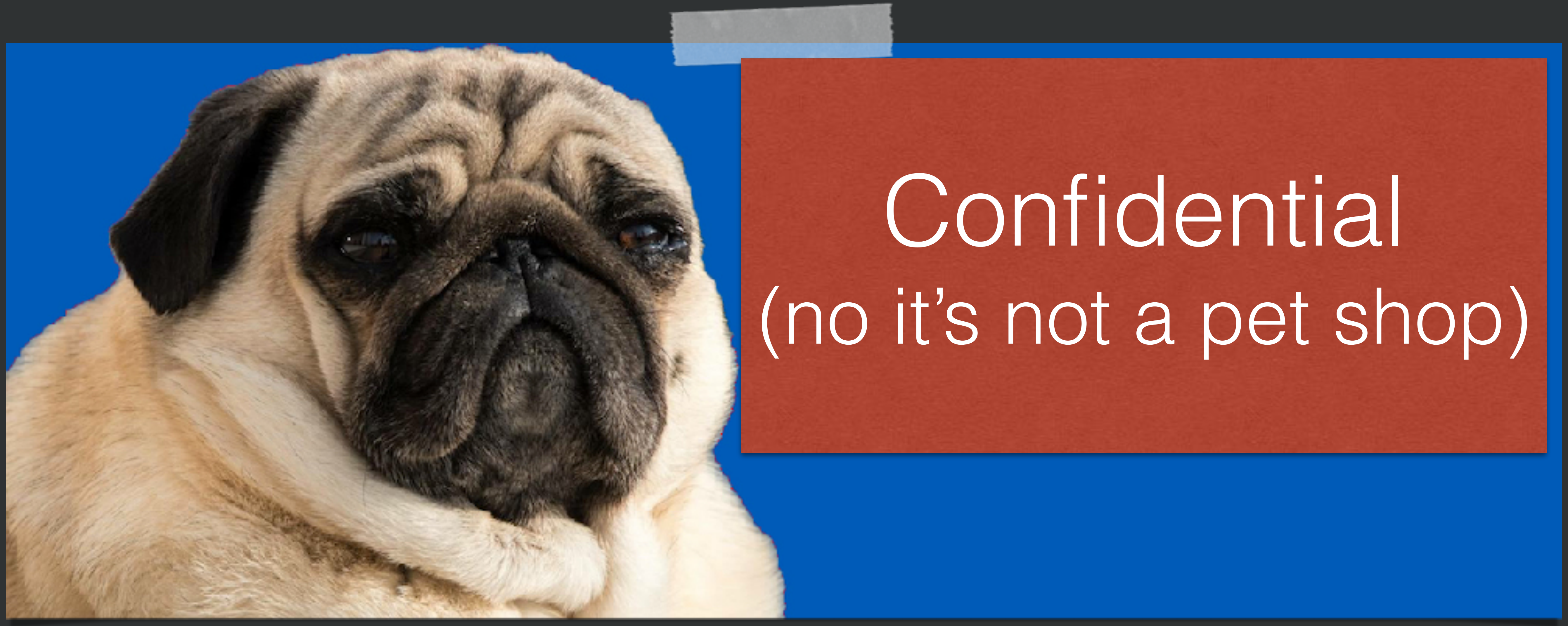
DDD

Strategic Design With Spring Boot



Michael Plöd - innoQ
@bitboss

The Spring Boot / code part of this presentation can be found at:

https://github.com/mploed/ddd-strategic-design-spring-boot

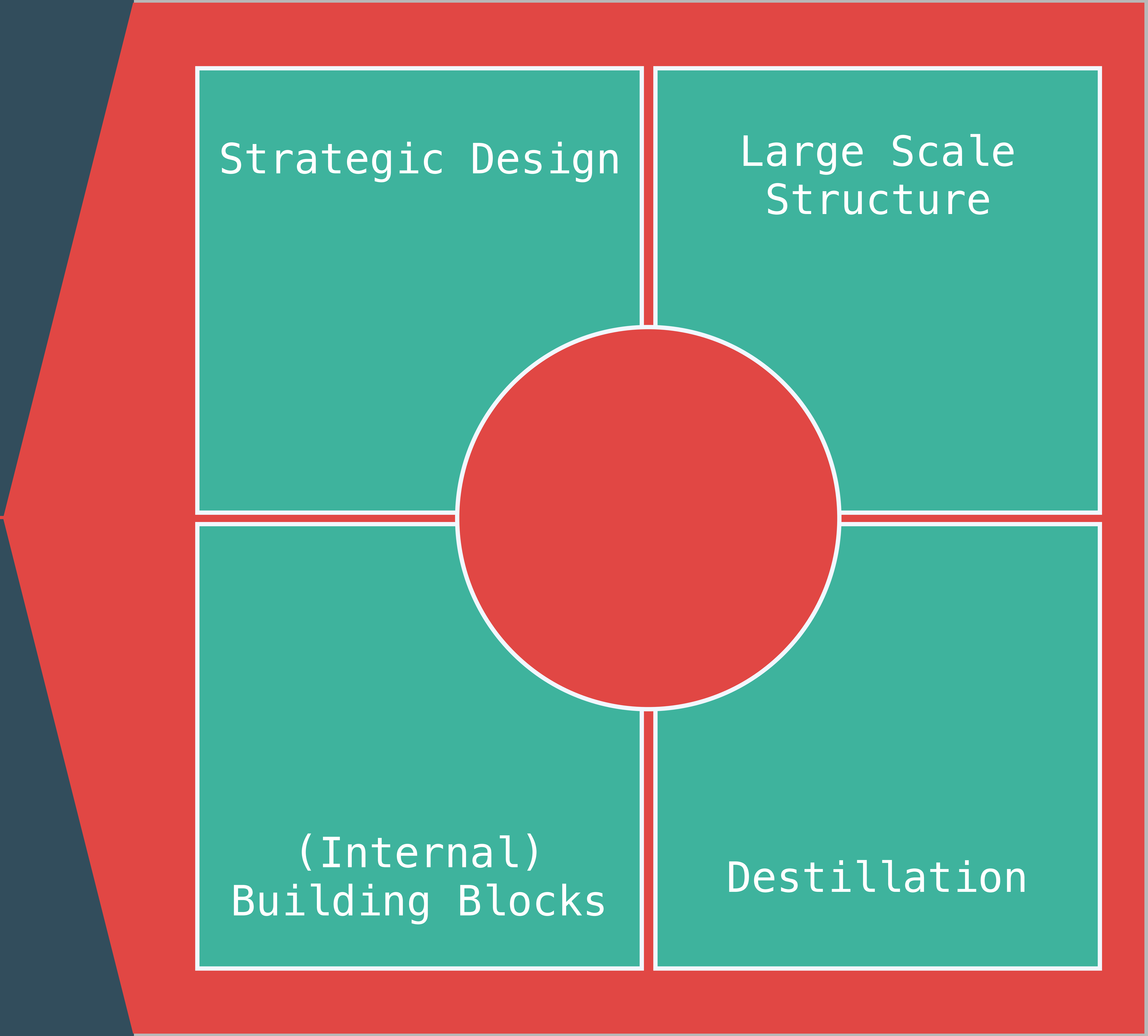
Michael Plöd - innoQ
@bitboss

Disclaimer

Most of these ideas do not come from me personally. I have to thank Eric Evans for all the inspiration / ideas. If you haven’t: go out and get his amazing book: Domain Driven Design.

