

DDD



# DDD



# Deadline Driven Design



# Domain Driven Design (DDD)



DDD Series



Domain-Driven

DESIGN

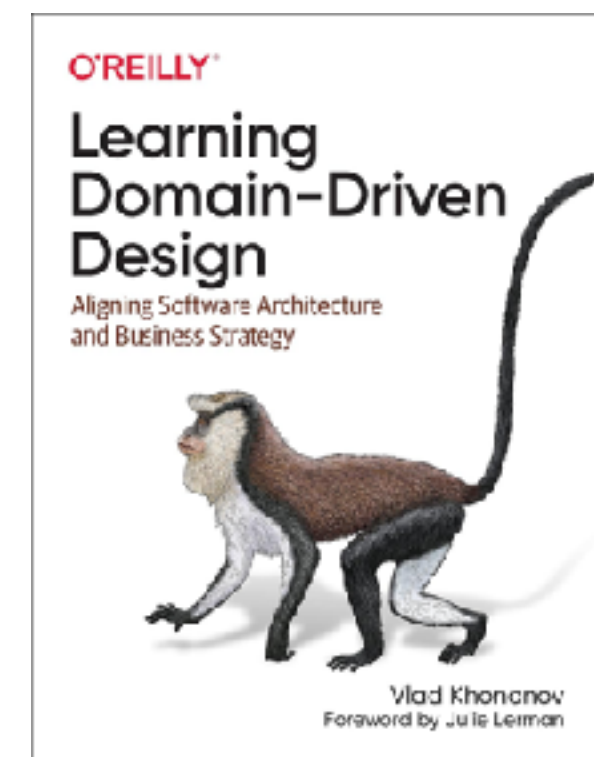
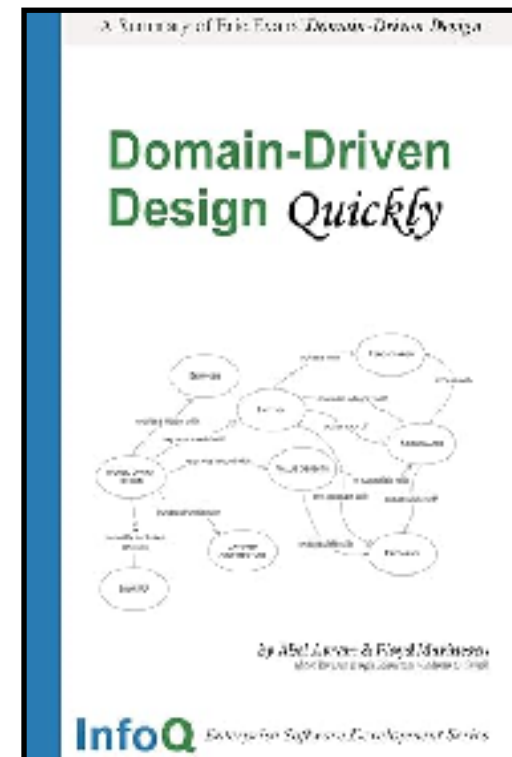
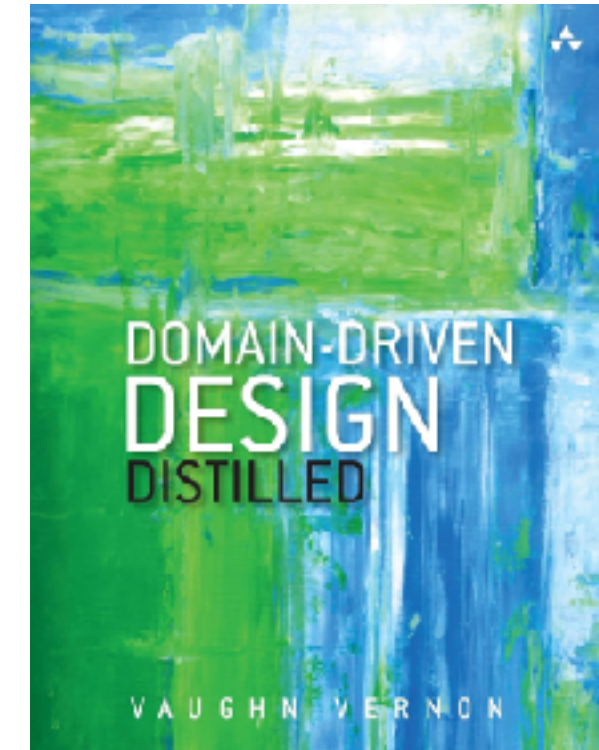
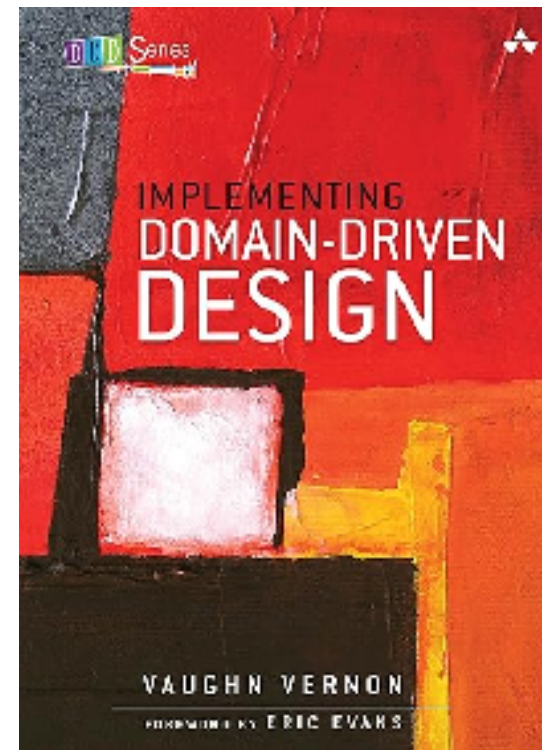
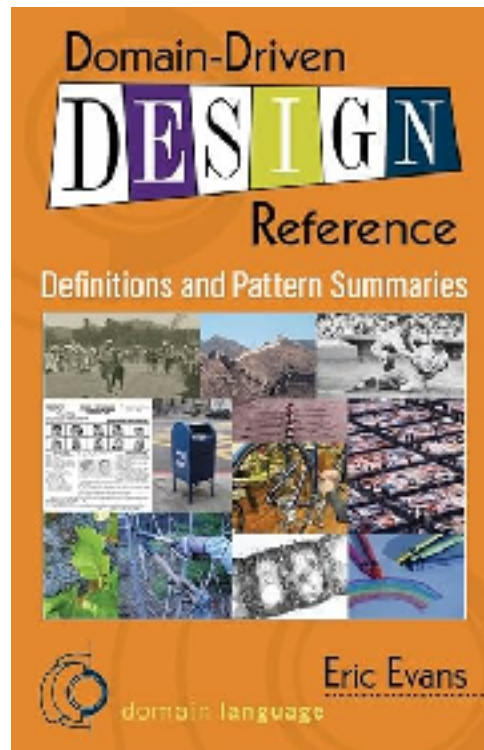
Tackling Complexity in the Heart of Software



Eric Evans

Foreword by Martin Fowler







# DDD ?

Software development methodology

Focus in model complex system around business domain

Reshape legacy mainframe data input biz-friendly

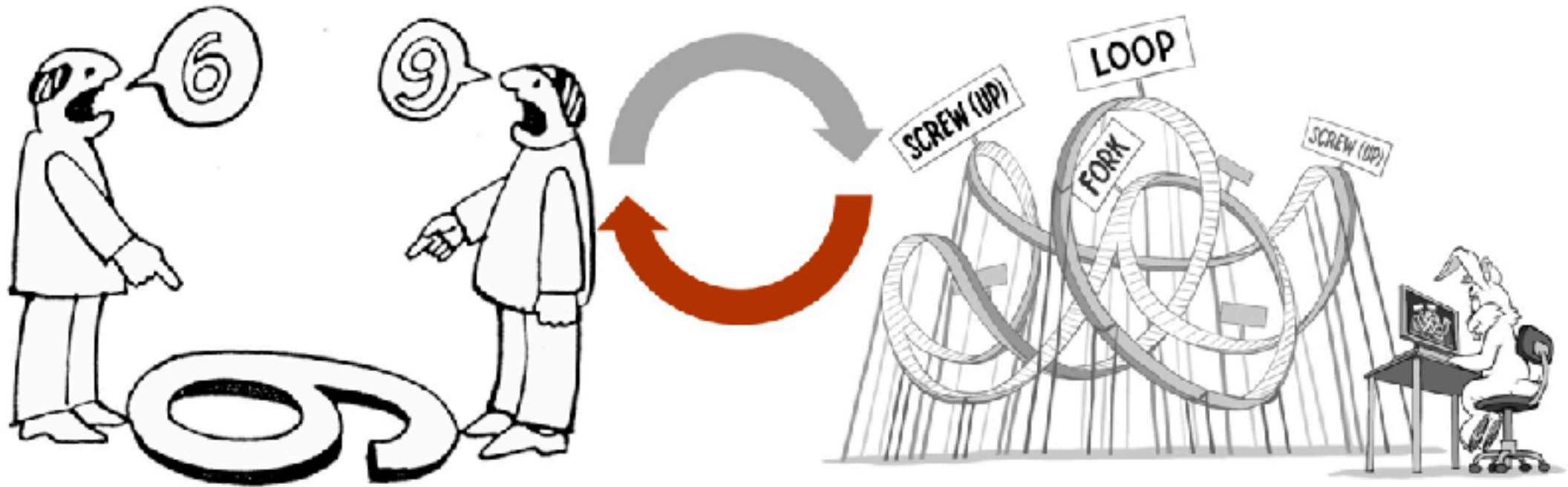
Easier to understand and evolve

Create common language between tech and biz



# Business

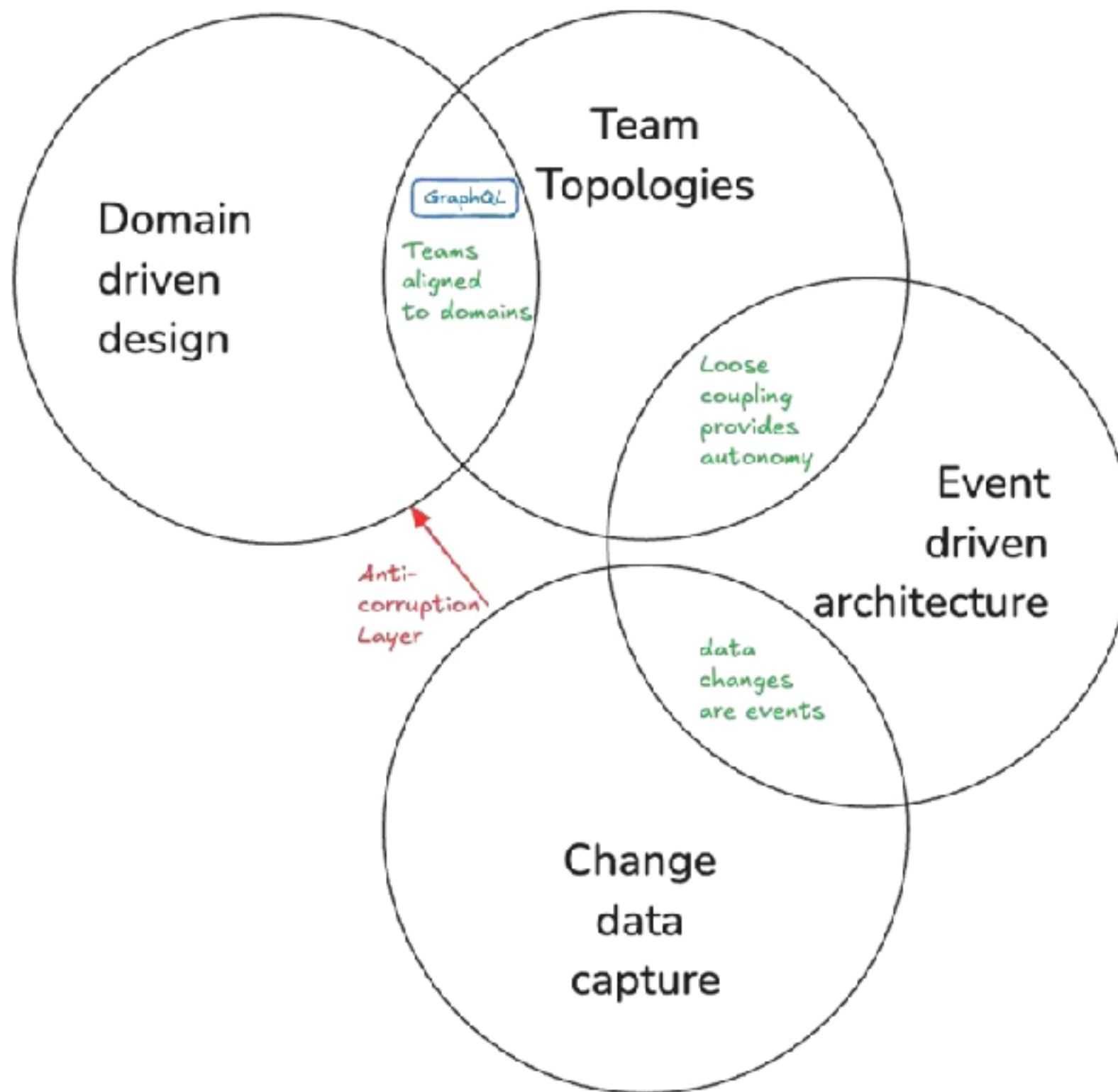
# Technical



Communication, Synchronous,  
discuss goal, way to success !!!







