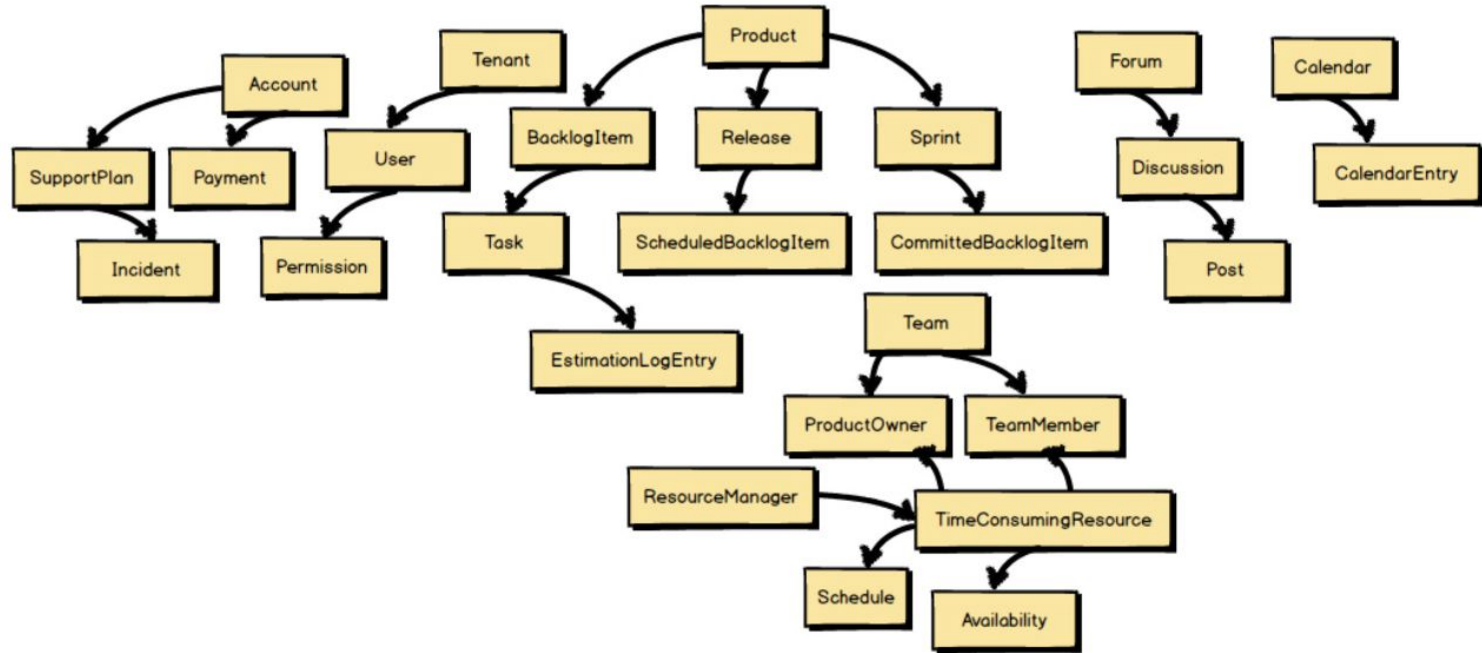


Bounded Contexts

Unbounded Context



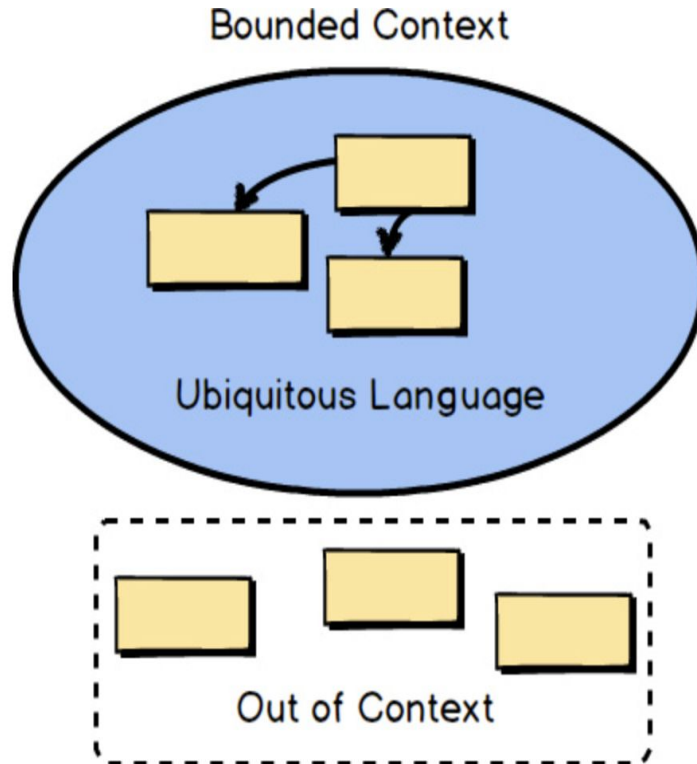
Fundamental Strategic Design Needed

Employing a Bounded Context forces us to answer the question “**What is core?**”

Bounded Contexts are not monolithic, the tests will be focused on one model and thus be fewer in number and will run more quickly.

- In context
- Out of context

Bounded Context

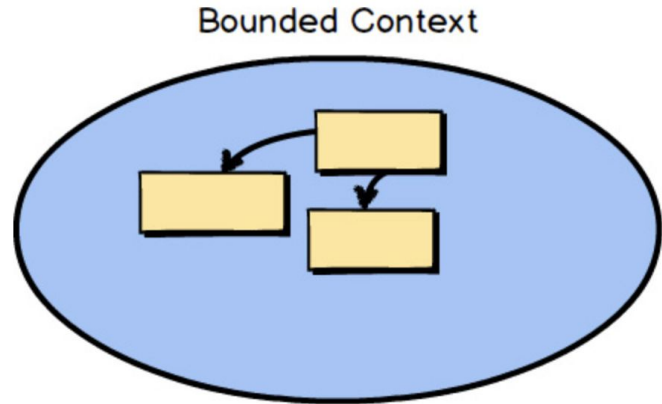


Bounded Context

In short, DDD is primarily about modeling a Ubiquitous Language in an explicitly Bounded Context.

The components inside a Bounded Context are context specific.

