetCompositeSchedule [▶ 54] aufgerufen werden.



Syntax

```
METHOD RecvGetCompositeSchedule : BOOL
VAR_OUTPUT
    hMessageId        : T_OCPP_MessageId;
    nConnectorId      : UDINT;
    nDuration         : UDINT;
    eChargingRateUnit : E_OCPP1_ChargingRateUnitType;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvGetCompositeSchedule	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

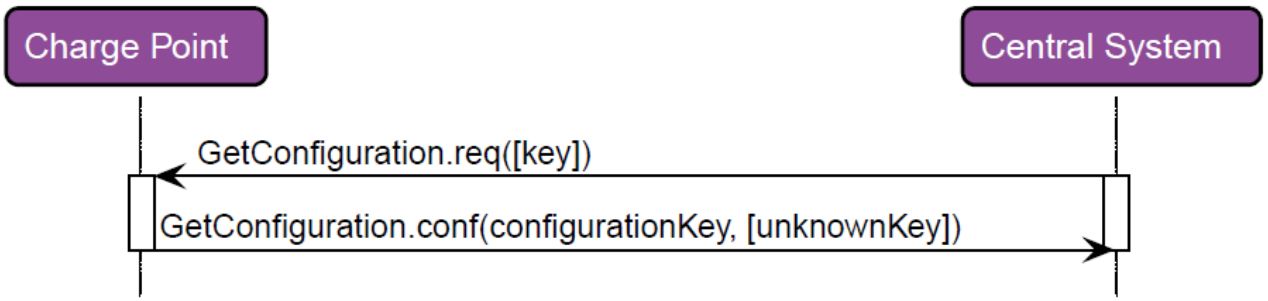
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
nDuration	UDINT	Zeit der angefragten Schedule in Sekunden.
eChargingRateUnit	E_OCPP1_ChargingRateUnitType [▶ 193]	Kann optional verwendet werden, um die Einheit der angefragten Schedule festzulegen.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.16 RecvGetConfiguration

Mit dieser Methode empfängt ein OCPP-Client einen Get Configuration-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespGetConfiguration [▶ 55] aufgerufen werden.



Syntax

```
METHOD RecvGetConfiguration : BOOL
VAR_OUTPUT
    hMessageId : T_OCPP_MessageId;
    nKeysCount : UDINT := 0;
    aKeys      : ARRAY[1..Param_OCPP.nConfigKeysMax] OF T_OCPP1_ConfigKey;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvGetConfiguration	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

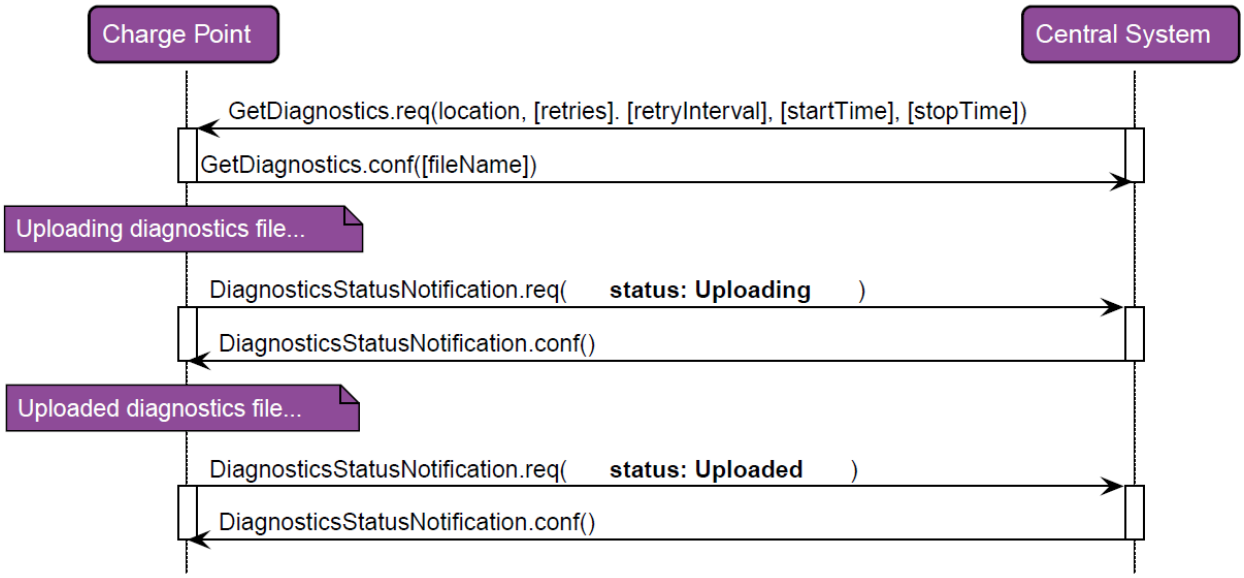
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nKeysCount	UDINT	Anzahl der in dieser Nachricht folgenden Configuration Keys.
aKeys	ARRAY[1..Param_OCPP [▶ 212].nConfigKeysMax] OF T_OCPP1_ConfigKey [▶ 211]	Liste der Configuration Keys, für die der Wert der Konfiguration angefragt wird.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.17 RecvGetDiagnostics

Mit dieser Methode empfängt ein OCPP-Client einen Get Diagnostics-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespGetDiagnostics [▶ 56] aufgerufen werden.



Syntax

```
METHOD RecvGetDiagnostics : BOOL
VAR_OUTPUT
    hMessageId     : T_OCPP_MessageId;
    sLocation      : STRING(255);
    nRetries       : UDINT;
    nRetryInterval : UDINT;
    nStartTime     : ULINT;
    nStopTime      : ULINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvGetDiagnostics	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
sLocation	STRING(255)	Enthält das Verzeichnis, in das die Diagnosedatei hochgeladen werden soll.
nRetries	UDINT	Enthält optional die Anzahl an Versuchen des Charge Points, die Datei hochzuladen.
nRetryInterval	UDINT	Enthält optional die Anzahl an Sekunden, nach denen der Upload erneut versucht wird.
nStartTime	ULINT	Markiert optional den ältesten Zeitpunkt, der in die Diagnosedatei integriert werden soll.
nStopTime	ULINT	Markiert optional den jüngsten Zeitpunkt, der in die Diagnosedatei integriert werden soll.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.18 RecvGetLocalListVersion

Mit dieser Methode empfängt ein OCPP-Client einen Get Local List Version-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespGetLocalListVersion [▶ 57] aufgerufen werden.

Syntax

```
METHOD RecvGetLocalListVersion : BOOL
VAR_OUTPUT
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvGetLocalListVersion	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

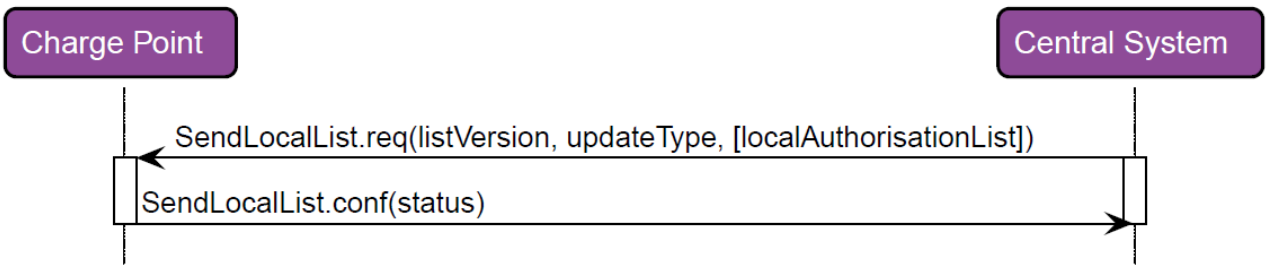
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.19 RecvLocalList

Mit dieser Methode empfängt ein OCPP-Client einen Send Local List-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespLocalList [▶ 58] aufgerufen werden.



Syntax

```
METHOD RecvLocalList : BOOL
VAR_IN_OUT
    aAuthList    : ARRAY[*] OF ST_OCPP1_AuthorizationData;
END_VAR
VAR_OUTPUT
    hMessageId   : T_OCPP_MessageId;
    nListVersion : UDINT;
    eUpdateType  : E_OCPP1_UpdateType;
    nAuthCount   : UDINT := 0;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvLocalList	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ein-/Ausgänge

Name	Typ	Beschreibung
aAuthList	ARRAY[*] OF ST_OCPP1_AuthorizationData [▶ 204]	Liste der autorisierten ID-Tags.

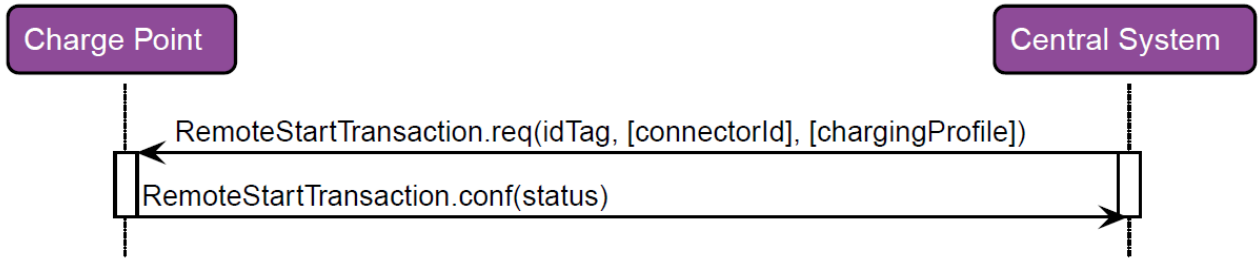
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nListVersion	UDINT	Enthält im Falle eines vollständigen Updates die Versionsnummer der neuen Liste, im Falle eines differentiellen Updates die Versionsnummer der Liste nach dem Update.
eUpdateType	E_OCPP1_UpdateType [▶ 199]	Information, ob ein vollständiges oder inkrementelles Update durchgeführt werden soll.
nAuthCount	UDINT	Anzahl an folgenden Einträgen in der Local Authorization List.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.20 RecvRemoteStartTransaction

Mit dieser Methode empfängt ein OCPP-Client einen Remote Start Transaction-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespRemoteStartTransaction [▶ 59] aufgerufen werden.



Syntax

```
METHOD RecvRemoteStartTransaction : BOOL
VAR_OUTPUT
    hMessageId   : T_OCPP_MessageId;
    nConnectorId : UDINT;
    sIdToken     : T_OCPP1_IdToken;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvRemoteStartTransaction	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

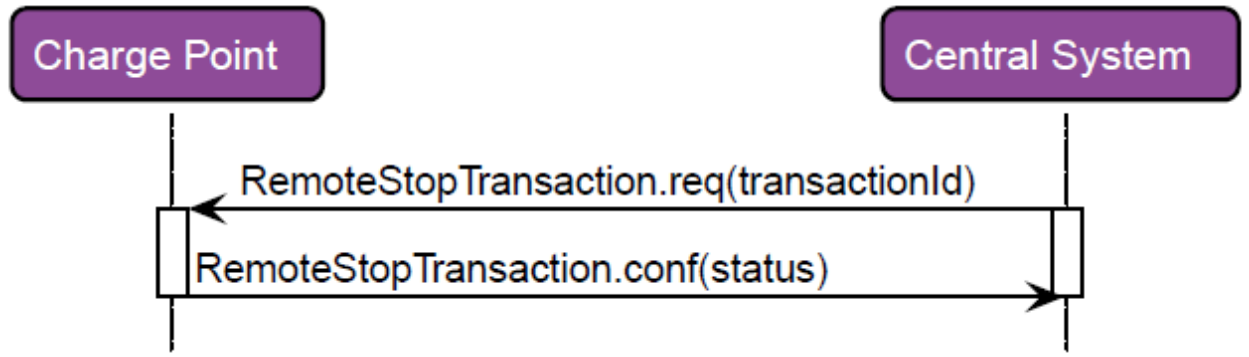
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
sIdToken	T_OCPP1_IdToken [▶ 212]	ID-Token des Charge Points im Central System.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.21 RecvRemoteStopTransaction

Mit dieser Methode empfängt ein OCPP-Client einen Remote Stop Transaction-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespRemoteStopTransaction [▶ 60] aufgerufen werden.



Syntax

```
METHOD RecvRemoteStopTransaction : BOOL
VAR_OUTPUT
    hMessageId   : T_OCPP_MessageId;
    nConnectorId : UDINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvRemoteStopTransaction	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

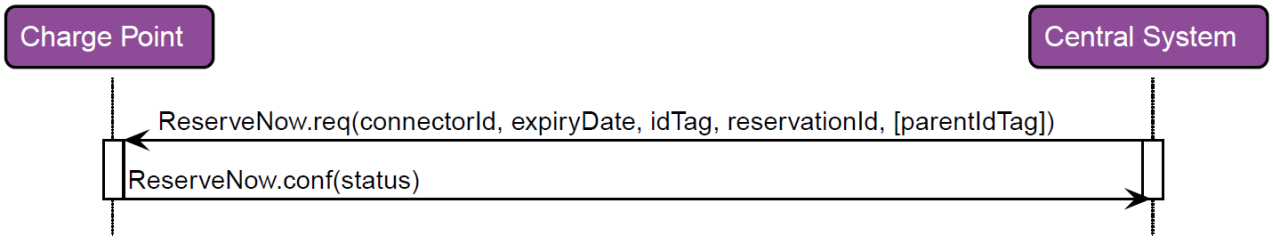
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nConnectorId	UDINT	ID des Connectors eines Charge Points. Die Zuordnung der vom Server gesendeten TransactionId zum passenden Connector wird vom OCPP-Treiber intern durchgeführt und muss vom Benutzer nicht gesondert beachtet werden.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.22 RecvReserveNow

Mit dieser Methode empfängt ein OCPP-Client einen Reserve Now-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespReserveNow [▶ 61] aufgerufen werden.



Syntax

```
METHOD RecvReserveNow : BOOL
VAR_OUTPUT
    hMessageId     : T_OCPP_MessageId;
    nConnectorId   : UDINT;
    nExpiryDate    : ULINT;
    sIdTag         : T_OCPP1_IdToken;
    sParentIdTag   : T_OCPP1_IdToken;
    nReservationId : UDINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvReserveNow	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

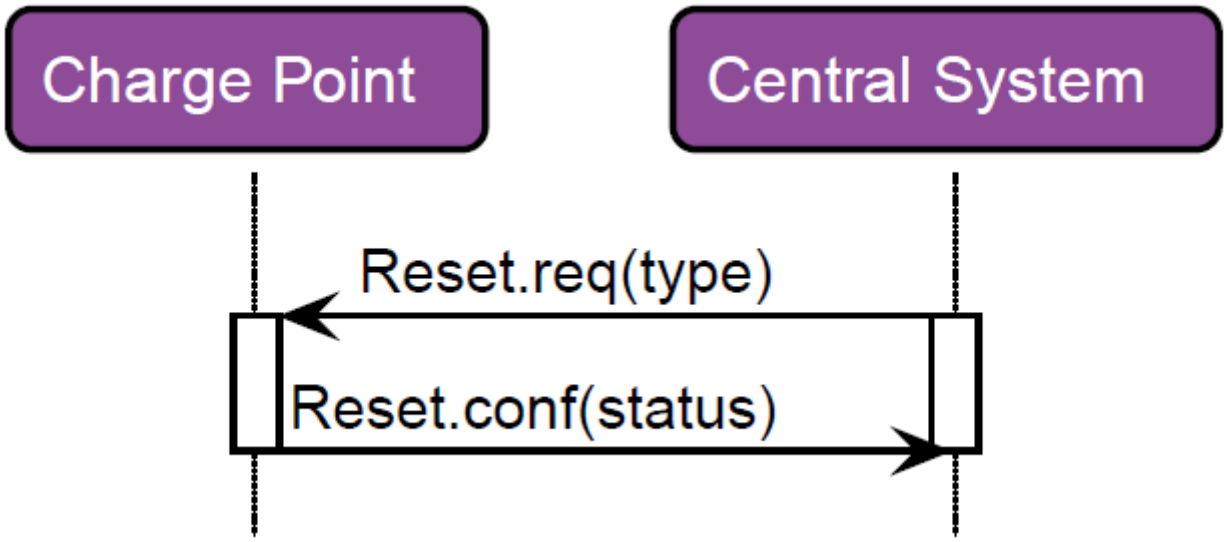
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nConnectorId	UDINT	Enthält die ID des zu reservierenden Connectors.
nExpiryDate	ULINT	Datum und Uhrzeit, an dem die Reservierung endet.
sIdTag	T_OCPP1_IdToken [▶ 212]	ID-Tag, für den der Connector reserviert werden soll.
sParentIdTag	T_OCPP1_IdToken [▶ 212]	Optionaler Parent ID-Tag.
nReservationId	UDINT	Eindeutige ID für die Reservierung.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.23 RecvReset

Mit dieser Methode empfängt ein OCPP-Client einen Reset-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespReset [▶ 62] aufgerufen werden.



Syntax

```
METHOD RecvReset : BOOL
VAR_OUTPUT
    hMessageId : T_OCPP_MessageId;
    eType : E_OCPP1_ResetType;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvReset	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

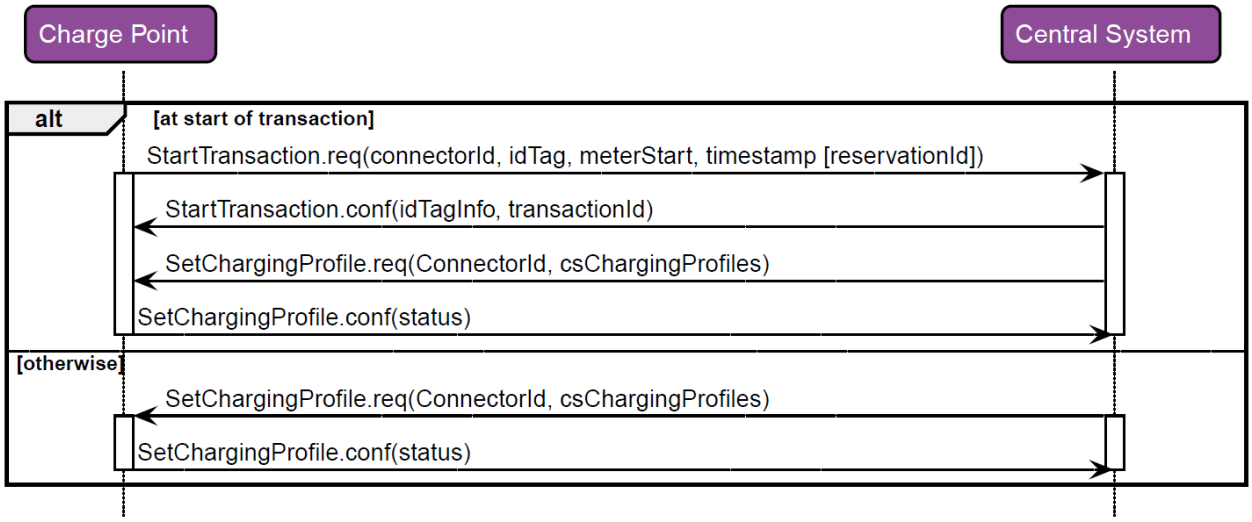
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eType	E_OCPP1_ResetType [▶ 198]	Typ des Resets, die der Charge Point durchführen soll.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.24 RecvSetChargingProfile

Mit dieser Methode empfängt ein OCPP-Client einen Set Charging Profile-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespSetChargingProfile [▶ 63] aufgerufen werden.



Syntax

```
METHOD RecvSetChargingProfile : BOOL
VAR_OUTPUT
    hMessageId       : T_OCPP_MessageId;
    nConnectorId     : UDINT;
    mChargingProfile : ST_OCPP1_ChargingProfileMax;
END_VAR
```

Rückgabewert

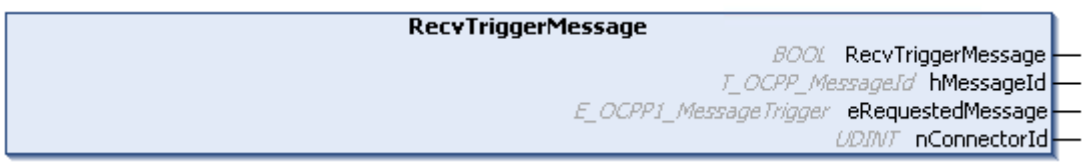
Name	Typ	Beschreibung
RecvSetChargingProfile	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ausgänge

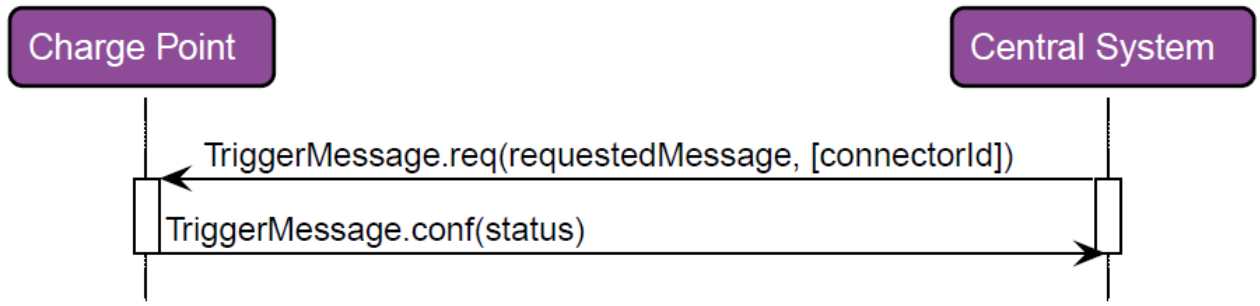
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
mChargingProfile	ST_OCPP1_ChargingProfileMax [▶ 206]	Das Charging Profile, das im Charge Point eingesetzt werden soll.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.25 RecvTriggerMessage



Mit dieser Methode empfängt ein OCPP-Client einen Trigger Message-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespTriggerMessage [▶ 64] aufgerufen werden.



Syntax

```
METHOD RecvTriggerMessage : BOOL
VAR_OUTPUT
    hMessageId        : T_OCPP_MessageId;
    eRequestedMessage : E_OCPP1_MessageTrigger;
    nConnectorId      : UDINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvTriggerMessage	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

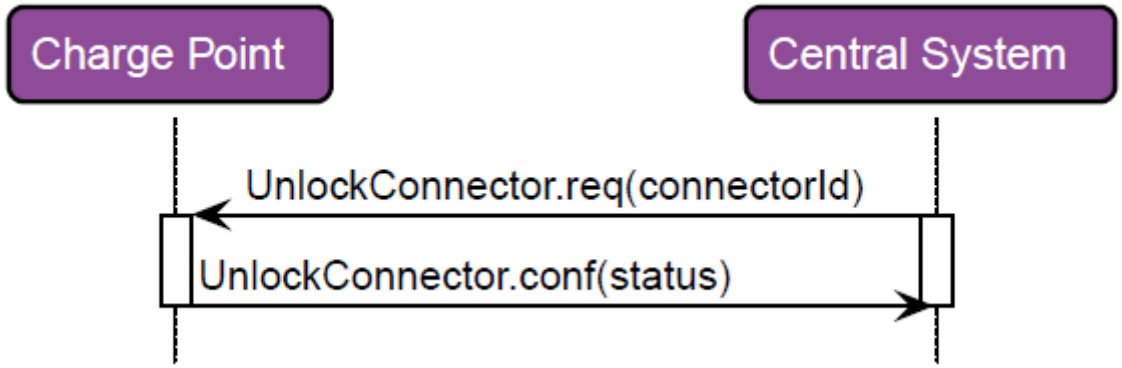
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eRequestedMessage	E_OCPP1_MessageTrigger [▶ 196]	Angefragte Nachricht, die der Charge Point an das Central System senden soll.
nConnectorId	UDINT	ID des Connectors eines Charge Points.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.26 RecvUnlockConnector

Mit dieser Methode empfängt ein OCPP-Client einen Unlock Connector-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespUnlockConnector [▶ 65] aufgerufen werden.



Syntax

```
METHOD RecvUnlockConnector : BOOL
VAR_OUTPUT
    hMessageId   : T_OCPP_MessageId;
    nConnectorId : UDINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvUnlockConnector	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nConnectorId	UDINT	ID des Connectors eines Charge Points.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.27 RecvUpdateFirmware

Mit dieser Methode empfängt ein OCPP-Client einen Update Firmware-Request vom entsprechenden OCPP-Server. Um auf den Request zu antworten, muss die Methode RespUpdateFirmware [▶ 65] aufgerufen werden.

Syntax

```
METHOD RecvUpdateFirmware : BOOL
VAR_OUTPUT
    hMessageId     : T_OCPP_MessageId;
    sLocation      : STRING(255);
    nRetries       : UDINT;
    nRetryInterval : UDINT;
    nRetrieveDate  : ULINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvUpdateFirmware	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
sLocation	STRING(255)	String mit der URI, von der die Firmware bezogen werden soll.
nRetries	UDINT	Enthält optional die Information, wie oft the Charge Point versuchen soll die Firmware herunterzuladen, bevor er aufgibt.
nRetryInterval	UDINT	Enthält optional die Information, nach welcher Zeit ein neuer Versuch unternommen werden soll.
nRetrieveDate	ULINT	Enthält die Information, zu welchem Zeitpunkt es dem Charge Point erlaubt ist, die neue Firmware zu beziehen.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.28 Reset

Mit dieser Methode wird der zuletzt am Baustein anliegende Fehler zurückgesetzt.

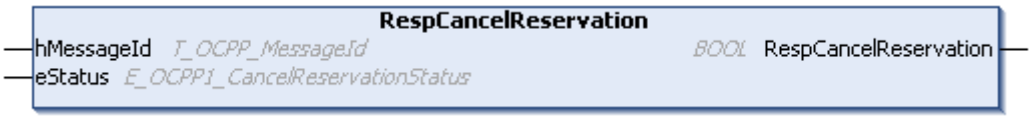
Syntax

```
METHOD Reset : BOOL
```

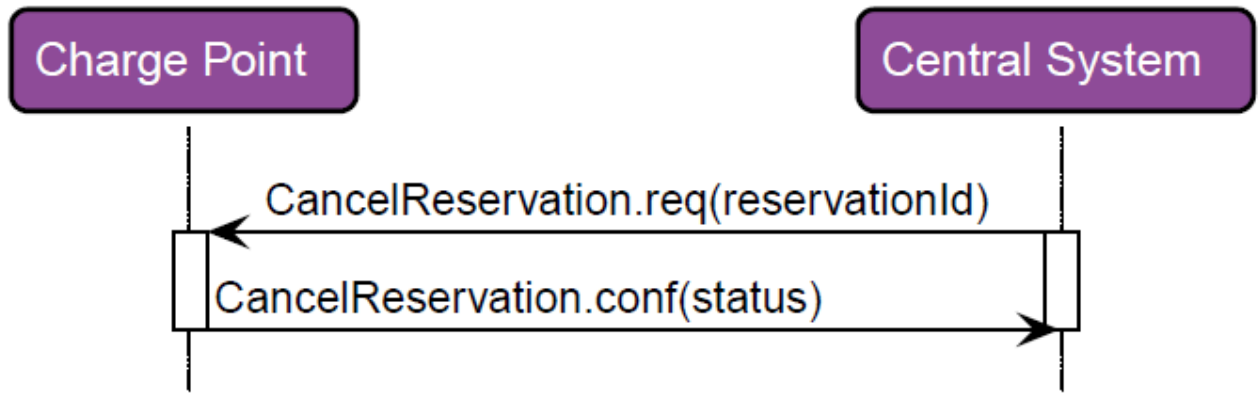
Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

5.1.1.29 RespCancelReservation



Mit dieser Methode antwortet ein OCPP-Client auf einen Cancel Reservation-Request vom entsprechenden OCPP-Server.



Syntax

```
METHOD RespCancelReservation : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_CancelReservationStatus;
END_VAR
```

Rückgabewert

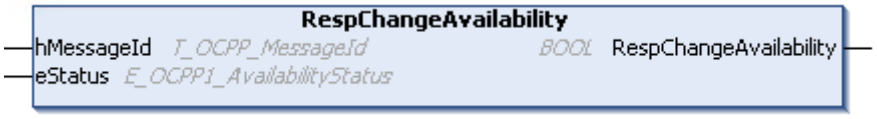
Name	Typ	Beschreibung
RespCancelReservation	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

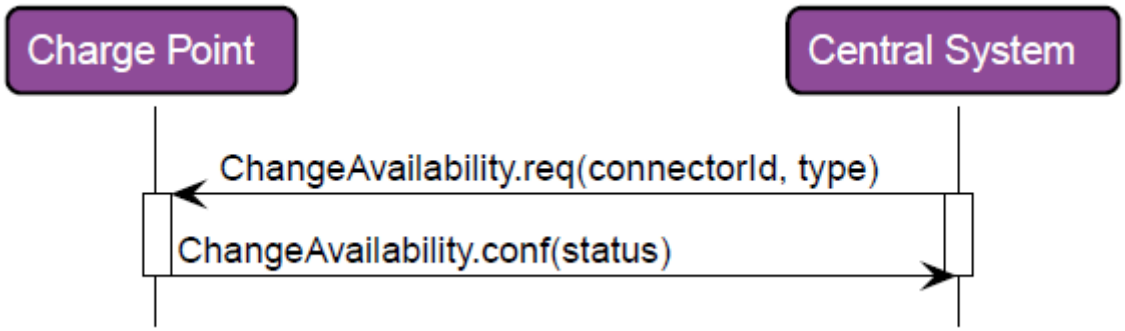
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_CancelReservationStatus [▶ 192]	Antwort, ob der Charge Point die angefragte Stornierung umsetzen konnte.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.30 RespChangeAvailability



Mit dieser Methode antwortet ein OCPP-Client auf einen Change Availability-Request vom entsprechenden OCPP-Server.



Syntax

```
METHOD RespChangeAvailability : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_AvailabilityStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RespChangeAvailability	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

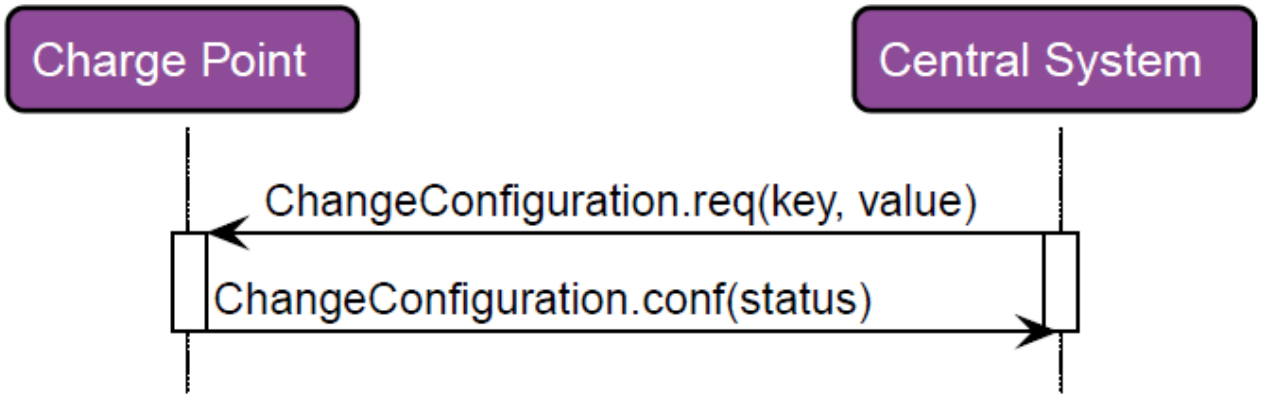
Eingänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_AvailabilityStatus [▶ 191]	Antwort, ob der Charge Point die angefragte Verfügbarkeitsänderung umsetzen konnte.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.31 RespChangeConfiguration

Mit dieser Methode antwortet ein OCPP-Client auf einen Change Configuration-Request vom entsprechenden OCPP-Server.



Syntax

```
METHOD RespChangeConfiguration : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_ConfigurationStatus;
END_VAR
```

Rückgabewert

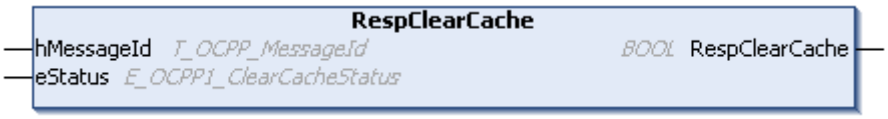
Name	Typ	Beschreibung
RespChangeAvailability	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

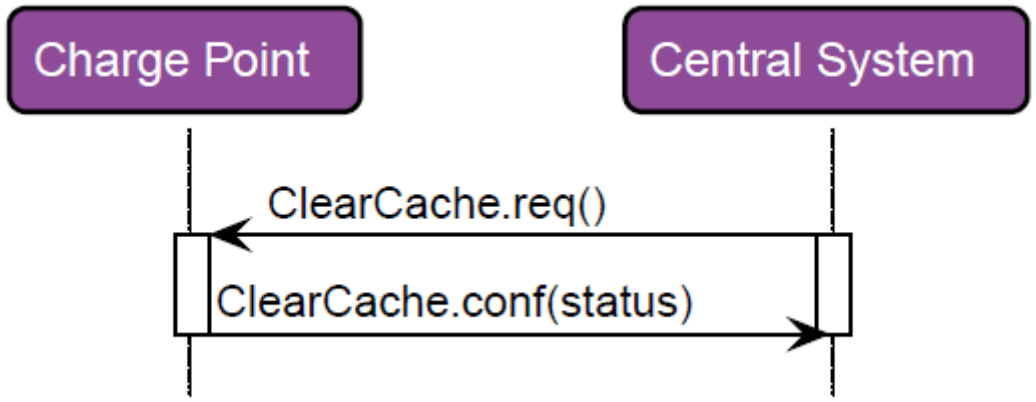
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_ConfigurationStatus [▶ 194]	Antwort, ob der Charge Point die angefragte Konfigurationsänderung umsetzen konnte.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.32 RespClearCache



Mit dieser Methode antwortet ein OCPP-Client auf einen Clear Cache-Request vom entsprechenden OCPP-Server.



Syntax

```
METHOD RespClearCache : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_ClearCacheStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RespClearCache	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

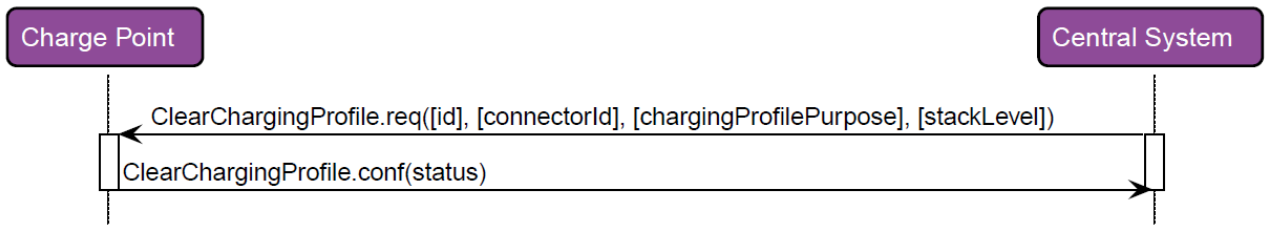
Eingänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_ClearCacheStatus [▶ 193]	Antwort, ob der Charge Point das angefrage Leeren des Caches umgesetzt hat.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.33 RespClearChargingProfile

Mit dieser Methode antwortet ein OCPP-Client auf einen Clear Charging Profile-Request vom entsprechenden OCPP-Server.



Syntax

```
METHOD RespClearChargingProfile : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    :  E_OCPP1_ClearChargingProfileStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RespClearChargingProfile	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

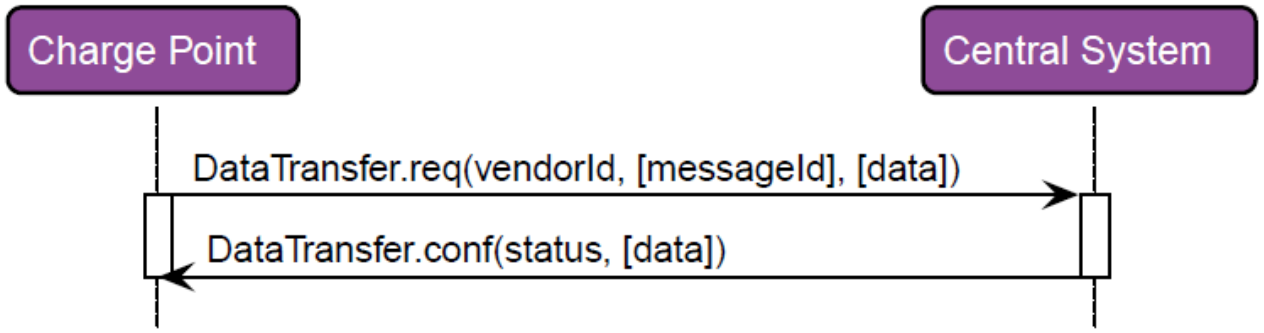
Eingänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_ClearChargingProfileStatus [▶ 193]	Antwort, ob der Charge Point das angefragte Löschen umsetzen konnte.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.34 RespDataTransfer

Mit dieser Methode antwortet ein OCPP-Client auf einen Data Transfer-Request vom entsprechenden OCPP-Server.



Syntax

```
METHOD RespDataTransfer : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_DataTransferStatus;
END_VAR
```

Rückgabewert

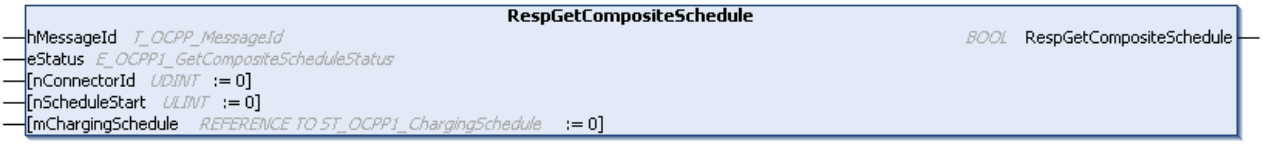
Name	Typ	Beschreibung
RespDataTransfer	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

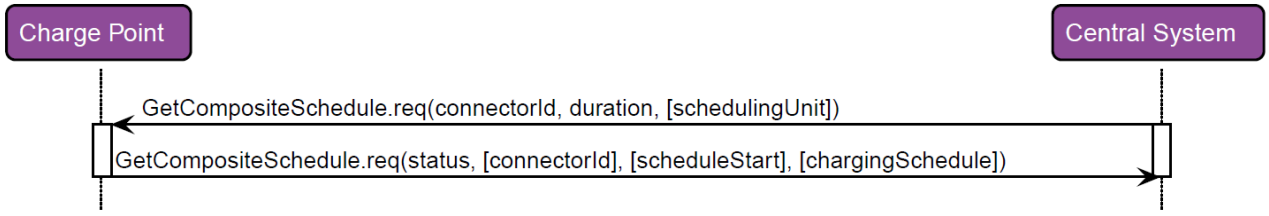
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_DataTransferStatus [▶ 194]	Antwort, ob die Data Transfer-Anfrage des OCPP-Servers erfolgreich abgeschlossen werden konnte.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.35 RespGetCompositeSchedule



Mit dieser Methode antwortet ein OCPP-Client auf einen Get Composite Schedule-Request vom entsprechenden OCPP-Server.



Syntax

```
METHOD RespGetCompositeSchedule : BOOL
VAR_INPUT
    hMessageId        : T_OCPP_MessageId;
    eStatus           : E_OCPP1_GetCompositeScheduleStatus;
    nConnectorId      : UDINT := 0;
    nScheduleStart    : ULINT := 0;
    mChargingSchedule : REFERENCE TO ST_OCPP1_ChargingSchedule REF= 0;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RespGetCompositeSchedule	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

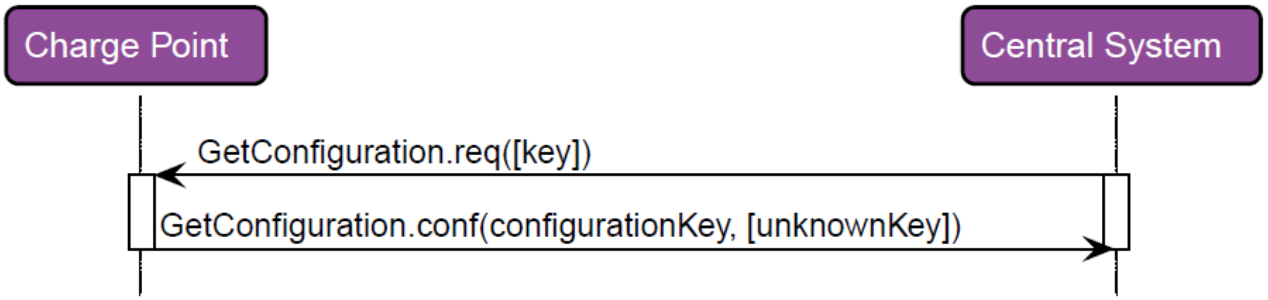
Eingänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_GetCompositeScheduleStatus [▶ 195]	Antwort, ob das Abfragen der Schedule auf Seite des Charge Points durchgeführt werden konnte.
nConnectorId	UDINT	Kann optional die ID eines Connectors eines Charge Points enthalten. Für diesen Connector würde dann die zurückgelieferte Schedule gelten.
nScheduleStart	ULINT	Enthält optional die Zeit, alle hier enthaltenen Werte werden relativ zu dieser Zeit dargestellt.
mChargingSchedule	REFERENCE TO ST_OCPP1_ChargingSchedule [▶ 207]	Enthält optional die geplante Schedule über die Zeit.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.36 RespGetConfiguration

Mit dieser Methode antwortet ein OCPP-Client auf einen Get Configuration-Request vom entsprechenden OCPP-Server.



Syntax

```
METHOD RespGetConfiguration : BOOL
VAR_INPUT
    hMessageId    : T_OCPP_MessageId;
    nKeysCount    : UDINT := 0;
    nUnknownCount : UDINT := 0;
END_VAR
VAR_IN_OUT CONSTANT
    aKeys         : ARRAY[*] OF ST_OCPP1_ConfigKeyValue;
    aUnknown      : ARRAY[*] OF T_OCPP1_ConfigKey;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RespGetConfiguration	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nKeysCount	UDINT	Anzahl der folgenden Configuration Keys.
nUnknownCount	UDINT	Anzahl der folgenden Unknown Configuration Keys.

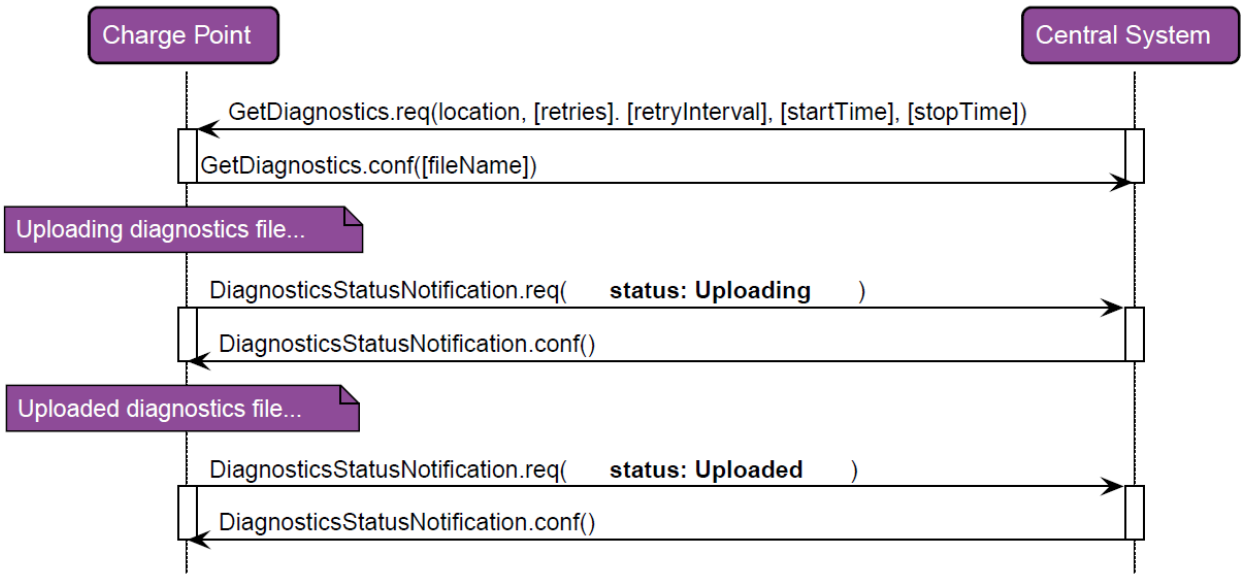
Ein-/Ausgänge

Name	Typ	Beschreibung
aKeys	ARRAY[*] OF ST_OCPP1_ConfigKeyValue [▶ 210]	Liste der angefragten oder bekannten Configuration Keys.
aUnknown	ARRAY[*] OF T_OCPP1_ConfigKey [▶ 211]	Angefragte Configuration Keys, die nicht bekannt sind.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.37 RespGetDiagnostics

Mit dieser Methode antwortet ein OCPP-Client auf einen Get Diagnostics-Request vom entsprechenden OCPP-Server.



Syntax

```
METHOD RespGetDiagnostics : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    sFileName  : STRING(255) := '';
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RespGetDiagnostics	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
sFileName	STRING(255)	Enthält optional den Dateinamen der Diagnosedatei, die hochgeladen wird.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.38 RespGetLocalListVersion

Mit dieser Methode antwortet ein OCPP-Client auf einen Get Local List Version-Request vom entsprechenden OCPP-Server.