Michael Plöd - innoQ
@bitboss

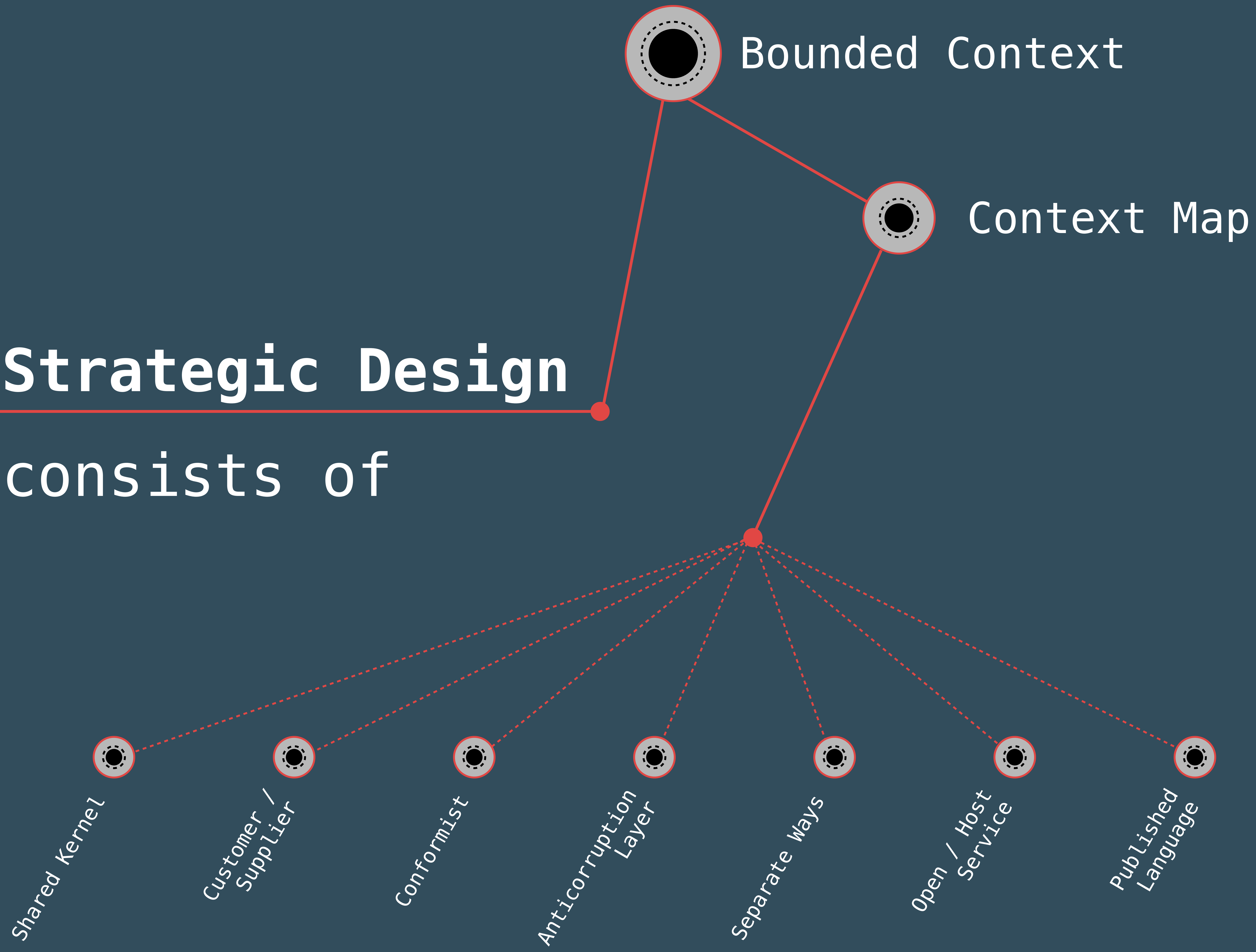
Domain Driven Design



Strategic Design

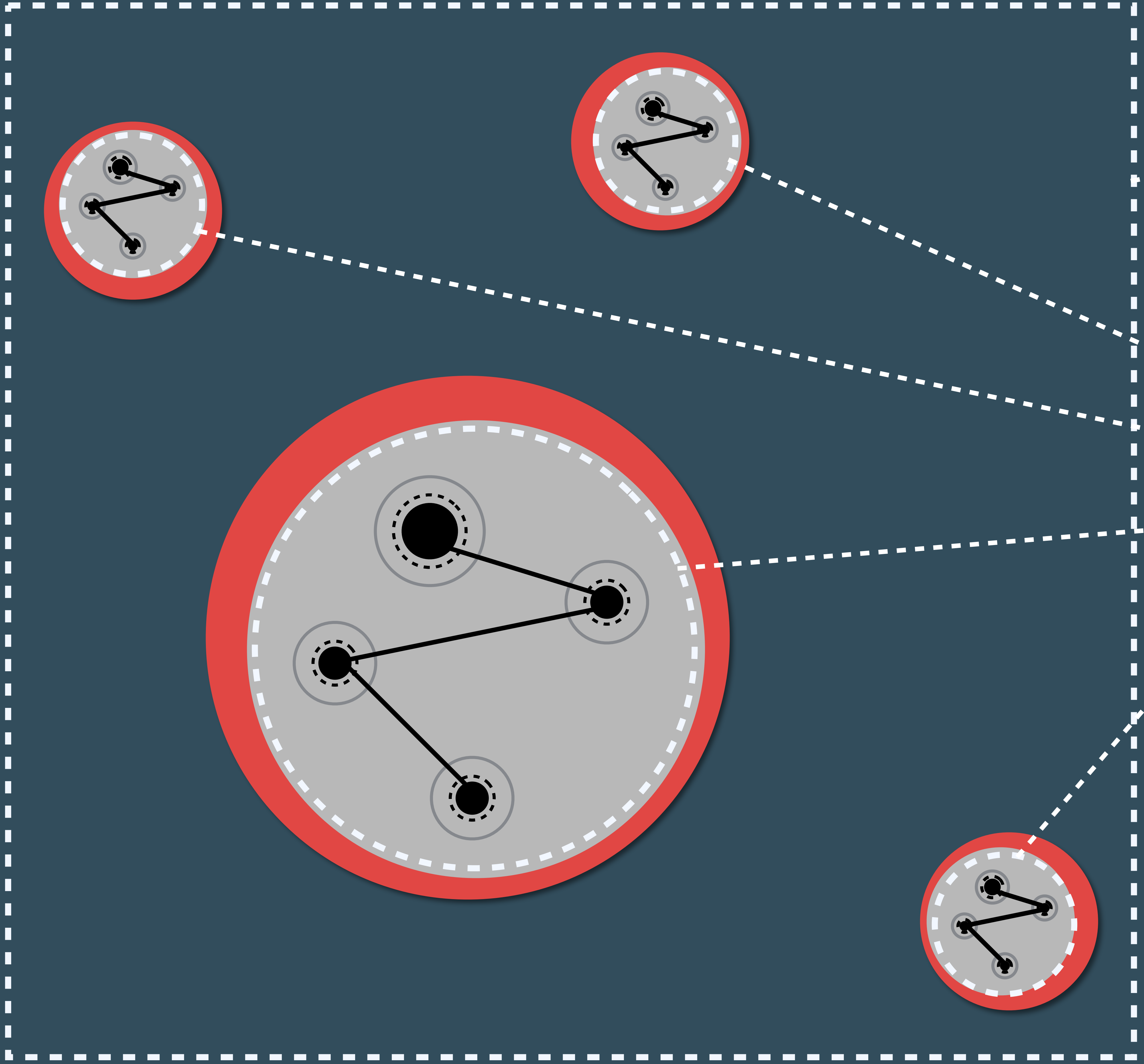
Strategic Design

consists of



Strategic Design

Bounded Context



Every sophisticated business domain consists of a bunch of **Bounded Contexts**

Each **Bounded Context** contains models and maybe other contexts

The **Bounded Context** is also a boundary for the meaning of a given model

Strategic Design

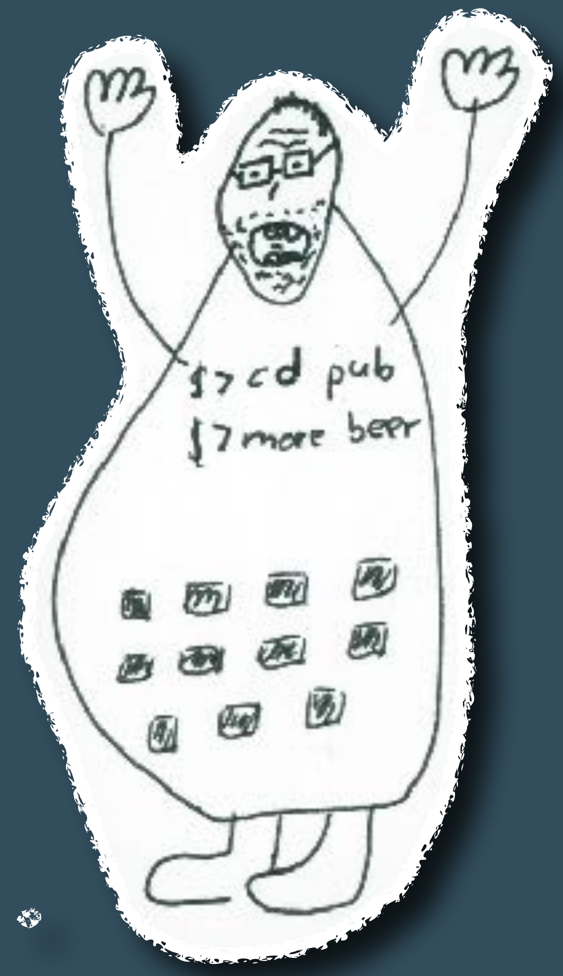
