Domain-Driven Design

A Gentle Introduction into Challenging Topic

Asher Sterkin
asher.sterkin@gmail.com

Clean Code Meetup
January 2, AppsFlyer

About DDD IL

Vladik Khononov

Erik Ashepa

Asher Sterkin

- White belt circle: spread the word about DDD
- Black belt circle: apply DDD to real problems

Feel free to reach out if you want to join any of them

About Myself

- Software technologist/architect
- Almost 40 years in the field
- Former VP Technology @ NDS and Distinguished Engineer @ Cisco
- Currently CTO @ IRKI
- C-level mentoring on software strategy
- Cross-discipline approach (connecting the dots):
  - (Strategic) Domain-Driven Design
  - Serverless Architecture
  - Cynefin
  - Wardley Maps
  - Lean Startup
  - Promise Theory
  - ...

Disclaimer

- Just an overview of the main DDD concepts
- All examples are purely illustrative
- No architectural or technological recommendations
- All wording is mine, as well as mistakes

Domain-Driven Design is More Relevant than Ever

- Software infrastructure is rapidly commoditized
- Focus on delivering business value “yesterday”
- AI, AR, VR, 3DP, IoT, Blockchain spiral complexity up
- Need a practical method to keep complexity under control
- So far DDD has an unsurpassed record in this area

Domain-Driven Design Acronym

L

M

B

N

Pronounced LA-M-BA-N (invention is mine)

Domain-Driven Design at a Glance

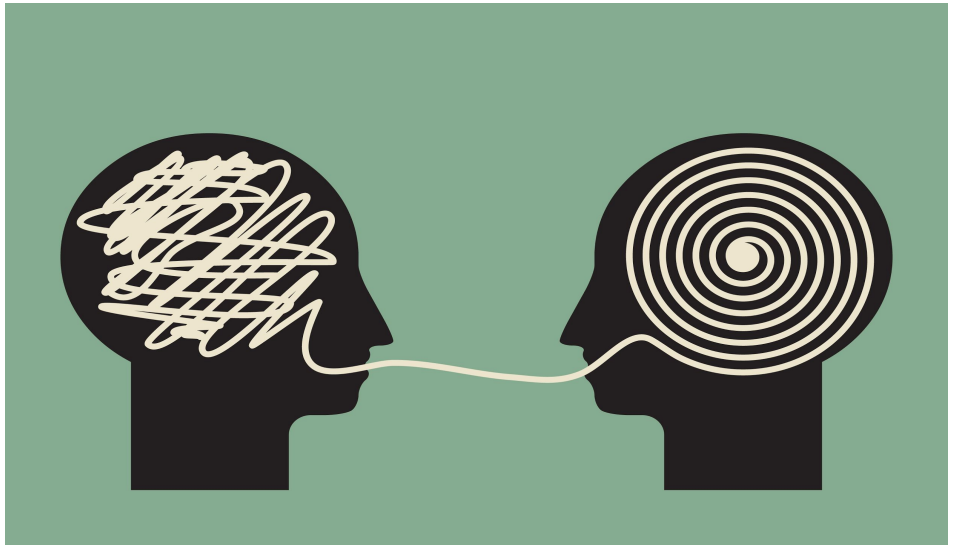
Language

Model

Boundaries

Nesting

Language



- Common (ubiquitous) language for domain and software experts
- Vocabulary reflected *directly* in software
- Normally a subset of domain-specific jargon

DDD is essentially a *linguistic* process

Model

- Language elements have meaning (*semantics*)
- Semantics is captured in a form of *formal model*:
  - Object-Oriented
  - Statecharts
  - Functional
  - Relational
  - Graph
  - Mathematical
  - Neural Networks
  - …