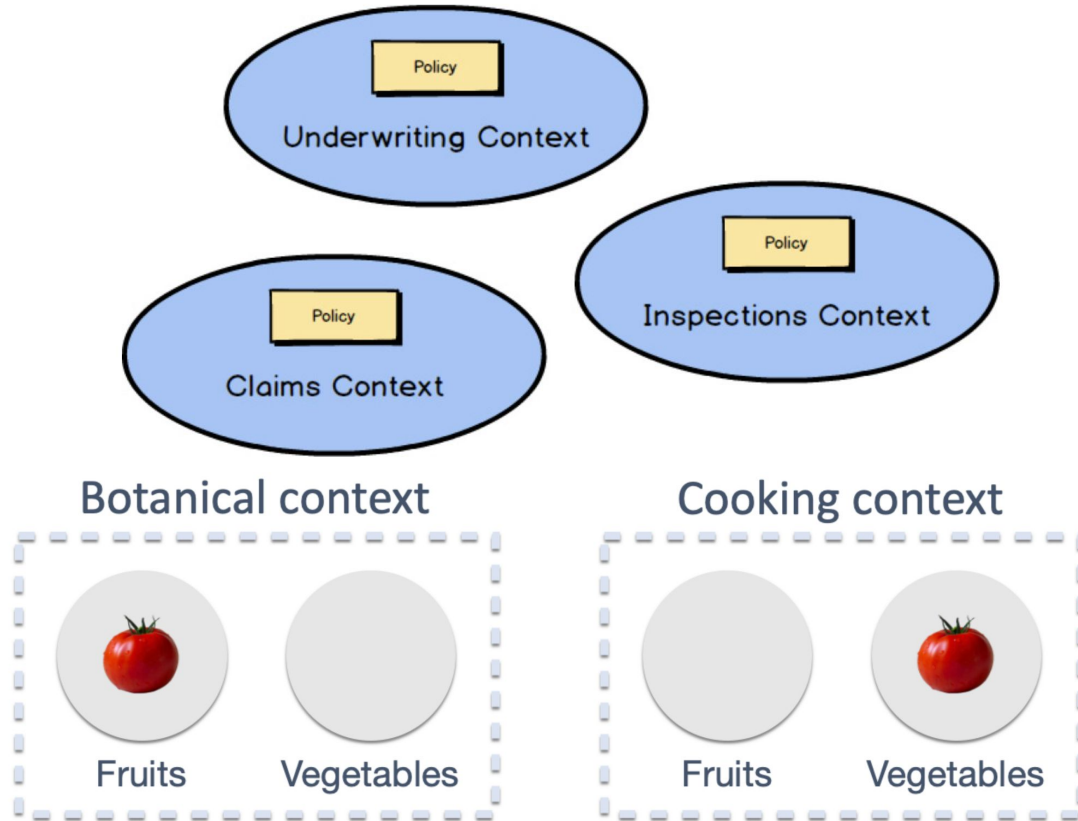
You develop your solution in the Bounded Context as code, both main source and test source.



Characteristics of Bounded Context

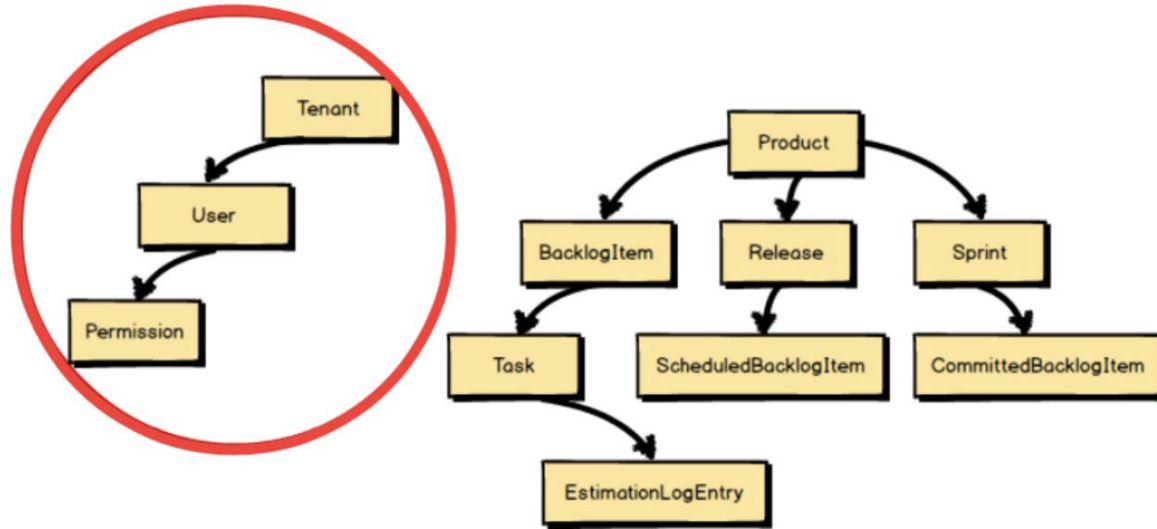
- There should be **one team assigned** to work on one Bounded Context.
- There should also be a **separate source code repository** for each Bounded Context.
- There should also be a **separate database schema** for each Bounded Context.
- There should also be **separate unit test cases** for each Bounded Context.
- There should also be **separate acceptance test cases** for each Bounded Context.