Bounded Context Example

Reservations

Event Management

Badges

Customer



Name

Payment Details

Adress

Company

Session Registrations

Lunch Preferences

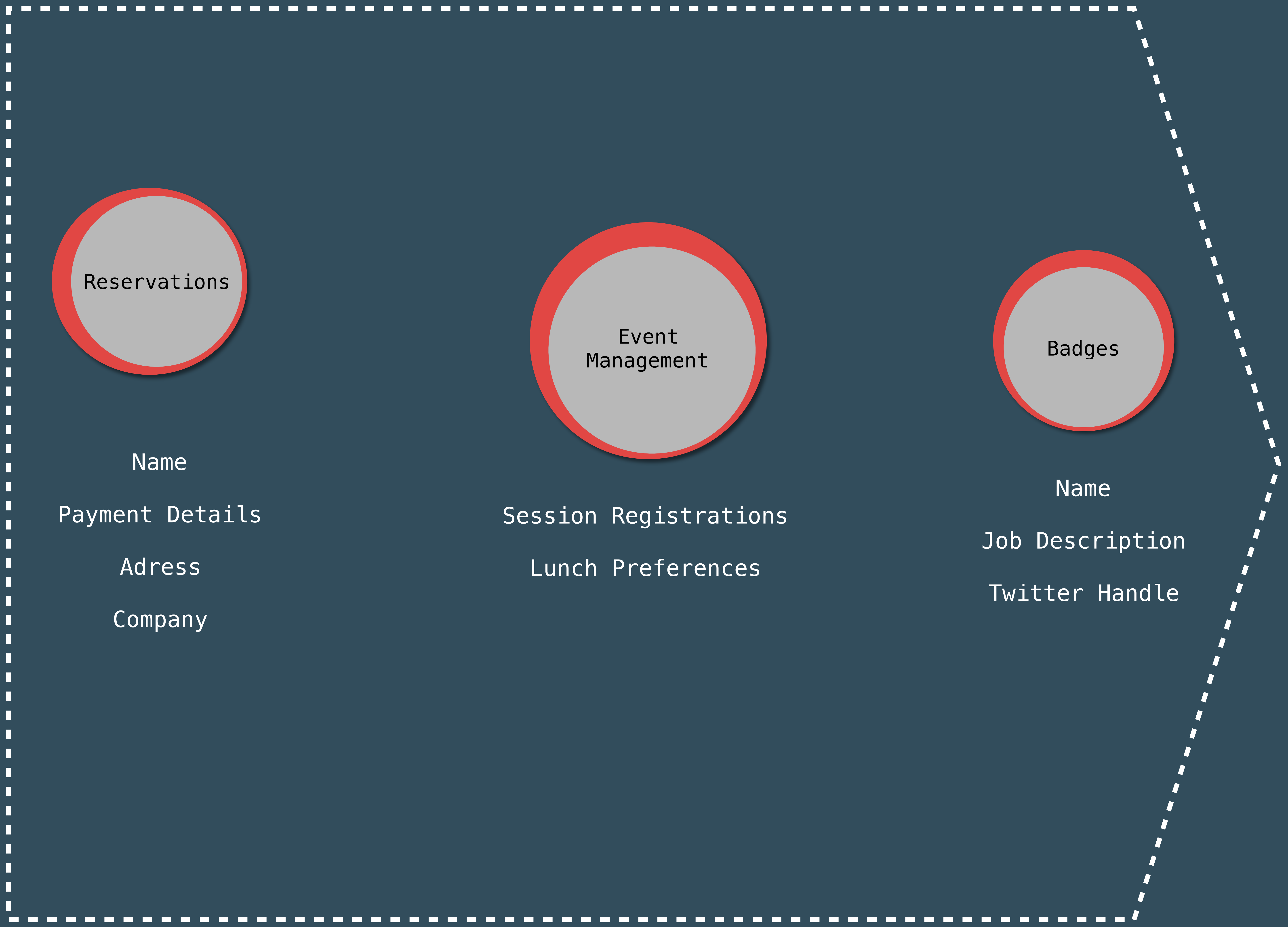
Name

Job Description

Twitter Handle

Strategic Design

Bounded Context Example



Each Bounded Context has its own model of a customer

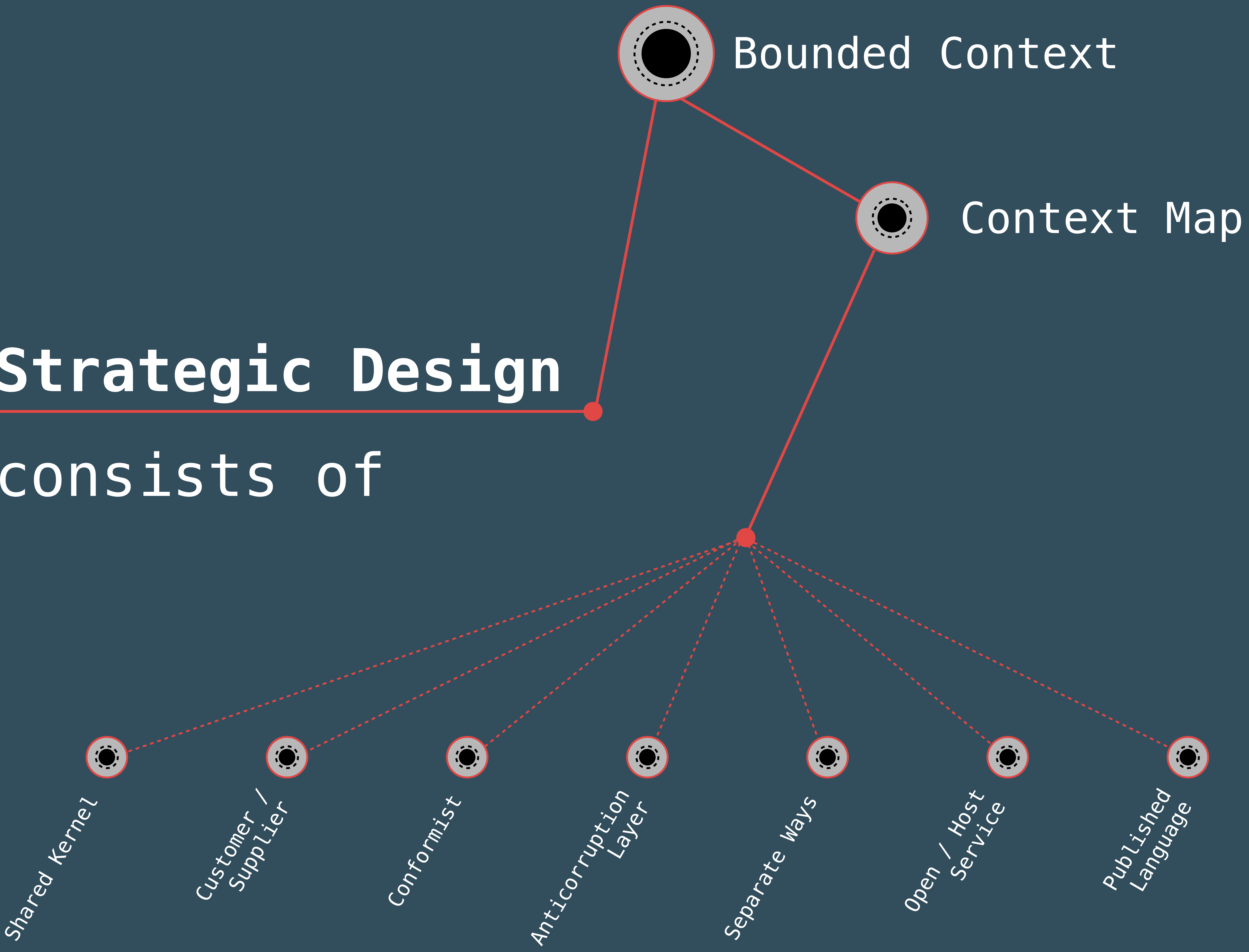
This is a major enabler for independent Microservices

Take a look at the name of the customer? Maybe we want some shared data?

