Microservices with Domain-Driven Design #1/3

Credit – copy paste engineer

Pros & Cons of Microservices

ข้อดี

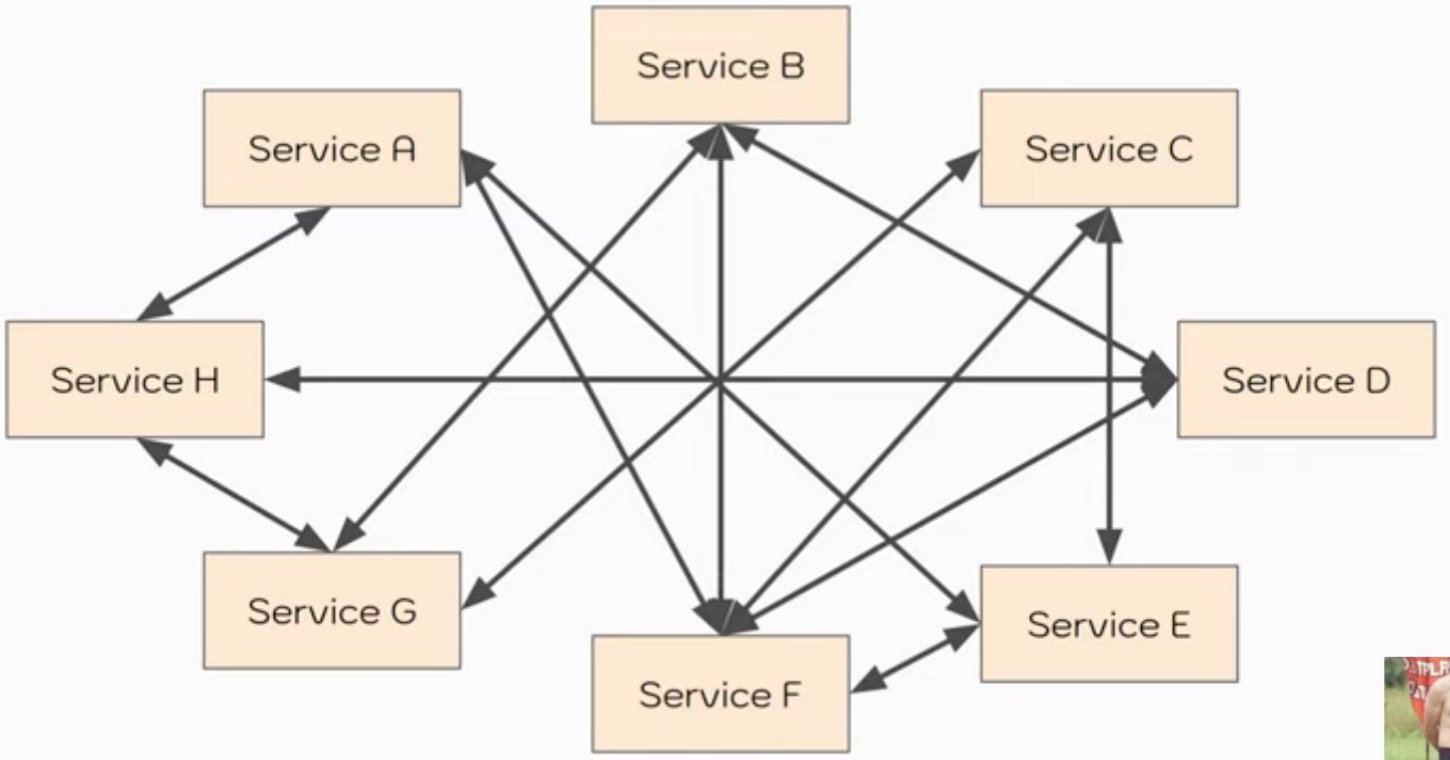
- แต่ละ team สามารถ focus งานแต่ละส่วนได้ แยกจากกัน
- แต่ละ service สามารถ test/deploy แยกกันได้ ไม่ต้องรอกัน
- scale แต่ละส่วนแยกกันได้ตามความจำเป็น

ข้อเสีย

- ออกแบบ (ให้ดี) ยากกว่า และงานหนักกว่าถ้าองค์กรมีคนไม่มาก
- ต้องใช้ techniques ที่ซับซ้อน เช่นในการสื่อสารระหว่าง services
- team ต้องมีความพร้อมในเรื่องการทำ automation

Why Microservices

- Highly maintainable and testable
- Loosely coupled
- Independently deployable
- Organized around business capabilities
- Owned by a small team



Big Ball of Mud Microservices - Anti-pattern

Big Ball of Mud

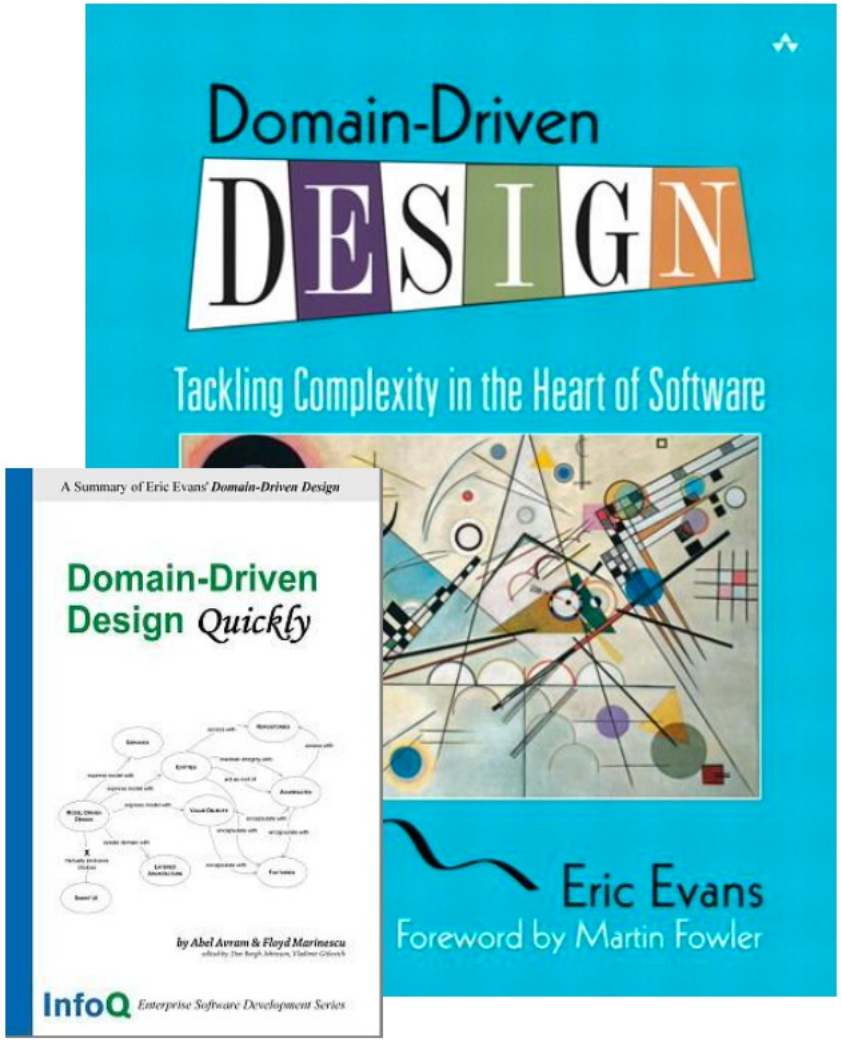
Microservices เป็นเรื่องของ Architecture

DDD เป็นเรื่องของ การออกแบบ

Domain-Driven Design

- Ubiquitous Language
- Bounded Context
- Subdomain
- Entity
- Value Object
- Service
- Repository
- Aggregate

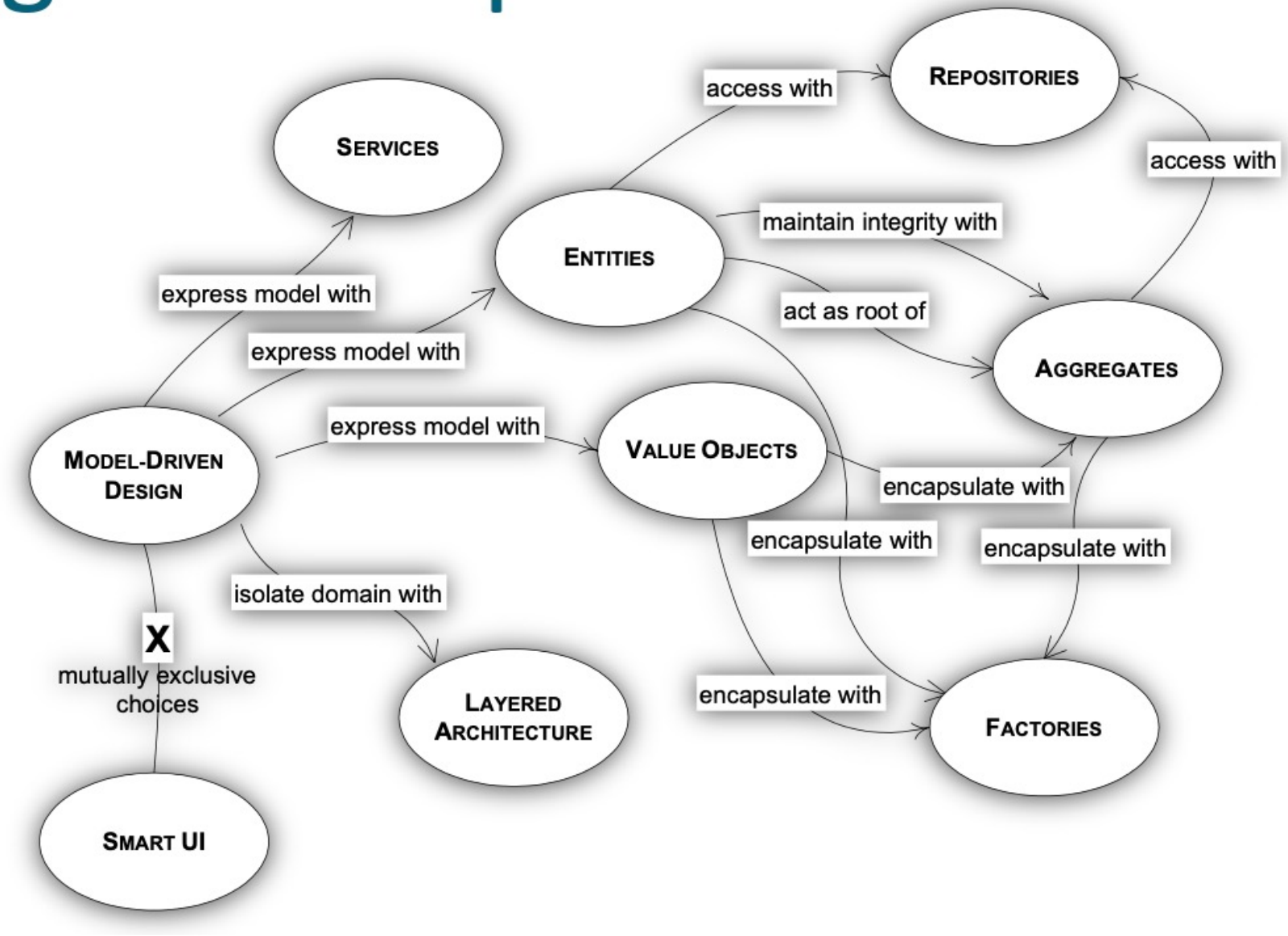
Domain-Driven Design: Tackling Complexity in the Heart of Software by Eric Evans



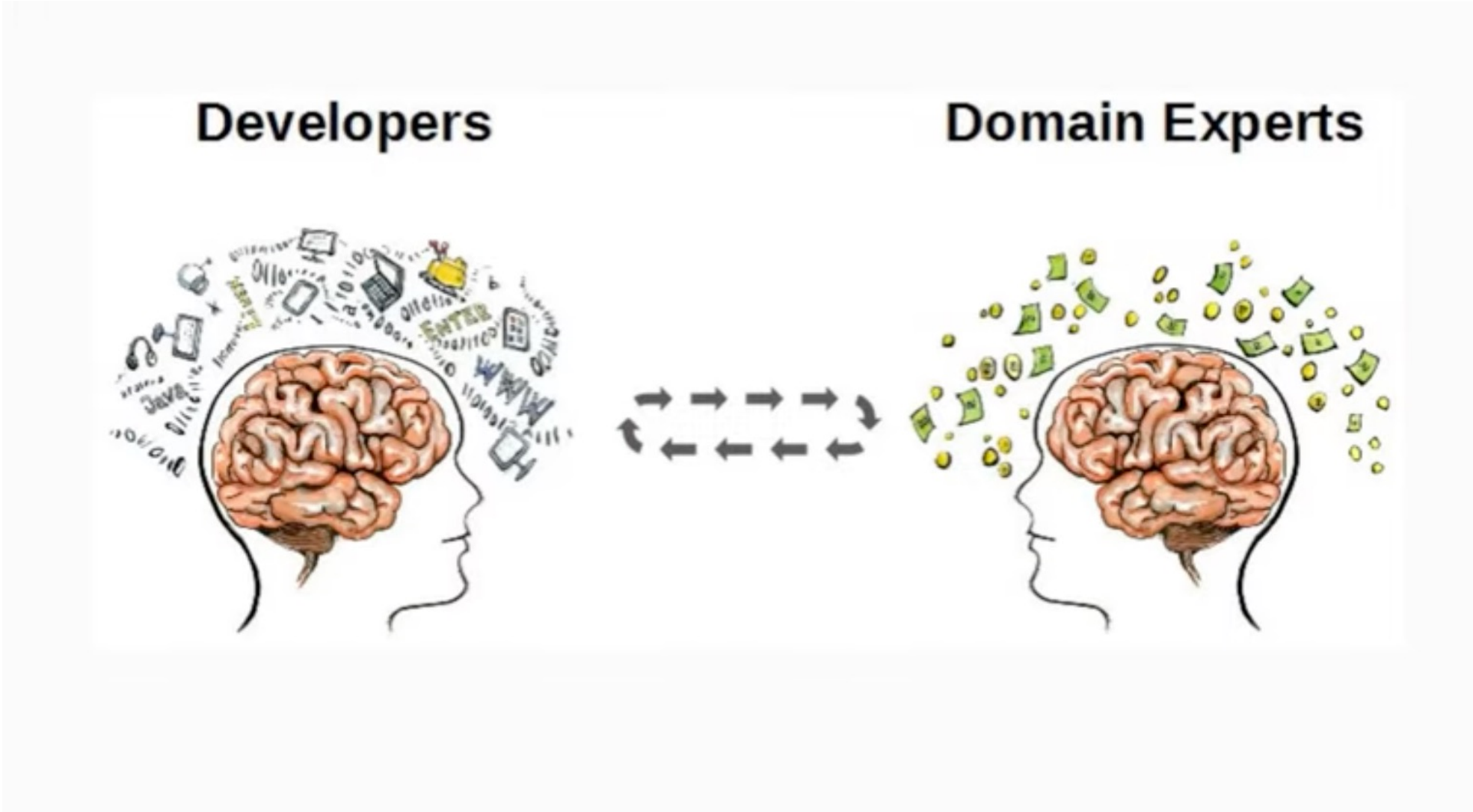
Domain-Driven Design

- Software models some aspect of the real world
- We build design models to understand what we are building, and how we will build it
- Symmetry between our software, design model, and the real world allow us to adjust to changes in the real world