<https://www.infoq.com/minibooks/architectures-2025/>



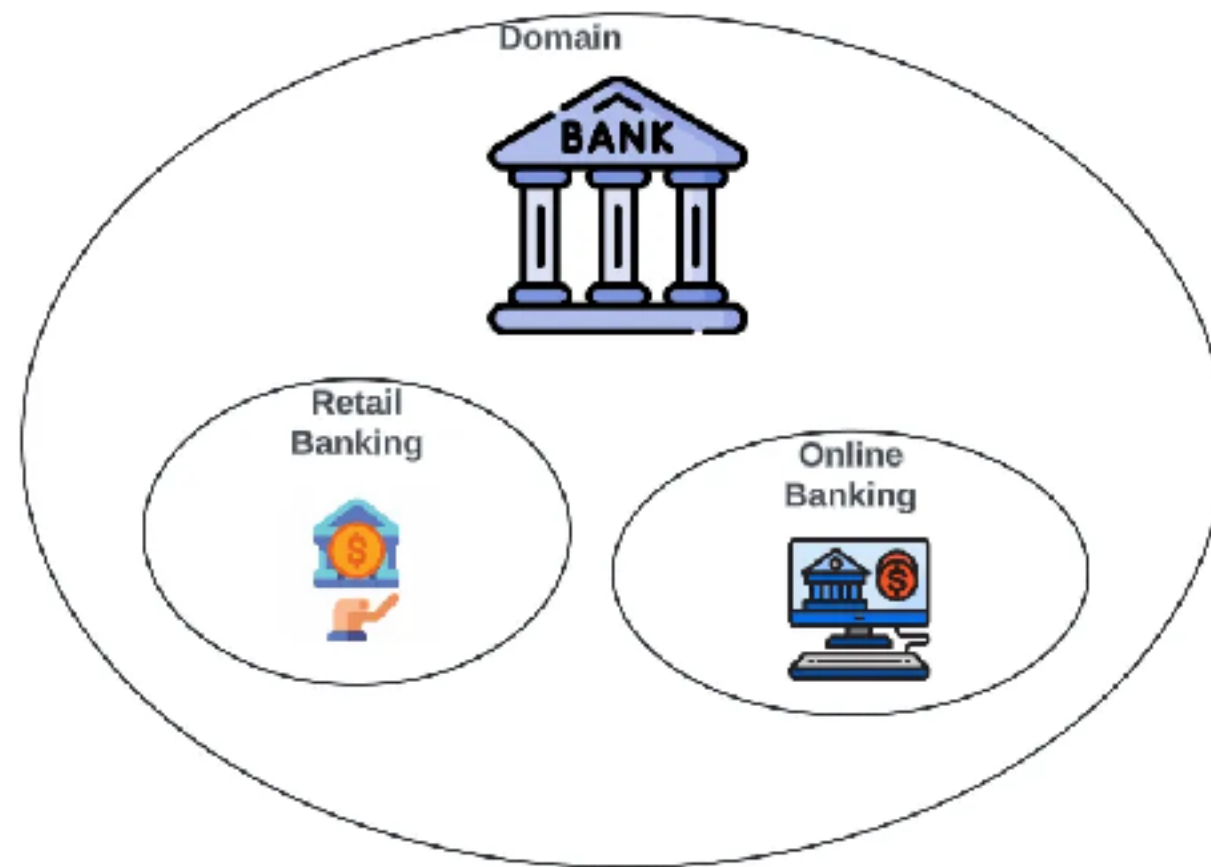
# Domain ?



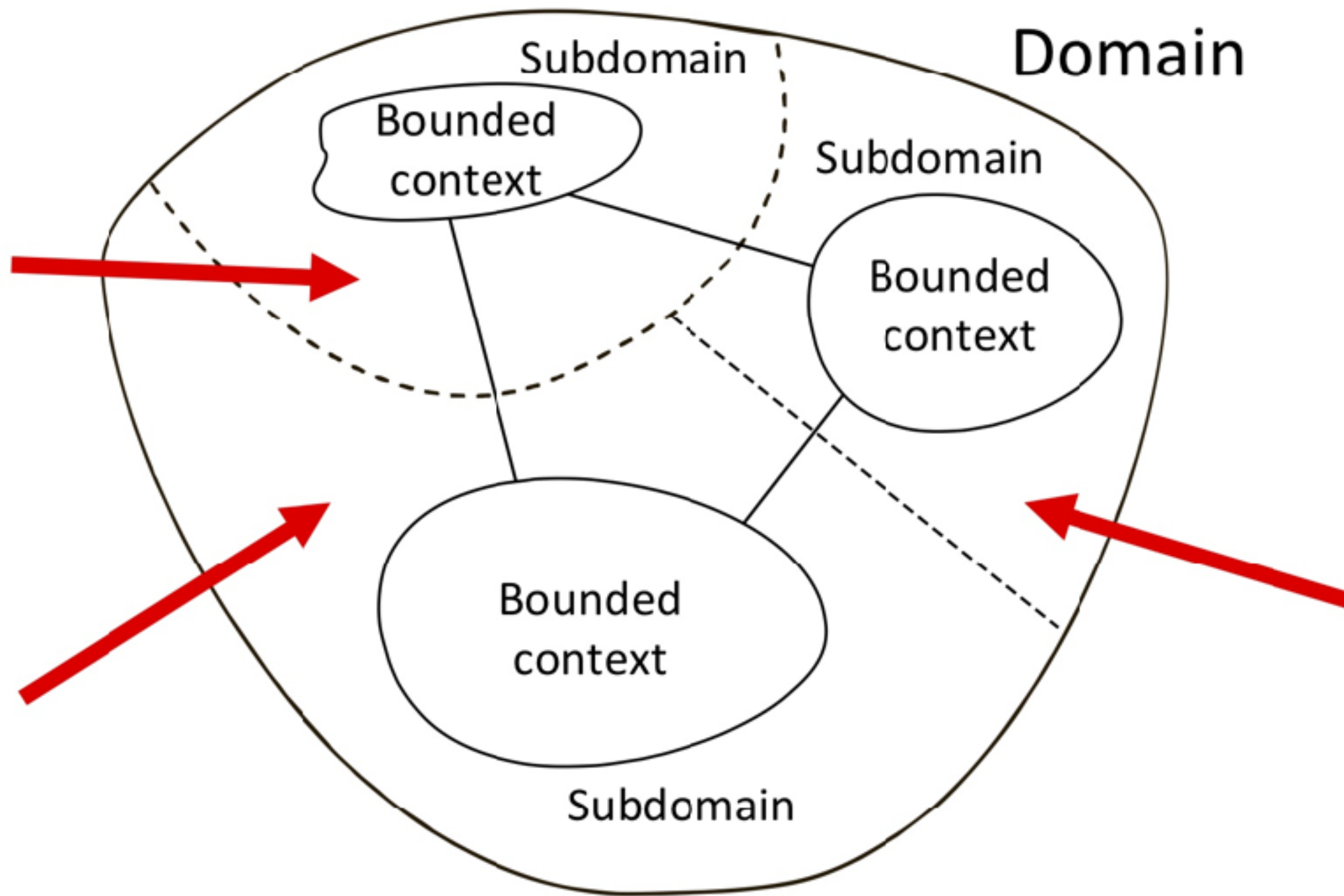
# Domain ?

The area of knowledge or activity around which the application is centered

## Bank Domain



# Domain ?



37



# Subdomain ?

Specific areas within the domain (**part of business**)  
Core, Supporting and Generic (important and role)

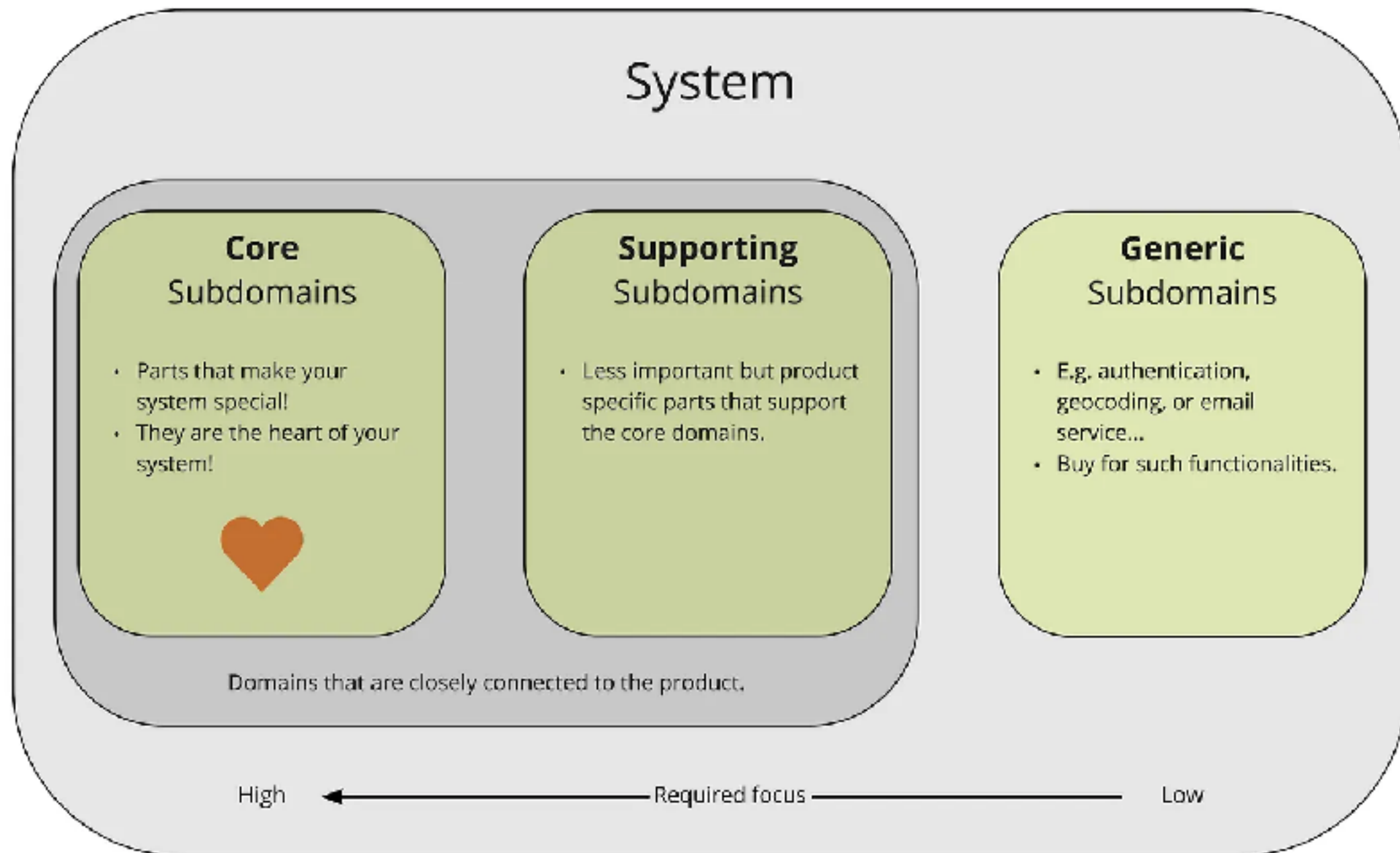
Core

Supporting

Generic



# Subdomain ?



<https://medium.com/claimsforce/what-is-domain-driven-design-2686c935fc2f>





# Core Subdomain

Heart of business (competitive advantage)