Syntax

```
METHOD RespGetLocalListVersion : BOOL
VAR_INPUT
    hMessageId   : T_OCPP_MessageId;
    nListVersion : UDINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RespGetLocalListVersion	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

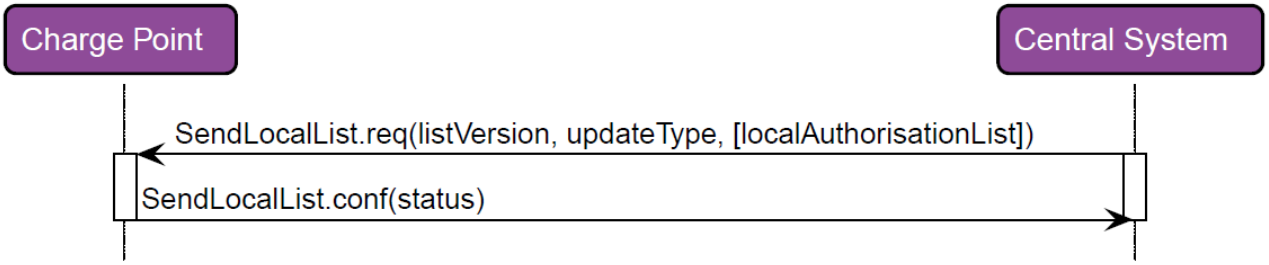
Eingänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nListVersion	UDINT	Enthält die aktuelle Versionsnummer der Local Authorization List im Charge Point.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.39 RespLocalList

Mit dieser Methode antwortet ein OCPP-Client auf einen Send Local List-Request vom entsprechenden OCPP-Server.



Syntax

```
METHOD RespLocalList : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_UpdateStatus;
END_VAR
```

Rückgabewert

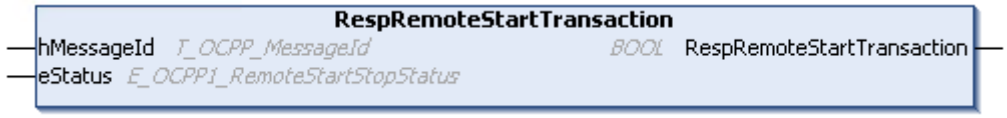
Name	Typ	Beschreibung
RespLocalList	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

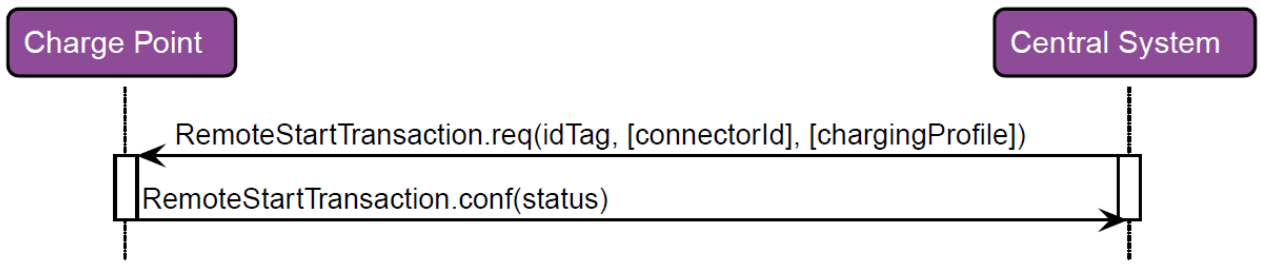
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_UpdateStatus [▶ 198]	Antwort, ob der Charge Point die gesendete Local Authorization List empfangen und das Update durchführen konnte.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.40 RespRemoteStartTransaction



Mit dieser Methode antwortet ein OCPP-Client auf einen Remote Start Transaction-Request vom entsprechenden OCPP-Server.



Syntax

```
METHOD RespRemoteStartTransaction : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_RemoteStartStopStatus;
END_VAR
```

Rückgabewert

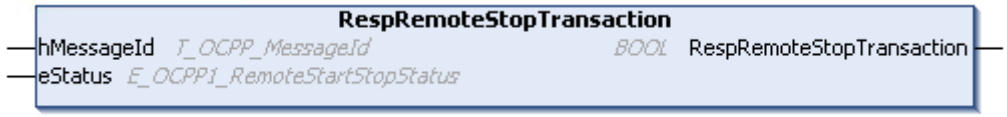
Name	Typ	Beschreibung
RespRemoteStartTransaction	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

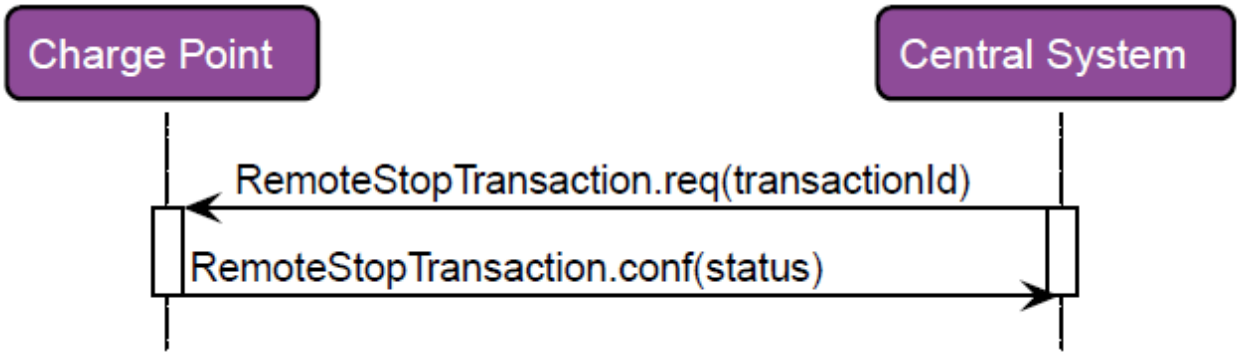
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_RemoteStartStopStatus [▶ 197]	Der Status zeigt an, ob der Charge Point den Request zum Starten einer Transaktion akzeptiert hat.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.41 RespRemoteStopTransaction



Mit dieser Methode antwortet ein OCPP-Client auf einen Remote Stop Transaction-Request vom entsprechenden OCPP-Server.



Syntax

```
METHOD RespRemoteStopTransaction : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_RemoteStartStopStatus;
END_VAR
```

Rückgabewert

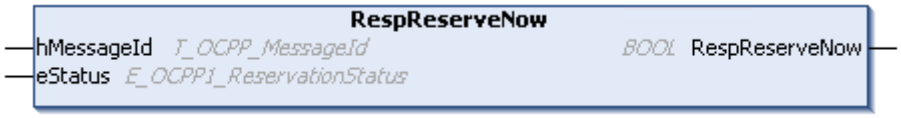
Name	Typ	Beschreibung
RespRemoteStopTransaction	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

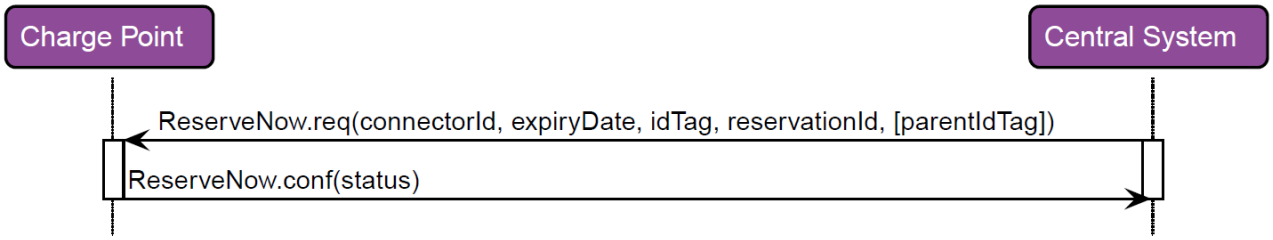
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_RemoteStartStopStatus [▶ 197]	Der Status zeigt an, ob der Charge Point den Request zum Stoppen einer Transaktion akzeptiert hat.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.42 RespReserveNow



Mit dieser Methode antwortet ein OCPP-Client auf einen Reserve Now-Request vom entsprechenden OCPP-Server.



Syntax

```
METHOD RespReserveNow : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_ReservationStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RespReserveNow	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

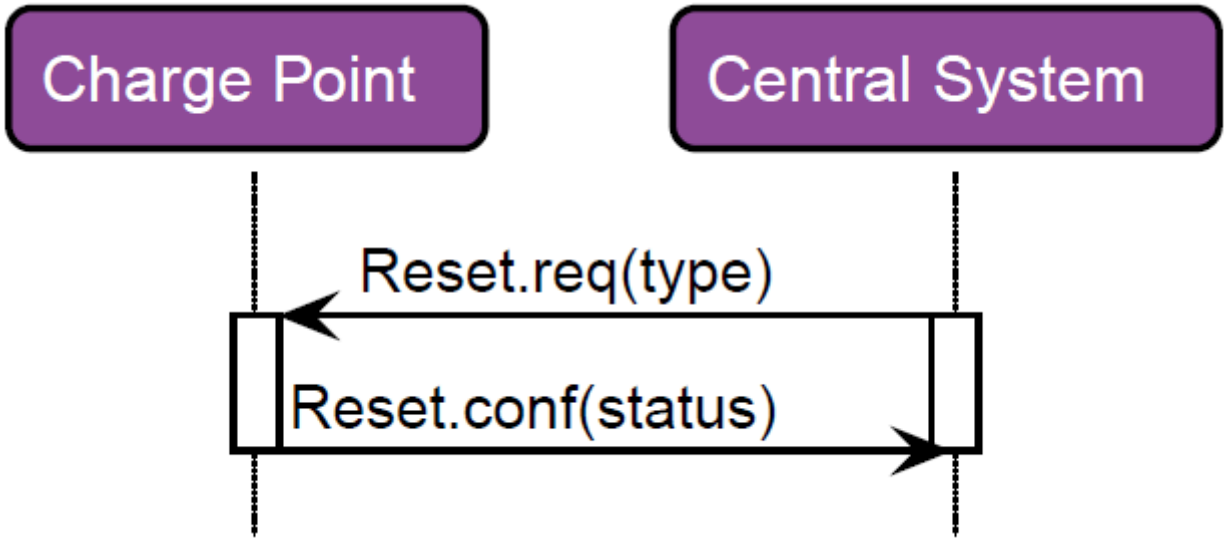
Eingänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_ReservationStatus [▶ 197]	Antwort, ob der Charge Point die angefragte Reservierung angenommen oder abgelehnt hat.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.43 RespReset

Mit dieser Methode antwortet ein OCPP-Client auf einen Reset-Request vom entsprechenden OCPP-Server.



Syntax

```
METHOD RespReset : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_ResetStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RespReset	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

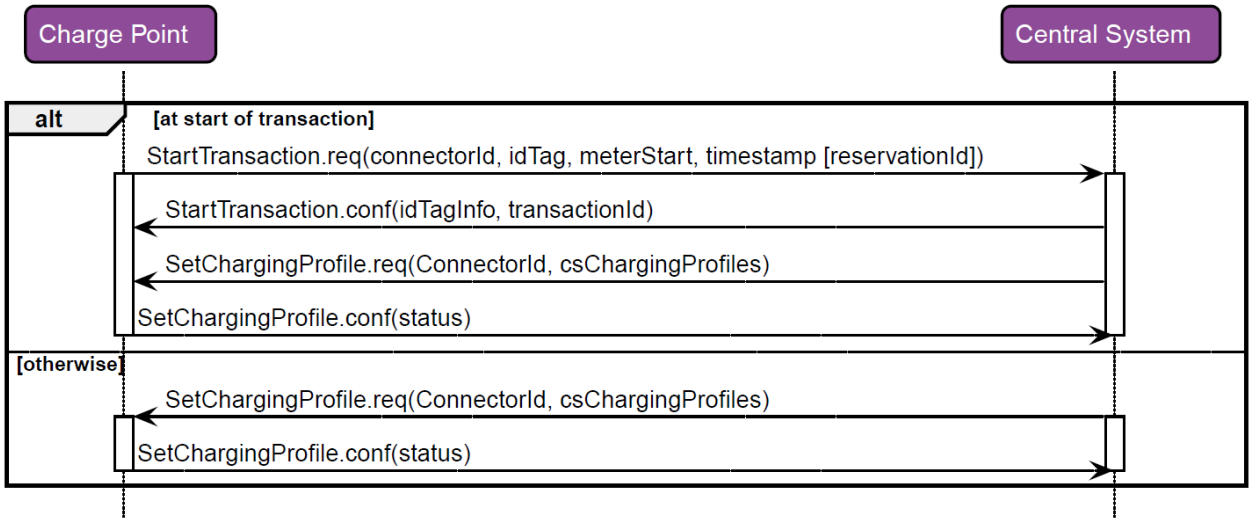
Eingänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_ResetStatus [▶ 198]	Der Status zeigt an, ob der Charge Point den Request zum Reset akzeptieren konnte.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.44 RespSetChargingProfile

Mit dieser Methode antwortet ein OCPP-Client auf einen Set Charging Profile-Request vom entsprechenden OCPP-Server.



Syntax

```
METHOD RespSetChargingProfile : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_ChargingProfileStatus;
END_VAR
```

Rückgabewert

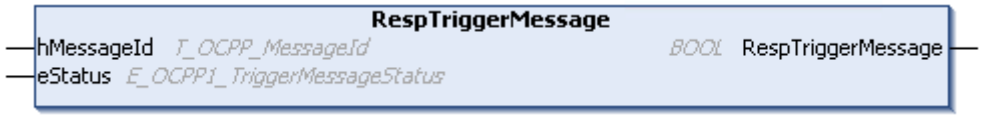
Name	Typ	Beschreibung
RespSetChargingProfile	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

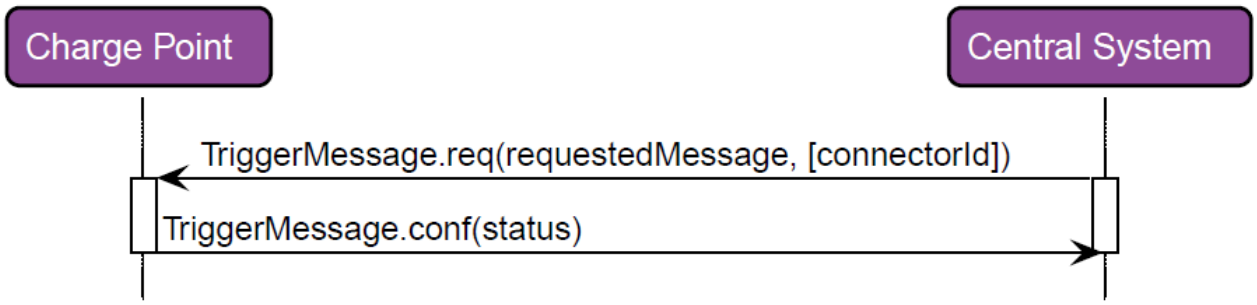
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_ChargingProfileStatus [▶ 193]	Antwort, ob der Charge Point das angefragte Charging Profile umsetzen konnte.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.45 RespTriggerMessage



Mit dieser Methode antwortet ein OCPP-Client auf einen Trigger Message-Request vom entsprechenden OCPP-Server.



Syntax

```
METHOD RespTriggerMessage : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    :E_OCPP1_TriggerMessageStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RespTriggerMessage	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

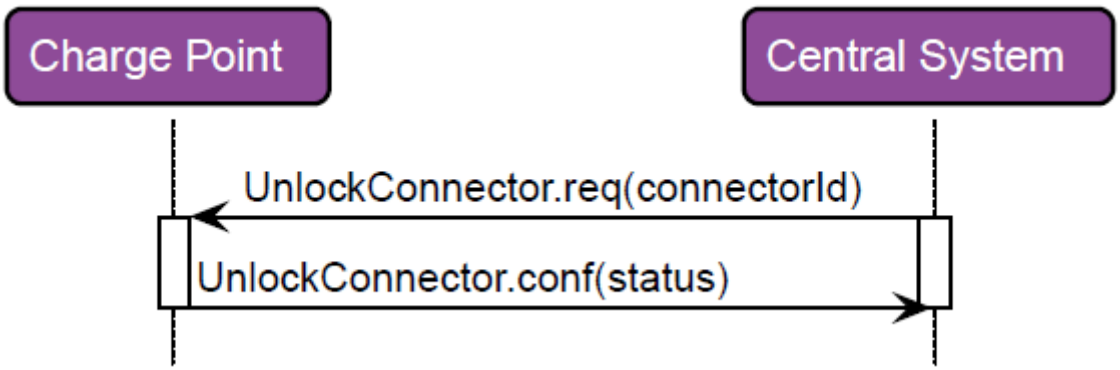
Eingänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_TriggerMessageStatus [▶ 198]	Antwort, ob der Charge Point die angefragte Nachricht senden wird oder nicht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.46 RespUnlockConnector

Mit dieser Methode antwortet ein OCPP-Client auf einen Unlock Connector-Request vom entsprechenden OCPP-Server.



Syntax

```
METHOD RespUnlockConnector : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_UnlockStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RespUnlockConnector	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_UnlockStatus [▶ 198]	Der Status zeigt an, ob der Connector freigeschaltet wurde.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.47 RespUpdateFirmware

Mit dieser Methode antwortet ein OCPP-Client auf einen Update Firmware-Request vom entsprechenden OCPP-Server.

Syntax

```
METHOD RespUpdateFirmware : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

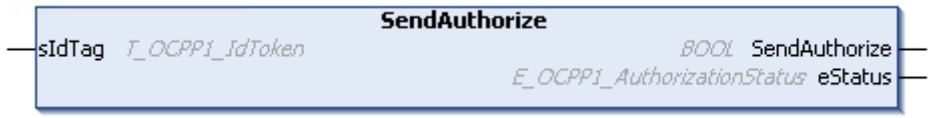
Name	Typ	Beschreibung
RespUpdateFirmware	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

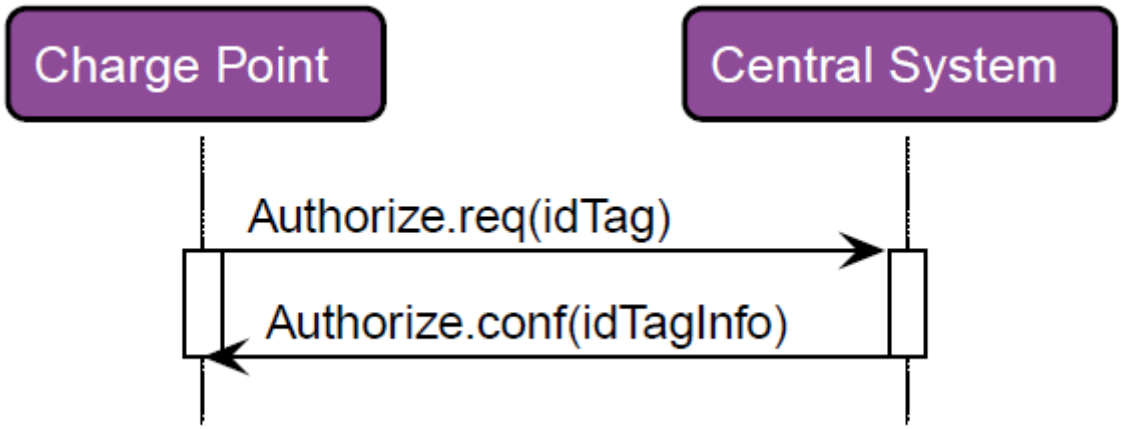
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.48 SendAuthorize



Mit dieser Methode sendet ein OCPP-Client einen Authorize-Request an den entsprechenden OCPP-Server. Die Antwort des OCPP-Servers wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendAuthorize : BOOL
VAR_INPUT
    sIdTag  : T_OCPP1_IdToken;
END_VAR
VAR_OUTPUT
    eStatus : E_OCPP1_AuthorizationStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendAuthorize	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
sIdTag	T_OCPP1_IdToken [▶ 212]	ID-Token, mit dem sich der Charge Point am Central System autorisieren lassen will.

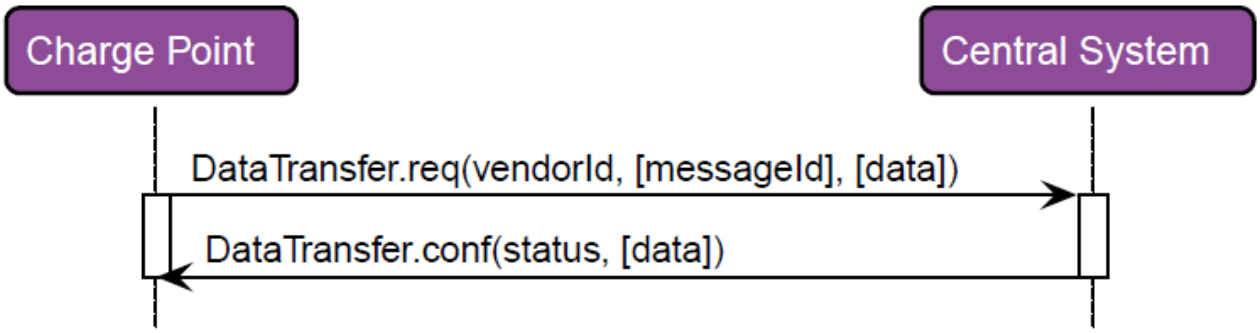
Ausgänge

Name	Typ	Beschreibung
eStatus	E_OCPP1_AuthorizationStatus [▶ 191]	Status der Autorisierung als Antwort des Central Systems.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.49 SendDataTransfer

Mit dieser Methode sendet ein OCPP-Client einen Data Transfer-Request an den entsprechenden OCPP-Server. Die Antwort des OCPP-Servers wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendDataTransfer : BOOL
VAR_INPUT
    sVendorId  : STRING(255);
    sMessageId : STRING(63) := '';
END_VAR
VAR_IN_OUT CONSTANT
    sData      : T_OCPP1_DataTransferData;
END_VAR
VAR_OUTPUT
    eStatus    : E_OCPP1_DataTransferStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendDataTransfer	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
sVendorId	STRING(255)	Identifier für den Hersteller, der die herstellerspezifische Implementierung kennzeichnet.
sMessageId	STRING(63)	Zusätzliches Identifikationsfeld für eine einzelne Nachricht.

Ein-/Ausgänge

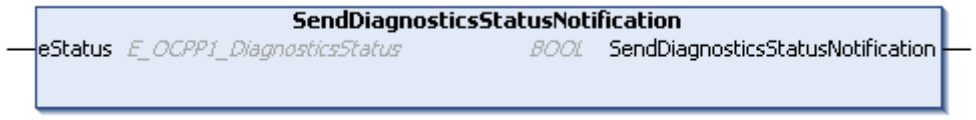
Name	Typ	Beschreibung
sData	T_OCPP1_DataTransferData [▶ 211]	Text ohne spezifizierte Länge und Format.

Ausgänge

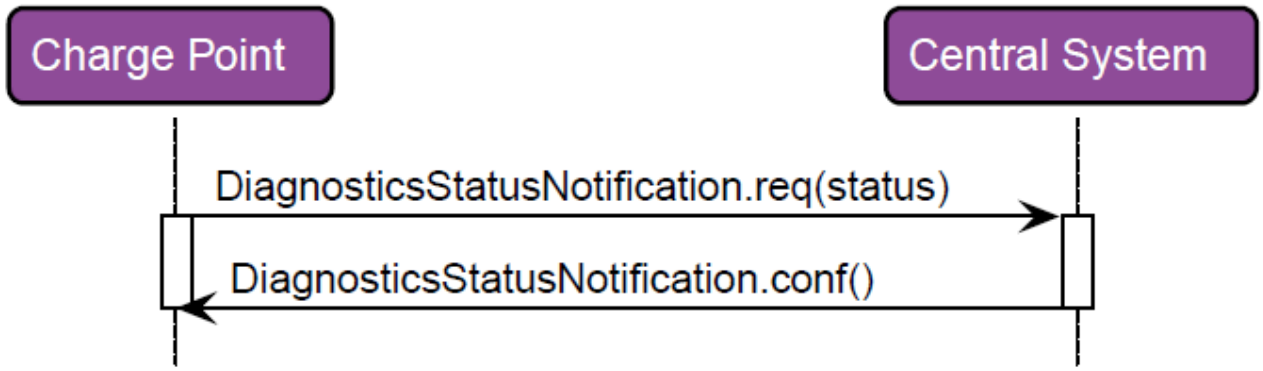
Name	Typ	Beschreibung
eStatus	E_OCPP1_DataTransferStatus [▶ 194]	Status des Data Transfers als Antwort des Central Systems.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.50 SendDiagnosticsStatusNotification



Mit dieser Methode sendet ein OCPP-Client eine Diagnostics Status Notification an den entsprechenden OCPP-Server. Die Antwort des OCPP-Servers wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendDiagnosticsStatusNotification : BOOL
VAR_INPUT
    eStatus : E_OCPP1_DiagnosticsStatus;
END_VAR
```

Rückgabewert

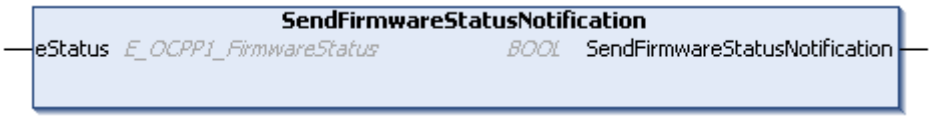
Name	Typ	Beschreibung
SendDiagnosticsStatusNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

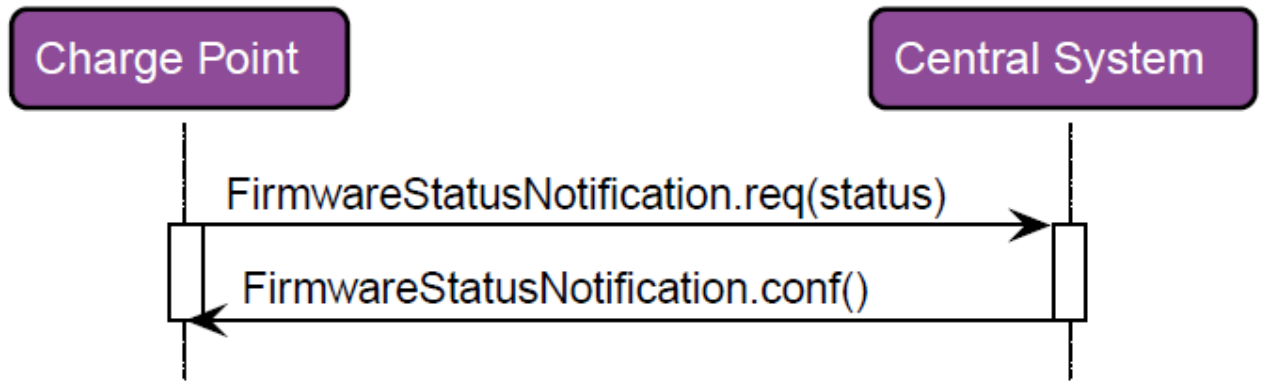
Name	Typ	Beschreibung
eStatus	E_OCPP1_DiagnosticsStatus [▶ 194]	Enthält den Status des Uploads der Diagnosedaten.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.51 SendFirmwareStatusNotification



Mit dieser Methode sendet ein OCPP-Client eine Firmware Status Notification an den entsprechenden OCPP-Server. Die Antwort des OCPP-Servers wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendFirmwareStatusNotification : BOOL
VAR_INPUT
    eStatus : E_OCPP1_FirmwareStatus;
END_VAR
```

Rückgabewert

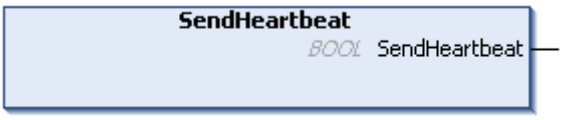
Name	Typ	Beschreibung
SendFirmwareStatusNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

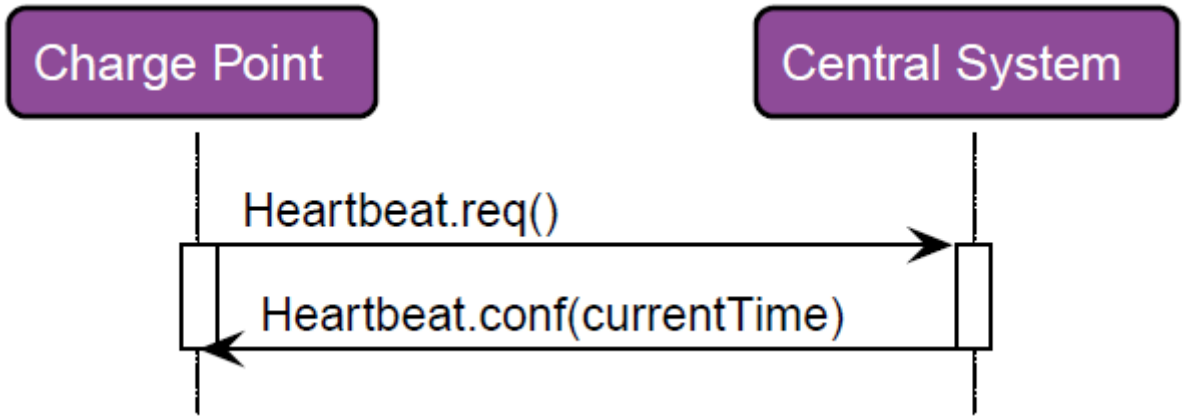
Name	Typ	Beschreibung
eStatus	E_OCPP1_FirmwareStatus [▶ 195]	Enthält den Fortschritt der Firmwareinstallation.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.52 SendHeartbeat



Mit dieser Methode sendet ein OCPP-Client einen Heartbeat an den entsprechenden OCPP-Server. Der Client schickt intern bereits einen Heartbeat, der über das Property HeartbeatInterval konfiguriert werden kann. Diese Methode würde parallel einen weiteren Heartbeat konfigurieren, sollte dies notwendig sein.



Syntax

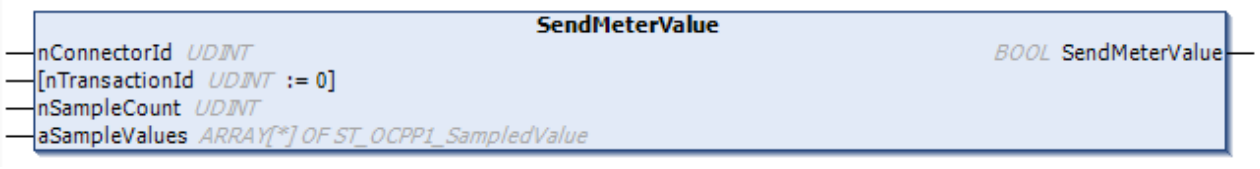
```
METHOD SendHeartbeat : BOOL
```

Rückgabewert

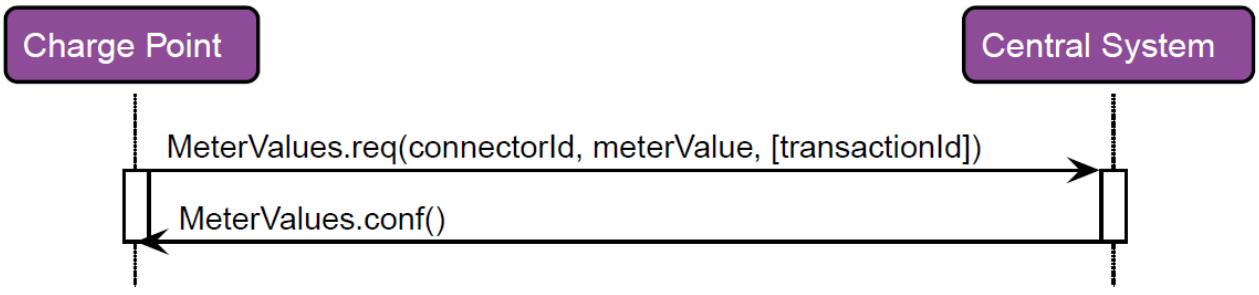
Name	Typ	Beschreibung
SendHeartbeat	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.53 SendMeterValue



Mit dieser Methode sendet ein OCPP-Client Meter Values an den entsprechenden OCPP-Server. Die Antwort des OCPP-Servers wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendMeterValue : BOOL
VAR_INPUT
    nConnectorId    : UDINT;
    nTransactionId  : UDINT := 0;
    nSampleCount    : UDINT;
END_VAR
```

```
VAR_IN_OUT
    arrSampleValues : ARRAY[*] OF ST_OCPP1_SampledValue;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendMeterValue	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
nConnectorId	UDINT	ID des Connectors eines Charge Points.
nTransactionId	UDINT	Enthält optional die ID der Transaction, zu der die angegebenen Meter Values gehören.
nSampleCount	UDINT	Die Anzahl der folgenden Messwerte.

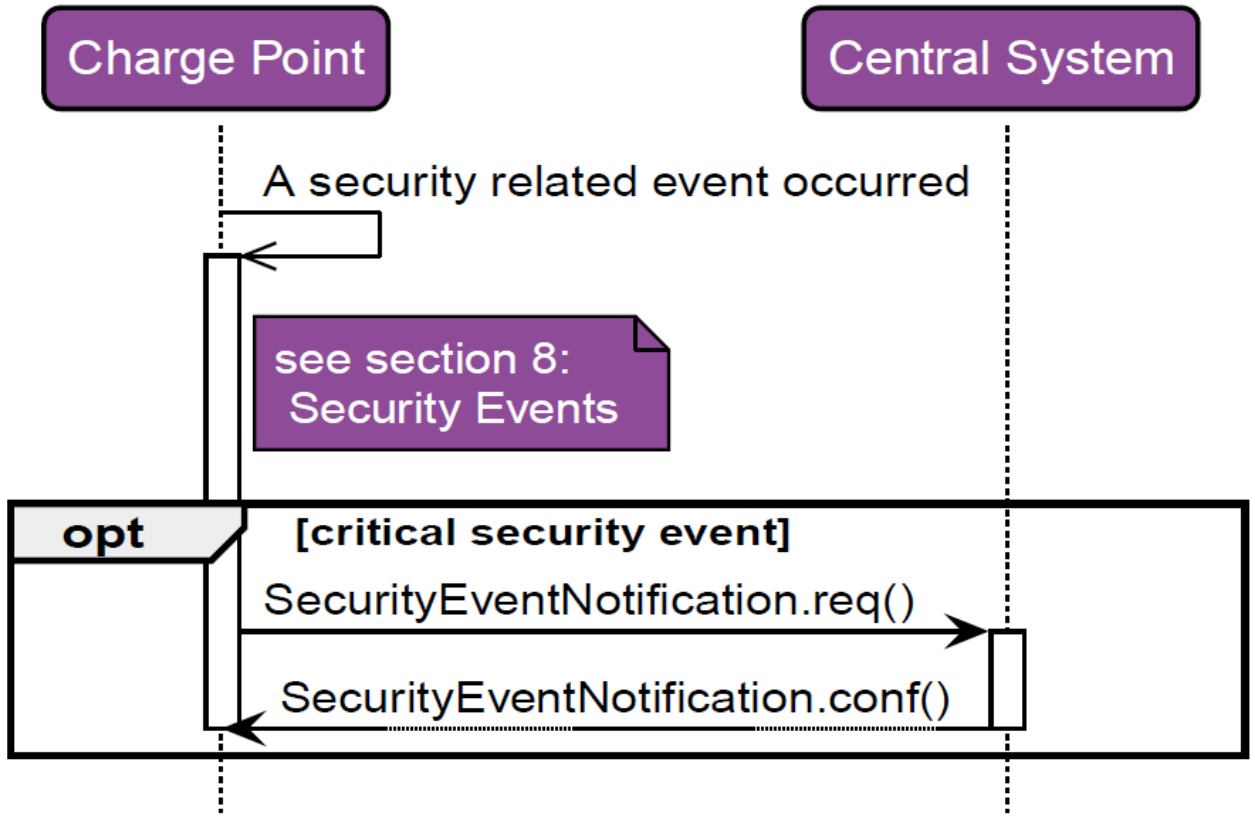
Ein-/Ausgänge

Name	Typ	Beschreibung
arrSampleValues	ARRAY [*] OF ST_OCPP1_SampledValue [▶ 210]	Die Messwerte, die an das Central System geschickt werden sollen.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.54 SendSecurityEventNotification

Mit dieser Methode sendet ein OCPP-Client eine Security Event Notification an den entsprechenden OCPP-Server. Die Antwort des OCPP-Servers wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendSecurityEventNotification : BOOL
VAR_INPUT
    nTimestamp : ULINT;
    sType      : STRING(63);
    sInfo      : STRING(255) := '';
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendDiagnosticsStatusNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

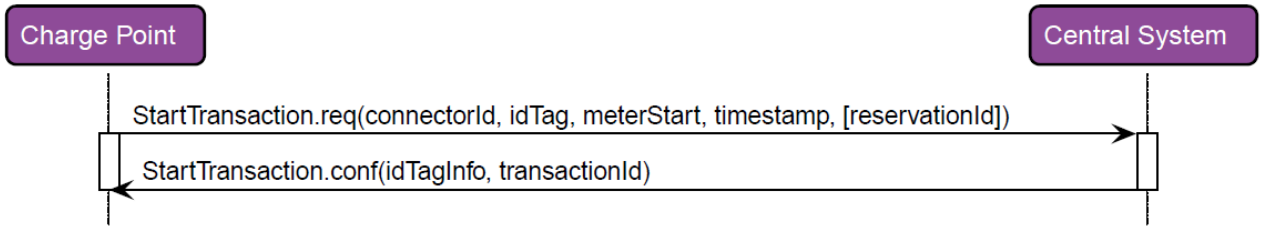
Eingänge

Name	Typ	Beschreibung
nTimestamp	ULINT	Datum und Zeit beim Auftreten des Security Events.
sType	STRING(63)	Typ des Security Events.
sInfo	STRING(255)	Enthält optional zusätzliche Informationen über das eingetretene Security Event.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.55 SendStartTransaction

Mit dieser Methode sendet ein OCPP-Client einen Start Transaction-Request an den entsprechenden OCPP-Server. Die Antwort des OCPP-Servers wird direkt innerhalb der Methode verarbeitet. Die Transaction-ID wird intern verwaltet, sodass der Benutzer nicht auf die Verwaltung achten muss.



Syntax

```
METHOD SendStartTransaction : BOOL
VAR_INPUT
    sIdTag          : T_OCPP1_IdToken;
    nConnectorId    : UDINT;
    nMeterStart     : UDINT;
    nReservationId : UDINT := 0;
END_VAR
VAR_OUTPUT
    eStatus         : E_OCPP1_AuthorizationStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendStartTransaction	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
sIdTag	T_OCPP1_IdToken [▶ 212]	ID-Token, mit dem die Transaktion gestartet werden soll.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
nMeterStart	UDINT	Wert in Wattstunden beim Start der Transaktion. Wenn der Wert 0 ist, wird der Wert nicht überprüft. Bei allen Werten größer als 0 muss dieser Wert größer oder gleich dem letzten nMeterStop-Wert sein.
nReservationId	UDINT	Optionale Reservierungs-ID.

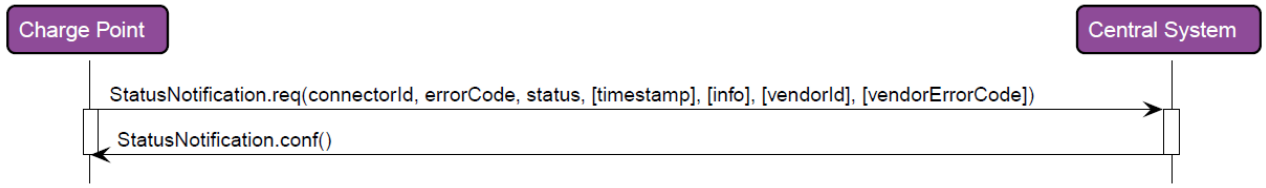
Ausgänge

Name	Typ	Beschreibung
eStatus	E_OCPP1_AuthorizationStatus [▶ 191]	Status der Autorisierung als Antwort des OCPP-Servers.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.56 SendStatusNotification

Mit dieser Methode sendet ein OCPP-Client eine Status Notification an den entsprechenden OCPP-Server. Die Antwort des OCPP-Servers wird direkt innerhalb der Methode verarbeitet, enthält in diesem Fall aber ohnehin keine gesonderten Variablen. Der Timestamp wird beim Senden des Requests intern gesetzt und kann nicht manuell bei Methodenaufruf gesetzt werden.



Syntax

```
METHOD SendStatusNotification : BOOL
VAR_INPUT
    nConnectorId : UDINT;
    eError       : E_OCPP1_ChargePointError;
    eStatus      : E_OCPP1_ChargePointStatus;
    sInfo        : STRING(50) := '';
    sVendorError : STRING(50) := '';
    sVendorId    : STRING(255) := '';
END_VAR
```

Rückgabewert

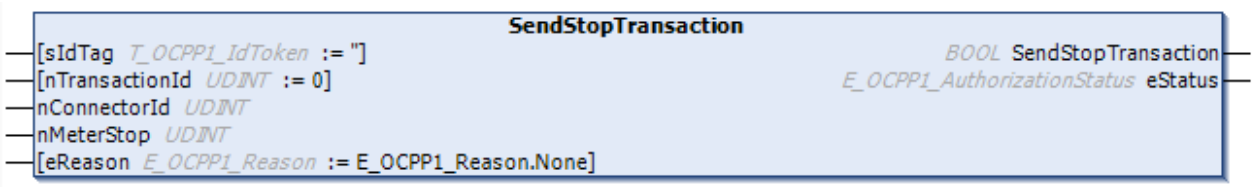
Name	Typ	Beschreibung
SendStatusNotificationEx	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

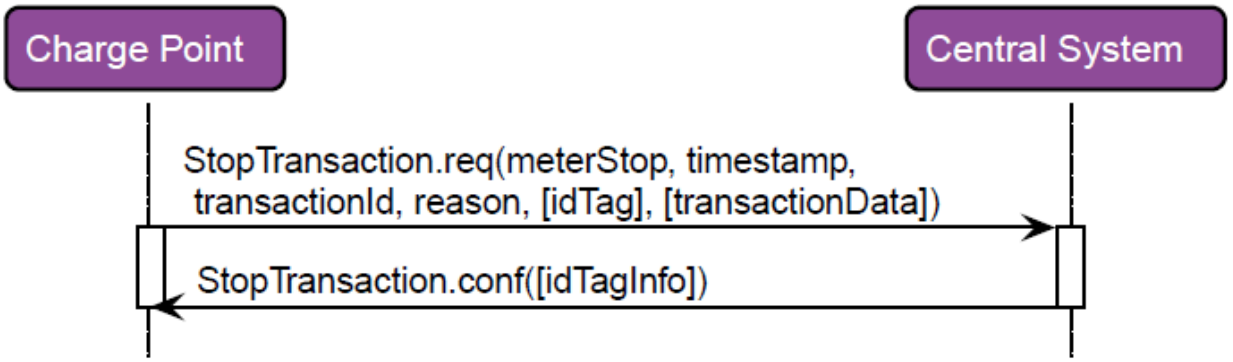
Name	Typ	Beschreibung
nConnectorId	UDINT	ID des Connectors eines Charge Points.
eError	E_OCPP1_ChargePointError [▶ 192]	Error Code, der in der Status Notification versendet werden soll.
eStatus	E_OCPP1_ChargePointStatus [▶ 192]	Status, der in der Status Notification versendet werden soll.
sInfo	STRING(50)	Enthält optional frei definierbare Zusatzinformationen zum Fehler.
sVendorError	STRING(50)	Enthält optional den herstellerspezifischen Fehlercode.
sVendorId	STRING(255)	Enthält optional den Identifier für die herstellerspezifische Implementierung.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.1.57 SendStopTransaction



Mit dieser Methode sendet ein OCPP-Client einen Stop Transaction-Request an den entsprechenden OCPP-Server. Die Antwort des OCPP-Servers wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendStopTransaction : BOOL
VAR_INPUT
    sIdTag          : T_OCPP1_IdToken := '';
    nTransactionId  : UDINT := 0;
    nConnectorId    : UDINT;
    nMeterStop      : UDINT;
    eReason         : E_OCPP1_Reason := E_OCPP1_Reason.None;
END_VAR
VAR_OUTPUT
    eStatus         : E_OCPP1_AuthorizationStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendStopTransaction	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
sIdTag	T_OCPP1_IdToken [▶ 212]	ID-Token, für das die Transaktion gestoppt werden soll.
nTransactionId	UDINT	Enthält alternativ die beim Start Transaction empfangene Transaction ID, wenn die Connector ID 0 ist.
nConnectorId	UDINT	ID des Connectors eines Charge Points, wenn die Transaction ID 0 ist.
nMeterStop	UDINT	Wert in Wattstunden am Ende der Transaktion. Dieser Wert muss größer oder gleich dem nMeterStart-Wert sein, mit dem die Transaktion gestartet wurde.
eReason	E_OCPP1_Reason [▶ 197]	Enthält optional den Grund für das Stoppen der Transaktion.

Ausgänge

Name	Typ	Beschreibung
eStatus	E_OCPP1_AuthorizationStatus [▶ 191]	Status der Autorisierung als Antwort des OCPP-Servers.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2 FB_OCPP1_Server

Dieser Baustein repräsentiert einen OCPP-Server, mit dem eine beliebige Anzahl an OCPP-Clients verbunden werden kann. Die Identifikation eines OCPP-Clients erfolgt über eine ID. Wenn eine 1-zu-1-Beziehung zwischen OCPP-Client und OCPP-Server gewünscht ist, kann anstelle dieses Funktionsbausteins der Funktionsbaustein FB_OCPP1_Station [▶ 132] verwendet werden.

Beim Senden von Anfragen an den Client werden die Send-Methoden verwendet. In diesen Methoden wird die Antwort des Clients direkt verarbeitet und in den Ausgabeparametern der Methoden hinterlegt.

Wird hingegen eine Anfrage vom Client durch eine der Receive-Methoden empfangen, muss die Antwort mittels der passenden Response-Methode gesendet werden.

Syntax

```
FUNCTION BLOCK FB_OCPP1_Server
VAR_OUTPUT
    bValid              : BOOL;
    bBusy               : BOOL;
    bError              : BOOL;
    eErrorResult        : HRESULT;
    eErrorAction        : E_OCPP1_Action;
END_VAR
```

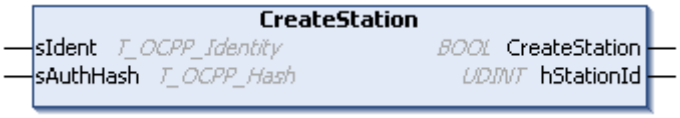
Ausgänge

Name	Typ	Beschreibung
bValid	BOOL	Die Schnittstelle zum Treiber im Hintergrund besteht.
bBusy	BOOL	Ist TRUE, solange der Baustein mit einer Bearbeitung beschäftigt ist.
bError	BOOL	Wird TRUE, sobald eine Fehlersituation auftritt.
eErrorResult	HRESULT	Zuletzt am Baustein anliegender Fehler.
eErrorAction	E_OCPP1_Action [▶ 190]	OCPP-Befehl, bei dem der Fehler aufgetreten ist.

Eigenschaften

Name	Typ	Zugriff	Beschreibung
IsOpen	BOOL	Get	Herstellung von Verbindungen ist möglich.
IsPending	BOOL	Get	Warten auf Fertigstellung der Anfrage.

5.1.2.1 CreateStation



Mit dieser Methode wird eine neue Station im OCPP-Server angelegt.

Syntax

```
METHOD CreateStation : HRESULT
VAR_INPUT
    sIdent      : T_OCPP_Identity;
    sAuthHash   : T_OCPP_Hash;
END_VAR
VAR_OUTPUT
    hStationId : UDINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
CreateStation	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
sIdent	T_OCPP_Identity [▶ 212]	Identitity des zu verbindenden OCPP-Clients.
sAuthHash	T_OCPP_Hash [▶ 212]	Hashwert für den jeweiligen OCPP-Client. Die Berechnung wird unter Hash-Berechnung [▶ 214] erläutert.

Ausgänge

hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.2 FB_Exit

Syntax

```
METHOD FB_Exit : BOOL
VAR_INPUT
    bInCopyCode : BOOL;
END_VAR
```

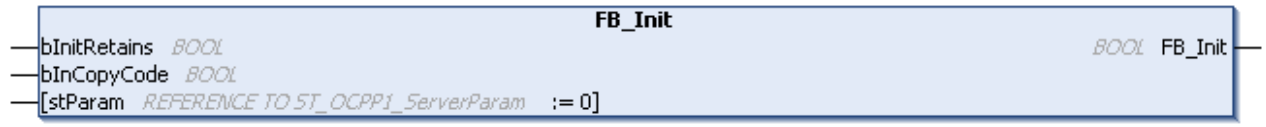
Rückgabewert

Name	Typ	Beschreibung
FB_Exit	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
bInCopyCode	BOOL	Wenn TRUE, wird die Exit-Methode zum Verlassen einer Instanz aufgerufen, die anschließend kopiert wird (Online Change).

5.1.2.3 FB_Init



Syntax

```
METHOD FB_Init : BOOL
VAR_INPUT
    bInitRetains : BOOL;
    bInCopyCode  : BOOL;
    stParam      : REFERENCE TO ST_OCPP1_ServerParam REF=0;
END_VAR
```

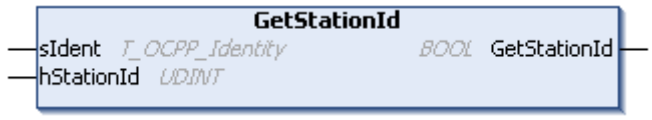
Rückgabewert

Name	Typ	Beschreibung
FB_Init	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
bInitRetains	BOOL	Wenn TRUE, dann werden die Retain-Variablen initialisiert (Warm Start/Cold Start).
bInCopyCode	BOOL	Wenn TRUE, wird die Exit-Methode zum Verlassen einer Instanz aufgerufen, die anschließend kopiert wird (Online Change).
stParam	REFERENCE TO ST_OCPP1_ServerParam [▶ 203]	Parameter für den OCPP-Server.

5.1.2.4 GetStationId



Mit dieser Methode kann die StationId eines OCPP-Clients in der Instanz des OCPP-Servers abgefragt werden.