Navigation Map



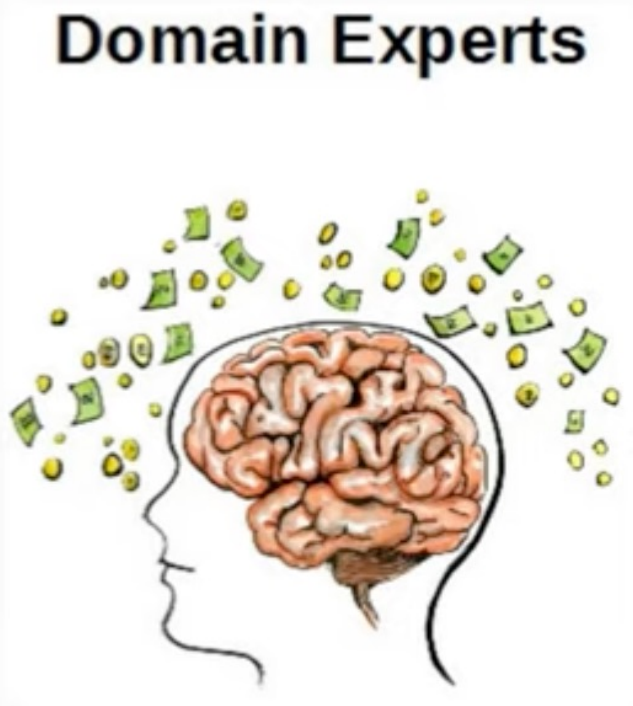
Communication Gap



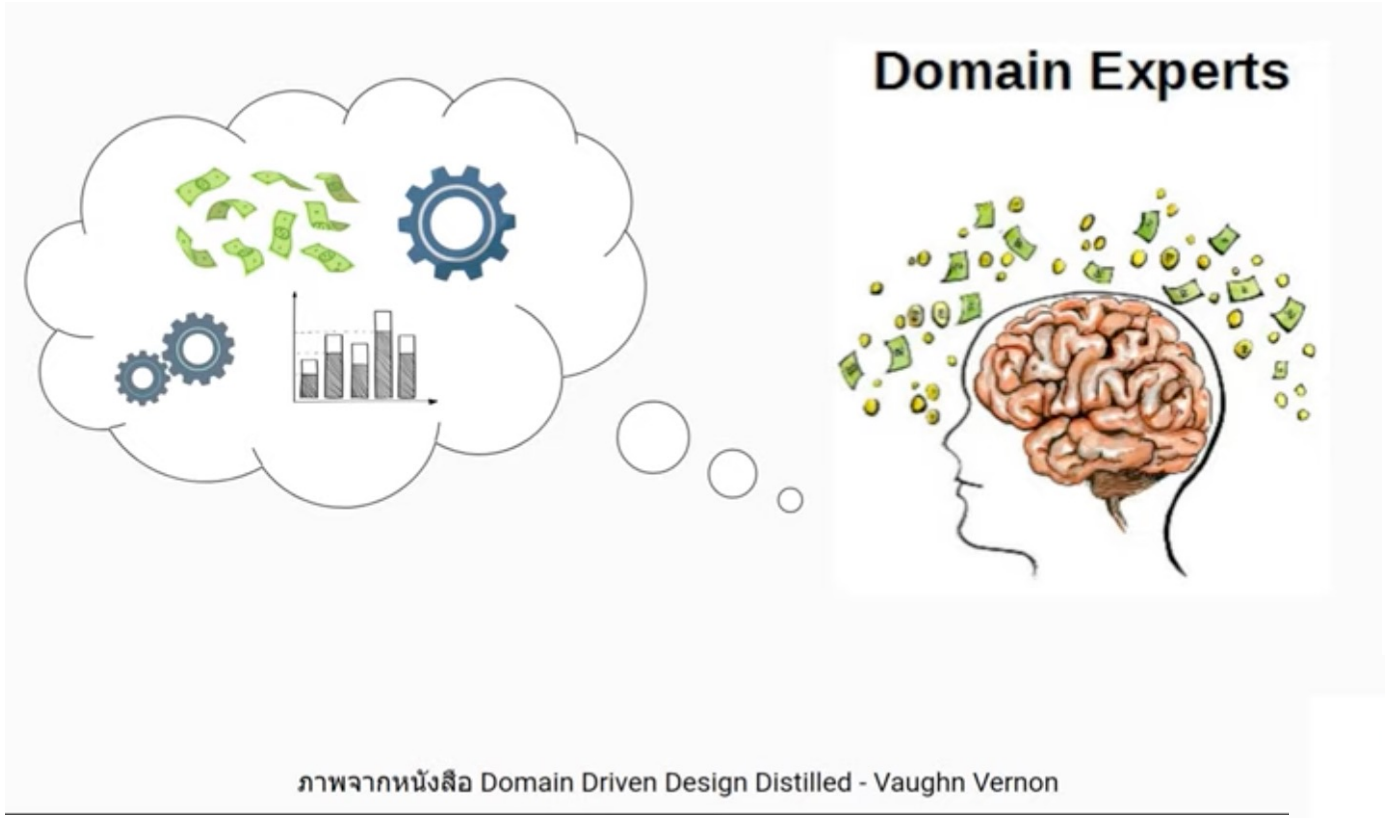
Developers need to work closely with Domain Experts!

คือ ใครก็ได้ ที่มีความรู้
ในงานที่เรากำลังจะทำ

Focus business = $$$

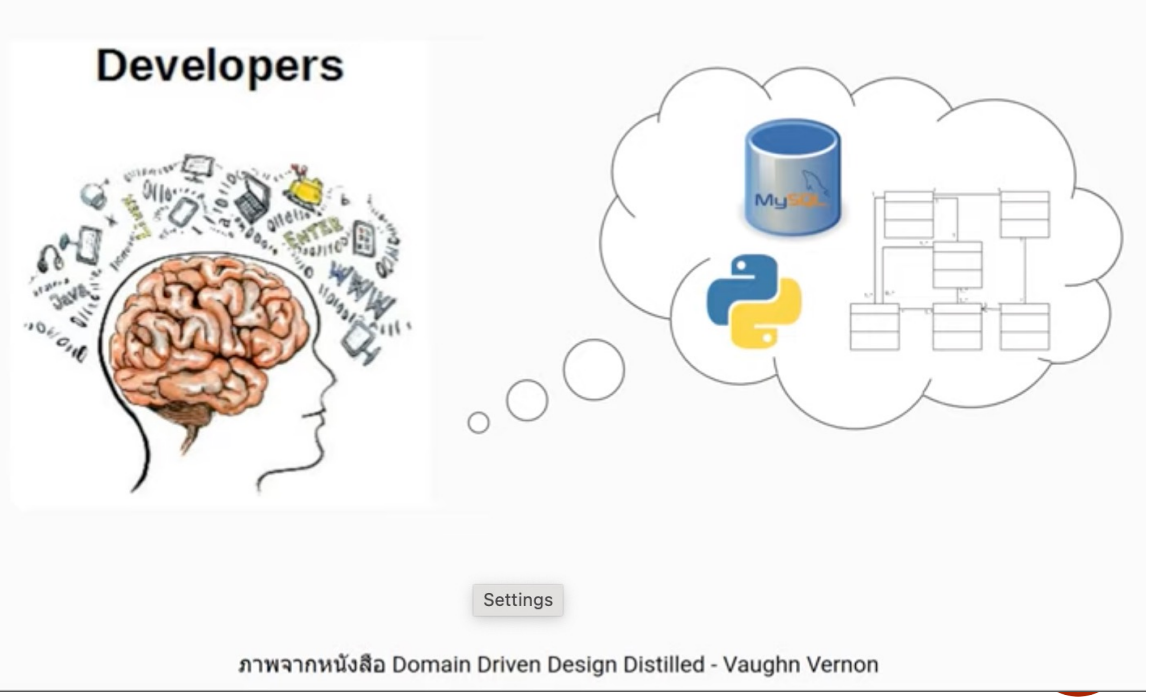


ภาพจากหนังสือ Domain Driven Design Distilled - Vaughn Vernon



ภาพจากหนังสือ Domain Driven Design Distilled - Vaughn Vernon

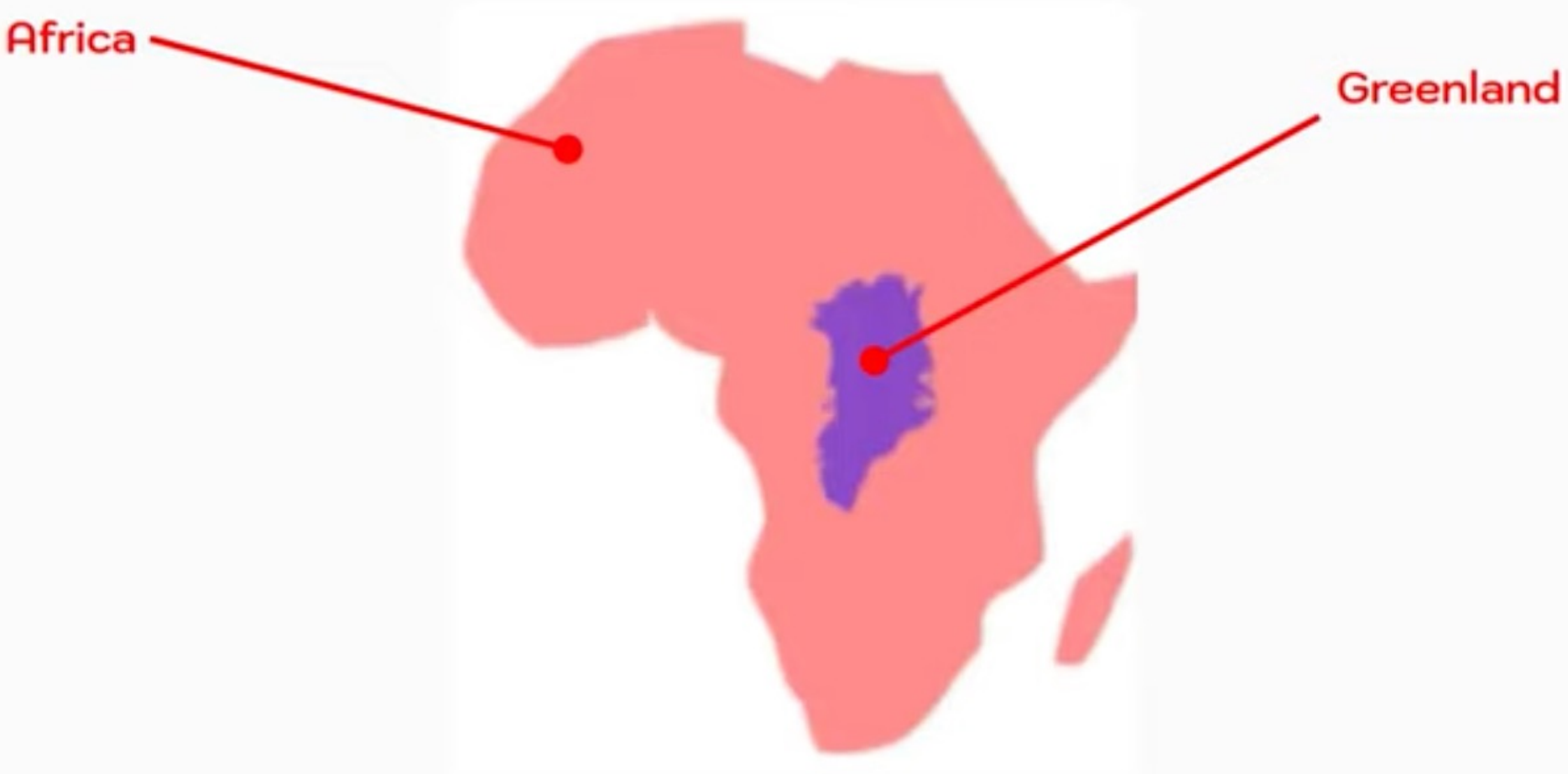
Different conceptual models between domain experts and developers are challenging!



Settings

ภาพจากหนังสือ Domain Driven Design Distilled - Vaughn Vernon

https://www.worldatlas.com/geography/world-map-mercator-projection.html

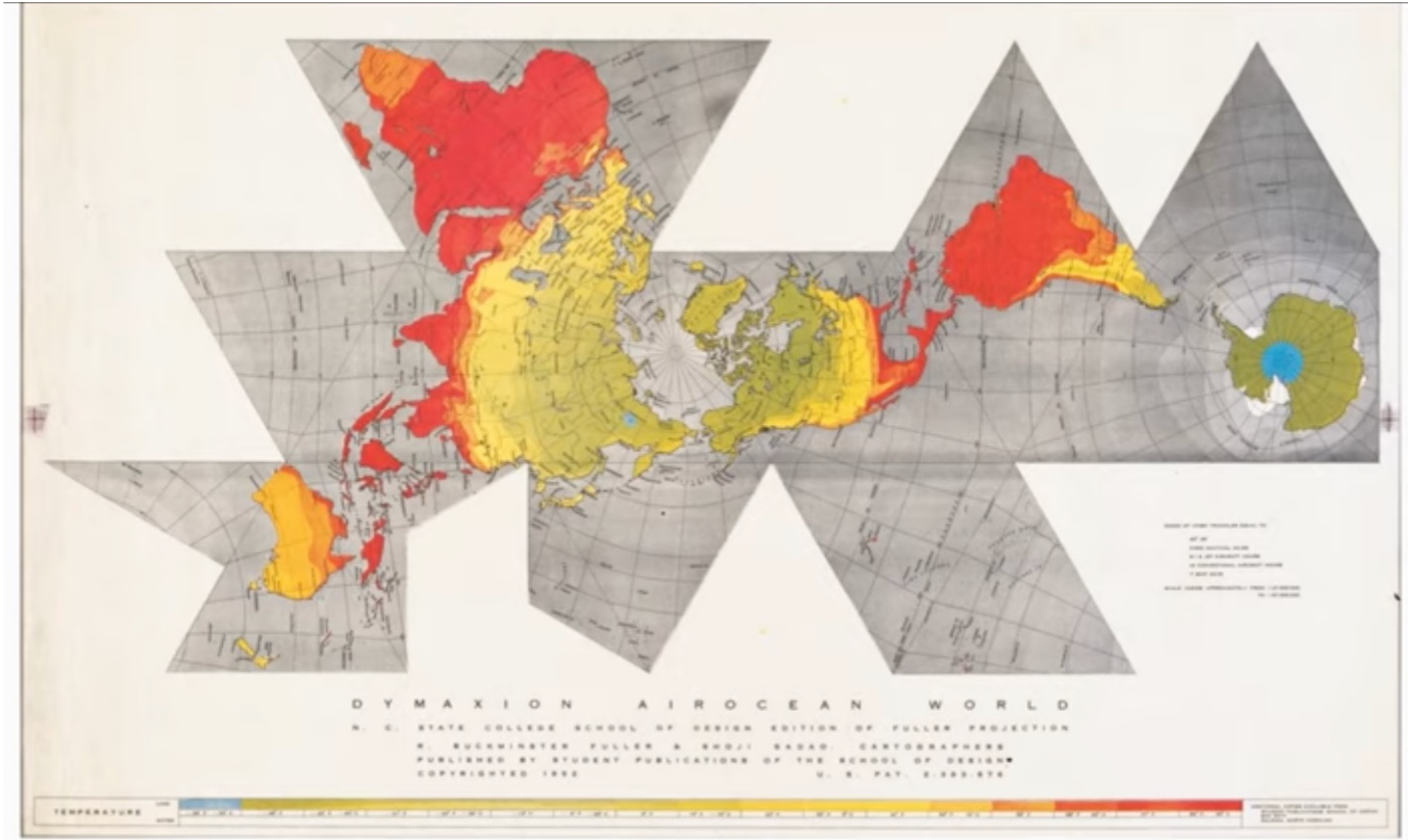


https://www.businessinsider.com/greenland-africa-comparison-2014-5

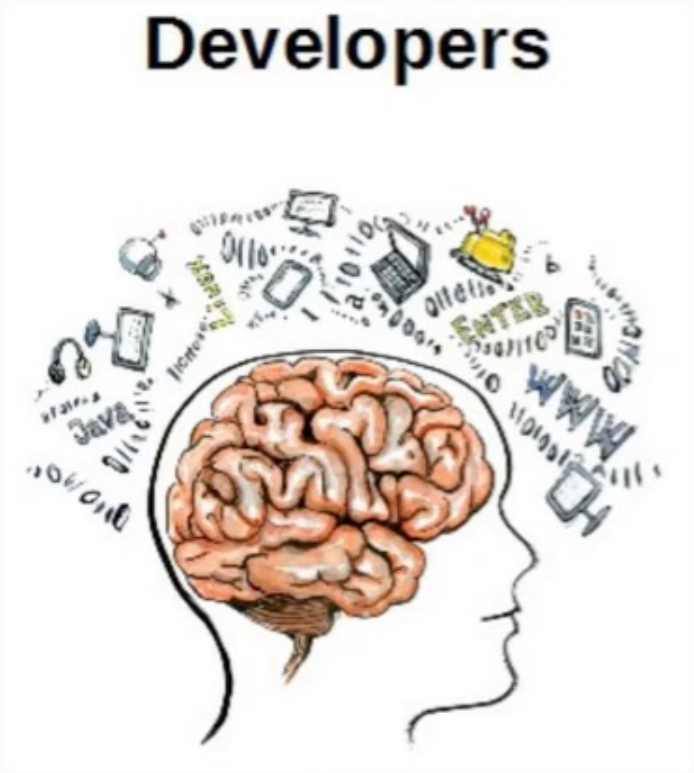
https://www.worldatlas.com/geography/world-map-mercator-projection.html

