

OMS — Tier 1 Multi-Currency: Implementation Plan

Service: OMS (merit-oms-service)

Status: Draft

Depends on:  Ecommerce Core — Tier 1 Multi-Currency: Implementation Plan

 Sales Channel / Storefront — Tier 1 Multi-Currency: Implementation Plan

OMS records the order as charged: amounts in `settlement_currency`, display currency for the customer, and the exchange rate versus settlement. It does not fetch markets from Core, does not convert, does not call tax, and does not initiate Payments.

Locked (from prior plans)

- Sales Channel / Storefront builds settlement lines + fee (from market) + tax on the settlement amount, then creates the OMS order and initiates Payments.
- FX runs on the client. The rate stored on the order is versus `settlement_currency`.
- `displayCurrencyCode` = market pricing_currency. `settlementCurrencyCode` = market settlement_currency (POINTS program currency, not hardcoded SAR).
- Core never sees the order. Payments are initiated by Sales Channel / Storefront with the OMS order id.

Locked (this plan)

- All order item money is stored in the market's `settlement_currency`. That is `order.settlementCurrencyCode`. `unitPrice`, `deliveryAmount`, `taxBreakdown` amounts, and derived line/order totals are settlement only.
- Mixed authored offer currencies are converted before create-order (client FX vs settlement). OMS does not persist offer-currency amounts on item money fields and does not convert.
- `displayCurrencyCode` and `displayExchangeRate` live on the order (pricing → settlement) for display. They do not change how items are stored.
- Required on `productReference` for order fulfilment: original `offer.currency` and the rate vs settlement. OMS math always uses the settlement item amounts.
- A refund converts at the rate stored on the order, never at the rate of the refund day

1. Role

OMS does	OMS does not
Persist display vs settlement currencies and the rate vs settlement	Convert prices
Persist line unitPrice, deliveryAmount, taxBreakdown in settlement	Call tax or Core for fee/country
Derive order totals from those settlement lines	Guess currency from sourceChannel
Fulfil in settlementCurrencyCode	Re-FX at fulfilment
Require payableExchangeRate when the payment leg currency ≠ settlement (POINTS → program currency)	Invent that rate

2. Create-order contract

Sales Channel / Storefront POST create-order must send:

Field	Value
displayCurrencyCode	market.pricing_currency
settlementCurrencyCode	market.settlement_currency
displayExchangeRate	Client FX display → settlement when those two differ; 1 when they are equal
items[].unitPrice	Line amount already in settlement (after client FX from offer.currency if needed)
items[].deliveryAmount	Market delivery_fee share in settlement (fee is fetched from market, not Core Identity)
items[].taxBreakdown	Tax already computed on settlement (product / delivery components as applicable)
metadata.marketId	Selected market
Authored offer snapshot	productReference (or item metadata): offerId, offer.currency, offer → settlement rate when those differ

Do not send display-currency amounts in unitPrice. Totals (subtotalAmount, taxAmount, totalAmount) stay computed by OMS from settlement lines, as today.

When display and settlement differ, displayExchangeRate is required (stop treating “both USD/SAR and rate omitted” as the only happy path).

3. Two rates (do not mix them)

Both are versus settlement. They are not the same number.

Rate	Meaning	Where it lives
Display → settlement	pricing_currency → settlement_currency	Existing orders.displayExchangeRate
Offer → settlement	Authored offer.currency → settlement_currency	Line productReference / metadata (client-supplied, stored as-is)
Payment → settlement	POINTS (or card) → settlement_currency	Existing paymentAllocations.payableExchangeRate (program rate for POINTS)

OMS does not recompute any of these. It stores them and uses allocation rate only to compute $\text{settlementAmount} = \text{payableAmount} \times \text{payableExchangeRate}$ (already the rule).

4. Payment allocations (Tier 1 POINTS)

- paymentAllocationType = POINTS.
- paymentCurrencyCode = points unit (POINTS). That already differs from settlement, so payableExchangeRate is required (existing constraint).
- payableAmount = points to burn.
- Implied settlement of the leg must equal order total in settlementCurrencyCode (subtotal – discount + tax + delivery).

OMS does not call Payments. After Sales Channel / Storefront initializes payment, inbound payment webhooks keep using the allocation's settlement amount.

5. Fulfilment

Merchandise fulfillment already prices MGC in order.settlementCurrencyCode (unit_price.currency, base + delivery components from unitPrice / deliveryAmount / taxBreakdown).

Keep that. Do not fall back to displayCurrencyCode for fulfillment money, and do not convert again. The charged settlement figure is what fulfillment must send.

Digital / giftcard lines follow the same settlement amounts; no shipping when fulfillment_profile is digital (Sales Channel / Storefront already omits shipping).

6. What not to add

- No markets table and no Core market fetch.
- No FX library.
- No tax service client.
- No requirement that sourceChannel implies a currency (merit-b2c is not "SAR").

- No market_id FK required this phase — metadata.marketId is enough for support/debugging. A dedicated column is optional later.

7. Validation to add

- If displayCurrencyCode != settlementCurrencyCode, displayExchangeRate must be present and > 0.
- Line amounts are interpreted only as settlement; reject a payload that claims a second money currency on unitPrice.
- If productReference.offer.currency is present and ≠ settlement, require the offer → settlement rate on that line.
- Sum of allocation settlement amounts must match totalAmount (settlement).
- displayExchangeRate precision: current column is DECIMAL(10, 2) — too coarse for FX. Widen (e.g. DECIMAL(18, 8)) so the client rate vs settlement is stored without rounding away.

8. Phases

Phase	Work	Done when
A — Contract	Accept split display/settlement from Sales Channel / Storefront; require displayExchangeRate when they differ; persist metadata.marketId + offer→settlement rate on lines	An IDR-display / SAR-settlement order creates without forcing the two currencies equal
B — Amounts	unitPrice, deliveryAmount, taxBreakdown treated as settlement; totals still derived in OMS	Tax and market fee are non-zero settlement figures, not hardcoded 0
C — Allocations	POINTS leg uses required payableExchangeRate; settlement of the leg matches order total	Payment webhook completes against settlement total
D — Fulfilment	Fulfil only in settlementCurrencyCode; no display fallback	MGC/giftcard money matches what was charged
E — Rate precision	Widen displayExchangeRate (and line offer→settlement storage) beyond 2 decimal places	Stored rate vs settlement matches the client rate

9. Open questions

1. Fulfilment currency (giftcard / merchandise) vs settlement — reconciliation.
Merit Service fulfillment still expects the base offer currency: merchandise
unit_price.currency + amounts in that catalog currency; giftcard denomination is the face
value in the product's currency (no currency field on the request). Sending settlement
amounts into fulfilment will miss-issue gift cards and mismatch MGC/catalog totals.

Options to lock:

- a. Use authored offer from productReference
- b. Change Merit/MGC giftcard + merchandise APIs to accept settlement currency
(downstream change, cleanest long-term).

Until this is locked, do not point fulfilment at settlementCurrencyCode alone

10. Handoffs

From	To OMS
Sales Channel / Storefront	Create-order payload in §2; then Payments initialize with OMS order id
Payments	Webhooks in settlement of the allocation; OMS does not convert
Ecommerce Core	Nothing on the checkout path. Inventory webhooks unchanged