Strategic Design

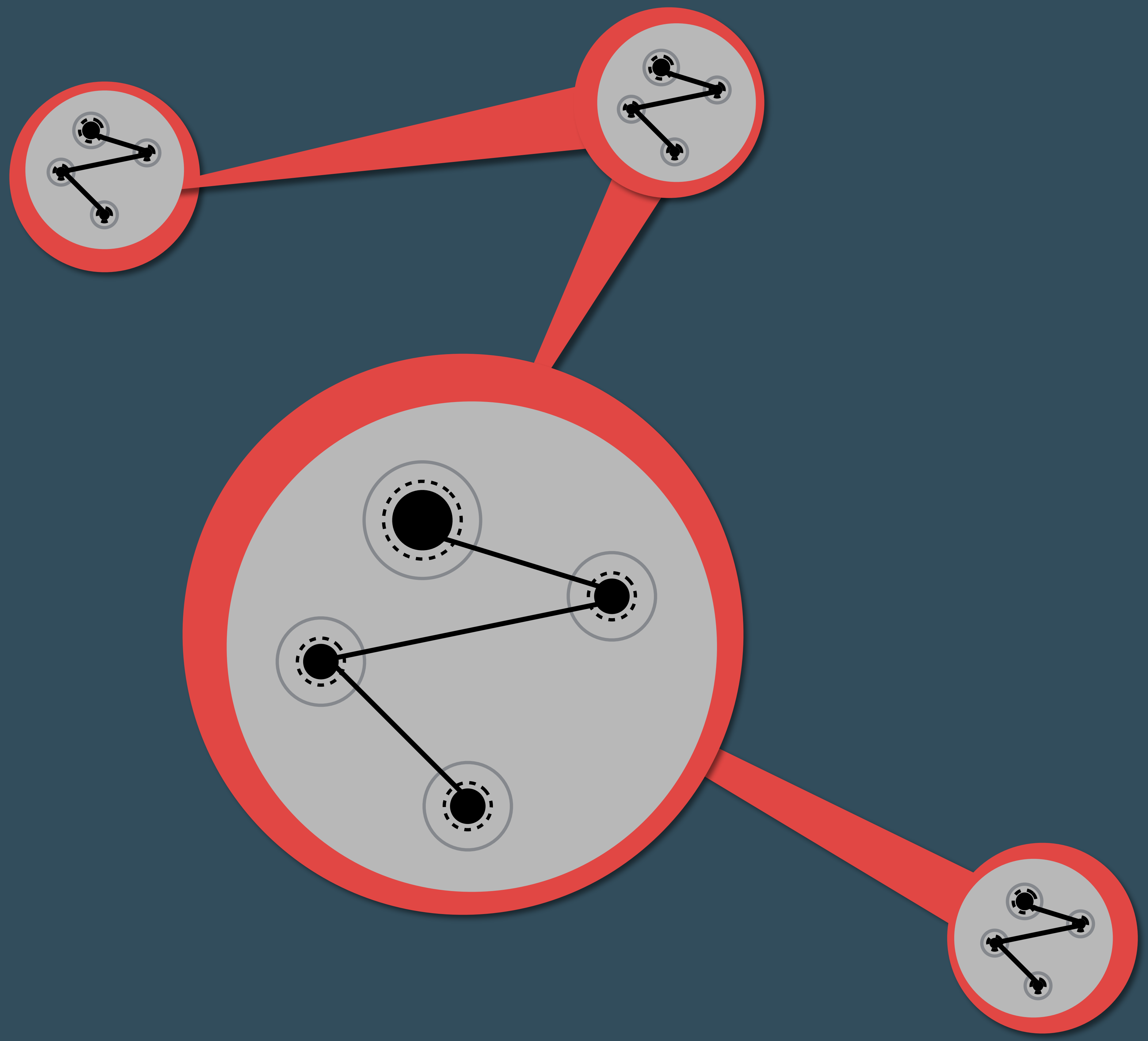
Strategic Design

consists of



Strategic Design

Context Map



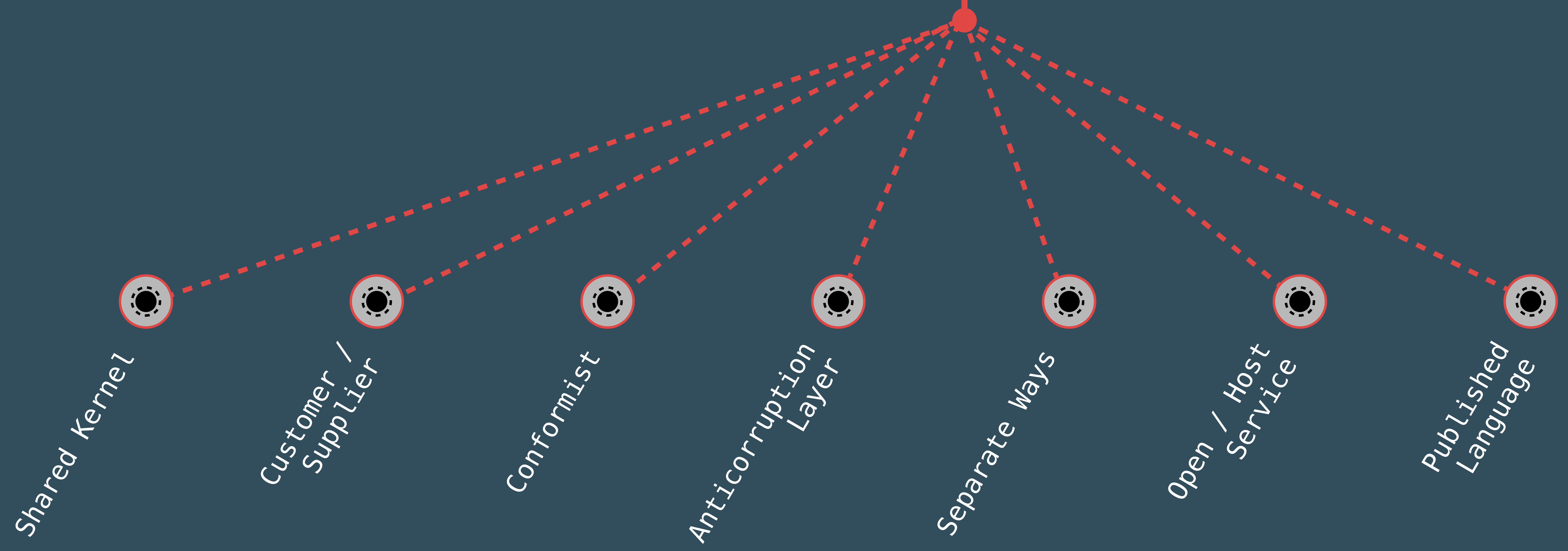
The Bounded Context by itself does not deliver an overview of the system

By introducing a **Context Map** we describe the contact between models / contexts

The **Context Map** is also a great starting point for future transformations

Strategic Design

Context Map – Patterns

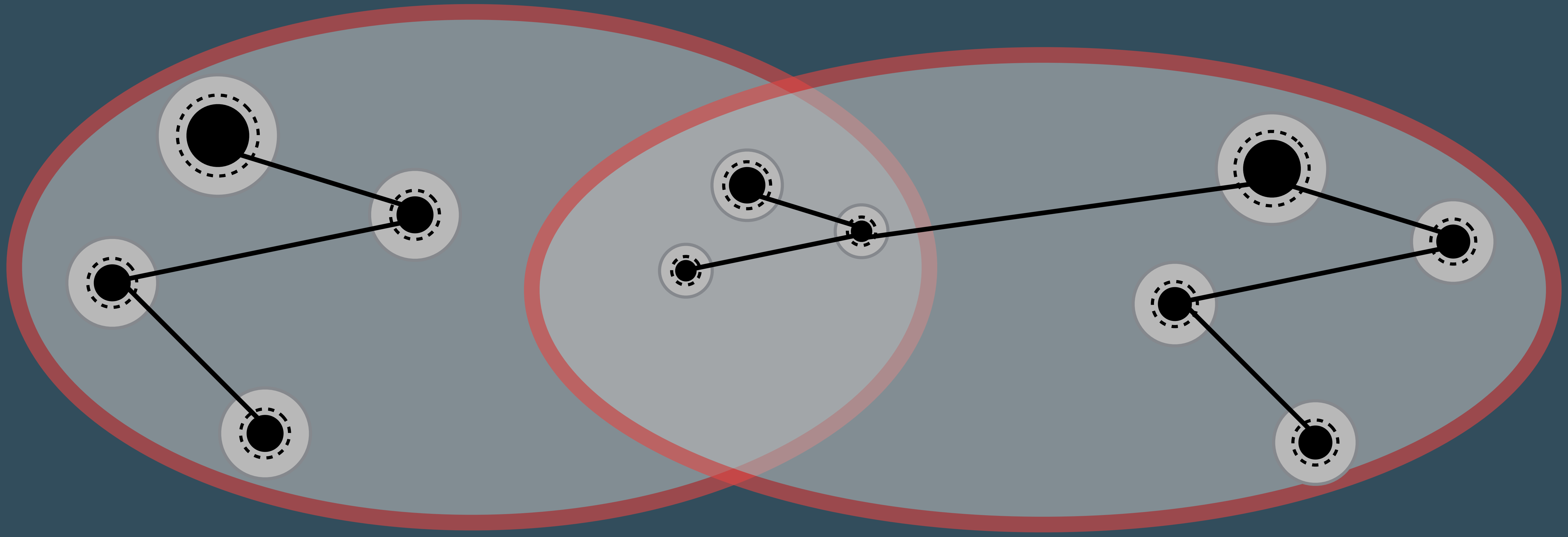


Strategic Design