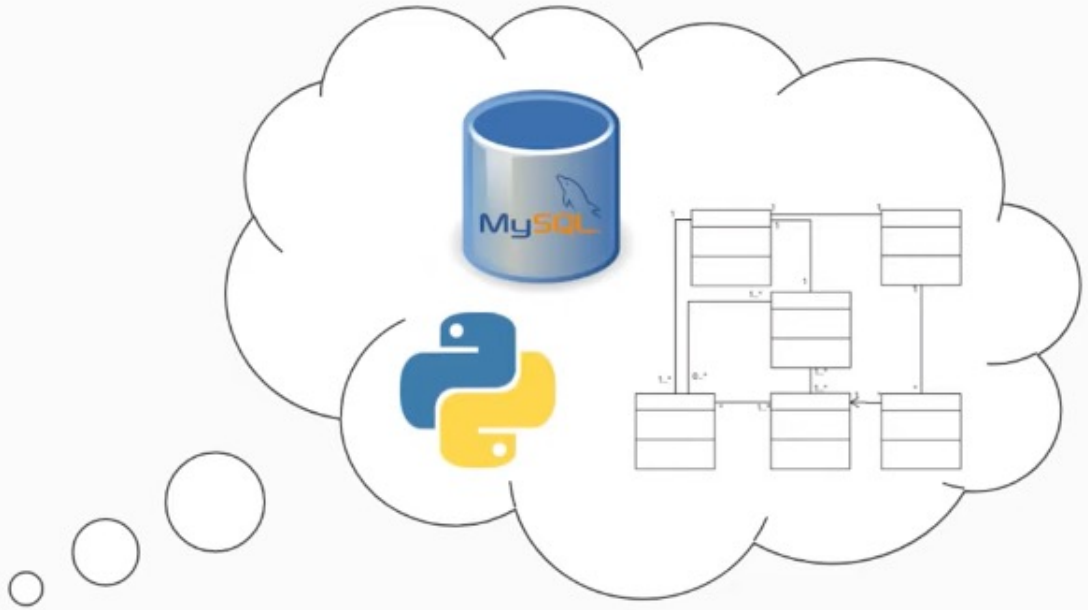
https://www.worldatlas.com/geography/world-map-mercator-projection.html

https://th.m.wikipedia.org/wiki/ไฟล์:Earth_Eastern_Hemisphere.jpg

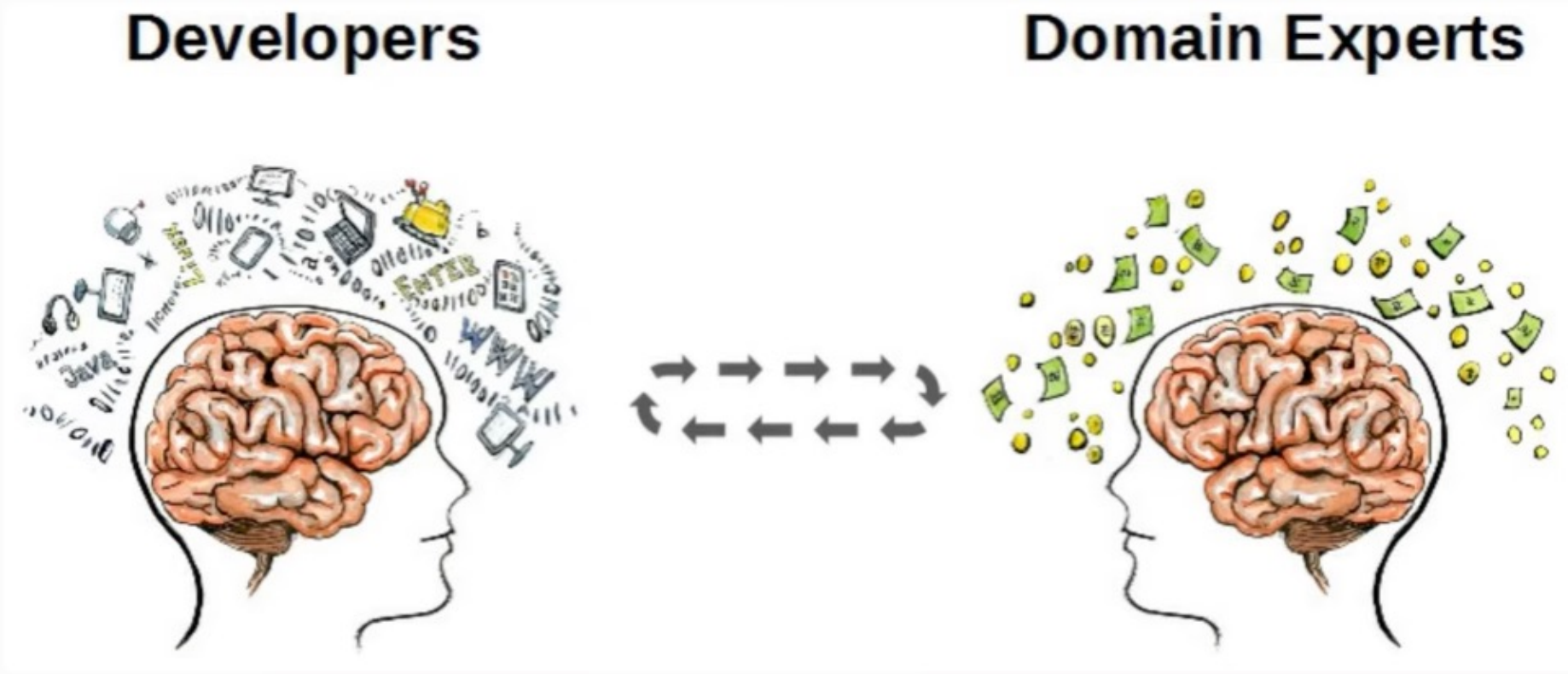


https://metropolismag.com/projects/deborah-berke-on-buckminster-fullers-dymaxion-map/





ภาพจากหนังสือ Domain Driven Design Distilled - Vaughn Vernon



ภาพจากหนังสือ Domain Driven Design Distilled - Vaughn Vernon

```python
def calculate(data):
  cart = data.cart
  if cart.get_total() < 3000:
    cost = 100

    if data.option == 0:
      for i in cart.get_items():
        if i.get_cat() == 'A':
          cost += 84
        if i.get_cat() == 'B':
          cost += i.get_weight()/1000 * 84

    elif data.option == 1:
      for i in cart.get_items():
        if i.get_cat() == 'A':
          cost += 100
        if i.get_cat() == 'B':
          cost += i.get_weight()/1000 * 84

    return cost

  else:
    return 0
```

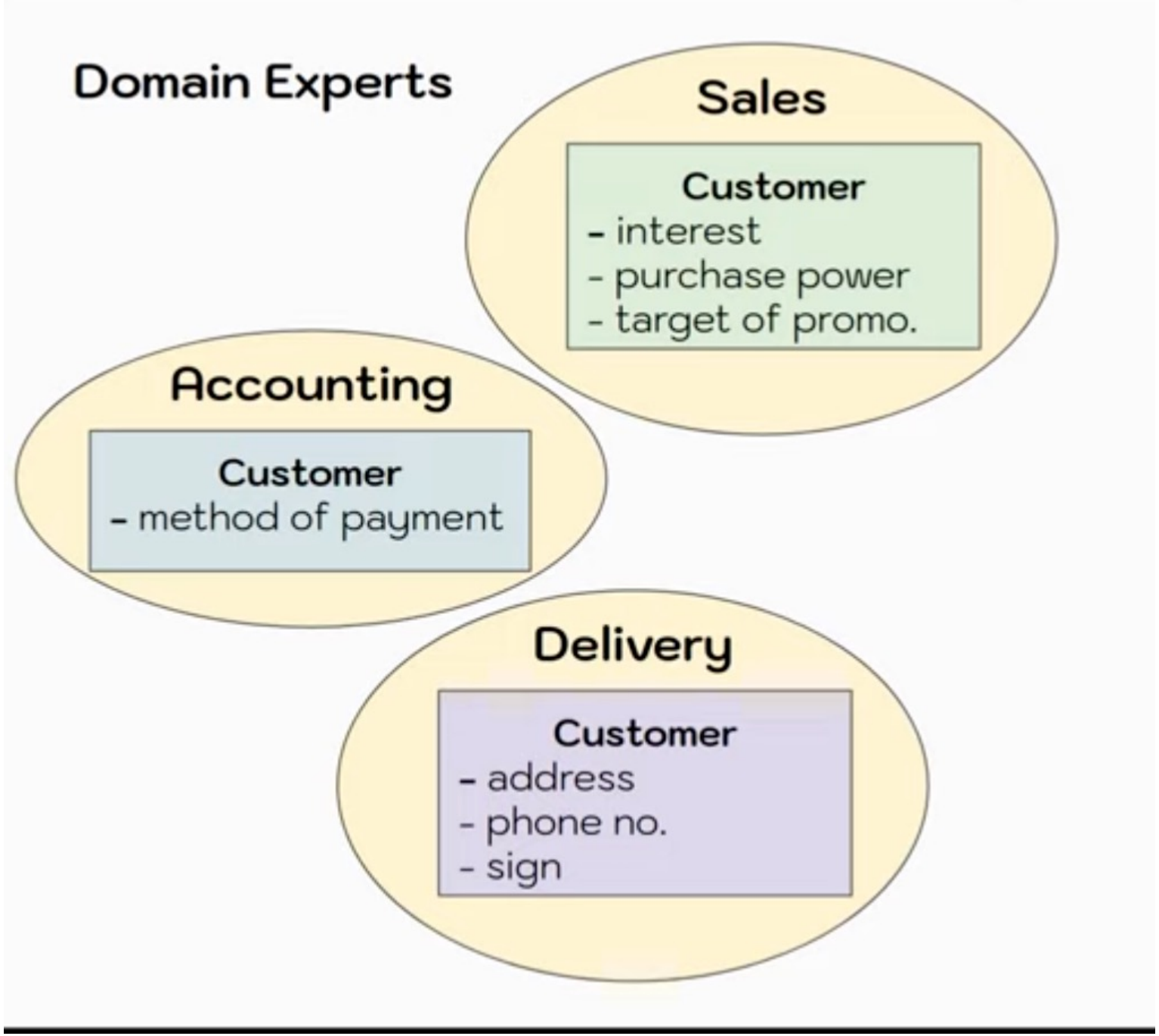
Before

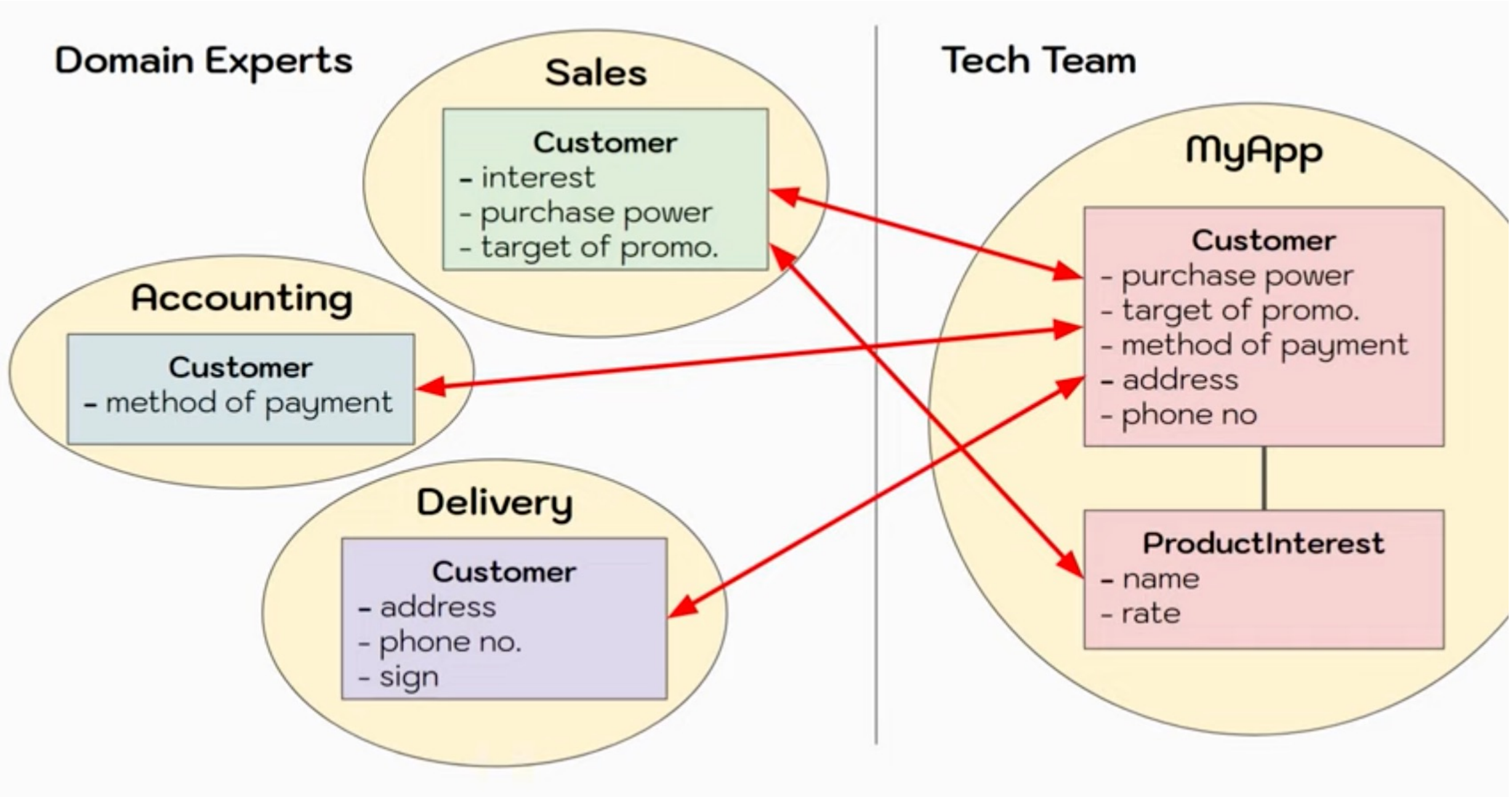
ไม่ดียังไง ดู caption เลย

```python
def calculate_shipping_price(cart, shipping_option):
  if cart.get_total_price() < FREE_SHIPPING_THRESHOLD:
    shipping_cost = BASE_SHIPPING_COST

    for item in cart.get_items():

      if shipping_option == STANDARD:
        shipping_cost += StandardShipping.get_item_cost(item)

      elif shipping_option == EXPRESS:
        shipping_cost += ExpressShipping.get_item_cost(item)

    return shipping_cost

  return 0
```

After

แก้อะไรไปบ้าง ดู caption เลย

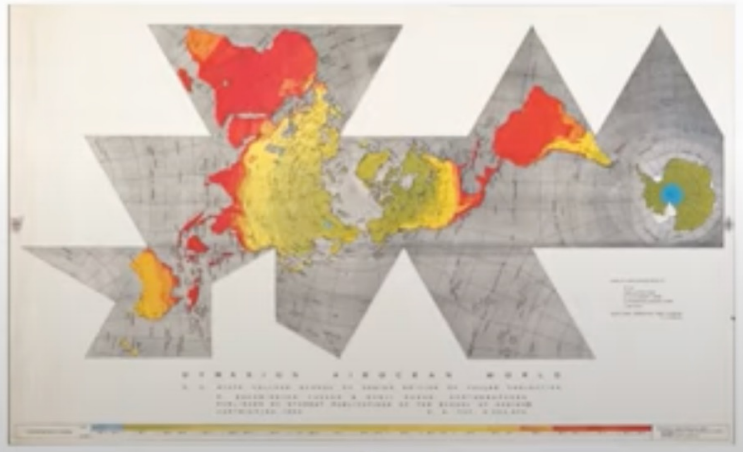

Domain Experts
Sales
Customer
- interest
- purchase power
- target of promo.
Accounting
Customer
- method of payment
Delivery
Customer
- address
- phone no.
- sign


Domain Experts
Sales
Customer
- interest
- purchase power
- target of promo.
Accounting
Customer
- method of payment
Delivery
Customer
- address
- phone no.
- sign
Tech Team
MyApp
Customer
- purchase power
- target of promo.
- method of payment
- address
- phone no
ProductInterest
- name
- rate

