

# Domain Driven Design

# Agenda

Introduction to Domain-Driven Design

Ubiquitous Language

Bounded Context

Entities, Value Objects, and Aggregates

Domain/Integration Events

Strategic Design

Implementing DDD

Benefits of DDD

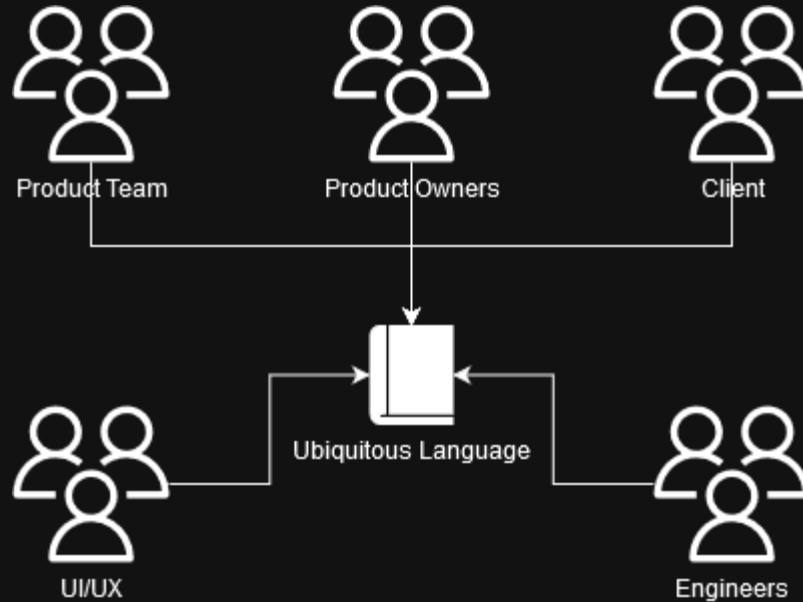
Q&A

# Introduction to Domain-Driven Design

- What is DDD?
- Key concepts
  - Ubiquitous Language
  - Bounded Context
  - Entities, Value Objects, and Aggregates
  - Domain Events

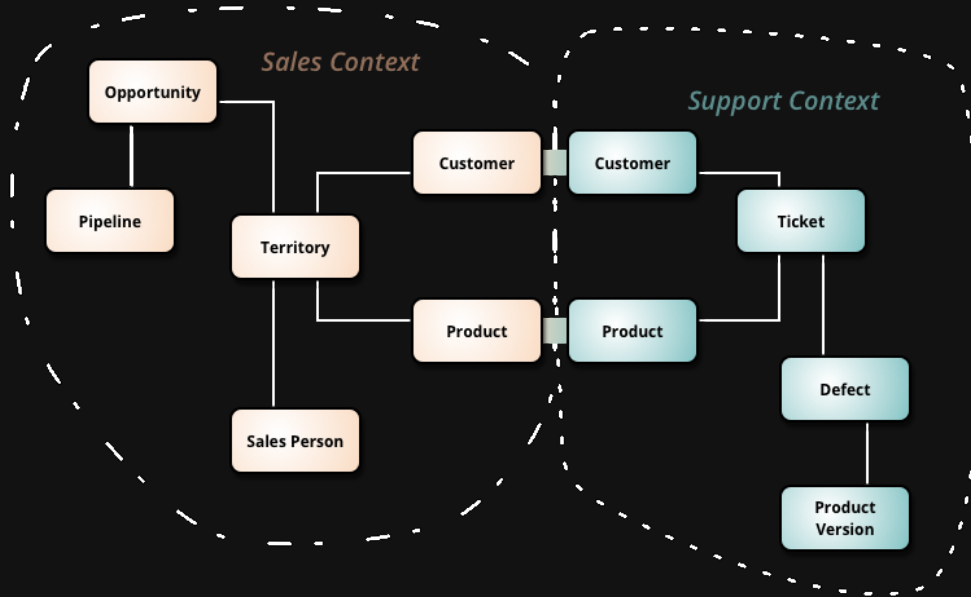
# Ubiquitous Language

A common, shared language between developers and domain experts



# Bounded Context

A boundary within which a particular model and set of terms apply.