Context Map – Patterns

- **Shared Kernel**
- Customer / Supplier
- Conformist
- Anticorruption Layer
- Separate Ways
- Open / Host Service
- Published Language

Two teams share a subset of the domain model including code and maybe the database. The shared kernel is often refered to as the core domain.

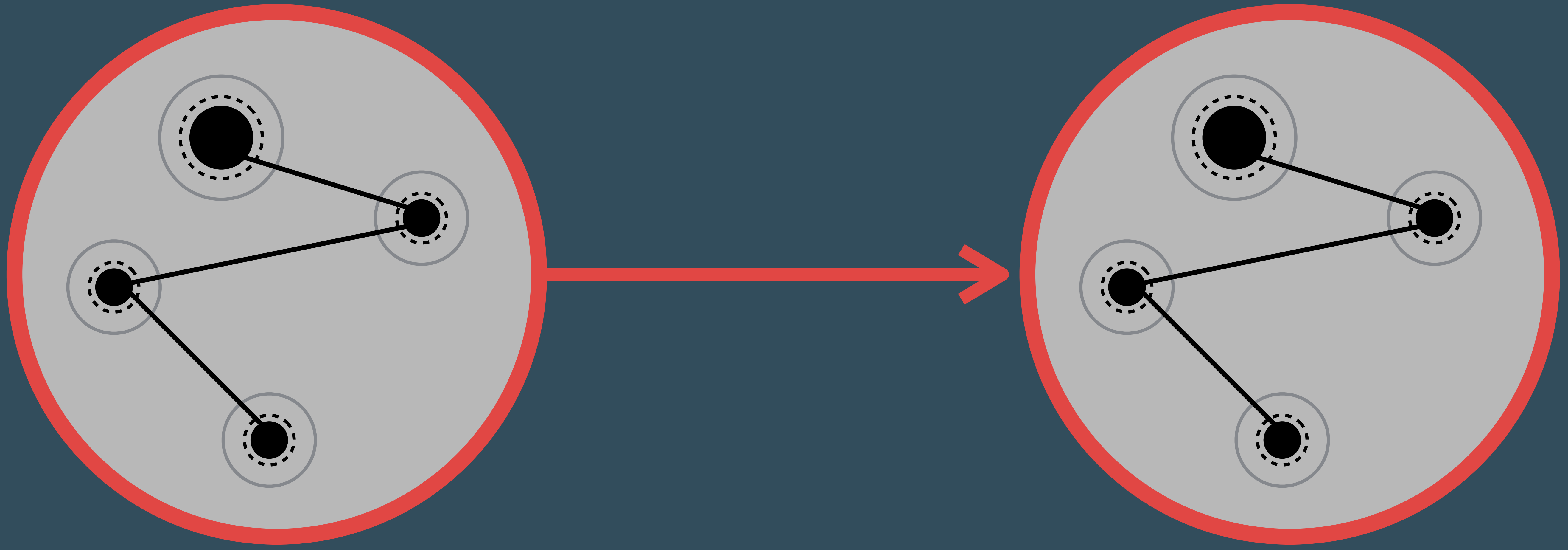


Strategic Design

Context Map – Patterns

- Shared Kernel
- Customer / Supplier
- Conformist
- Anticorruption Layer
- Separate Ways
- Open / Host Service
- Published Language