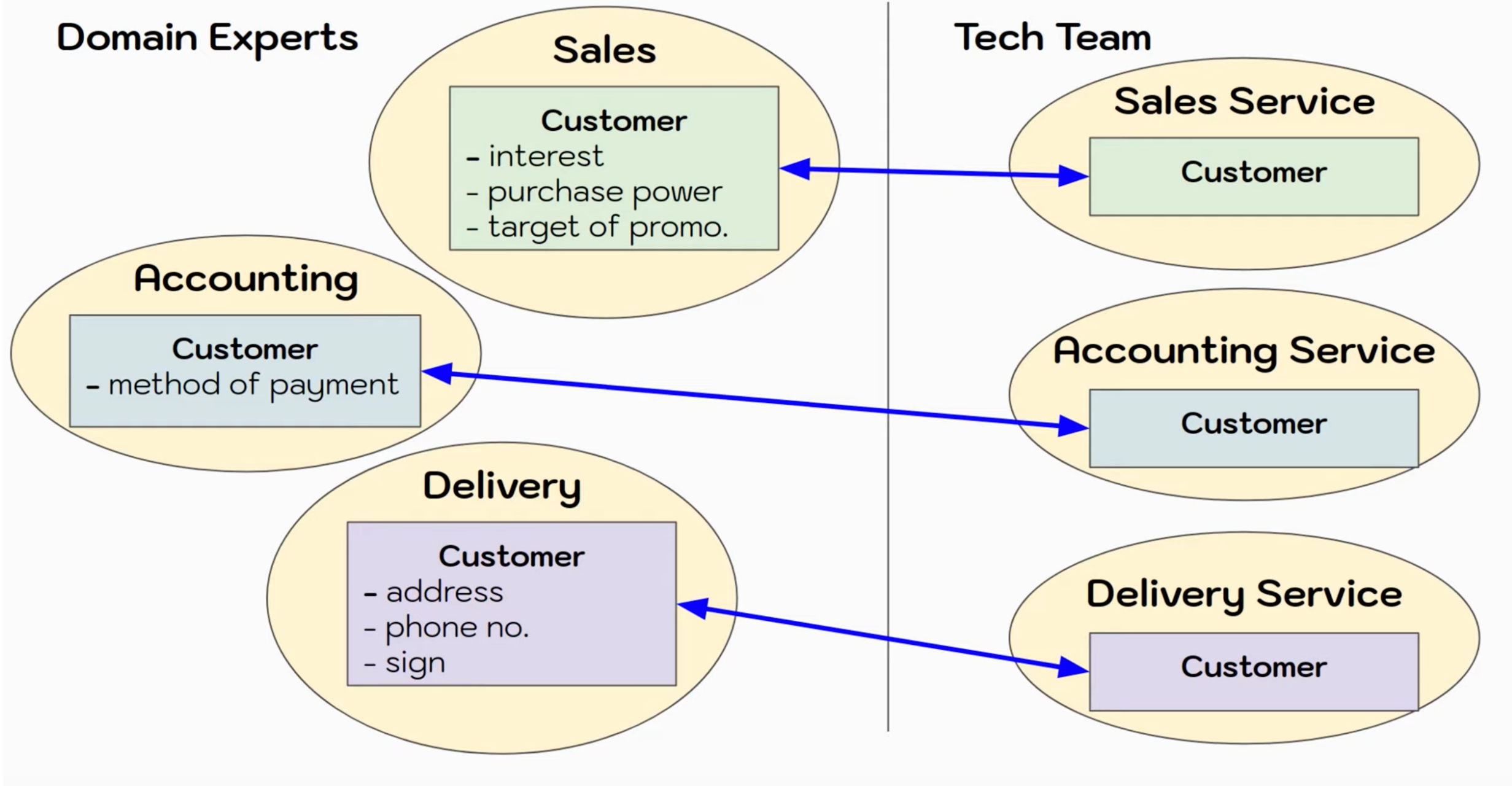
Different models are useful for different purposes.

ใช้ทำอะไรไม่ได้

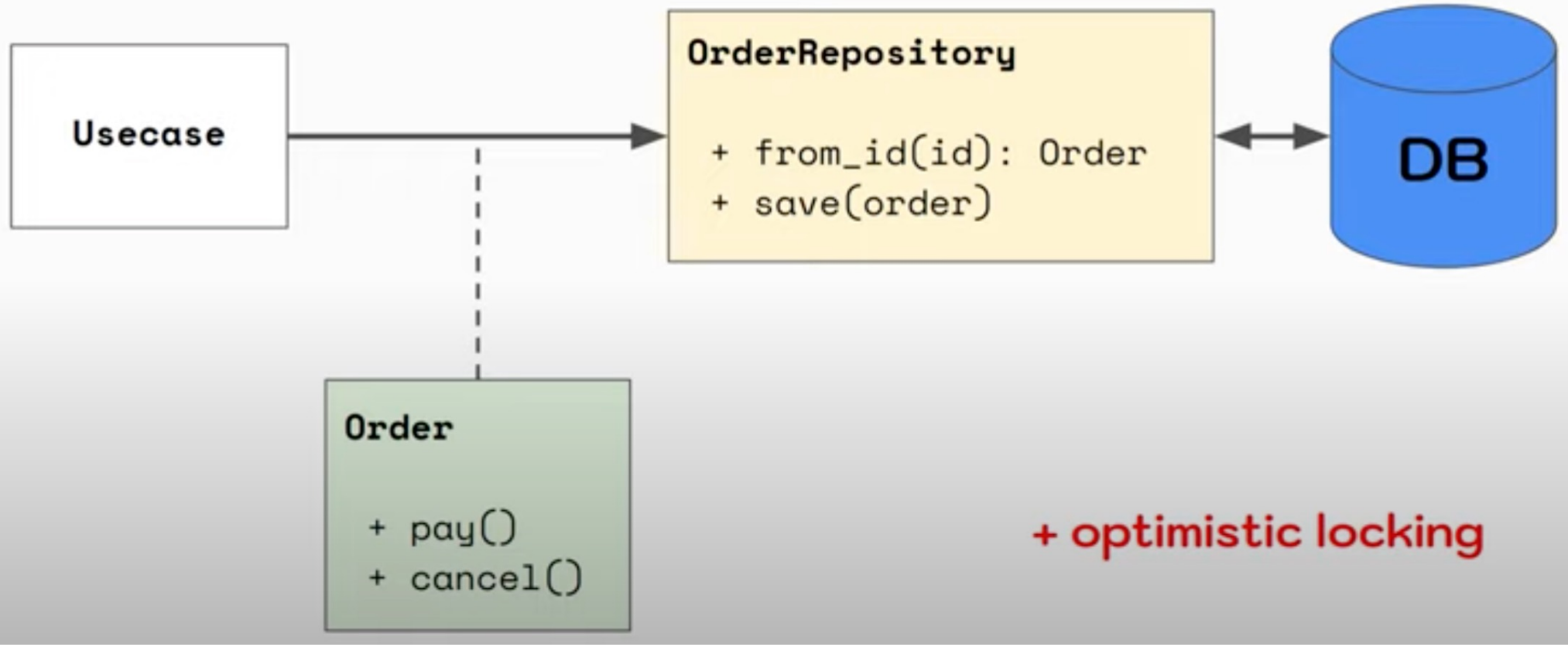
ใช้นำทางได้



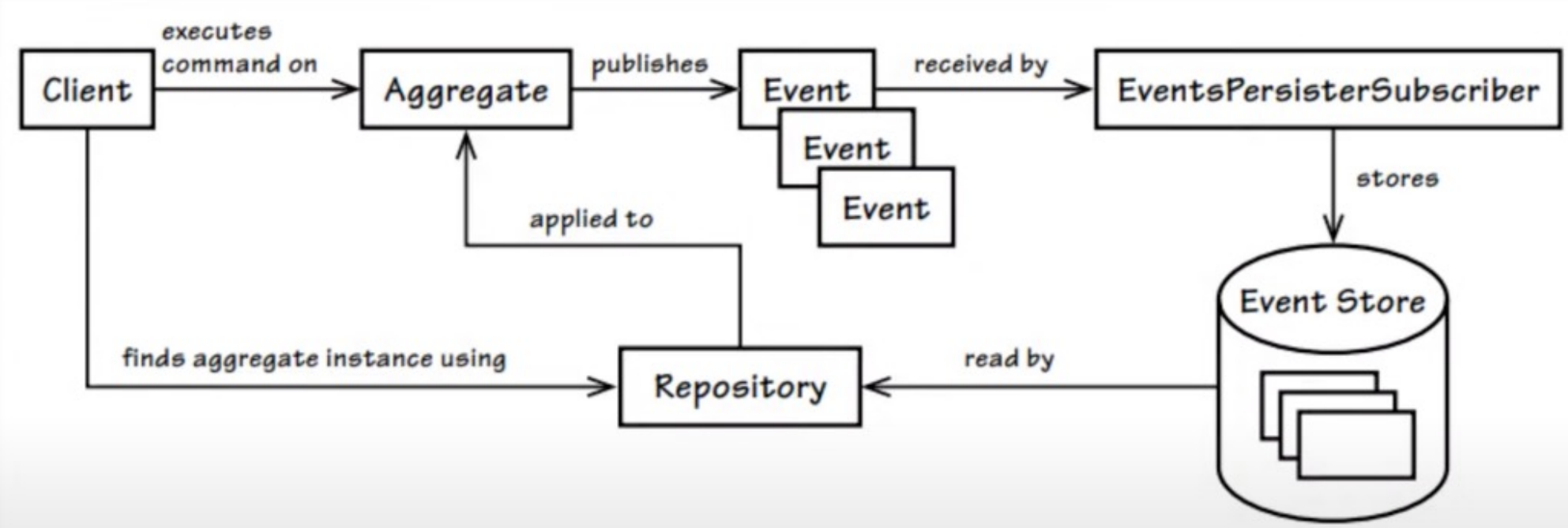
ใช้หาพื้นที่ได้


Domain Experts
Tech Team
Sales
Customer
- interest
- purchase power
- target of promo.
Sales Service
Customer
Accounting
Customer
- method of payment
Accounting Service
Customer
Delivery
Customer
- address
- phone no.
- sign
Delivery Service
Customer