Syntax

```
METHOD GetStationId : BOOL
VAR_INPUT
    sIdent     : T_OCPP_Identity;
END_VAR
VAR_IN_OUT
    hStationId : UDINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
GetStationId	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

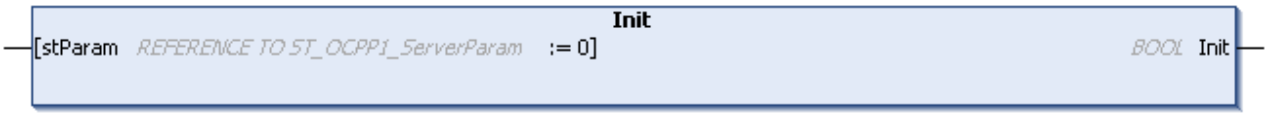
Eingänge

Name	Typ	Beschreibung
sIdent	T_OCPP_Identity [▶ 212]	Identity des OCPP-Clients.

Ein-/Ausgänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.

5.1.2.5 Init



Diese Methode wird beim Initialisieren des Funktionsbausteins in der Methode FB_Init [▶ 24] aufgerufen und spezifiziert die Parameter des WebSockets-Servers. Wenn die Verbindungsparameter im Nachhinein geändert werden sollen, kann diese Methode während des Laufens der Applikation erneut aufgerufen werden.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    stParam : REFERENCE TO ST_OCPP1_ServerParam REF=0;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
stParam	REFERENCE TO ST_OCPP1_ServerParam [▶ 203]	Verbindungsparameter für den OCPP-Server.

5.1.2.6 PollRequest

Diese Methode muss aufgerufen werden, um eingehende Requests der OCPP-Clients zu empfangen.

Syntax

```
METHOD PollRequest : BOOL
VAR_OUTPUT
    hStationId : UDINT;
    eAction    : E_OCPP1_Action;
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
PollRequest	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

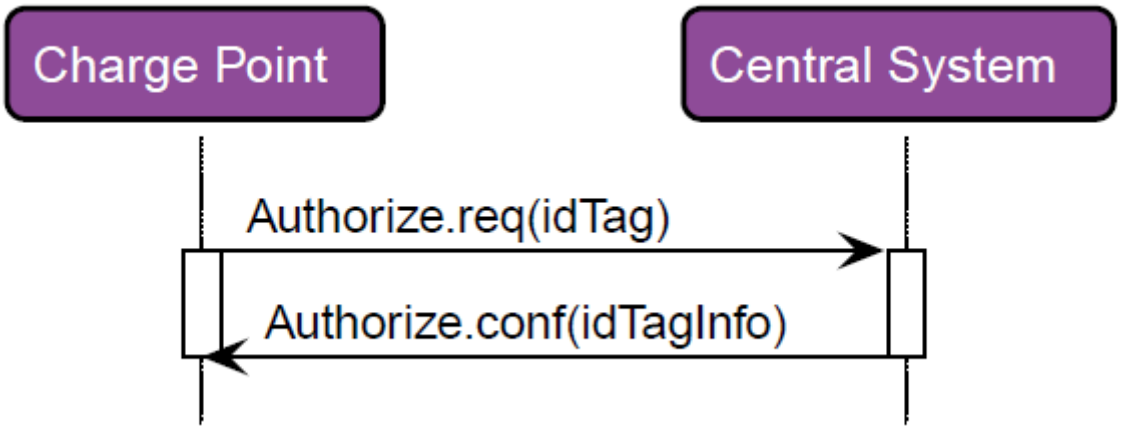
Ausgänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
eAction	E_OCPP1_Action [▶ 190]	Art des vom Server empfangenen OCPP-Requests.
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.7 RecvAuthorize

Mit dieser Methode empfängt ein OCPP-Server einen Authorize-Request von einem OCPP-Client. Um auf den Request zu antworten, muss die Methode RespAuthorize [▶ 97] aufgerufen werden.



Syntax

```
METHOD RecvAuthorize : BOOL
VAR_INPUT
    hStationId : UDINT;
END_VAR
VAR_OUTPUT
    hMessageId : T_OCPP_MessageId;
    sIdTag     : T_OCPP1_IdToken;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvAuthorize	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

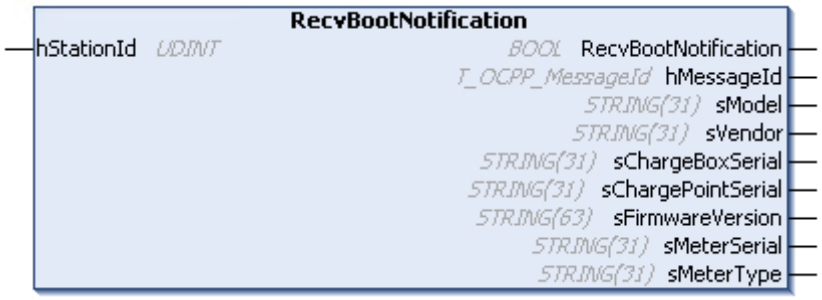
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.

Ausgänge

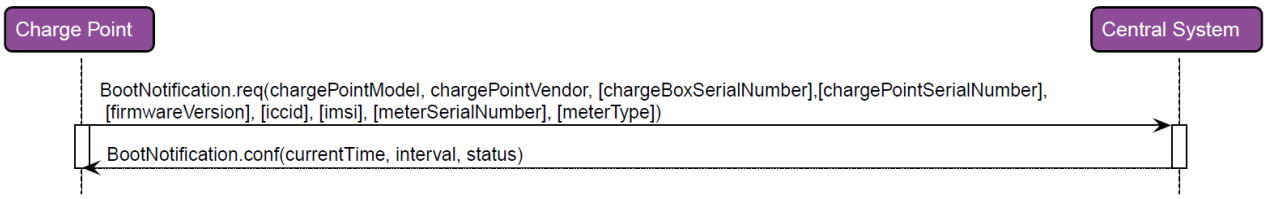
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
sIdTag	T_OCPP1_IdToken [▶ 212]	ID-Token, mit dem sich der Charge Point am Central System autorisieren lassen will.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.8 RecvBootNotification



Mit dieser Methode empfängt ein OCPP-Server eine Boot Notification von einem OCPP-Client. Um auf die Boot Notification zu antworten, muss die Methode RespBootNotification [▶ 98] aufgerufen werden.



Syntax

```
METHOD RecvBootNotification : BOOL
VAR_INPUT
    hStationId         : UDINT;
END_VAR
VAR_OUTPUT
    hMessageId         : T_OCPP_MessageId;
    sModel             : STRING(31);
    sVendor            : STRING(31);
    sChargeBoxSerial   : STRING(31);
    sChargePointSerial : STRING(31);
    sFirmwareVersion   : STRING(63);
    sMeterSerial       : STRING(31);
    sMeterType         : STRING(31);
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvBootNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.

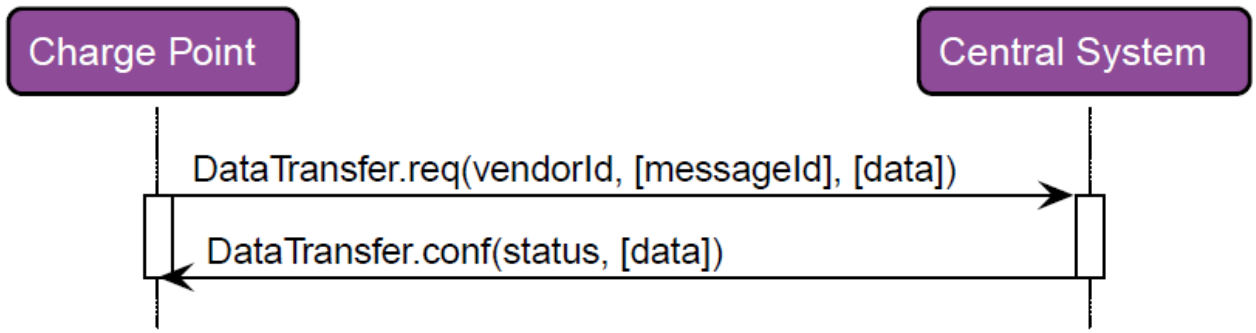
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
sModel	STRING(31)	Modell des Charge Points.
sVendor	STRING(31)	Hersteller des Charge Points.
sChargeBoxSerial	STRING(31)	Seriennummer der Charge Box innerhalb des Charge Points.
sChargePointSerial	STRING(31)	Seriennummer des Charge Points.
sFirmwareVersion	STRING(63)	Firmwareversion des Charge Points.
sMeterSerial	STRING(31)	Seriennummer des Hauptstromzählers des Charge Points.
sMeterType	STRING(31)	Typ des Hauptstromzählers des Charge Points

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.9 RecvDataTransfer

Mit dieser Methode empfängt ein OCPP-Server einen Data Transfer-Request von einem OCPP-Client. Um auf den Request zu antworten, muss die Methode RespDataTransfer [▶ 99] aufgerufen werden.



Syntax

```
METHOD RecvDataTransfer : BOOL
VAR_INPUT
    hStationId : UDINT;
END_VAR
VAR_OUTPUT
    hMessageId : T_OCPP_MessageId;
    sVendorId  : STRING(255);
    sMessageId : STRING(63);
END_VAR
VAR_IN_OUT
    sData      : T_OCPP1_DataTransferData;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvDataTransfer	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.

Ausgänge

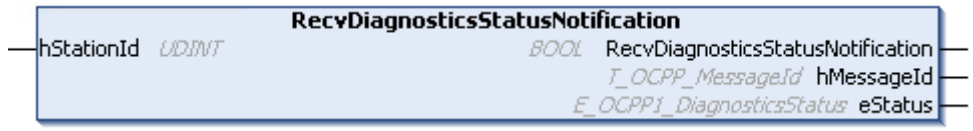
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
sVendorId	STRING(255)	Identifier für die herstellerspezifische Implementierung.
sMessageId	STRING(63)	Identifier für die einzelne Nachricht.

Ein-/Ausgänge

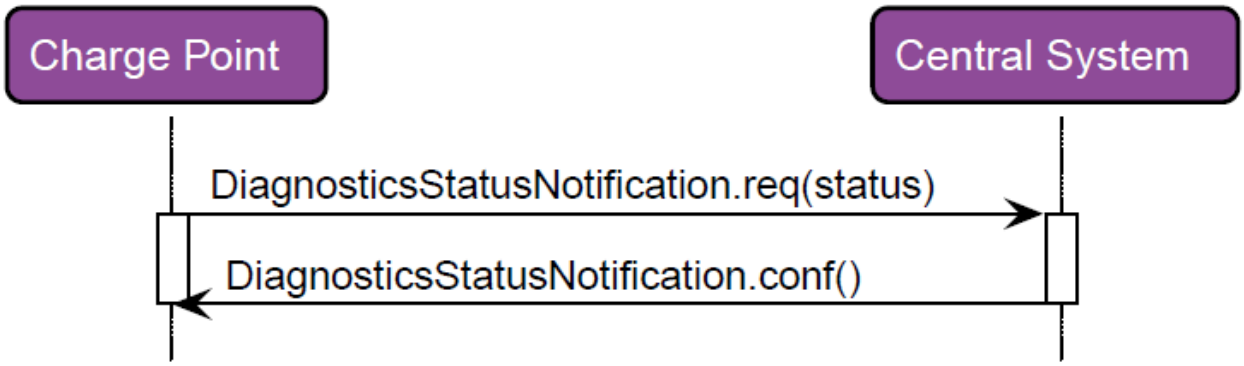
Name	Typ	Beschreibung
sData	T_OCPP1_DataTransferData [▶ 211]	Text ohne spezifizierte Länge und Format.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.10 RecvDiagnosticsStatusNotification



Mit dieser Methode empfängt ein OCPP-Server eine Diagnostics Status Notification von einem OCPP-Client. Um auf die Diagnostics Status Notification zu antworten, muss die Methode RespDiagnosticsStatusNotification [▶ 100] aufgerufen werden.



Syntax

```
METHOD RecvDiagnosticsStatusNotification : BOOL
VAR_INPUT
    hStationId : UDINT;
END_VAR
VAR_OUTPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_DiagnosticsStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvDiagnosticsStatusNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

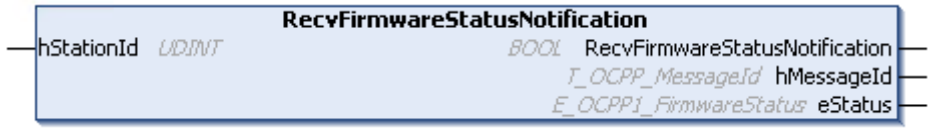
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.

Ausgänge

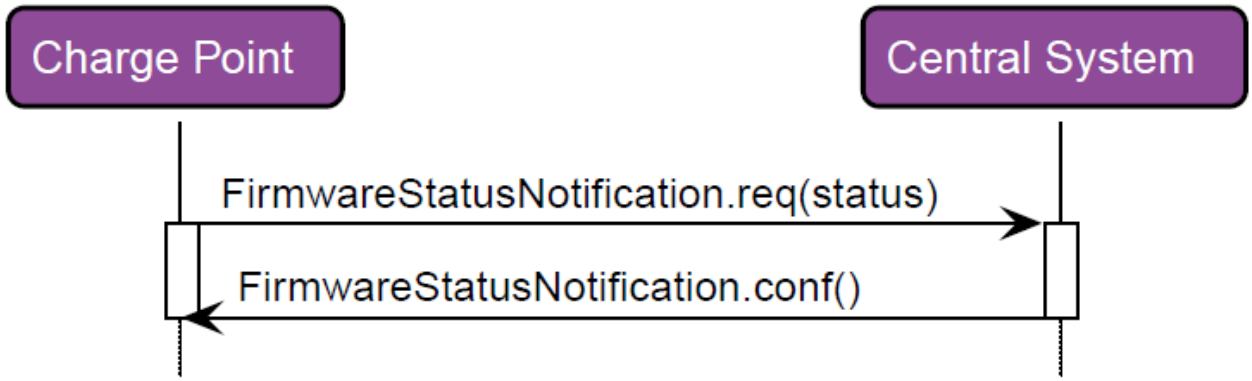
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_DiagnosticsStatus [▶ 194]	Status des Uploads der Diagnose-Daten.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.11 RecvFirmwareStatusNotification



Mit dieser Methode empfängt ein OCPP-Server eine Firmware Status Notification von einem OCPP-Client. Um auf die Firmware Status Notification zu antworten, muss die Methode RespFirmwareStatusNotification [▶ 102] aufgerufen werden.



Syntax

```
METHOD RecvFirmwareStatusNotification : BOOL
VAR_INPUT
    hStationId : UDINT;
END_VAR
VAR_OUTPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_FirmwareStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvFirmwareStatusNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

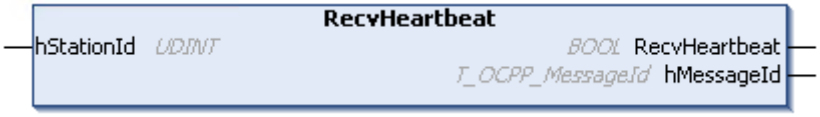
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.

Ausgänge

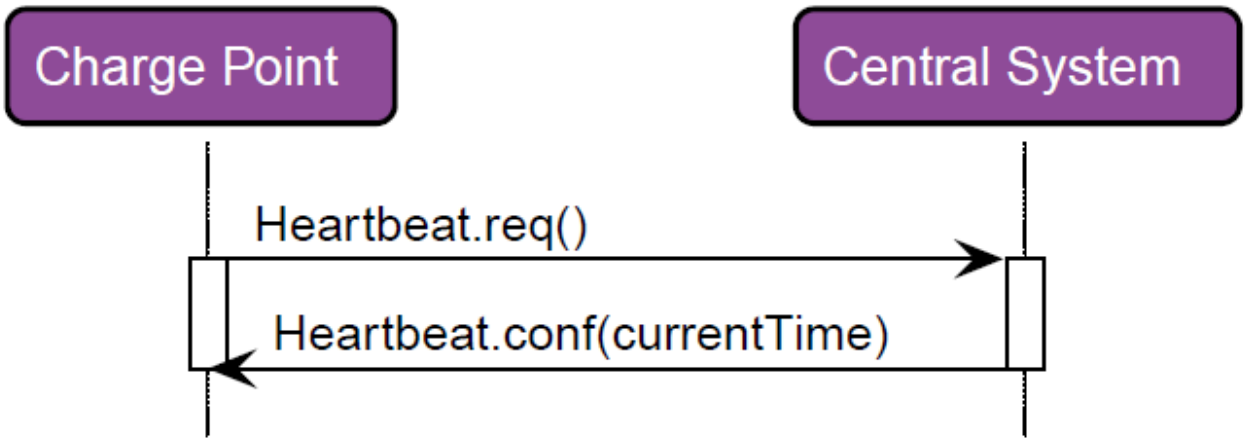
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_FirmwareStatus [▶ 195]	Status einer Firmware-Installation.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.12 RecvHeartbeat



Mit dieser Methode empfängt ein OCPP-Server einen Heartbeat von einem OCPP-Client. Um auf den Heartbeat zu antworten, muss die Methode RespHeartbeat [▶ 103] aufgerufen werden.



Syntax

```
METHOD RecvHeartbeat : BOOL
VAR_INPUT
    hStationId : UDINT;
END_VAR
VAR_OUTPUT
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvHeartbeat	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.

Ausgänge

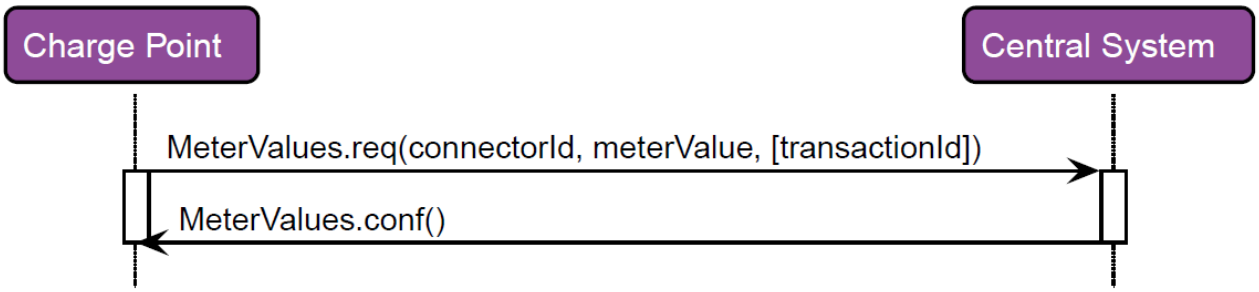
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.13 RecvMeterValue

Mit dieser Methode empfängt ein OCPP-Server Meter Values von einem OCPP-Client. Um auf den Erhalt der Meter Values zu antworten, muss die Methode RespMeterValue [▶ 104] aufgerufen werden.

Es ist entgegen der Spezifikation möglich, Meter Value-Nachrichten ohne Meter Values zu erhalten, um die Kompatibilität mit anderen Herstellern zu erhöhen.



Syntax

```
METHOD RecvMeterValue : BOOL
VAR_INPUT
    hStationId      : UDINT;
END_VAR
VAR_IN_OUT
    arrSampleValues : ARRAY[*] OF ST_OCPP1_SampledValue;
END_VAR
VAR_OUTPUT
    hMessageId      : T_OCPP_MessageId;
    nConnectorId    : UDINT;
    nTransactionId  : UDINT;
    nSampleCount    : UDINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvMeterValue	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.

Ein-/Ausgänge

Name	Typ	Beschreibung
arrSampleValues	ARRAY [*] OF ST_OCPP1_SampledValue [▶ 210]	Die vom Client gesendeten Messwerte.

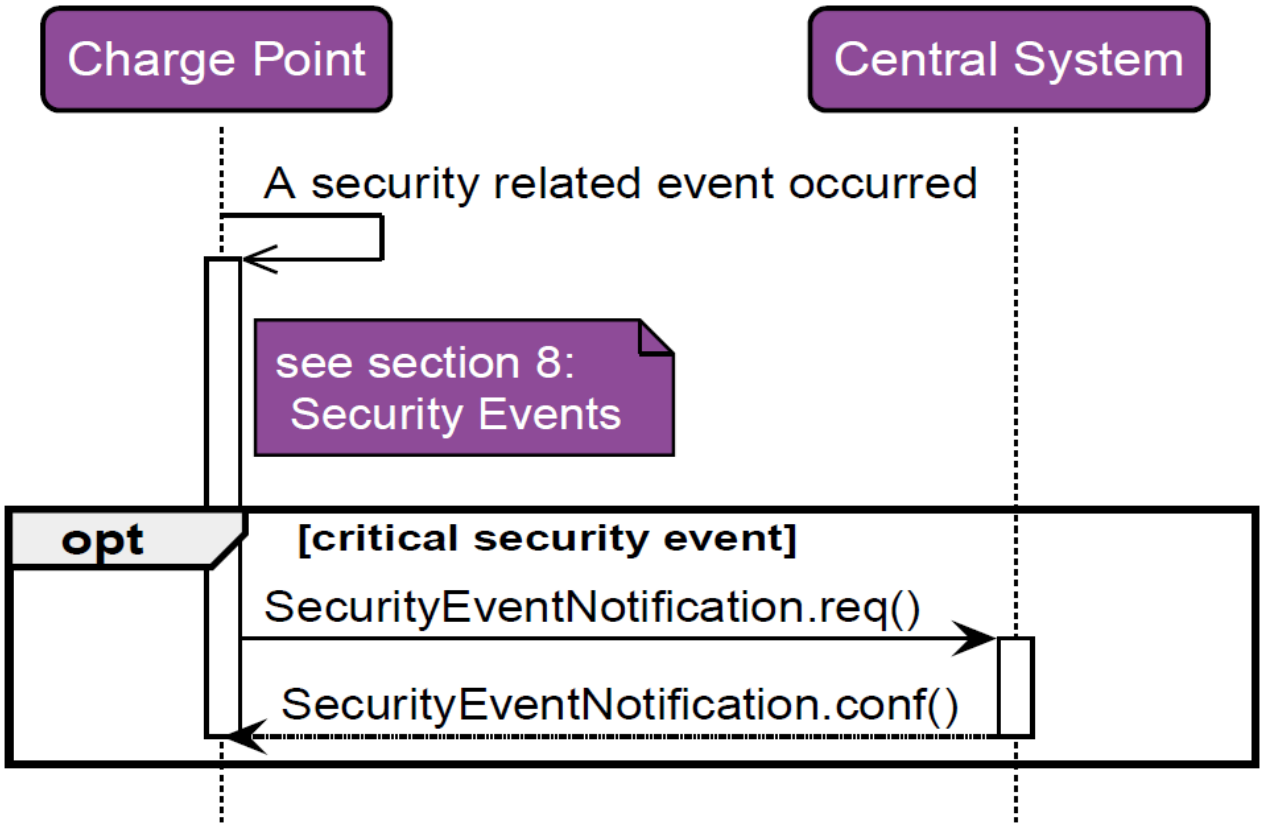
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
nTransactionId	UDINT	Die Transaktion, zu der die Messwerte gehören.
nSampleCount	UDINT	Die Anzahl der enthaltenen Messwerte.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.14 RecvSecurityEventNotification

Mit dieser Methode empfängt ein OCPP-Server eine Security Event Notification von einem OCPP-Client. Um auf die Security Event Notification zu antworten, muss die Methode RespSecurityEventNotification [▶ 105] aufgerufen werden.



Syntax

```
METHOD RecvSecurityEventNotification : BOOL
VAR_INPUT
    hStationId : UDINT;
END_VAR
VAR_OUTPUT
    hMessageId : T_OCPP_MessageId;
    nTimestamp : ULINT;
    sType      : STRING(63);
    sInfo      : STRING(256);
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvSecurityEventNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.

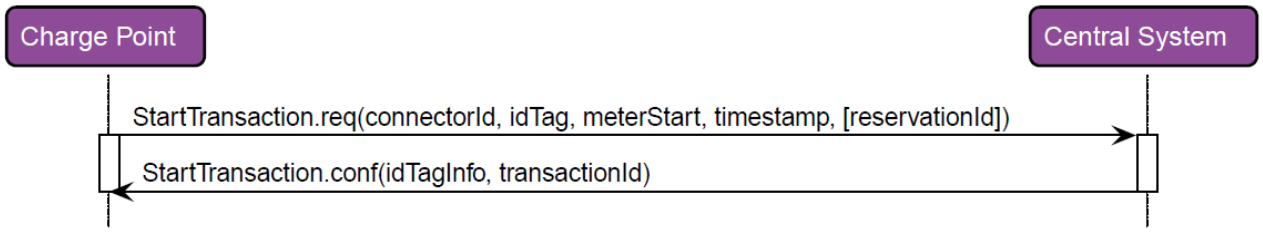
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nTimestamp	ULINT	Datum und Zeit beim Auftreten des Events.
sType	STRING(63)	Typ des Security-Events laut OCPP-Spezifikation.
sInfo	STRING(256)	Zusätzliche Informationen über das aufgetretene Security Event.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.15 RecvStartTransaction

Mit dieser Methode empfängt ein OCPP-Server einen Start Transaction-Request von einem OCPP-Client. Um auf den Request zu antworten, muss die Methode RespStartTransaction [▶ 106] aufgerufen werden.



Syntax

```
METHOD RecvStartTransaction : BOOL
VAR_INPUT
    hStationId     : UDINT;
END_VAR
VAR_OUTPUT
    hMessageId     : T_OCPP_MessageId;
    sIdTag         : T_OCPP1_IdToken;
    nConnectorId   : UDINT;
    nMeterStart    : UDINT;
    nReservationId : UDINT;
    nTimestamp     : ULINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvStartTransaction	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.

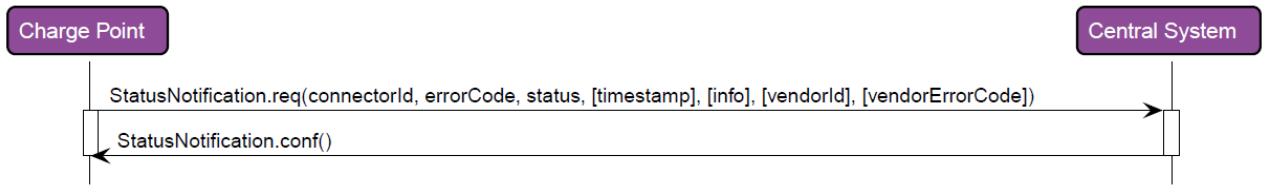
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
sIdTag	T_OCPP1_IdToken [▶ 212]	ID-Token, mit dem die Transaktion gestartet werden soll.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
nMeterStart	UDINT	Wert in Wattstunden beim Start der Transaktion.
nReservationId	UDINT	Optionale Reservierungs-ID.
nTimestamp	ULINT	Datum und Zeit beim Start der Transaktion.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.16 RecvStatusNotification

Mit dieser Methode empfängt ein OCPP-Server eine Status Notification von einem OCPP-Client. Um auf die Status Notification zu antworten, muss die Methode RespStatusNotification [▶ 107] aufgerufen werden.



Syntax

```
METHOD RecvStatusNotification : BOOL
VAR_INPUT
    hStationId   : UDINT;
END_VAR
VAR_OUTPUT
    hMessageId   : T_OCPP_MessageId;
    nConnectorId : UDINT;
    eError       : E_OCPP1_ChargePointError;
    eStatus      : E_OCPP1_ChargePointStatus;
    nTimestamp   : ULINT := 0;
    sInfo        : STRING(63) := '';
```

```
    sVendorError : STRING(63) := '';
    sVendorId    : STRING(255) := '';
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvStatusNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

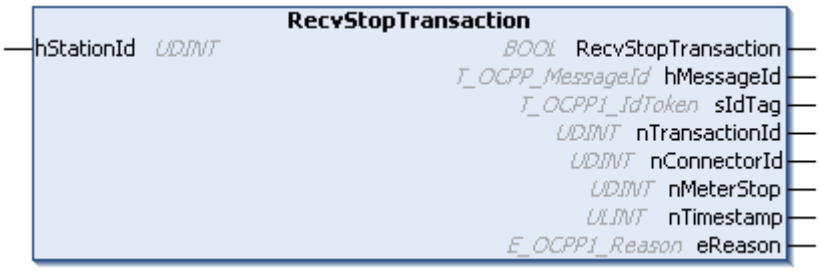
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.

Ausgänge

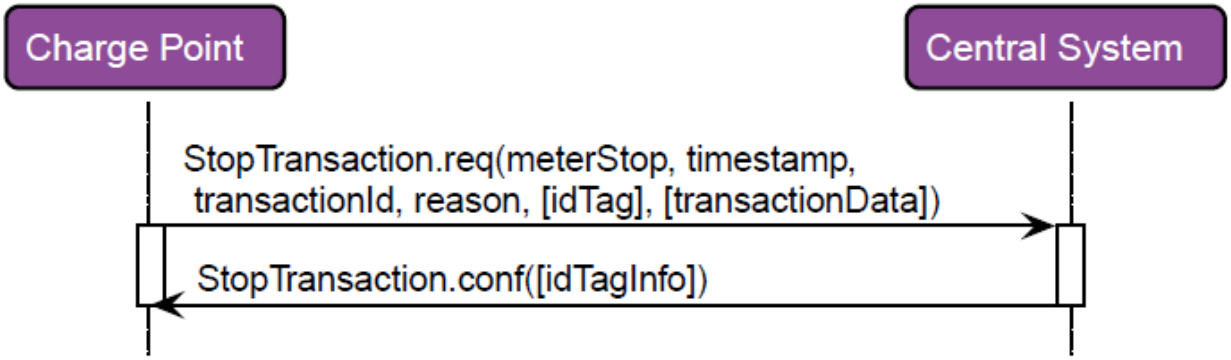
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
eError	E_OCPP1_ChargePointError [▶ 192]	Error Code, der in der Status Notification empfangen wurde.
eStatus	E_OCPP1_ChargePointStatus [▶ 192]	Status, der in der Status Notification empfangen wurde.
nTimestamp	ULINT	Der Zeitstempel der Status Notification.
sInfo	STRING(63)	Enthält optional frei definierbare Zusatzinformationen zum Fehler.
sVendorError	STRING(63)	Enthält optional den herstellerspezifischen Fehlercode.
sVendorId	STRING(255)	Enthält optional den Identifier für die herstellerspezifische Implementierung.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.17 RecvStopTransaction



Mit dieser Methode empfängt ein OCPP-Server einen Stop Transaction-Request von einem OCPP-Client. Um auf den Request zu antworten, muss die Methode RespStopTransaction [▶ 108] aufgerufen werden.



Syntax

```
METHOD RecvStopTransaction : BOOL
VAR_INPUT
    hStationId     : UDINT;
END_VAR
VAR_OUTPUT
    hMessageId     : T_OCPP_MessageId;
    sIdTag         : T_OCPP1_IdToken;
    nTransactionId : UDINT;
    nConnectorId   : UDINT;
    nMeterStop     : UDINT;
    nTimestamp     : ULINT;
    eReason        : E_OCPP1_Reason;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvStopTransaction	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

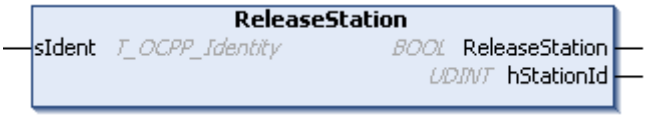
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.

Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
sIdTag	T_OCPP1_IdToken [▶ 212]	ID-Token, für das die Transaktion gestoppt werden soll.
nTransactionId	UDINT	ID der Transaktion, die gestoppt werden soll. Diese ID ist in der Antwort des Central Systems auf einen Start Transaction-Request enthalten.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
nMeterStop	UDINT	Wert in Wattstunden am Ende der Transaktion.
nTimestamp	ULINT	Datum und Zeit beim Stoppen der Transaktion.
eReason	E_OCPP1_Reason [▶ 197]	Kann optional den Grund für das Stoppen der Transaktion enthalten.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.18 ReleaseStation



Mit dieser Methode wird eine verbundene Station aus dem OCPP-Server entfernt.

Syntax

```
METHOD ReleaseStation : HRESULT
VAR_INPUT
    sIdent     : T_OCPP_Identity;
END_VAR
VAR_OUTPUT
    hStationId : UDINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
ReleaseStation	HRESULT	

Eingänge

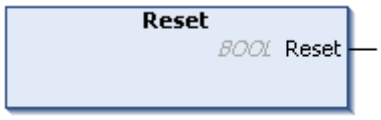
Name	Typ	Beschreibung
sIdent	T_OCPP_Identity [▶ 212]	Identitity des zu verbindenden OCPP-Clients.

Ausgänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.19 Reset



Mit dieser Methode wird der zuletzt am Baustein anliegende Fehler zurückgesetzt.

Syntax

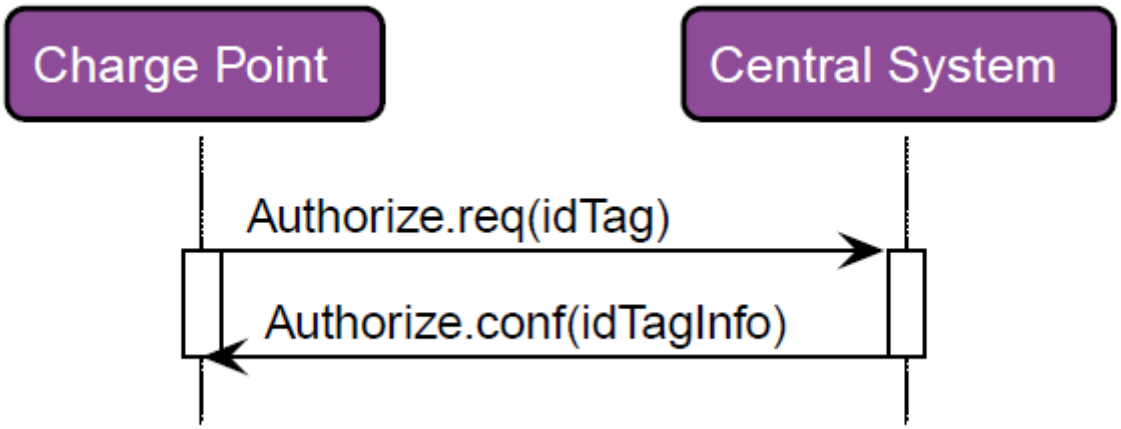
```
METHOD Reset : BOOL
```

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

5.1.2.20 RespAuthorize

Mit dieser Methode antwortet ein OCPP-Server auf einen Authorize-Request von einem OCPP-Client.



Syntax

```
METHOD RespAuthorize : BOOL
VAR_INPUT
    hStationId : UDINT;
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_AuthorizationStatus;
END_VAR
```

Rückgabewert

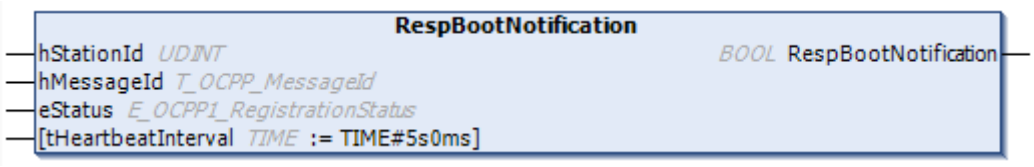
Name	Typ	Beschreibung
RespAuthorize	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

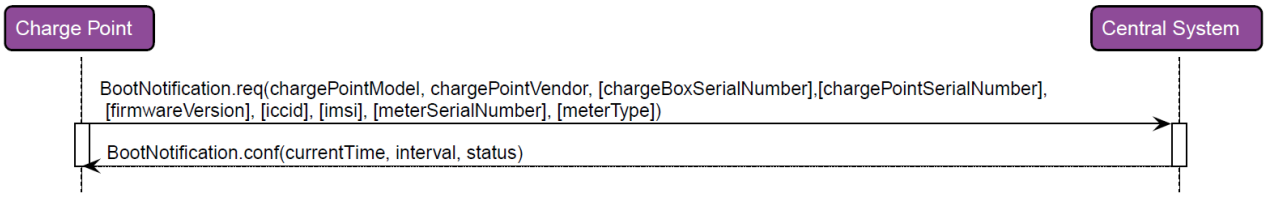
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_AuthorizationStatus [▶ 191]	Status der Autorisierung als Antwort an den OCPP-Client.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.21 RespBootNotification



Mit dieser Methode antwortet ein OCPP-Server auf eine Boot Notification von einem OCPP-Client.



Syntax

```
METHOD RespBootNotification : BOOL
VAR_INPUT
    hStationId         : UDINT;
    hMessageId         : T_OCPP_MessageId;
    eStatus            : E_OCPP1_RegistrationStatus;
    tHeartbeatInterval : TIME := T#5S;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RespBootNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

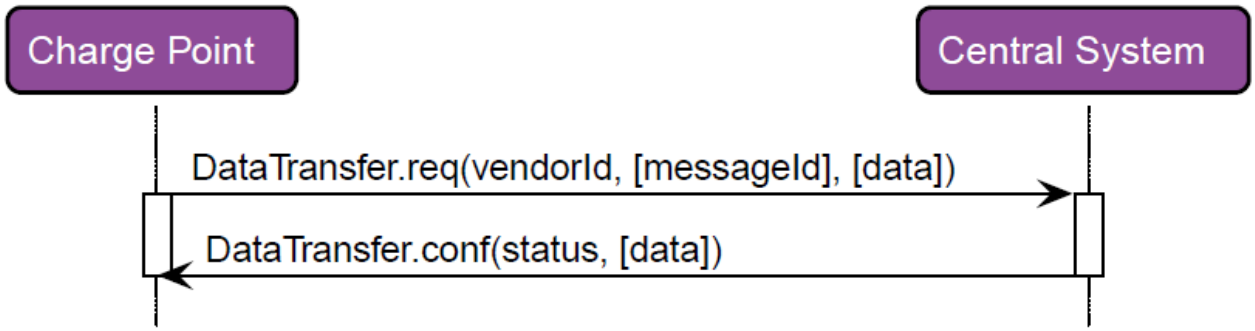
Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_RegistrationStatus [▶ 197]	Status der Registrierung als Antwort an den OCPP-Client.
tHeartbeatInterval	TIME	Wenn der Registration Status Accepted ist, enthält diese Variable das Heartbeat Interval für die Verbindung.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.22 RespDataTransfer

Mit dieser Methode antwortet ein OCPP-Server auf einen Data Transfer-Request von einem OCPP-Client.



Syntax

```
METHOD RespDataTransfer : BOOL
VAR_INPUT
    hStationId : UDINT;
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_DataTransferStatus;
END_VAR
```

Rückgabewert

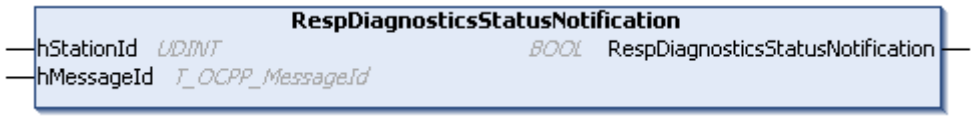
Name	Typ	Beschreibung
RespDataTransfer	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

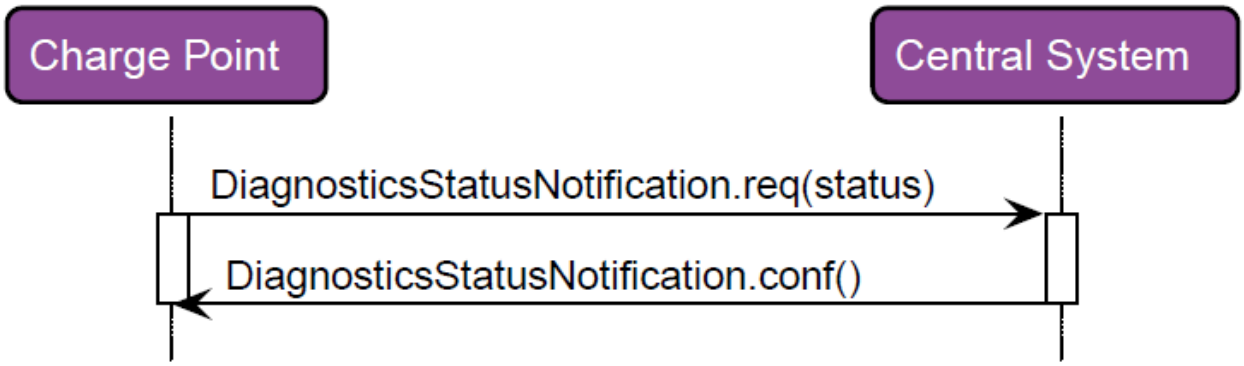
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_DataTransferStatus [▶ 194]	Status des Data Transfers als Antwort an den OCPP-Client.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.23 RespDiagnosticsStatusNotification



Mit dieser Methode antwortet ein OCPP-Server auf eine Diagnostics Status Notification von einem OCPP-Client.



Syntax

```
METHOD RespDiagnosticsStatusNotification : BOOL
VAR_INPUT
    hStationId : UDINT;
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

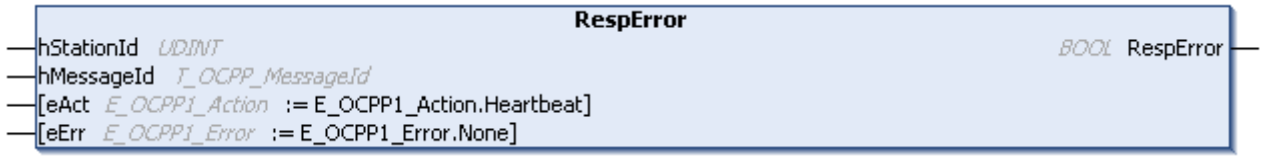
Name	Typ	Beschreibung
RespDiagnosticsStatusNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.24 RespError



Mit dieser Methode antwortet ein OCPP-Server einem OCPP-Client, wenn der gesendete Request einen internen Fehler ausgelöst hat. Das kann beispielsweise der Fall sein, wenn eine angefragte Aktion nicht verfügbar ist.

Syntax

```
METHOD RespError : BOOL
VAR_INPUT
    hStationId : UDINT;
    hMessageId : T_OCPP_MessageId;
    eAct       : E_OCPP1_Action := E_OCPP1_Action.Heartbeat;
    eErr       : E_OCPP1_Error := E_OCPP1_Error.None;
END_VAR
```

Rückgabewert

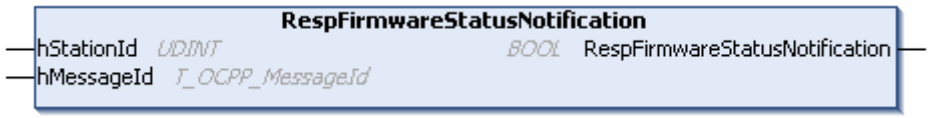
Name	Typ	Beschreibung
RespError	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

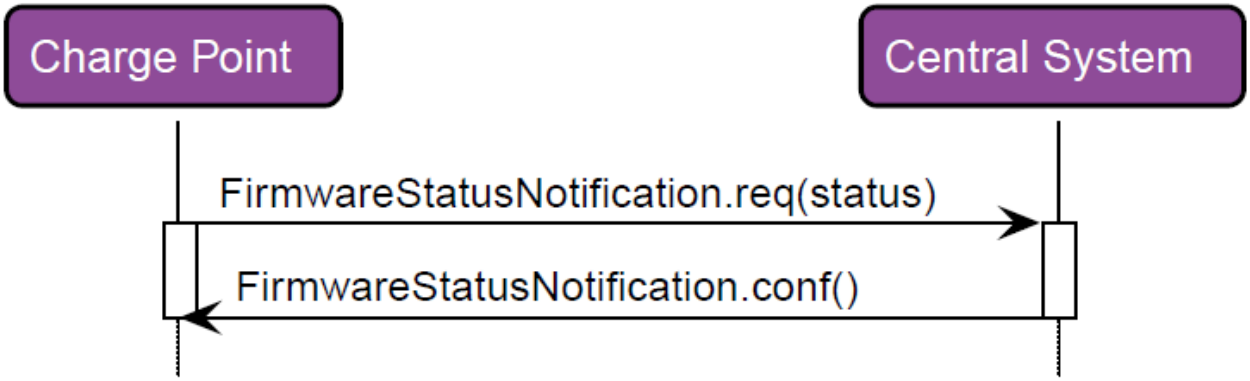
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eAct	E_OCPP1_Action [▶ 190]	Art des OCPP-Requests, bei dem der Fehler aufgetreten ist.
eErr	E_OCPP1_Error [▶ 195]	OCPP-Error, mit dem der Request beantwortet werden soll.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.25 RespFirmwareStatusNotification



Mit dieser Methode antwortet ein OCPP-Server auf eine Firmware Status Notification von einem OCPP-Client.



Syntax

```
METHOD RespFirmwareStatusNotification : BOOL
VAR_INPUT
    hStationId : UDINT;
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

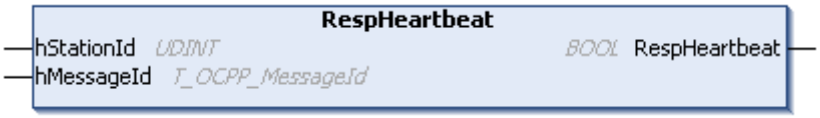
Name	Typ	Beschreibung
RespFirmwareStatusNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

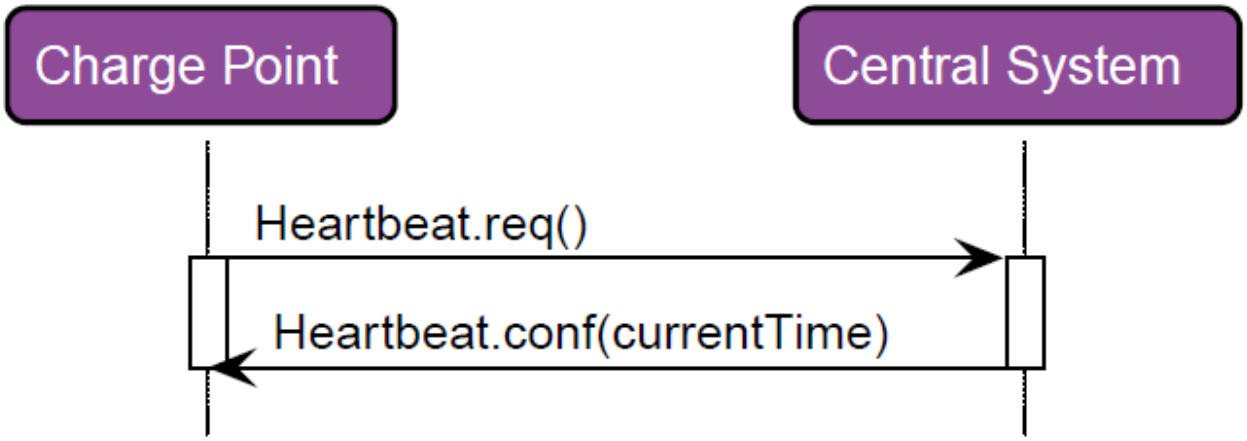
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.26 RespHeartbeat



Mit dieser Methode antwortet ein OCPP-Server auf einen Heartbeat von einem OCPP-Client. Der Zeitstempel wird intern gesetzt und muss nicht separat in der SPS gesetzt werden.



Syntax

```
METHOD RespHeartbeat : BOOL
VAR_INPUT
    hStationId : UDINT;
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RespHeartbeat	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

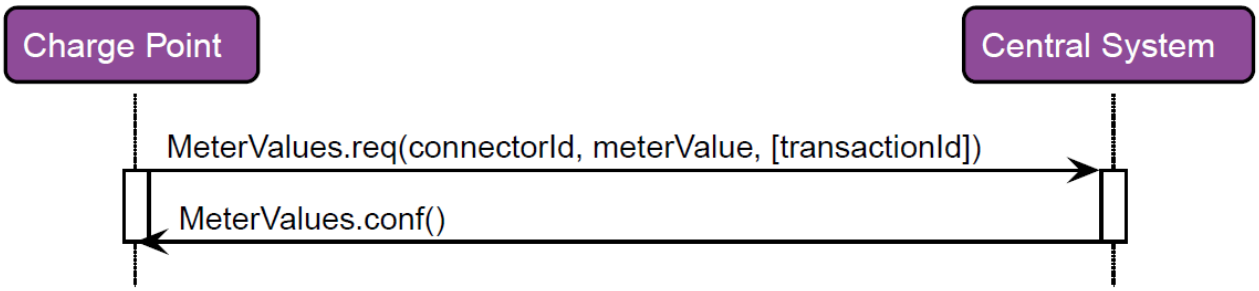
Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.27 RespMeterValue

Mit dieser Methode antwortet ein OCPP-Server auf das Senden von Meter Values von einem OCPP-Client.



