OCPP & M-PESA mobile payments

v1.0, October 2025

Table of Contents

Authors: Open Charge Alliance

1. Introduction

Digital payments are rapidly becoming the preferred method for everyday transactions around the world. In many countries, the rise of mobile-first payment systems has created opportunities to make public EV charging more accessible and convenient without the need for cards, proprietary apps, or RFID tokens.

For Charge Point Operators (CPOs), integrating these mobile payment systems is not just about offering another way to pay. The design of the integration has a direct impact on how seamless the driver experience will be, how efficiently charging sessions can be started and stopped, and how reliably the operator can reconcile payments.

Two of the most prominent examples of mobile-first payment ecosystems are:

- **UPI (Unified Payments Interface)** in India, which enables instant account-to-account payments using QR codes or mobile apps.
- **M-PESA** in Kenya and other African countries, which allows users to send and receive money through their mobile phones without requiring a bank account.

Although the underlying technologies differ, both systems present similar integration challenges for EV charging. Operators must decide how to connect payment flows to their Charging Station Management System (CSMS), how to link transactions to specific chargers, and how to ensure that prepaid limits or refunds are handled correctly.

This paper explores how M-PESA can be integrated into EV charging payments, looking at both simple and advanced models of integration. It also highlights how the Open Charge Point Protocol (OCPP) supports these scenarios through features such as remote transaction control and prepaid transaction limits.

OCA have published a similar paper on UPI payments, which can be found on the openchargealliance.org website.

2. Using M-PESA for EV charging payments

M-PESA has been a driving force in mobile financial services across Africa since its launch in Kenya in 2007. Operated by Safaricom and Vodacom, M-PESA enables users to deposit, withdraw, transfer money, and pay for goods and services using a mobile phone without a traditional bank account.

Its ability to move funds instantly, securely, and without requiring card infrastructure makes M-PESA an ideal payment channel for public EV charging, particularly in regions where mobile money penetration far exceeds card or bank account usage.

For Charge Point Operators (CPOs), integrating M-PESA requires understanding both the mobile money flow and how it can be connected to the backend charging management system.

2.1. Two integration models for M-PESA payments

Like UPI in India, M-PESA can be implemented for EV charging in two main ways: a **manual approach** and a **real-time API-based integration**.

2.1.1. Manual approach

In the manual model, each charging location or operator account has a fixed M-PESA PayBill or Till number. The driver sends a payment from their M-PESA account by selecting “Lipa na M-PESA” (pay by M-PESA) on their phone, entering the PayBill/Till number, and adding a reference (for example, the charger ID).

Once payment is made, the CPO is notified, either by checking their M-PESA merchant portal or by receiving an SMS confirmation from Safaricom. After confirming the payment, the operator or remote support team manually starts the charging session through the Charging Station Management System (CSMS). The operator will have to ensure that the transaction is stopped before the prepaid amount is exceeded. How this can be done, is discussed in section How OCPP supports prepaid payments.

This method is simple to set up and requires minimal technical integration. However, it relies heavily on manual verification, which can slow down the start of a charging session and is prone to reference errors or delays if multiple drivers are paying at the same time.