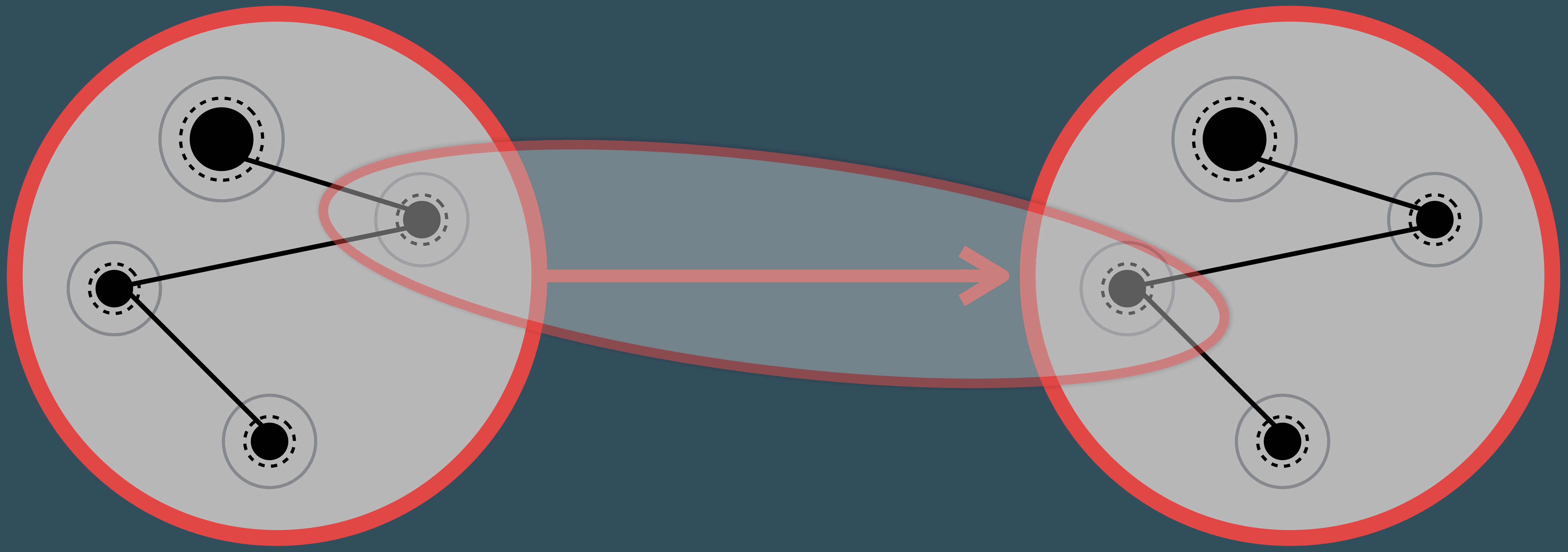
There is a customer / supplier relation ship between two teams. The downstream team is considered to be the customer, sometimes with veto rights.



Context Map – Patterns

- Shared Kernel
- Customer / Supplier
- **Conformist**
- Anticorruption Layer
- Separate Ways
- Open / Host Service
- Published Language

The downstream team conforms to the model of the upstream team. There is no translation of models and no vetoing. If the upstream model is a mess, it propagates to the downstream model.

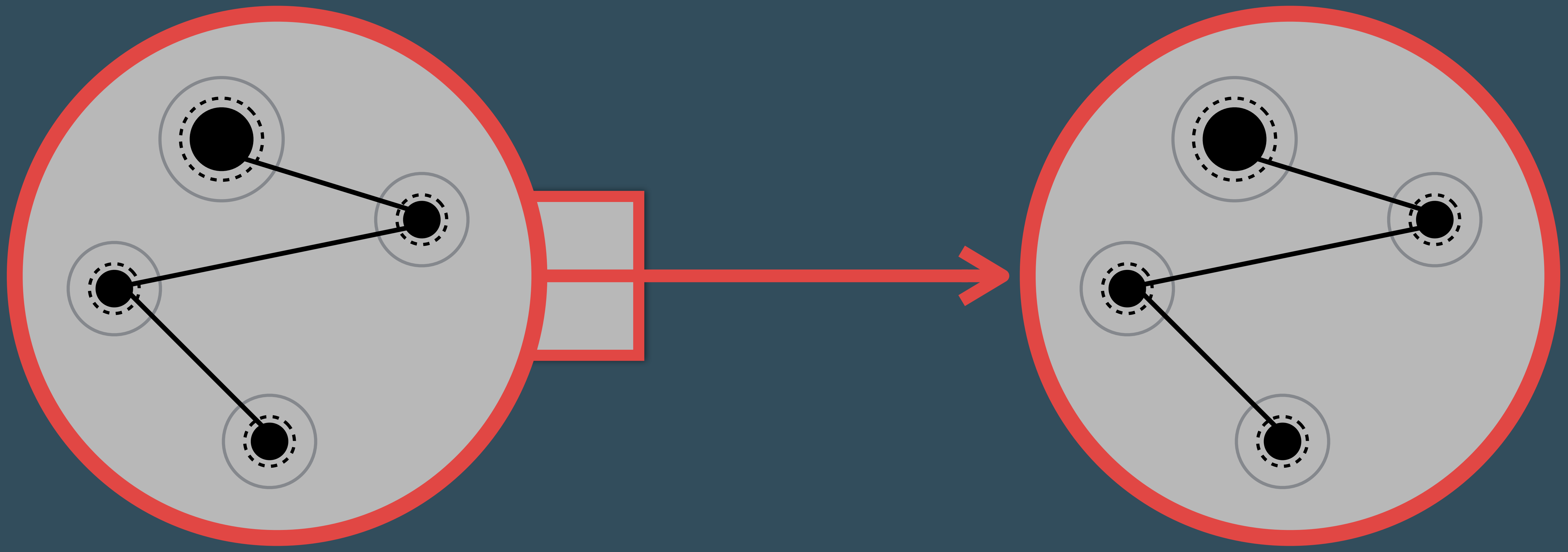


Strategic Design

Context Map – Patterns

- Shared Kernel
- Customer / Supplier
- Conformist
- Anticorruption Layer
- Separate Ways
- Open / Host Service
- Published Language

The anticorruption layer is a layer that isolates a client's model from another system's model by translation.