Syntax

```
METHOD RespMeterValue : BOOL
VAR_INPUT
    hStationId : UDINT;
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

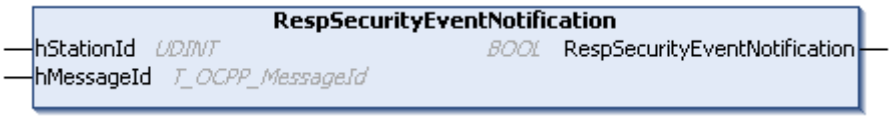
Name	Typ	Beschreibung
RespMeterValue	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

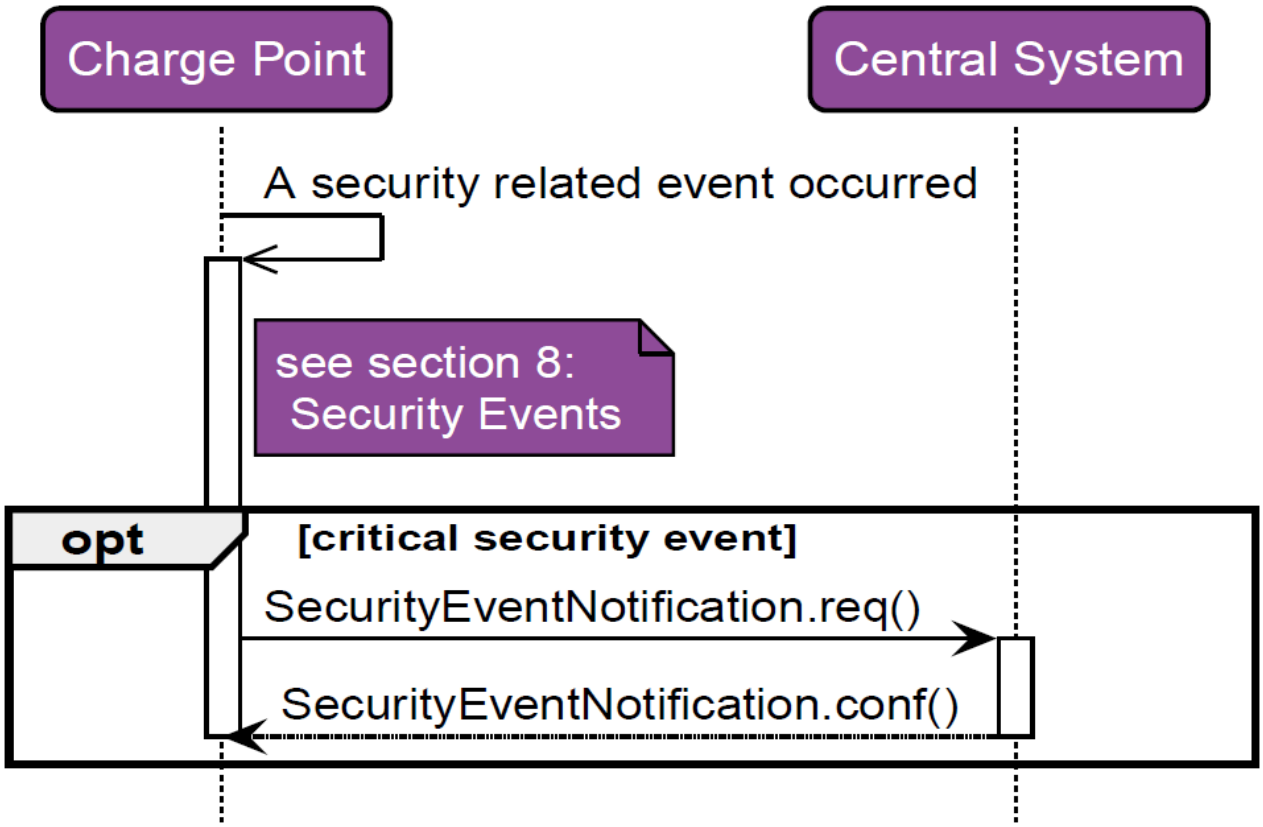
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.28 RespSecurityEventNotification



Mit dieser Methode antwortet ein OCPP-Server auf eine Security Event Notification von einem OCPP-Client.



Syntax

```
METHOD RespSecurityEventNotification : BOOL
VAR_INPUT
    hStationId : UDINT;
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

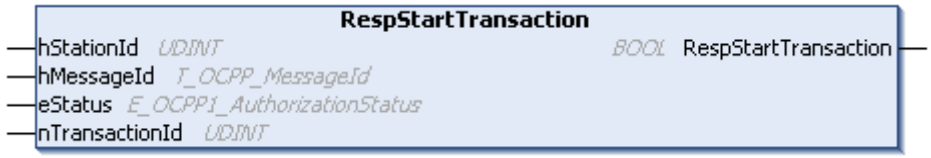
Name	Typ	Beschreibung
RespSecurityEventNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

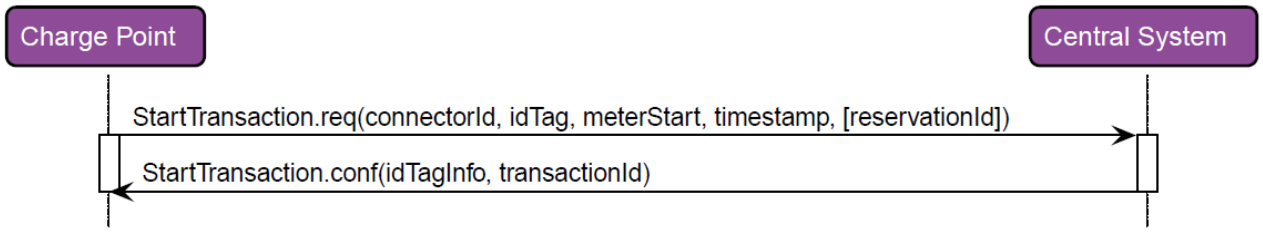
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.29 RespStartTransaction



Mit dieser Methode antwortet ein OCPP-Server auf einen Start Transaction-Request von einem OCPP-Client.



Syntax

```
METHOD RespStartTransaction : BOOL
VAR_INPUT
    hStationId     : UDINT;
    hMessageId     : T_OCPP_MessageId;
    eStatus        : E_OCPP1_AuthorizationStatus;
    nTransactionId : UDINT;
END_VAR
```

Rückgabewert

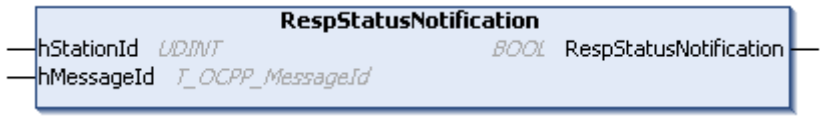
Name	Typ	Beschreibung
RespStartTransaction	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

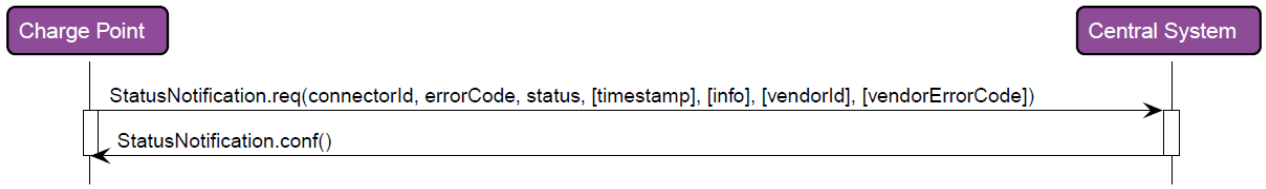
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_AuthorizationStatus [▶ 191]	Status der Autorisierung als Antwort an den OCPP-Client.
nTransactionId	UDINT	Vom Central System festgelegte ID für die Transaktion.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.30 RespStatusNotification



Mit dieser Methode antwortet ein OCPP-Server auf eine Status Notification von einem OCPP-Client.



Syntax

```
METHOD RespStatusNotification : BOOL
VAR_INPUT
    hStationId : UDINT;
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RespStatusNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

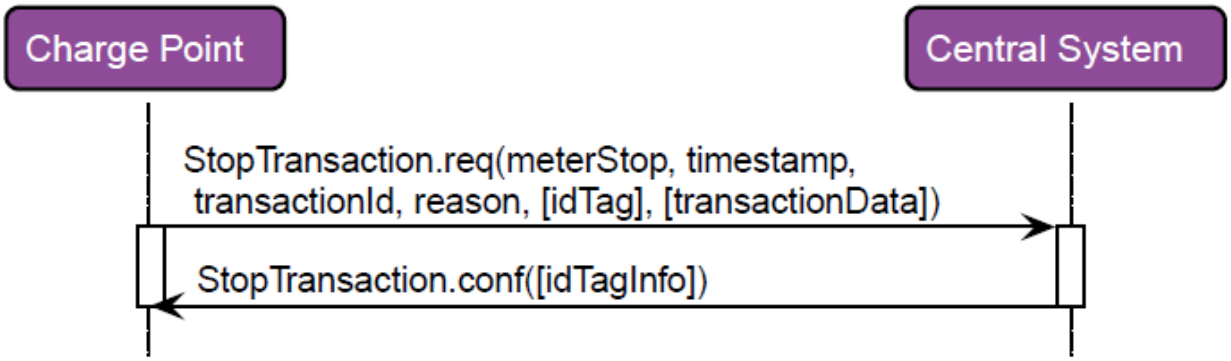
Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.31 RespStopTransaction

Mit dieser Methode antwortet ein OCPP-Server auf einen Stop Transaction-Request von einem OCPP-Client.



Syntax

```
METHOD RespStopTransaction : BOOL
VAR_INPUT
    hStationId : UDINT;
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_AuthorizationStatus;
END_VAR
```

Rückgabewert

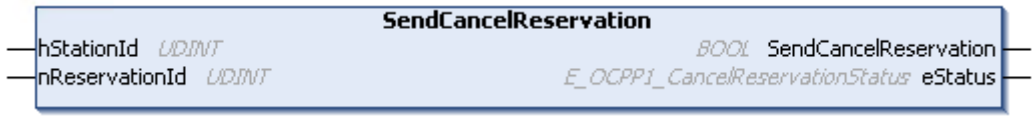
Name	Typ	Beschreibung
RespStopTransaction	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

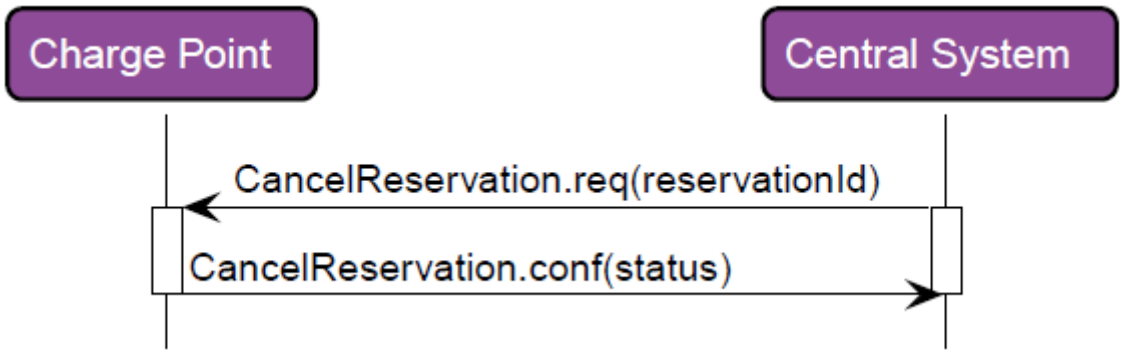
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_AuthorizationStatus [▶ 191]	Status der Autorisierung als Antwort an den OCPP-Client.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.32 SendCancelReservation



Mit dieser Methode sendet ein OCPP-Server einen Cancel Reservation-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendCancelReservation : BOOL
VAR_INPUT
    hStationId     : UDINT;
    nReservationId : UDINT;
END_VAR
VAR_OUTPUT
    eStatus        : E_OCPP1_CancelReservationStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendCancelReservation	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

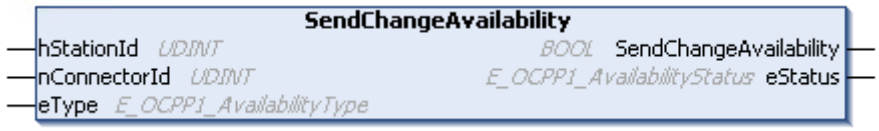
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
nReservationId	UDINT	ID der zu stornierenden Reservierung.

Ausgänge

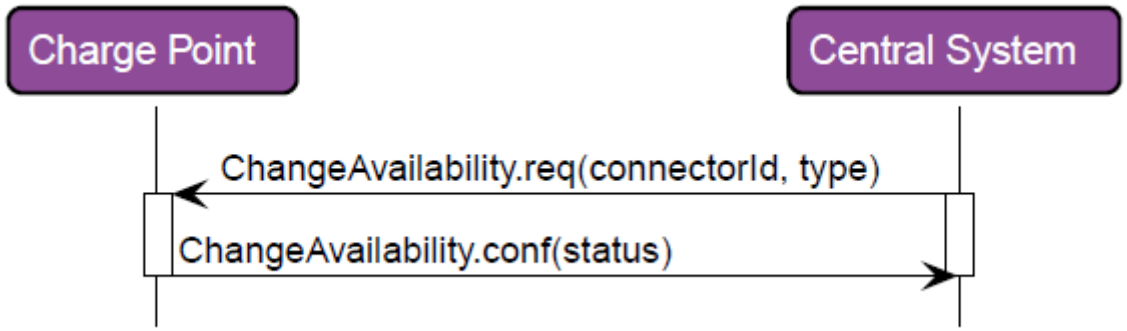
Name	Typ	Beschreibung
eStatus	E_OCPP1_CancelReservationStatus [▶ 192]	Der Status zeigt an, ob die Stornierung der Reservierung erfolgreich war.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.33 SendChangeAvailability



Mit dieser Methode sendet ein OCPP-Server einen Change Availability-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendChangeAvailability : BOOL
VAR_INPUT
    hStationId   : UDINT;
    nConnectorId : UDINT;
    eType        : E_OCPP1_AvailabilityType;
END_VAR
VAR_OUTPUT
    eStatus      : E_OCPP1_AvailabilityStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendChangeAvailability	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
eType	E_OCPP1_AvailabilityType [▶ 192]	Typ der Verfügbarkeitsänderung, die der Charge Point durchführen soll.

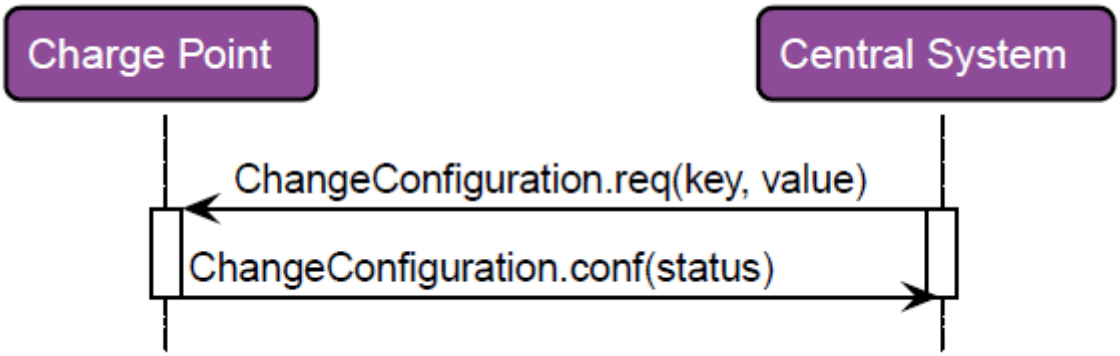
Ausgänge

Name	Typ	Beschreibung
eStatus	E_OCPP1_AvailabilityStatus [▶ 191]	Antwort, ob der Charge Point die angefragte Verfügbarkeitsänderung umsetzen konnte.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.34 SendChangeConfiguration

Mit dieser Methode sendet ein OCPP-Server einen Change Configuration-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendChangeConfiguration : BOOL
VAR_INPUT
    hStationId : UDINT;
    sKey       : T_OCPP1_ConfigKey;
    sValue     : T_OCPP1_ConfigValue;
END_VAR
VAR_OUTPUT
    eStatus    : E_OCPP1_ConfigurationStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendChangeConfiguration	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

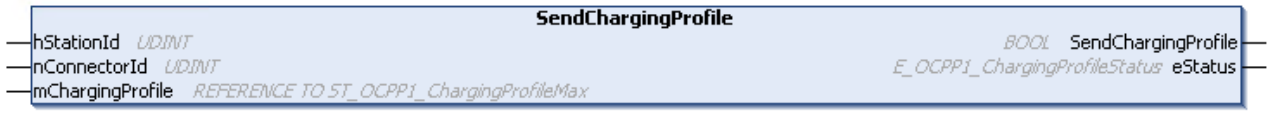
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
sKey	T_OCPP1_ConfigKey [▶ 211]	Der Name der zu ändernden Konfigurationseinstellung.
sValue	T_OCPP1_ConfigValue [▶ 211]	Der neue Wert der Konfigurationseinstellung.

Ausgänge

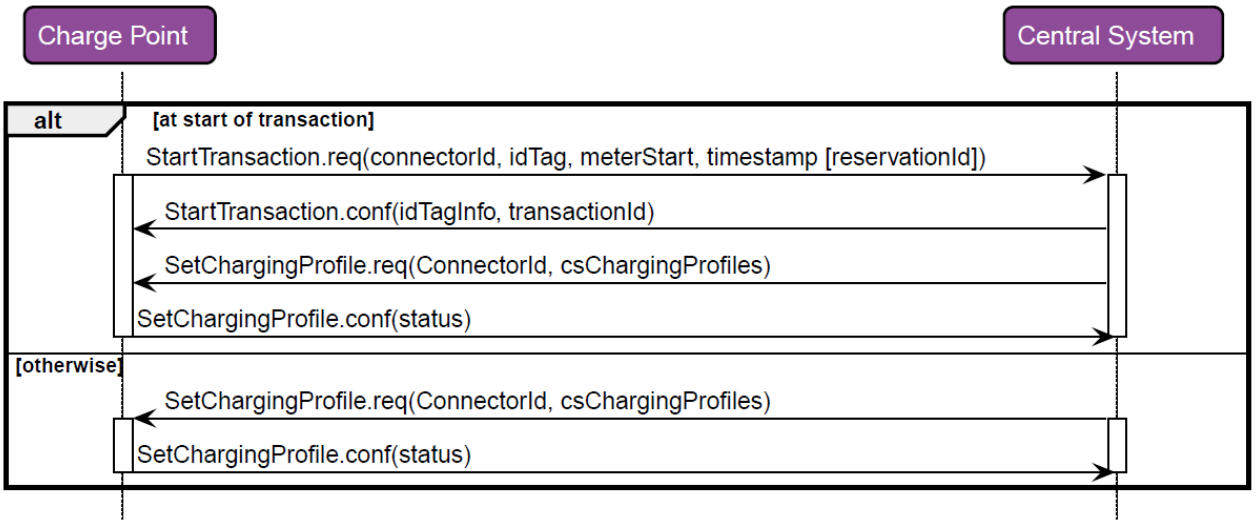
Name	Typ	Beschreibung
eStatus	E_OCPP1_ConfigurationStatus [▶ 194]	Der Status zeigt an, ob die Konfigurationsänderung akzeptiert wurde.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.35 SendChargingProfile



Mit dieser Methode sendet ein OCPP-Server ein Charging Profile an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendChargingProfile : BOOL
VAR_INPUT
    hStationId       : UDINT;
    nConnectorId     : UDINT;
    mChargingProfile : REFERENCE TO ST_OCPP1_ChargingProfileMax;
END_VAR
VAR_OUTPUT
    eStatus          : E_OCPP1_ChargingProfileStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendChargingProfile	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

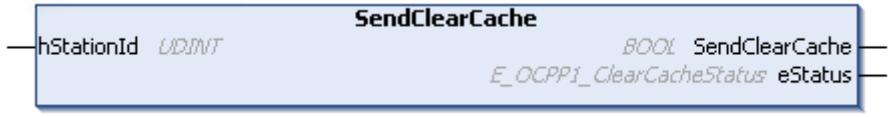
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
mChargingProfile	REFERENCE TO ST_OCPP1_ChargingProfileMax [▶ 206]	An den Client zu sendendes Charging Profile.

Ausgänge

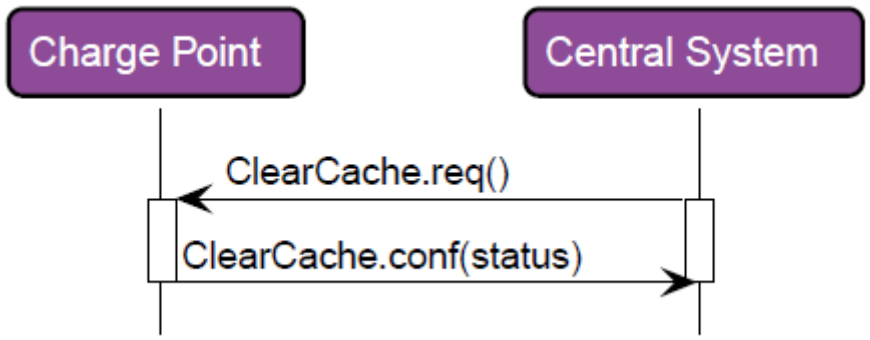
Name	Typ	Beschreibung
eStatus	E_OCPP1_ChargingProfileStatus [▶ 193]	Der Status zeigt an, ob das Charging Profile von dem Charge Point akzeptiert wurde.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.36 SendClearCache



Mit dieser Methode sendet ein OCPP-Server einen Clear Cache-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendClearCache : BOOL
VAR_INPUT
    hStationId : UDINT;
END_VAR
```

```
VAR_OUTPUT
    eStatus     : E_OCPP1_ClearCacheStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendClearCache	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

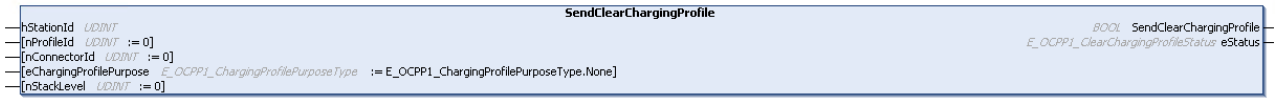
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.

Ausgänge

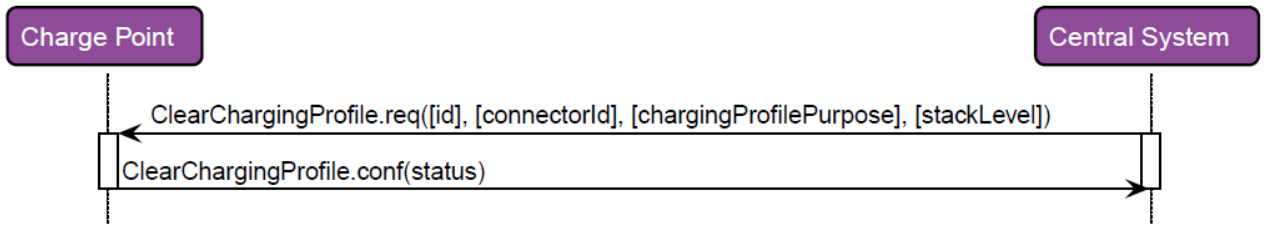
Name	Typ	Beschreibung
eStatus	E_OCPP1_ClearCacheStatus [▶ 193]	Der Status zeigt an, ob das Clearen des Caches Profile von dem Charge Point akzeptiert wurde.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.37 SendClearChargingProfile



Mit dieser Methode sendet ein OCPP-Server einen Clear Charging Profile-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendClearChargingProfile : BOOL
VAR_INPUT
    hStationId              : UDINT;
    nProfileId              : UDINT := 0;
    nConnectorId            : UDINT := 0;
    eChargingProfilePurpose : E_OCPP1_ChargingProfilePurposeType := E_OCPP_ChargingProfilePurposeTyp
e.None;
    nStackLevel             : UDINT := 0;
END_VAR
VAR_OUTPUT
    eStatus                 : E_OCPP1_ClearChargingProfileStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendClearChargingProfile	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

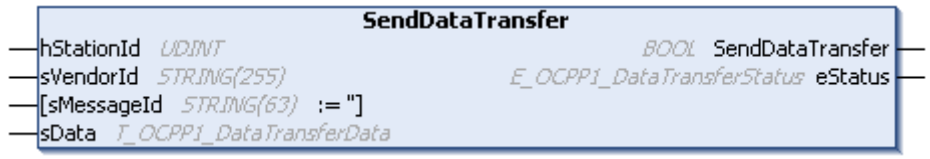
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
nProfileId	UDINT	Identifier des zu löschenden Charging Profile.
nConnectorId	UDINT	ID des Connectors eines Charge Points. Der Wert 0 besagt, dass das Löschen sich auf den gesamten Charge Point bezieht.
eChargingProfilePurpose	E_OCPP1_ChargingProfilePurposeType [▶ 193]	Spefiziert den Zweck der zu löschenden Charging Profiles.
nStackLevel	UDINT	Spezifiziert das StackLevel, für das Charging Profile gelöscht werden.

Ausgänge

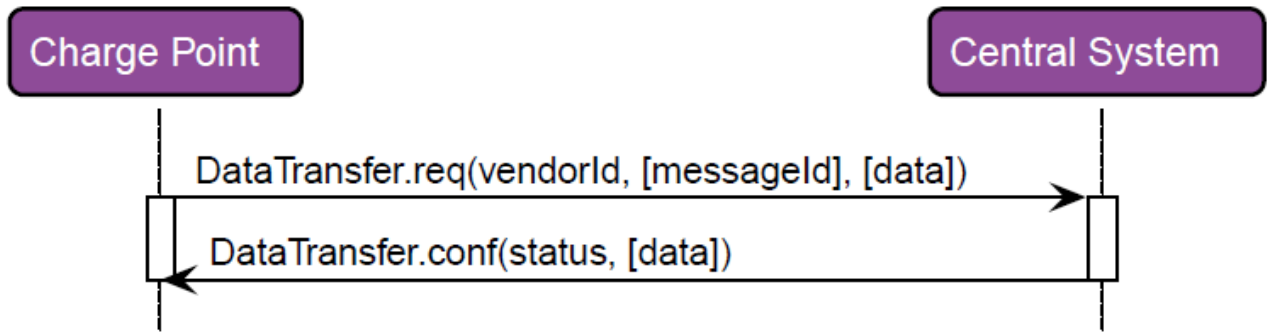
Name	Typ	Beschreibung
eStatus	E_OCPP1_ClearChargingProfileStatus [▶ 193]	Der Status zeigt an, ob das Clearen des Charging Profiles von dem Charge Point durchgeführt werden konnte.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.38 SendDataTransfer



Mit dieser Methode sendet ein OCPP-Server einen DataTransfer-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendDataTransfer : BOOL
VAR_INPUT
    hStationId : UDINT;
    sVendorId  : STRING(255);
    sMessageId : STRING(63) := '';
END_VAR
VAR_IN_OUT CONSTANT
    sData      : T_OCPP1_DataTransferData;
END_VAR
VAR_OUTPUT
    eStatus    : E_OCPP1_DataTransferStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendDataTransfer	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
sVendorId	STRING(255)	Identifier für den Hersteller, der die herstellerspezifische Implementierung kennzeichnet.
sMessageId	STRING(63)	Zusätzliches Identifikationsfeld für eine einzelne Nachricht.

Ein-/Ausgänge

Name	Typ	Beschreibung
sData	T_OCPP1_DataTransferData [▶ 211]	Text ohne spezifizierte Länge und Format.

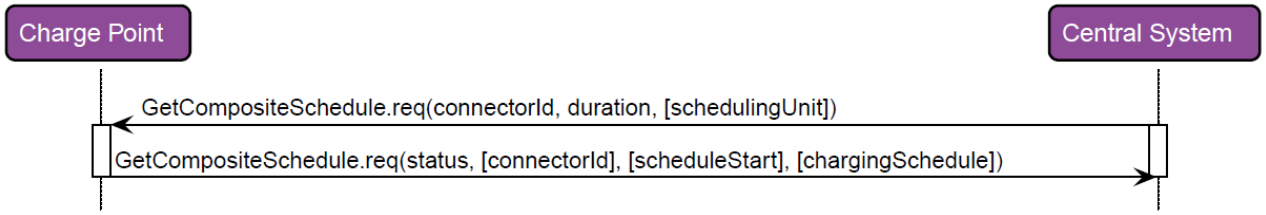
Ausgänge

Name	Typ	Beschreibung
eStatus	E_OCPP1_DataTransferStatus [▶ 194]	Status des Data Transfers.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.39 SendGetCompositeSchedule

Mit dieser Methode sendet ein OCPP-Server einen Get Composite Schedule-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendGetCompositeSchedule : BOOL
VAR_INPUT
    hStationId        : UDINT;
    nConnectorId      : UDINT;
    nDuration         : UDINT;
    eChargingRateUnit : E_OCPP1_ChargingRateUnitType := E_OCPP_ChargingRateUnitType.None;
END_VAR
VAR_OUTPUT
    eStatus           : E_OCPP1_GetCompositeScheduleStatus;
    nConnectorIdRes   : UDINT;
    nScheduleStart    : ULINT := 0;
    mChargingSchedule : ST_OCPP1_ChargingScheduleMax;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendGetCompositeSchedule	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
nDuration	UDINT	Länge des angefragten Zeitplans.
eChargingRateUnit	E_OCPP1_ChargingRateUnitType [▶ 193]	Wird zur Angabe der Einheit für die Anfrage verwendet.

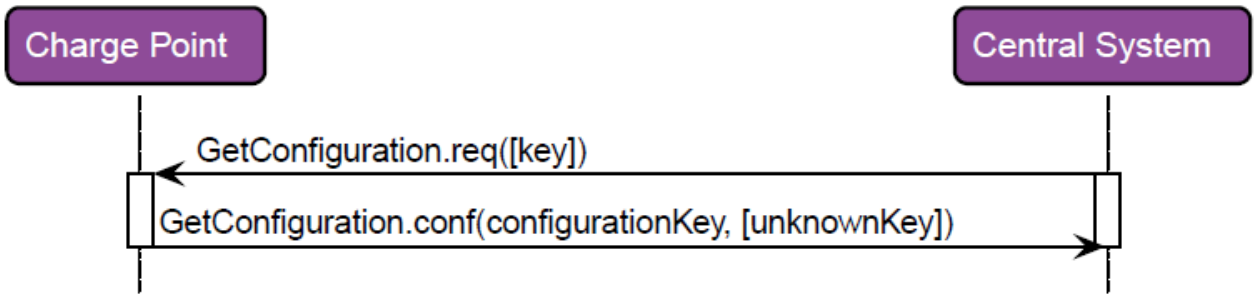
Ausgänge

Name	Typ	Beschreibung
eStatus	E_OCPP1_GetCompositeScheduleStatus [▶ 195]	Der Status zeigt an, ob die Anfrage des Zeitplans von dem Charge Point beantwortet werden konnte.
nConnectorIdRes	UDINT	Identifier des Connectors, auf den sich der Zeitplan in der Antwort bezieht.
nScheduleStart	ULINT	Startzeit des Zeitplans.
mChargingSchedule	ST_OCPP1_ChargingScheduleMax [▶ 206]	Angefragter Zeitplan.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.40 SendGetConfiguration

Mit dieser Methode sendet ein OCPP-Server einen Get Configuration-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendGetConfiguration : BOOL
VAR_INPUT
    hStationId    : UDINT;
END_VAR
VAR_IN_OUT
    arrKeys       : ARRAY[*] OF ST_OCPP1_ConfigKeyValue;
END_VAR
VAR_OUTPUT
    nKeysCount    : UDINT := 0;
    nUnknownCount : UDINT := 0;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendGetConfiguration	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.

Ein-/Ausgänge

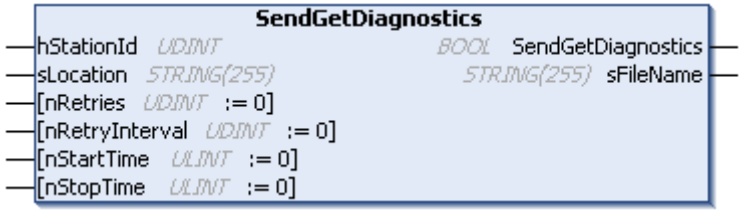
Name	Typ	Beschreibung
arrKeys	ARRAY [*] OF ST_OCPP1_ConfigKeyValue [▶ 210]	Liste von angefragten oder bekannten Konfigurationsschlüsseln, für die die Konfiguration angefragt wird.

Ausgänge

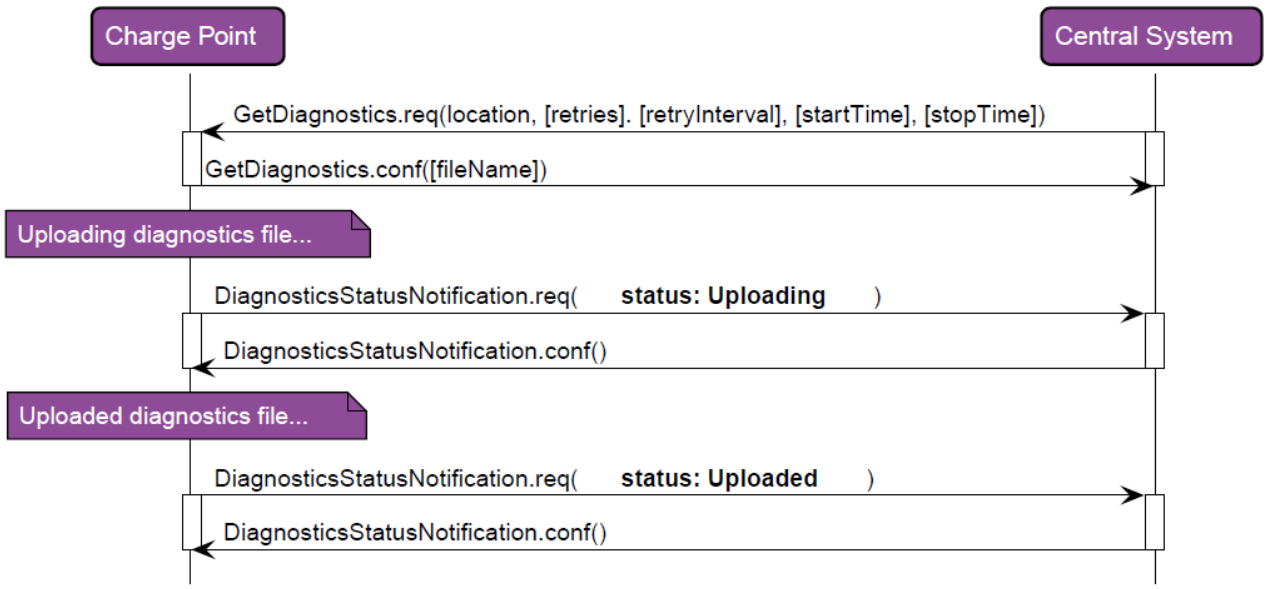
Name	Typ	Beschreibung
nKeysCount	UDINT	Anzahl der folgenden Konfigurationsschlüssel.
nUnknownCount	UDINT	Anzahl der folgenden unbekannten Konfigurationsschlüssel.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.41 SendGetDiagnostics



Mit dieser Methode sendet ein OCPP-Server einen Get Diagnostics-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendGetDiagnostics : BOOL
VAR_INPUT
    hStationId     : UDINT;
    sLocation      : STRING(255);
    nRetries       : UDINT := 0;
    nRetryInterval : UDINT := 0;
    nStartTime     : ULINT := 0;
    nStopTime      : ULINT := 0;
END_VAR
VAR_OUTPUT
    sFileName      : STRING(255);
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendGetDiagnostics	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

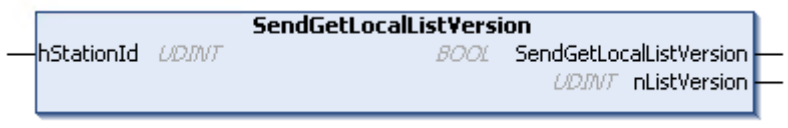
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
sLocation	STRING(255)	Verzeichnis, in das die Diagnosedatei hochgeladen werden soll.
nRetries	UDINT	Anzahl der Versuche, die der Charge Point bei fehlschlagendem Upload unternimmt.
nRetryInterval	UDINT	Intervall, nach dem ein neuer Upload versucht wird.
nStartTime	ULINT	Startzeit der in der Diagnose enthaltenen Logging-Informationen.
nStopTime	ULINT	Endzeit der in der Diagnose enthaltenen Logging-Informationen.

Ausgänge

Name	Typ	Beschreibung
sFileName	STRING(255)	Enthält den Namen der Diagnosedatei, die hochgeladen werden wird.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.42 SendGetLocalListVersion



Mit dieser Methode sendet ein OCPP-Server einen Get Local List Version-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.

Syntax

```
METHOD SendGetLocalListVersion : BOOL
VAR_INPUT
    hStationId   : UDINT;
END_VAR
VAR_OUTPUT
    nListVersion : UDINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendGetLocalListVersion	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.

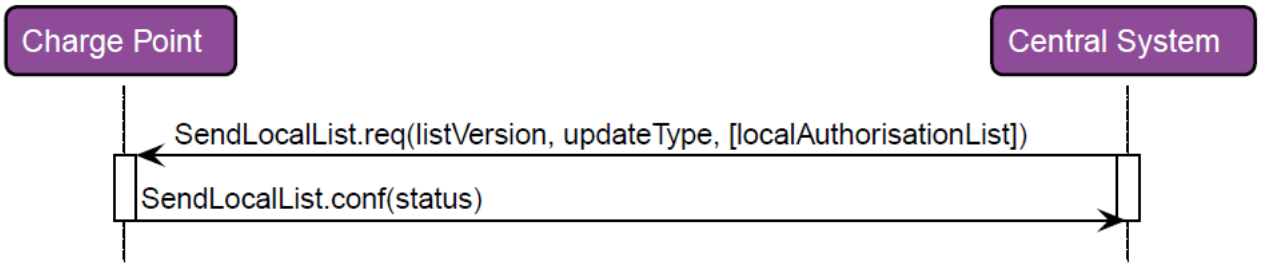
Ausgänge

Name	Typ	Beschreibung
nListVersion	UDINT	Enthält die Versionsnummer der aktuell im Charge Point vorliegenden Local Authorization List.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.43 SendLocalList

Mit dieser Methode sendet ein OCPP-Server einen Local List-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendLocalList : BOOL
VAR_INPUT
    hStationId   : UDINT;
    nListVersion : UDINT;
    eUpdateType  : E_OCPP1_UpdateType;
    nAuthCount   : UDINT := 0;
END_VAR
VAR_IN_OUT
    arrAuthList  : ARRAY[*] OF ST_OCPP1_AuthorizationData;
END_VAR
VAR_OUTPUT
    eStatus      : E_OCPP1_UpdateStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendLocalList	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
nListVersion	UDINT	Version der als Update bereitgestellten Local Authorization List.
eUpdateType	E_OCPP1_UpdateType [▶ 199]	Festlegung des Typs des Updates, entweder als Update oder als vollständiger Austausch.
nAuthCount	UDINT	Anzahl an Einträgen in der Local Authorization List.

Ein-/Ausgänge

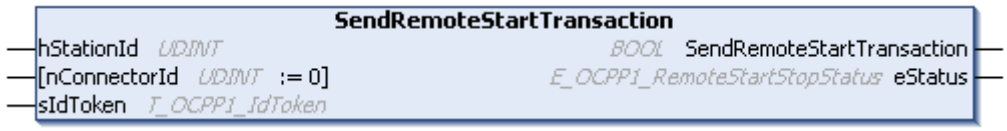
Name	Typ	Beschreibung
arrAuthList	ARRAY [*] OF ST_OCPP1_AuthorizationData [▶ 204]	Entweder die Werte der neuen Local Authorization List (vollständiger Austausch) oder hinzuzufügende Werte der bestehenden Local Authorization List (Update).

Ausgänge

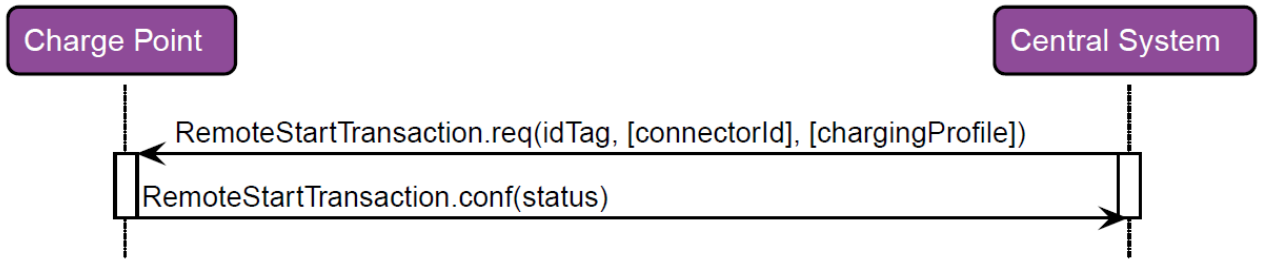
Name	Typ	Beschreibung
eStatus	E_OCPP1_UpdateStatus [▶ 198]	Der Status zeigt an, ob der Charge Point das Update der Local Authorization List erhalten und durchgeführt hat.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.44 SendRemoteStartTransaction



Mit dieser Methode sendet ein OCPP-Server einen Remote Start Transaction-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendRemoteStartTransaction : BOOL
VAR_INPUT
    hStationId   : UDINT;
    nConnectorId : UDINT := 0;
    sIdToken     : T_OCPP1_IdToken;
END_VAR
VAR_OUTPUT
    eStatus      : E_OCPP1_RemoteStartStopStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendRemoteStartTransaction	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

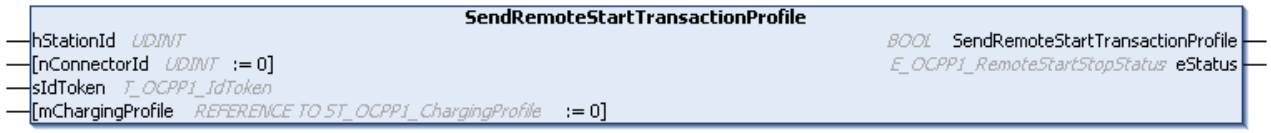
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
sIdToken	T_OCPP1_IdToken [▶ 212]	ID-Token des Charge Points im Central System.

Ausgänge

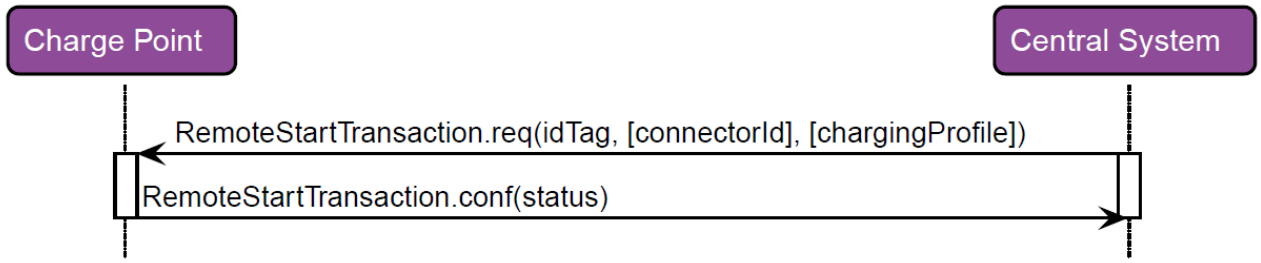
Name	Typ	Beschreibung
eStatus	E_OCPP1_RemoteStartStopStatus [▶ 197]	Der Status zeigt an, ob der Charge Point den Request zum Starten einer Transaktion akzeptiert hat.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.45 SendRemoteStartTransactionProfile



Mit dieser Methode sendet ein OCPP-Server einen Remote Start Transaction-Request inklusive Charging Profile an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendRemoteStartTransactionProfile : BOOL
VAR_INPUT
    hStationId       : UDINT;
    nConnectorId     : UDINT := 0;
    sIdToken         : T_OCPP1_IdToken;
    mChargingProfile : REFERENCE TO ST_OCPP1_ChargingProfile := 0;
END_VAR
VAR_OUTPUT
    eStatus          : E_OCPP1_RemoteStartStopStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendRemoteStartTransactionProfile	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

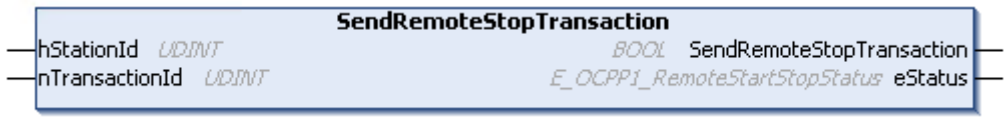
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
sIdToken	T_OCPP1_IdToken [▶ 212]	ID-Token des Charge Points im Central System.
mChargingProfile	REFERENCE TO ST_OCPP1_ChargingProfileMax [▶ 206]	An den Client zu sendendes Charging Profile.

Ausgänge

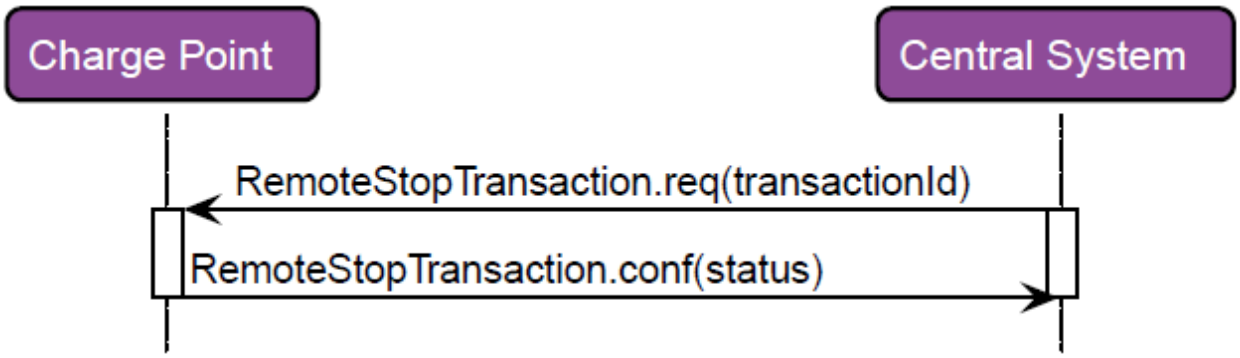
Name	Typ	Beschreibung
eStatus	E_OCPP1_RemoteStartStopStatus [▶ 197]	Der Status zeigt an, ob der Charge Point den Request zum Starten einer Transaktion akzeptiert hat.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.46 SendRemoteStopTransaction



Mit dieser Methode sendet ein OCPP-Server einen Remote Stop Transaction-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendRemoteStopTransaction : BOOL
VAR_INPUT
    hStationId     : UDINT;
    nTransactionId : UDINT;
END_VAR
VAR_OUTPUT
    eStatus        : E_OCPP1_RemoteStartStopStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendRemoteStopTransaction	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
nTransactionId	UDINT	Identifier der Transaktion, die gestoppt werden soll.

Ausgänge

Name	Typ	Beschreibung
eStatus	E_OCPP1_RemoteStartStopStatus [▶ 197]	Der Status zeigt an, ob der Charge Point den Request zum Stoppen einer Transaktion akzeptiert hat.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.47 SendReserveNow

Mit dieser Methode sendet ein OCPP-Server einen Reserve Now-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.