Domains are Fuzzy & Contextual



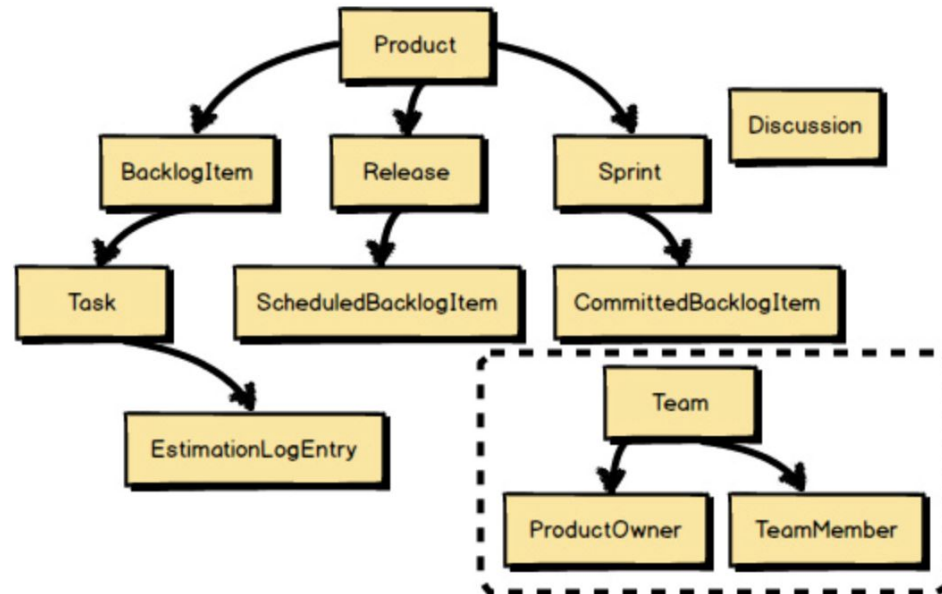
Case Study - 1 with DDD

Is it a Ubiquitous Language of Scrum ?



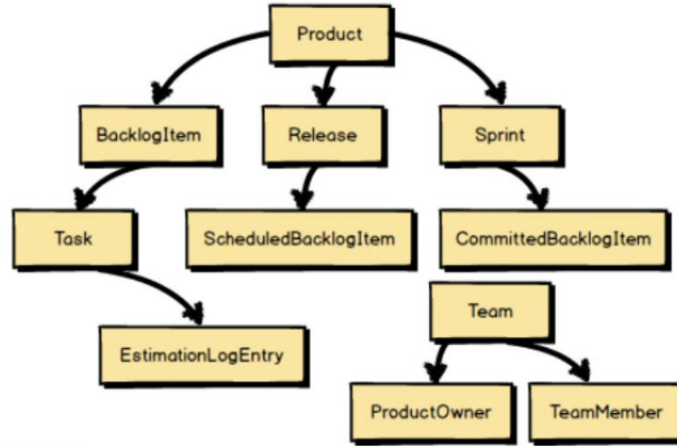
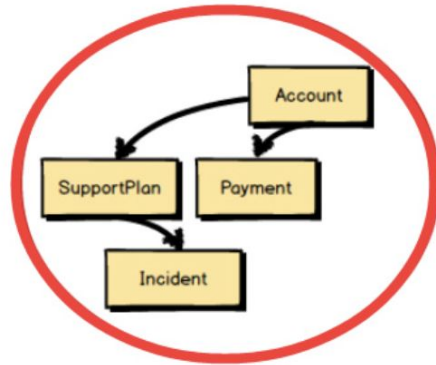
Case Study - 1 with DDD

ProductOwner and **TeamMember** we adhere to the Ubiquitous Language of Scrum.



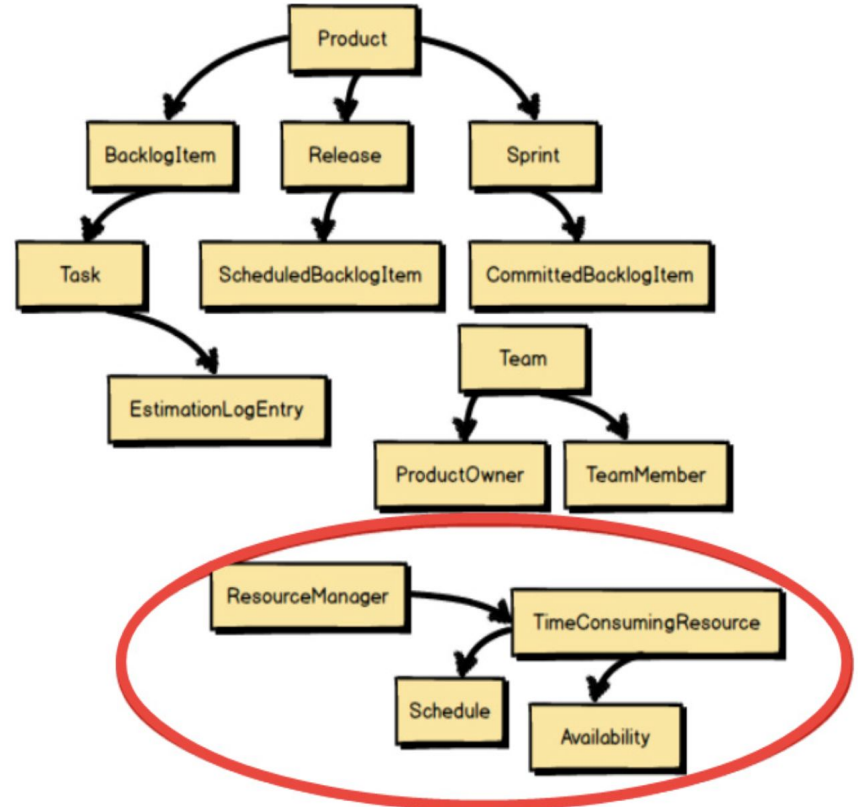
Case Study - 1 with DDD

SupportPlans and Payments?



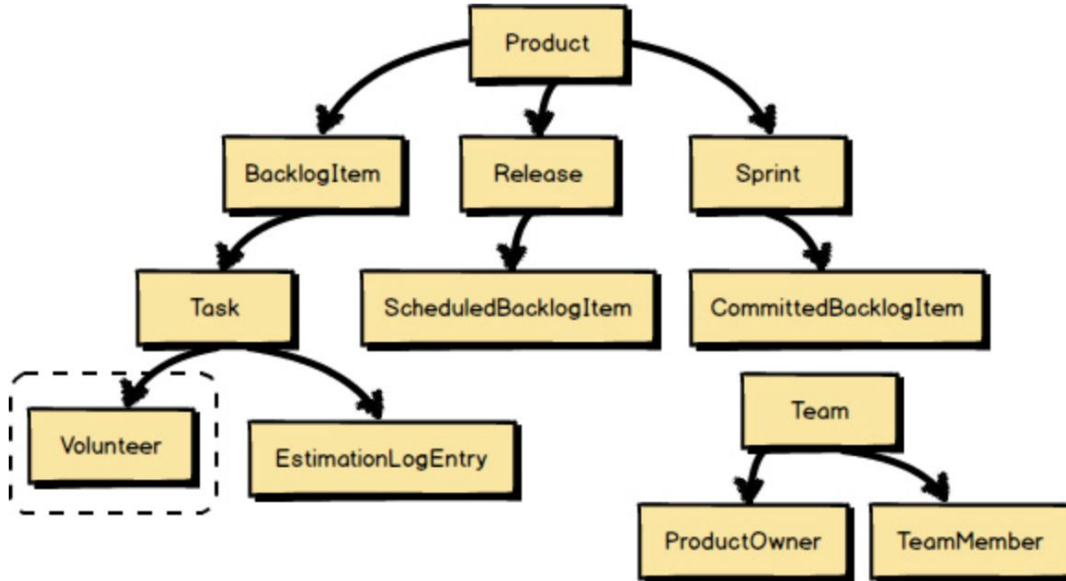
Case Study - 1 with DDD

Human Resource Utilization concerns'



