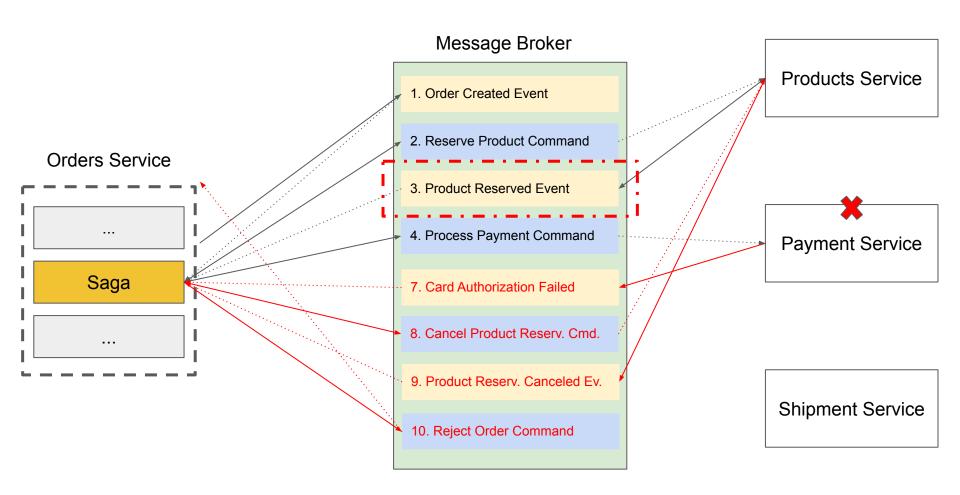
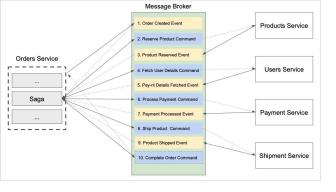
# SAGA Design Pattern

# Orchestration-Based

Saga

# Message Broker 1. Order Created Event **Products Service** 2. Reserve Product Command **Orders Service** 3. Product Reserved Event 4. Process Payment Command Payment Service 5. Payment Processed Event Saga 6. Create Shipment Ticket 7. Shipment Ticket Created Event 8. Approve Order Command **Shipment Service** 9. Order Approved Event





## Orchestration-based Saga

### Benefits:

- Good for complex workflows involving many Microservices,
- Easier to control the flow of activities,
- Doesn't introduce cyclic dependencies, because the orchestrator unilaterally depends on the saga participants,
- Saga participants don't need to know about commands for other participants. Clear separation of concerns simplifies business logic.

### Drawbacks:

- Additional design complexity requires an implementation of a coordination logic.
- There's an additional point of failure, because the orchestrator manages the complete workflow.