Syntax

```
METHOD SendReserveNow : BOOL
VAR_INPUT
    hStationId     : UDINT;
    nConnectorId   : UDINT;
    nExpiryDate    : ULINT;
    sIdTag         : T_OCPP1_IdToken := '';
    sParentIdTag   : T_OCPP1_IdToken;
    nReservationId : UDINT;
END_VAR
```

```
VAR_OUTPUT
    eStatus         : E_OCPP1_ReservationStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendReserveNow	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

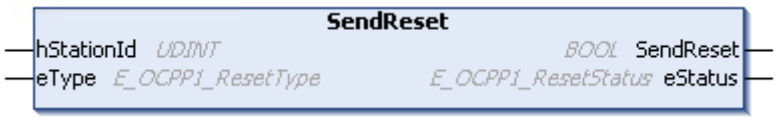
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
nExpiryDate	ULINT	Datum und Zeit des Endes der Reservierung.
sIdTag	T_OCPP1_IdToken [▶ 212]	Identifier, für den der Charge Point einen Connector reservieren soll.
sParentIdTag	T_OCPP1_IdToken [▶ 212]	Der Parent des Identifiers, für den der Charge Point einen Connector reservieren soll.
nReservationId	UDINT	Einzigartige ID der Reservierung.

Ausgänge

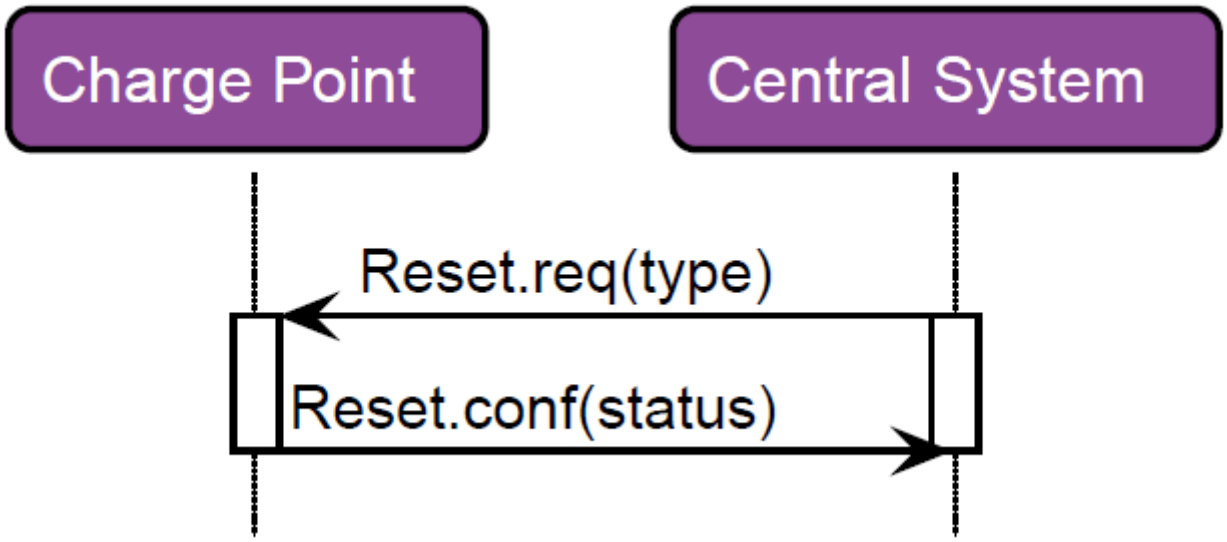
Name	Typ	Beschreibung
eStatus	E_OCPP1_ReservationStatus [▶ 197]	Der Status zeigt an, ob die Reservierung erfolgreich war.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.48 SendReset



Mit dieser Methode sendet ein OCPP-Server einen Reset-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendReset : BOOL
VAR_INPUT
    hStationId : UDINT;
    eType      : E_OCPP1_ResetType;
END_VAR
VAR_OUTPUT
    eStatus    : E_OCPP1_ResetStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendReset	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

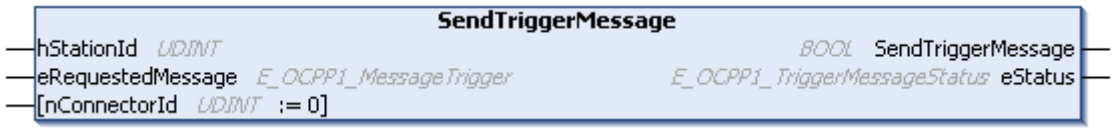
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
eType	E_OCPP1_ResetType [▶ 198]	Typ des Resets, die der Charge Point durchführen soll.

Ausgänge

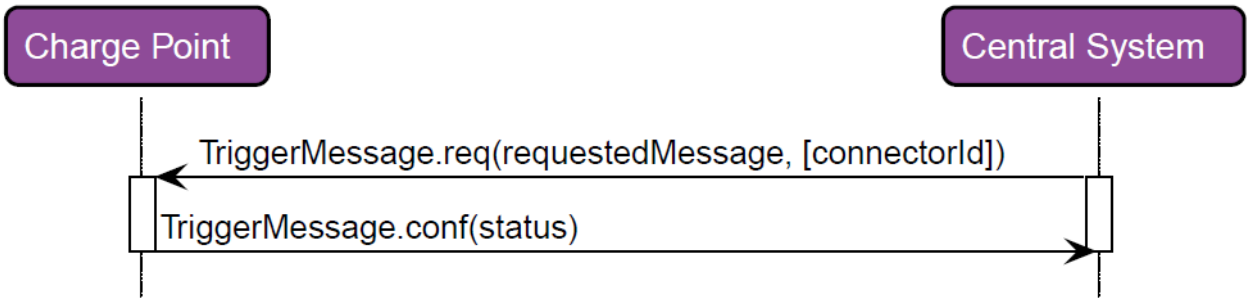
Name	Typ	Beschreibung
eStatus	E_OCPP1_ResetStatus [▶ 198]	Der Status zeigt an, ob der Charge Point den Request zum Reset akzeptieren konnte.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.49 SendTriggerMessage



Mit dieser Methode sendet ein OCPP-Server einen Trigger Message-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendTriggerMessage : BOOL
VAR_INPUT
    hStationId        : UDINT;
    eRequestedMessage : E_OCPP1_MessageTrigger;
    nConnectorId      : UDINT := 0;
END_VAR
VAR_OUTPUT
    eStatus           : E_OCPP1_TriggerMessageStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendTriggerMessage	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
eRequestedMessage	E_OCPP1_MessageTrigger [▶ 196]	Typ der Nachricht, die getriggert werden soll.
nConnectorId	UDINT	ID des Connectors eines Charge Points.

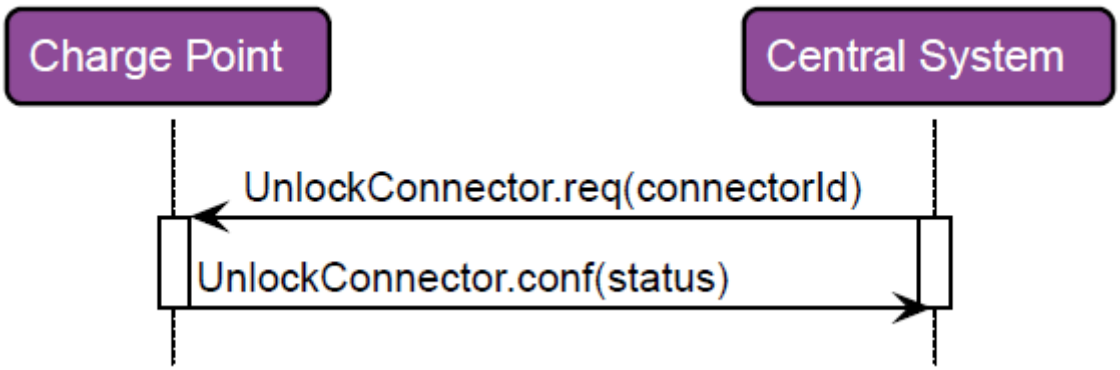
Ausgänge

Name	Typ	Beschreibung
eStatus	E_OCPP1_TriggerMessageStatus [▶ 198]	Der Status zeigt an, ob der Charge Point die angefragte Nachricht senden wird oder nicht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.50 SendUnlockConnector

Mit dieser Methode sendet ein OCPP-Server einen Unlock Connector-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendUnlockConnector : BOOL
VAR_INPUT
    hStationId   : UDINT;
    nConnectorId : UDINT;
END_VAR
VAR_OUTPUT
    eStatus      : E_OCPP1_UnlockStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendUnlockConnector	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

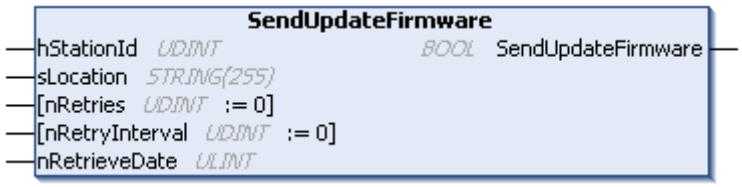
Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
nConnectorId	UDINT	ID des Connectors eines Charge Points.

Ausgänge

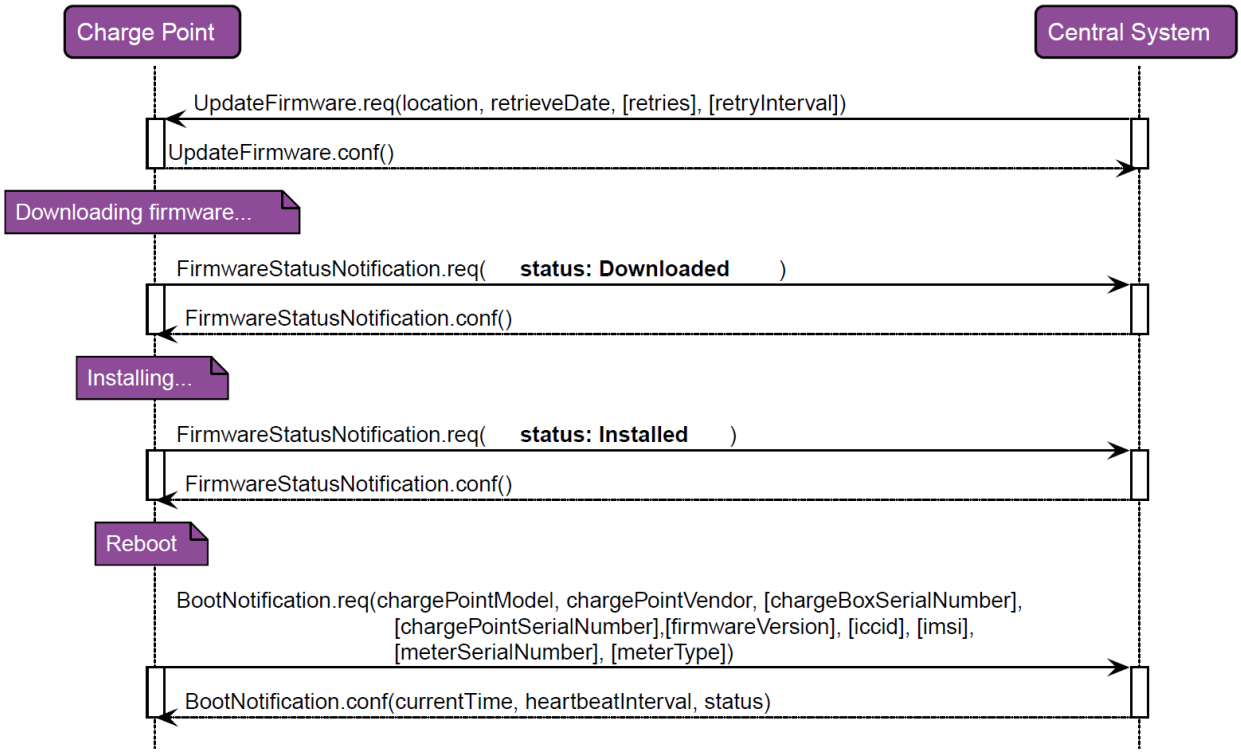
Name	Typ	Beschreibung
eStatus	E_OCPP1_UnlockStatus [▶ 198]	Der Status zeigt an, ob der Connector freigeschaltet wurde.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.2.51 SendUpdateFirmware



Mit dieser Methode sendet ein OCPP-Server einen Update Firmware-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendUpdateFirmware : BOOL
VAR_INPUT
    hStationId     : UDINT;
    sLocation      : STRING(255);
    nRetries       : UDINT := 0;
    nRetryInterval : UDINT := 0;
    nRetrieveDate  : ULINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendUpdateFirmware	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hStationId	UDINT	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.
sLocation	STRING(255)	URI, von der die Firmware abgerufen werden soll.
nRetries	UDINT	Anzahl an Versuchen, die Firmware bei fehlgeschlagenem Download erneut herunterzuladen.
nRetryInterval	UDINT	Zeit, nach der der erneute Download versucht wird.
nRetrieveDate	ULINT	Zeit und Datum, ab dem der Charge Point die neue Firmware beziehen darf.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3 FB_OCPP1_Station

Syntax

```
FUNCTION BLOCK FB_OCPP1_Station
VAR_OUTPUT
    bBusy               : BOOL;
    bError              : BOOL;
    eErrorResult        : HRESULT;
    eErrorAction        : E_OCPP1_Action;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
bValid	BOOL	Die Schnittstelle zum Treiber im Hintergrund besteht.
bBusy	BOOL	Ist TRUE, solange der Baustein mit einer Bearbeitung beschäftigt ist.
bError	BOOL	Wird TRUE, sobald eine Fehlersituation auftritt.
eErrorResult	HRESULT	Zuletzt am Baustein anliegender Fehler.
eErrorAction	E_OCPP1_Action [▶ 190]	OCPP-Befehl, bei dem der Fehler aufgetreten ist.

Eigenschaften

Name	Typ	Zugriff	Beschreibung
IsConnected	BOOL	Get	Status der Verbindung zum OCPP-Server.
IsPending	BOOL	Get	Warten auf Fertigstellung der Anfrage.
StationId	UDINT	Get	Identifier des OCPP-Clients in der Instanz des OCPP-Servers.

5.1.3.1 FB_Exit

Syntax

```
METHOD FB_Exit : BOOL
VAR_INPUT
    bInCopyCode : BOOL;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
FB_Exit	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
bInCopyCode	BOOL	Wenn TRUE, wird die Exit-Methode zum Verlassen einer Instanz aufgerufen, die anschließend kopiert wird (Online Change).

5.1.3.2 FB_Init

Syntax

```
METHOD FB_Init : BOOL
VAR_INPUT
    bInitRetains : BOOL;
    bInCopyCode  : BOOL;
```

```
    sIdent       : T_OCPP_Identity;
    sAuthHash    : T_OCPP_Hash;
    fbServer     : REFERENCE TO FB_OCPP1_Server;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
FB_Init	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
bInitRetains	BOOL	Wenn TRUE, dann werden die Retain-Variablen initialisiert (Warm Start/Cold Start).
bInCopyCode	BOOL	Wenn TRUE, wird die Exit-Methode zum Verlassen einer Instanz aufgerufen, die anschließend kopiert wird (Online Change).
sIdent	T_OCPP_Identity [▶ 212]	Identität des OCPP-Clients.
sAuthHash	T_OCPP_Hash [▶ 212]	Aus der Idendität und dem Passwort generierter Hash. Mehr Informationen zur Berechnung des Hashes finden Sie im Anhang [▶ 214].
fbServer	REFERENCE TO FB_OCPP1_Server [▶ 77]	OCPP-Server-Instanz, die für diesen Station-Funktionsbaustein genutzt werden soll.

5.1.3.3 Init

Diese Methode wird beim Initialisieren des Funktionsbausteins in der Methode FB_Init [▶ 133] aufgerufen und spezifiziert die Parameter des WebSockets-Servers. Wenn die Verbindungsparameter im Nachhinein geändert werden sollen, kann diese Methode während des Laufens der Applikation erneut aufgerufen werden.

Syntax

```
METHOD Init : BOOL
VAR_INPUT
    sIdent    : T_OCPP_Identity;
    sAuthHash : T_OCPP_Hash;
    fbServer  : REFERENCE TO FB_OCPP1_Server;
END_VAR
```

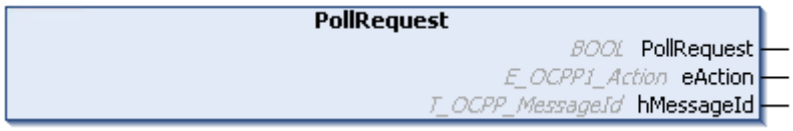
Rückgabewert

Name	Typ	Beschreibung
Init	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE . Auch im Fehlerfall gilt ein Method enaufruf als erfolgreich abgeschlos sen.

Eingänge

Name	Typ	Beschreibung
sIdent	T_OCPP_Identity [▶ 212]	Identitity des zu verbindenden OCPP-Clients.
sAuthHash	T_OCPP_Hash [▶ 212]	Hashwert für den jeweiligen OCPP-Client. Die Berechnung wird unter Hash-Berechnung [▶ 214] erläutert.
fbServer	REFERENCE TO FB_OCPP1_Server [▶ 77]	Instanz des OCPP-Servers für den die Station angelegt werden soll.

5.1.3.4 PollRequest



Diese Methode muss aufgerufen werden, um eingehende Requests des verbundenen OCPP-Clients zu empfangen.

Syntax

```
METHOD PollRequest : BOOL
VAR_OUTPUT
    eAction    : E_OCPP1_Action;
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

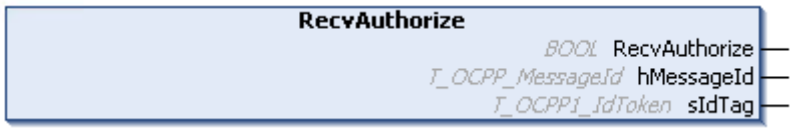
Name	Typ	Beschreibung
PollRequest	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ausgänge

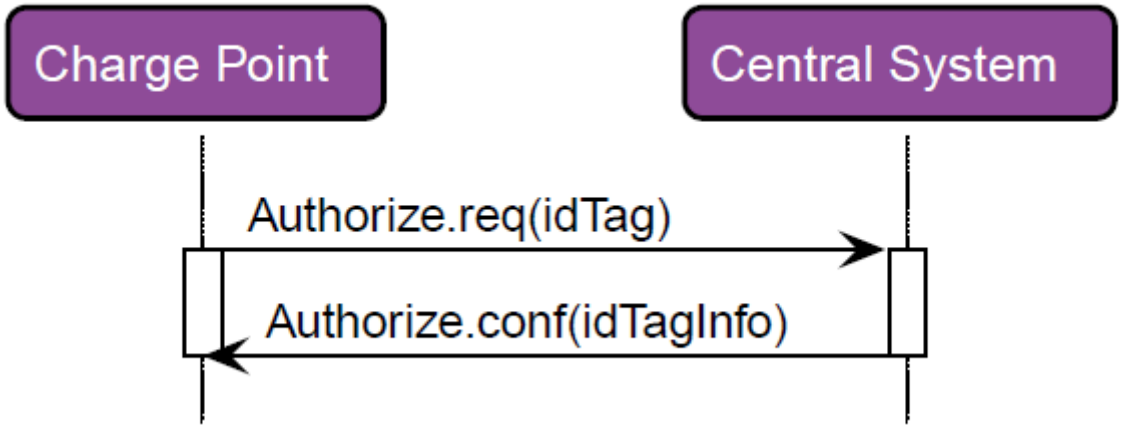
Name	Typ	Beschreibung
eAction	E_OCPP1_Action [▶ 190]	Art des vom Server empfangenen OCPP-Requests.
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.5 RecvAuthorize



Mit dieser Methode empfängt ein OCPP-Server einen Authorize-Request von dem verbundenen OCPP-Client. Um auf den Request zu antworten, muss die Methode RespAuthorize [▶ 149] aufgerufen werden.



Syntax

```
METHOD RecvAuthorize : BOOL
VAR_OUTPUT
    hMessageId : T_OCPP_MessageId;
    sIdTag     : T_OCPP1_IdToken;
END_VAR
```

Rückgabewert

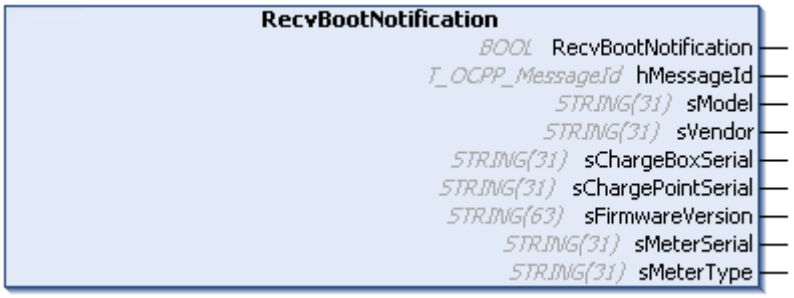
Name	Typ	Beschreibung
RecvAuthorize	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ausgänge

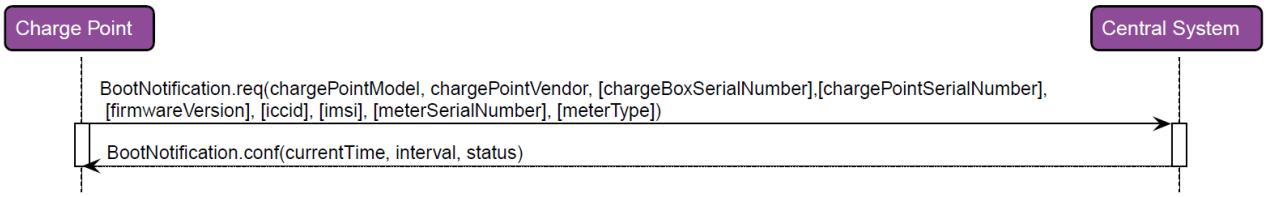
Name	Typ	Beschreibung	
hMessageId		T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
sIdTag		T_OCPP1_IdToken [▶ 212]	ID-Token, mit dem sich der Charge Point am Central System autorisieren lassen will.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.6 RecvBootNotification



Mit dieser Methode empfängt ein OCPP-Server eine Boot Notification von dem verbundenen OCPP-Client. Um auf die Boot Notification zu antworten, muss die Methode RespBootNotification [▶ 150] aufgerufen werden.



Syntax

```
METHOD RecvBootNotification : BOOL
VAR_OUTPUT
    hMessageId          : T_OCPP_MessageId;
    sModel              : STRING(31);
    sVendor             : STRING(31);
    sChargeBoxSerial    : STRING(31);
    sChargePointSerial  : STRING(31);
    sFirmwareVersion    : STRING(63);
    sMeterSerial        : STRING(31);
    sMeterType          : STRING(31);
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvBootNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

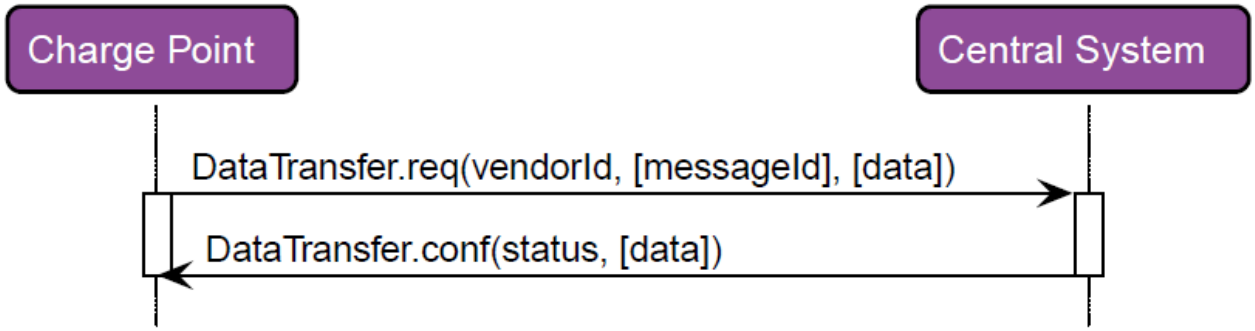
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
sModel	STRING(31)	Modell des Charge Points.
sVendor	STRING(31)	Hersteller des Charge Points.
sChargeBoxSerial	STRING(31)	Seriennummer der Charge Box innerhalb des Charge Points.
sChargePointSerial	STRING(31)	Seriennummer des Charge Points.
sFirmwareVersion	STRING(63)	Firmwareversion des Charge Points.
sMeterSerial	STRING(31)	Seriennummer des Hauptstromzählers des Charge Points.
sMeterType	STRING(31)	Typ des Hauptstromzählers des Charge Points

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.7 RecvDataTransfer

Mit dieser Methode empfängt ein OCPP-Server einen Data Transfer-Request von dem verbundenen OCPP-Client. Um auf den Request zu antworten, muss die Methode RespDataTransfer [▶ 150] aufgerufen werden.



Syntax

```
METHOD RecvDataTransfer : BOOL
VAR_OUTPUT
    hMessageId : T_OCPP_MessageId;
    sVendorId  : STRING(255);
    sMessageId : STRING(63);
END_VAR
VAR_IN_OUT
    sData      : T_OCPP1_DataTransferData;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvDataTransfer	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ausgänge

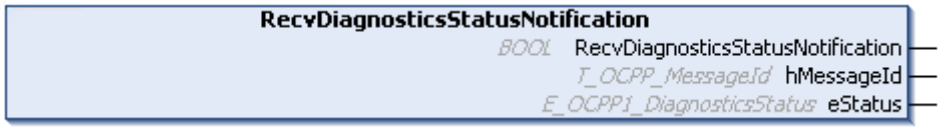
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
sVendorId	STRING(255)	Identifier für die herstellerspezifische Implementierung.
sMessageId	STRING(63)	Identifier für die einzelne Nachricht.

Ein-/Ausgänge

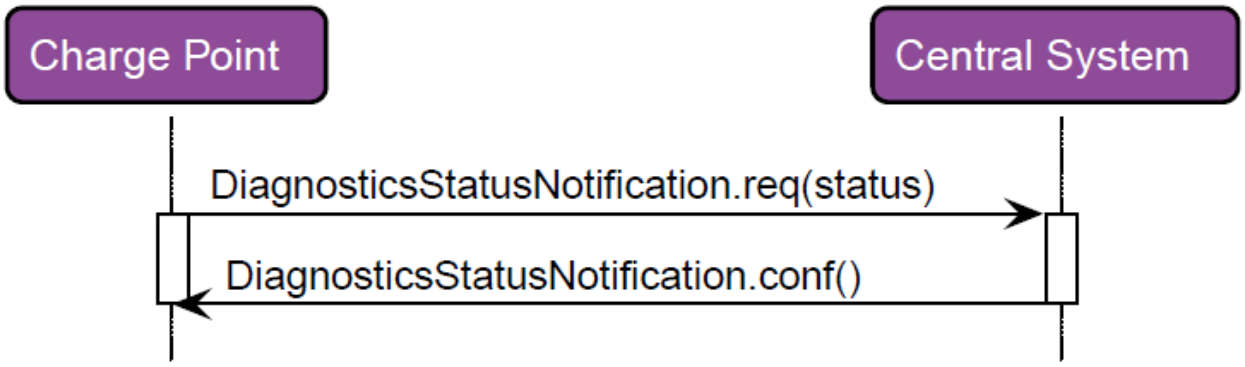
Name	Typ	Beschreibung
sData	T_OCPP1_DataTransferData [▶ 211]	Text ohne spezifizierte Länge und Format.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.8 RecvDiagnosticsStatusNotification



Mit dieser Methode empfängt ein OCPP-Server eine Diagnostics Status Notification von dem verbundenen OCPP-Client. Um auf die Diagnostics Status Notification zu antworten, muss die Methode RespDiagnosticsStatusNotification [▶ 151] aufgerufen werden.



Syntax

```
METHOD RecvDiagnosticsStatusNotification : BOOL
VAR_OUTPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_DiagnosticsStatus;
END_VAR
```

Rückgabewert

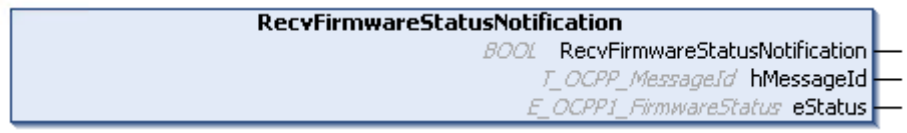
Name	Typ	Beschreibung
RecvDiagnosticsStatusNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ausgänge

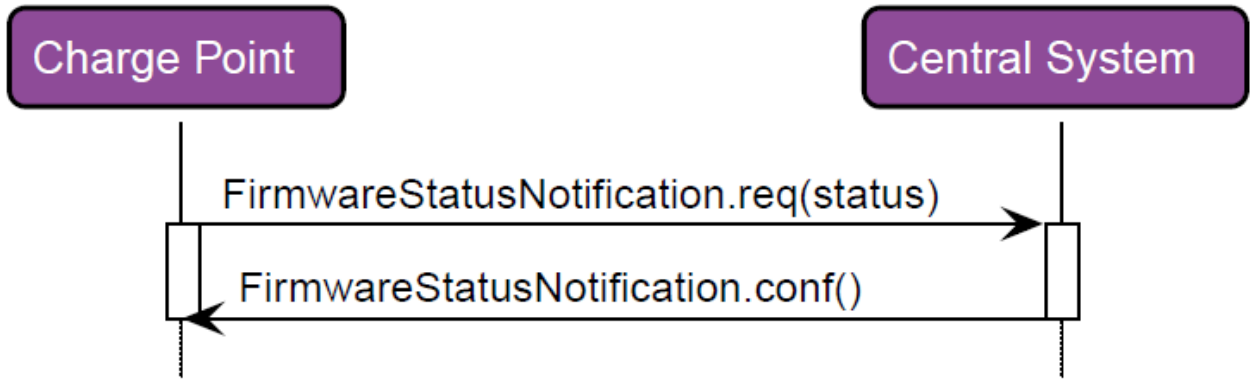
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_DiagnosticsStatus [▶ 194]	Status des Uploads der Diagnose-Daten.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.9 RecvFirmwareStatusNotification



Mit dieser Methode empfängt ein OCPP-Server eine Firmware Status Notification von dem verbundenen OCPP-Client. Um auf die Firmware Status Notification zu antworten, muss die Methode RespFirmwareStatusNotification [▶ 153] aufgerufen werden.



Syntax

```
METHOD RecvFirmwareStatusNotification : BOOL
VAR_OUTPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_FirmwareStatus;
END_VAR
```

Rückgabewert

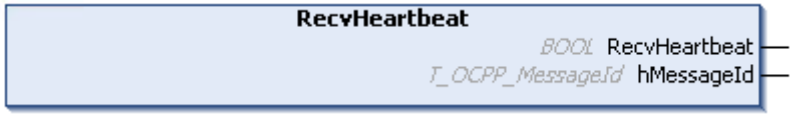
Name	Typ	Beschreibung
RecvFirmwareStatusNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ausgänge

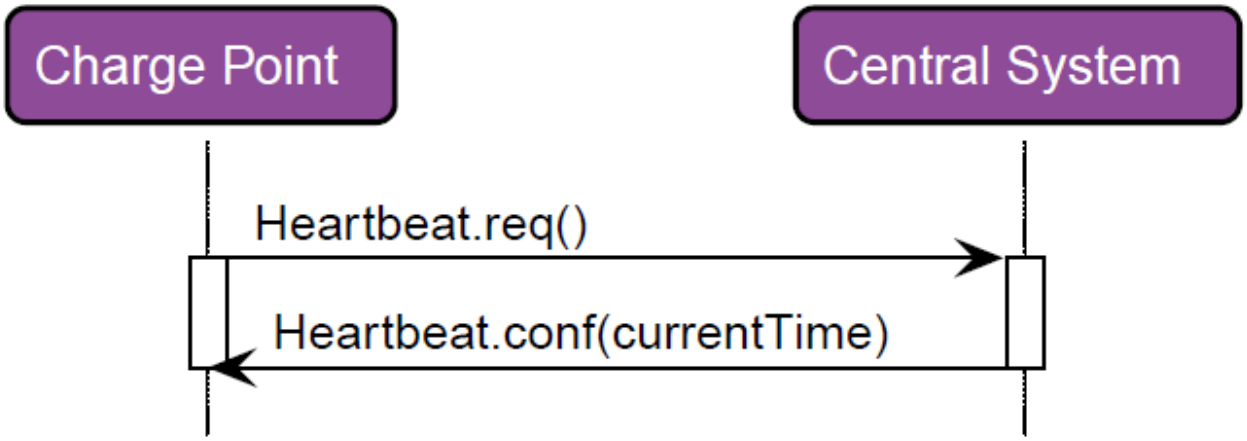
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_FirmwareStatus [▶ 195]	Status einer Firmware-Installation.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.10 RecvHeartbeat



Mit dieser Methode empfängt ein OCPP-Server einen Heartbeat von dem verbundenen OCPP-Client. Um auf den Heartbeat zu antworten, muss die Methode RespHeartbeat [▶ 154] aufgerufen werden.



Syntax

```
METHOD RecvHeartbeat : BOOL
VAR_OUTPUT
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvHeartbeat	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ausgänge

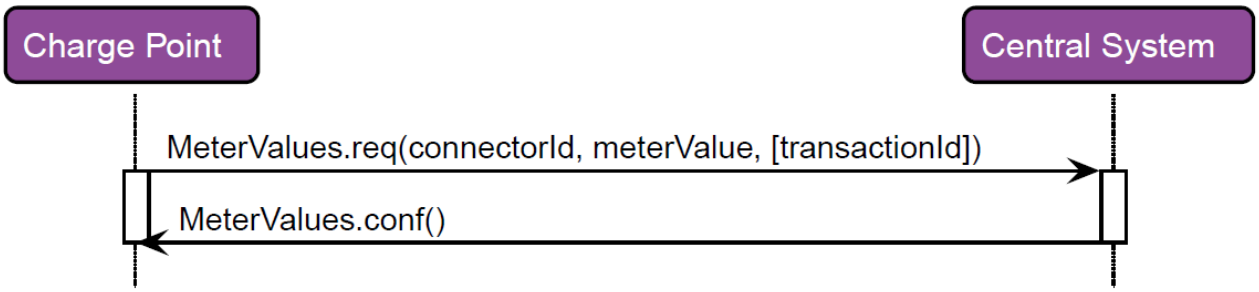
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.11 RecvMeterValue

Mit dieser Methode empfängt ein OCPP-Server Meter Values von dem verbundenen OCPP-Client. Um auf den Erhalt der Meter Values zu antworten, muss die Methode RespMeterValue [▶ 155] aufgerufen werden.

Es ist entgegen der Spezifikation möglich, Meter Value-Nachrichten ohne Meter Values zu erhalten, um die Kompatibilität mit anderen Herstellern zu erhöhen.



Syntax

```
METHOD RecvMeterValue : BOOL
VAR_IN_OUT
    arrSampleValues : ARRAY[*] OF ST_OCPP1_SampledValue;
END_VAR
VAR_OUTPUT
    hMessageId      : T_OCPP_MessageId;
    nConnectorId    : UDINT;
    nTransactionId  : UDINT;
    nSampleCount    : UDINT;
```

Rückgabewert

Name	Typ	Beschreibung
RecvMeterValue	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ein-/Ausgänge

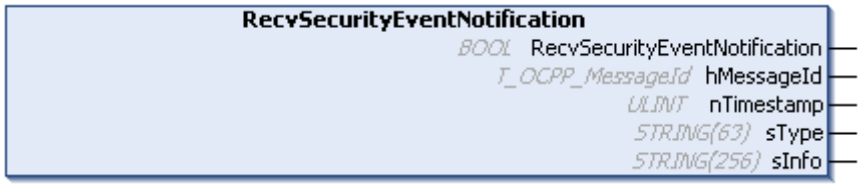
Name	Typ	Beschreibung
arrSampleValues	ARRAY [*] OF ST_OCPP1_SampledValue [▶ 210]	Die vom Client gesendeten Messwerte.

Ausgänge

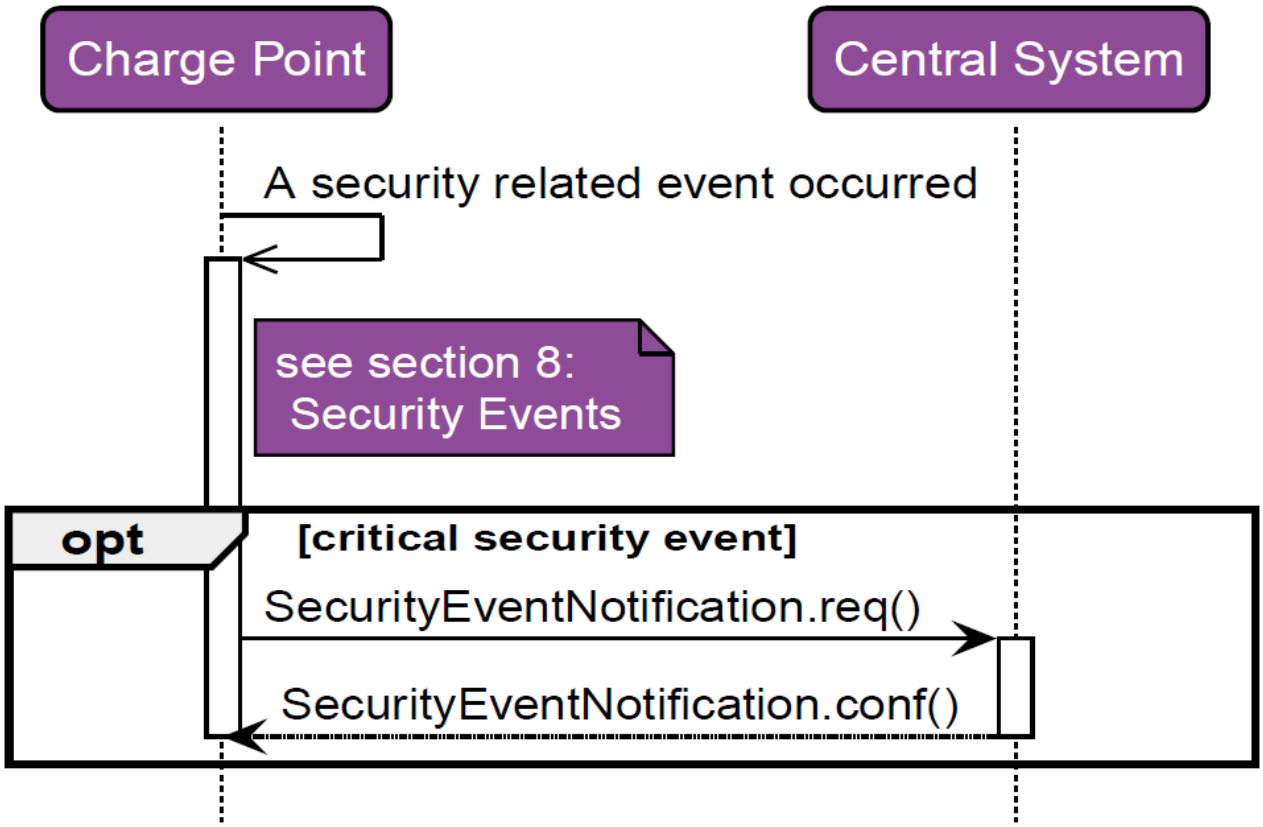
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
nTransactionId	UDINT	Die Transaktion, zu der die Messwerte gehören.
nSampleCount	UDINT	Die Anzahl der enthaltenen Messwerte.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.12 RecvSecurityEventNotification



Mit dieser Methode empfängt ein OCPP-Server eine Security Event Notification von dem verbundenen OCPP-Client. Um auf die Security Event Notification zu antworten, muss die Methode RespSecurityEventNotification [▶ 156] aufgerufen werden.



Syntax

```
METHOD RecvSecurityEventNotification : BOOL
VAR_OUTPUT
    hMessageId : T_OCPP_MessageId;
    nTimestamp : ULINT;
    sType      : STRING(63);
    sInfo      : STRING(256);
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvSecurityEventNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

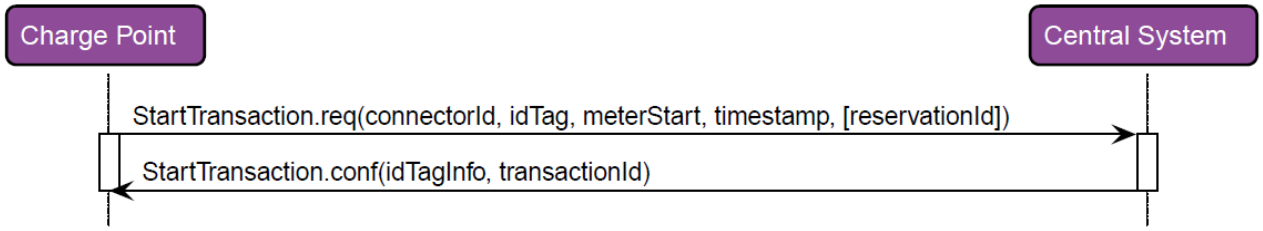
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nTimestamp	ULINT	Datum und Zeit beim Auftreten des Events.
sType	STRING(63)	Typ des Security-Events laut OCPP-Spezifikation.
sInfo	STRING(256)	Zusätzliche Informationen über das aufgetretene Security Event.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.13 RecvStartTransaction

Mit dieser Methode empfängt ein OCPP-Server einen Start Transaction-Request von dem verbundenen OCPP-Client. Um auf den Request zu antworten, muss die Methode RespStartTransaction [▶ 157] aufgerufen werden.



Syntax

```
METHOD RecvStartTransaction : BOOL
VAR_OUTPUT
    hMessageId     : T_OCPP_MessageId;
    sIdTag         : T_OCPP1_IdToken;
    nConnectorId   : UDINT;
    nMeterStart    : UDINT;
    nReservationId : UDINT;
    nTimestamp     : ULINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvStartTransaction	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

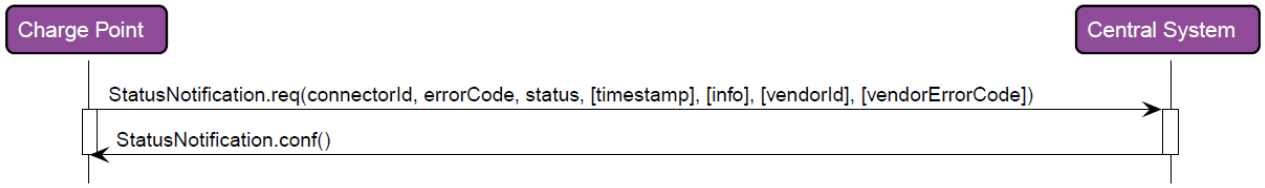
Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
sIdTag	T_OCPP1_IdToken [▶ 212]	ID-Token, mit dem die Transaktion gestartet werden soll.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
nMeterStart	UDINT	Wert in Wattstunden beim Start der Transaktion.
nReservationId	UDINT	Optionale Reservierungs-ID.
nTimestamp	ULINT	Datum und Zeit beim Start der Transaktion.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.14 RecvStatusNotification

Mit dieser Methode empfängt ein OCPP-Server eine Status Notification von dem verbundenen OCPP-Client. Um auf die Status Notification zu antworten, muss die Methode RespStatusNotification [▶ 157] aufgerufen werden.



Syntax

```
METHOD RecvStatusNotification : BOOL
VAR_OUTPUT
    hMessageId    : T_OCPP_MessageId;
    nConnectorId  : UDINT;
    eError        : E_OCPP1_ChargePointError;
    eStatus       : E_OCPP1_ChargePointStatus;
    nTimestamp    : ULINT := 0;
    sInfo         : STRING(63) := '';
    sVendorError  : STRING(63) := '';
    sVendorId     : STRING(255) := '';
END_VAR
```

Rückgabewert

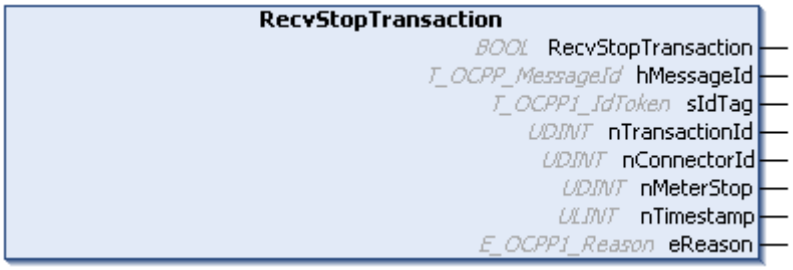
Name	Typ	Beschreibung
RecvStatusNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ausgänge

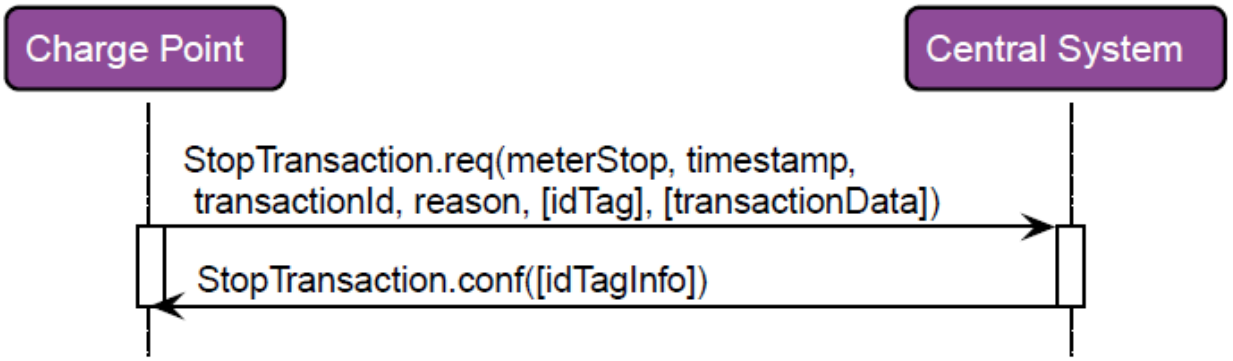
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
eError	E_OCPP1_ChargePointError [▶ 192]	Error Code, der in der Status Notification empfangen wurde.
eStatus	E_OCPP1_ChargePointStatus [▶ 192]	Status, der in der Status Notification empfangen wurde.
nTimestamp	ULINT	Der Zeitstempel der Status Notification.
sInfo	STRING(63)	Enthält optional frei definierbare Zusatzinformationen zum Fehler.
sVendorError	STRING(63)	Enthält optional den herstellerspezifischen Fehlercode.
sVendorId	STRING(255)	Enthält optional den Identifier für die herstellerspezifische Implementierung.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.15 RecvStopTransaction



Mit dieser Methode empfängt ein OCPP-Server einen Stop Transaction-Request von dem verbundenen OCPP-Client. Um auf den Request zu antworten, muss die Methode RespStopTransaction [▶ 158] aufgerufen werden.



Syntax

```
METHOD RecvStopTransaction : BOOL
VAR_OUTPUT
    hMessageId     : T_OCPP_MessageId;
    sIdTag         : T_OCPP1_IdToken;
    nTransactionId : UDINT;
```

```
    nConnectorId    : UDINT;
    nMeterStop      : UDINT;
    nTimestamp      : ULINT;
    eReason         : E_OCPP1_Reason;
END_VAR
```

Rückgabewert

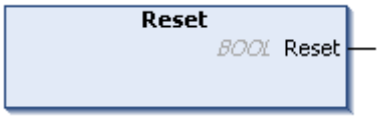
Name	Typ	Beschreibung
RecvStopTransaction	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ausgänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
sIdTag	T_OCPP1_IdToken [▶ 212]	ID-Token, für das die Transaktion gestoppt werden soll.
nTransactionId	UDINT	ID der Transaktion, die gestoppt werden soll. Ist in der Antwort des Central Systems auf einen Start Transaction-Request enthalten.
nConnectorId	UDINT	ID des Connectors eines Charge Points.
nMeterStop	UDINT	Wert in Wattstunden am Ende der Transaktion.
nTimestamp	ULINT	Datum und Zeit beim Stoppen der Transaktion.
eReason	E_OCPP1_Reason [▶ 197]	Kann optional den Grund für das Stoppen der Transaktion enthalten.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.16 Reset



Mit dieser Methode wird der zuletzt am Baustein anliegende Fehler zurückgesetzt.

Syntax

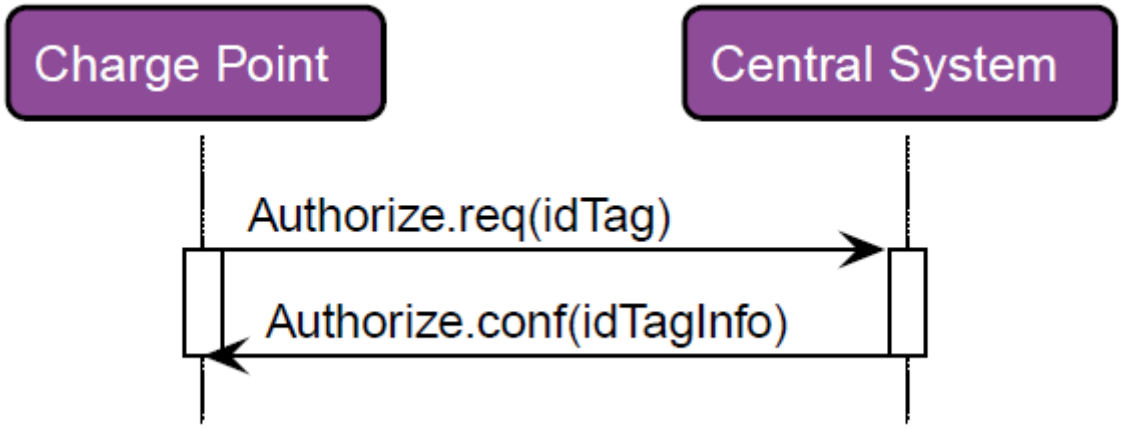
```
METHOD Reset : BOOL
```

Rückgabewert

Name	Typ	Beschreibung
Reset	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

5.1.3.17 RespAuthorize

Mit dieser Methode antwortet ein OCPP-Server auf einen Authorize-Request von dem verbundenen OCPP-Client.