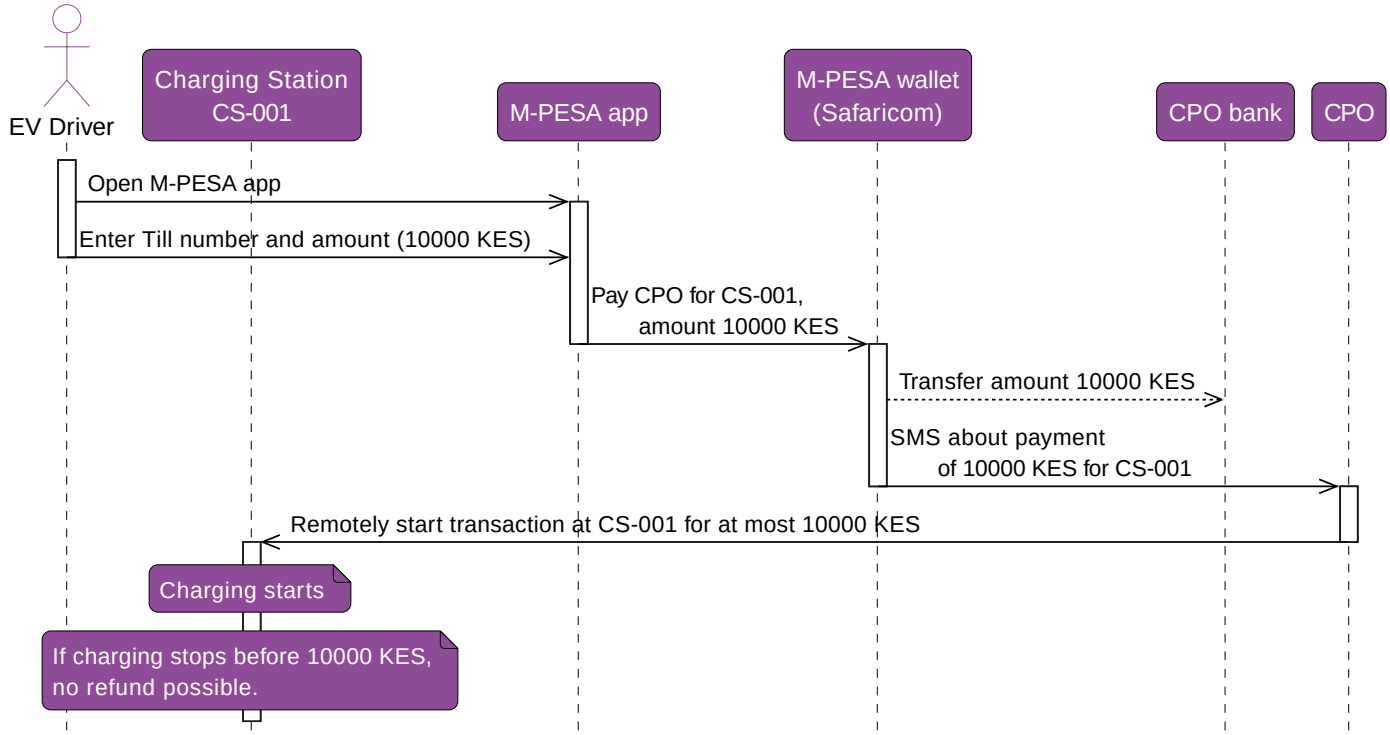

Figure 1. Steps for a manual approach

2.1.2. Real-time API-based integration

A more advanced approach uses the M-PESA Daraja API from Safaricom (or an equivalent API in other countries) to integrate the payment flow directly with the CPO's backend. When the driver selects a charger in the operator's mobile app or kiosk, the backend triggers an "STK Push" (Sim Tool Kit Push). This is an interactive prompt sent directly to the driver's phone via M-PESA. It allows a business or merchant to initiate a payment request directly to a customer's mobile phone. When triggered, the customer receives a pop-up notification on their phone's M-PESA menu, pre-filled with the payment details.

The driver authorizes the payment by entering their M-Pesa PIN through a USSD Push or SIM Toolkit interface. Once the payment is confirmed, M-Pesa sends a real-time notification (callback) with transaction details to the Charge Point Operator's (CPO) system. The CPO system can then automatically send an OCPP command (RemoteStartTransaction for OCPP 1.6 or RequestStartTransaction for OCPP 2.x) to the selected EV charger, initiating the charging session without manual intervention.

To prevent overcharging, the CPO system must stop the charging session before the prepaid amount is exceeded. For details on managing prepaid transactions, refer to How OCPP supports prepaid payments. If the prepaid amount is not fully used, the remaining balance may be refunded to the driver's M-Pesa wallet, depending on the operator's refund policy. Drivers should check with the operator for specific terms, as not all providers offer automatic refunds.

This real-time integration ties each payment to a specific charging session, eliminates manual verification delays, and supports automated refunds or adjustments if necessary. The trade-off is the need for API integration work and M-PESA merchant onboarding.

