**MYST-658: Analysis of Kafka for Service Area (SA) Changes**

**Overview** For BDC Returns Transfer Orders (TOs), the Transfer Order Management (TOM) system needs to validate whether the source and destination pair mapping is correct. This is achieved by subscribing to **Service Area (SA) events** published by the Network team.

For this, a new microservice in TOM side will be created (“toms-service-area-consumer-service”).

This document outlines:

1. How to use Kafka to take actions on a Transfer Order (TO).
2. The level at which SA changes impact a TO.
3. The appropriate course of action once an SA change is detected.

**Deliverables**

**1. Using Kafka for TO Actions**

* Subscribe to Kafka topic **service\_area\_snapshot**, which is published by the Service Area service.
* On detecting a change in the Service Area, a Kafka message will be published from SA side.
* TOM must process these messages to determine the validity of TOs and take corrective actions.

**2. Levels of Impact on TOs** SA changes can impact TOs at two levels:

* **TO Level** – If the origin location itself is no longer valid for servicing, the entire TO must be canceled.
* **Item Level** – If only specific items in a TO are no longer supported from the origin, then only those items are canceled, and the rest of the TO remains valid.

**3. Course of Action for SA Change Detection**

**When an SA change is detected through Kafka:**

1. Process the Kafka message payload.

2. Run a query on the TOM DB (using crud-commons) to fetch open and relevant TOs:

- TO Level Status = OPEN

- TO Order Line Status = CREATED

- TO Type != ‘CT’

- TO destination\_location = event.destination\_location

3. Compare Service Area Response

1. Case A – No itemOverrides[] in response:

- If TO’s origin location is not present in servicingNodes[] with ACTIVE status, cancel the entire TO.

- Mark with reason: INVALID SERVICE AREA.

- Publish TO\_CANCELLED event.

2. Case B – itemOverrides[] present:

- If TO origin exists in servicingNodes[] with ACTIVE Status → continue validation at item level.

For each order line:

(I) If line item does not exist in itemOverrides[].itemNumbers[] and origin location does not match itemOverrides[].servicingNode for all line items—

ALL order lines on a TO is VALID, then No Action required.

(II) If line item exists in itemOverrides[].itemNumbers[] and origin location matches itemOverrides[].servicingNode

a. If all Order Lines Invalid: Entire TO (ALL LINES) should be CANCELLED with Reason of INVALID SERVICE AREA and publish TO\_CANCELLED event

b. If only some of the orderlines are invalid:

- Cancel only the impacted orderline(s).

3. Case C – No itemOverrides[] in response:

- If TO origin exists in servicingNodes[] with ACTIVE Status, TO is valid.

- No action required.

4. Case D – itemOverrides[] present:

- If TO’s origin location is not present in servicingNodes[] at all or TO’s origin is present but status is not ACTIVE, cancel the entire TO.

- Mark with reason: INVALID SERVICE AREA.

- Publish TO\_CANCELLED event.

\*\*\*

Exceptions/Errors -

1. SaveAndPublishException

Note – We will be using Metric Service to send real time data to Grafana dashboard and will be increasing exception count if there is any exception.

\*\*\*

Pros & Cons of Using Kafka

\*\*Pros:\*\*

- Real-time event processing for immediate validation and correction.

- Ensures TO accuracy by reacting to SA changes dynamically.

\*\*Cons:\*\*

- Multiple queries – System load may increase due to repeated DB lookups for TO validation.

\*\*\*

Reason to choose Kafka:

1. In case of REST API, there will be multiple calls involved to SA API.

1. It will be the real time processing.
2. SA will send the full data for the destination\_location if there are any changes in that particular destination\_location. So, no need to save these data in TOM DB.



