EV Charging Station Management System

Functional Requirements Document

1. System Overview

This document outlines the functional requirements for an Electric Vehicle (EV) charging station management system that implements the Open Charge Point Protocol (OCPP) version 1.6. The system consists of charge points (client) communicating with a central management system (server) via WebSocket connections.

2. Communication Protocol

- **Protocol**: OCPP 1.6
- **Connection**: WebSocket (wss://)
- **Configuration**: Each charge point is configured with:
    - Unique Charge Point ID
    - Central System WebSocket URL

3. Connection & Boot Sequence

3.1 Initial Connection

1. Charge point establishes WebSocket connection to central system
2. Connection format: `{centralSystemUrl}/{chargePointId}`

3.2 Boot Notification

1. Upon successful boot, charge point sends `BootNotification` request containing:

    - Charge Point Model
    - Charge Point Vendor
    - Charge Point Serial Number
    - Firmware Version
    - ICCID (if applicable)
    - IMSI (if applicable)
    - Meter Type (if applicable)
    - Meter Serial Number (if applicable)

2. Central system responds with:
    - Status (`Accepted`, `Rejected`)
    - Current time (for clock synchronization)
    - Heartbeat interval (set to 45 seconds)
3. Central system uses boot notification parameters to authenticate the charge point

4. Operational Communication

4.1 Heartbeat Mechanism

1. Charge point sends `Heartbeat` request every 45 seconds
2. Central system responds with current time
3. If heartbeat is delayed more than 60 seconds:
    - Charge point is marked as offline
    - Status is restored once heartbeat resumes or new boot notification is received

4.2 Status Notifications

1. Charge point sends `StatusNotification` when it's status changes:
    - Available
    - Preparing
    - Charging
    - SuspendedEVSE
    - SuspendedEV
    - Finishing
    - Reserved
    - Unavailable
    - Faulted
2. Central system uses status updates to:
    - Display current charge point status to users
    - Determine if charge point is available for new transactions
    - Perform pre-transaction checks

5. Transaction Flow

5.1 Starting a Transaction

1. User initiates charging through web application
2. Central system verifies charge point status is appropriate (Available)
3. Central system requests meter values via `GetMeterValues` request

4. Once meter values are received, central system sends `RemoteStartTransaction` with:
    - User ID
    - Transaction ID
    - Charging profile (if applicable)
5. Charge point begins charging and sends `StartTransaction` confirmation

5.2 During Transaction

1. Charge point sends `MeterValues` every 30 seconds containing:
    - Transaction ID
    - Meter reading (kWh)
    - Timestamp
    - Current, voltage, power readings (if available)
2. Central system stores meter values for monitoring and billing purposes

5.3 Stopping a Transaction

1. User stops charging via web application
2. Central system sends `RemoteStopTransaction` request with transaction ID
3. Charge point stops charging and sends final `MeterValues`
4. Charge point sends `StopTransaction` with:
    - Transaction ID
    - Final meter reading
    - Timestamp
    - Reason for stop
5. Central system calculates energy consumption:
    - Energy Used = Final Meter Reading - Initial Meter Reading
6. Central system handles user billing based on energy consumed
7. Note: Charger has to send StopTransaction when charger disconnects from vehicle

6. Error Handling & Recovery

6.1 Communication Failures

1. If WebSocket connection is lost:

    - Charge point attempts to reconnect with exponential backoff:
        - Initial retry after 5 seconds
        - Subsequent retries: 10s, 20s, 40s, 80s, capped at 5 minutes
    - Maximum retry attempts: 10 before requiring manual intervention
2. During connection loss:

- Charge point stores meter values locally
- If during active transaction, charge point continues charging when safe
- Upon reconnection, charge point sends cached transaction data and meter values

6.2 Transaction Failures

1. If communication fails during transaction start:
   - Central system aborts transaction request
   - User is notified to retry
2. If communication fails during active transaction:
   - Charge point completes charging if already started
   - Upon reconnection, charge point sends final meter values
   - Transaction is closed with appropriate status

7. Security Considerations

1. All production WebSocket connections should use secure WebSockets (wss://)
2. Server-side authentication and authorization for all transactions
3. Data validation for all incoming messages
4. Rate limiting to prevent denial-of-service attacks

8. Data Persistence Requirements

1. Charge point must maintain:
   - Transaction data during active transactions
   - Meter values during connection loss
   - Configuration settings after power cycle
2. Central system must persist:
   - All transaction records
   - Historical meter values
   - Charging station status history
   - User account and payment information

This functional specification defines the minimum requirements for the initial implementation. Additional features like firmware updates, reservations, and advanced load management will be considered in future versions.