Focus on delivering **business value** that matter the most for the **customer**



# Supporting Subdomain

Crucial for the car subdomain to operate effectively

But don't provide the level of differentiation and not  
unique to the company



# Generic Subdomain

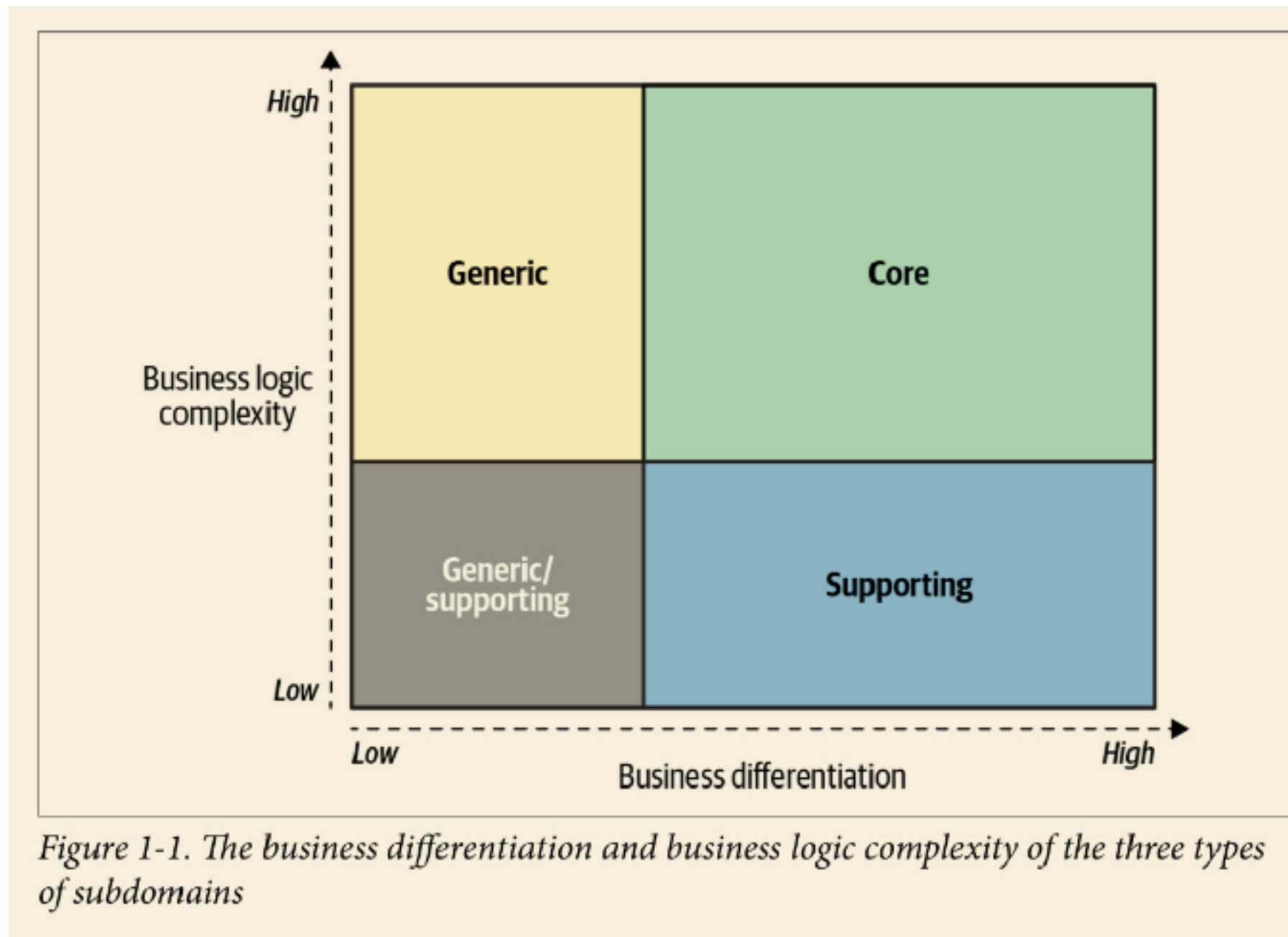
Common functionality

Not specific to the business domain

Often be addressed with readily available solutions



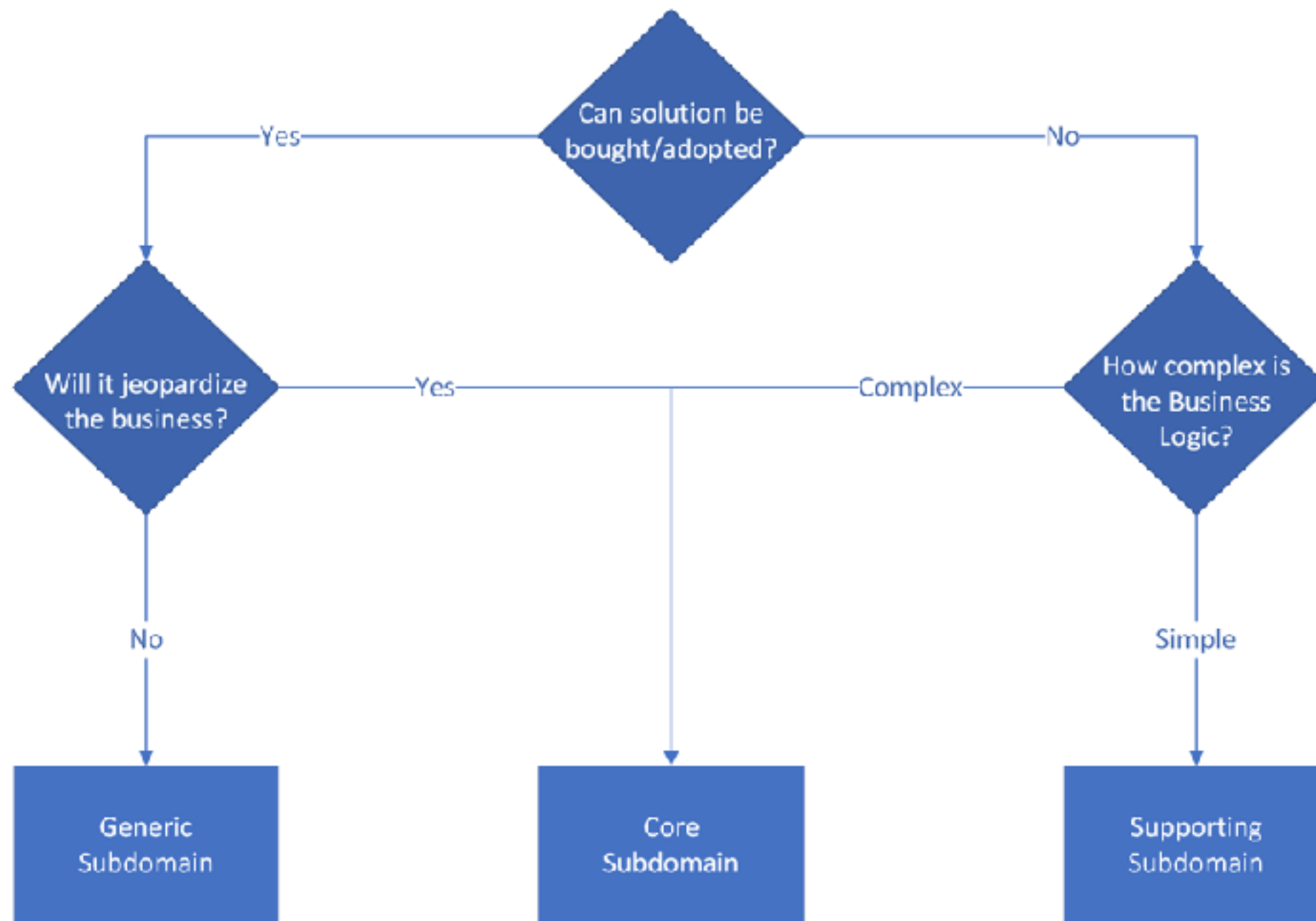
# Subdomain



<https://datahonor.com/se/ddd/learning-ddd/#chap1-analyzing-business-domains>



# Categorize Subdomain ?



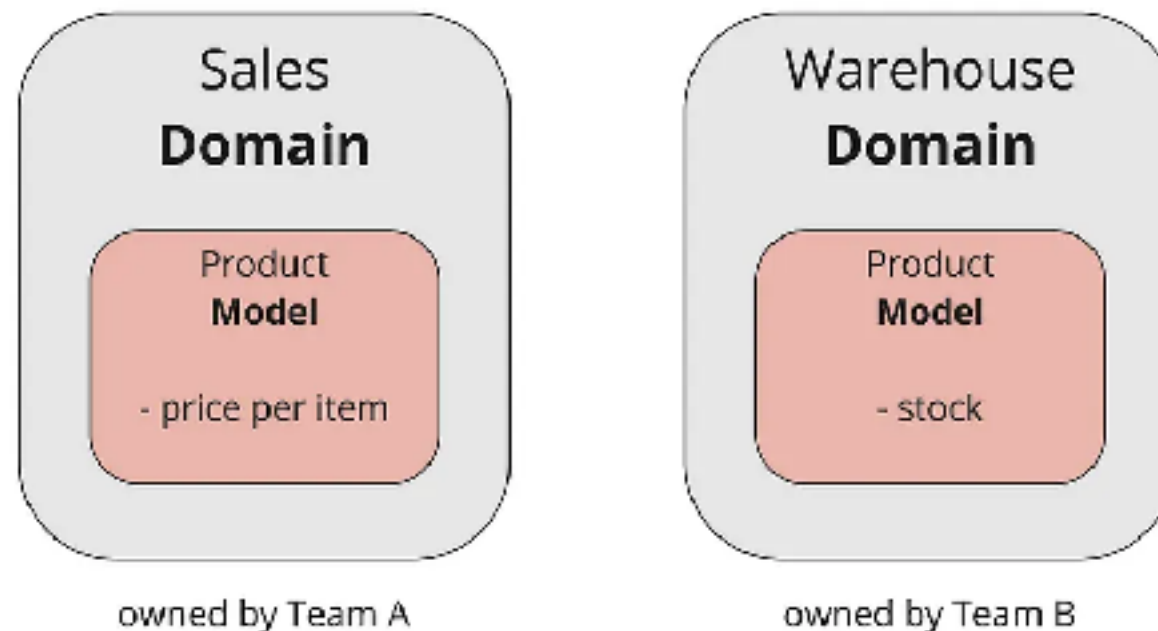
<https://vladikk.com/2018/01/26/revisiting-the-basics-of-ddd/>



# Boundary Context

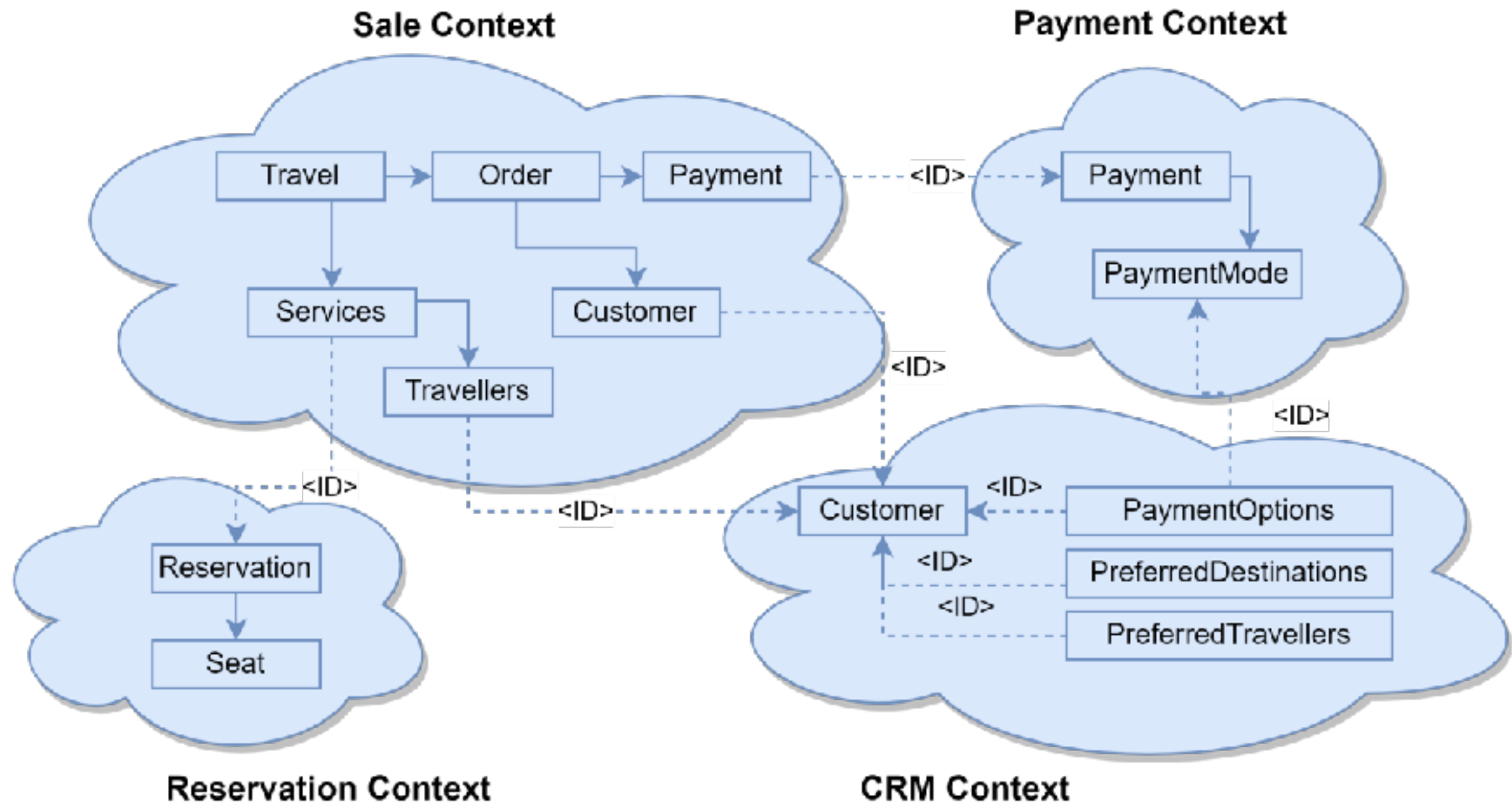
Enable the whole team that working in the same domain to build a shared language

Short and precise communication with fewer misunderstanding





# Boundary Context



# How to commutation between domains ?



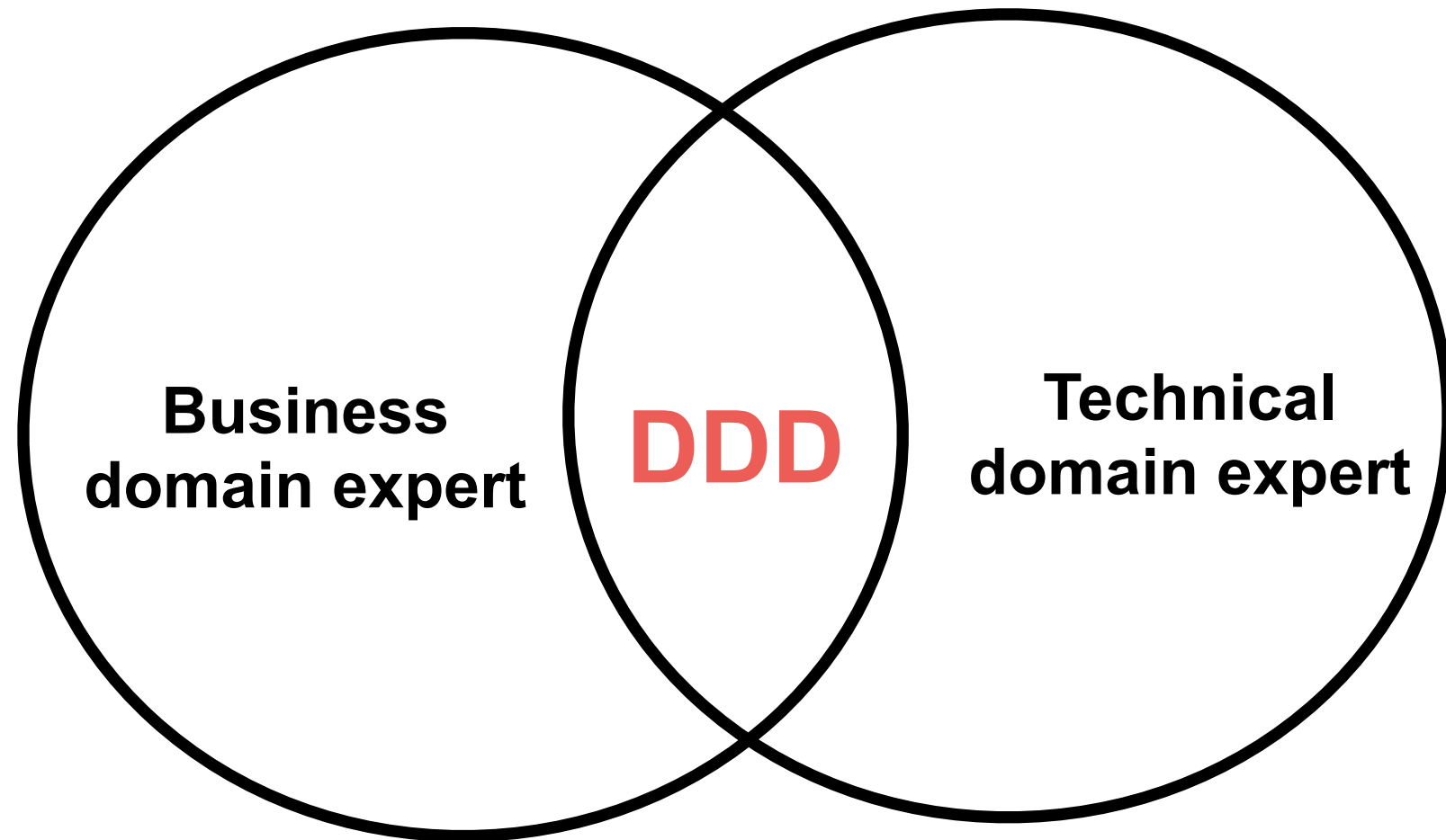
# Avoid domain dependencies and Data sharing !!