Syntax

```
METHOD RespAuthorize : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_AuthorizationStatus;
END_VAR
```

Rückgabewert

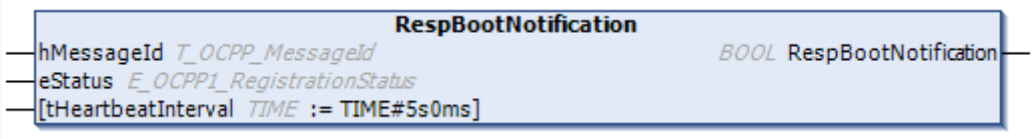
Name	Typ	Beschreibung
RespAuthorize	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

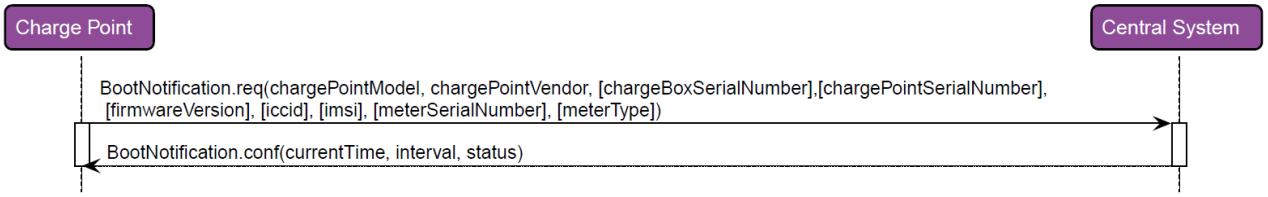
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_AuthorizationStatus [▶ 191]	Status der Autorisierung als Antwort an den OCPP-Client.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.18 RespBootNotification



Mit dieser Methode antwortet ein OCPP-Server auf eine Boot Notification von dem verbundenen OCPP-Client.



Syntax

```
METHOD RespBootNotification : BOOL
VAR_INPUT
    hMessageId          : T_OCPP_MessageId;
    eStatus             : E_OCPP1_RegistrationStatus;
    tHeartbeatInterval : TIME := T#5S;
END_VAR
```

Rückgabewert

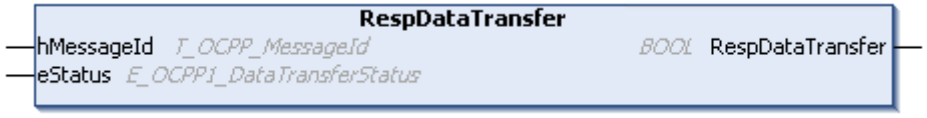
Name	Typ	Beschreibung
RespBootNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

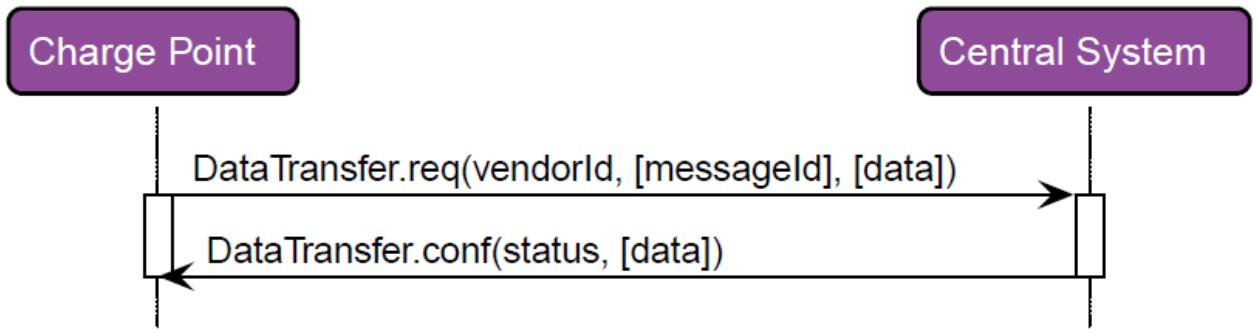
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_RegistrationStatus [▶ 197]	Status der Registrierung als Antwort an den OCPP-Client.
tHeartbeatInterval	TIME	Wenn der Registration Status Accepted ist, enthält diese Variable das Heartbeat Interval für die Verbindung.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.19 RespDataTransfer



Mit dieser Methode antwortet ein OCPP-Server auf einen Data Transfer-Request von dem verbundenen OCPP-Client.



Syntax

```
METHOD RespDataTransfer : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_DataTransferStatus;
END_VAR
```

Rückgabewert

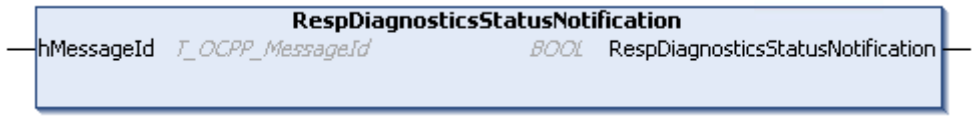
Name	Typ	Beschreibung
RespDataTransfer	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

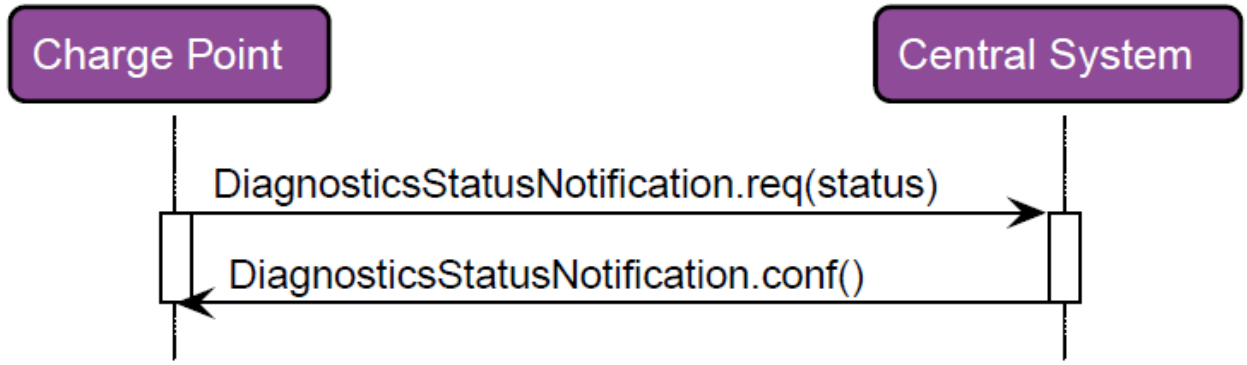
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_DataTransferStatus [▶ 194]	Status des Data Transfers als Antwort an den OCPP-Client.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.20 RespDiagnosticsStatusNotification



Mit dieser Methode antwortet ein OCPP-Server auf eine Status Notification von dem verbundenen OCPP-Client.



Syntax

```
METHOD RespDiagnosticsStatusNotification : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RespDiagnosticsStatusNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.21 RespError

Mit dieser Methode antwortet ein OCPP-Server dem verbundenen OCPP-Client, wenn der gesendete Request einen internen Fehler ausgelöst hat. Das kann beispielsweise der Fall sein, wenn eine angefragte Aktion nicht verfügbar ist.

Syntax

```
METHOD RespError : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    eAct       : E_OCPP1_Action := E_OCPP1_Action.Heartbeat;
    eErr       : E_OCPP1_Error := E_OCPP1_Error.None;
END_VAR
```

Rückgabewert

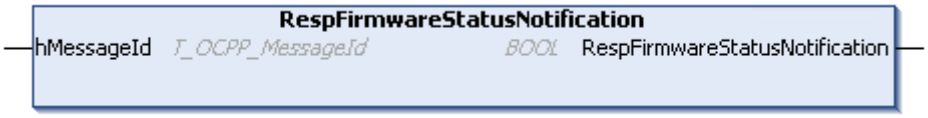
Name	Typ	Beschreibung
RespError	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

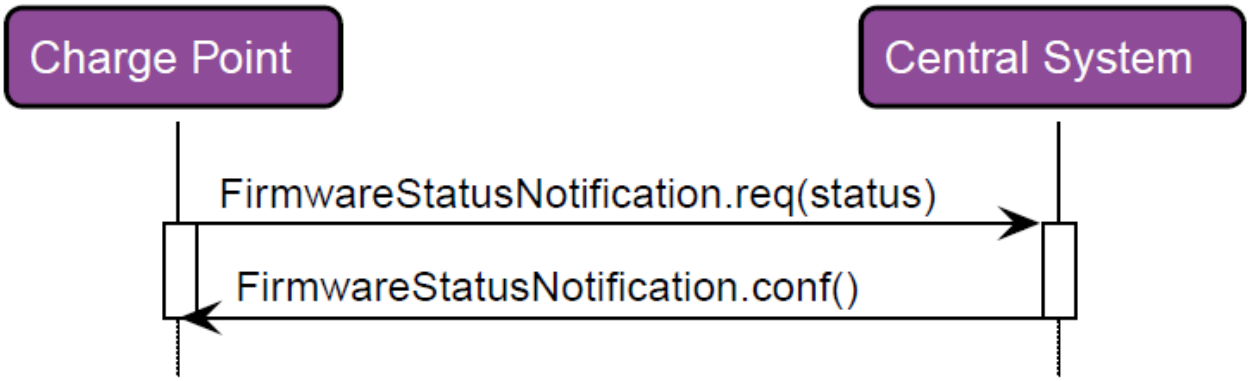
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eAct	E_OCPP1_Action [▶ 190]	Art des OCPP-Requests, bei dem der Fehler aufgetreten ist.
eErr	E_OCPP1_Error [▶ 195]	OCPP-Error, mit dem der Request beantwortet werden soll.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.22 RespFirmwareStatusNotification



Mit dieser Methode antwortet ein OCPP-Server auf eine Firmware Status Notification von dem verbundenen OCPP-Client.



Syntax

```
METHOD RespFirmwareStatusNotification : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

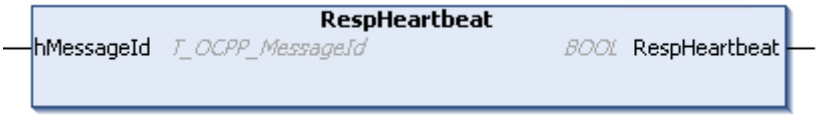
Name	Typ	Beschreibung
RespFirmwareStatusNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

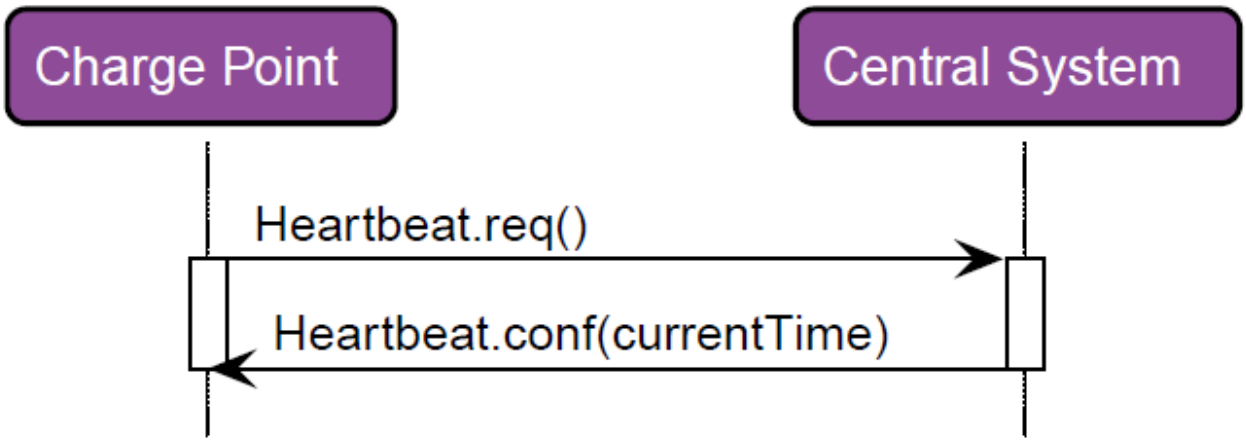
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.23 RespHeartbeat



Mit dieser Methode antwortet ein OCPP-Server auf einen Heartbeat von dem verbundenen OCPP-Client.



Syntax

```
METHOD RespHeartbeat : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RespHeartbeat	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

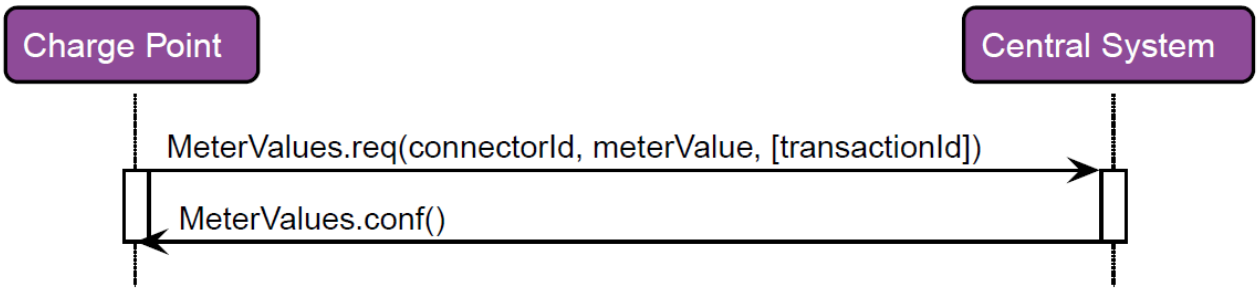
Eingänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.24 RespMeterValue

Mit dieser Methode antwortet ein OCPP-Server auf das Senden von Meter Values von dem verbundenen OCPP-Client.



Syntax

```
METHOD RespMeterValue : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

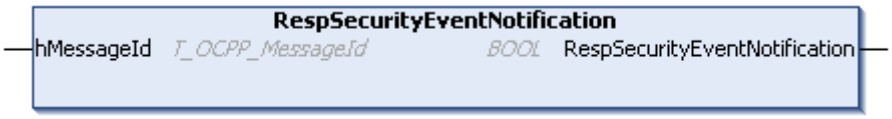
Name	Typ	Beschreibung
RespMeterValue	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

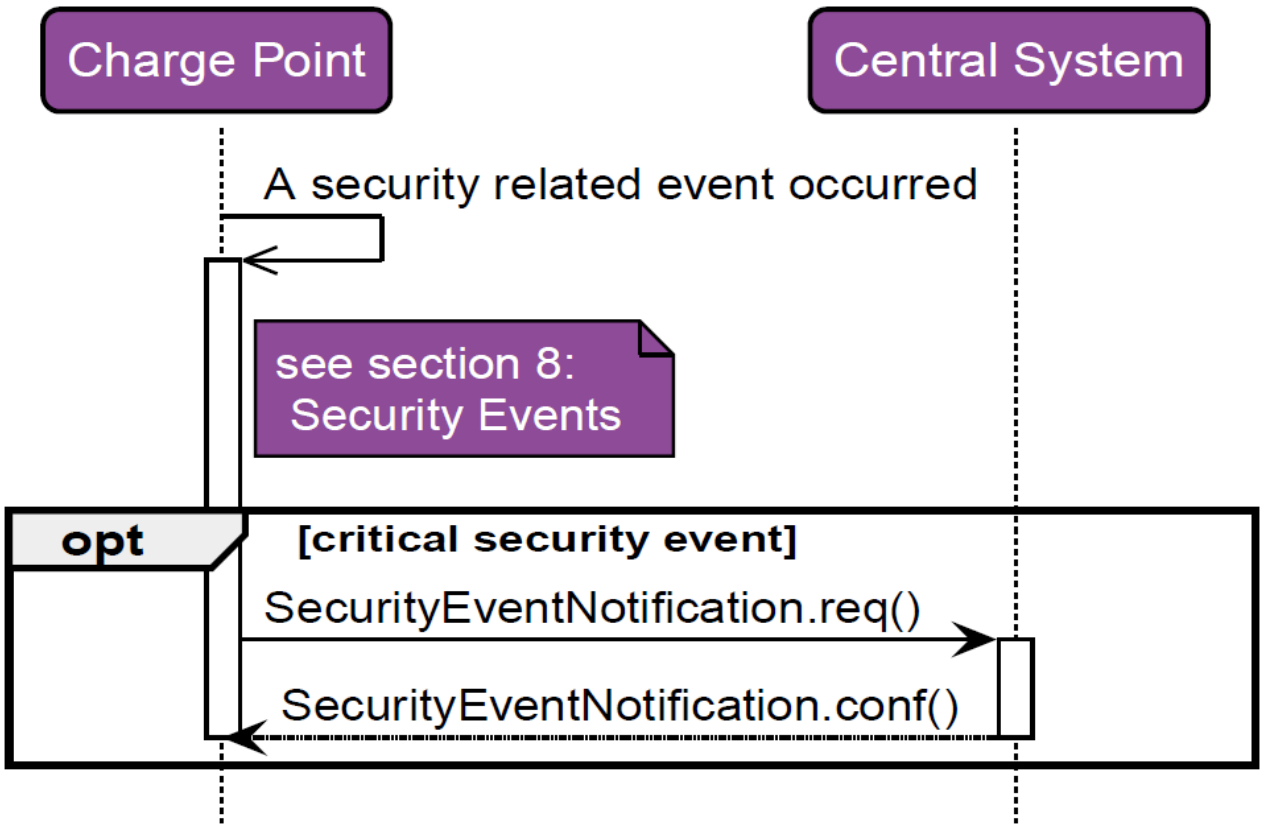
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.25 RespSecurityEventNotification



Mit dieser Methode antwortet ein OCPP-Server auf eine Security Event Notification von dem verbundenen OCPP-Client.



Syntax

```
METHOD RespSecurityEventNotification : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RespSecurityEventNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

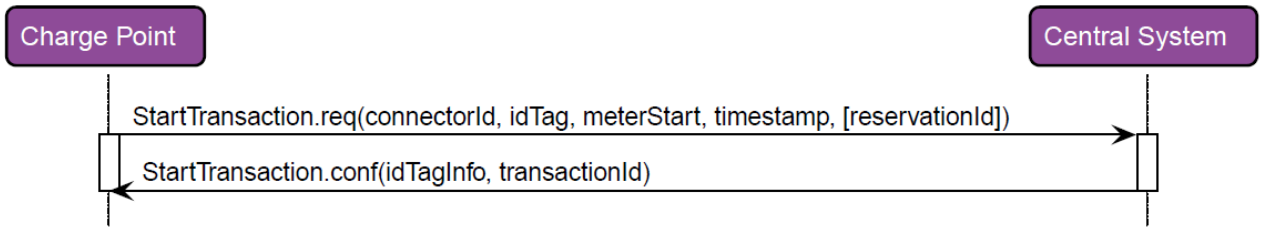
Eingänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.26 RespStartTransaction

Mit dieser Methode antwortet ein OCPP-Server auf einen Start Transaction-Request von dem verbundenen OCPP-Client.



Syntax

```
METHOD RespStartTransaction : BOOL
VAR_INPUT
    hMessageId     : T_OCPP_MessageId;
    eStatus        : E_OCPP1_AuthorizationStatus;
    nTransactionId : UDINT;
END_VAR
```

Rückgabewert

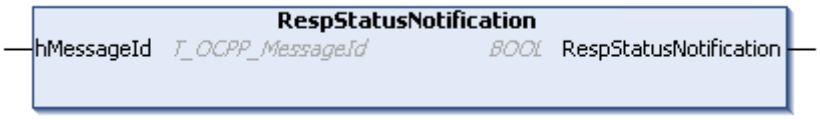
Name	Typ	Beschreibung
RespStartTransaction	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

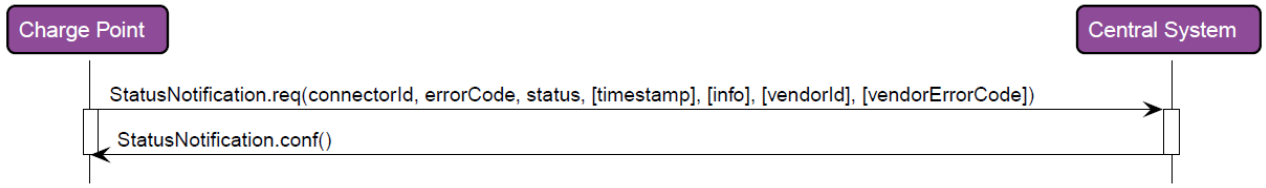
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_AuthorizationStatus [▶ 191]	Status der Autorisierung als Antwort an den OCPP-Client.
nTransactionId	UDINT	Vom Central System festgelegte ID für die Transaktion.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.27 RespStatusNotification



Mit dieser Methode antwortet ein OCPP-Server auf eine Status Notification von dem verbundenen OCPP-Client.



Syntax

```
METHOD RespStatusNotification : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RespStatusNotification	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

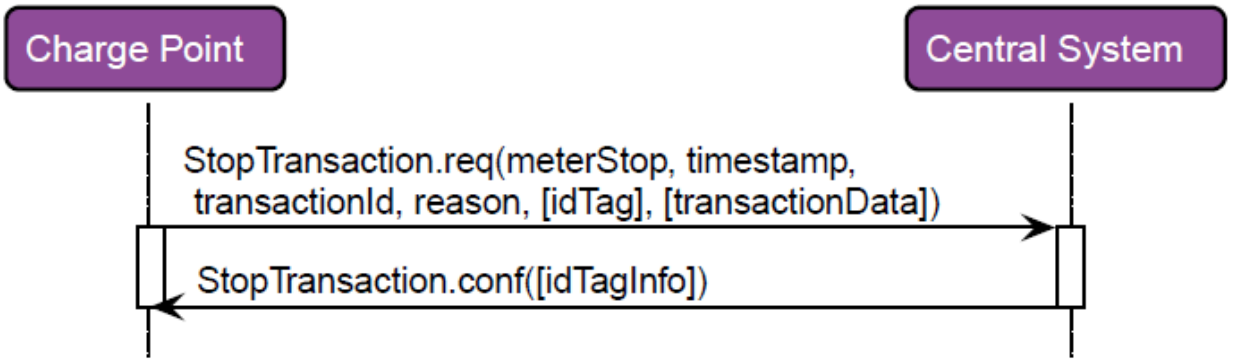
Eingänge

Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.28 RespStopTransaction

Mit dieser Methode antwortet ein OCPP-Server auf einen Stop Transaction-Request von dem verbundenen OCPP-Client.



Syntax

```
METHOD RespStopTransaction : BOOL
VAR_INPUT
    hMessageId : T_OCPP_MessageId;
    eStatus    : E_OCPP1_AuthorizationStatus;
END_VAR
```

Rückgabewert

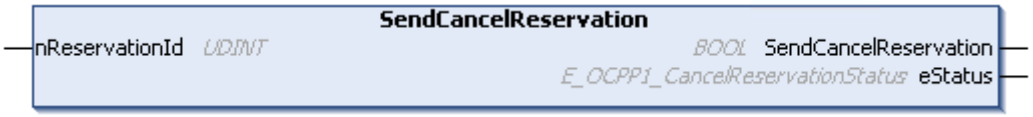
Name	Typ	Beschreibung
RespStopTransaction	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

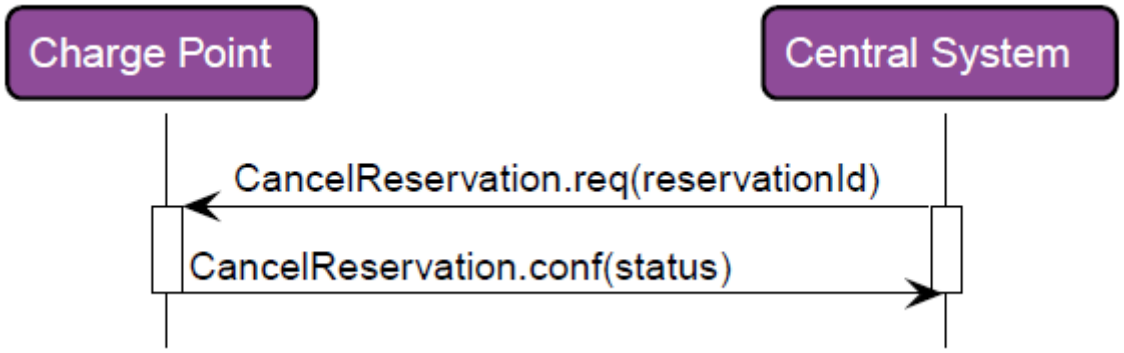
Name	Typ	Beschreibung
hMessageId	T_OCPP_MessageId [▶ 212]	MessageId der empfangenen Nachricht.
eStatus	E_OCPP1_AuthorizationStatus [▶ 191]	Status der Autorisierung als Antwort an den OCPP-Client.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.29 SendCancelReservation



Mit dieser Methode sendet ein OCPP-Server einen Cancel Reservation-Request an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendCancelReservation : BOOL
VAR_INPUT
    nReservationId : UDINT;
END_VAR
VAR_OUTPUT
    eStatus        : E_OCPP1_CancelReservationStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendCancelReservation	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

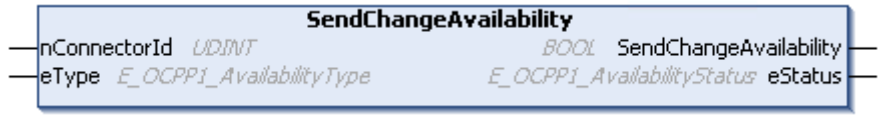
Name	Typ	Beschreibung
nReservationId	UDINT	ID der zu stornierenden Reservierung.

Ausgänge

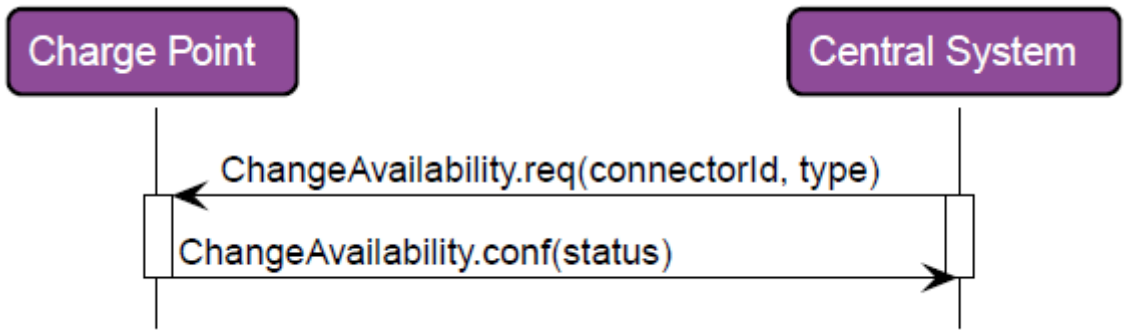
Name	Typ	Beschreibung
eStatus	E_OCPP1_CancelReservationStatus [▶ 192]	Der Status zeigt an, ob die Stornierung der Reservierung erfolgreich war.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.30 SendChangeAvailability



Mit dieser Methode sendet ein OCPP-Server einen Change Availability-Request an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendChangeAvailability : BOOL
VAR_INPUT
    nConnectorId : UDINT;
    eType        : E_OCPP1_AvailabilityType;
END_VAR
VAR_OUTPUT
    eStatus      : E_OCPP1_AvailabilityStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendChangeAvailability	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

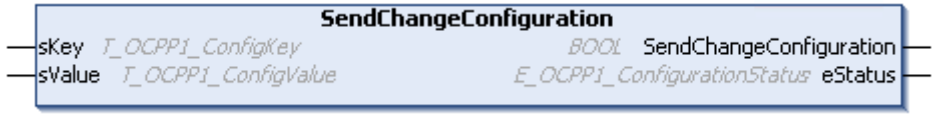
Name	Typ	Beschreibung
nConnectorId	UDINT	ID des Connectors eines Charge Points.
eType	E_OCPP1_AvailabilityType [▶ 192]	Typ der Verfügbarkeitsänderung, die der Charge Point durchführen soll.

Ausgänge

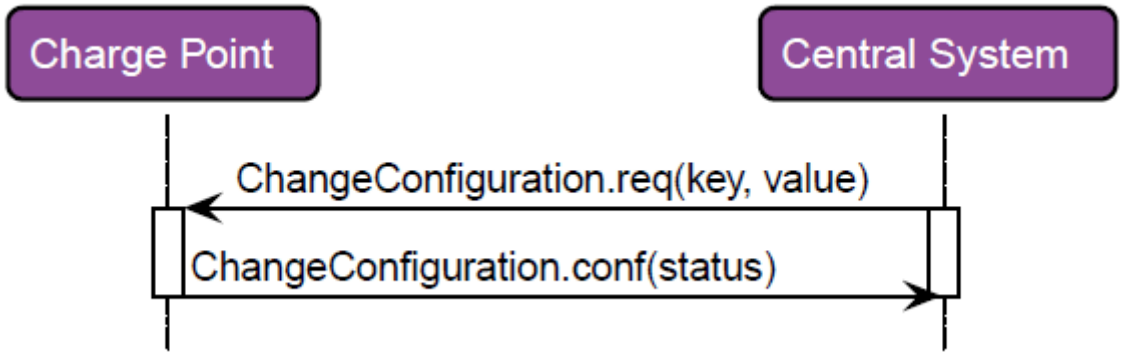
Name	Typ	Beschreibung
eStatus	E_OCPP1_AvailabilityStatus [▶ 191]	Antwort, ob der Charge Point die angefragte Verfügbarkeitsänderung umsetzen konnte.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.31 SendChangeConfiguration



Mit dieser Methode sendet ein OCPP-Server einen Change Configuration-Request an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendChangeConfiguration : BOOL
VAR_INPUT
    sKey        : T_OCPP1_ConfigKey;
    sValue      : T_OCPP1_ConfigValue;
END_VAR
```

```
VAR_OUTPUT
    eStatus    : E_OCPP1_ConfigurationStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendChangeConfiguration	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
sKey	T_OCPP1_ConfigKey [▶ 211]	Der Name der zu ändernden Konfigurationseinstellung.
sValue	T_OCPP1_ConfigValue [▶ 211]	Der neue Wert der Konfigurationseinstellung.

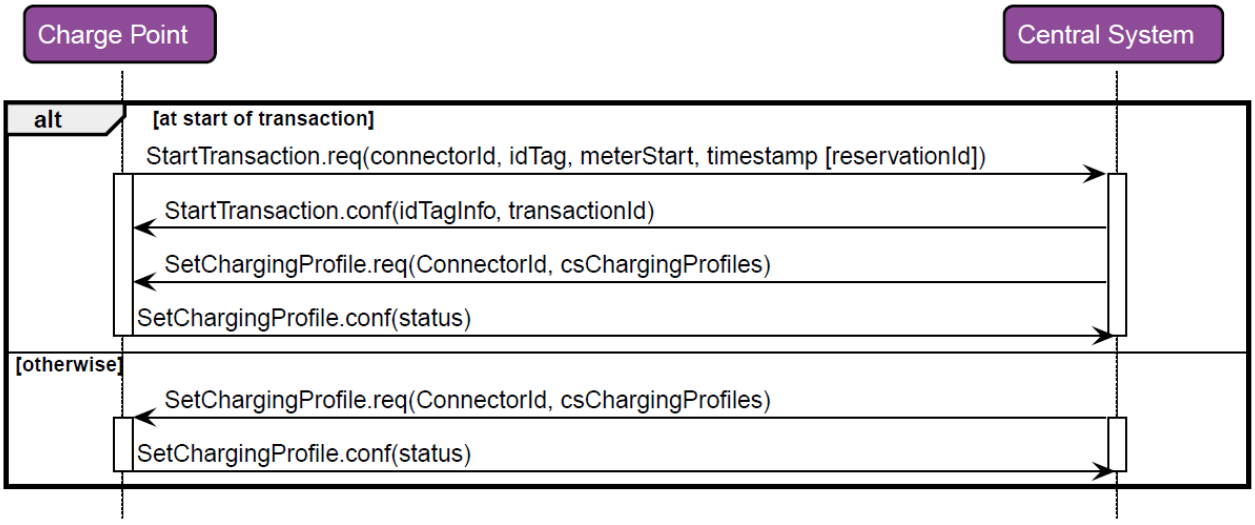
Ausgänge

Name	Typ	Beschreibung
eStatus	E_OCPP1_ConfigurationStatus [▶ 194]	Der Status zeigt an, ob die Konfigurationsänderung akzeptiert wurde.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.32 SendChargingProfile

Mit dieser Methode sendet ein OCPP-Server ein Charging Profile an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendChargingProfile : BOOL
VAR_INPUT
    nConnectorId     : UDINT;
    mChargingProfile : REFERENCE TO ST_OCPP1_ChargingProfileMax;
END_VAR
VAR_OUTPUT
    eStatus          : E_OCPP1_ChargingProfileStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendChargingProfile	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

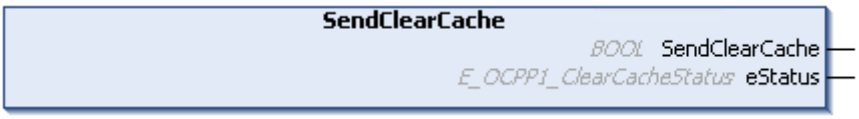
Name	Typ	Beschreibung
nConnectorId	UDINT	ID des Connectors eines Charge Points.
mChargingProfile	REFERENCE TO ST_OCPP1_ChargingProfileMax [▶ 206]	An den Client zu sendendes Charging Profile.

Ausgänge

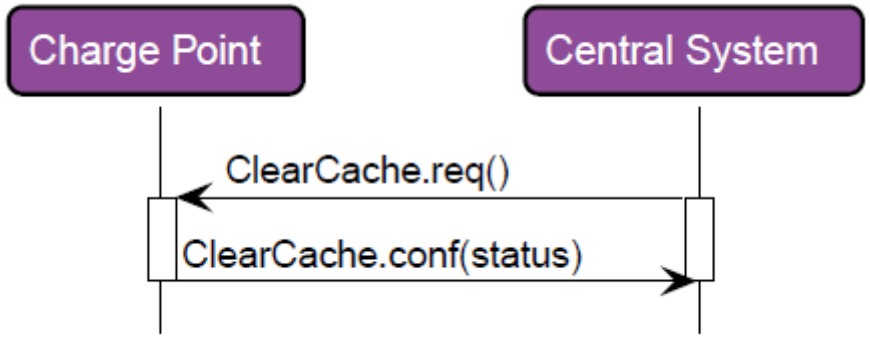
Name	Typ	Beschreibung
eStatus	E_OCPP1_ChargingProfileStatus [▶ 193]	Der Status zeigt an, ob das Charging Profile von dem Charge Point akzeptiert wurde.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.33 SendClearCache



Mit dieser Methode sendet ein OCPP-Server einen Clear Cache-Request an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendClearCache : BOOL
VAR_OUTPUT
    eStatus    : E_OCPP1_ClearCacheStatus;
END_VAR
```

Rückgabewert

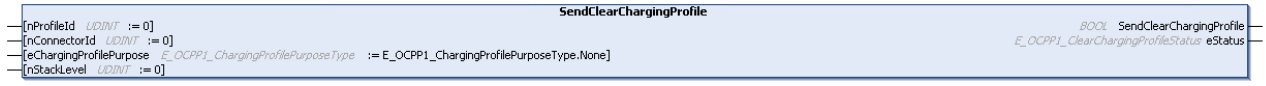
Name	Typ	Beschreibung
SendClearCache	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ausgänge

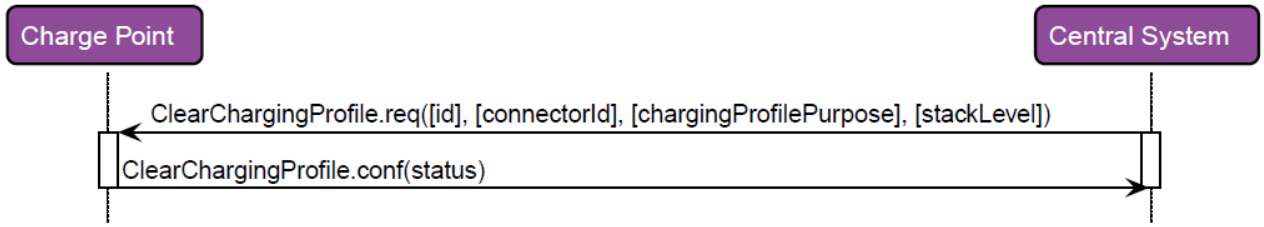
Name	Typ	Beschreibung
eStatus	E_OCPP1_ClearCacheStatus [▶ 193]	Der Status zeigt an, ob das Clearen des Caches Profile von dem Charge Point akzeptiert wurde.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.34 SendClearChargingProfile



Mit dieser Methode sendet ein OCPP-Server einen Clear Charging Profile-Request an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendClearChargingProfile : BOOL
VAR_INPUT
    nProfileId              : UDINT := 0;
    nConnectorId            : UDINT := 0;
    eChargingProfilePurpose : E_OCPP1_ChargingProfilePurposeType := E_OCPP_ChargingProfilePurposeTyp
e.None;
    nStackLevel             : UDINT := 0;
END_VAR
VAR_OUTPUT
    eStatus                 : E_OCPP1_ClearChargingProfileStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendClearChargingProfile	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

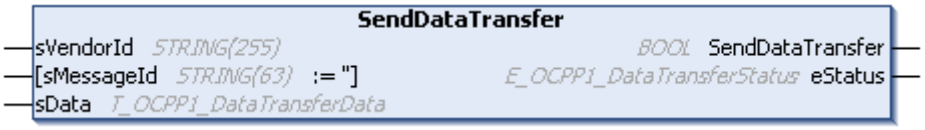
Name	Typ	Beschreibung
nProfileId	UDINT	Identifier des zu löschenden Charging Profile.
nConnectorId	UDINT	ID des Connectors eines Charge Points. Der Wert 0 besagt, dass das Löschen sich auf den gesamten Charge Point bezieht.
eChargingProfilePurpose	E_OCPP1_ChargingProfilePurposeType [▶ 193]	Spefiziert den Zweck der zu löschenden Charging Profiles.
nStackLevel	UDINT	Spezifiziert das StackLevel, für das Charging Profile gelöscht werden.

Ausgänge

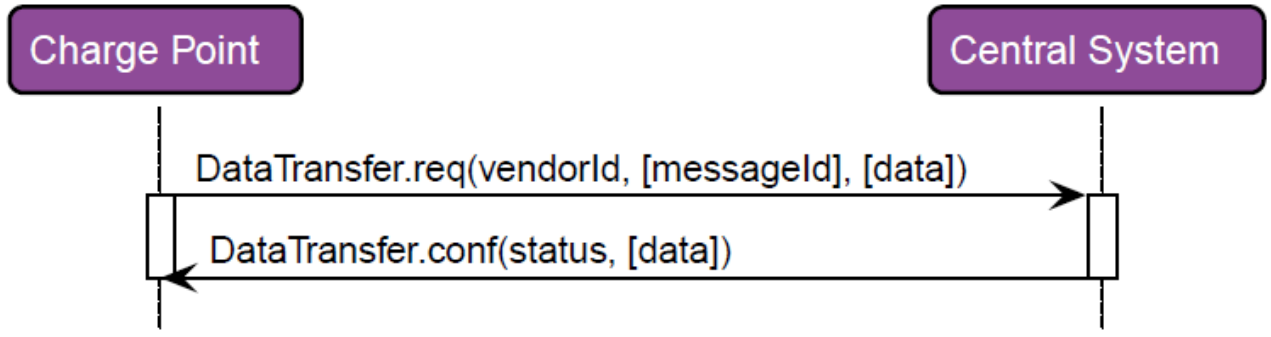
Name	Typ	Beschreibung
eStatus	E_OCPP1_ClearChargingProfileStatus [▶ 193]	Der Status zeigt an, ob das Clearen des Charging Profiles von dem Charge Point durchgeführt werden konnte.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.35 SendDataTransfer



Mit dieser Methode sendet ein OCPP-Server einen DataTransfer-Request an den entsprechenden OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendDataTransfer : BOOL
VAR_INPUT
    sVendorId  : STRING(255);
    sMessageId : STRING(63) := '';
END_VAR
VAR_IN_OUT CONSTANT
    sData      : T_OCPP1_DataTransferData;
END_VAR
VAR_OUTPUT
    eStatus    : E_OCPP1_DataTransferStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendDataTransfer	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
sVendorId	STRING(255)	Identifier für den Hersteller, der die herstellerspezifische Implementierung kennzeichnet.
sMessageId	STRING(63)	Zusätzliches Identifikationsfeld für eine einzelne Nachricht.

Ein-/Ausgänge

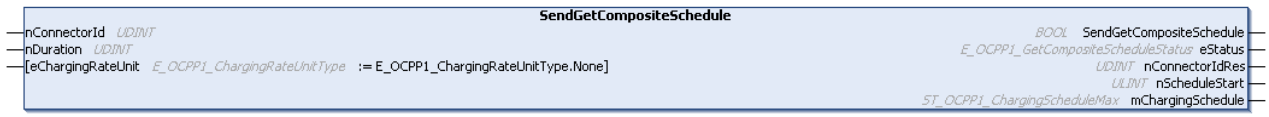
Name	Typ	Beschreibung
sData	T_OCPP1_DataTransferData [▶ 211]	Text ohne spezifizierte Länge und Format.

Ausgänge

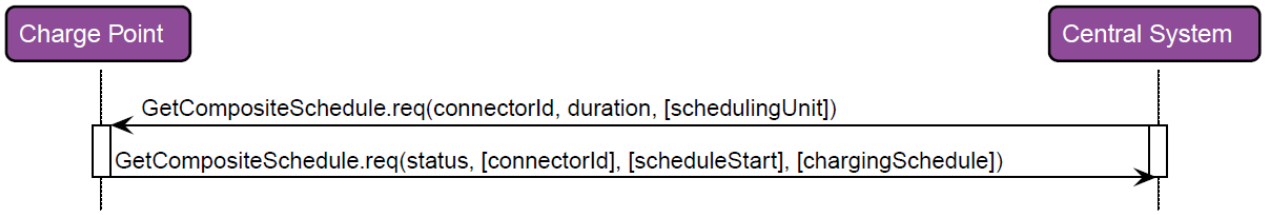
Name	Typ	Beschreibung
eStatus	E_OCPP1_DataTransferStatus [▶ 194]	Status des Data Transfers.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.36 SendGetCompositeSchedule



Mit dieser Methode sendet ein OCPP-Server einen Get Composite Schedule-Request an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendGetCompositeSchedule : BOOL
VAR_INPUT
    nConnectorId      : UDINT;
    nDuration         : UDINT;
    eChargingRateUnit : E_OCPP1_ChargingRateUnitType := E_OCPP_ChargingRateUnitType.None;
END_VAR
VAR_OUTPUT
    eStatus           : E_OCPP1_GetCompositeScheduleStatus;
    nConnectorIdRes   : UDINT;
    nScheduleStart    : ULINT := 0;
    mChargingSchedule : ST_OCPP1_ChargingScheduleMax;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendGetCompositeSchedule	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
nConnectorId	UDINT	ID des Connectors eines Charge Points.
nDuration	UDINT	Länge des angefragten Zeitplans.
eChargingRateUnit	E_OCPP1_ChargingRateUnitType [▶ 193]	Wird zur Angabe der Einheit für die Anfrage verwendet.

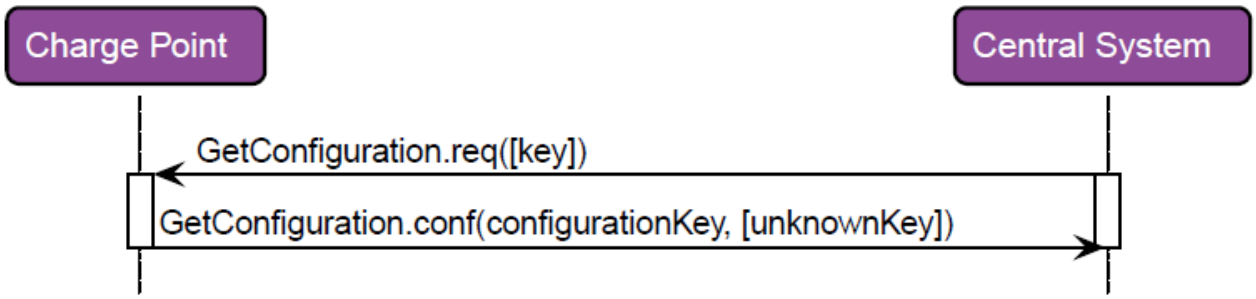
Ausgänge

Name	Typ	Beschreibung
eStatus	E_OCPP1_GetCompositeScheduleStatus [▶ 195]	Der Status zeigt an, ob die Anfrage des Zeitplans von dem Charge Point beantwortet werden konnte.
nConnectorIdRes	UDINT	Identifier des Connectors, auf den sich der Zeitplan in der Antwort bezieht.
nScheduleStart	ULINT	Startzeit des Zeitplans.
mChargingSchedule	ST_OCPP1_ChargingScheduleMax [▶ 206]	Angefragter Zeitplan.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.37 SendGetConfiguration

Mit dieser Methode sendet ein OCPP-Server einen Get Configuration-Request an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendGetConfiguration : BOOL
VAR_IN_OUT
    arrKeys       : ARRAY[*] OF ST_OCPP1_ConfigKeyValue;
END_VAR
VAR_OUTPUT
    nKeysCount    : UDINT := 0;
    nUnknownCount : UDINT := 0;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendGetConfiguration	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Ein-/Ausgänge

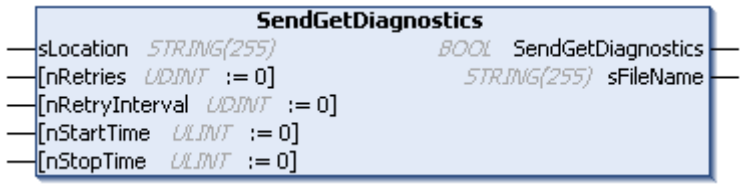
Name	Typ	Beschreibung
arrKeys	ARRAY [*] OF ST_OCPP1_ConfigKeyValue [▶ 210]	Liste von angefragten oder bekannten Konfigurationsschlüsseln, für die die Konfiguration angefragt wird.

Ausgänge

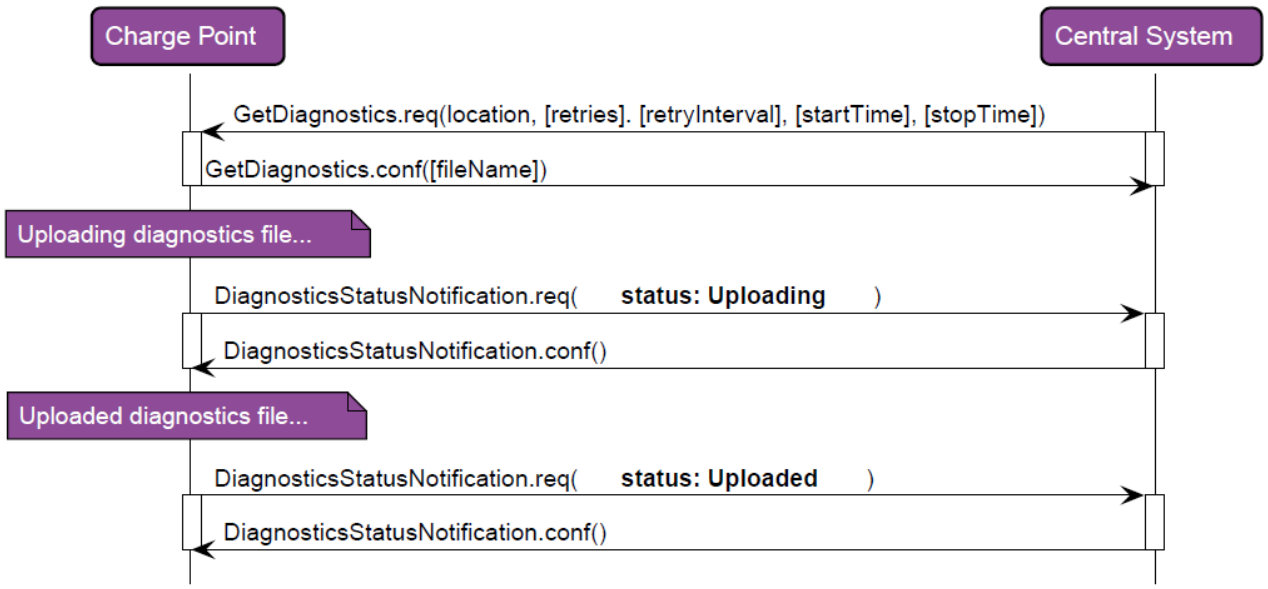
Name	Typ	Beschreibung
nKeysCount	UDINT	Anzahl der folgenden Konfigurationsschlüssel.
nUnknownCount	UDINT	Anzahl der folgenden unbekannten Konfigurationsschlüssel.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.38 SendGetDiagnostics



Mit dieser Methode sendet ein OCPP-Server einen Get Diagnostics-Request an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendGetDiagnostics : BOOL
VAR_INPUT
    sLocation      : STRING(255);
    nRetries       : UDINT := 0;
    nRetryInterval : UDINT := 0;
    nStartTime     : ULINT := 0;
    nStopTime      : ULINT := 0;
END_VAR
VAR_OUTPUT
    sFileName      : STRING(255);
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendGetDiagnostics	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
sLocation	STRING(255)	Verzeichnis, in das die Diagnosedatei hochgeladen werden soll.
nRetries	UDINT	Anzahl der Versuche, die der Charge Point bei fehlschlagendem Upload unternimmt.
nRetryInterval	UDINT	Intervall, nach dem ein neuer Upload versucht wird.
nStartTime	ULINT	Startzeit der in der Diagnose enthaltenen Logging-Informationen.
nStopTime	ULINT	Endzeit der in der Diagnose enthaltenen Logging-Informationen.

Ausgänge

Name	Typ	Beschreibung
sFileName	STRING(255)	Enthält den Namen der Diagnosedatei, die hochgeladen werden wird.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.39 SendGetLocalListVersion

Mit dieser Methode sendet ein OCPP-Server einen Get Local List Version-Request an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.

Syntax

```
METHOD SendGetLocalListVersion : BOOL
VAR_OUTPUT
    nListVersion : UDINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendGetLocalListVersion	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

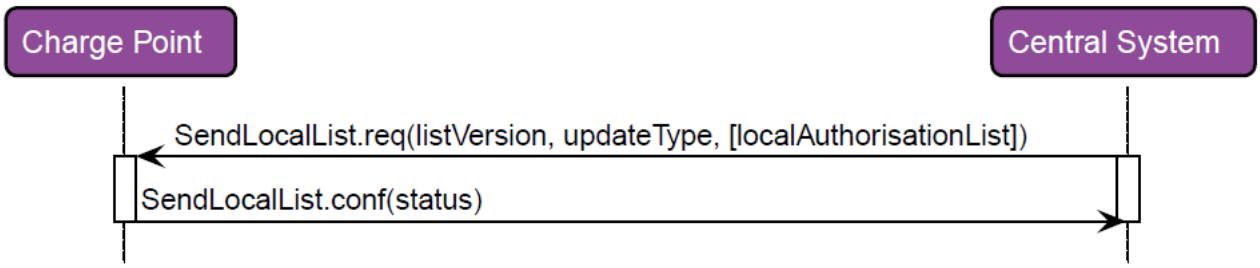
Ausgänge

Name	Typ	Beschreibung
nListVersion	UDINT	Enthält die Versionsnummer der aktuell im Charge Point vorliegenden Local Authorization List.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.40 SendLocalList

Mit dieser Methode sendet ein OCPP-Server einen Send Local List-Request an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendLocalList : BOOL
VAR_INPUT
    nListVersion : UDINT;
    eUpdateType  : E_OCPP1_UpdateType;
    nAuthCount   : UDINT := 0;
END_VAR
VAR_IN_OUT
    arrAuthList  : ARRAY[*] OF ST_OCPP1_AuthorizationData;
END_VAR
VAR_OUTPUT
    eStatus      : E_OCPP1_UpdateStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendLocalList	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
nListVersion	UDINT	Version der als Update bereitgestellten Local Authorization List.
eUpdateType	E_OCPP1_UpdateType [▶ 199]	Festlegung des Typs des Updates, entweder als Update oder als vollständiger Austausch.
nAuthCount	UDINT	Anzahl an Einträgen in der Local Authorization List.