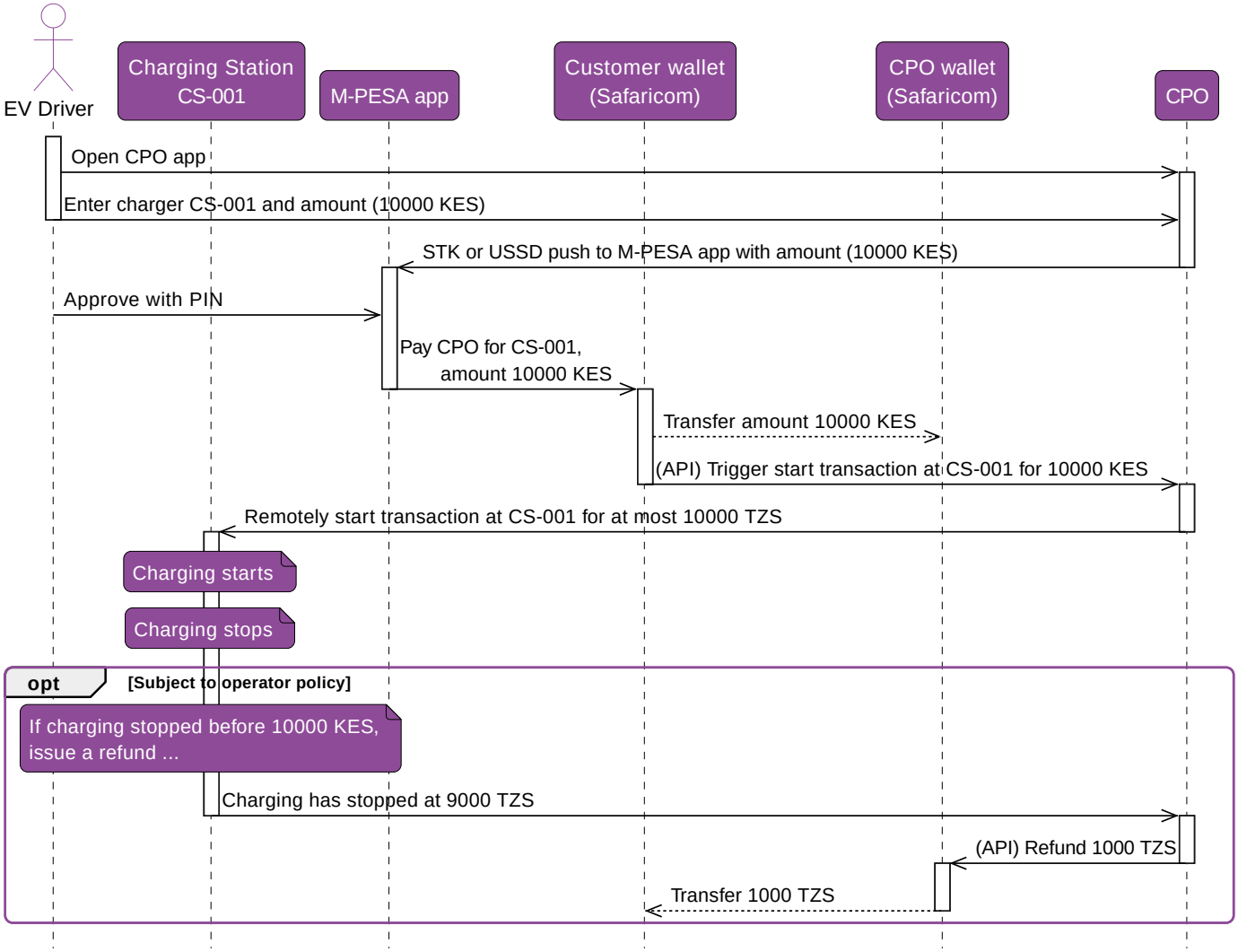

Figure 2. Steps for API-based approach

2.2. How the M-PESA payment server fits in

The M-PESA payment server is operated by the mobile network provider (e.g., Safaricom in Kenya, Vodacom in Tanzania) and acts as the bridge between the driver's mobile wallet and the CPO backend. In an API-linked setup, it accepts the payment, verifies success, and sends a callback (webhook) to the CPO's system. The CMS or CSMS then uses OCPP to instruct the charger to start or stop.

In the manual model, the payment server's role is limited to processing the transfer and sending an SMS or email notification to the merchant.