# Domain Driven Design **is not** About technologies



# DDD



# Problem ?

**Business  
domain expert**

**Business  
language**

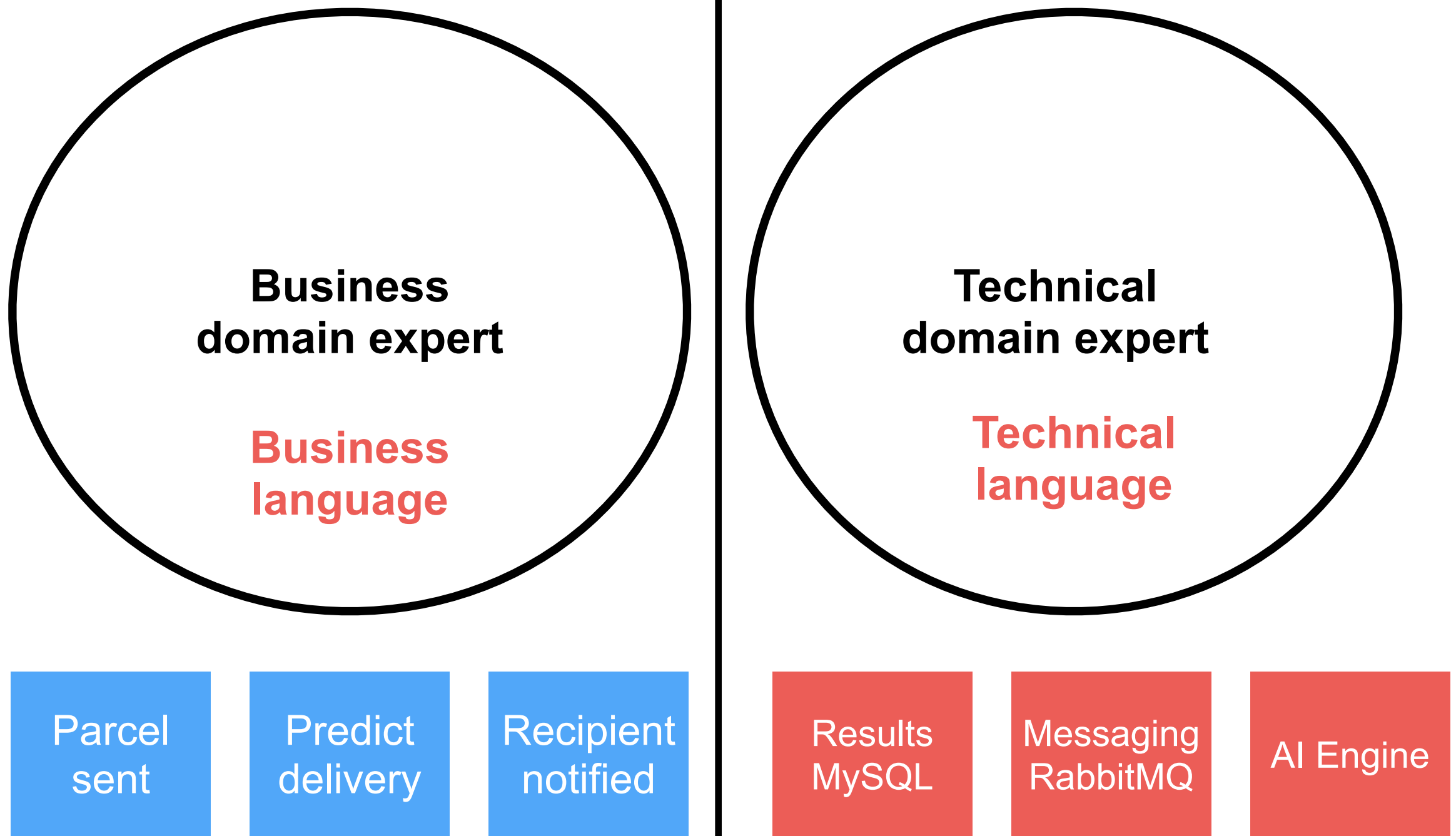
**Technical  
domain expert**

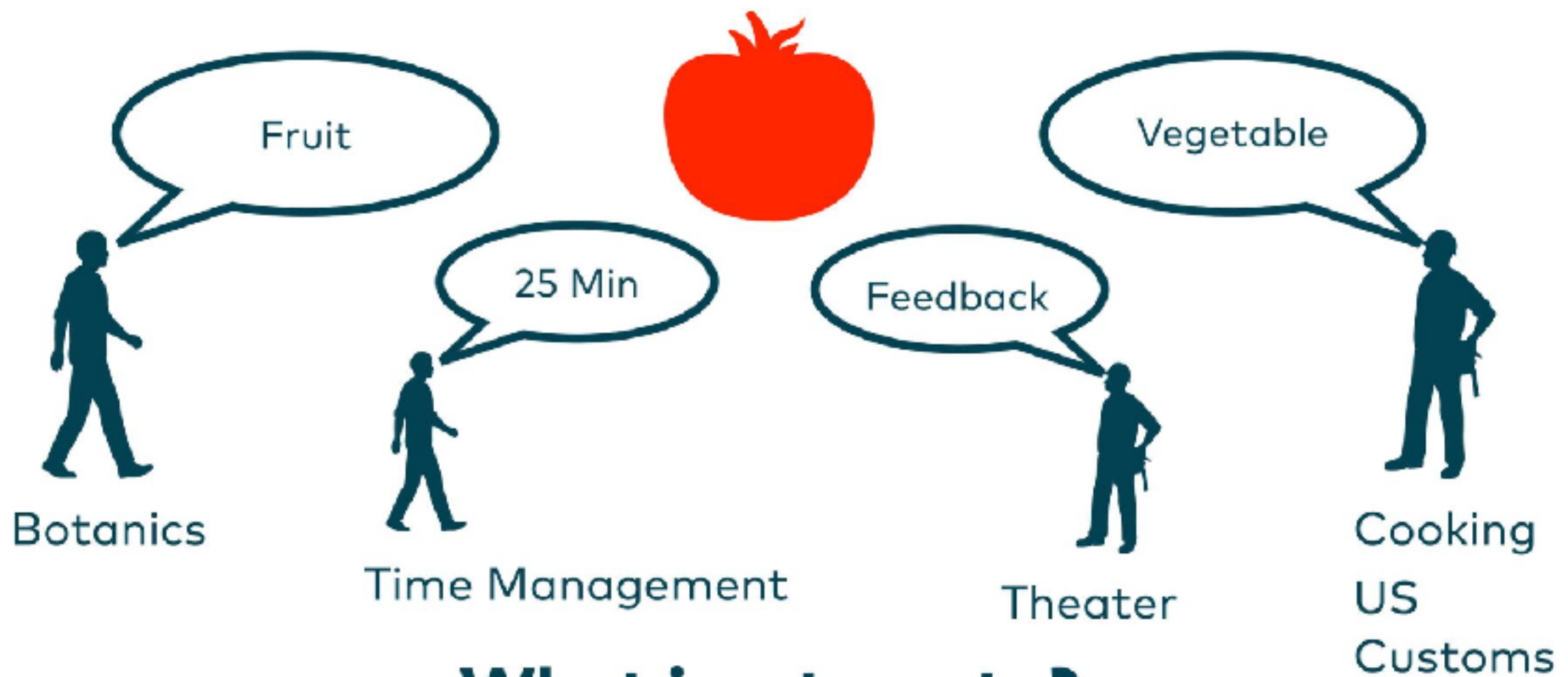
**Technical  
language**





# Problem with language ?





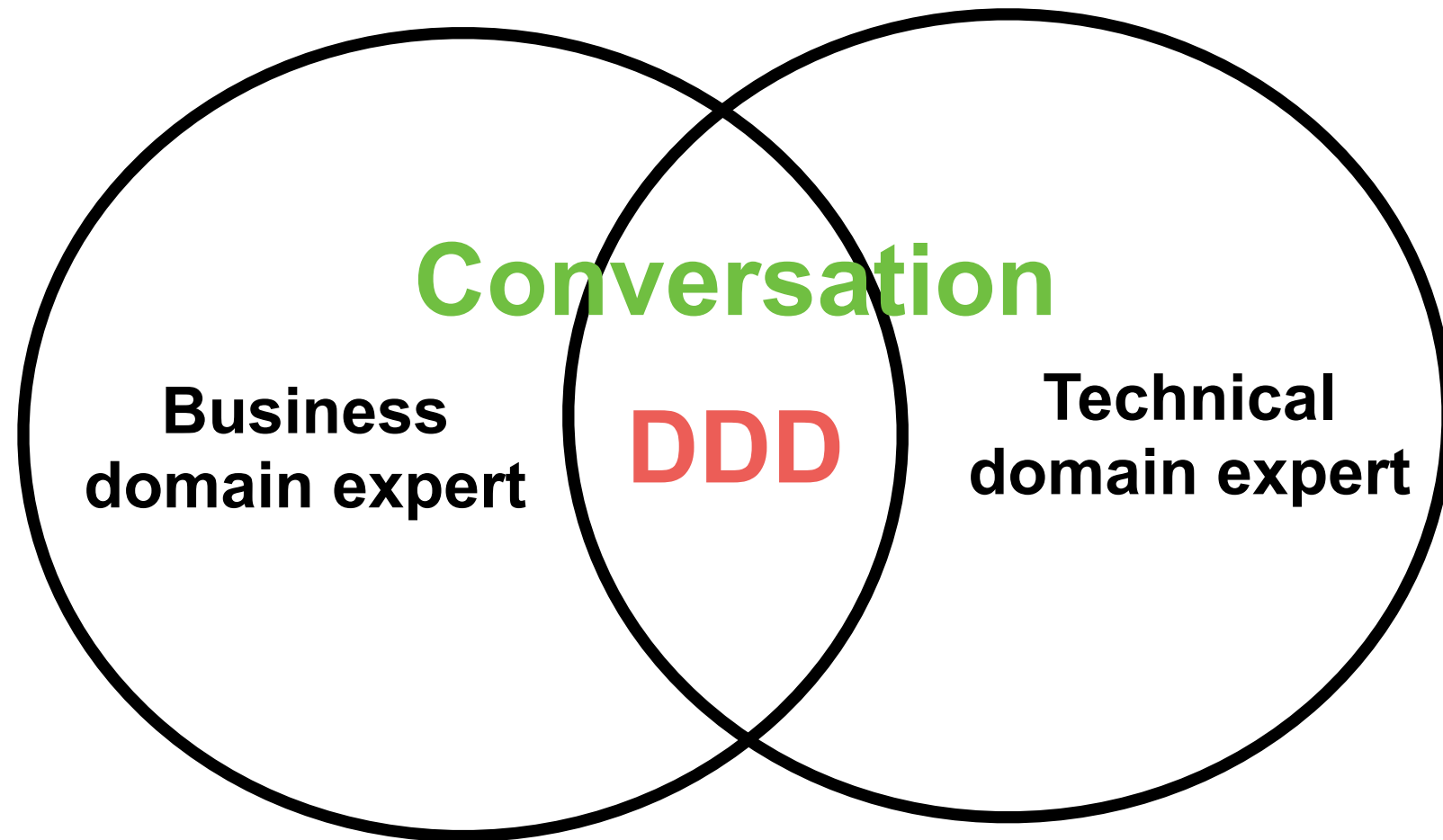
**What is a tomato?**

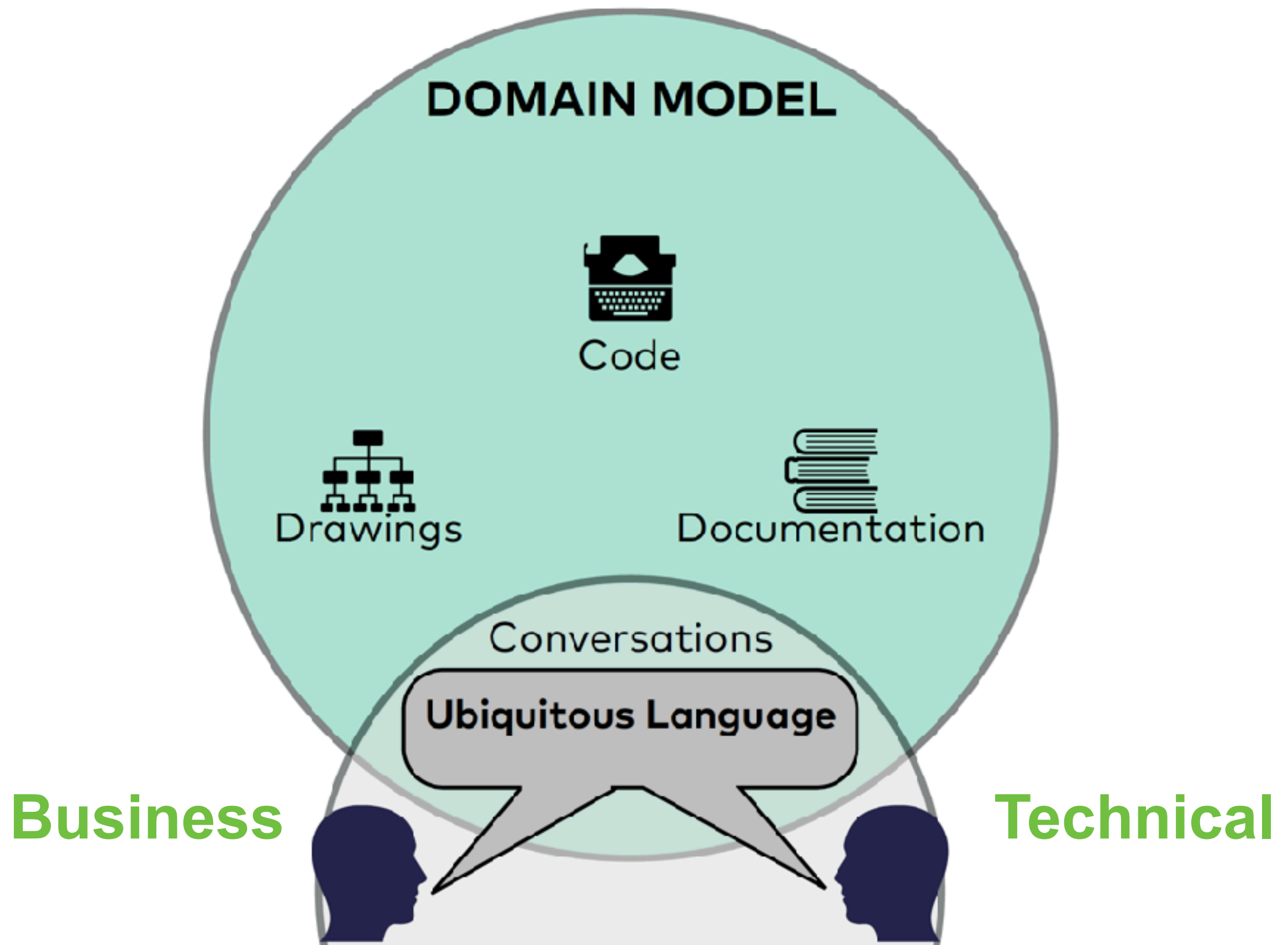


# Need Same Language ?



# Ubiquitous Language





# Need Same Direction ?





# Need for DDD

Top-down approach

Aligning with business

Communication (ubiquitous language)

