Domain Driven Design
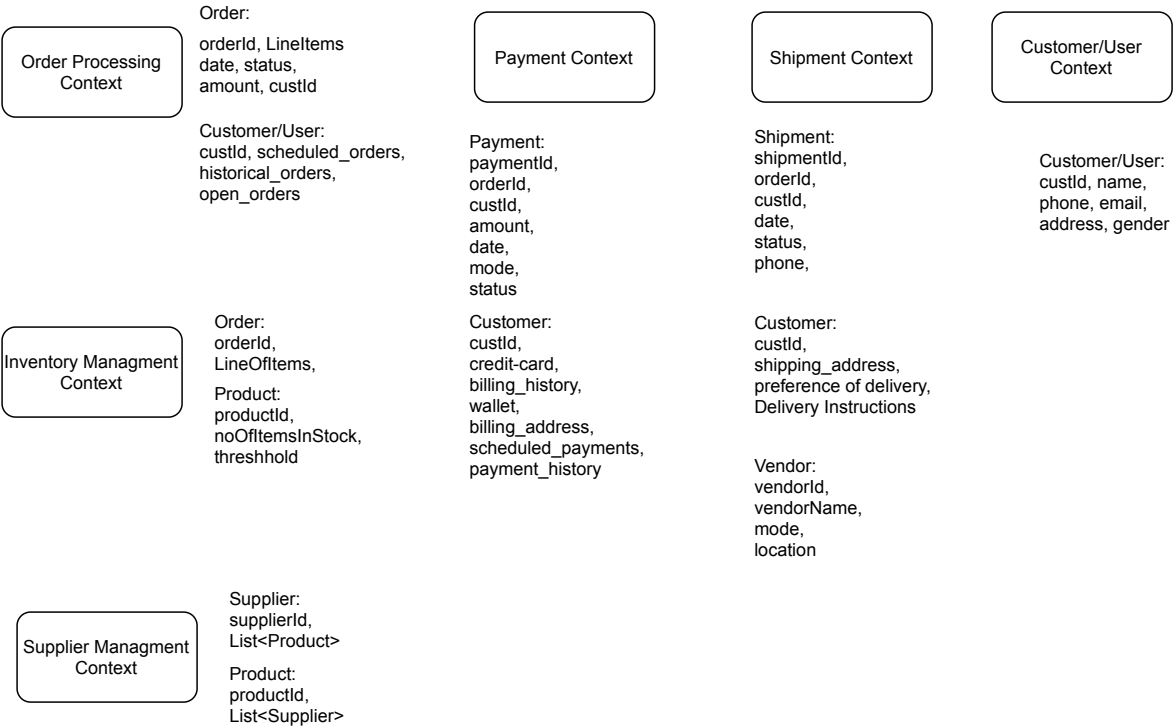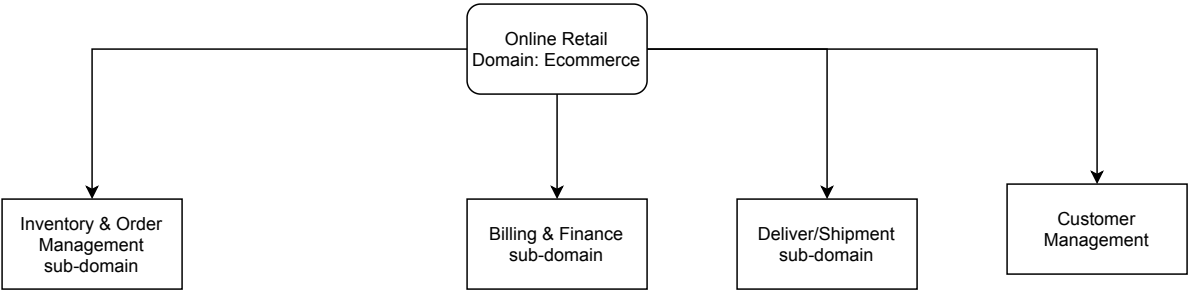
1. Domain/Sub-Domain
2. Context/Bounded Context
3. Model
4. Ubiquitous Language

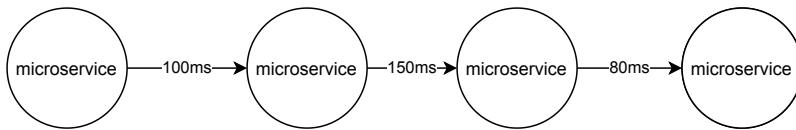Divide & Conquer
Technical Team + Business Experts

```
                        ┌────────────────┐
                        │ Online Retail  │
                        │Domain: Ecommerce│
                        └────────────────┘
```

Inventory & Order Management sub-domain	Billing & Finance sub-domain	Deliver/Shipment sub-domain	Customer Management