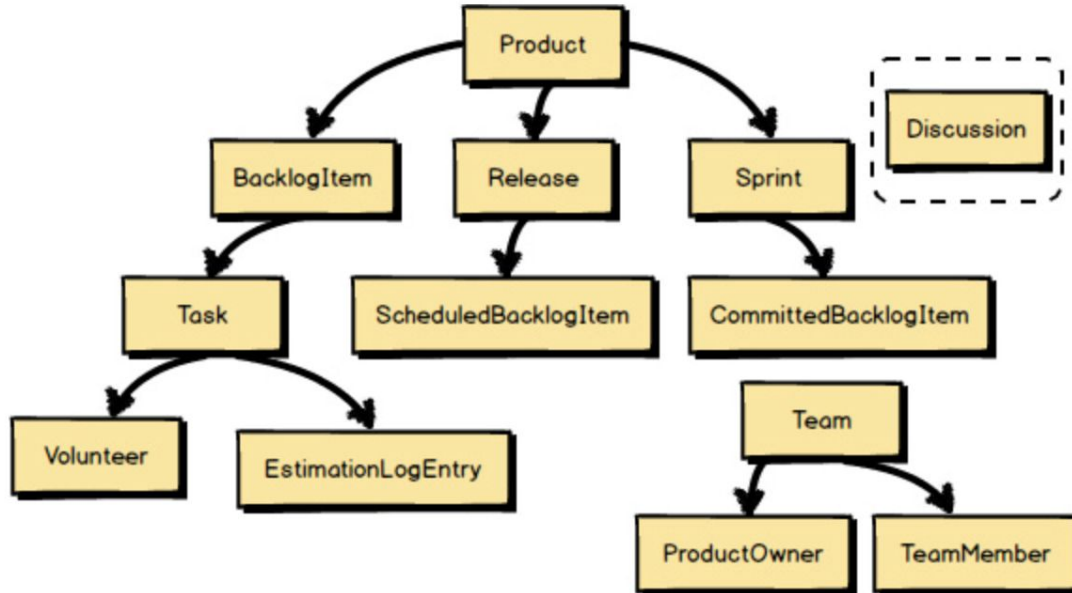
Case Study - 1 with DDD

Volunteer is in context and was included in the language of the core model.



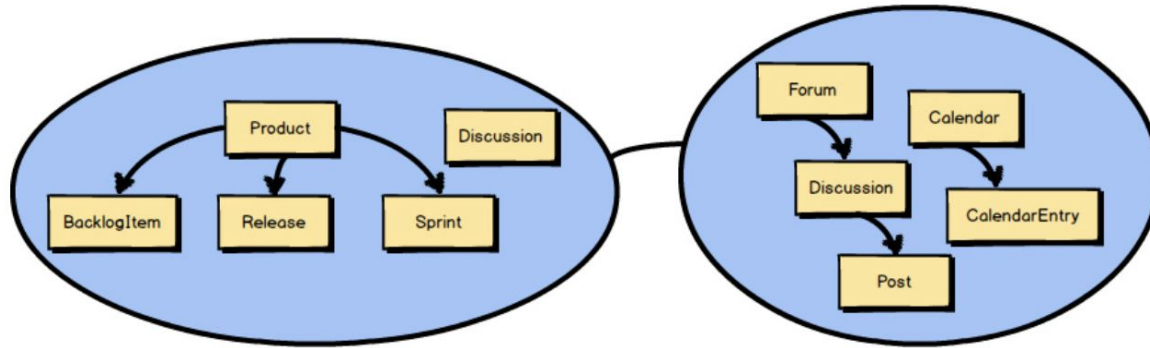
Case Study - 1 with DDD

Discussion is part of the team's Ubiquitous Language, and thus inside the Bounded Context.



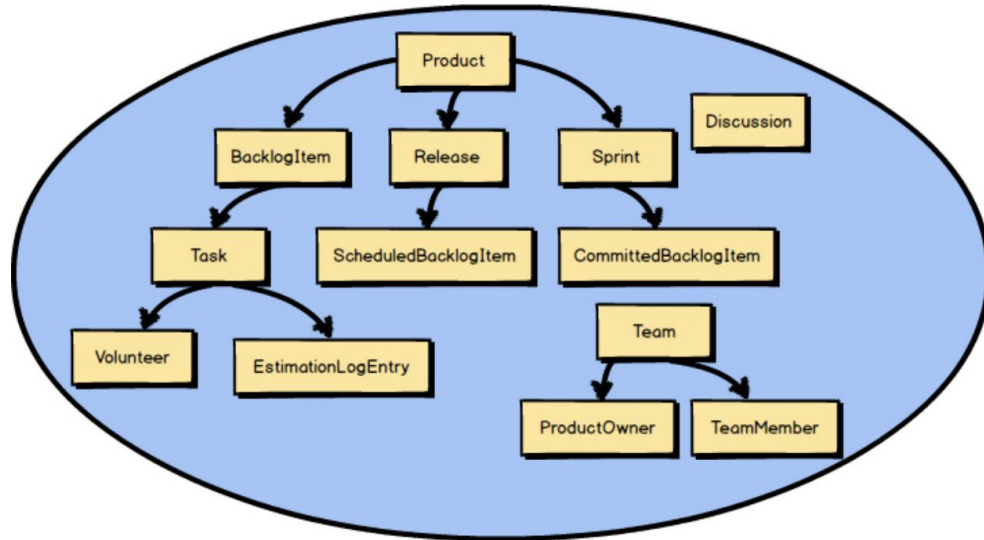
Case Study - 1 with DDD

The **Discussion** will be supported by **integrating** with another Bounded Context—the Collaboration Context.



