

## Transaction Event

- Start Transaction
- Update Transaction
- Transaction Failed
- Transaction Completed

The **Start Transaction** Event will create a new transaction with a given id. If the id already exists, the event will be ignored.

The **Update Transaction** Event will update an existing Transaction with a given Worklist. If the Worklist has no pending items after the update, a Transaction completed event is generated.

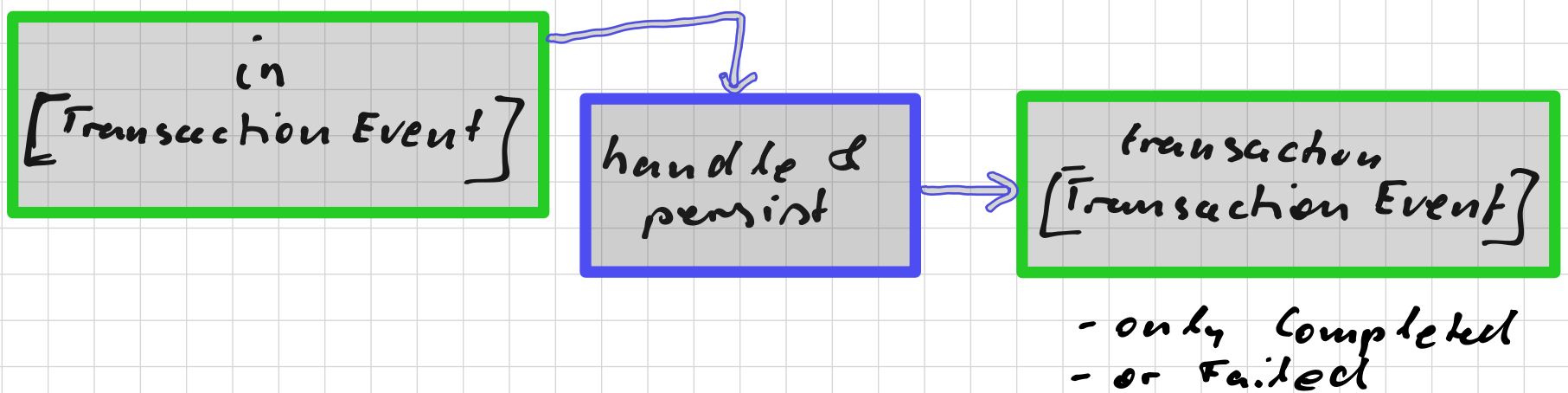
If the Worklist has any failed items after the update, a **Transaction Failed** event is generated.

If the update references a non-existent transaction, the event will be ignored.

A **Transaction Failed** Event on the inbound stream fails the given Transaction regardless of its worklist state and passes the Transaction Failed downstream.

A **Transaction Completed** on the inbound stream completes the given Transaction regardless of its worklist state and passes the **Transaction Completed** downstream.

The Transaction state is persisted, so that it survives a container restart.




Transactions do not timeout by themselves. However, if the underlying message has a TTL configured, the worklist may timeout and that will also fail the transaction.

## Application log:

The application log will log all Transaction Started, Completed and Failed Events.

## CBE:

The CBE support will generate a CBE xml message for each Started, Completed or Failed Transaction event.



All Transaction processing happen asynchronous. Failures to generate CBE Events or application logs will not cause the business transaction to fail.

⇒ Any Processing that should cause a failure of the business transaction must be included in the dispatcher Flow.

## Transaction manager

- should create a new transaction
- should ignore Transaction Started for existing Transactions
- should update correctly
- should generate "completed" "failed" or "started correctly"
- should survive a container restart