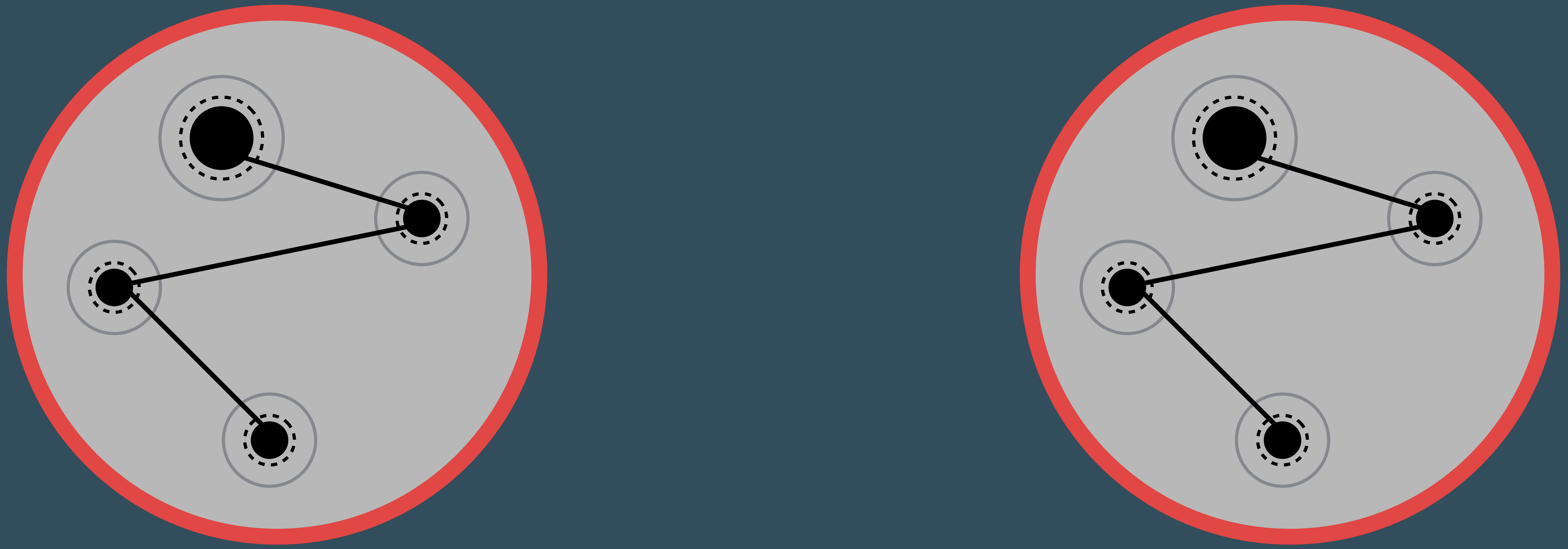


Strategic Design

Context Map – Patterns

- Shared Kernel
- Customer / Supplier
- Conformist
- Anticorruption Layer
- Separate Ways
- Open / Host Service
- Published Language

There is no connection between the bounded contexts of a system. This allows teams to find their own solutions in their domain.

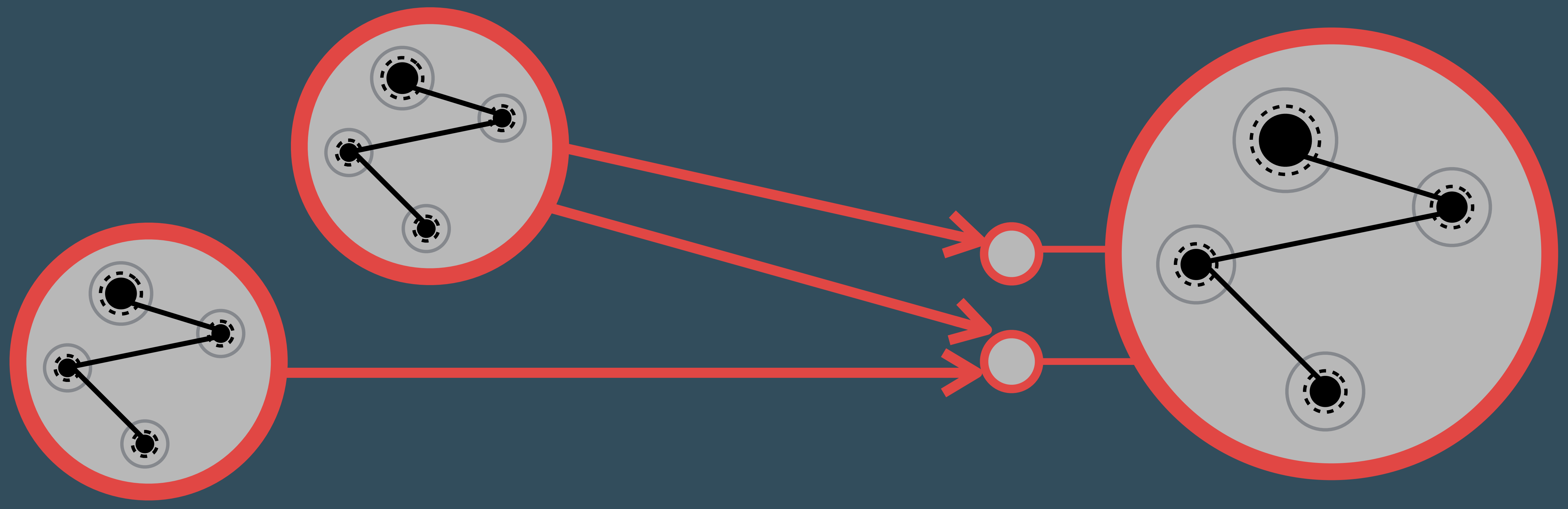


Strategic Design

Context Map – Patterns

- Shared Kernel
- Customer / Supplier
- Conformist
- Anticorruption Layer
- Separate Ways
- Open / Host Service
- Published Language

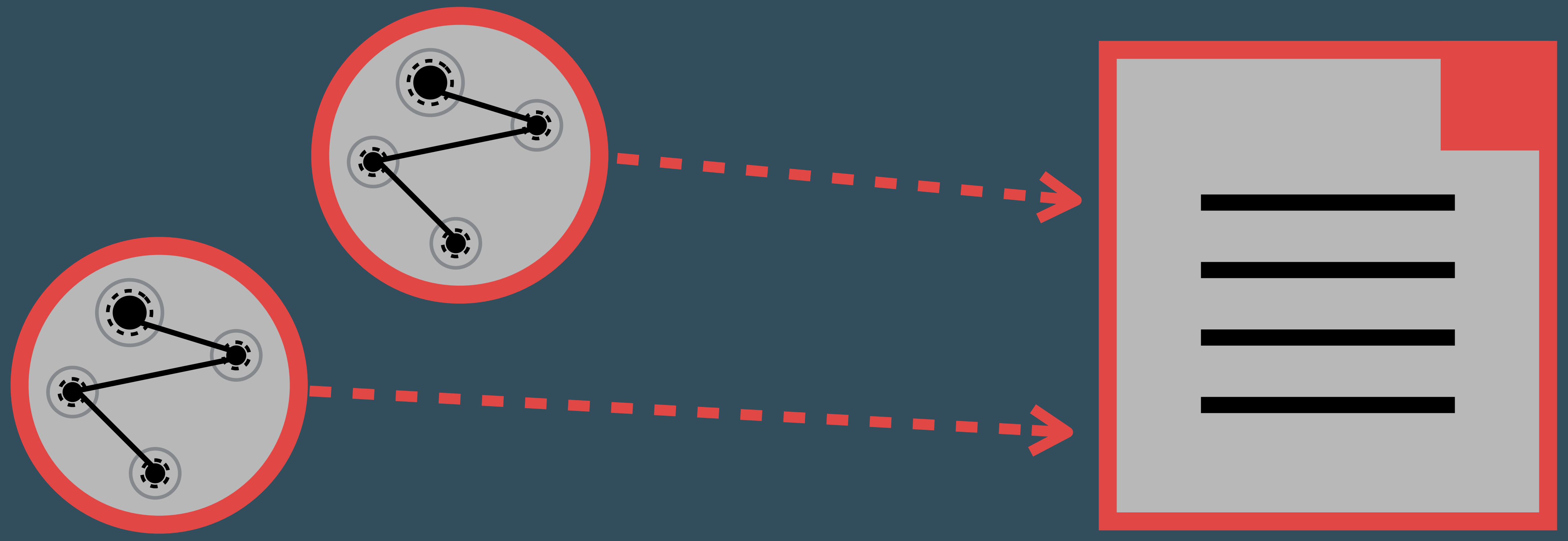
Each Bounded Context offers a defined set of services that expose functionality for other systems. Any downstream system can then implement their own integration. This is especially useful for integration requirements with many other systems.