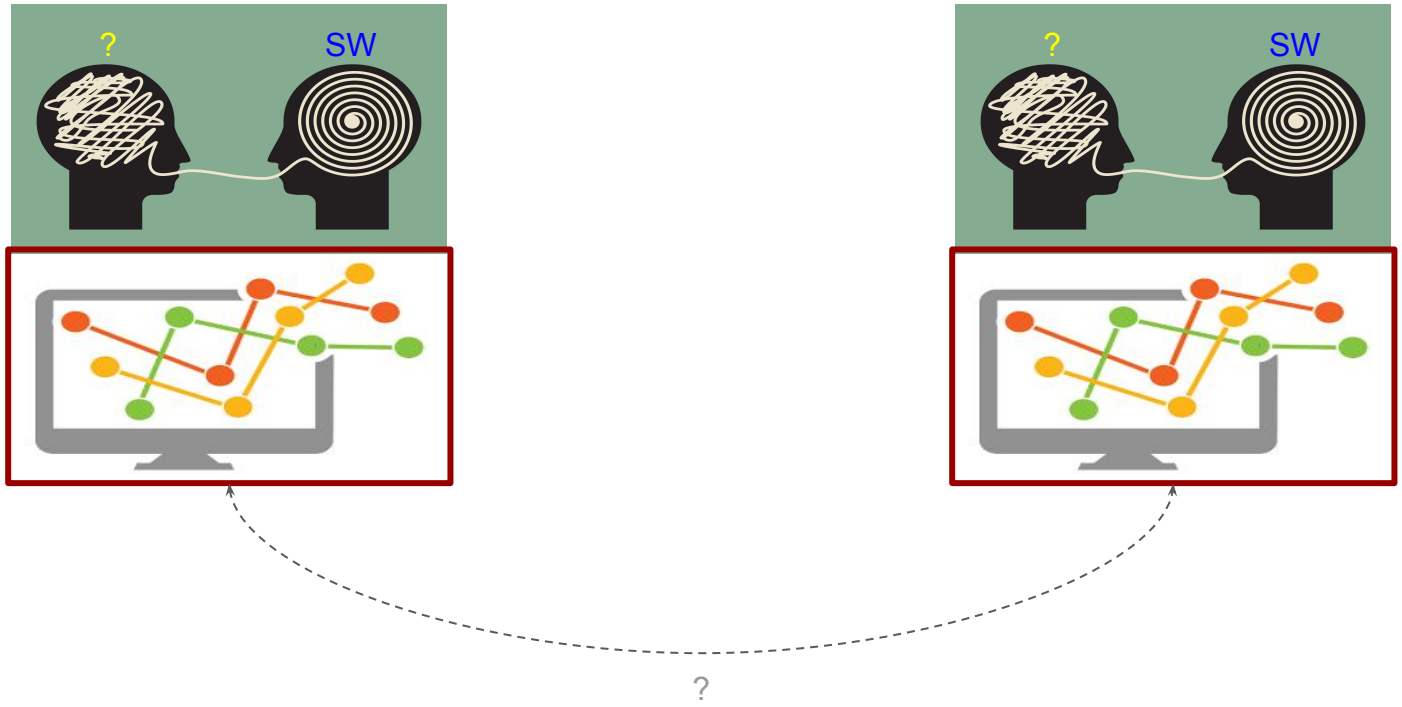
DDD is also a *modeling* process

Boundaries

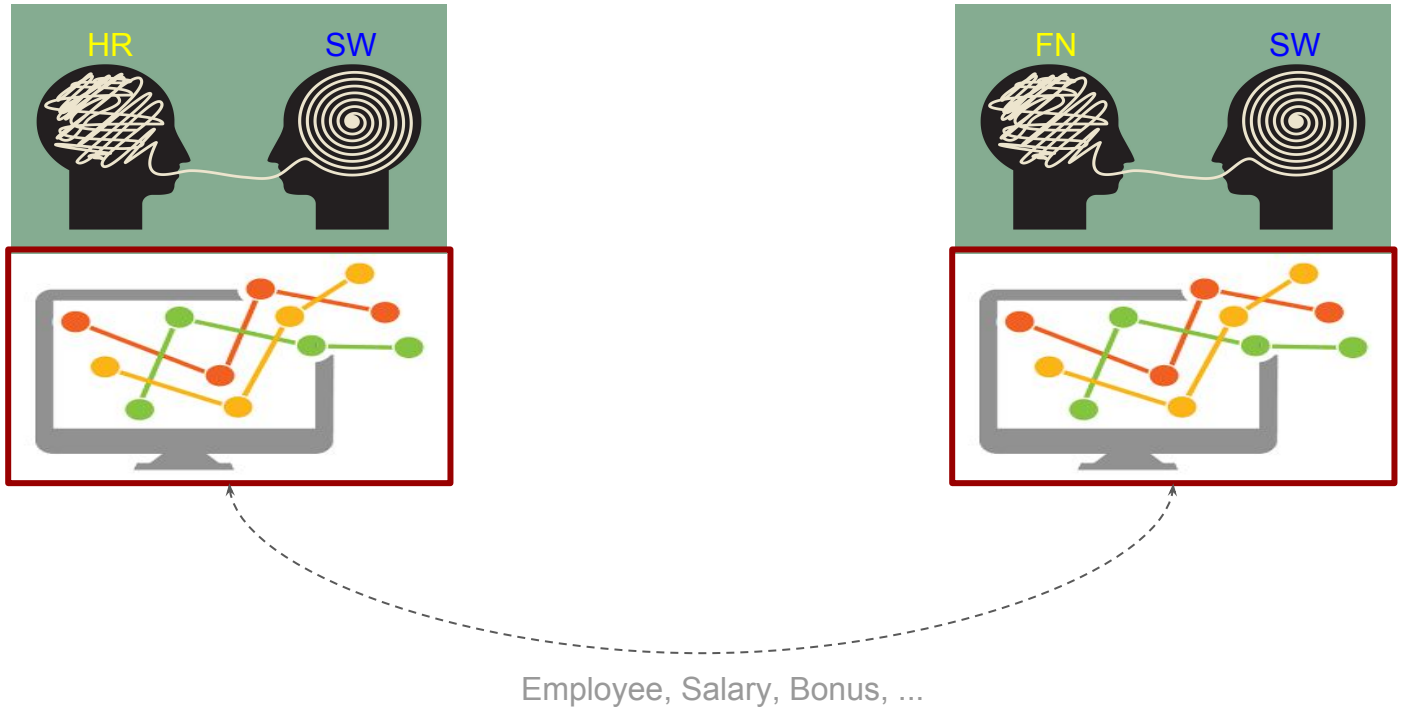
- “One size fits all” models are impractical
- G-d like models result in Big Ball of Mud
- DDD insists on identifying multiple models:
  - Coherent within their own boundaries
  - Mapped onto others where appropriate