Remaining flexible and scalable for biz changing

Mapping dependencies (clear boundary)



# Start with Problem Space



## Problem Space

- Defined by Users -

- WHAT to solve
- Fact: How things are
- Feelings: How users feel

\* Absolutely NO SOLUTIONS



## Solution Space

- Defined by Product Team -

- HOW product solves the problem
- A specific implementation

\* Prerequisite: Understanding of the problem



# Start with Problem ?

Ubiquitous  
language

Effective  
communication

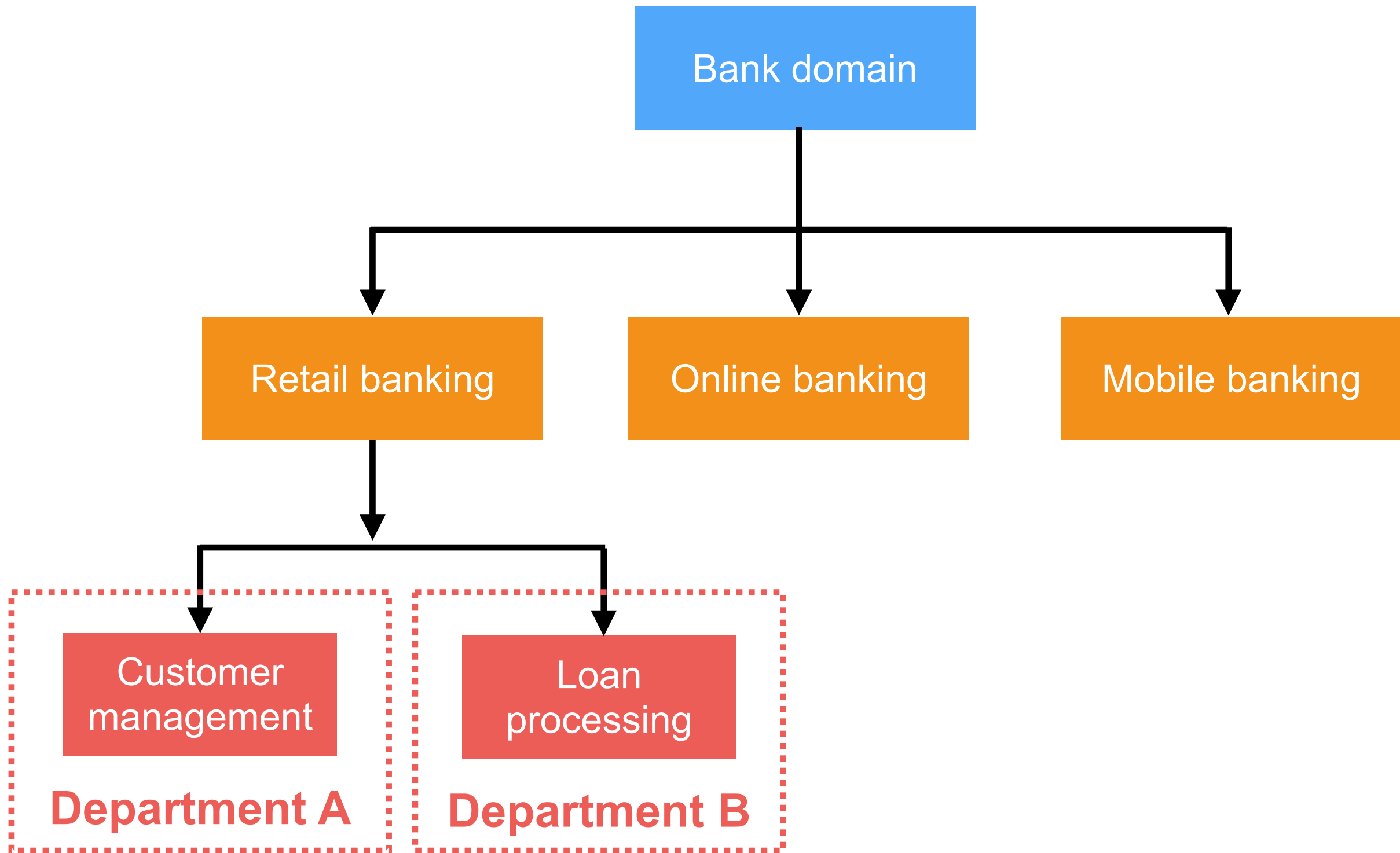
Knowledge  
Crunching

Business rules

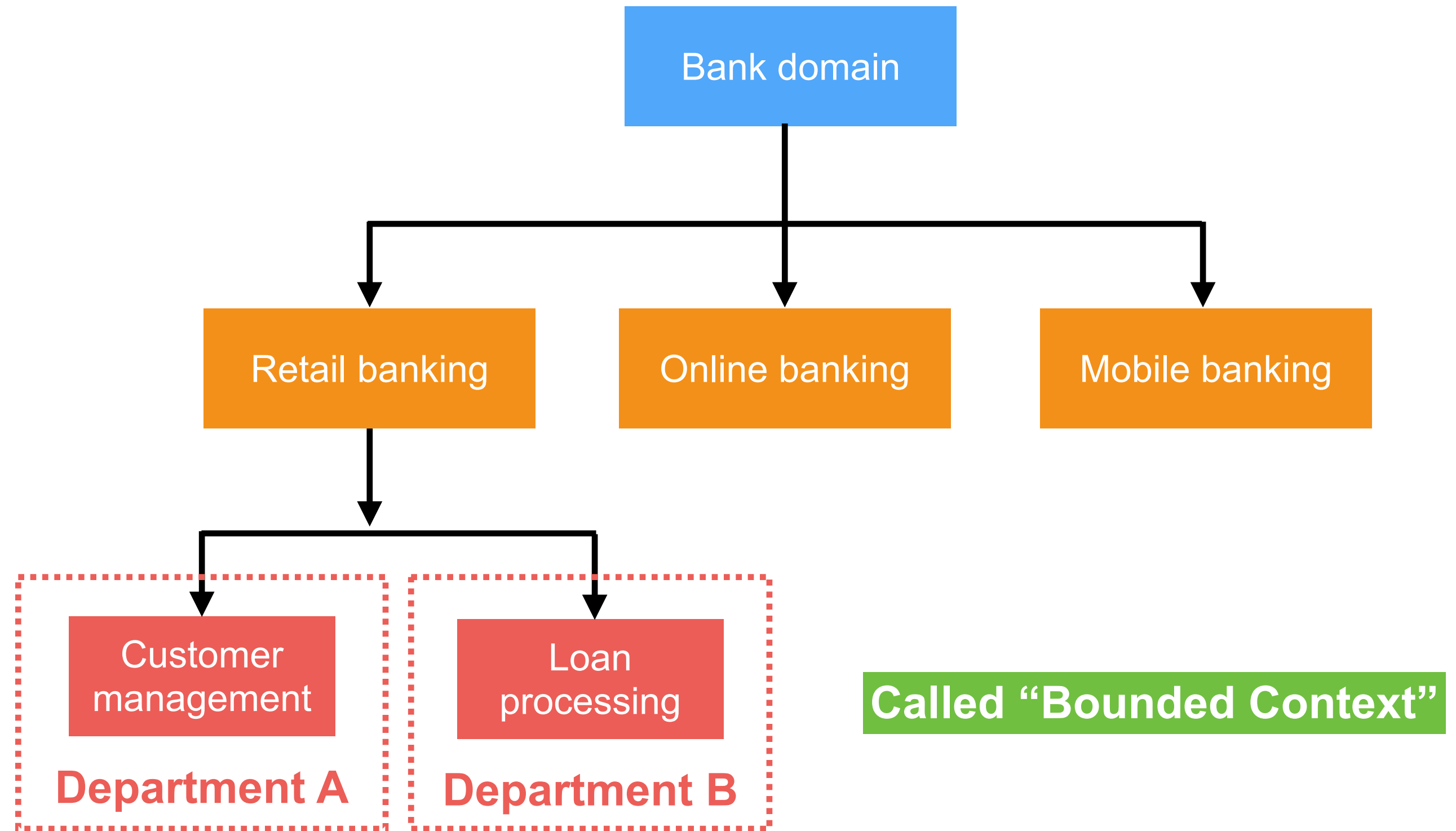
Features



# Example of Bank



# Example of Bank

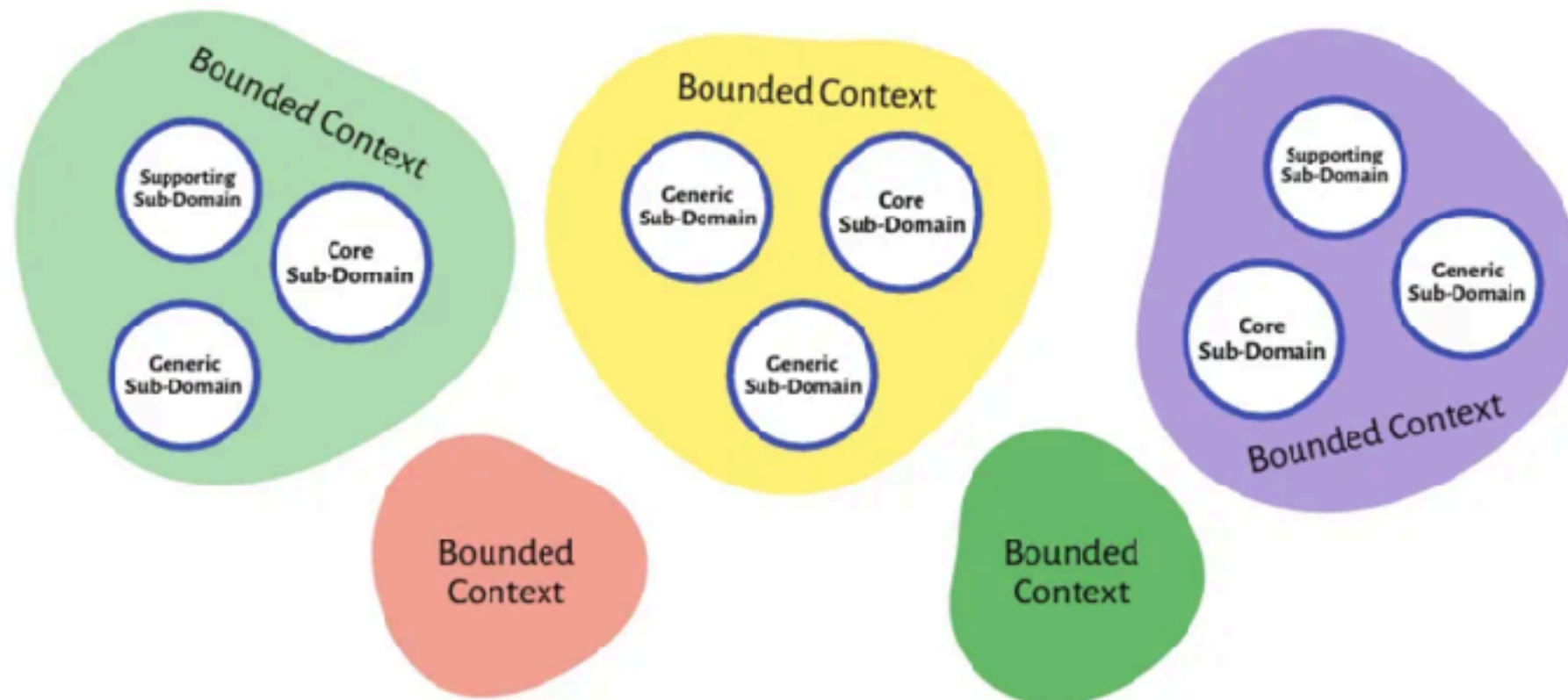


# Retail Banking

Bounded Context !!