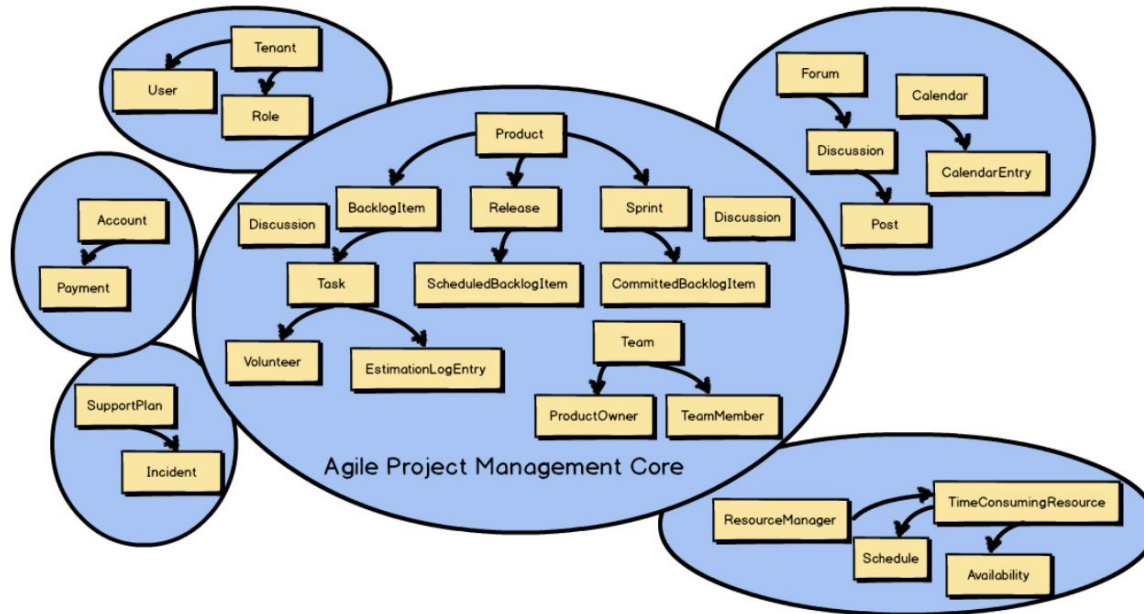
Case Study - 1 with DDD

Core Domain



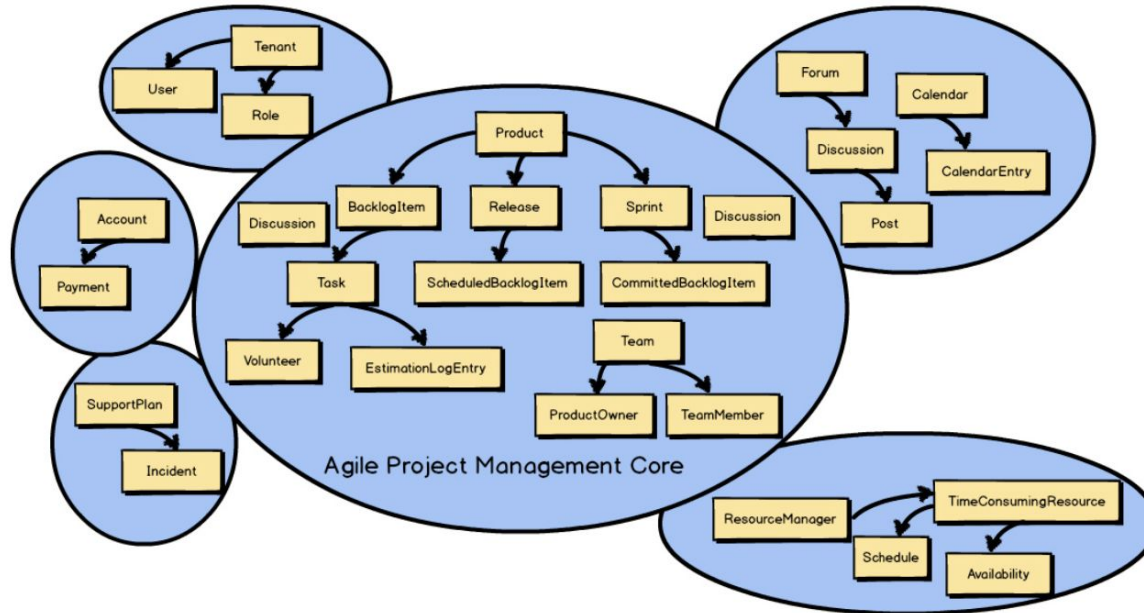
Case Study - 1 with DDD

Each Bounded Context is adhering to its own Ubiquitous Language.



Case Study - 1 with DDD

Each Bounded Context is adhering to its own Ubiquitous Language.



Developing a Ubiquitous Language

- Don't limit your Core Domain to nouns alone.
- Consider expressing your Core Domain as a set of concrete scenarios.
- How the domain model should work?
- You can actually have conversations about how the domain model works—its design

Scenario - 1

"The product owner commits each backlog item to a sprint . . ."

Who does the committing of backlog items to a sprint?

*"The product owner **Isabel** commits the **View User Profile** backlog item to the **Deliver User Profiles** sprint . . ."*

Giving names or other distinguishing identities to concepts in the scenario helps.

Scenario - 1

Wait a minute! Product owner is the sole individual responsible for deciding that a backlog item will be committed to a sprint.

The team to enable the product owner to perform the commitment?

“The product owner commits a backlog item to a sprint. The backlog item may be committed only if it is already scheduled for release, and if a quorum of team members have approved commitment . . .”

Scenario - 2

"When the commit completes, notify interested parties."

Who or what are the interested parties?

Who needs to know when a backlog item has been committed to a sprint?

Answer is **"Sprint"**

The sprint needs to track total sprint commitment.