2.3. When to use which model

Manual reference-based payments work well in small networks or rural areas where internet connectivity is unreliable, as they do not depend on API calls and can be confirmed via SMS. Real-time API-linked payments are better suited for urban charging hubs and fast-charging stations, where customer experience and quick turnaround are priorities.

3. How OCPP supports prepaid payments

Ad hoc payments are payments that are not tied to a subscription model, i.e. there is no relation between the CPO or charge card provider and the customer. The most well-known method of ad hoc payment is direct payment via a credit or debit card using a payment terminal. M-PESA payments are both ad hoc (not tied to a subscription) and prepaid (an amount is paid prior to charging).

A ad hoc session via credit or debit card is in fact similar to a prepaid charging session, even though the exact amount is debited at the end of the session. When paying with a credit or debit card, the CPO requests a reservation on the customer's bank account for a certain amount of money before starting the charging session, to be sure that enough funds are available to pay for the cost. When the cost of a charging session is about to exceed this amount, the charging session will be ended, unless it is possible to increase the reservation amount.

In OCPP version 1.6 and 2.0.1 there was no explicit support for prepaid and ad hoc payments. Although many implementations exist that support payment terminals, these are all bespoke solutions. As of version 2.1 OCPP offers explicit support for both prepaid and ad hoc payments using integrated payment terminals, payment kiosks or dynamic QR codes.

The scenarios for prepaid mobile payments that are described in this paper, evolve around the following steps:

1. the user paying a prepaid amount to the operator,
2. the operator backend (CSMS) remotely starting a charging session for the user,
3. the CSMS stopping the charging session when the prepaid amount runs out.

This process relies on two OCPP concepts: remotely starting a transaction, and limiting the transaction to a maximum energy or cost.

3.1. Step 1: Paying a prepaid amount

When using M-PESA the user transfers money from their M-PESA wallet to the operator's wallet (Manual approach, Real-time API-based integration). This takes place outside the charger and is out-of-scope of OCPP.

3.2. Step 2: Remotely starting a transaction

The backend (CSMS) can remotely start a transaction on a charging station by issuing the `RemoteStartTransaction` command for a specific EVSE (connector) of the charging station. After acknowledging the message the charging station will start a transaction on the specified EVSE and send a `StartTransaction` message when the transaction has started.