Context Map – Patterns

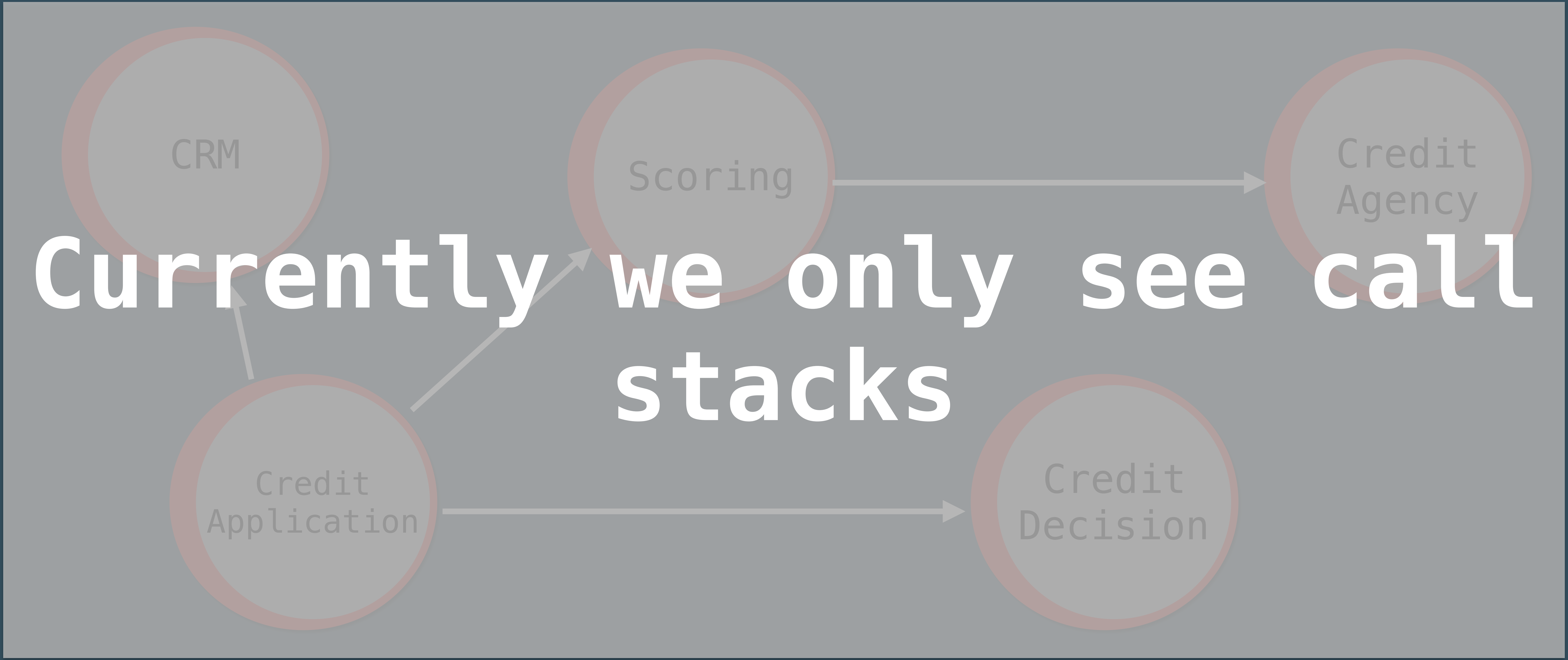
- Shared Kernel
- Customer / Supplier
- Conformist
- Anticorruption Layer
- Separate Ways
- Open / Host Service
- Published Language

Published Language is quite similar to Open / Host Service. However it goes as far as to model a Domain as a common language between bounded contexts.



Strategic Design

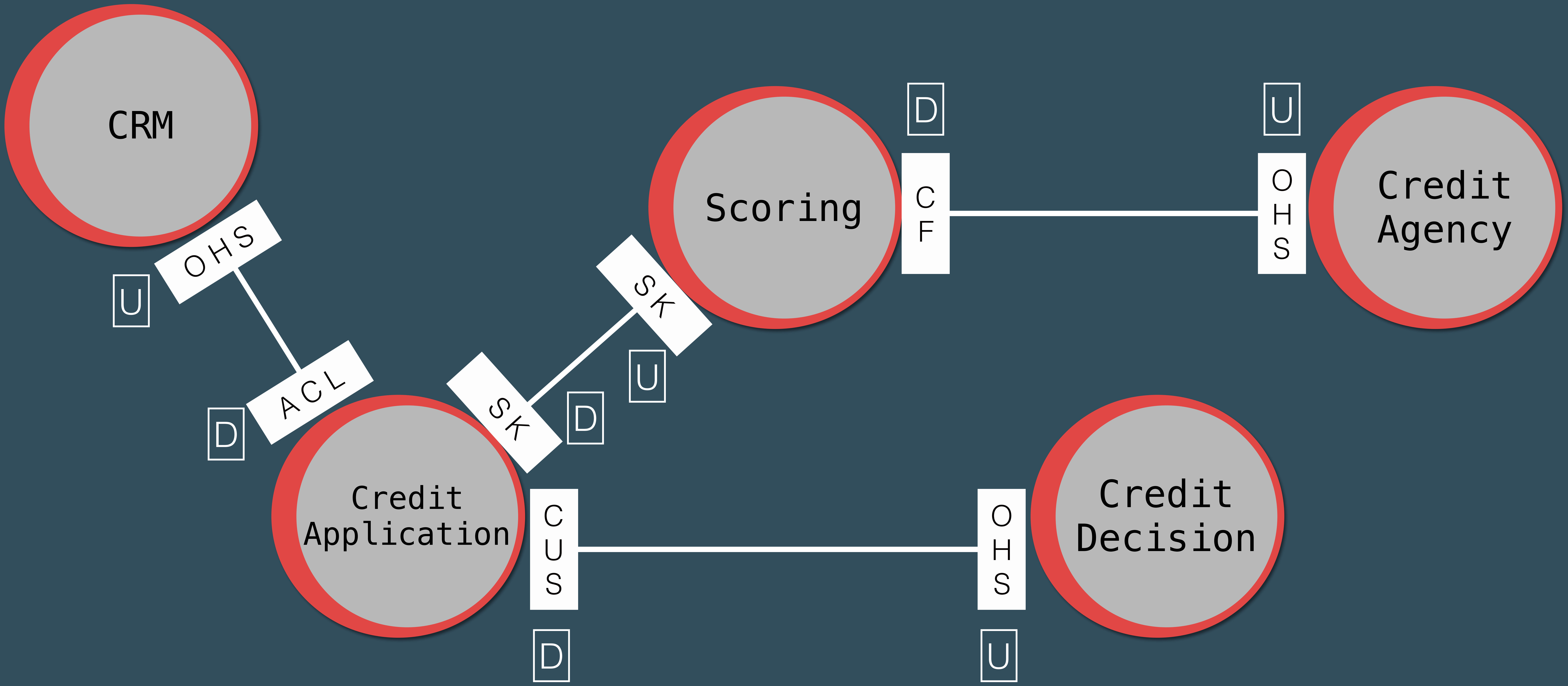
Context Map – Why?



Currently we only see call stacks

Strategic Design

Context Map



Strategic Design

Example Spring Boot App

https://github.com/mploed/ddd-strategic-design-spring-boot

THANK YOU!

Michael Plöd - innoQ
@bitboss

Contact me for DDD Trainings / Consulting