# Summary

Traditional application transaction management and high throughput use cases require different design patterns when utilizing micro services architecture, particularly in the AWS cloud which offers many useful services to accommodate granular processing. This POC covers the following use cases

## Use Case 1: Multi-step Order fulfillment & Processing

Update multiple services to handle an operation traditionally bound by an atomic operation utilizing automatic (likely XA) transactions, manual workflow or other technique.

### POC Scenario:

The creation and fulfillment of a new order requires the following services be updated:

* OrderCapture – capture initial order request details and allocate orderId.
* OrderFulfillment -
* InventoryService – verify & deduct inventory – remove items if out of stock.
* OrderCompletion – ship and confirm to user.

### Summary of Design

Synchronous to Dynamo DB

Provides immediate feedback to user that the order with order id xyz has been lodged. Verbage back to the user is that they will receive final confirmation one way or other when order has been processed.

API Gateway ->Lambda->Controller->DynamoDB

From that point on

DynamoDB->SQS->Dynamo DB->additional workflow steps

Use SWF?

LOTS of other design patterns here. In real world, each Controller would likely be a separate microservice but utilizing this for sake of simplicity.

In event of error at any point, all previous steps are rolled back and user is notified.

## User Case 2: Large Data Processing Fanning

There are many ways to handle processing of large amounts of data, and again AWS

POC Scenario

Processing large product updates from suppliers can .

Note that in this simple POC we assume no dependencies on the entries within the Item updates.

Design

Fanning approach from within Lambda.

* Split the data records into buckets.
* Allocate buckets to child Lambda functions & Triage the records by first identifying which of them have actually changed? Require a timestamp of the latest state of the record?
* Then only update the items that are required

Currently 5 minute cap on Lambda to take into account.