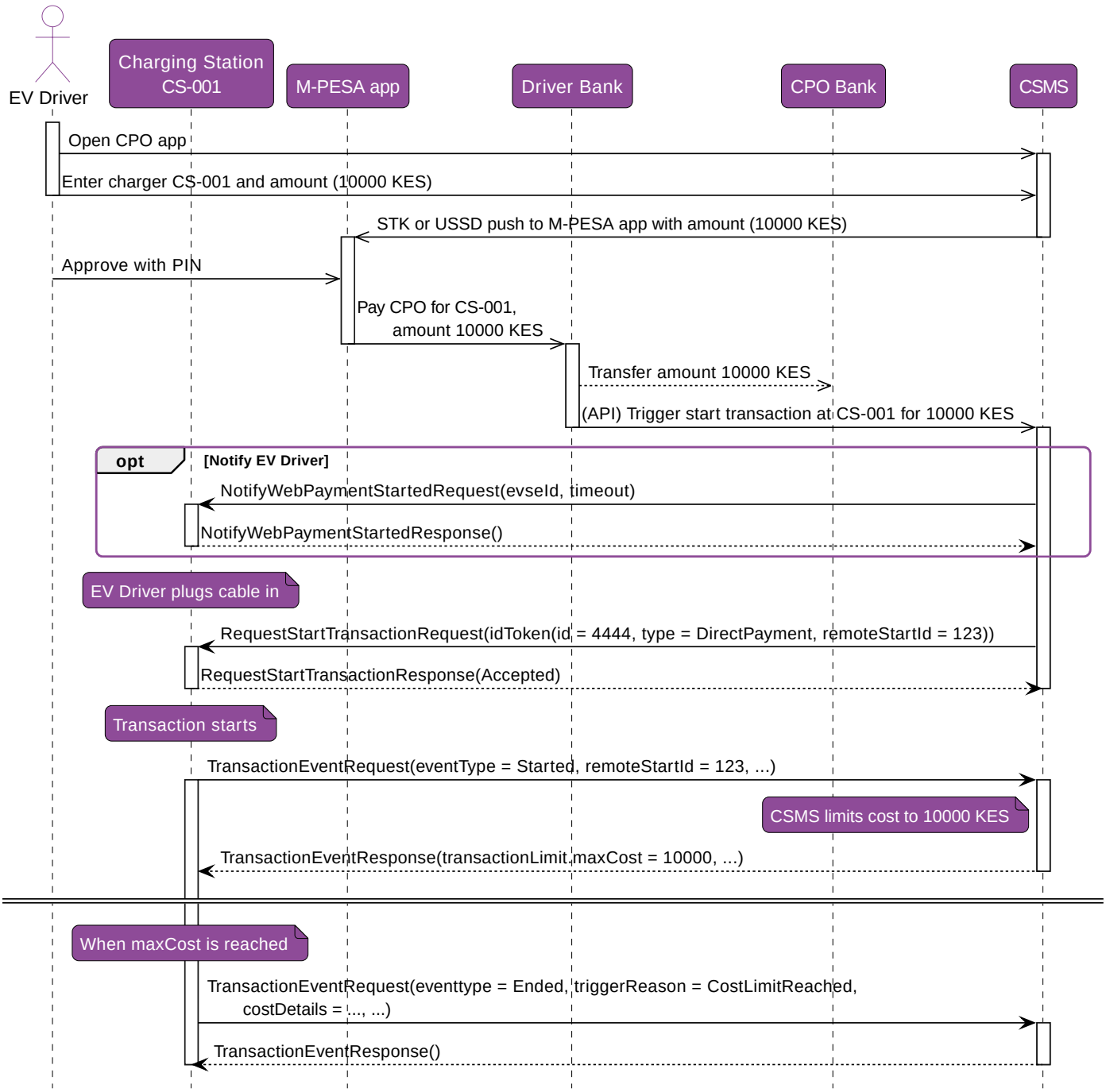

Figure 3. OCPP prepaid transactions

3.3. Step 3: Limiting transaction cost or energy

Prior to OCPP 2.1 the only way to stop a transaction when reaching a certain energy or cost limit, was by having CSMS continously check the meter values coming from the charging station, and stopping as soon as energy amount or calculated cost reached (or passed) the limit. This was never exact, because meter values are only sent at certain intervals.

Since OCPP 2.1 there is a new "transaction limit" feature. This adds a *transactionLimit* value to transactions by means of which the transaction can be limited in time, energy, cost and even state of charge of the vehicle. When CSMS receives a `StartTransaction` message, and it wishes limit the cost or energy of the transaction, then CSMS will add a *transactionLimit* to the response message that acknowledges the `StartTransaction` message.

In order to avoid exceeding the prepaid amount, the operator can choose to use a *transactionLimit* with either a cost (*maxCost*) or an energy (*maxEnergy*) limit. For a cost limit the charger must be able to locally calculate the transaction cost. Not all chargers are capable of that. It is therefore, in many cases more practical to convert the prepaid cost to the amount of energy that equates to this amount.

If the energy price is 50 KES per kWh, then a prepaid amount of 1000 KES equates to a maximum energy of 20 kWh. The CSMS will then set a *transactionLimit* with *maxEnergy* = 20 kWh. As soon as the consumed energy amount reaches 20 kWh, the charger will stop the energy transfer. It does not require monitoring by CSMS anymore.

OCPP 1.6 & 2.0.1

The older OCPP releases, 1.6 and 2.0.1, do not offer full support for ad hoc and prepaid payments. However, these releases do include a customization mechanism, which one can use to create a bespoke solution to support the mechanism described in this paper.

In OCPP 2.0.1 one can add a so-called *customData* extension to the TransactionEvent message to add the *transactionLimit* from OCPP 2.1 to the message. In OCPP 1.6 a possible solution might be to define a DataTransfer message to communicate the *transactionLimit* to the charger.

The OCA white paper "Customizing OCPP Implementations" (https://openchargealliance.org/ocpp-info-whitepapers/customizing-ocpp-implementations/) describes in detail how a customization can be added to your OCPP implementation.