Scenario - 2

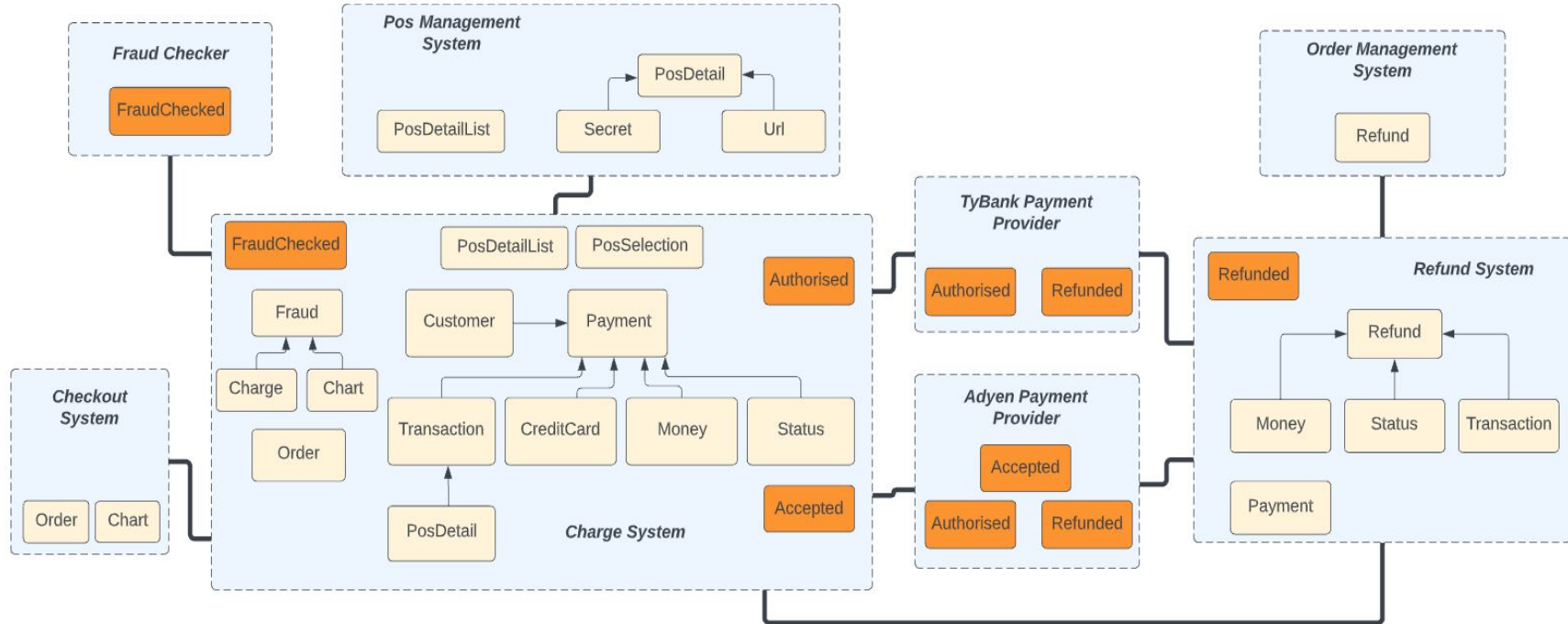
Okey, What kind of **restrictions** exist when the commit is completed?

What if this job has already been committed to another sprint?

“If it is already committed to a different sprint, it must be uncommitted first. When the commitment completes, notify the sprint from which it was uncommitted and the sprint to which it is now committed.”

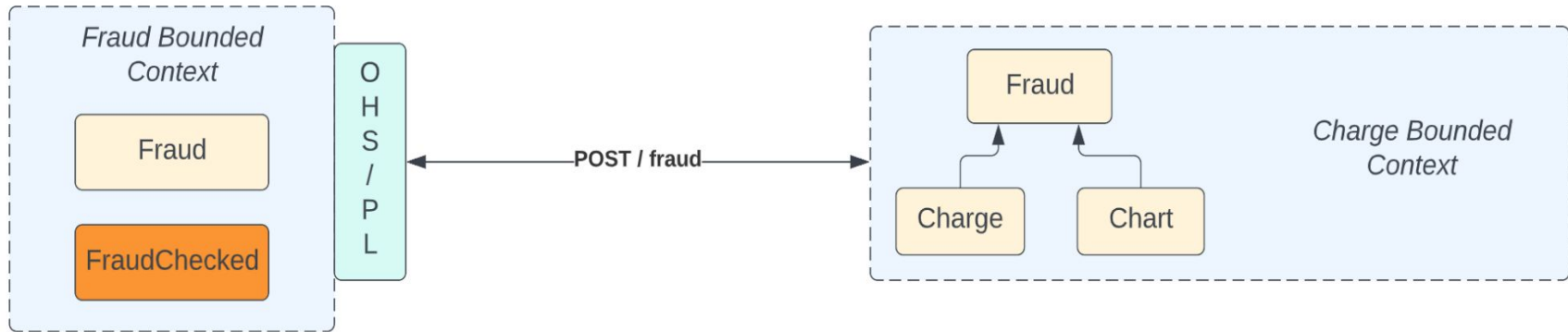
Case Study - 2 with DDD

Each Bounded Context is adhering to its own Ubiquitous Language.