Ein-/Ausgänge

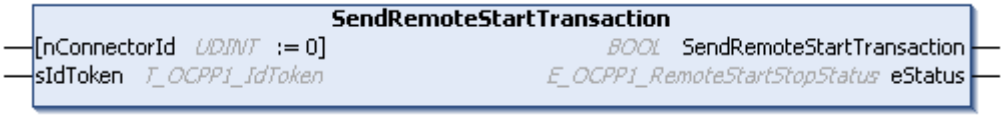
Name	Typ	Beschreibung
arrAuthList	ARRAY [*] OF ST_OCPP1_AuthorizationData [▶ 204]	Entweder die Werte der neuen Local Authorization List (vollständiger Austausch) oder hinzuzufügende Werte der bestehenden Local Authorization List (Update).

Ausgänge

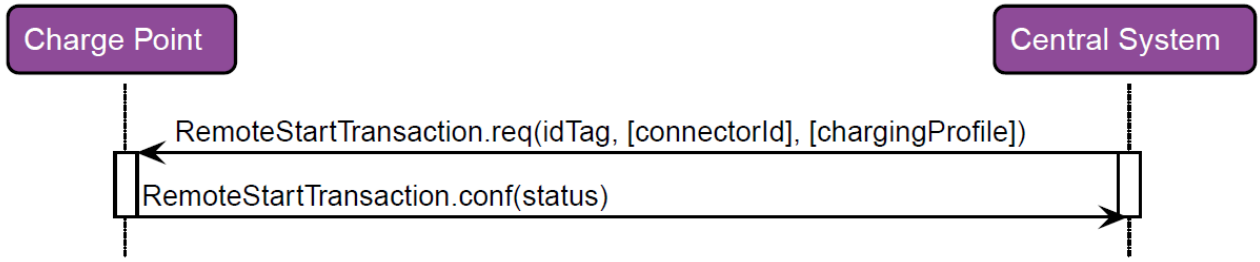
Name	Typ	Beschreibung
eStatus	E_OCPP1_UpdateStatus [▶ 198]	Der Status zeigt an, ob der Charge Point das Update der Local Authorization List erhalten und durchgeführt hat.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.41 SendRemoteStartTransaction



Mit dieser Methode sendet ein OCPP-Server einen Remote Start Transaction-Request an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendRemoteStartTransaction : BOOL
VAR_INPUT
    nConnectorId : UDINT := 0;
    sIdToken     : T_OCPP1_IdToken;
END_VAR
VAR_OUTPUT
    eStatus      : E_OCPP1_RemoteStartStopStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendRemoteStartTransaction	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
nConnectorId	UDINT	ID des Connectors eines Charge Points.
sIdToken	T_OCPP1_IdToken [▶ 212]	ID-Token des Charge Points im Central System.

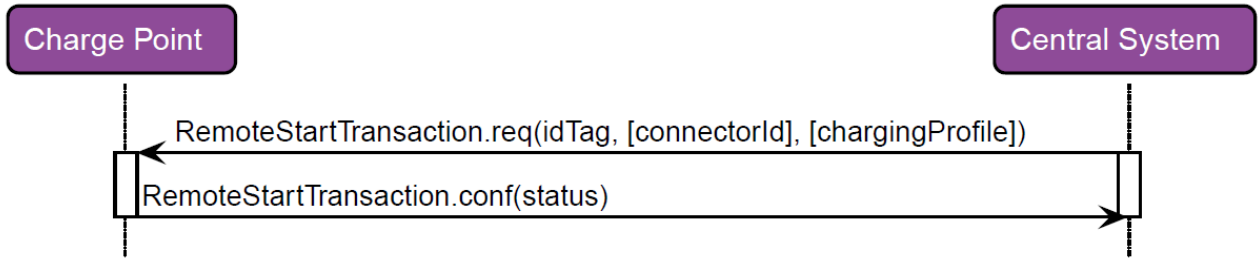
Ausgänge

Name	Typ	Beschreibung
eStatus	E_OCPP1_RemoteStartStopStatus [▶ 197]	Der Status zeigt an, ob der Charge Point den Request zum Starten einer Transaktion akzeptiert hat.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.42 SendRemoteStartTransactionProfile

Mit dieser Methode sendet ein OCPP-Server einen Remote Start Transaction-Request inklusive Charging Profile an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendRemoteStartTransactionProfile : BOOL
VAR_INPUT
    nConnectorId     : UDINT := 0;
    sIdToken         : T_OCPP1_IdToken;
    mChargingProfile : REFERENCE TO ST_OCPP1_ChargingProfile REF= 0;
END_VAR
VAR_OUTPUT
    eStatus          : E_OCPP1_RemoteStartStopStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendRemoteStartTransactionProfile	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

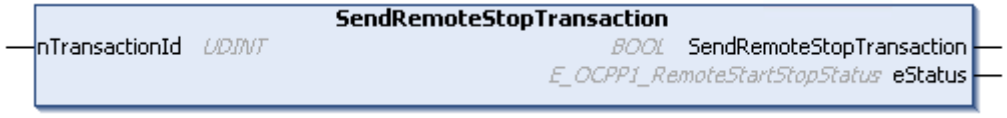
Name	Typ	Beschreibung
nConnectorId	UDINT	ID des Connectors eines Charge Points.
sIdToken	T_OCPP1_IdToken [▶ 212]	ID-Token des Charge Points im Central System.
mChargingProfile	REFERENCE TO ST_OCPP1_ChargingProfileMax [▶ 206]	An den Client zu sendendes Charging Profile.

Ausgänge

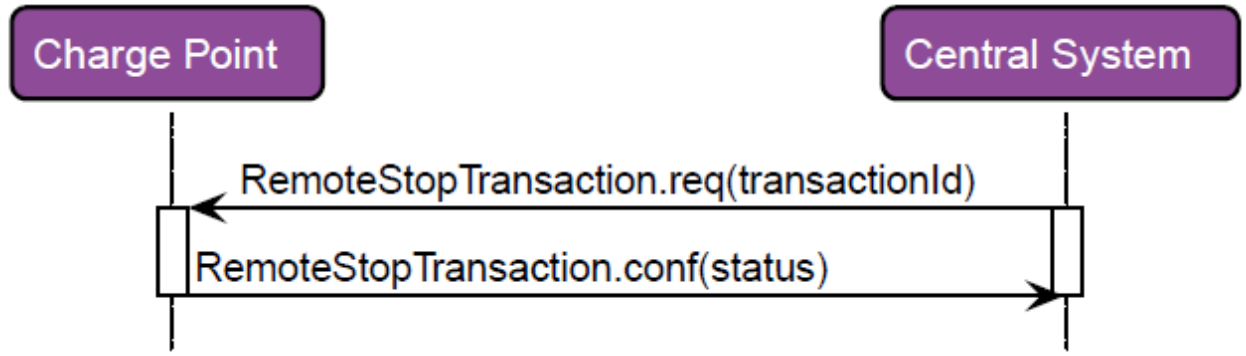
Name	Typ	Beschreibung
eStatus	E_OCPP1_RemoteStartStopStatus [▶ 197]	Der Status zeigt an, ob der Charge Point den Request zum Starten einer Transaktion akzeptiert hat.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.43 SendRemoteStopTransaction



Mit dieser Methode sendet ein OCPP-Server einen Remote Stop Transaction-Request an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendRemoteStopTransaction : BOOL
VAR_INPUT
    nTransactionId : UDINT;
END_VAR
VAR_OUTPUT
    eStatus        : E_OCPP1_RemoteStartStopStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendRemoteStopTransaction	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
nTransactionId	UDINT	Identifier der Transaktion, die gestoppt werden soll.

Ausgänge

Name	Typ	Beschreibung
eStatus	E_OCPP1_RemoteStartStopStatus [▶ 197]	Der Status zeigt an, ob der Charge Point den Request zum Stoppen einer Transaktion akzeptiert hat.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.44 SendReserveNow

Mit dieser Methode sendet ein OCPP-Server einen Reserve Now-Request an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.

Syntax

```
METHOD SendReserveNow : BOOL
VAR_INPUT
    hStationId     : UDINT;
    nConnectorId   : UDINT;
    nExpiryDate    : ULINT;
    sIdTag         : T_OCPP1_IdToken;
    sParentIdTag   : T_OCPP1_IdToken := '';
    nReservationId : UDINT;
END_VAR
VAR_OUTPUT
    eStatus        : E_OCPP1_ReservationStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendReserveNow	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
nConnectorId	UDINT	ID des Connectors eines Charge Points.
nExpiryDate	ULINT	Datum und Zeit des Endes der Reservierung.
sIdTag	T_OCPP1_IdToken [▶ 212]	Identifier, für den der Charge Point einen Connector reservieren soll.
sParentIdTag	T_OCPP1_IdToken [▶ 212]	Der Parent des Identifiers, für den der Charge Point einen Connector reservieren soll.
nReservationId	UDINT	Einzigartige ID der Reservierung.

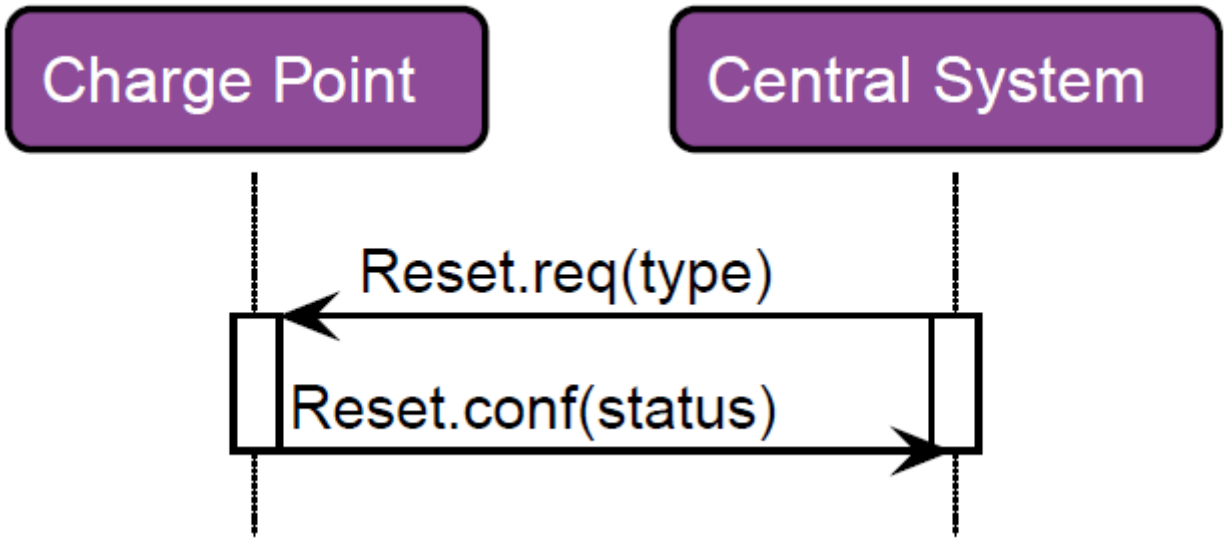
Ausgänge

Name	Typ	Beschreibung
eStatus	E_OCPP1_ReservationStatus [▶ 197]	Der Status zeigt an, ob die Reservierung erfolgreich war.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.45 SendReset

Mit dieser Methode sendet ein OCPP-Server einen Reset-Request an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendReset : BOOL
VAR_INPUT
    eType       : E_OCPP1_ResetType;
END_VAR
VAR_OUTPUT
    eStatus     : E_OCPP1_ResetStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendReset	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

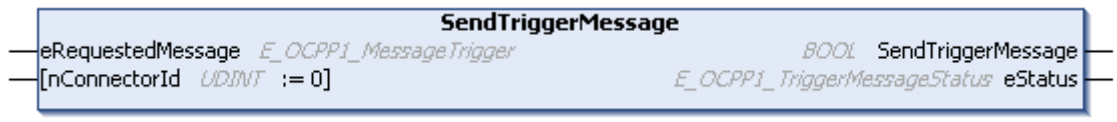
Name	Typ	Beschreibung
eType	E_OCPP1_ResetType [▶ 198]	Typ des Resets, die der Charge Point durchführen soll.

Ausgänge

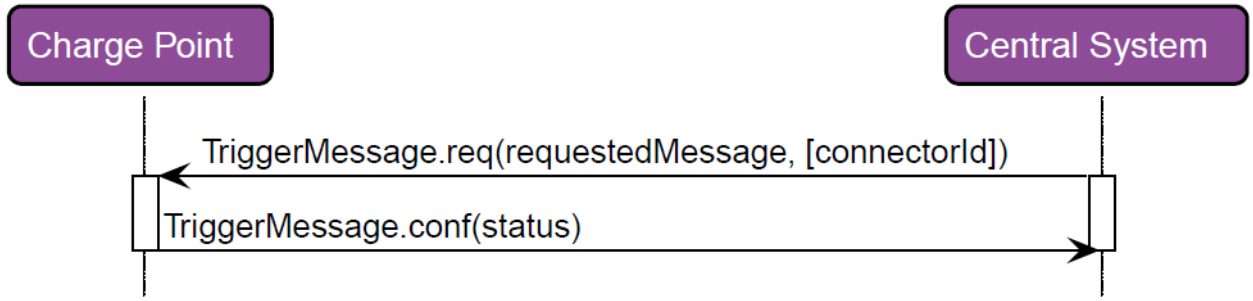
Name	Typ	Beschreibung
eStatus	E_OCPP1_ResetStatus [▶ 198]	Der Status zeigt an, ob der Charge Point den Request zum Reset akzeptieren konnte.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.46 SendTriggerMessage



Mit dieser Methode sendet ein OCPP-Server einen Trigger Message-Request an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendTriggerMessage : BOOL
VAR_INPUT
    eRequestedMessage : E_OCPP1_MessageTrigger;
    nConnectorId      : UDINT := 0;
END_VAR
VAR_OUTPUT
    eStatus           : E_OCPP1_TriggerMessageStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendTriggerMessage	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
eRequestedMessage	E_OCPP1_MessageTrigger [▶ 196]	Typ der Nachricht, die getriggert werden soll.
nConnectorId	UDINT	ID des Connectors eines Charge Points.

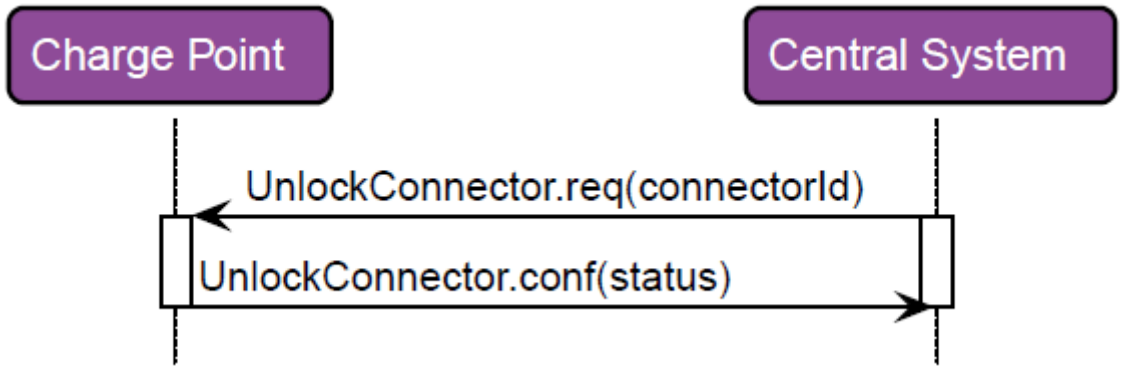
Ausgänge

Name	Typ	Beschreibung
eStatus	E_OCPP1_TriggerMessageStatus [▶ 198]	Der Status zeigt an, ob der Charge Point die angefragte Nachricht senden wird oder nicht.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.47 SendUnlockConnector

Mit dieser Methode sendet ein OCPP-Server einen Unlock Connector-Request an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendUnlockConnector : BOOL
VAR_INPUT
    nConnectorId : UDINT;
END_VAR
VAR_OUTPUT
    eStatus      : E_OCPP1_UnlockStatus;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendUnlockConnector	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

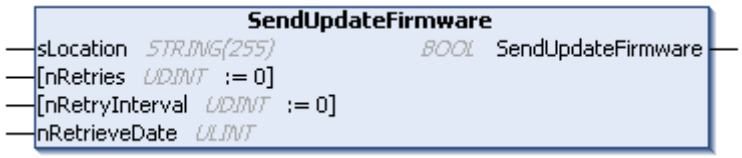
Name	Typ	Beschreibung
nConnectorId	UDINT	ID des Connectors eines Charge Points.

Ausgänge

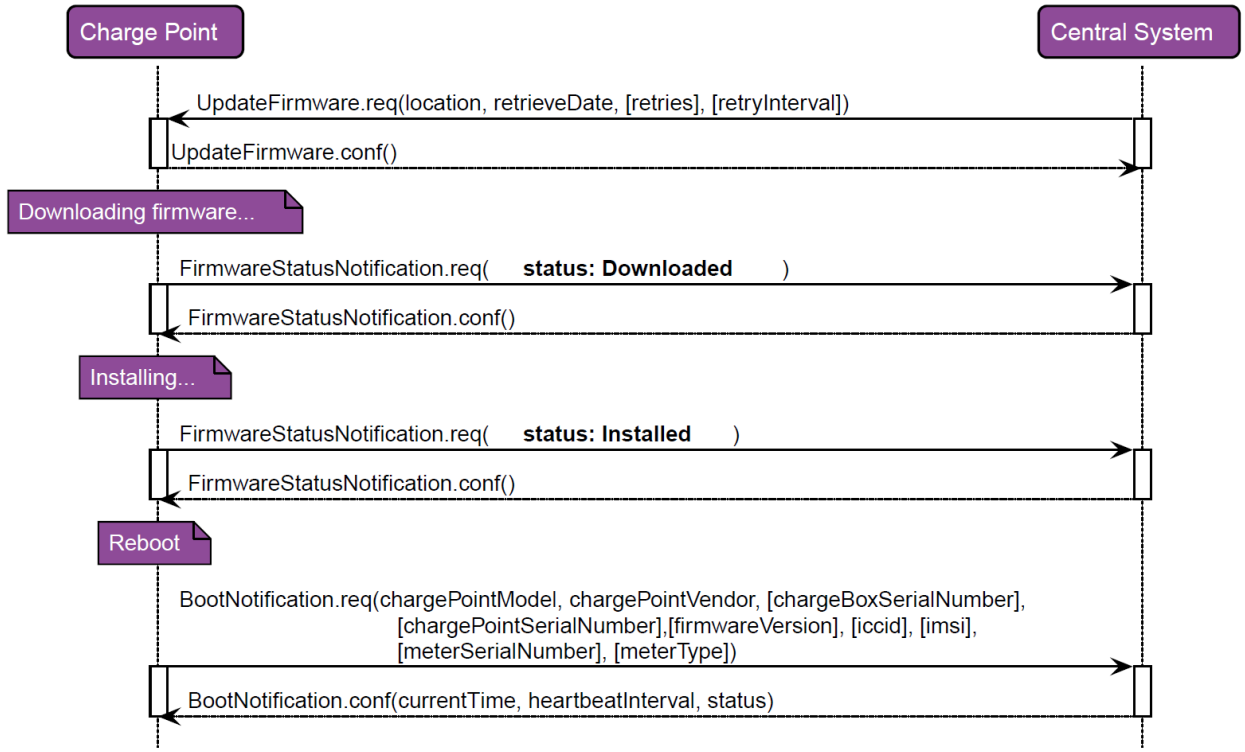
Name	Typ	Beschreibung
eStatus	E_OCPP1_UnlockStatus [▶ 198]	Der Status zeigt an, ob der Connector freigeschaltet wurde.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.3.48 SendUpdateFirmware



Mit dieser Methode sendet ein OCPP-Server einen Update Firmware-Request an den verbundenen OCPP-Client. Die Antwort des OCPP-Clients wird direkt innerhalb der Methode verarbeitet.



Syntax

```
METHOD SendUpdateFirmware : BOOL
VAR_INPUT
    sLocation      : STRING(255);
    nRetries       : UDINT := 0;
    nRetryInterval : UDINT := 0;
    nRetrieveDate  : ULINT;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SendUpdateFirmware	BOOL	Bei erfolgreichem Aufruf liefert die Methode den Rückgabewert TRUE. Auch im Fehlerfall gilt ein Methodenaufruf als erfolgreich abgeschlossen.

Eingänge

Name	Typ	Beschreibung
sLocation	STRING(255)	URI, von der die Firmware abgerufen werden soll.
nRetries	UDINT	Anzahl an Versuchen, die Firmware bei fehlgeschlagenem Download erneut herunterzuladen.
nRetryInterval	UDINT	Zeit, nach der der erneute Download versucht wird.
nRetrieveDate	ULINT	Zeit und Datum, ab dem der Charge Point die neue Firmware beziehen darf.

Mögliche Fehler werden an den Ausgängen bError und hrErrorCode der Bausteininstanz ausgegeben.

5.1.4 FB_OCPP1_ChargingProfile

Mit diesem Funktionsbaustein können die aktuell vorhandenen Charging Profiles abgefragt werden.

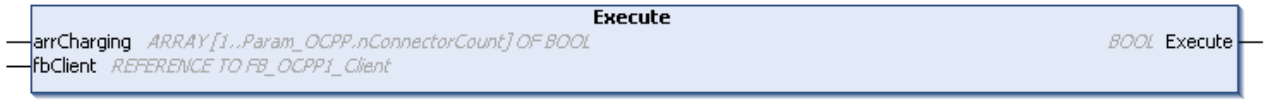
Syntax

```
FUNCTION BLOCK FB_OCPP1_ChargingProfile
VAR_OUTPUT
    nScheduleTime       : ULINT;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
nScheduleTime	ULINT	Der aktuelle Zeitstempel, anhand dessen die Charging Profiles geplant werden.

5.1.4.1 Execute



Diese Methode muss zyklisch aufgerufen werden, wenn der Funktionsbaustein verwendet werden soll. Es muss die Instanz des FB_OCPP1_Client [▶ 22] übergeben werden. Zusätzlich wird die Information benötigt, welcher Connector gerade am Laden ist. Dadurch kann der Startpunkt der Charging Schedule berechnet werden.

Syntax

```
METHOD PUBLIC Execute : BOOL
VAR_INPUT
    arrCharging : ARRAY[1..Param_OCPP.nConnectorCount] OF BOOL;
    fbClient    : REFERENCE TO FB_OCPP1_Client;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
Execute	BOOL	Der Rückgabewert der Methode ist aktuell nicht belegt und gibt immer den Wert FALSE zurück.

Eingänge

Name	Typ	Beschreibung
arrCharging	ARRAY [1..Param_OCPP [▶ 212].nConnectorCount] OF BOOL	Pro Connector innerhalb eines Charge Points die Information, ob gerade geladen wird oder nicht.
fbClient	REFERENCE TO FB_OCPP1_Client [▶ 22]	Instanz des OCPP-Clients.

5.1.4.2 GetCurrentPeriod

Mit dieser Methode lässt sich für einen bestimmten Connector eines Charge Points die aktuell vorliegende Charging Schedule Period auslesen.

Syntax

```
METHOD PUBLIC GetCurrentPeriod : BOOL
VAR_INPUT
    nConnectorId : UDINT;
END_VAR
VAR_IN_OUT
    mPeriod      : ST_OCPP1_ChargingSchedulePeriod;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
GetCurrentPeriod	BOOL	Der Rückgabewert der Methode ist aktuell nicht belegt und gibt immer den Wert FALSE zurück.

Eingänge

Name	Typ	Beschreibung
nConnectorId	UDINT	ID des Connectors eines Charge Points.

Ein-/Ausgänge

Name	Typ	Beschreibung
mPeriod	ST_OCPP1_ChargingSchedulePeriod [▶ 209]	Die aktuelle Charging Schedule Period am Charge Point.

5.1.4.3 GetCurrentSchedule

Mit dieser Methode lässt sich für einen bestimmten Connector eines Charge Points die aktuell vorliegende Charging Schedule auslesen.

Syntax

```
METHOD PUBLIC GetCurrentSchedule : BOOL
VAR_INPUT
    nConnectorId      : UDINT;
END_VAR
VAR_IN_OUT
    mChargingSchedule : ST_OCPP1_ChargingScheduleMax;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
GetCurrentSchedule	BOOL	Der Rückgabewert der Methode ist aktuell nicht belegt und gibt immer den Wert FALSE zurück.

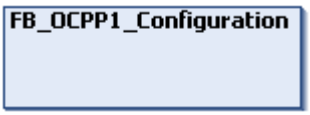
Eingänge

Name	Typ	Beschreibung
nConnectorId	UDINT	ID des Connectors eines Charge Points.

Ein-/Ausgänge

Name	Typ	Beschreibung
mChargingSchedule	ST_OCPP1_ChargingScheduleMax [▶ 208]	Die aktuelle Charging Schedule am Charge Point.

5.1.5 FB_OCPP1_Configuration



Mit diesem Funktionsbaustein können die Configuration Keys einer Instanz des FB_OCPP1_Client [▶ 22] verwaltet werden.

Dazu werden verschiedene Methoden zu folgenden Zwecken bereitgestellt:

- Abfragen der letzten Änderungen an Configuration Keys
- Initialisieren neuer Configuration Keys
- Initialisieren von Standard-Werten (Liste siehe Configuration Keys [▶ 214])
- Setzen neuer Werte für Configuration Keys
- Empfangen von Change Configuration- und Get Configuration-Requests eines OCPP-Servers mit automatischer Beantwortung innerhalb des OCPP-Clients

Syntax

```
FUNCTION BLOCK FB_OCPP1_Configuration
```

5.1.5.1 GetChanged

Mit dieser Methode lässt sich die letzte Änderung an den Config Keys abfragen, die Teil der Aufzählung E_OCPP1_ConfigKey [▶ 193] sind.

Syntax

```
METHOD GetChanged : BOOL
VAR_OUTPUT
    eKey   : E_OCPP1_ConfigKey;
    sValue : T_OCPP1_ConfigValue;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
GetChanged	BOOL	Bei erfolgreichem Methodenaufruf liefert die Methode den Rückgabewert TRUE.

Ausgänge

Name	Typ	Beschreibung
eKey	E_OCPP1_ConfigKey [▶ 193]	Config Key, für den die letzte Änderung stattgefunden hat.
sValue	T_OCPP1_ConfigValue [▶ 211]	Der zugehörige Wert.

5.1.5.2 GetChangedEx

Mit dieser Methode lässt sich die letzte Änderung an den Config Keys abfragen, die nicht Teil der Aufzählung E_OCPP1_ConfigKey [▶ 193] sind.

Syntax

```
METHOD GetChangedEx : BOOL
VAR_OUTPUT
    sKey   : T_OCPP1_ConfigKey;
    sValue : T_OCPP1_ConfigValue;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
GetChanged	BOOL	Bei erfolgreichem Methodenaufruf liefert die Methode den Rückgabewert TRUE.

Ausgänge

Name	Typ	Beschreibung
sKey	T_OCPP1_ConfigKey [▶ 211]	Config Key, für den die letzte Änderung stattgefunden hat.
sValue	T_OCPP1_ConfigValue [▶ 211]	Der zugehörige Wert.

5.1.5.3 GetValue

Mit dieser Methode lässt sich der aktuelle Wert eines Config Keys abfragen, der Teil der Aufzählung E_OCPP1_ConfigKey [▶ 193] ist.

Syntax

```
METHOD GetValue : T_OCPP1_ConfigValue
VAR_INPUT
    eKey : E_OCPP1_ConfigKey;
END_VAR
```

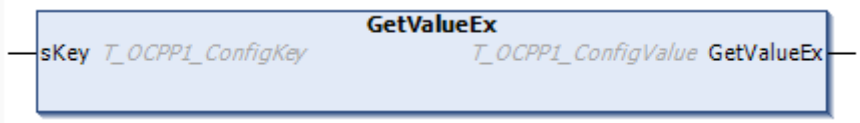
Rückgabewert

Name	Typ	Beschreibung
GetValue	T_OCPP1_ConfigValue [▶ 211]	Aktueller Wert des abgefragten Config Keys.

Eingänge

Name	Typ	Beschreibung
eKey	E_OCPP1_ConfigKey [▶ 193]	Config Key, für den der aktuelle Wert abgefragt werden soll.

5.1.5.4 GetValueEx



Mit dieser Methode lässt sich der aktuelle Wert eines Config Keys abfragen.

Syntax

```
METHOD GetValueEx : T_OCPP1_ConfigValue
VAR_INPUT
    sKey : T_OCPP1_ConfigKey;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
GetValue	T_OCPP1_ConfigValue [▶ 211]	Aktueller Wert des abgefragten Config Keys.

Eingänge

Name	Typ	Beschreibung
sKey	T_OCPP1_ConfigKey [▶ 211]	Config Key, für den der aktuelle Wert abgefragt werden soll.

5.1.5.5 InitKeyValue

Mit dieser Methode können neue Key/Value-Paare für Configuration Keys hinzugefügt werden, die Teil der Aufzählung E_OCPP1_ConfigKey [▶ 193] sind. Wenn ein Key übergeben wird, der im OCPP-Client schon bekannt ist, wird der Wert überschrieben. Für das Hinzufügen benutzerdefinierter Keys kann die Methode InitKeyValueEx [▶ 186] verwendet werden.

Syntax

```
METHOD InitKeyValue : UDINT
VAR_INPUT
    eKey      : E_OCPP1_ConfigKey;
    sValue    : T_OCPP1_ConfigValue;
    bReadonly : BOOL;
END_VAR
```

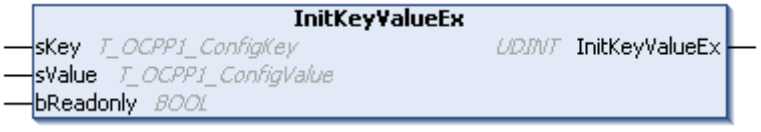
Rückgabewert

Name	Typ	Beschreibung
InitKeyValue	UDINT	Der zugewiesene Index des Config Keys. Zum aktuellen Zeitpunkt wird dieser Index allerdings nur intern weiterverwendet und muss bei der Programmierung nicht beachtet werden.

Eingänge

Name	Typ	Beschreibung
eKey	E_OCPP1_ConfigKey [▶ 193]	Configuration Key, der überschrieben oder hinzugefügt werden soll.
sValue	T_OCPP1_ConfigValue [▶ 211]	Wert des zugehörigen Configuration Keys.
bReadonly	BOOL	Bestimmt das Zugriffsrecht des Central Systems, entweder darf der Wert nur gelesen (TRUE) oder auch geschrieben werden (FALSE).

5.1.5.6 InitKeyValueEx



Mit dieser Methode können neue Key/Value-Paare für Configuration Keys hinzugefügt werden. Wenn ein Key übergeben wird, der im OCPP-Client schon bekannt ist, wird der Wert überschrieben. Die Methode kann dazu verwendet werden, benutzerdefinierte Keys hinzuzufügen, die nicht Teil der Aufzählung E_OCPP1_ConfigKey [▶ 193] sind.

Syntax

```
METHOD InitKeyValueEx : UDINT
VAR_INPUT
    sKey      : T_OCPP1_ConfigKey;
    sValue    : T_OCPP1_ConfigValue;
    bReadonly : BOOL;
END_VAR
```

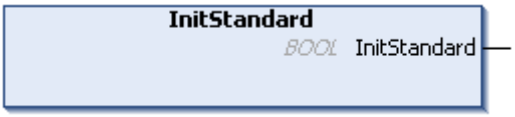
Rückgabewert

Name	Typ	Beschreibung
InitKeyValueEx	UDINT	Der zugewiesene Index des Config Keys. Zum aktuellen Zeitpunkt wird dieser Index allerdings nur intern weiterverwendet und muss bei der Programmierung nicht beachtet werden.

Eingänge

Name	Typ	Beschreibung
sKey	T_OCPP1_ConfigKey [▶ 211]	Configuration Key, der überschrieben oder hinzugefügt werden soll.
sValue	T_OCPP1_ConfigValue [▶ 211]	Wert des zugehörigen Configuration Keys.
bReadonly	BOOL	Bestimmt das Zugriffsrecht des Central Systems, entweder darf der Wert nur gelesen (TRUE) oder auch geschrieben werden (FALSE).

5.1.5.7 InitStandard



Mit dieser Methode werden Standard-Werte für einige in der OCPP-Spezifikation beschriebenen Configuration Keys gesetzt. Die Liste mit den Configuration Keys und ihrer Standardbelegung befindet sich im Anhang [▶ 214].

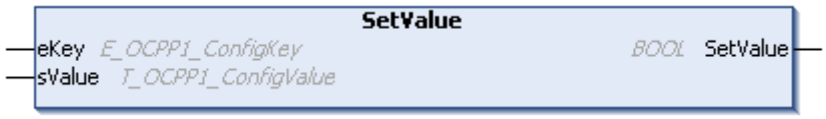
Syntax

```
METHOD InitStandard : BOOL
```

Rückgabewert

Name	Typ	Beschreibung
InitStandard	BOOL	Der Rückgabewert der Methode ist aktuell nicht belegt und gibt immer den Wert FALSE zurück.

5.1.5.8 SetValue



Mit dieser Methode kann für einen Config Key aus der Aufzählung E_OCPP1_ConfigKey [▶ 193] ein neuer Wert geschrieben werden.

Syntax

```
METHOD SetValue : BOOL
VAR_INPUT
    eKey : E_OCPP1_ConfigKey;
END_VAR
VAR_IN_OUT CONSTANT
    sValue : T_OCPP1_ConfigValue;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SetValue	BOOL	Bei erfolgreichem Methodenaufruf liefert die Methode den Rückgabewert TRUE.

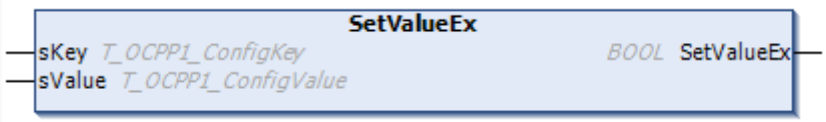
Eingänge

Name	Typ	Beschreibung
eKey	E_OCPP1_ConfigKey [▶ 193]	Config Key, der neu beschrieben werden soll.

Ein-/Ausgänge

Name	Typ	Beschreibung
sValue	T_OCPP1_ConfigValue [▶ 211]	Neuer Wert für den zu beschreibenden Config Key.

5.1.5.9 SetValueEx



Mit dieser Methode kann für einen Config Key ein neuer Wert geschrieben werden.

Syntax

```
METHOD SetValueEx : BOOL
VAR_INPUT
    sKey : T_OCPP1_ConfigKey;
END_VAR
VAR_IN_OUT CONSTANT
    sValue : T_OCPP1_ConfigValue;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
SetValueEx	BOOL	Bei erfolgreichem Methodenaufruf liefert die Methode den Rückgabewert TRUE.

Eingänge

Name	Typ	Beschreibung
sKey	T_OCPP1_ConfigKey [▶ 211]	Config Key, der neu beschrieben werden soll.

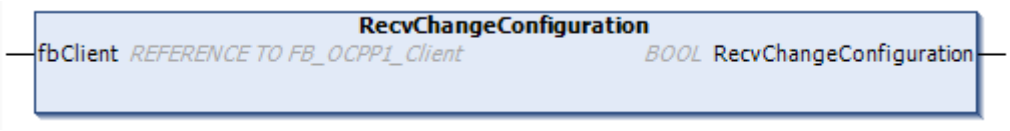
Ein-/Ausgänge

Name	Typ	Beschreibung
sValue	T_OCPP1_ConfigValue [▶ 211]	Neuer Wert für den zu beschreibenden Config Key.

Sehen Sie dazu auch

E_OCPP1_ConfigKey [▶ 193]

5.1.5.10 RecvChangeConfiguration



Diese Methode ist eine alternative Möglichkeit, im OCPP-Client einen Change Configuration-Request vom entsprechenden OCPP-Server zu empfangen und zu beantworten. Im Unterschied zu der RecvChangeConfiguration [▶ 30]-Methode am FB_OCPP1_Client [▶ 22] wird hier die Antwort an den OCPP-Server automatisch gesendet.

Wenn die zu schreibenden Configuration Keys readonly sind, werden sie nicht geschrieben. Alle beschreibbaren Configuration Keys werden von der Methode gespeichert.

Syntax

```
METHOD RecvChangeConfiguration : BOOL
VAR_INPUT
    fbClient : REFERENCE TO FB_OCPP1_Client;
END_VAR
```

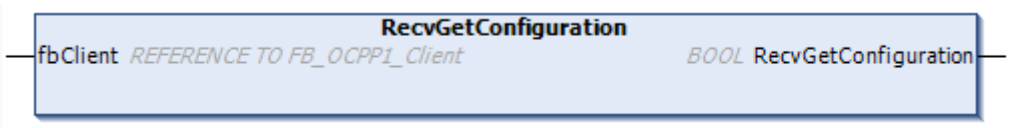
Rückgabewert

Name	Typ	Beschreibung
RecvChangeConfiguration	BOOL	Aktuell nicht implementiert. In zukünftigen Versionen: Bei erfolgreichem Methodenaufruf liefert die Methode den Rückgabewert TRUE.

Ein-/Ausgänge

Name	Typ	Beschreibung
fbClient	REFERENCE TO FB_OCPP1_Client [▶ 22]	Instanz des verwendeten Client-Bausteins.

5.1.5.11 RecvGetConfiguration



Diese Methode ist eine alternative Möglichkeit, im OCPP-Client einen Get Configuration-Request vom entsprechenden OCPP-Server zu empfangen und zu beantworten. Im Unterschied zu der RecvGetConfiguration [▶ 35]-Methode am FB_OCPP1_Client [▶ 22] wird hier die Antwort an den OCPP-Server automatisch gesendet. Sie enthält je nach Anfrage alle oder spezifisch angefragte Configuration Keys.

Syntax

```
METHOD RecvGetConfiguration : BOOL
VAR_INPUT
    fbClient : REFERENCE TO FB_OCPP1_Client;
END_VAR
```

Rückgabewert

Name	Typ	Beschreibung
RecvGetConfiguration	BOOL	Aktuell nicht implementiert. In zukünftigen Versionen: Bei erfolgreichem Methodenaufruf liefert die Methode den Rückgabewert TRUE.

Ein-/Ausgänge

Name	Typ	Beschreibung
fbClient	REFERENCE TO FB_OCPP1_Client [▶ 22]	Instanz des verwendeten Client-Bausteins.

5.2 DUTs

5.2.1 Enumerations

5.2.1.1 E_OCPP1_Action

```
TYPE E_OCPP1_Action :
(
    None                            := 0,
    // Core
    BootNotification                := 10001,
    Heartbeat                       := 10002,
    MeterValues                     := 10003,
    StatusNotification              := 10004,
    Authorize                       := 10005,
    StartTransaction                := 10006,
    StopTransaction                 := 10007,
    RemoteStartTransaction          := 10011,
    RemoteStopTransaction           := 10012,
```

```
    ChangeAvailability               := 10013,
    ChangeConfiguration              := 10014,
    GetConfiguration                 := 10015,
    UnlockConnector                  := 10016,
    ClearCache                       := 10017,
    Reset                            := 10018,
    // Remote Trigger
    TriggerMessage                   := 10021,
    // Reservation
    ReserveNow                       := 10031,
    CancelReservation                := 10032,
    // Smart Charging
    SetChargingProfile               := 10041,
    GetCompositeSchedule             := 10042,
    ClearChargingProfile             := 10043,
    // Authentication
    GetLocalListVersion              := 10051,
    SendLocalList                    := 10052,
    // Firmware
    DiagnosticsStatusNotification    := 10061,
    FirmwareStatusNotification       := 10062,
    GetDiagnostics                   := 10063,
    UpdateFirmware                   := 10064,
    // Vendor
    DataTransfer                     := 10091,
    // Security
    SecurityEventNotification        := 11001,
    ExtendedTriggerMessage           := 11002,
    LogStatusNotification            := 11003,
    GetLog                           := 11004,
    GetInstalledCertificateIds       := 11005,
    CertificateSigned                := 11006,
    SignCertificate                  := 11007,
    InstallCertificate               := 11008,
    DeleteCertificate                := 11009,
    SignedFirmwareStatusNotification := 11010,
    SignedUpdateFirmware             := 11011
) UDINT;
END_TYPE
```

5.2.1.2 E_OCPP1_AuthenticationMode

```
TYPE E_OCPP1_AuthenticationMode :
(
    None  := 0, // No Authentication
    Ident := 1, // Ident Authentication
    Basic := 2  // Basic Authentication
) UDINT;
END_TYPE
```

5.2.1.3 E_OCPP1_AuthorizationStatus

```
TYPE E_OCPP1_AuthorizationStatus :
(
    None := 0,       // Any status not covered by this implementation
    Accepted,        // Identifier is allowed for charging.
    Blocked,         // Identifier has been blocked. Not allowed for charging.
    Expired,         // Identifier has expired. Not allowed for charging.
    Invalid,         // Identifier is unknown. Not allowed for charging.
    ConcurrentTx     // Identifier is already involved in another transaction and multiple transacti
ons are not allowed.
) UDINT;
END_TYPE
```

5.2.1.4 E_OCPP1_AvailabilityStatus

```
TYPE E_OCPP1_AvailabilityStatus :
(
    None     := 0, // Any status not covered by this implementation
    Accepted,      // Request has been accepted and will be executed.
    Rejected,      // Request has not been accepted and will not be executed.
    Scheduled      // Request has been accepted and will be executed when transaction(s) in progress
 have finished.
) UDINT;
END_TYPE
```

5.2.1.5 E_OCPP1_AvailabilityType

```
TYPE E_OCPP1_AvailabilityType :
(
    None := 0,        // Any status not covered by this implementation
    Inoperative,      // Charge point is not available for charging.
    Operative         // Charge point is available for charging.
) UDINT;
END_TYPE
```

5.2.1.6 E_OCPP1_CancelReservationStatus

```
TYPE E_OCPP1_CancelReservationStatus :
(
    None := 0,     // Any status not covered by this implementation
    Accepted,      // Reservation for the identifier has been cancelled.
    Rejected       // Reservation could not be cancelled, because there is no reservation active for
 the identifier.
) UDINT;
END_TYPE
```

5.2.1.7 E_OCPP1_ChargePointError

```
TYPE E_OCPP1_ChargePointError :
(
    None := 0,            // Any status not covered by this implementation
    NoError,              // No error to report.
    ConnectorLockFailure, // Failure to lock or unlock connector.
    EVCommunicationError, // Communication failure with the vehicle, might be Mode 3 or other commun
ication protocol problem.
    GroundFailure,        // Ground fault circuit interrupter has been activated.
    HighTemperature,      // Temperature inside Charge Point is too high.
    InternalError,        // Error in internal hard - or software component.
    LocalListConflict,    // The authorization information received from the Central System is in co
nflict with the LocalAuthorizationList.
    OverCurrentFailure,   // Over current protection device has tripped.
    OverVoltage,          // Voltage has risen above an acceptable level.
    PowerMeterFailure,    // Failure to read electrical / energy / power meter.
    PowerSwitchFailure,   // Failure to control power switch.
    ReaderFailure,        // Failure with idTag reader.
    ResetFailure,         // Unable to perform a reset.
    UnderVoltage,         // Voltage has dropped below an acceptable level.
    WeakSignal,           // Wireless communication device reports a weak signal.
    OtherError            // Other type of error.More information in vendorErrorCode.
) UDINT;
END_TYPE
```

5.2.1.8 E_OCPP1_ChargePointStatus

```
TYPE E_OCPP1_ChargePointStatus :
(
    None := 0,     // Any status not covered by this implementation
    Available,     // When a Connector becomes available for a new user. (Operative)
    Preparing,     // When a Connector becomes no longer available for a new user but there is no on
going Transaction. (Operative)
    Charging,      // When the contactor of a Connector closes, allowing the vehicle to charge. (Ope
rative)
    SuspendedEVSE, // When the EV is connected to the EVSE but the EVSE is not offering energy to th
e EV. (Operative)
    SuspendedEV,   // When the EV is connected to the EVSE and the EVSE is offering energy but the E
V is not taking any energy. (Operative)
    Finishing,     // When a Transaction has stopped at a Connector, but the Connector is not yet av
ailable for a new user. (Operative)
    Reserved,      // When a Connector becomes reserved as a result of a Reserve Now command. (Opera
tive)
    Unavailable,   // When a Connector becomes unavailable as the result of a Change Availability co
mmand or an event upon which the Charge Point transitions to unavailable at its discretion. (Inopera
tive)
    Faulted        // When a Charge Point or connector has reported an error and is not available fo
r energy delivery . (Inoperative)
) UDINT;
END_TYPE
```

5.2.1.9 E_OCPP1_ChargingProfileKindType

```
TYPE E_OCPP1_ChargingProfileKindType :
(
    None := 0, // Any status not covered by this implementation
    Absolute,  // Schedule periods are relative to a fixed point in time defined in the schedule.
    Recurring, // The schedule restarts periodically at the first schedule period.
    Relative   // Schedule periods are relative to a situation-
specific start point that is determined by the charge point.
) UDINT;
END_TYPE
```

5.2.1.10 E_OCPP1_ChargingProfilePurposeType

```
TYPE E_OCPP1_ChargingProfilePurposeType :
(
    None := 0,             // Any status not covered by this implementation
    ChargePointMaxProfile, // Configuration for the maximum power or current available for an entire
 Charge Point.
    TxDefaultProfile,      // Default profile that can be configured in the Charge Point.
    TxProfile              // Profile with constraints to be imposed by the Charge Point on the curr
ent transaction.
) UDINT;
END_TYPE
```

5.2.1.11 E_OCPP1_ChargingProfileStatus