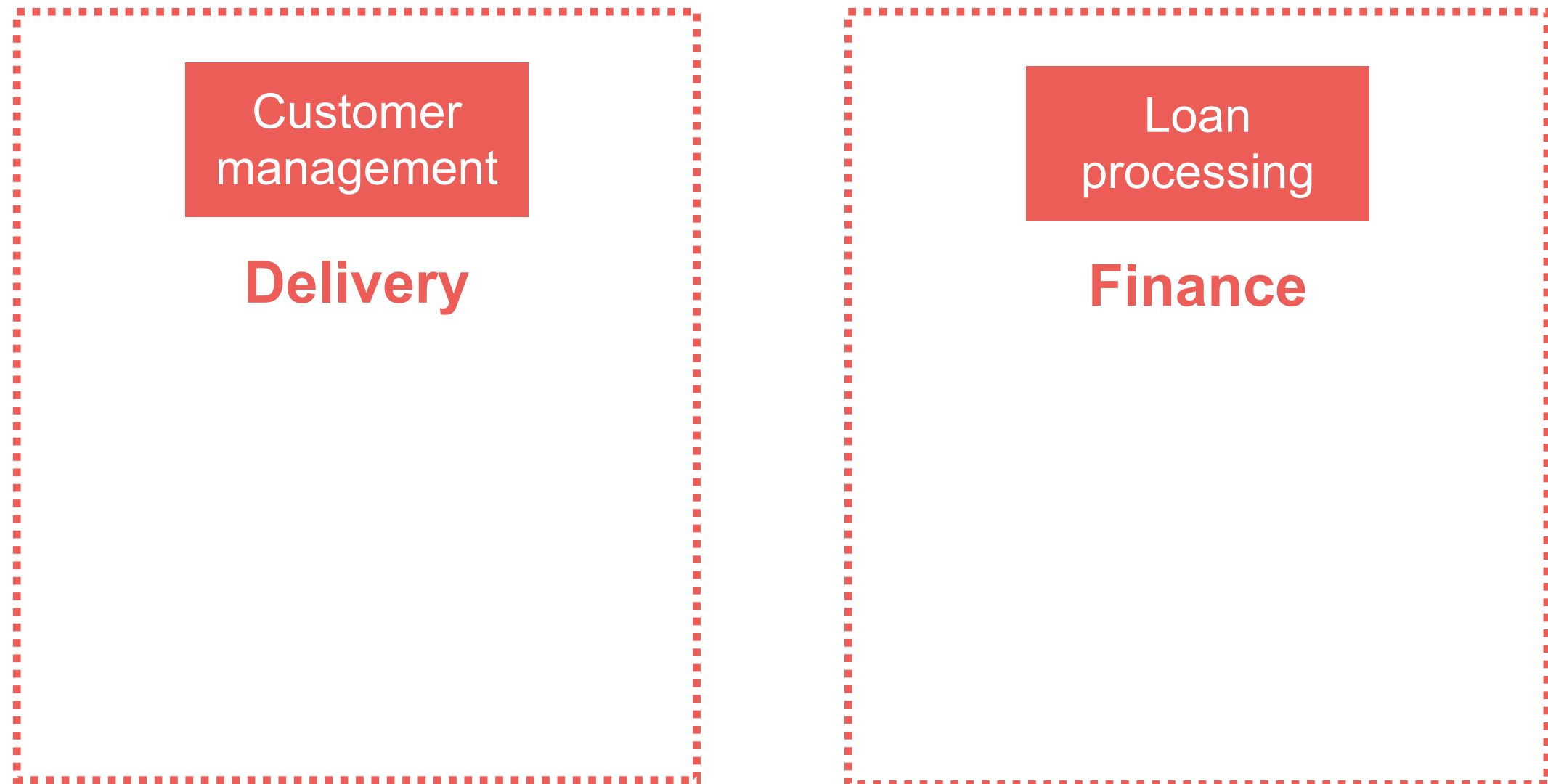


# Bounded Context



# Retail Banking

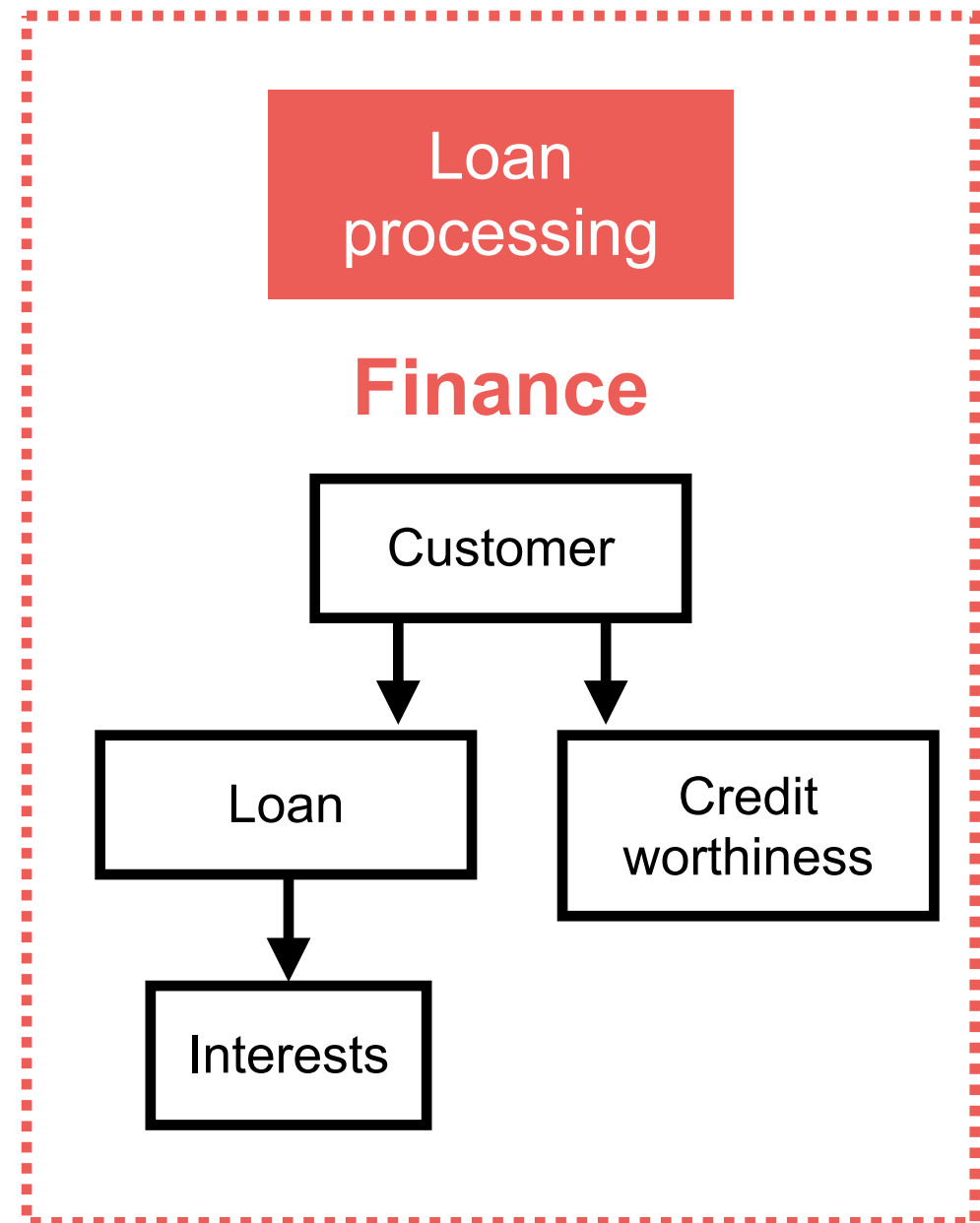
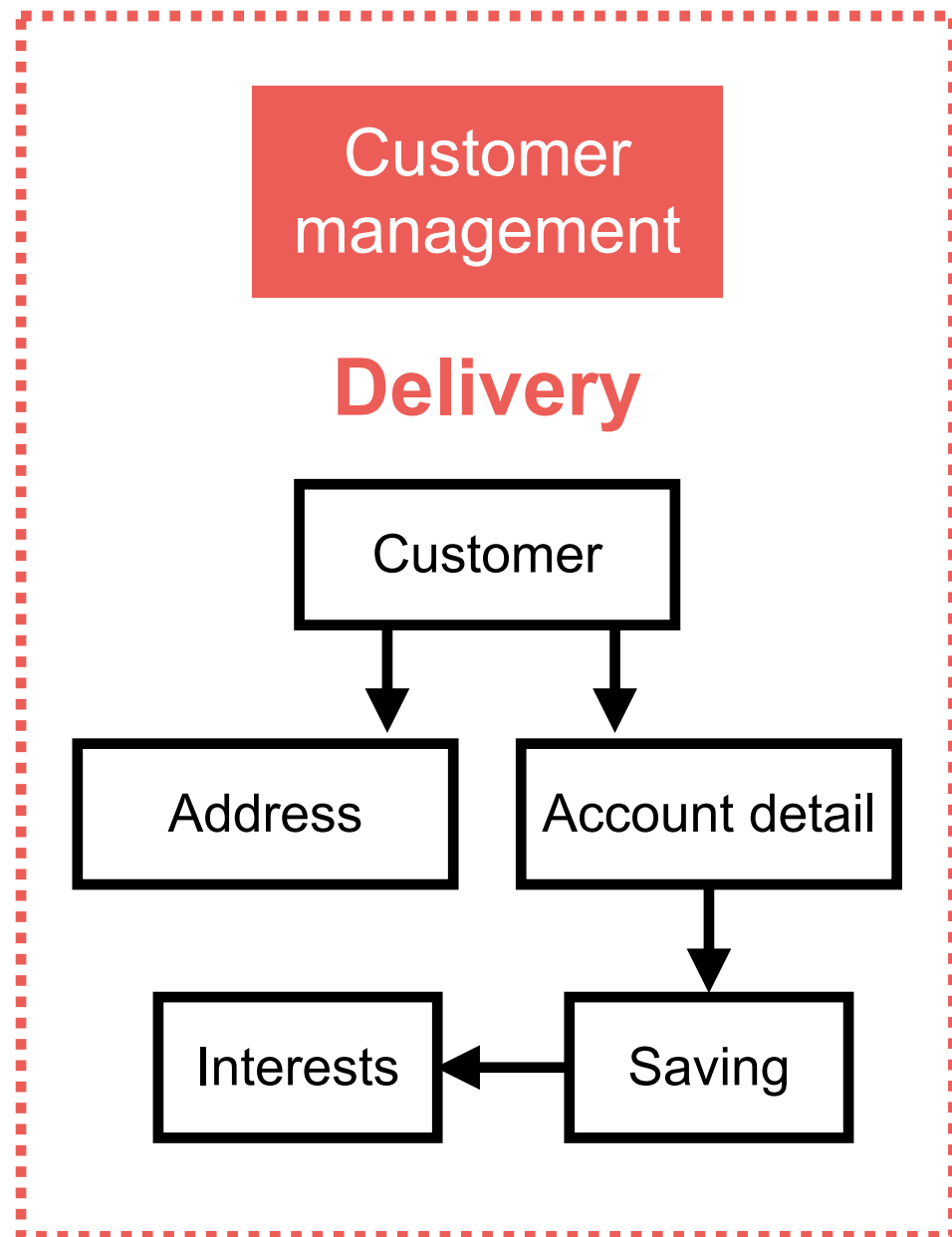
Language in each Bounded Context !!





# Each context has own language

Need “Domain Expert”