Repository Pattern

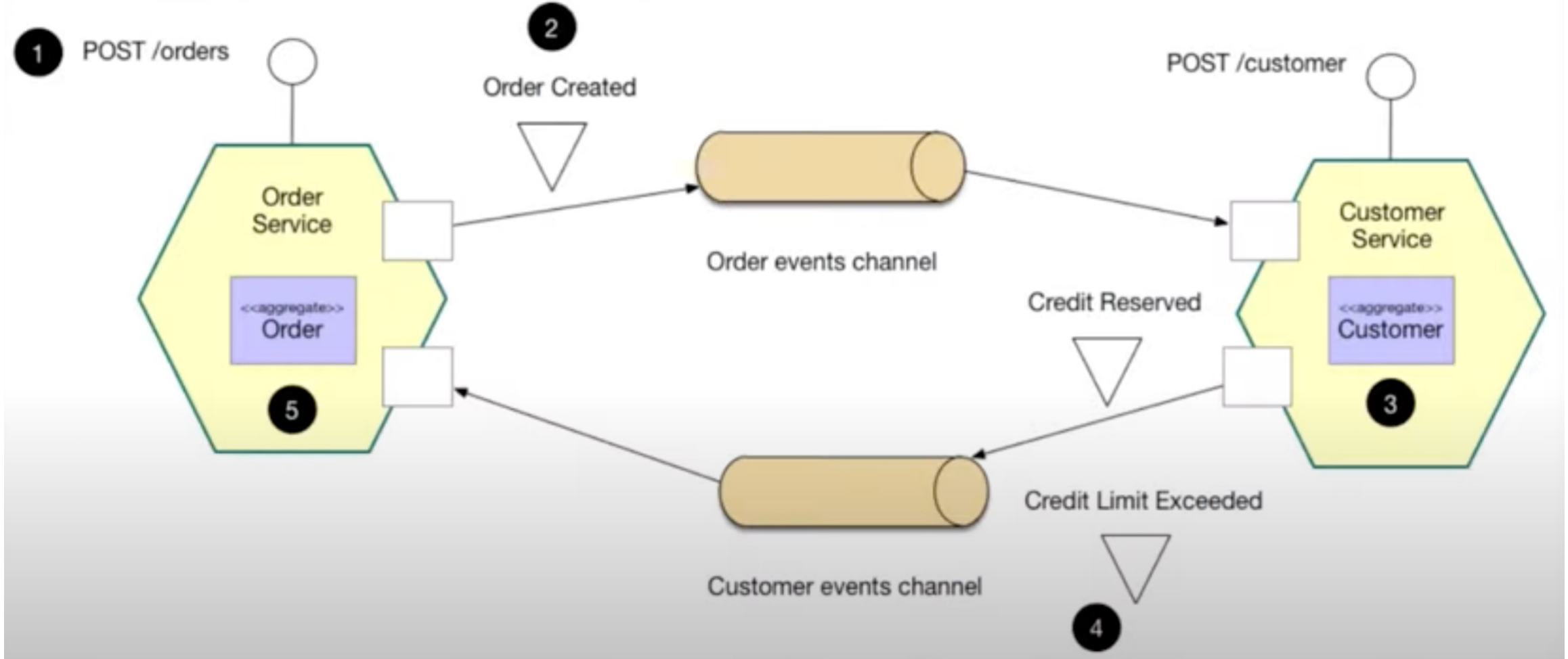


Event Sourcing

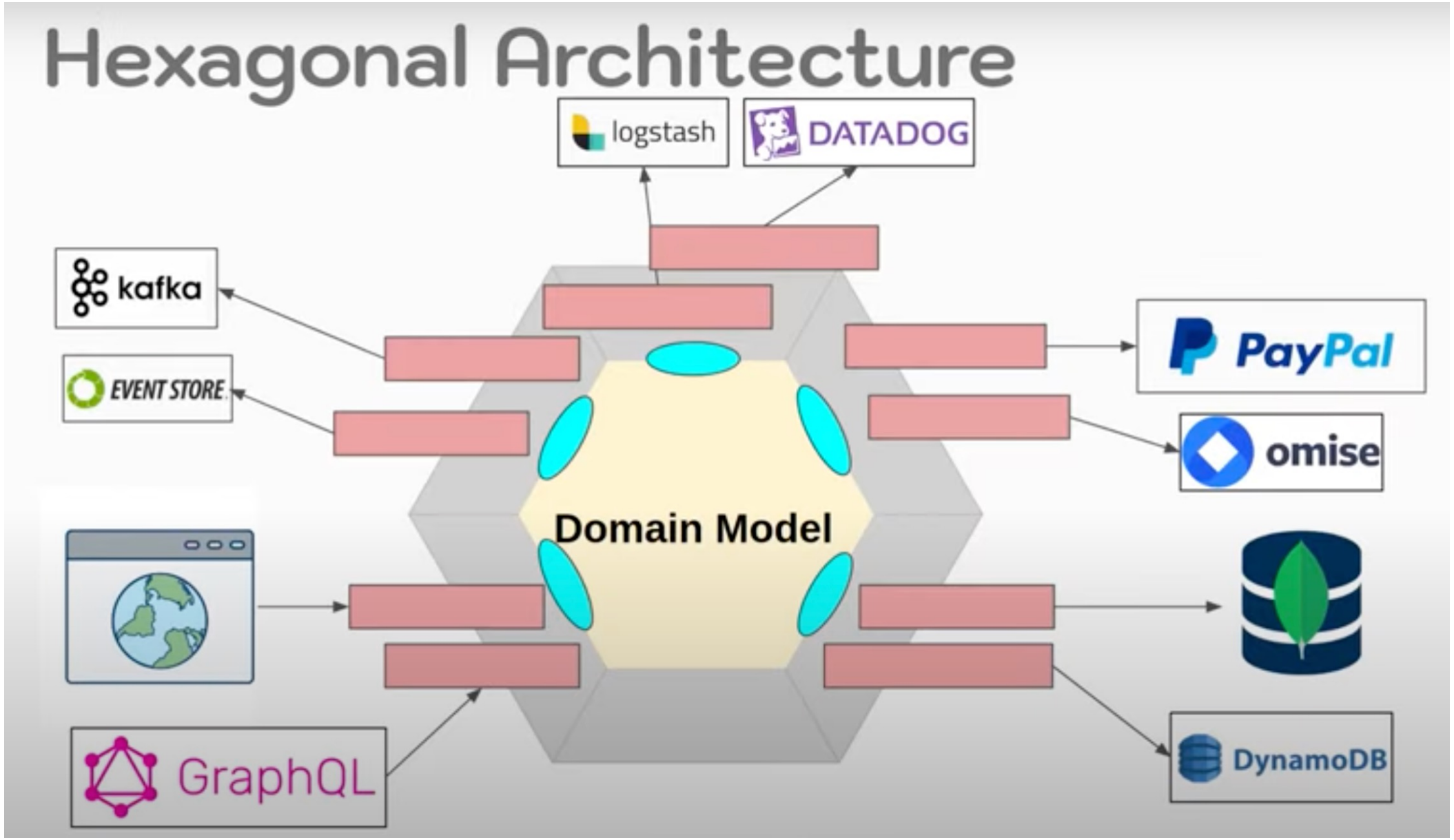


ภาพจากหนังสือ Implementing Domain Driven Design - Vaughn Vernon

Saga Pattern

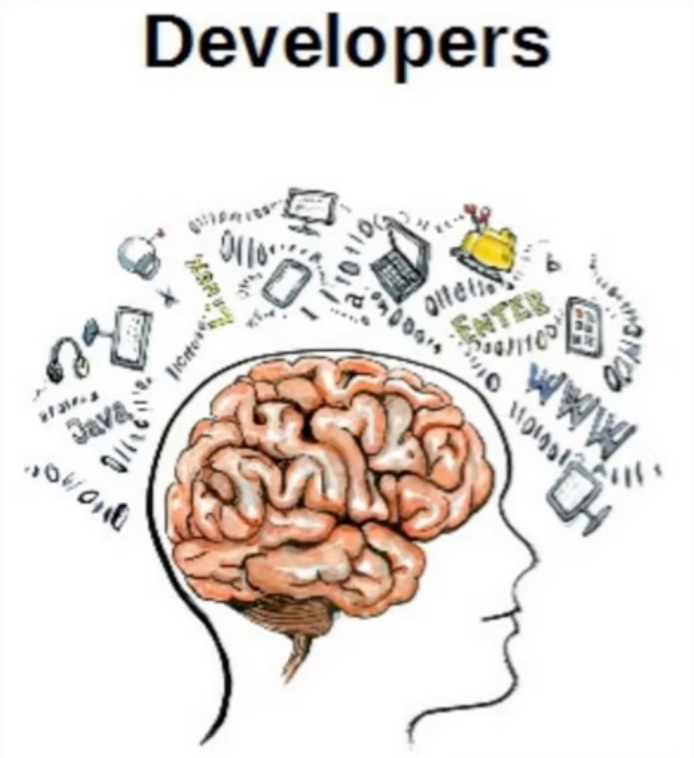


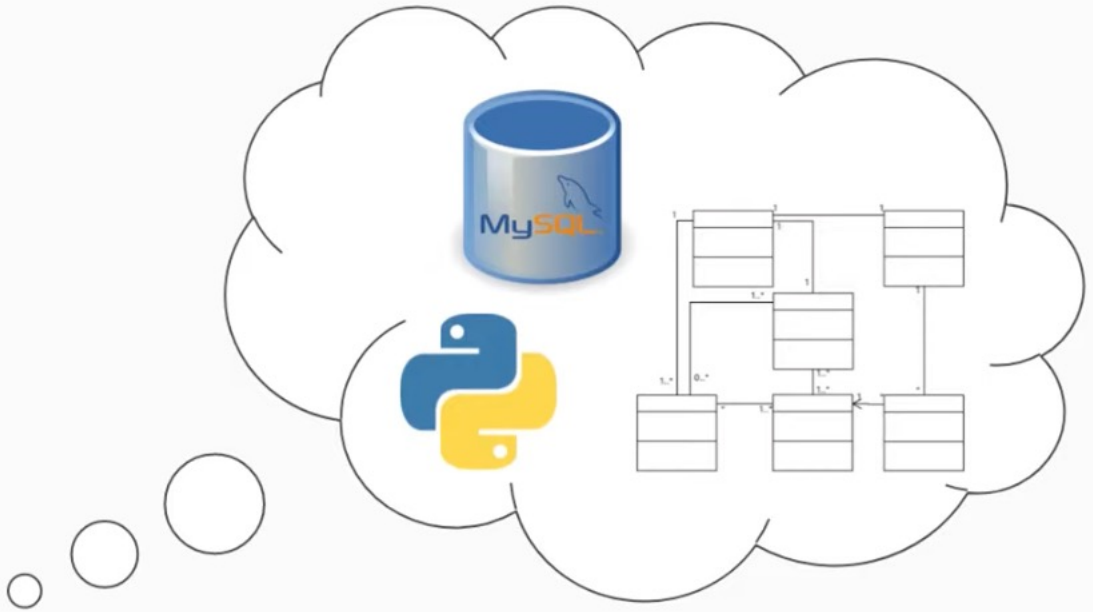
https://microservices.io/patterns/data/saga.html


Hexagonal Architecture
logstash
DATADOG
kafka
EVENT STORE
PayPal
omise
Domain Model
GraphQL
DynamoDB

Microservices with Domain-Driven Design #2/3

Credit – copy paste engineer





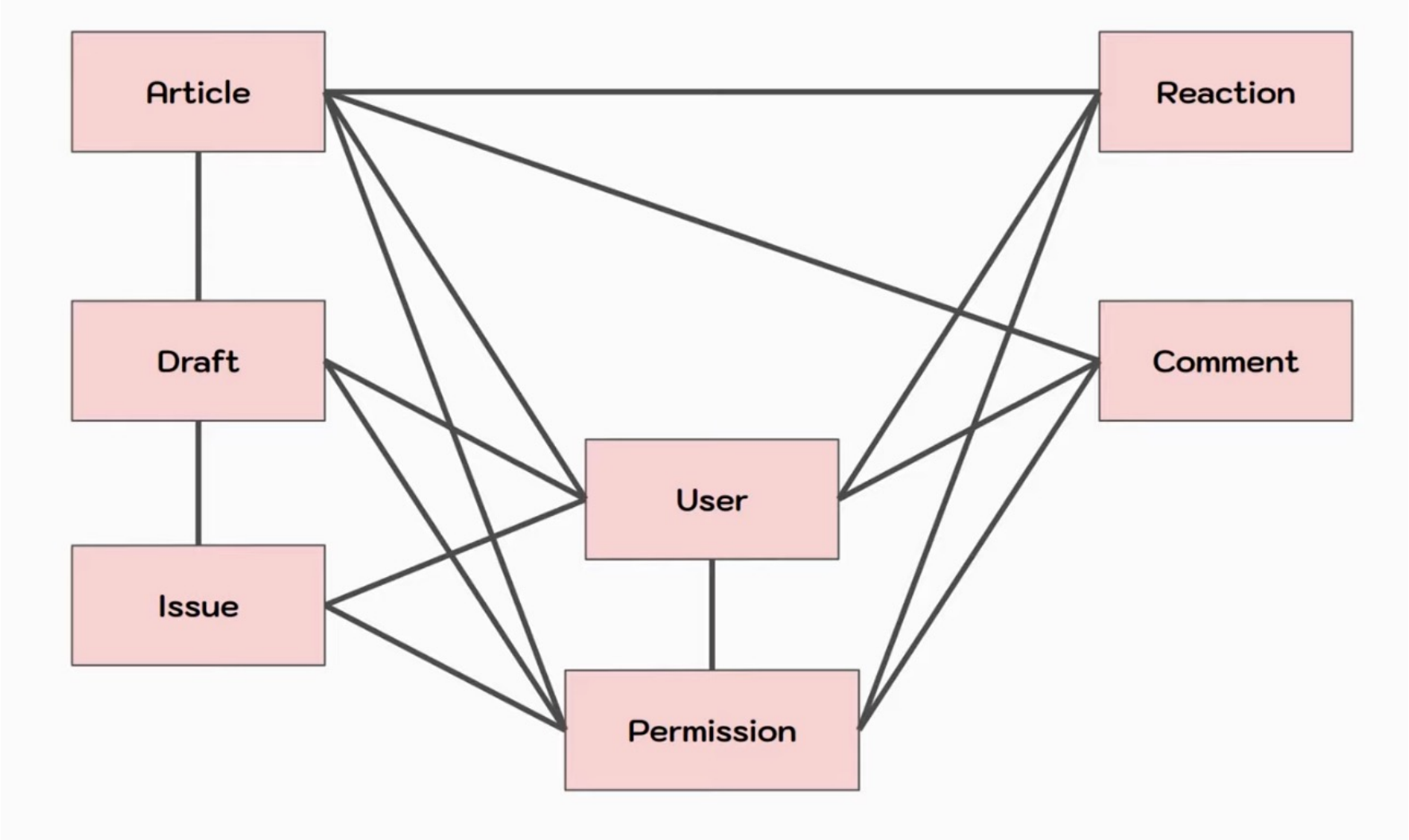
ภาพจากหนังสือ Domain Driven Design Distilled - Vaughn Vernon


Medium

Requirements

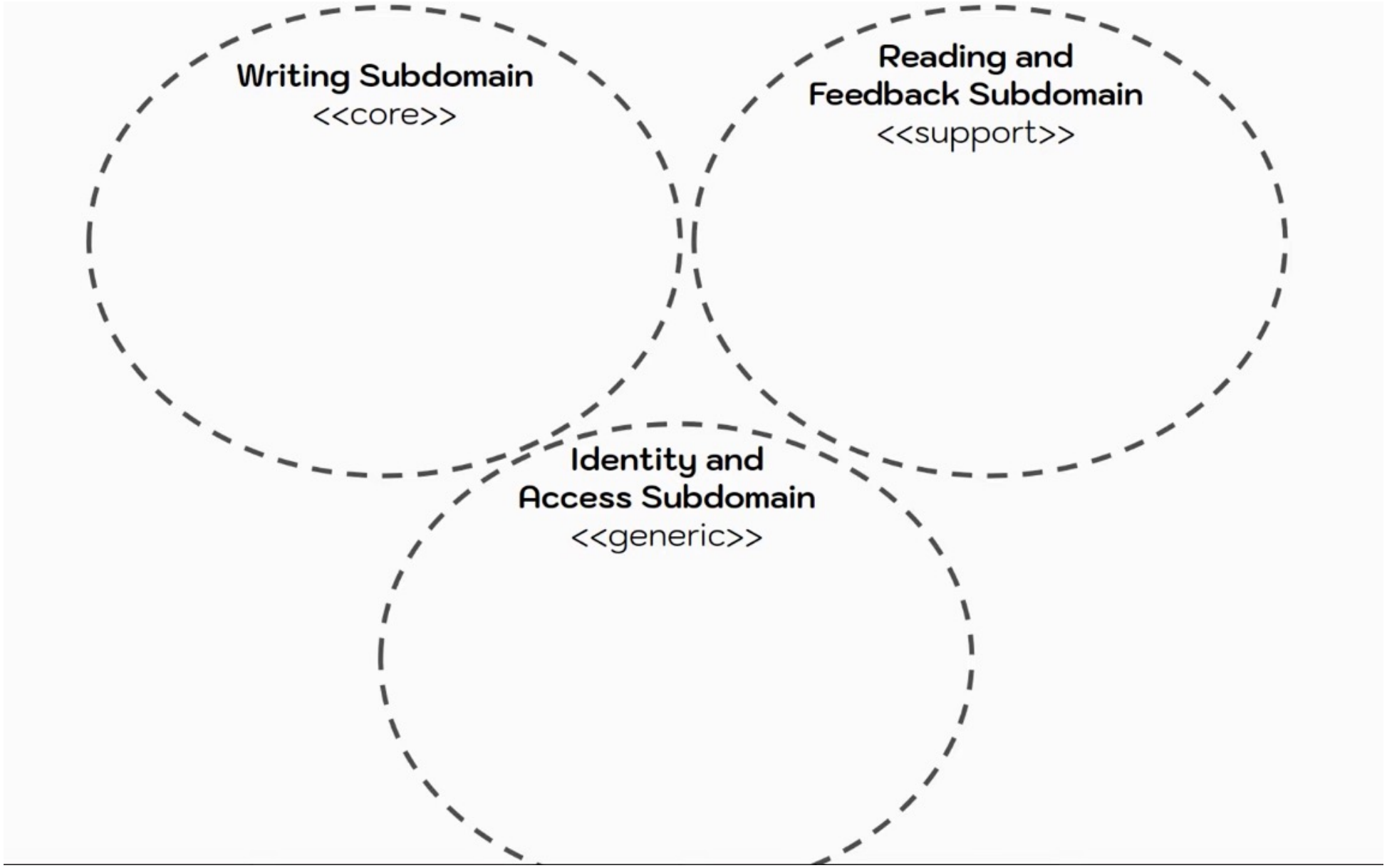
1. Writer สามารถ *สร้างและแก้ไข* Draft
2. Writer สามารถ *เพิ่ม* Reviewer
3. Reviewer สามารถ *แก้ไข* Draft
4. Reviewer สามารถ *แจ้ง* Issue ของ Draft ให้ Writer ทราบ
5. Writer สามารถ *publish* Article จาก Draft
6. Reader สามารถ *อ่าน* Article
7. Reader สามารถ *เพิ่ม* Reaction/Comment ให้ Article

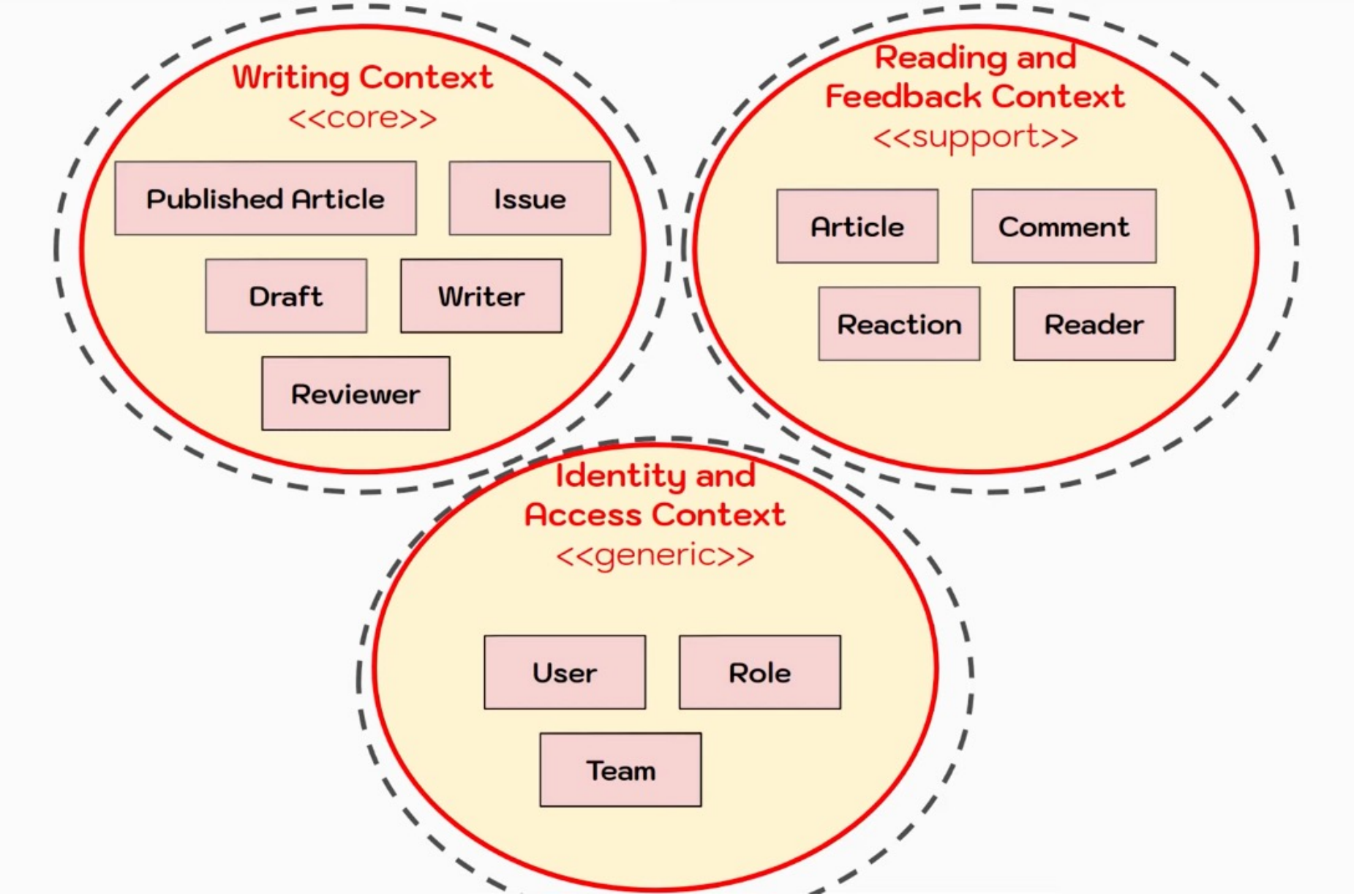
Wrong Models? โมเดลไม่เหมาะสมกับงาน
(โมเดลข้างล่าง มี coupling สูง ทำให้แบ่งงานยาก)