```
TYPE E_OCPP1_ChargingProfileStatus :
(
    None := 0,   // Any status not covered by this implementation
    Accepted,    // Request has been acceptedand will be executed.
    Rejected,    // Request has not been accepted and will not be executed.
    NotSupported // Charge Point indicates that the request is not supported.
) UDINT;
END_TYPE
```

5.2.1.12 E_OCPP1_ChargingRateUnitType

```
TYPE E_OCPP1_ChargingRateUnitType :
(
    None := 0, // Any status not covered by this implementation
    W,         // Watts (power).
    A          // Amperes (current).
) UDINT;
END_TYPE
```

5.2.1.13 E_OCPP1_ClearCacheStatus

```
TYPE E_OCPP1_ClearCacheStatus :
(
    None := 0, // Any status not covered by this implementation
    Accepted,  // Command has been executed.
    Rejected   // Command has not been executed.
) UDINT;
END_TYPE
```

5.2.1.14 E_OCPP1_ClearChargingProfileStatus

```
TYPE E_OCPP1_ClearChargingProfileStatus :
(
    None := 0, // Any status not covered by this implementation
    Accepted,  // Request has been accepted and will be executed.
    Unknown    // No Charging Profile(s) were found matching the request.
) UDINT;
END_TYPE
```

5.2.1.15 E_OCPP1_ConfigKey

```
TYPE E_OCPP1_ConfigKey :
(
    None := 0,
    AllowOfflineTxForUnknownId, // optional
    AuthorizationCacheEnabled, // optional
    AuthorizeRemoteTxRequests,
```

```
    BlinkRepeat,     // optional
    ClockAlignedDataInterval,
    ConnectionTimeOut,
    ConnectorPhaseRotation,
    ConnectorPhaseRotationMaxLength, // optional readonly
    GetConfigurationMaxKeys,// readonly
    HeartbeatInterval,
    LightIntensity, // optional
    LocalAuthorizeOffline,
    LocalPreAuthorize,
    MeterValuesAlignedData,
    MeterValuesAlignedDataMaxLength, // optional readonly
    MeterValuesSampledData,
    MeterValuesSampledDataMaxLength, // optional readonly
    MeterValueSampleInterval,
    MinimumStatusDuration, // optional
    NumberOfConnectors, // readonly
    ResetRetries,
    StopTransactionOnEVSideDisconnect,
    StopTransactionOnInvalidId,
    StopTxnAlignedData,
    StopTxnAlignedDataMaxLength, // optional readonly
    StopTxnSampledData,
    StopTxnSampledDataMaxLength, // optional readonly
    SupportedFeatureProfiles, // readonly
    SupportedFeatureProfilesMaxLength, // optional readonly
    TransactionMessageAttempts,
    TransactionMessageRetryInterval,
    UnlockConnectorOnEVSideDisconnect,
    WebSocketPingInterval, // optional
    LocalAuthListEnabled,
    LocalAuthListMaxLength, // readonly
    SendLocalListMaxLength, // readonly
    ReserveConnectorZeroSupported, // optional readonly
    ChargeProfileMaxStackLevel, // Readonly
    ChargingScheduleAllowedChargingRateUnit, // readonly
    ChargingScheduleMaxPeriods, // readonly
    ConnectorSwitch3to1PhaseSupported, // optional readonly
    MaxChargingProfilesInstalled, // readonly
    MaxKeys // amount of keys +1, not a Configkey!
) UDINT;
END_TYPE
```

5.2.1.16 E_OCPP1_ConfigurationStatus

```
TYPE E_OCPP1_ConfigurationStatus :
(
    None := 0,      // Any status not covered by this implementation
    Accepted,       // Configuration key is supported and setting has been changed.
    Rejected,       // Configuration key is supported, but setting could not be changed.
    RebootRequired, // Configuration key is supported and setting has been changed, but change will
be available after reboot.
    NotSupported    // Configuration key is not supported.
) UDINT;
END_TYPE
```

5.2.1.17 E_OCPP1_DataTransferStatus

```
TYPE E_OCPP1_DataTransferStatus :
(
    None := 0,        // Any status not covered by this implementation
    Accepted,         // Message has been accepted and the contained request is accepted.
    Rejected,         // Message has been accepted but the contained request is rejected.
    UnknownMessageId, // Message could not be interpreted due to unknown messageId string.
    UnknownVendorId   // Message could not be interpreted due to unknown vendorId string.
) UDINT;
END_TYPE
```

5.2.1.18 E_OCPP1_DiagnosticsStatus

```
TYPE E_OCPP1_DiagnosticsStatus :
(
    None := 0,    // Any status not covered by this implementation
    Idle,         // Charge Point is not performing diagnostics related tasks.
    Uploaded,     // Diagnostics information has been uploaded.
    UploadFailed, // Uploading of diagnostics failed.
```

```
    Uploading     // File is being uploaded.
) UDINT;
END_TYPE
```

5.2.1.19 E_OCPP1_Error

```
TYPE E_OCPP1_Error :
(
    None := 0,                    // Any error not covered by this implementation
    GenericError,                 // Any other error not covered by the previous ones
    NotImplemented,               // Requested Action is not known by receiver
    NotSupported,                 // Requested Action is recognized but not supported by the receive
r
    InternalError,                // An internal error occurred and the receiver was not able to pro
cess the requested Action successfully
    ProtocolError,                // Payload for Action is incomplete
    SecurityError,                // During the processing of Action a security issue occurred preve
nting receiver from completing the Action successfully
    FormationViolation,           // Payload for Action is syntactically incorrect or not conform th
e PDU structure for Action
    PropertyConstraintViolation,  // Payload is syntactically correct but at least one field contain
s an invalid value
    OccurenceConstraintViolation, // Payload for Action is syntactically correct but at least one of
 the fields violates occurence constraints
    TypeConstraintViolation       // Payload for Action is syntactically correct but at least one of
 the fields violates data type constraints
) UDINT;
END_TYPE
```

5.2.1.20 E_OCPP1_FirmwareStatus

```
TYPE E_OCPP1_FirmwareStatus :
(
    None := 0,          // Any status not covered by this implementation
    Downloaded,         // New firmware has been downloaded by Charge Point.
    DownloadFailed,     // Charge point failed to download firmware.
    Downloading,        // Firmware is being downloaded.
    Idle,               // Charge Point is not performing firmware update related tasks.
    InstallationFailed, // Installation of new firmware has failed.
    Installing,         // Firmware is being installed.
    Installed           // New firmware has successfully been installed in charge point.
) UDINT;
END_TYPE
```

5.2.1.21 E_OCPP1_GetCompositeScheduleStatus

```
TYPE E_OCPP1_GetCompositeScheduleStatus :
(
    None := 0, // Any status not covered by this implementation
    Accepted,  // Request has been accepted and will be executed.
    Rejected   // Request has not been accepted and will not be executed.
) UDINT;
END_TYPE
```

5.2.1.22 E_OCPP1_Location

```
TYPE E_OCPP1_Location :
(
    None := 0, // Any status not covered by this implementation
    Body,      // Measurement inside body of Charge Point(e.g.Temperature)
    Cable,     // Measurement taken from cable between EV and Charge Point
    EV,        // Measurement taken by EV
    Inlet,     // Measurement at network("grid") inlet connection
    Outlet     // Measurement at a Connector.Default value
) UDINT;
END_TYPE
```

5.2.1.23 E_OCPP1_Measurand

```
TYPE E_OCPP1_Measurand :
(
    None := 0,                    // Any status not covered by this implementation
    CurrentExport,                // Instantaneous current flow from EV
    CurrentImport,                // Instantaneous current flow to EV
    CurrentOffered,               // Maximum current offered to EV
```

```
    EnergyActiveExportRegister,   // Numerical value read from the "active electrical energy" (Wh or
 kWh) register of the (most authoritative) electrical meter measuring energy exported (to the grid).
    EnergyActiveImportRegister,   // Numerical value read from the "active electrical energy" (Wh or
 kWh) register of the (most authoritative) electrical meter measuring energy imported (from the grid
 supply).
    EnergyReactiveExportRegister, // Numerical value read from the "reactive electrical energy" (VAR
h or kVARh) register of the (most authoritative) electrical meter measuring energy exported (to the
grid).
    EnergyReactiveImportRegister, // Numerical value read from the "reactive electrical energy" (VAR
h or kVARh) register of the (most authoritative) electrical meter measuring energy imported (from th
e grid supply).
    EnergyActiveExportInterval,   // Absolute amount of "active electrical energy" (Wh or kWh) expor
ted (to the grid) during an associated time "interval".
    EnergyActiveImportInterval,   // Absolute amount of "active electrical energy" (Wh or kWh) impor
ted(from the grid supply) during an associated time "interval".
    EnergyReactiveExportInterval, // Absolute amount of "reactive electrical energy" (VARh or kVARh)
 exported (to the grid) during an associated time "interval".
    EnergyReactiveImportInterval, // Absolute amount of "reactive electrical energy" (VARh or kVARh)
 imported (from the grid supply) during an associated time "interval".
    Frequency,                    // Instantaneous reading of powerline frequency.
    PowerActiveExport,            // Instantaneous active power exported by EV. (W or kW)
    PowerActiveImport,            // Instantaneous active power imported by EV. (W or kW)
    PowerFactor,                  // Instantaneous power factor of total energy flow
    PowerOffered,                 // Maximum power offered to EV
    PowerReactiveExport,          // Instantaneous reactive power exported by EV. (var or kvar)
    PowerReactiveImport,          // Instantaneous reactive power imported by EV. (var or kvar)
    RPM,                          // Fan speed in RPM
    SoC,                          // State of charge of charging vehicle in percentage
    Temperature,                  // Temperature reading inside Charge Point.
    Voltage                       // Instantaneous AC RMS supply voltage
) UDINT;
END_TYPE
```

5.2.1.24 E_OCPP1_MessageTrigger

```
TYPE E_OCPP1_MessageTrigger :
(
    None := 0,                     // Any status not covered by this implementation
    BootNotification,              // To trigger a BootNotification request
    DiagnosticsStatusNotification, // To trigger a DiagnosticsStatusNotification request
    FirmwareStatusNotification,    // To trigger a FirmwareStatusNotification request
    Heartbeat,                     // To trigger a Heartbeat request
    MeterValues,                   // To trigger a MeterValues request
    StatusNotification             // To trigger a StatusNotification request
) UDINT;
END_TYPE
```

5.2.1.25 E_OCPP1_Phase

```
TYPE E_OCPP1_Phase :
(
    None := 0, // Any status not covered by this implementation
    L1,        // Measured on L1
    L2,        // Measured on L2
    L3,        // Measured on L3
    N,         // Measured on Neutral
    L1N,       // Measured on L1 with respect to Neutral conductor
    L2N,       // Measured on L2 with respect to Neutral conductor
    L3N,       // Measured on L3 with respect to Neutral conductor
    L12,       // Measured between L1 and L2
    L23,       // Measured between L2 and L3
    L31        // Measured between L3 and L1
) UDINT;
END_TYPE
```

5.2.1.26 E_OCPP1_ReadingContext

```
TYPE E_OCPP1_ReadingContext :
(
    None := 0,         // Any status not covered by this implementation
    InterruptionBegin, // Value taken at start of interruption.
    InterruptionEnd,   // Value taken when resuming after interruption.
    Other,             // Value for any other situations.
    SampleClock,       // Value taken at clock aligned interval.
    SamplePeriodic,    // Value taken as periodic sample relative to start time of transaction.
    TransactionBegin,  // Value taken at start of transaction.
    TransactionEnd,    // Value taken at end of transaction.
```

```
    Trigger             // Value taken in response to a TriggerMessage
) UDINT;
END_TYPE
```

5.2.1.27 E_OCPP1_Reason

```
TYPE E_OCPP1_Reason :
(
    None := 0,      // Any status not covered by this implementation
    DeAuthorized,   // The transaction was stopped because of the authorization status in a StartTra
nsaction response
    EmergencyStop,  // Emergency stop button was used.
    EVDisconnected, // Disconnecting of cable, vehicle moved away from inductive charge unit.
    HardReset,      // A hard reset command was received.
    Local,          // Stopped locally on request of the user at the Charge Point.
    Other,          // Any other reason.
    PowerLoss,      // Complete loss of power.
    Reboot,         // A locally initiated reset / reboot occurred. (for instance watchdog kicked in
)
    Remote,         // Stopped remotely on request of the user. This is a regular termination of a t
ransaction.
    SoftReset,      // A soft reset command was received.
    UnlockCommand   // Central System sent an Unlock Connector command.
) UDINT;
END_TYPE
```

5.2.1.28 E_OCPP1_RecurrencyKindType

```
TYPE E_OCPP1_RecurrencyKindType :
(
    None := 0, // Any status not covered by this implementation
    Daily,     // The schedule restarts every 24 hours.
    Weekly     // The schedule restarts every 7 days.
) UDINT;
END_TYPE
```

5.2.1.29 E_OCPP1_RegistrationStatus

```
TYPE E_OCPP1_RegistrationStatus :
(
    None := 0, // Any status not covered by this implementation
    Accepted,  // Charge point is accepted by Central System.
    Pending,   // Central System is not yet ready to accept the Charge Point.
    Rejected   // Charge point is not accepted by Central System. This may happen when the Charge Po
int id is not known by Central System.
) UDINT;
END_TYPE
```

5.2.1.30 E_OCPP1_RemoteStartStopStatus

```
TYPE E_OCPP1_RemoteStartStopStatus :
(
    None := 0, // Any status not covered by this implementation
    Accepted,  // Command will be executed.
    Rejected   // Command will not be executed.
) UDINT;
END_TYPE
```

5.2.1.31 E_OCPP1_ReservationStatus

```
TYPE E_OCPP1_ReservationStatus :
(
    None := 0, // Any status not covered by this implementation
    Accepted,  // Reservation has been made.
    Faulted,   // Reservation has not been made, because connectors or specified connector are in a
faulted state.
    Occupied,  // Reservation has not been made.All connectors or the specified connector are occupi
ed.
    Rejected   // Reservation has not been made.Charge Point is not configured to accept reservation
s.
) UDINT;
END_TYPE
```

5.2.1.32 E_OCPP1_ResetStatus

```
TYPE E_OCPP1_ResetStatus :
(
    None := 0, // Any status not covered by this implementation
    Accepted,  // Configuration key is supported and setting has been changed.
    Rejected   // Configuration key is supported, but setting could not be changed.
) UDINT;
END_TYPE
```

5.2.1.33 E_OCPP1_ResetType

```
TYPE E_OCPP1_ResetType :
(
    None := 0, // Any status not covered by this implementation
    Hard,      // Restart(all) the hardware, the Charge Point is not required to gracefully stop ong
oing transaction.
    Soft       // Stop ongoing transactions gracefully and sending StopTransaction for every ongoing
 transaction.
) UDINT;
END_TYPE
```

5.2.1.34 E_OCPP1_TriggerMessageStatus

```
TYPE E_OCPP1_TriggerMessageStatus :
(
    None := 0,     // Any status not covered by this implementation
    Accepted,      // Requested notification will be sent.
    Rejected,      // Requested notification will not be sent.
    NotImplemented // Requested notification cannot be sent because it is either not implemented or
unknown.
) UDINT;
END_TYPE
```

5.2.1.35 E_OCPP1_Unit

```
TYPE E_OCPP1_Unit :
(
    None := 0,  // Any status not covered by this implementation
    Wh,         // watt-hours (energy).Default.
    kWh,        // kilo-watt-hours (energy).
    VARh,       // var-hours (reactive energy).
    kVARh,      // kilo-var-hours (reactive energy).
    W,          // watts (power).
    kW,         // kilo-watts (power).
    VA,         // volt-ampere (apparent power).
    kVA,        // kilo-volt-ampere (apparent power).
    VARs,       // vars (reactive power).
    kVAr,       // kilo-vars (reactive power).
    A,          // amperes (current).
    V,          // voltage (RMS AC).
    Celsius,    // degrees (temperature).
    Fahrenheit, // degrees (temperature).
    K,          // degrees kelvin (temperature).
    Percent     // percentage.
) UDINT;
END_TYPE
```

5.2.1.36 E_OCPP1_UnlockStatus

```
TYPE E_OCPP1_UnlockStatus :
(
    None := 0,    // Any status not covered by this implementation
    Unlocked,     // Connector has successfully been unlocked.
    UnlockFailed, // Failed to unlock the connector.
    NotSupported  // Charge Point has no connector lock, or ConnectorId is unknown.
) UDINT;
END_TYPE
```

5.2.1.37 E_OCPP1_UpdateStatus

```
TYPE E_OCPP1_UpdateStatus :
(
    None := 0,      // Any status not covered by this implementation
    Accepted,       // Local Authorization List successfully updated.
```

```
    Failed,         // Failed to update the Local Authorization List.
    NotSupported,   // Update of Local Authorization List is not supported by Charge Point.
    VersionMismatch // Version number in the request for a differential update is less or equal then
 version number of current list.
) UDINT;
END_TYPE
```

5.2.1.38 E_OCPP1_UpdateType

```
TYPE E_OCPP1_UpdateType :
(
    None := 0,    // Any status not covered by this implementation
    Differential, // Indicates that the current Local Authorization List must be updated with the va
lues in this message.
    Full          // Indicates that the current Local Authorization List must be replaced by the val
ues in this message.
) UDINT;
END_TYPE
```

5.2.1.39 E_OCPP1_ValueFormat

```
TYPE E_OCPP1_ValueFormat :
(
    None := 0, // Any status not covered by this implementation
    Raw,       // Data is to be interpreted as integer / decimal numeric data.
    SignedData // Data is represented as a signed binary data block, encoded as hex data.
) UDINT;
END_TYPE
```

5.2.1.40 E_OCPP_DebugLevel

```
TYPE E_OCPP_DebugLevel :
(
    None              := 0, // No Encryption
    MessageLogFile    := 1  // Write a log file of all messages to the boot directory.
) UDINT;
END_TYPE
```

5.2.1.41 E_OCPP_EncryptionMode

```
TYPE E_OCPP_EncryptionMode :
(
    None              := 0, // No Encryption
    Enable            := 1, // TLS enable, but without certificate validation (insecure)
    ServerCertificate := 2, // TLS enable, with server certificate validation
    ClientCertificate := 3  // TLS enable, with client certificate authentication
) UDINT;
END_TYPE
```

5.2.1.42 E_OCPP_EncryptionProtocol

```
TYPE E_OCPP_EncryptionProtocol :
(
    None    := 0,
    TLSv1p2 := 2, // TLS version 1.2
    TLSv1p3 := 3  // TLS version 1.3
) UDINT;
END_TYPE
```

5.2.2 Properties

5.2.2.1 ST_OCPP1_Client_BootInfo

In dieser Struktur werden die in der Boot Notification versendeten Parameter zusammengefasst.

Syntax

```
TYPE ST_OCPP_Client1_BootInfo :
STRUCT
    sChargePointModel        : STRING(23) := 'TcIotOcpp';
    sChargePointVendor       : STRING(23) := 'Beckhoff Automation';
```

```
    sChargeBoxSerialNumber   : STRING(23);
    sChargePointSerialNumber : STRING(23);
    sFirmwareVersion         : STRING(23);
    sMeterSerialNumber       : STRING(23);
    sMeterType               : STRING(23);
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Optional (OCPP)	Beschreibung
sChargePointModel	STRING(23)	Nein	Modell des Charge Points.
sChargePointVendor	STRING(23)	Nein	Hersteller des Charge Points.
sChargeBoxSerialNumber	STRING(23)	Ja	Seriennummer der Charge Box innerhalb des Charge Points.
sChargePointSerialNumber	STRING(23)	Ja	Seriennummer des Charge Points.
sFirmwareVersion	STRING(23)	Ja	Firmwareversion des Charge Points.
sMeterSerialNumber	STRING(23)	Ja	Seriennummer des Hauptstromzählers des Charge Points.
sMeterType	STRING(23)	Ja	Typ des Hauptstromzählers des Charge Points.

5.2.2.2 ST_OCPP1_Client_Options

Mithilfe dieser Struktur können optionale Funktionen im OCPP-Client ein- und ausgeschaltet werden.

Syntax

```
TYPE ST_OCPP_Client1_Options :
STRUCT
    bAllowOfflineTxForUnknownId : BOOL := FALSE;
    bAuthListEnabled            : BOOL := TRUE;
    bAuthCacheEnabled           : BOOL := TRUE;
    bLocalAuthOffline           : BOOL := TRUE;
    bLocalPreAuth               : BOOL := TRUE;
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Optional (OCPP)	Beschreibung
bAllowOfflineTxForUnknownId	BOOL	Nein	Bei TRUE ist Unknown Offline Authorization aktiviert.
bAuthListEnabled	BOOL	Nein	Bei TRUE ist die Local Authorization List aktiviert.
bAuthCacheEnabled	BOOL	Ja	Bei TRUE ist der Authorization Cache aktiviert.
bLocalAuthOffline	BOOL	Nein	Bei TRUE starten die Charge Points eine Transaction für lokal autorisierte Identifier.
bLocalPreAuth	BOOL	Nein	Bei TRUE starten die Charge Points eine Transaction für lokal autorisierte Identifier ohne auf das Central System zu warten.

5.2.2.3 ST_OCPP1_Client_Param

Mithilfe dieser Struktur werden die Parameter der WebSockets-Verbindung des OCPP-Clients gesetzt.

Syntax

```
TYPE ST_OCPP_Client1_Param :
STRUCT
    nOID            : OTCID := 0;
    nTaskOID        : OTCID := 0;
    bConnect        : BOOL := TRUE;
    sHost           : T_MaxString := '';
    nPort           : UINT := 443;
    sPath           : T_MaxString;
    sIdentity       : T_MaxString;
    eAuthMode       : E_OCPP1_AuthenticationMode;
    sAuthKey        : T_MaxString;
    eEncryptionMode : E_OCPP_EncryptionMode;
    eEncryptionProt : E_OCPP_EncryptionProtocol;
    sCaFile         : T_MaxString := '%TC_TARGETPATH%\Certificates\CA.crt';
    sCrtFile        : T_MaxString := '%TC_TARGETPATH%\Certificates\OCPP.crt';
    sKeyFile        : T_MaxString := '%TC_TARGETPATH%\Certificates\OCPP.key';
    eDebugLevel     : E_OCPP_DebugLevel := E_OCPP_DebugLevel.None;
    eTraceLevel     : TcTraceLevel := TcTraceLevel.tlWarning;
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Beschreibung
nOID	OTCID	ObjectID des TcIotOcppClient-Objekts. Wenn dieser Wert auf 0 (default) gelassen wird, wird eine neue Instanz erzeugt.
nTaskOID	OTCID	ObjectID der verwendeten Task. Wenn dieser Wert auf 0 (default) gelassen wird, wird die I/O Idle Task verwendet.
bConnect	BOOL	Wenn TRUE, dann verbindet sich der OCPP-Client automatisch mit dem OCPP-Server. Wenn FALSE, muss die Verbindung mittels der Methode Execute [▶ 23] hergestellt werden.
sHost	T_MaxString	Hostname oder IP-Adresse des OCPP-Servers.
nPort	UINT	Port des OCPP-Servers.
sPath	T_MaxString	Gibt optional eine URI des OCPP-Servers an.
sIdentity	T_MaxString	Legt die Identity des Clients fest.
eAuthMode	E_OCPP1_AuthenticationMode [▶ 191]	Optionaler Authentication Mode.
sAuthKey	T_MaxString	Optionaler Authentication Key.
eEncryptionMode	E_OCPP_EncryptionMode [▶ 199]	Optionaler Encryption Mode.
eEncryptionProt	E_OCPP_EncryptionProtocol [▶ 199]	Definiert das Encryption Protocol.
sCaFile	T_MaxString	Zertifikat der Certificate Authority (CA) als Dateipfad (PEM- oder DER-Format).
sCrtFile	T_MaxString	Zertifikat des Clients (Public Key) als Dateipfad (PEM- oder DER-Format).
sKeyFile	T_MaxString	Private Key des Clients als Dateipfad (PEM- oder DER-Format).
eDebugLevel	E_OCPP_DebugLevel [▶ 199]	Das Debug-Level (None oder MessageLogFile). MessageLogFile sorgt dafür, dass eine Log-Datei mit allen OCPP-Nachrichten ins TwinCAT-Boot-Verzeichnis geschrieben wird.
eTraceLevel	TcTraceLevel	Das maximale Trace-Level vom ADS-Logging.

5.2.2.4 ST_OCPP1_Client_Settings

Mithilfe dieser Struktur können weitere Einstellungen im OCPP-Client getroffen werden.

Syntax

```
TYPE ST_OCPP_Client1_Settings :
STRUCT
    tConnectTimeout:                : TIME := T#10S;
    tReconnectTimeout:              : TIME := T#10S;
    tBootTimeout:                   : TIME := T#5S;
    tMessageTimeout:                : TIME := T#5S;
    nTransactionMessageAttempts     : UDINT := 5;
```

```
    tTransactionMessageRetryInterval : TIME := T#5S;
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Beschreibung
tConnectTimeout	TIME	Spezifiziert die Zeit, nach der ein Verbindungsversuch abgebrochen wird.
tReconnectTimeout	TIME	Spezifiziert die Zeit nach einem Disconnect, bis der Client wieder versucht, eine Verbindung aufzubauen.
tBootTimeout	TIME	Spezifiziert die Zeit, wie lange auf eine Antwort auf die Boot Notification gewartet wird.
tMessageTimeout	TIME	Spezifiziert die Zeit, wie lange auf eine Antwort auf eine Message gewartet wird.
nTransactionMessageAttempts	UDINT	Spezifiziert, wie oft der Charge Point versucht Transkationen betreffende Nachrichten nochmal zu senden, wenn das Central System diese nicht verarbeitet hat.
tTransactionMessageRetryInterval	TIME	Spezifiziert die Zeit in Sekunden, nach der ein weiterer Versuch des Sendens unternommen wird.

5.2.2.5 ST_OCPP1_Server_Param

Mithilfe dieser Struktur werden die Parameter der WebSockets-Verbindung des OCPP-Servers gesetzt.

Syntax

```
TYPE ST_OCPP1_ServerParam :
STRUCT
    nOID            : OTCID := 0;
    nTaskOID        : OTCID := 0;
    sHost           : T_MaxString := '0.0.0.0';
    nPort           : UINT := 443;
    sPath           : T_MaxString := 'ocpp';
    eAuthMode       : E_OCPP1_AuthenticationMode := E_OCPP1_AuthenticationMode.Basic;
    eEncryptionMode : E_OCPP_EncryptionMode := E_OCPP_EncryptionMode.Enable;
    eEncryptionProt : E_OCPP_EncryptionProtocol;
    sCaFile         : T_MaxString := '%TC_TARGETPATH%\Certificates\CA.crt';
    sCrtFile        : T_MaxString := '%TC_TARGETPATH%\Certificates\OCPP.crt';
    sKeyFile        : T_MaxString := '%TC_TARGETPATH%\Certificates\OCPP.key';
    eDebugLevel     : E_OCPP_DebugLevel := E_OCPP_DebugLevel.None;
    eTraceLevel     : TcTraceLevel := TcTraceLevel.tlWarning;
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Beschreibung
nOID	OTCID	ObjectID des TcIotOcppServer-Objekts. Wenn dieser Wert auf 0 (default) gelassen wird, wird eine neue Instanz erzeugt.
nTaskOID	OTCID	ObjectID der verwendeten Task. Wenn dieser Wert auf 0 (default) gelassen wird, wird die I/O Idle Task verwendet.
sHost	T_MaxString	Definiert die Server-Adresse als Hostname, Domainname oder IP-Adresse.
nPort	UINT	Definiert den Server-Port.
sPath	T_MaxString	Definiert den Server-Pfad.
eAuthMode	E_OCPP1_AuthenticationMode [▶ 191]	Optionaler Authentication Mode.
eEncryptionMode	E_OCPP_EncryptionMode [▶ 199]	Optionaler Encryption Mode.
eEncryptionProt	E_OCPP_EncryptionProtocol [▶ 199]	Definiert das Encryption Protocol.
sCaFile	T_MaxString	Zertifikat der Certificate Authority (CA) als Dateipfad (PEM- oder DER-Format).
sCrtFile	T_MaxString	Zertifikat des Servers (Public Key) als Dateipfad (PEM- oder DER-Format).
sKeyFile	T_MaxString	Private Key des Servers als Dateipfad (PEM- oder DER-Format).
eDebugLevel	E_OCPP_DebugLevel [▶ 199]	Das Debug-Level (None oder MessageLogFile). MessageLogFile sorgt dafür, dass eine Log-Datei mit allen OCPP-Nachrichten ins TwinCAT-Boot-Verzeichnis geschrieben wird.
eTraceLevel	TcTraceLevel	Das maximale Trace-Level vom ADS-Logging.

5.2.3 Types

5.2.3.1 ST_OCPP1_AuthorizationData

Syntax

```
TYPE ST_OCPP1_AuthorizationData :
STRUCT
    sIdTag      : T_OCPP1_IdToken;
    stIdTagInfo : ST_OCPP1_IdTagInfo;
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Optional	Beschreibung
sIdTag	T_OCPP1_IdToken [▶ 212]	Nein	ID-Token, zu dem diese Autorisierungsinformationen gehören.
stIdTagInfo	ST_OCPP1_IdTagInfo [▶ 210]	Ja	Enthält Informationen über Authorization Status, Ablaufdatum und Parent-ID.

5.2.3.2 ST_OCPP1_ChargingProfile

Syntax

```
TYPE ST_OCPP1_ChargingProfile :
STRUCT
    nProfileId      : UDINT;
    nTransactionId  : UDINT;
    nStackLevel     : UDINT;
    eProfilePurpose : E_OCPP1_ChargingProfilePurposeType;
    eProfileKind    : E_OCPP1_ChargingProfileKindType;
    eRecurrencyKind : E_OCPP1_RecurrencyKindType;
    nValidFrom      : ULINT;
    nValidTo        : ULINT;
    mSchedule       : ST_OCPP1_ChargingSchedule;
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Optional	Beschreibung
nProfileId	UDINT	Nein	Eindeutiger Identifier für dieses Charging Profile.
nTransactionId	UDINT	Ja	Nur gültig, wenn eProfilePurpose auf TxProfile gesetzt ist. Die Transaction-ID kann verwendet werden, um das Profil einer bestimmten Transaktion zuzuordnen.
nStackLevel	UDINT	Nein	Wert, der die Priorität eines Charging Profiles festlegt. Der Wert 0 definiert die niedrigste Priorität.
eProfilePurpose	E_OCPP1_ChargingProfilePurposeType [▶ 193]	Nein	Zweck des übermittelten Charging Profiles.
eProfileKind	E_OCPP1_ChargingProfileKindType [▶ 193]	Nein	Bestimmt die Art des Charging Profiles.
eRecurrencyKind	E_OCPP1_RecurrencyKindType [▶ 197]	Ja	Gibt den Startpunkt einer Wiederholung an.
nValidFrom	ULINT	Ja	Zeitpunkt, ab dem das Charging Profile gültig ist. Wenn kein Wert angegeben ist, ist das Charging Profile gültig, sobald es den Charge Point erreicht.
nValidTo	ULINT	Ja	Zeitpunkt, ab dem das Charging Profile nicht mehr gültig ist. Wenn kein Wert angegeben ist, ist das Charging Profile gültig, bis der Charge Point ein neues Charging Profile erhält.
mSchedule	ST_OCPP1_ChargingSchedule [▶ 207]	Nein	Enthält die Charging Schedule und damit die Grenzwerte für die verfügbare Leistung oder den Strom über die Zeit.

5.2.3.3 ST_OCPP1_ChargingProfileMax

Syntax

```
TYPE ST_OCPP1_ChargingProfileMax :
STRUCT
    nProfileId      : UDINT;
    nTransactionId  : UDINT;
    nStackLevel     : UDINT;
    eProfilePurpose : E_OCPP1_ChargingProfilePurposeType;
    eProfileKind    : E_OCPP1_ChargingProfileKindType;
    eRecurrencyKind : E_OCPP1_RecurrencyKindType;
    nValidFrom      : ULINT;
    nValidTo        : ULINT;
    mSchedule       : ST_OCPP1_ChargingScheduleMax;
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Optional	Beschreibung
nProfileId	UDINT	Nein	Eindeutiger Identifier für dieses Charging Profile.
nTransactionId	UDINT	Ja	Nur gültig, wenn eProfilePurpose auf TxProfile gesetzt ist. Die Transaction-ID kann verwendet werden, um das Profil einer bestimmten Transaktion zuzuordnen.
nStackLevel	UDINT	Nein	Wert, der die Priorität eines Charging Profiles festlegt. Der Wert 0 definiert die niedrigste Priorität.
eProfilePurpose	E_OCPP1_ChargingProfilePurposeType [▶ 193]	Nein	Zweck des übermittelten Charging Profiles.
eProfileKind	E_OCPP1_ChargingProfileKindType [▶ 193]	Nein	Bestimmt die Art des Charging Profiles.
eRecurrencyKind	E_OCPP1_RecurrencyKindType [▶ 197]	Ja	Gibt den Startpunkt einer Wiederholung an.
nValidFrom	ULINT	Ja	Zeitpunkt, ab dem das Charging Profile gültig ist. Wenn kein Wert angegeben ist, ist das Charging Profile gültig, sobald es den Charge Point erreicht.
nValidTo	ULINT	Ja	Zeitpunkt, ab dem das Charging Profile nicht mehr gültig ist. Wenn kein Wert angegeben ist, ist das Charging Profile gültig, bis der Charge Point ein neues Charging Profile erhält.
mSchedule	ST_OCPP1_ChargingScheduleMax [▶ 208]	Nein	Enthält die Charging Schedule und damit die Grenzwerte für die verfügbare Leistung oder den Strom über die Zeit.

5.2.3.4 ST_OCPP1_ChargingSchedule

Syntax

```
TYPE ST_OCPP1_ChargingSchedule :
STRUCT
    nStart       : ULINT;
    nDuration    : UDINT;
    eRateUnit    : E_OCPP1_ChargingRateUnitType;
    fRateMin     : REAL;
    nPeriodCount : UDINT;
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Optional	Beschreibung
nStart	ULINT	Ja	Startpunkt einer absoluten Charging Schedule. Wenn der Wert nicht angegeben ist, ist der Zeitplan relativ zum Start des Ladevorgangs.
nDuration	UDINT	Ja	Dauer der Charging Schedule in Sekunden. Wenn der Wert nicht angegeben ist, wird die letzte Charging Schedule Period unbegrenzt oder bis zum Ende der Transaktion fortgesetzt.
eRateUnit	E_OCPP1_ChargingRateUnitType [▶ 193]	Nein	Die Einheit des Limits, zu erwarten in Watt (Leistung) oder Ampere (Strom).
fRateMin	REAL	Ja	Mindestrate zum Laden, die von dem elektrischen Fahrzeug unterstützt wird. Die Einheit wird durch eRateUnit festgelegt.
nPeriodCount	UDINT	Nein	Anzahl der folgenden Charging Schedule Periods.

5.2.3.5 ST_OCPP1_ChargingScheduleMax

Syntax

```
TYPE ST_OCPP1_ChargingScheduleMax :
STRUCT
    nStart       : ULINT;
    nDuration    : UDINT;
    eRateUnit    : E_OCPP1_ChargingRateUnitType;
    fRateMin     : REAL;
    nPeriodCount : UDINT;
    arrPeriods   : ARRAY[1..*] OF ST_OCPP1_ChargingSchedulePeriod;
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Optional	Beschreibung
nStart	ULINT	Ja	Startpunkt einer absoluten Charging Schedule. Wenn der Wert nicht angegeben ist, ist der Zeitplan relativ zum Start des Ladevorgangs.
nDuration	UDINT	Ja	Dauer der Charging Schedule in Sekunden. Wenn der Wert nicht angegeben ist, wird die letzte Charging Schedule Period unbegrenzt oder bis zum Ende der Transaktion fortgesetzt.
eRateUnit	E_OCPP1_ChargingRateUnitType [▶ 193]	Nein	Die Einheit des Limits, zu erwarten in Watt (Leistung) oder Ampere (Strom).
fRateMin	REAL	Ja	Mindestrate zum Laden, die von dem elektrischen Fahrzeug unterstützt wird. Die Einheit wird durch eRateUnit festgelegt.
nPeriodCount	UDINT	Nein	Anzahl der folgenden Charging Schedule Periods.
arrPeriods	ARRAY[1..Param_OCPP.nSchedulePeriodsMax] OF ST_OCPP1_ChargingSchedulePeriod [▶ 209]	Nein	Liste der Charging Schedule Period-Elemente.

5.2.3.6 ST_OCPP1_ChargingSchedulePeriod

Syntax

```
TYPE ST_OCPP1_ChargingSchedulePeriod :
STRUCT
    nStart     : UDINT;
    fLimit     : REAL;
    nPhases    : UDINT := 0;
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Optional	Beschreibung
nStart	UDINT	Nein	Start der Charging Schedule Period, in Sekunden ab Start der Charging Schedule.
fLimit	REAL	Nein	Charging Rate Limit während der jeweiligen Charging Schedule Period.
nPhases	UDINT	Ja	Die Anzahl an Phasen, die zum Laden genutzt werden können.

5.2.3.7 ST_OCPP1_ConfigKeyValue

Syntax

```
TYPE ST_OCPP1_ConfigKeyValue :
STRUCT
    sKey         : T_OCPP1_ConfigKey;
    sValue       : T_OCPP1_ConfigValue;
    bReadonly    : BOOL;
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Optional	Beschreibung
sKey	T_OCPP1_ConfigKey [▶ 211]	Nein	Configuration Key, der überschrieben oder hinzugefügt werden soll.
sValue	T_OCPP1_ConfigValue [▶ 211]	Ja	Wert des zugehörigen Configuration Keys. Kann weggelassen werden, wenn der Configuration Key bekannt ist und nicht neu gesetzt werden soll.
bReadonly	BOOL	Nein	Bestimmt das Zugriffsrecht des Central Systems, entweder darf der Wert nur gelesen (TRUE) oder auch geschrieben werden (FALSE).

5.2.3.8 ST_OCPP1_IdTagInfo

Syntax

```
TYPE ST_OCPP1_IdTagInfo :
STRUCT
    eStatus      : E_OCPP1_AuthorizationStatus;
    nExpiryDate  : ULINT;
    sParentIdTag : T_OCPP1_IdToken;
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Optional	Beschreibung
eStatus	E_OCPP1_AuthorizationStatus [▶ 191]	Nein	Enthält die Information, ob der ID-Tag vom Central System akzeptiert wurde.
nExpiryDate	ULINT	Ja	Enthält optional das Datum, an dem der ID-Tag aus dem Authorization Cache entfernt werden soll.
sParentIdTag	T_OCPP1_IdToken [▶ 212]	Ja	Enthält optional den Parent ID-Tag

5.2.3.9 ST_OCPP1_SampledValue

Aus Sicht der OCPP-Spezifikation ist ein Meter Value ein Sampled Value mit Zeitstempel. In der vorliegenden SPS-Bibliothek wird der Zeitstempel ebenfalls im Sampled Value geführt, um eine Strukturebene weniger zu haben.

Syntax

```
TYPE ST_OCPP1_SampledValue :
STRUCT
    fValue     : LREAL;
    nTimestamp : ULINT := 0;
    eContext   : E_OCPP1_ReadingContext := E_OCPP_ReadingContext.SamplePeriodic;
    eFormat    : E_OCPP1_ValueFormat := E_OCPP_ValueFormat.Raw;
    eMeasurand : E_OCPP1_Measurand;
    ePhase     : E_OCPP1_Phase;
    eLocation  : E_OCPP1_Location;
    eUnit      : E_OCPP1_Unit;
END_STRUCT
END_TYPE
```

Parameter

Name	Typ	Optional	Beschreibung
fValue	LREAL	Nein	Sampled Value als dezimale Zahl („Raw“). „SignedData“ wird nicht unterstützt.
nTimestamp	ULINT	Ja	Zeitstempel des Sampled Values.
eContext	E_OCPP1_ReadingContext [▶ 196]	Ja	Kontext, in dem der Sampled Value aufgenommen wurde (Start, Ende oder Sample).
eFormat	E_OCPP1_ValueFormat [▶ 199]	Ja	In der vorliegenden Implementierung ist es immer das „Raw“-Format.
eMeasurand	E_OCPP1_Measurand [▶ 195]	Ja	Typ des Sampled Values.
ePhase	E_OCPP1_Phase [▶ 196]	Ja	Informationen über die Interpretation des Sampled Values.
eLocation	E_OCPP1_Location [▶ 195]	Ja	Ort des Sampled Values.
eUnit	E_OCPP1_Unit [▶ 198]	Ja	Einheit des Sampled Values.

5.2.3.10 T_OCPP1_ConfigKey

Syntax

```
TYPE T_OCPP1_ConfigKey : STRING(63); END_TYPE
```

5.2.3.11 T_OCPP1_ConfigValue

Syntax

```
TYPE T_OCPP1_ConfigValue : STRING(183); END_TYPE
```

5.2.3.12 T_OCPP1_DataTransferData

Syntax

```
TYPE T_OCPP1_DataTransferData : STRING(Param_OCPP.nDataTransferMax); END_TYPE
```

5.2.3.13 T_OCPP1_IdToken

Syntax

```
TYPE T_OCPP1_IdToken : STRING(23); END_TYPE
```

5.2.3.14 T_OCPP_Hash

Syntax

```
TYPE T_OCPP_Hash : STRING(95); END_TYPE
```

5.2.3.15 T_OCPP_Identity

Syntax

```
TYPE T_OCPP_Identity : STRING(31); END_TYPE
```

5.2.3.16 T_OCPP_MessageId

Syntax

```
TYPE T_OCPP_MessageId : STRING(63); END_TYPE
```

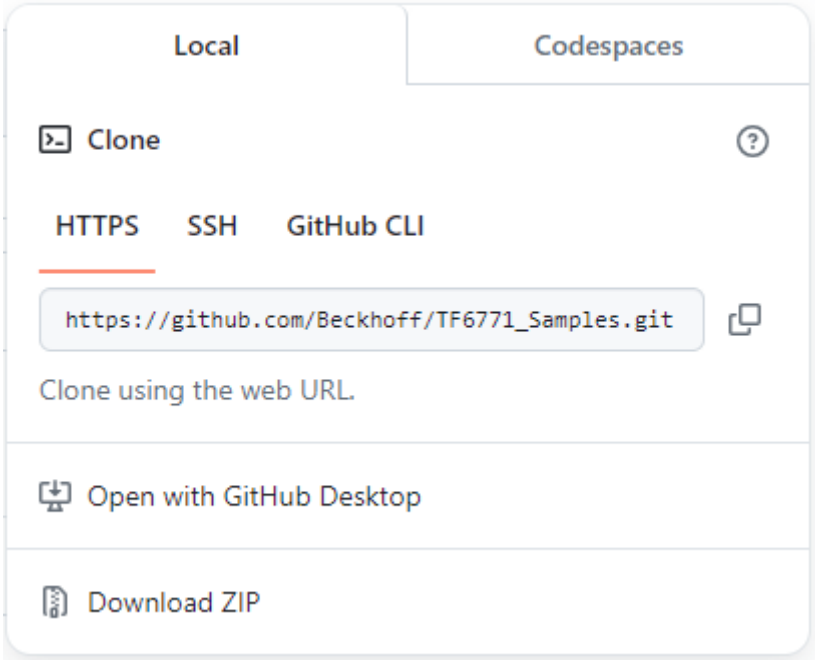
5.3 GVLs

5.3.1 Param_OCPP

Name	Typ	Default-Wert	Beschreibung
nConnectorCount	UDINT	2	Maximale Anzahl von Konnektoren pro Charge Point.
nSampledValuesMax	UDINT	64	Maximale Anzahl von SampledValues in einem MeterValue.
nChargingProfileMax	UDINT	10	Maximale Anzahl an Charging Profiles.
nSchedulePeriodsMax	UDINT	24	Maximale Anzahl von Charging Schedule Periods in einer Charging Schedule.
nConfigKeysMax	UDINT	100	Maximale Anzahl von Configuration Keys in einem GetConfiguration-Aufruf.
nAuthListMax	UDINT	10	Maximale Anzahl an Einträgen in der Local Authorization List.
fChargingLimitMax	REAL	40	Standard Ladeleistungsgrenze für Zeitpläne.
nDataTransferMax	UDINT	1023	Maximale Anzahl an Bytes im Payload für die SendDataTransfer-Methode.

6 Beispiele

Beispielcode und -konfigurationen für dieses Produkt können über das entsprechende Repository auf GitHub bezogen werden: https://github.com/Beckhoff/TF6771_Samples. Sie haben dort die Möglichkeit das Repository zu clonen oder ein ZIP File mit den Samples herunterzuladen.



Die folgende Tabelle gibt auch hier in der Dokumentation eine Übersicht über die aktuell verfügbaren Samples:

Sample	Kurzbeschreibung
TF6771_IotOcppGettingStarted	Dieses Beispiel stellt ein einfaches Getting Started zur Verfügung und wird im Kapitel Quick Start [▶ 15] genauer beschrieben.
TF6771_IotOcppClient1	Dieses Beispiel stellt die grundsätzliche Verwendung des OCPP-Clients dar.
TF6771_IotOcppServer1	Dieses Beispiel stellt die grundsätzliche Verwendung des OCPP-Servers dar.

7 Anhang

7.1 Hash-Berechnung

Der Hash setzt sich aus Identität (Benutzername) und Passwort zusammen. Diese werden durch einen Doppelpunkt getrennt.

%username%:%password%

Bei der Hash-Berechnung wird der SHA256-Algorithmus verwendet.

Beispiel:

Identität (Benutzername): BeckhoffOCPP

Passwort: 1234abc$

Berechneter Hash: 30206a1e174ac04ea140234f2c1edd854d8dbca1d664b92d763cb5ef78c8e8dc

Es kann an dieser Stelle jeder beliebige SHA256-Rechner genutzt werden. Die finden sich beispielsweise online oder als Funktionen in verschiedenen Bibliotheken.

7.2 Configuration Keys

Hier finden Sie alle Configuration Keys, die mit der Methode InitStandard [▶ 187] initialisiert werden.

Name	Profil	Wert	Beschreibung
GetConfigurationMaxKeys	Core (9.1)	Param_OCPP [▶ 212].nConfigKeysMax	Maximale Anzahl von Configuration Keys in einem GetConfiguration-Aufruf.
NumberOfConnectors	Core (9.1)	Param_OCPP [▶ 212].nConnectorCount	Maximale Anzahl von Konnektoren pro Charge Point.
SupportedFeatureProfiles	Core (9.1)	Core, Reservation, SmartCharging, RemoteTrigger	Liste aller unterstützten Feature Profiles.
LocalAuthListMaxLength	Local Auth List Management (9.2)	Param_OCPP [▶ 212].nAuthListMax	Maximale Anzahl an Einträgen in der Local Authorization List.
SendLocalListMaxLength	Local Auth List Management (9.2)	Param_OCPP [▶ 212].nAuthListMax	Maximale Anzahl an Einträgen, die in einem SendLocalList [▶ 122] an den Charge Point gesendet werden können.
ChargingScheduleAllowedChargingRateUnit	Smart Charging (9.4)	Current	Liste der unterstützten Mengen, die in einer ChargingSchedule verwendet werden dürfen. Erlaubte Werte sind “Current” und “Power”.
ChargingScheduleMaxPeriods	Smart Charging (9.4)	Param_OCPP [▶ 212].nSchedulePeriodsMax	Maximale Anzahl von Charging Schedule Periods in einer Charging Schedule.
MaxChargingProfilesInstalled	Smart Charging (9.4)	Param_OCPP [▶ 212].nChargingProfileMax	Maximale Anzahl an Charging Profiles, die parallel installiert sein können.

7.3 Troubleshooting

Verhalten	Erläuterung
Das Heartbeat-Intervall im Client wurde geändert. Nach Herstellung der Verbindung zum Server ist jedoch ein anderer Wert gesetzt.	In der BootNotification.conf sendet der OCPP-Server ein vorgesehenes Heartbeat-Intervall für den Client. Dieses wird intern im Client auch gesetzt. Es ist jedoch im Anschluss möglich, zur Laufzeit das interne HeartbeatInterval zu ändern.

Mehr Informationen:
www.beckhoff.com/tf6771/

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Deutschland
Telefon: +49 5246 9630
info@beckhoff.com
www.beckhoff.com