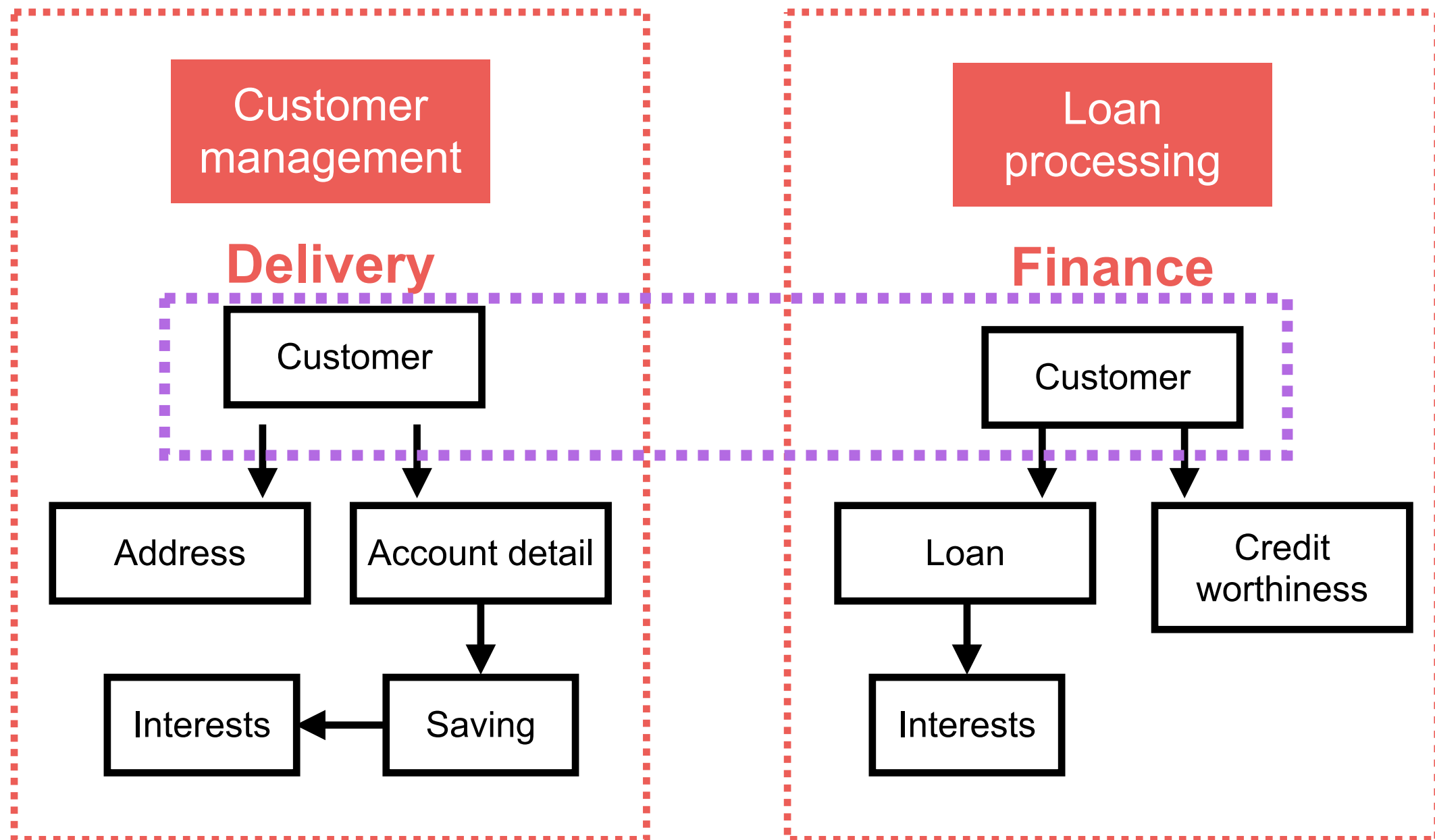


# **Relationship between bounded contexts ?**

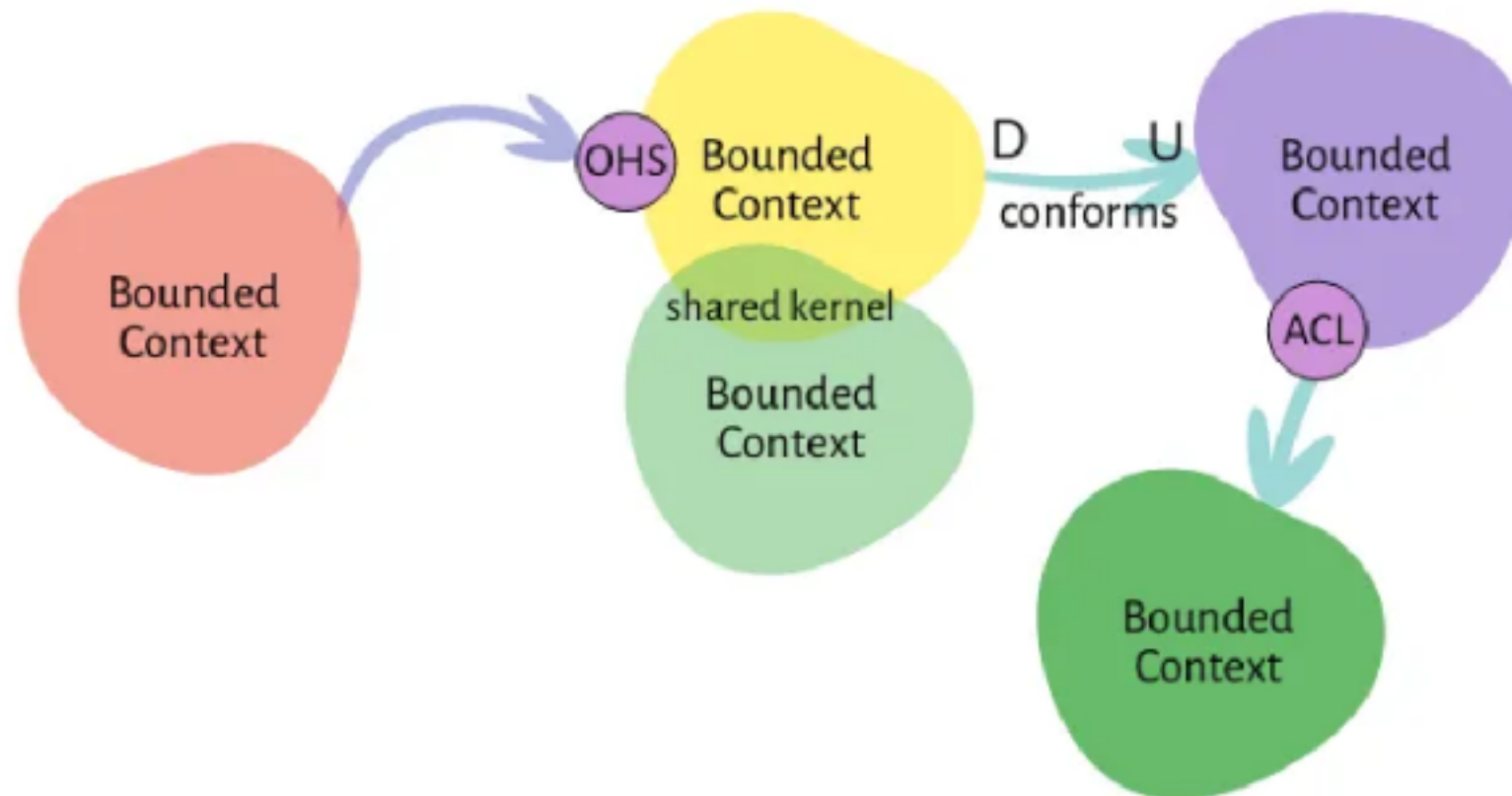


# Relationship ?

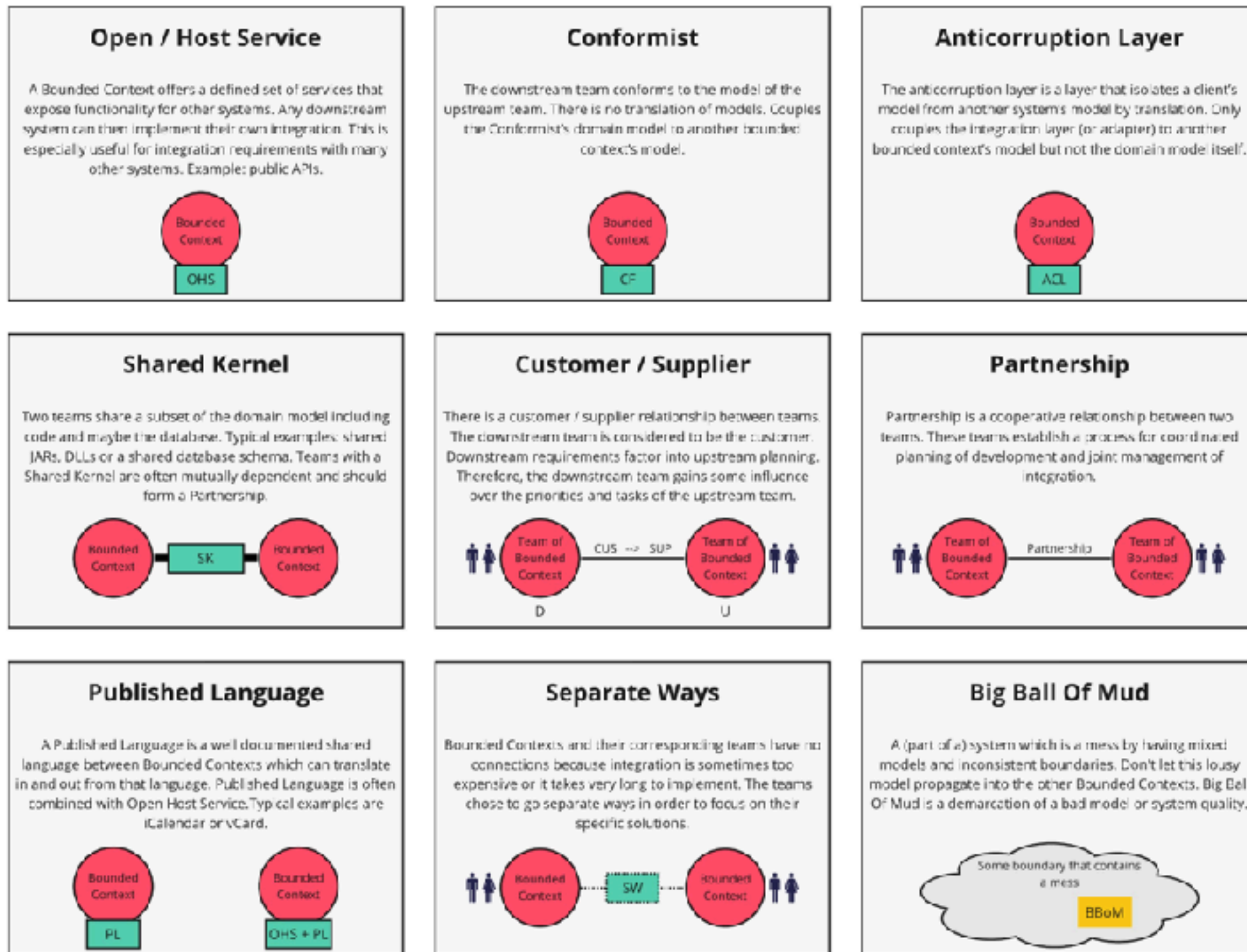
Called “Context Mapping”



# Context Mapping



# Context Mapping



<https://github.com/ddd-crew/context-mapping/>



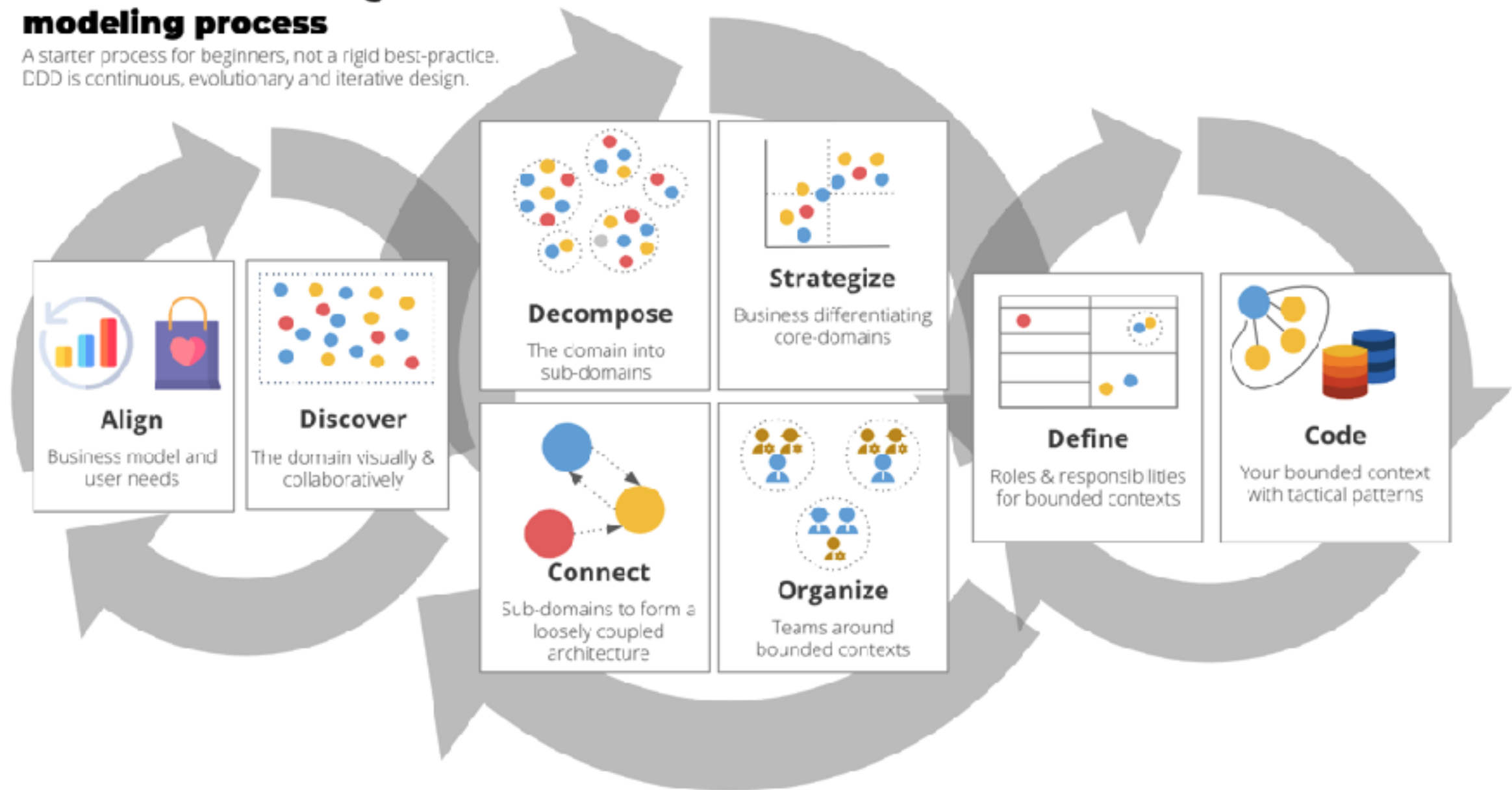
# DDD Modeling process

Continuous, evolutionary and iterative design



## Domain-Driven Design starter modeling process

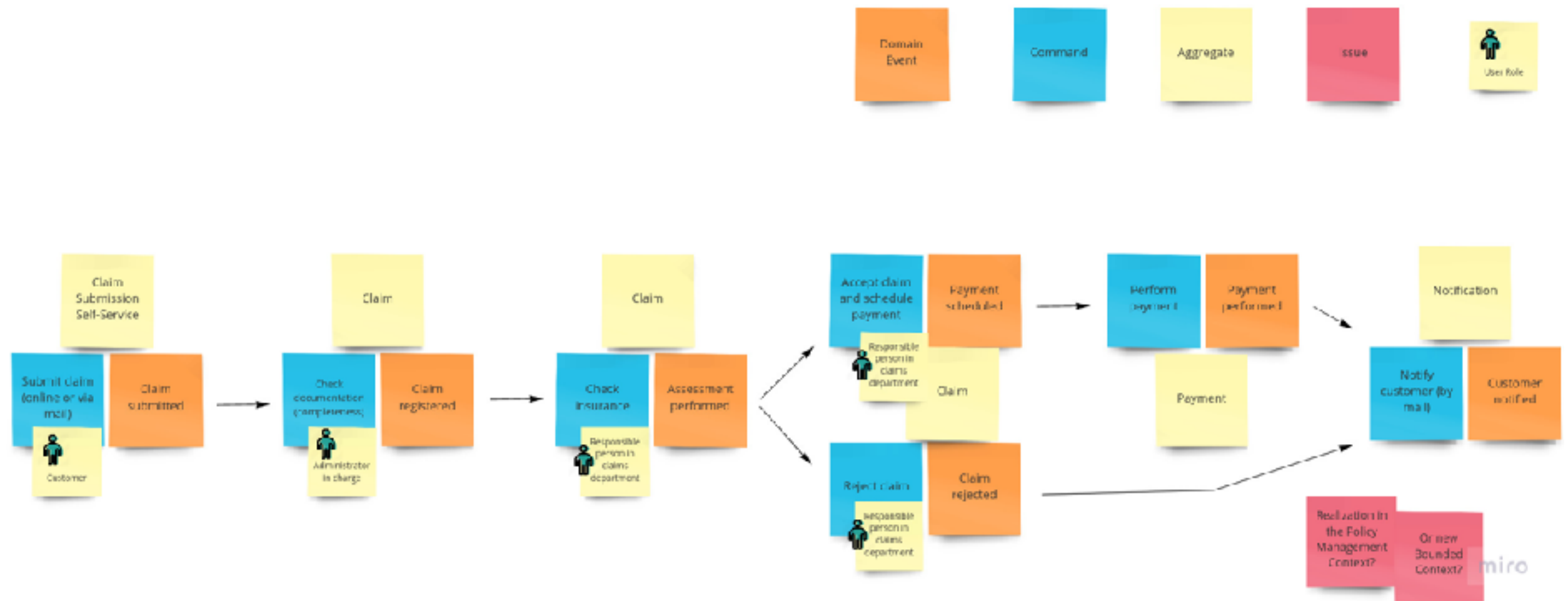
A starter process for beginners, not a rigid best-practice.  
DDD is continuous, evolutionary and iterative design.



