Инструкция по эксплуатации на «Программный комплекс управления зарядной станцией РЕОН» после установки

1. Программа не имеет визуального интерфейса так как является внутренней программой управления станцией и имеет только backend часть.
2. После подключения к websocket OCCP сервера оператора взаимодействие, мониторинг и обработка сообщений происходит в рамках протокола OCPP 2.0.1.
3. При успешном подключении станция присылает транзакцию «boot notification» следующего вида:

BootNotificationRequest shifted from queue and send reason: 'PowerUp',

chargingStation:

serialNumber: 'уникальный ключ идентификации станции', model: 'REON',

vendorName: 'REWATT',

firmwareVersion: '0.1.1',

modem: { iccid: ххххххх, imsi: ххххххх }

1. Далее пользователь может выводить информацию о работе станции стандартными методами визуализации данных.



Пример вывода программы

5. Примеры вывода сообщений

**Ответ на BootNotification:**

{

 "status": "Accepted",

 "currentTime": "2023-06-10T12:34:56Z",

 "interval": 3600

}

Запрос StatusNotification:

{

 "action": "StatusNotification",

 "payload": {

 "connectorStatus": "Available",

 "evseId": 1,

 "connectorId": 1

 }

}

**Ответ на StatusNotification:**

{

 "status": "Accepted"

}

**Запрос TransactionEvent (начало транзакции):**

{

 "action": "TransactionEvent",

 "payload": {

 "eventType": "Started",

 "timestamp": "2023-06-10T12:34:56Z",

 "triggerReason": "Authorized",

 "seqNo": 0,

 "transactionInfo": {

 "transactionId": "1",

 "chargingState": "Charging"

 },

 "evse": {

 "id": 1

 },

 "connectorId": 1,

 "idToken": {

 "idToken": "123456",

 "type": "ISO14443"

 },

 "meterValue": [

 {

 "sampledValue": [

 {

 "value": "0",

 "context": "Transaction.Begin",

 "measurand": "Energy.Active.Import.Register",

 "unit": "Wh"

 }

 ],

 "timestamp": "2023-06-10T12:34:56Z"

 }

 ]

 }

}

**Ответ на TransactionEvent (начало транзакции):**

{

 "status": "Accepted"

}

**Запрос TransactionEvent (окончание транзакции):**

{

 "action": "TransactionEvent",

 "payload": {

 "eventType": "Ended",

 "timestamp": "2023-06-10T14:34:56Z",

 "triggerReason": "EVCommunicationLost",

 "seqNo": 1,

 "transactionInfo": {

 "transactionId": "1",

 "chargingState": "EvDisconnected"

 },

 "evse": {

 "id": 1

 },

 "connectorId": 1,

 "meterValue": [

 {

 "sampledValue": [

 {

 "value": "10000",

 "context": "Transaction.End",

 "measurand": "Energy.Active.Import.Register",

 "unit": "Wh"

 }

 ],

 "timestamp": "2023-06-10T14:34:56Z"

 }

 ]

 }

}

**Ответ на TransactionEvent (окончание транзакции):**

{

 "status": "Accepted"

}

Запрос GetVariables:

{

 "action": "GetVariables",

 "payload": {

 "getVariableData": [

 {

 "component": {

 "name": "EVSE"

 },

 "variable": {

 "name": "AvailabilityStatus"

 },

 "evseId": 1

 }

 ]

 }

}

**Ответ на GetVariables:**

{

 "getVariableResult": [

 {

 "status": "Accepted",

 "component": {

 "name": "EVSE"

 },

 "variable": {

 "name": "AvailabilityStatus"

 },

 "attributeStatus": "Supported",

 "attributeValue": "Available",

 "evseId": 1

 }

 ]

}

**Запрос SetVariables:**

{

 "action": "SetVariables",

 "payload": {

 "setVariableData": [

 {

 "component": {

 "name": "EVSE"

 },

 "variable": {

 "name": "AvailabilityStatus"

 },

 "attributeValue": "Unavailable",

 "evseId": 1

 }

 ]

 }

}

**Ответ на SetVariables:**

{

 "setVariableResult": [

 {

 "status": "Accepted",

 "component": {

 "name": "EVSE"

 },

 "variable": {

 "name": "AvailabilityStatus"

 },

 "evseId": 1

 }

 ]

}

**Запрос UpdateFirmware:**

{

 "action": "UpdateFirmware",

 "payload": {

 "location": "https://firmware-update.example.com/new\_firmware.bin",

 "retrieveDate": "2023-04-10T12:00:00Z",

 "firmwareVersion": "2.0.0"

 }

}

**Ответ на UpdateFirmware:**

{

 "status": "Accepted"

}