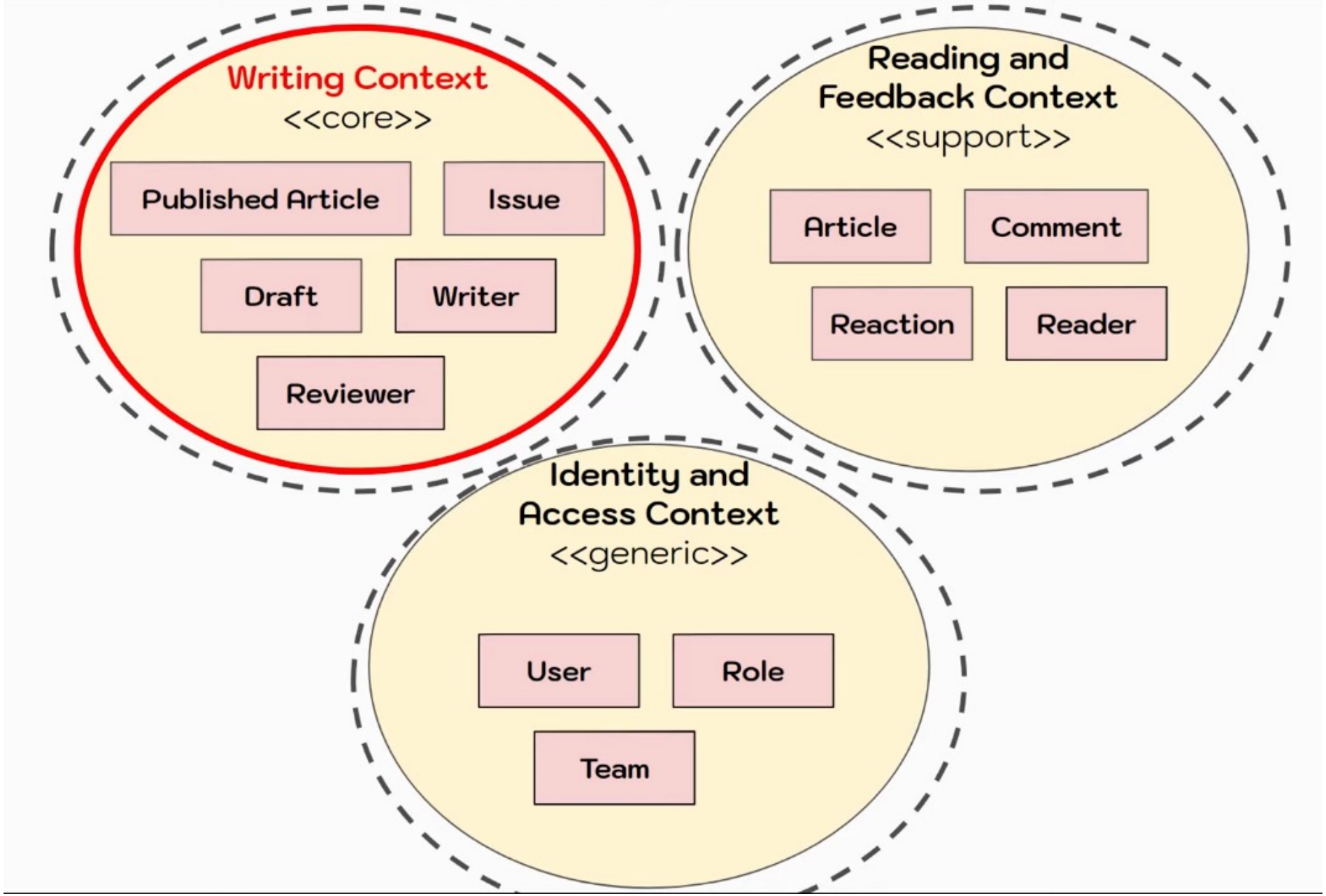


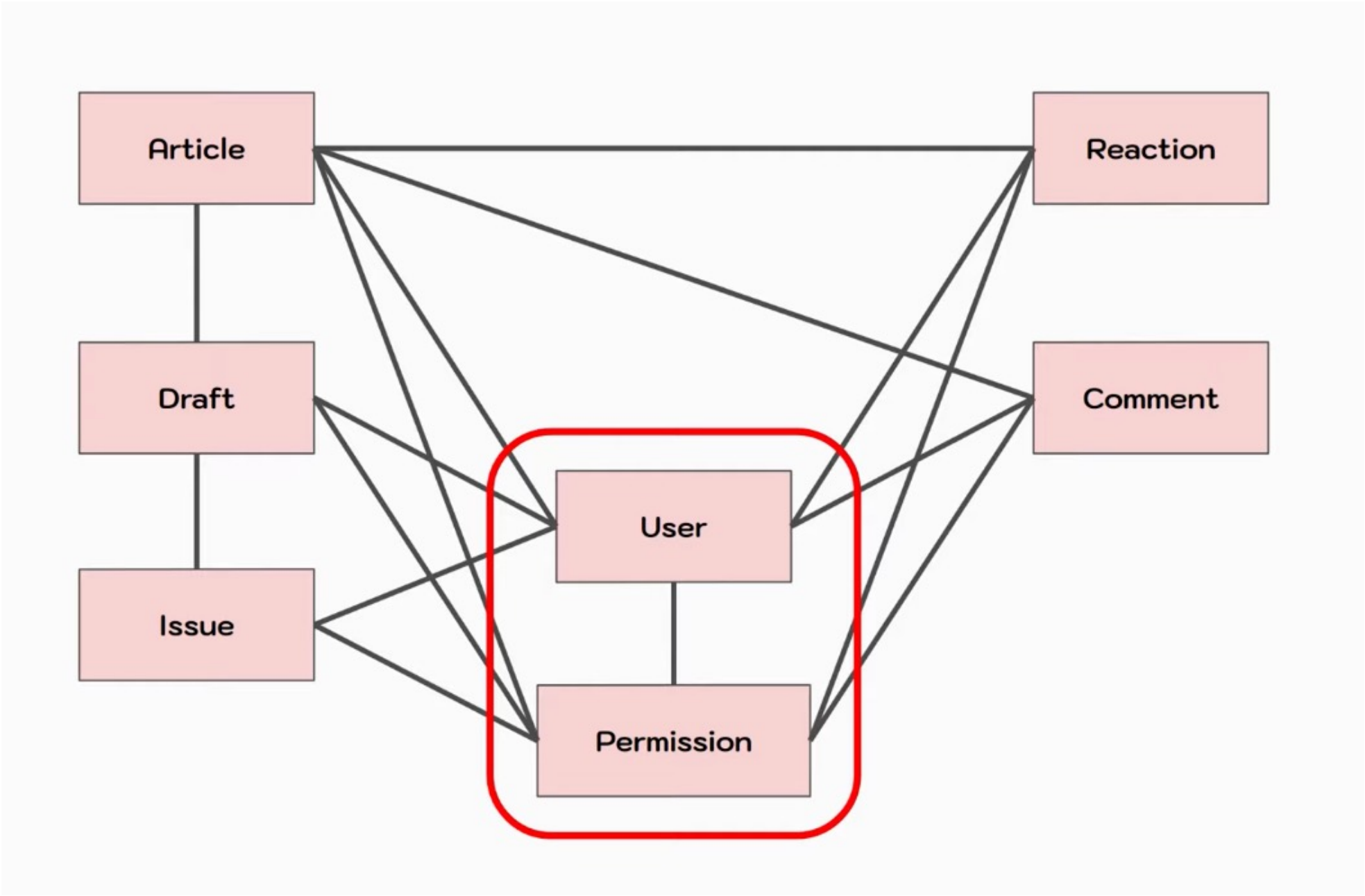
Domain-Driven Design (DDD)

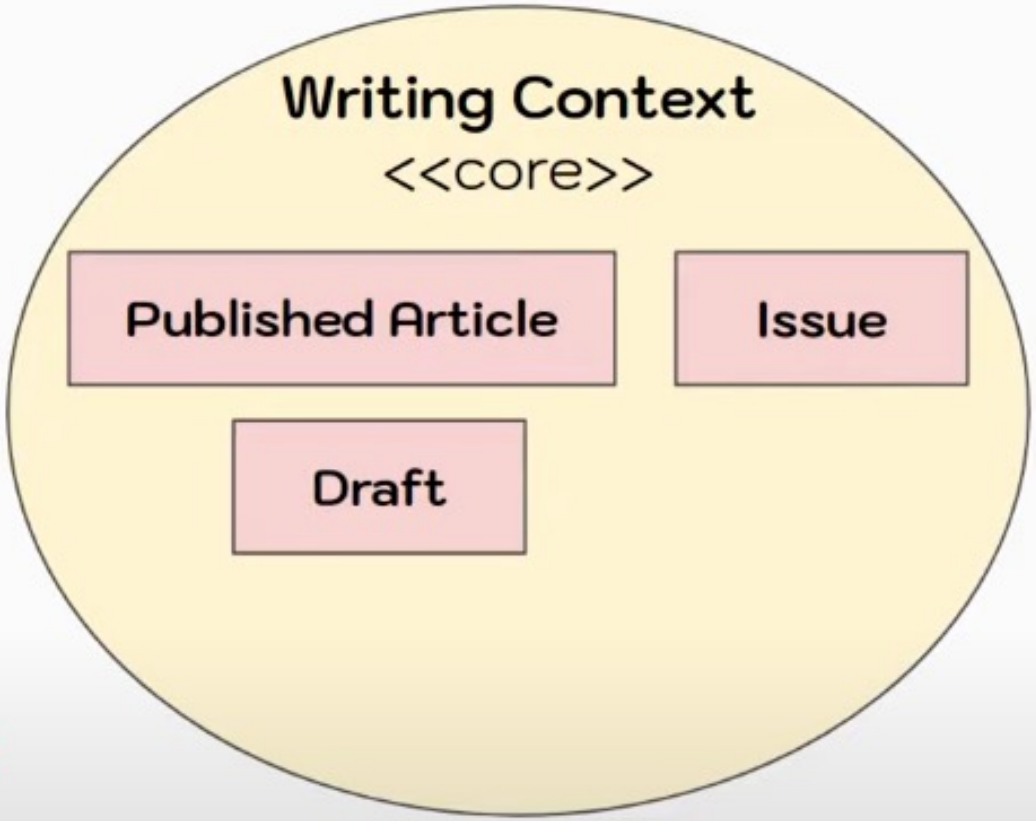
- Subdomain
- Bounded Context
- Ubiquitious Langauage


Writing Subdomain
<<core>>
Reading and
Feedback Subdomain
<<support>>
Identity and
Access Subdomain
<<generic>>


Writing Context
<<core>>
Published Article
Issue
Draft
Writer
Reviewer
Reading and
Feedback Context
<<support>>
Article
Comment
Reaction
Reader
Identity and
Access Context
<<generic>>
User
Role
Team


Writing Context
<<core>>
Published Article
Issue
Draft
Writer
Reviewer
Reading and
Feedback Context
<<support>>
Article
Comment
Reaction
Reader
Identity and
Access Context
<<generic>>
User
Role
Team

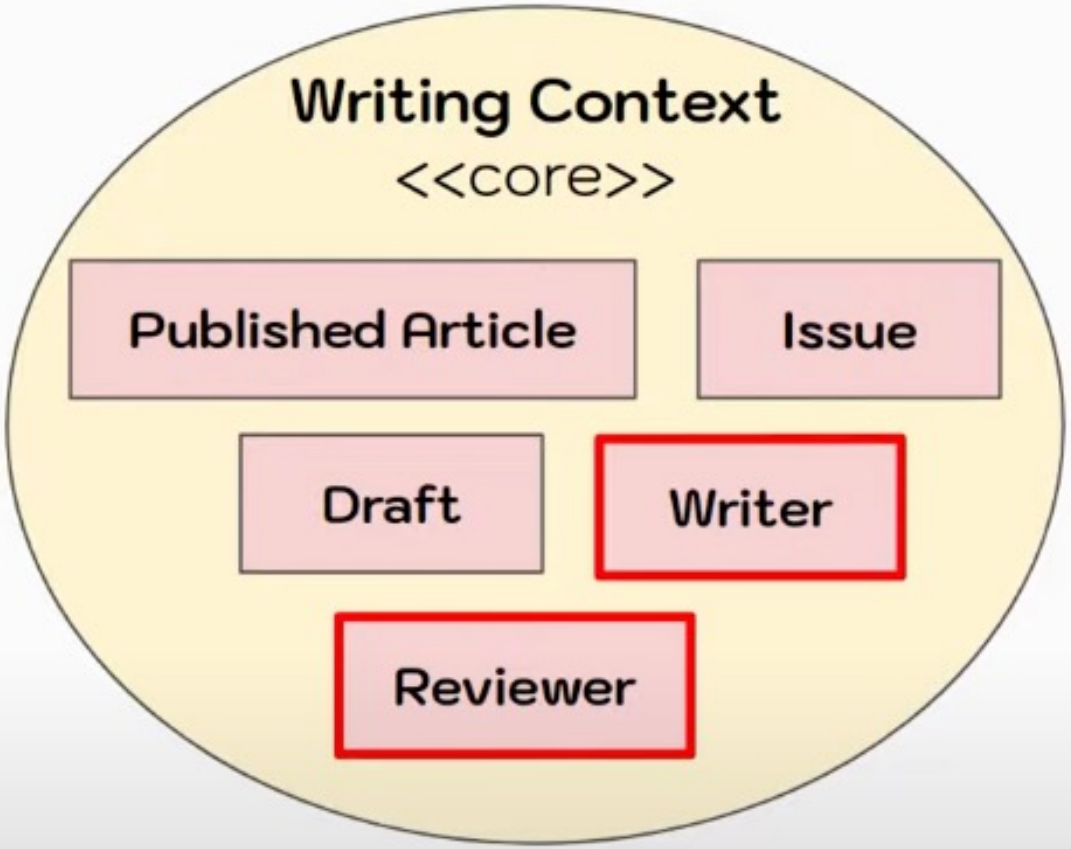

Article
Reaction
Draft
Comment
User
Issue
Permission

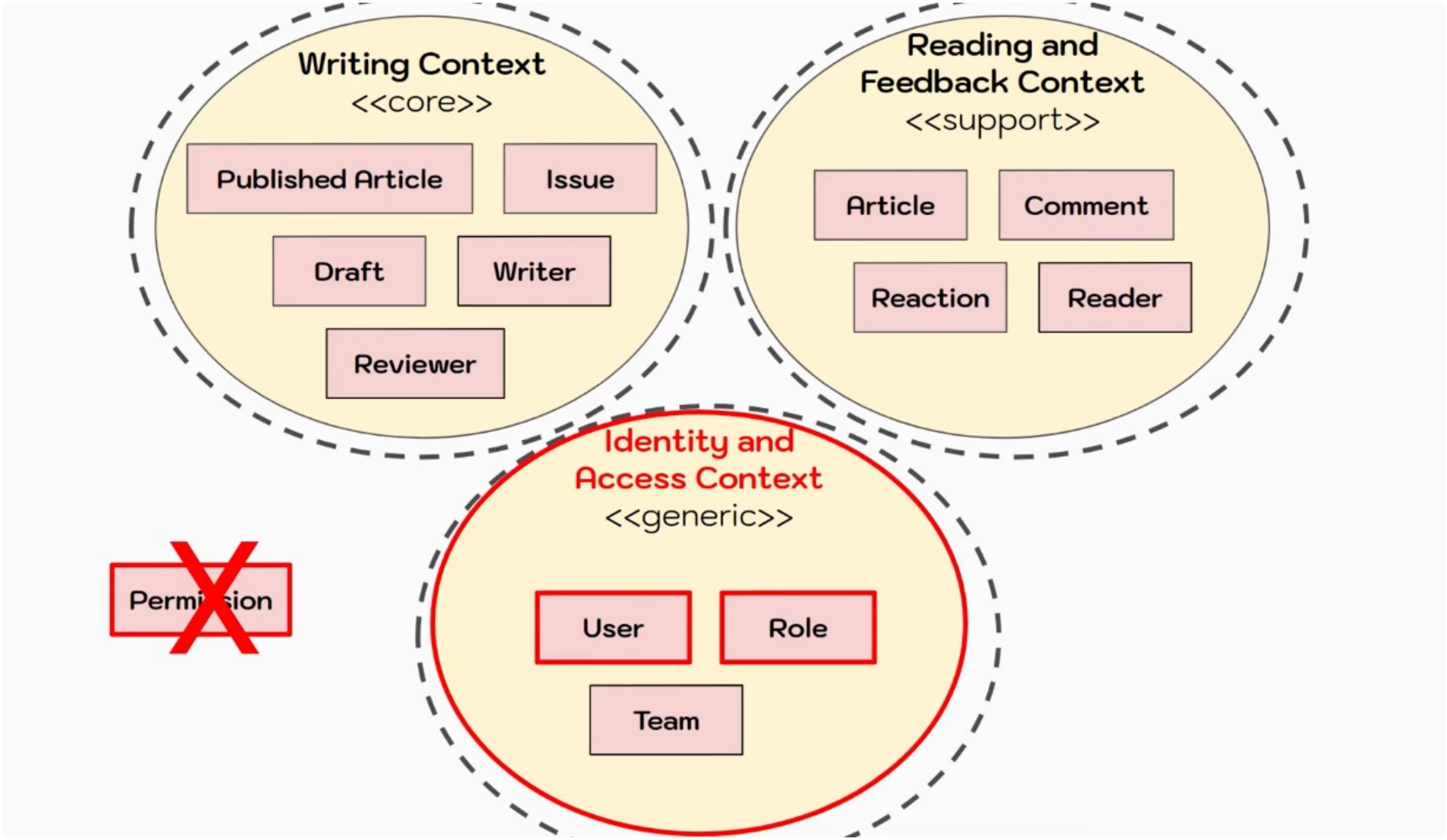


User

Permission

ไม่เกี่ยวกับ Writing Context


Writing Context
<<core>>
Published Article
Issue
Draft
Writer
Reviewer

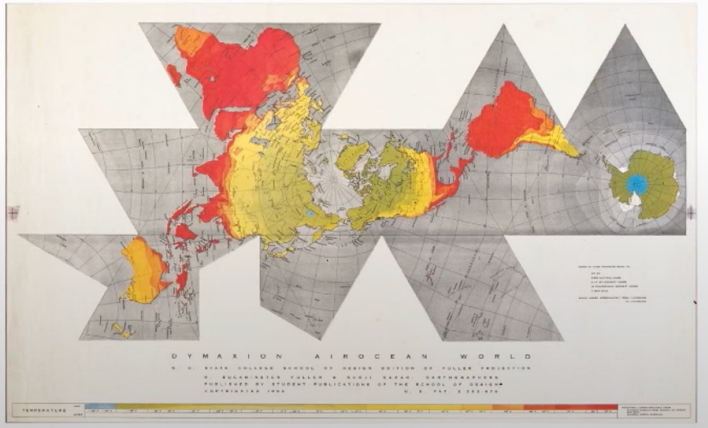

Writing Context
<<core>>
Published Article
Issue
Draft
Writer
Reviewer
Reading and
Feedback Context
<<support>>
Article
Comment
Reaction
Reader
Identity and
Access Context
<<generic>>
User
Role
Team
Permission