<https://github.com/ddd-crew/ddd-starter-modelling-process>



# Event Storming



<https://contextmapper.org/docs/event-storming/>





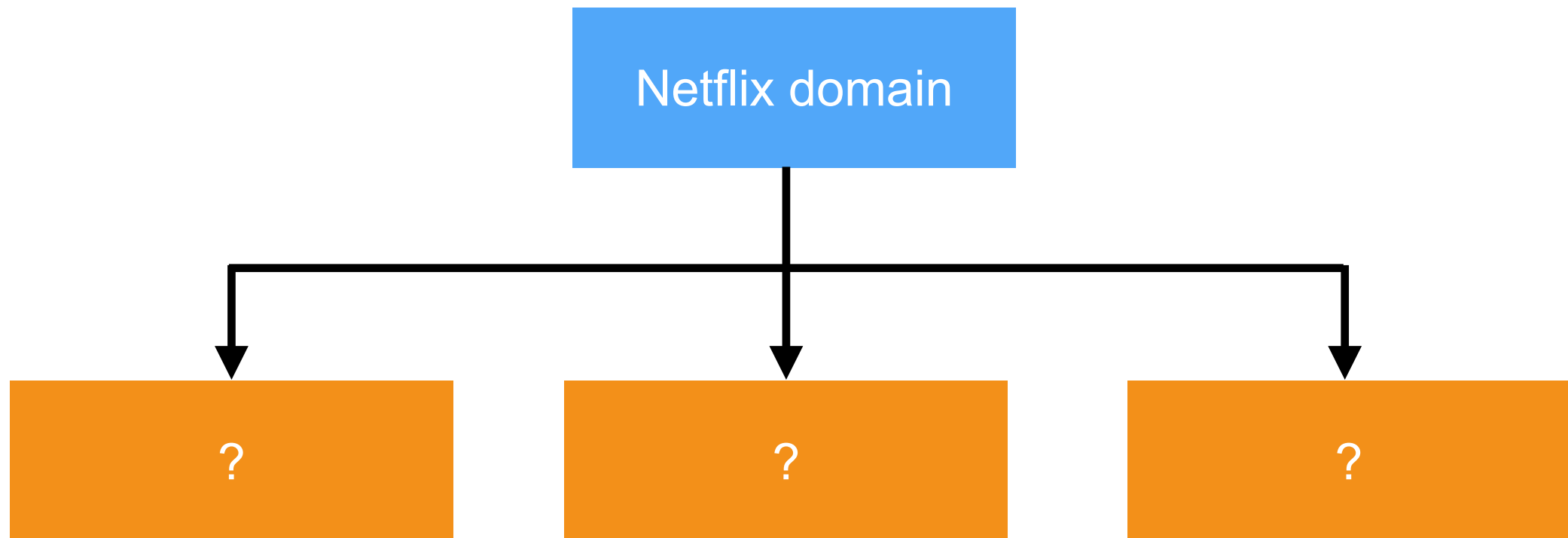
# Workshop



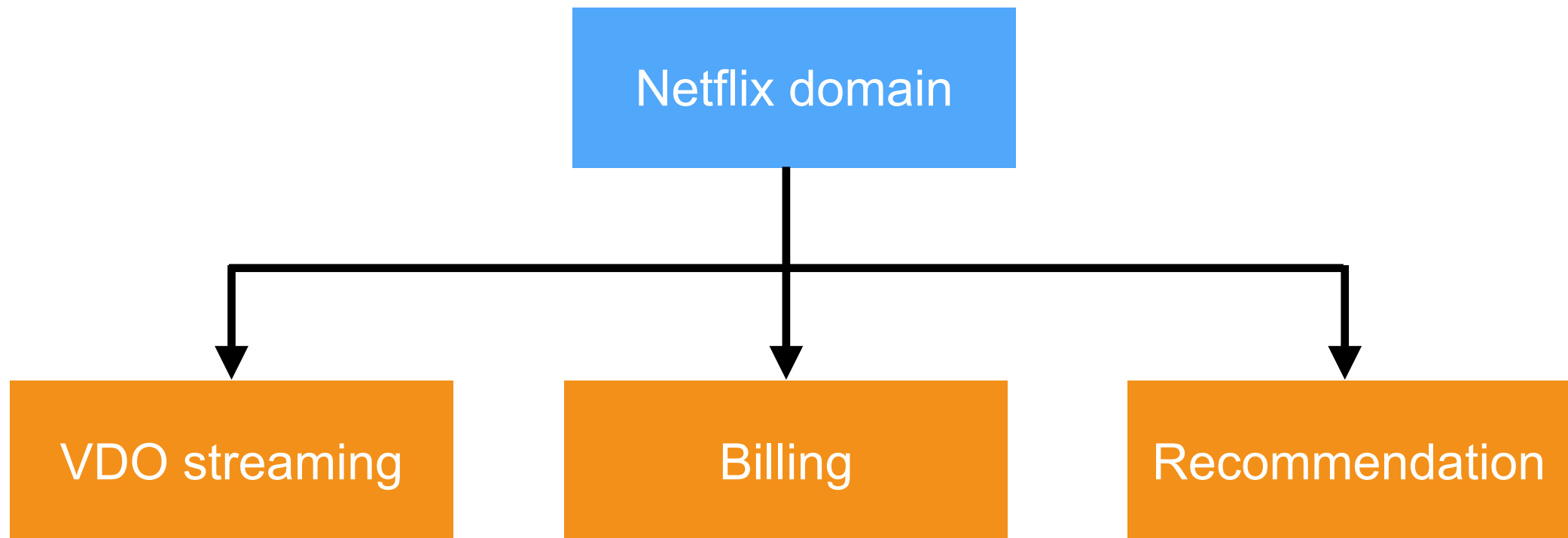
# Example



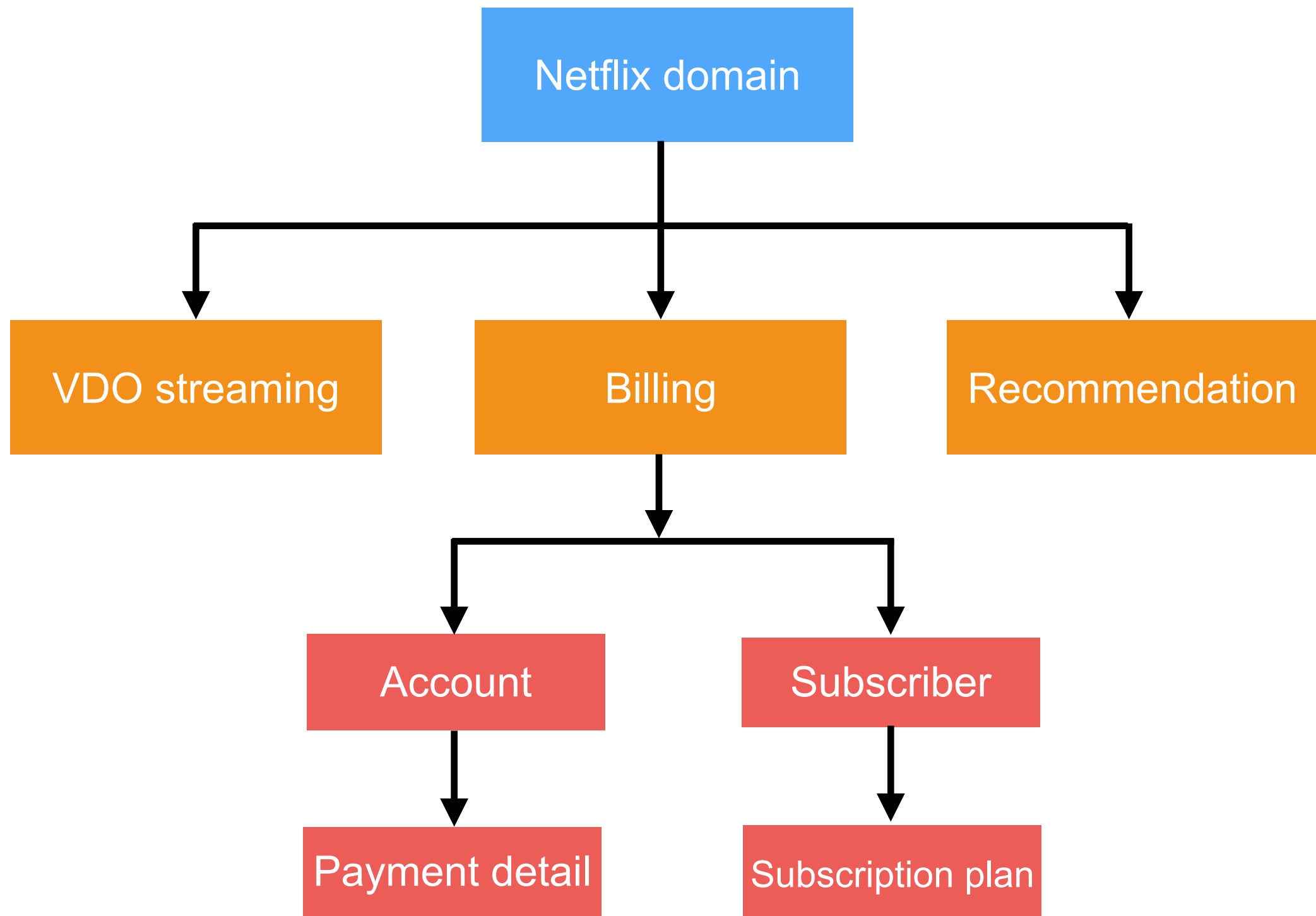
# Streaming Platform (1)



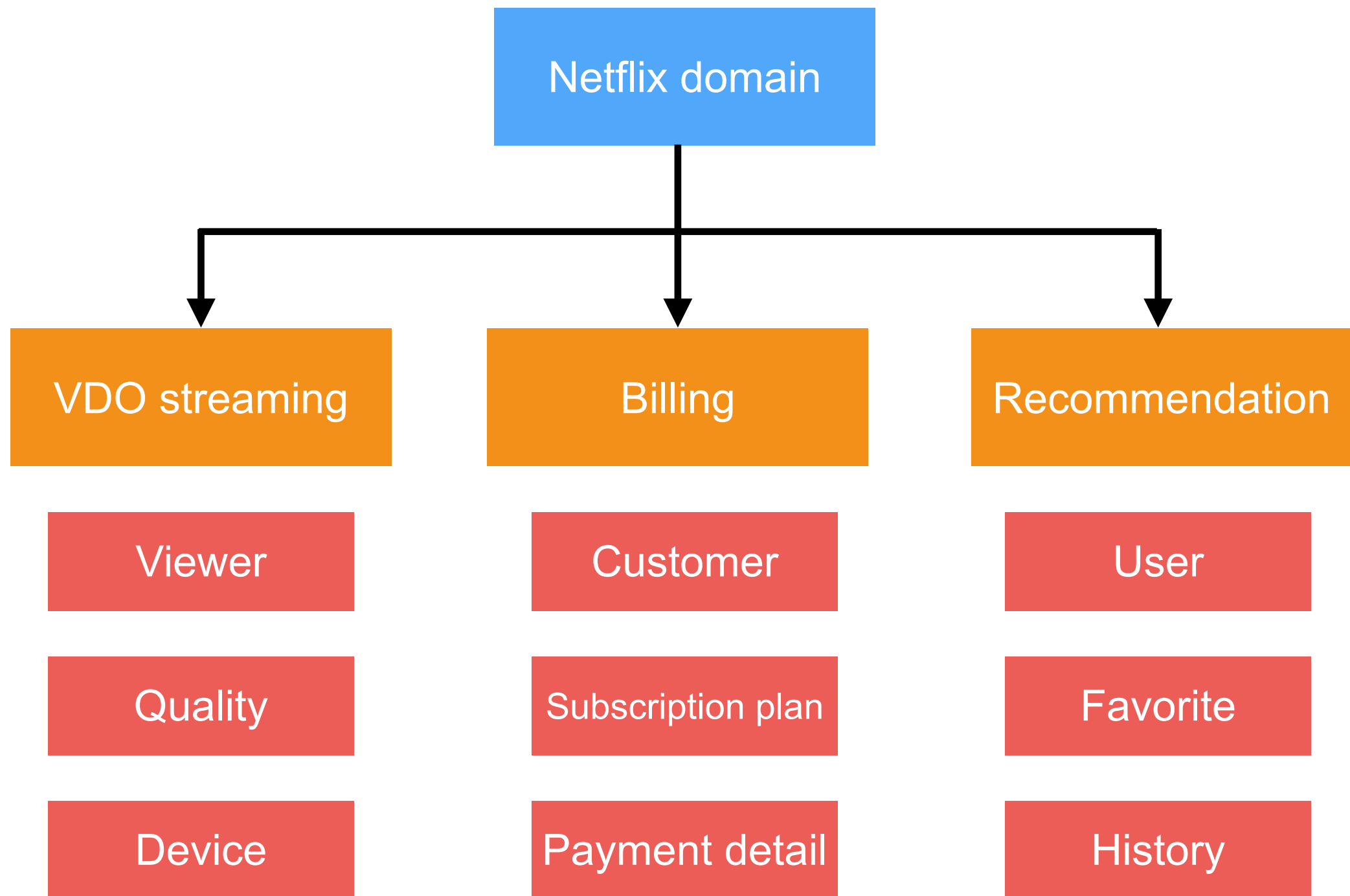
# Subdomain ?



# Streaming Platform (3)



# Streaming Platform (4)



# EA (Enterprise Architect)

responsible for aligning an organization's IT strategy  
with its overall business goals



# EA (Enterprise Architect)