# Entities, Value Objects, and Aggregates

## Entities

Objects that have a distinct identity

e.g. Product



Color: Red

Weight: 0.5kg

Sr #: 12345



Color: Red

Weight: 0.5kg

Sr #: 95316

## Value Objects

Objects without a distinct identity; defined by their attributes

e.g. Price



Currency: PKR

Value: 30



Currency: PKR

Value: 30

## Aggregates

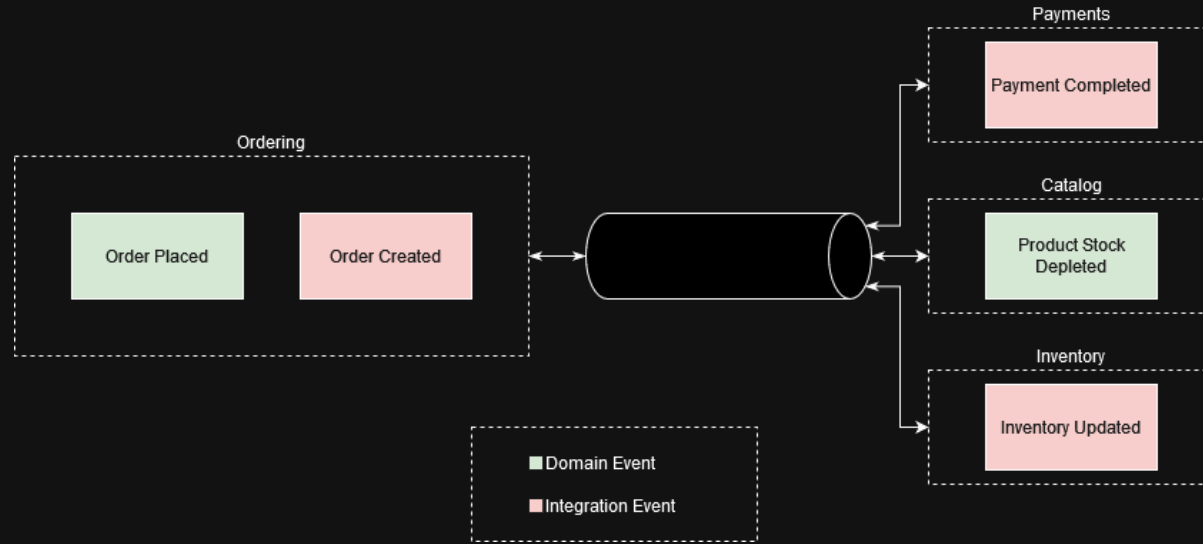
Clusters of related entities and value objects

e.g. Order

Order
+DateTime OrderDate
+Address Address
+int BuyerId
+OrderStatus OrderStatus
+string Description
+bool IsDraft
+List OrderItems
+int PaymentMethodId

# Domain/Integration Events

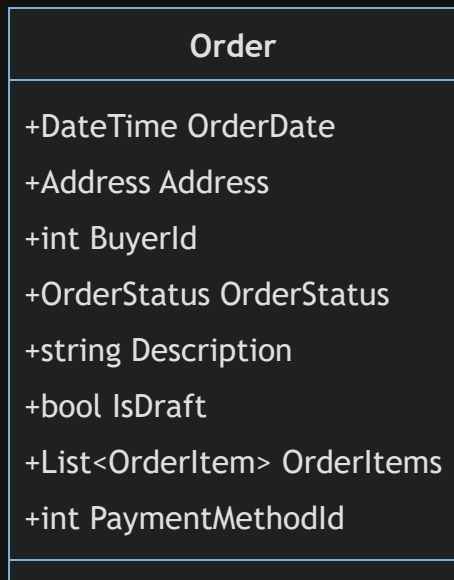
An event that captures a state change **WITHIN/BETWEEN** the bounded context(s)



# Strategic Design

The high-level design decisions that shape the architecture and organization of the software

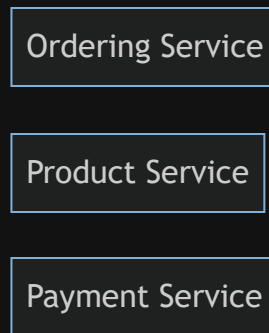
## Aggregates



## Repositories

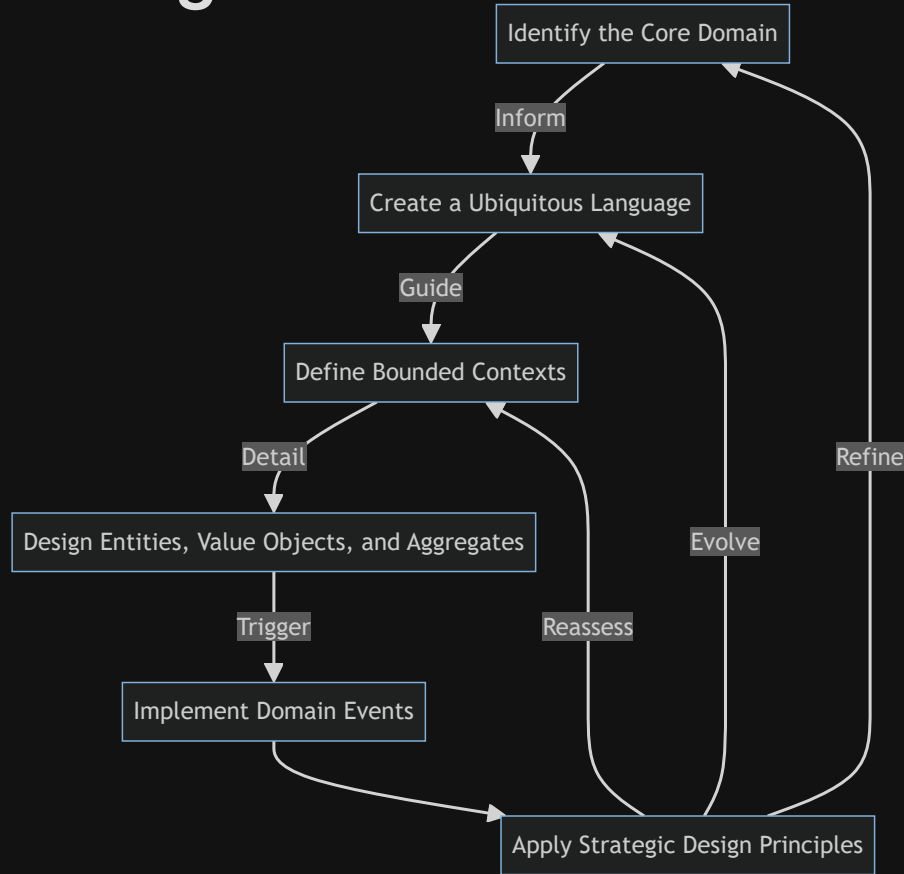


## Services





# Implementing DDD



# Benefits of DDD

Alignment with Business Needs

Clear and Shared Understanding

Modularization

Maintainability

Flexibility and Adaptability

Enhanced Testing

Q&A