Models ที่เหมาะสมกับงาน
จะทำให้เราทำงานได้ง่ายขึ้น

จาก Part ที่ 1

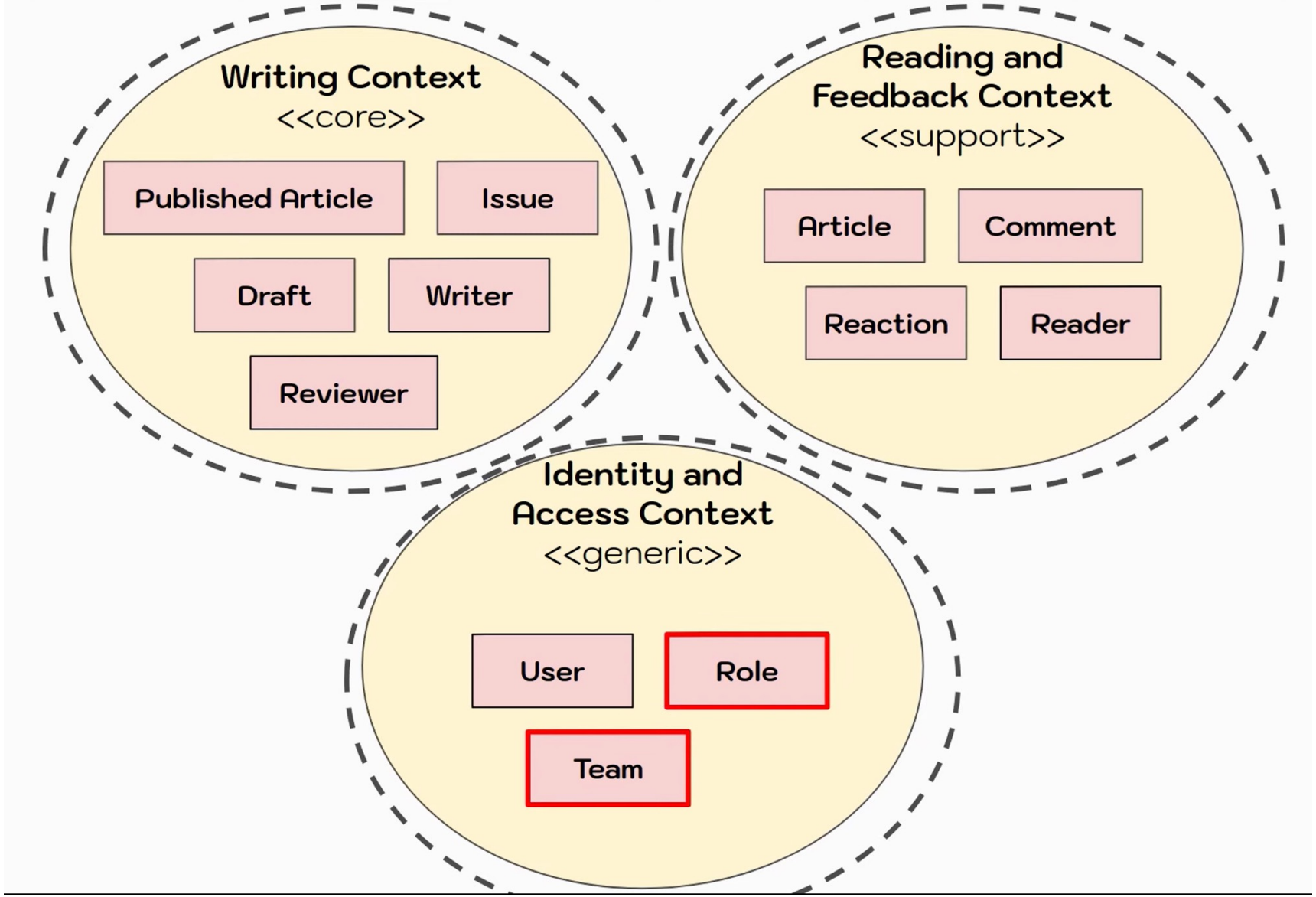
ใช้ทำอะไรไม่ได้

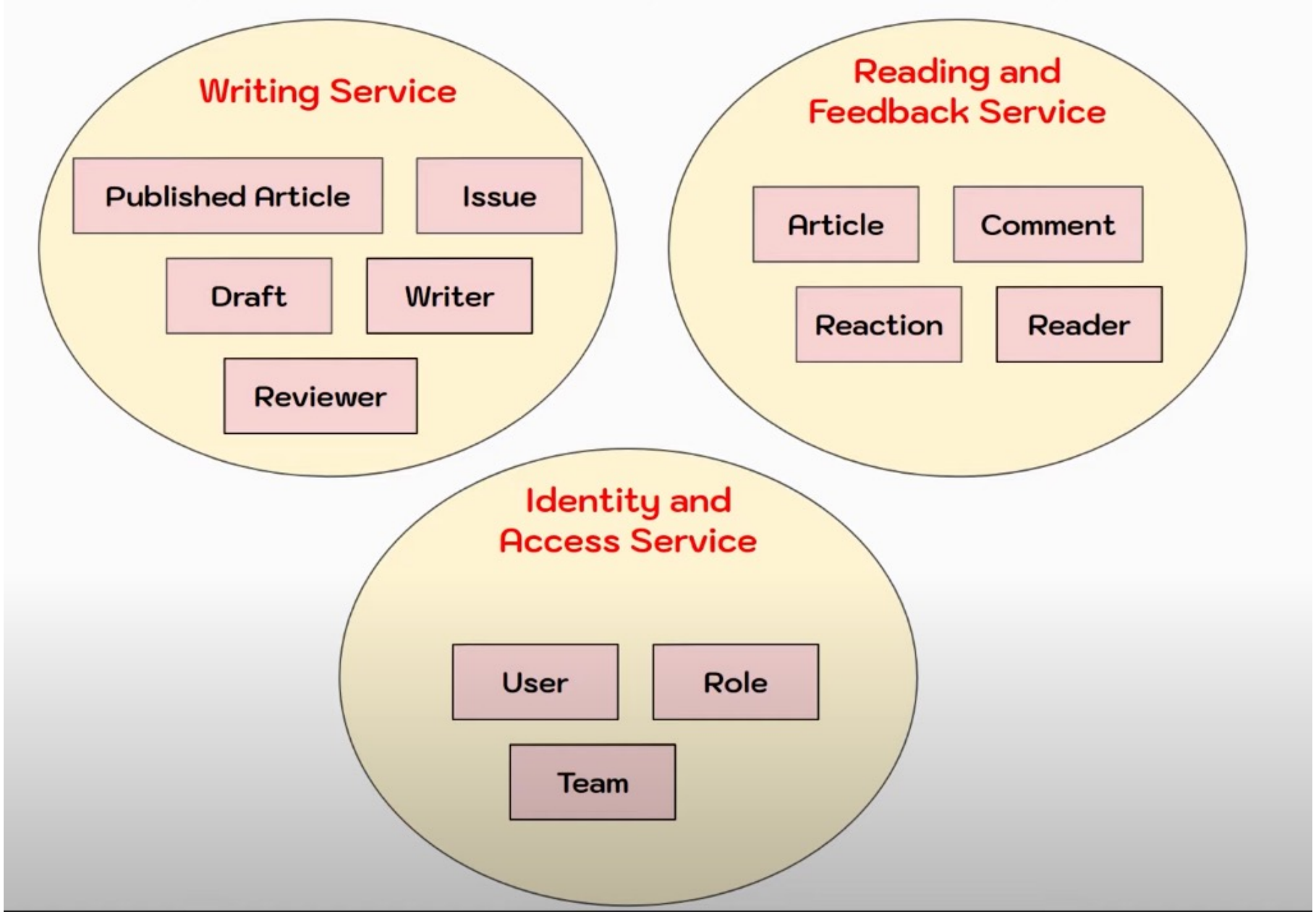
ใช้นำทางได้



ใช้หาพื้นที่ได้

การวิเคราะห์ทางภาษา




Writing Service
Published Article
Issue
Draft
Writer
Reviewer
Reading and Feedback Service
Article
Comment
Reaction
Reader
Identity and Access Service
User
Role
Team

Subdomain ?

Bounded Context ?

ภาษา ?

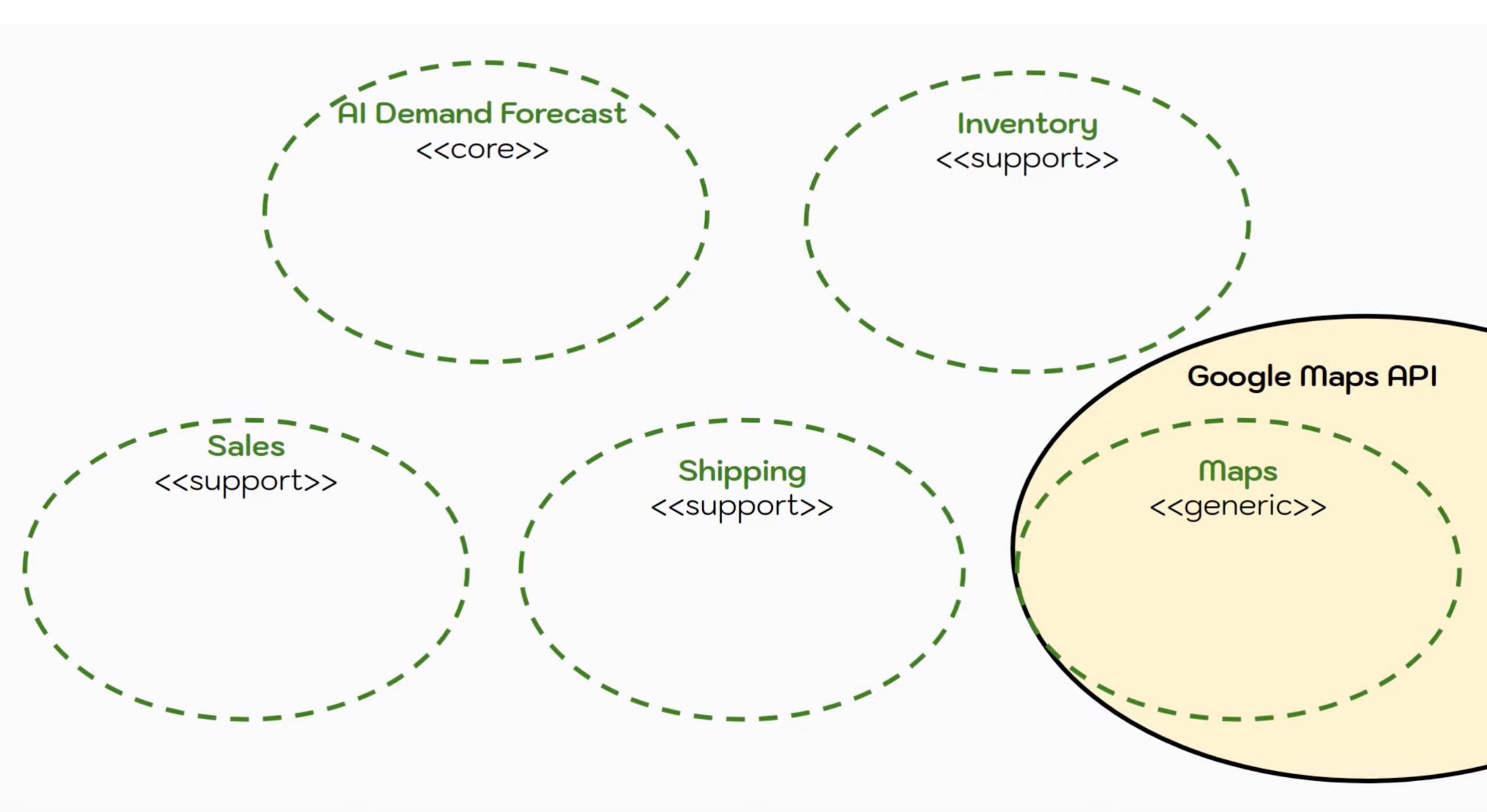
Subdomain

- ส่วนย่อยของ *โจทย์หลัก (ใหญ่+ซับซ้อน)* ในระบบ
- ใช้ในการจำกัดขอบเขตของปัญหา -> **วิเคราะห์ง่ายขึ้น**

Subdomain แบ่งเป็น 3 ประเภท

1. **Core Domain:** หัวใจของระบบ ทำให้เราเหนือกว่าคู่แข่ง
2. **Supporting Subdomain:** ต้องทำเพื่อให้ Core สมบูรณ์
3. **Generic Subdomain:** โจทย์ทั่ว ๆ ไป หาซื้อ/outsource ได้

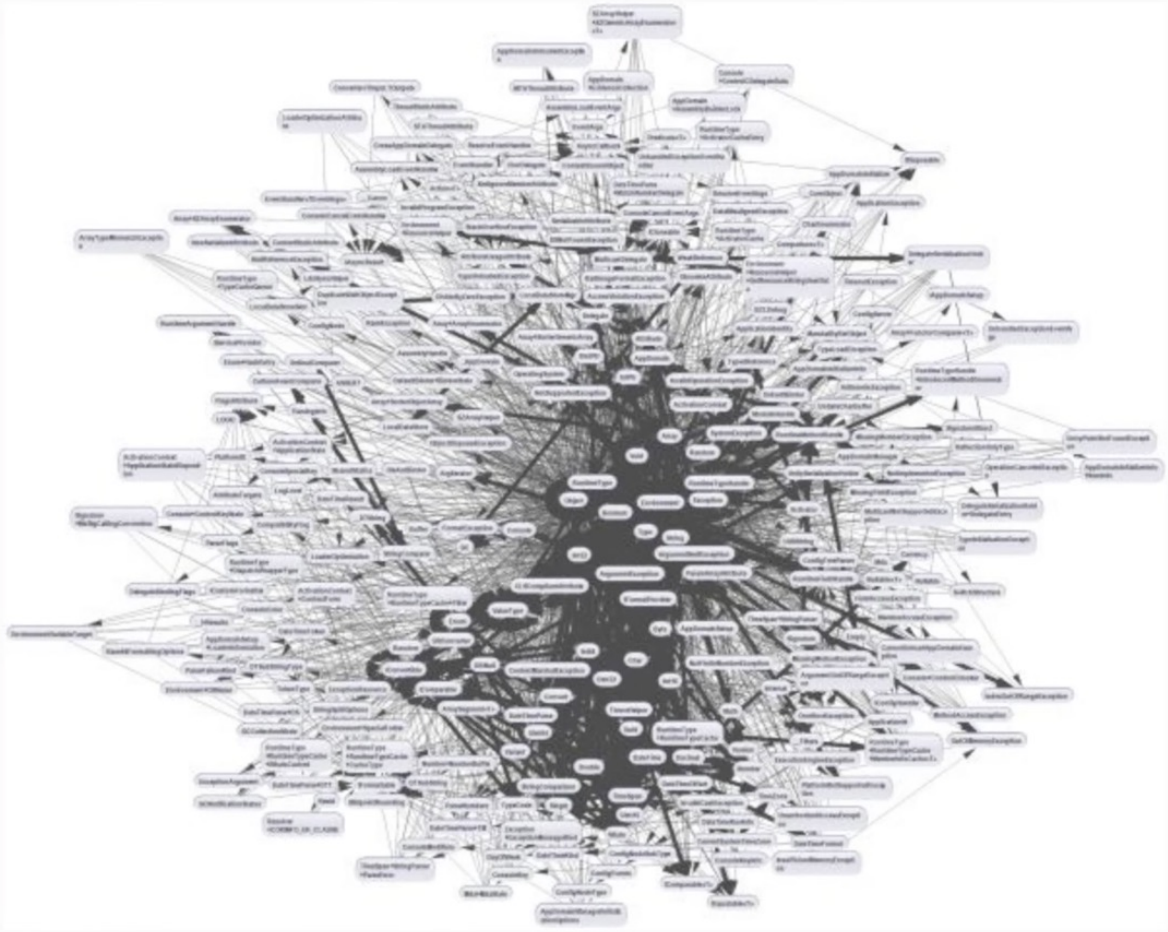
ตัวอย่าง Subdomain


AI Demand Forecast
<<core>>
Inventory
<<support>>
Google Maps API
Maps
<<generic>>
Sales
<<support>>
Shipping
<<support>>