Order Processing Context

Order:
orderId, LineItems
date, status,
amount, custId

Customer/User:
custId, scheduled_orders,
historical_orders,
open_orders

Payment Context

Payment:
paymentId,
orderId,
custId,
amount,
date,
mode,
status

Shipment Context

Shipment:
shipmentId,
orderId,
custId,
date,
status,
phone,

Customer/User Context

Customer/User:
custId, name,
phone, email,
address, gender

Inventory Managment Context

Order:
orderId,
LineOfItems,

Product:
productId,
noOfItemsInStock,
threshhold

Customer:
custId,
credit-card,
billing_history,
wallet,
billing_address,
scheduled_payments,
payment_history

Customer:
custId,
shipping_address,
preference of delivery,
Delivery Instructions

Vendor:
vendorId,
vendorName,
mode,
location

Supplier Managment Context

Supplier:
supplierId,
List<Product>

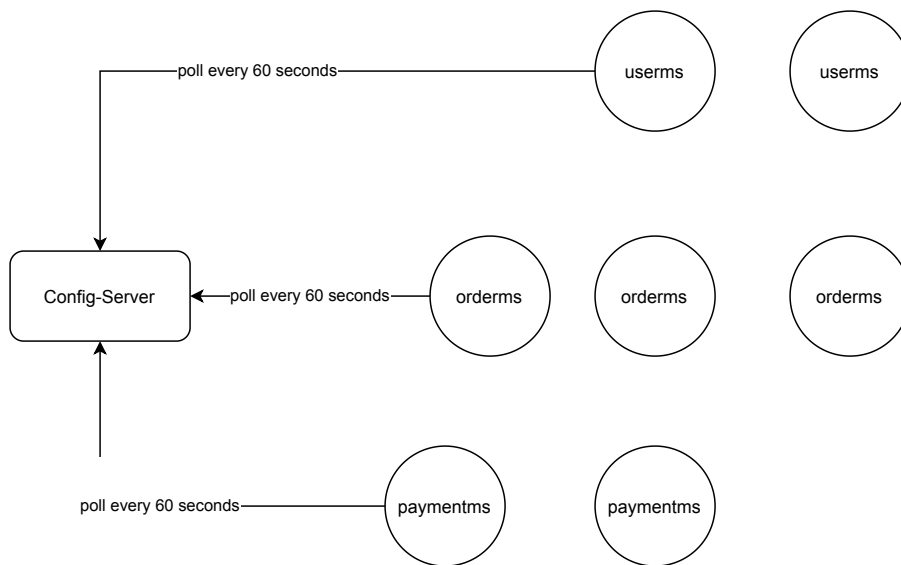Product:
productId,
List<Supplier>

Granularity of Microservcies

1. DDD
2. Thumb Rule
3. Benchmark

2. Thumb Rule: Every request should not span across more than 3-5 microservice calls in chain

```
microservice  ─100ms→  microservice  ─150ms→  microservice  ─80ms→  microservice
```

3. Benchmark

GET: /orders, /users, /payments, /shipments
POST: /orders, /users, /payments, /shipments
PUT: /orders, /users, /payments, /shipments
DELETE: /orders, /users, /payments, /shipments

Configuration Management:

1. Includes properties as part of code
2. Externalize the properties
3. Config-Server (Poll)
4. Config-Server (Message Bus)

Configuration Management: Spring Cloud Bus + RabbitMQ   (Spring Cloud Config Project)

Config-Server

Config-Server

Config-Server

/refresh

get latest config

userms
application.properties
application-
dev.properties

cache:TTL: 20 minutes

config-client

Webhook (Notification)

refresh

Message Bus

RabbitMQ

orderms

orderms

paymentms

paymentms

config-repo
(hello_world)
Git

properties:
common.propert
userms.properties
orderms.properties
inventoryms.properties
shipmentms.properties
userms-dev.properties
config-client.properties

userms.properties:
cache:TTL: 20 minutes

make changes to properties
Push

DevOps

Kafka

1. Apache Kafka
2. Confluent Kafka

1. Broker
2. Topic
3. Record(Message)
4. Producer
5. Consumer
6. Consumer Group
7. Partition (Scalable)
8. Replication Factor (High Availability)
9. Offset
10. Zookeeper

Zookeeper
(Kafka Cluster Manager)
(soon to be removed)

orderms
(Producer)

paymentms

Sephin-Order

Neeraj-order

Broker(Kafka Instance)

Broker(Kafka Instance)

1000 messages per second

Topic-1
Two Partitions
(P0 and P1)

Topic-2
One Partitions
(P0)

Topic-1
Two Partitions
(P0 and P1)

Topic-3
One Partitions
(P0, P1, P2, P3)
every partition
gets 250
messags/second

Poll

Poll

Poll

paymentms-1
(P0, P1)

paymentms-2
(P1)

shipmentms
(Topic-1)
(P0 & P1)

ConsumerGroup-Id: 123

clientms
(P0)

clientms
(P2)

clientms
(P3)

clientms
(P1)

clientms

clientms

ConsumerGroup-Id: 456

Sagar —Good→ myTopic (P0 & P1) —Good→ Satyasri

—Good→ Satyasri

Sagar —Excellent→ myTopic (P0 & P1) —Excellent→ Satyasri

Satyasri