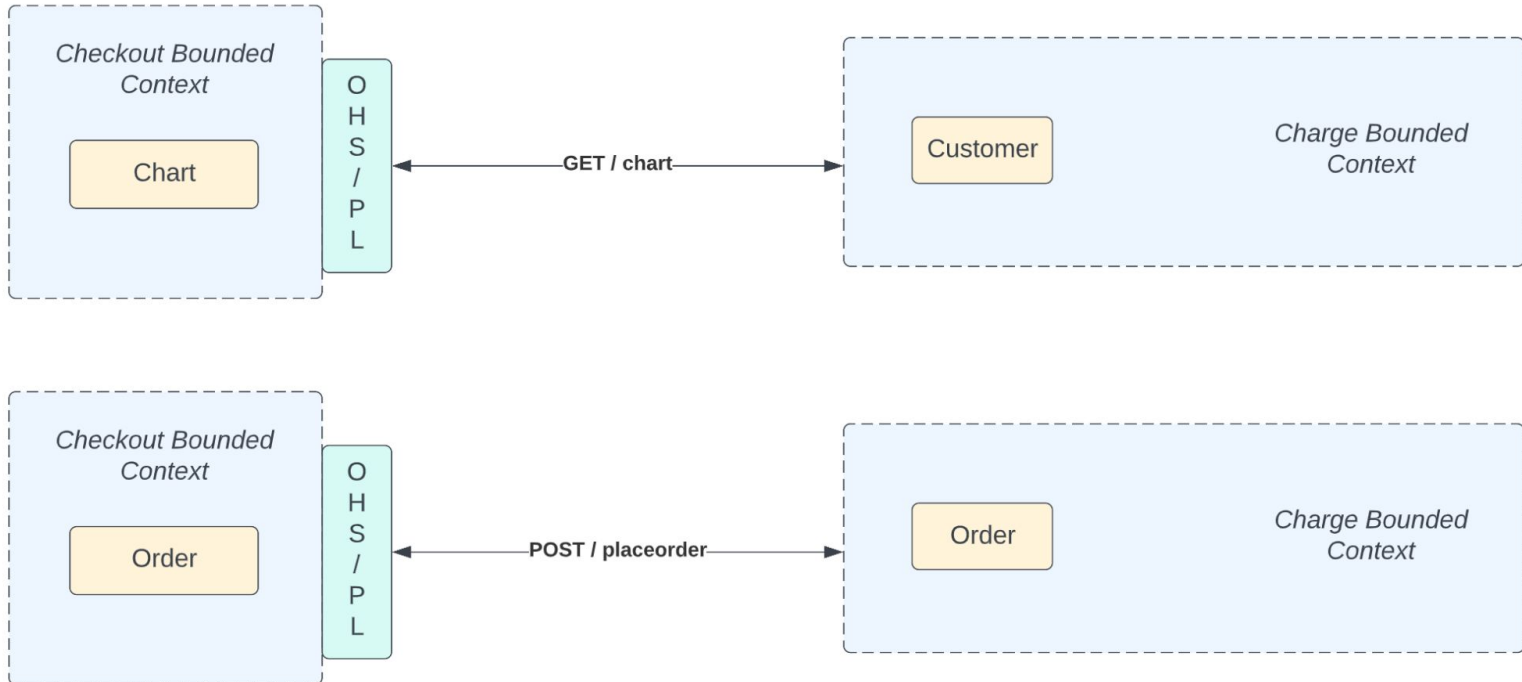
Case Study - 2 with DDD (Context Mappings)

1- Fraud Bounded Context and Charge Context Mapping



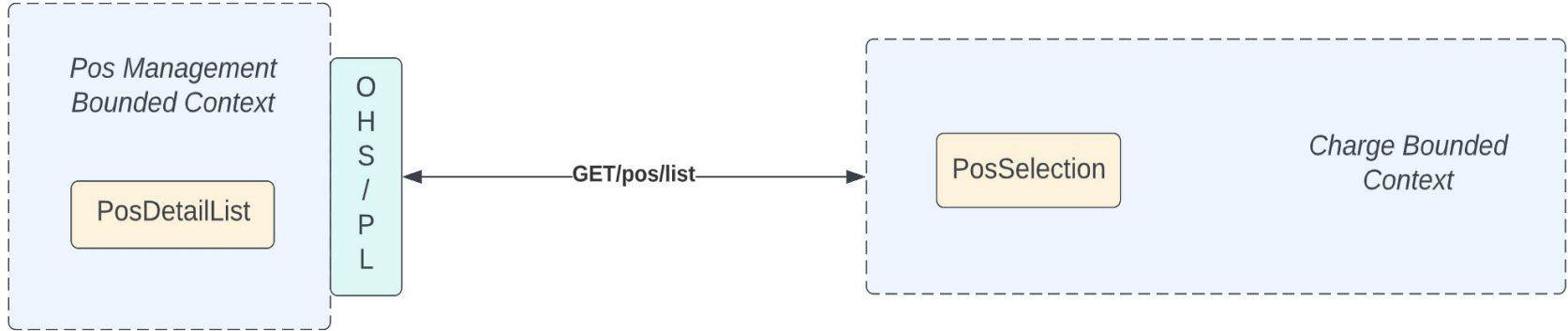
Case Study - 2 with DDD (Context Mappings)

2- Checkout Bounded Context and Charge Context Mapping



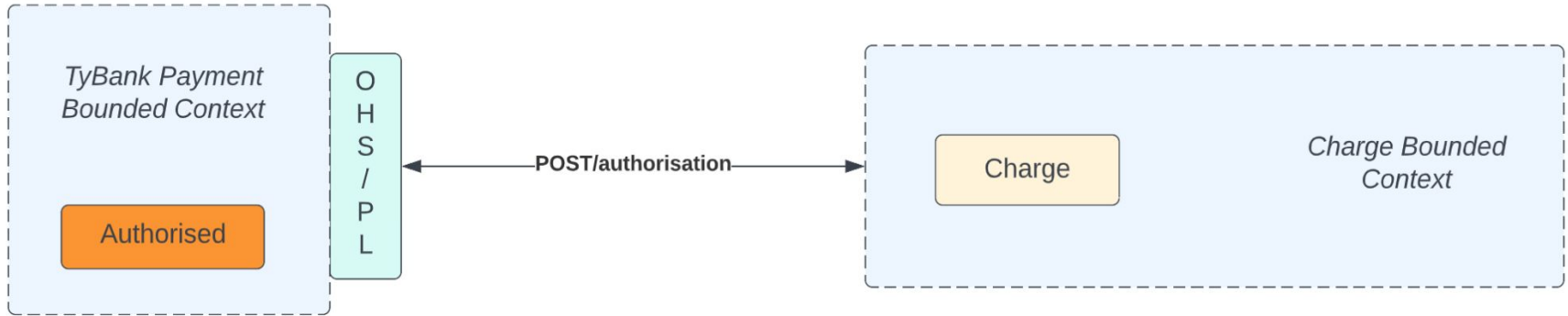
Case Study - 2 with DDD (Context Mappings)