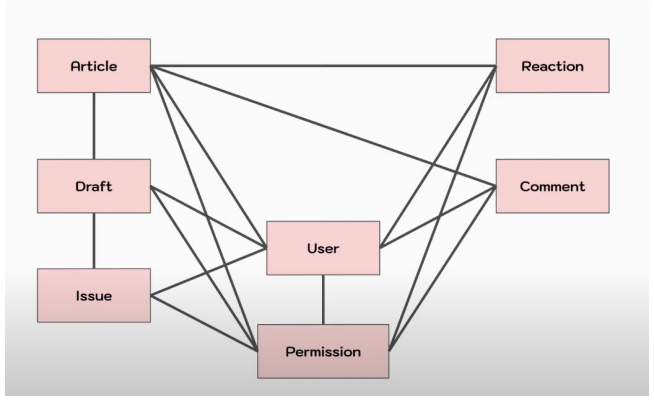
Bounded Context

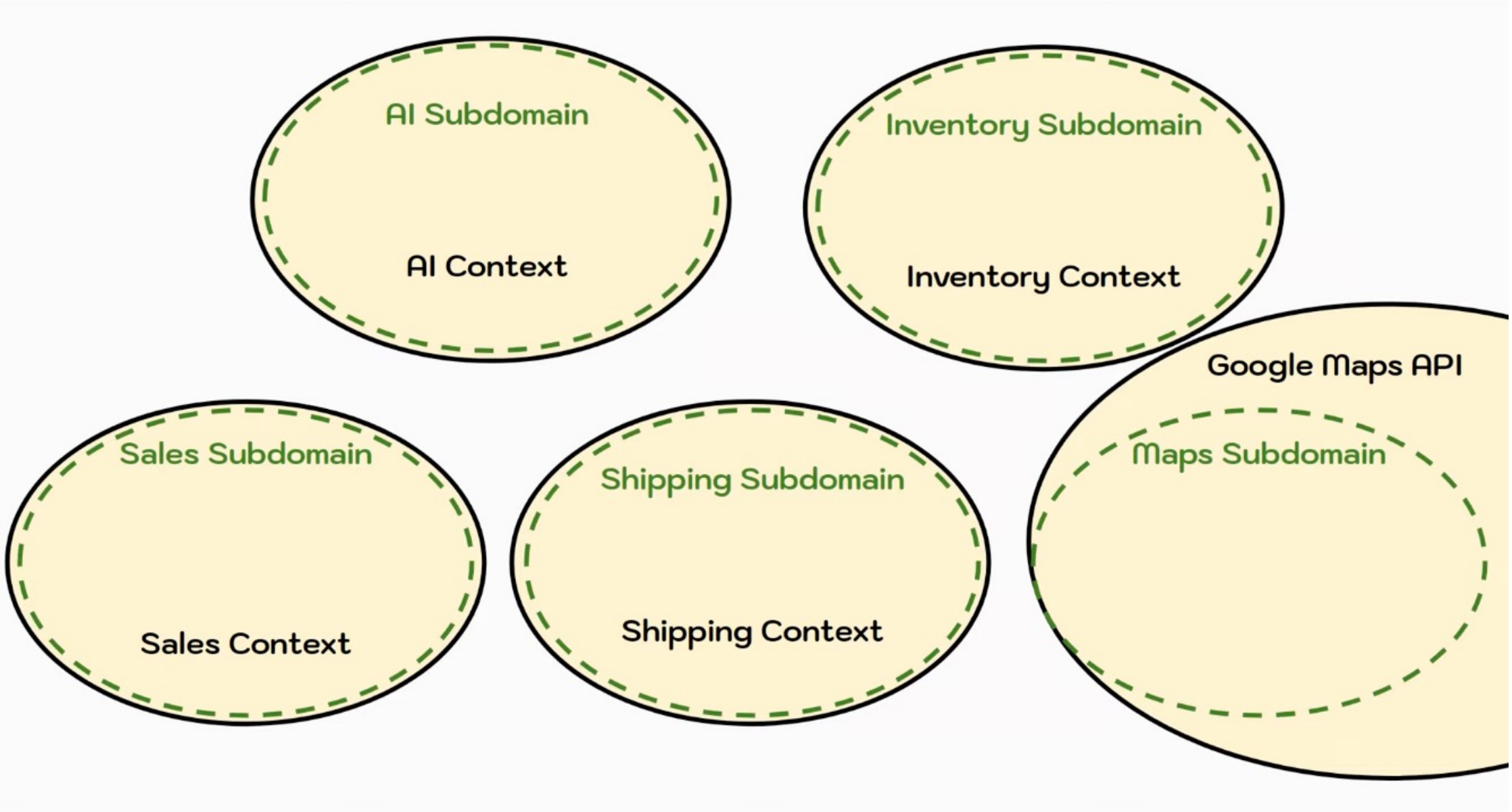
- **ขอบเขต**ของ model/concept/ภาษา
- ถ้า subdomain เป็น *ส่วนของปัญหา*

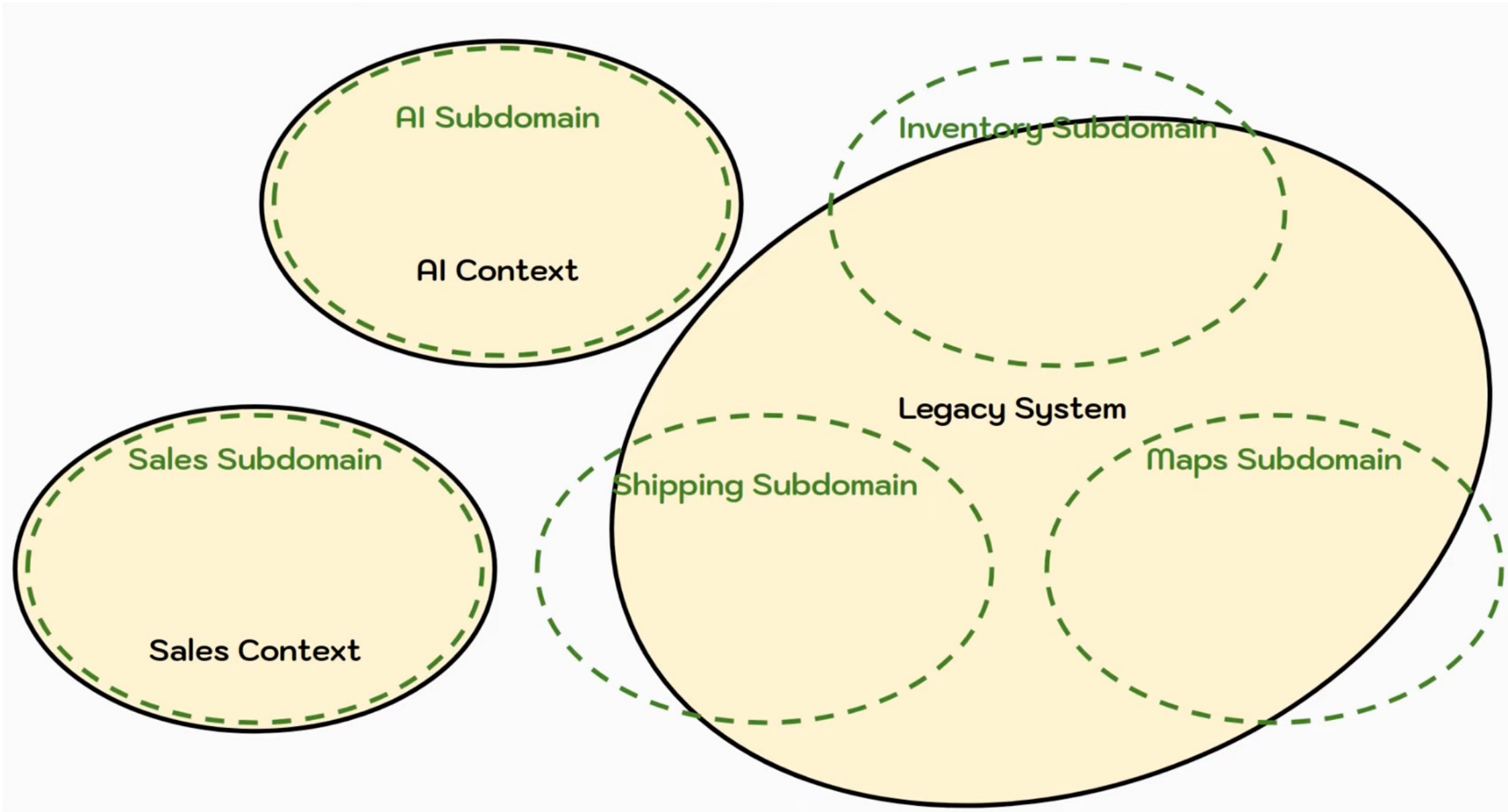
  Bounded Context เป็น ***ส่วนของคำตอบ***



Big Ball of Mud




AI Subdomain
AI Context
Inventory Subdomain
Inventory Context
Google Maps API
Maps Subdomain
Sales Subdomain
Sales Context
Shipping Subdomain
Shipping Context


AI Subdomain
AI Context
Inventory Subdomain
Legacy System
Sales Subdomain
Sales Context
Shipping Subdomain
Maps Subdomain

Ubiquitous Language

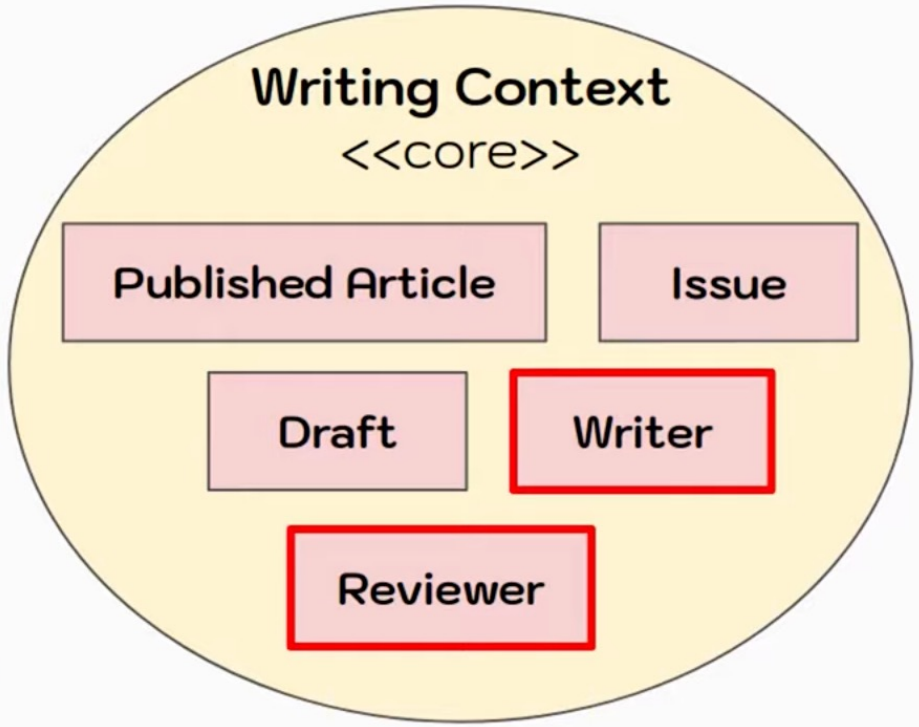
- English-Thai: NECTEC's Lexitron-2 Dictionary [with local updates]

ubiquitous	[ADJ] ซึ่งมีอยู่ทุกหนทุกแห่ง, **Syn.** omnipresent, pervasive

Ubiquitous Language

- ภาษาเฉพาะ ที่ใช้กันภายในแต่ละ Bounded Context
- ทุกคนในทีม (dev + domain experts) เข้าใจตรงกัน
- ไม่กำกวม

Ubiquitous Language



User

Permission

ไม่เกี่ยวกับ Writing Context

Vaccine X ถูกนำไปใช้	`vaccine.take_out()`
พยาบาล A ฉีด Vaccine X ให้ คนไข้ P	`nurse.vaccinate(patient, vaccine)`
คนไข้ P ได้รับ Vaccine X โดย พยาบาล A	`patient.take(vaccine, by=nurse)`

Vaccine X ถูกนำไปใช้

```
vaccine.take_out()
```

Warehouse

พยาบาล A ฉีด Vaccine X ให้ คนไข้ P

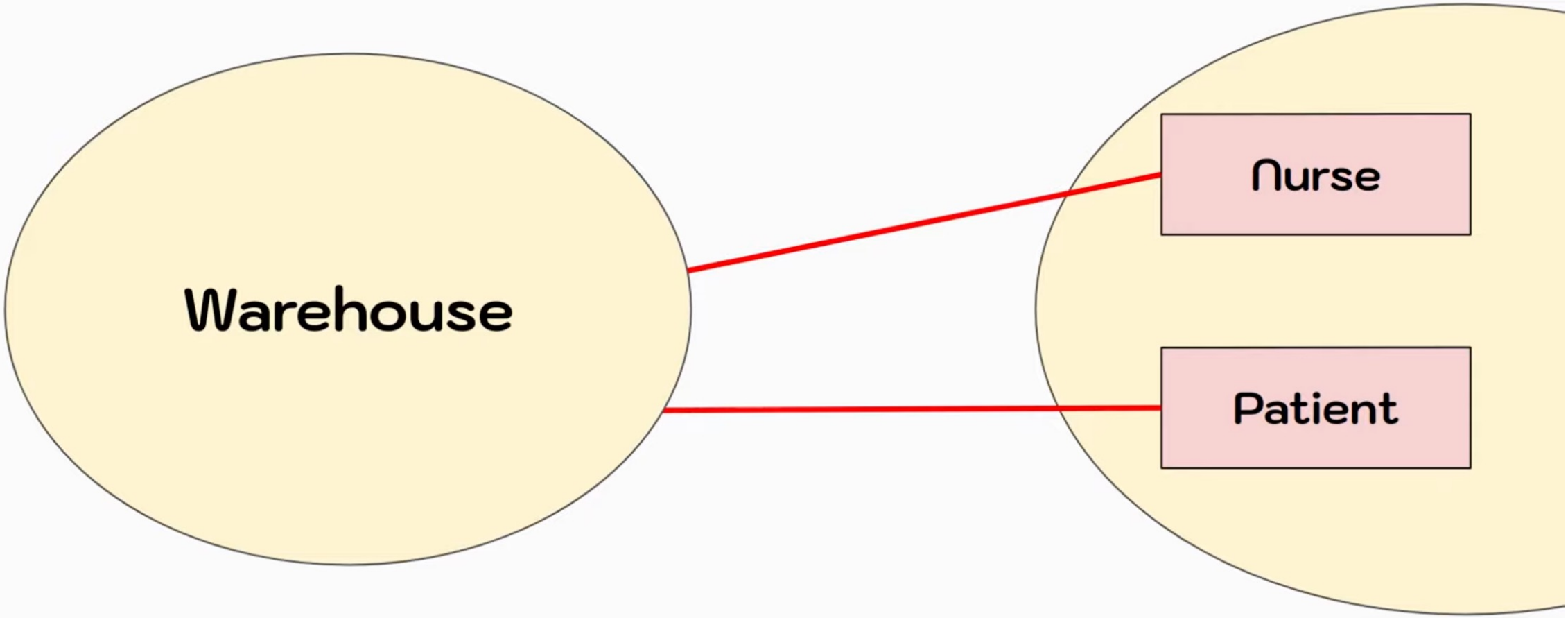
```
nurse.vaccinate(patient, vaccine)
```

Scheduling

คนไข้ P ได้รับ Vaccine X โดย พยาบาล A

```
patient.take(vaccine, by=nurse)
```

Medication Record


Warehouse
Nurse
Patient