3- Pos Management Bounded Context and Charge Context Mapping



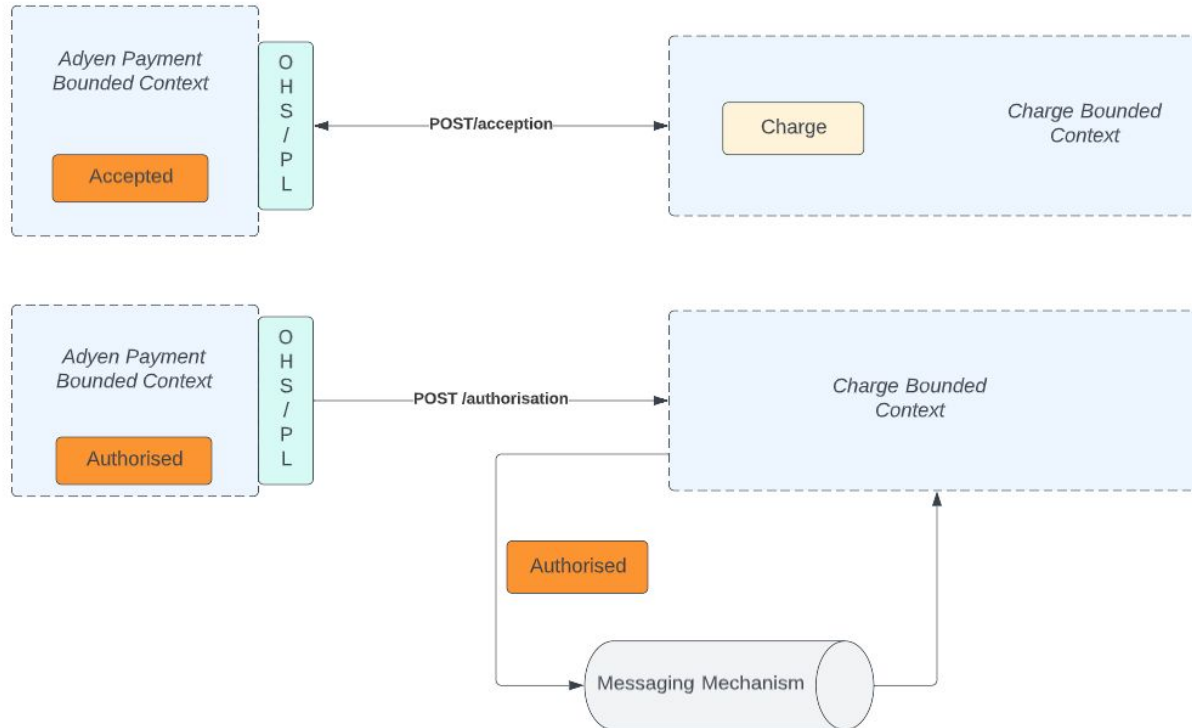
Case Study - 2 with DDD (Context Mappings)

4- TyBank Payment Bounded Context and Charge Context Mapping



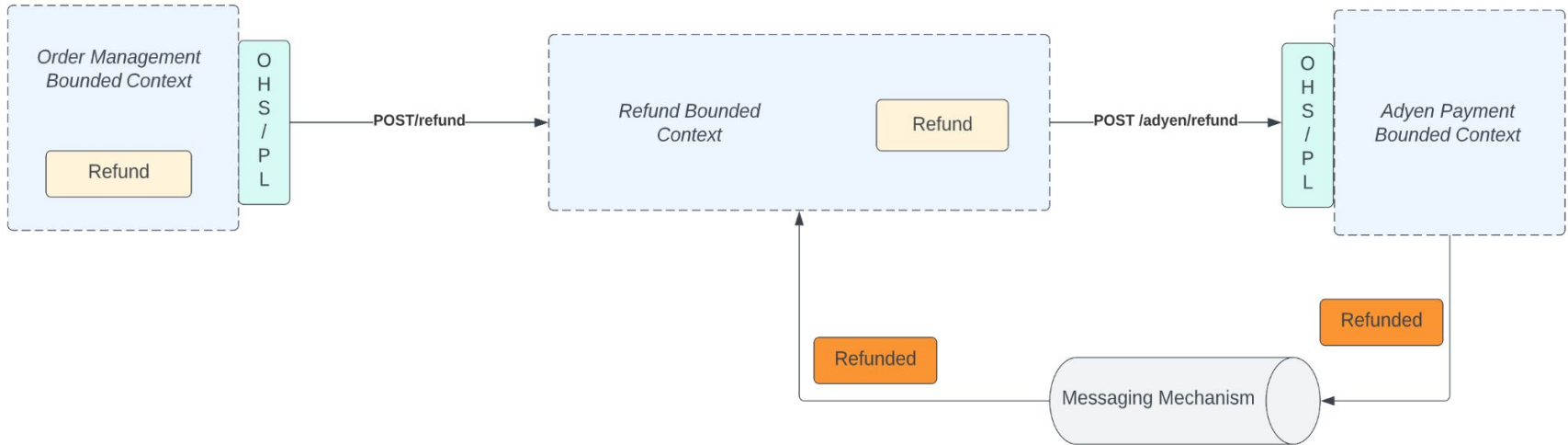
Case Study - 2 with DDD (Context Mappings)

5- Adyen Payment Bounded Context and Charge Context Mapping



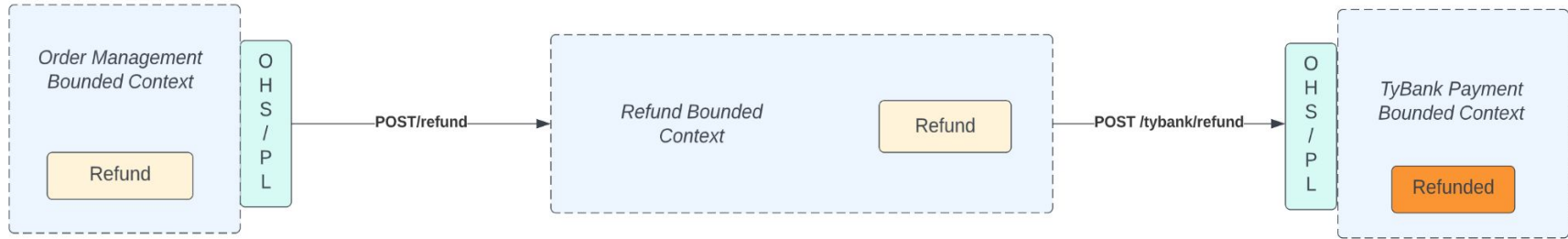
Case Study - 2 with DDD (Context Mappings)

6- Adyen Payment Bounded Context and Refund Context Mapping



Case Study - 2 with DDD (Context Mappings)

7- TyBank Payment Bounded Context and Refund Context Mapping



Case Study - 2 with DDD (Context Mappings)

8- Charge Bounded Context and Refund Context Mapping