DDD is about maintaining boundaries for- and mappings between- models

Example



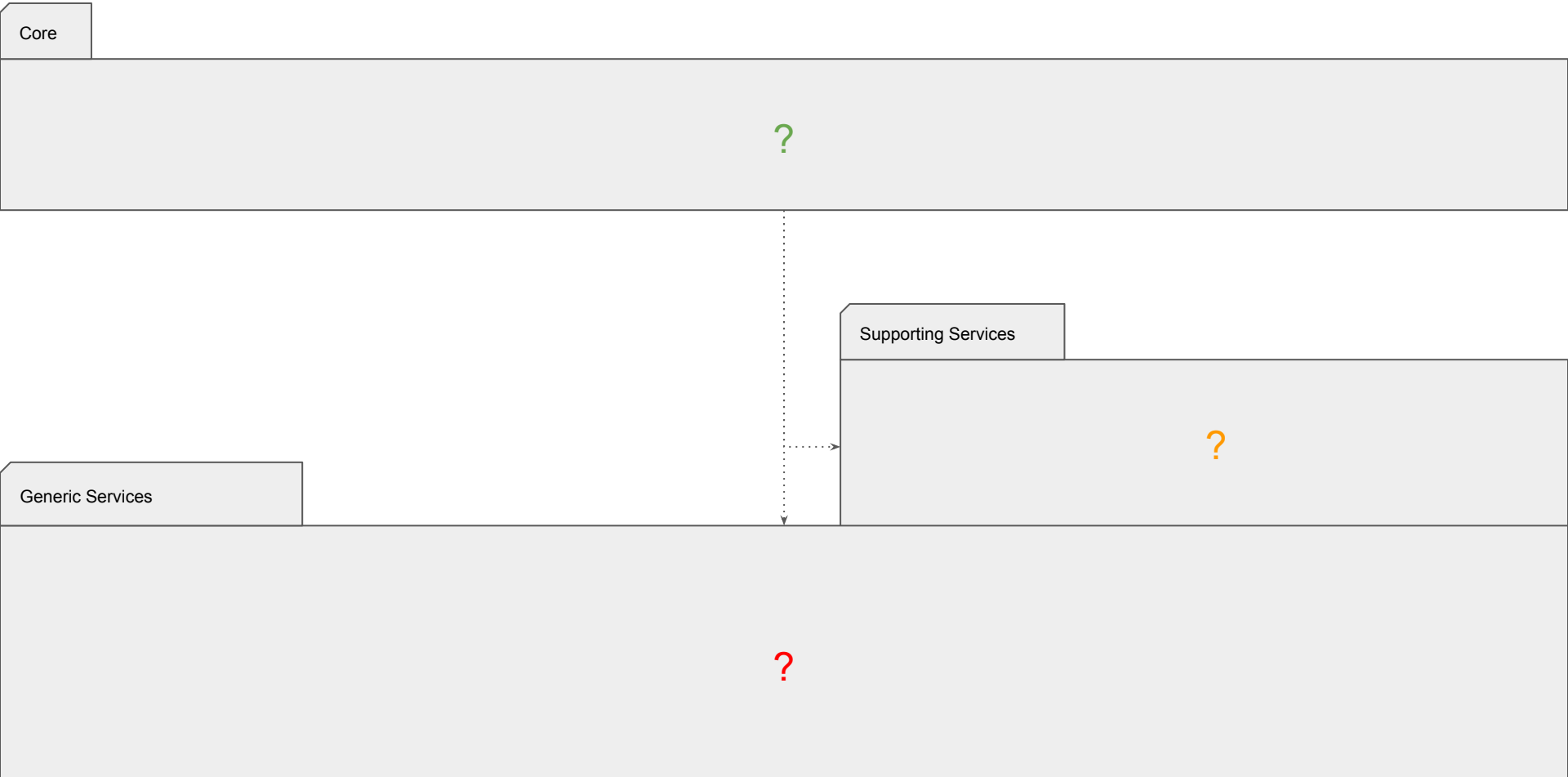
Example



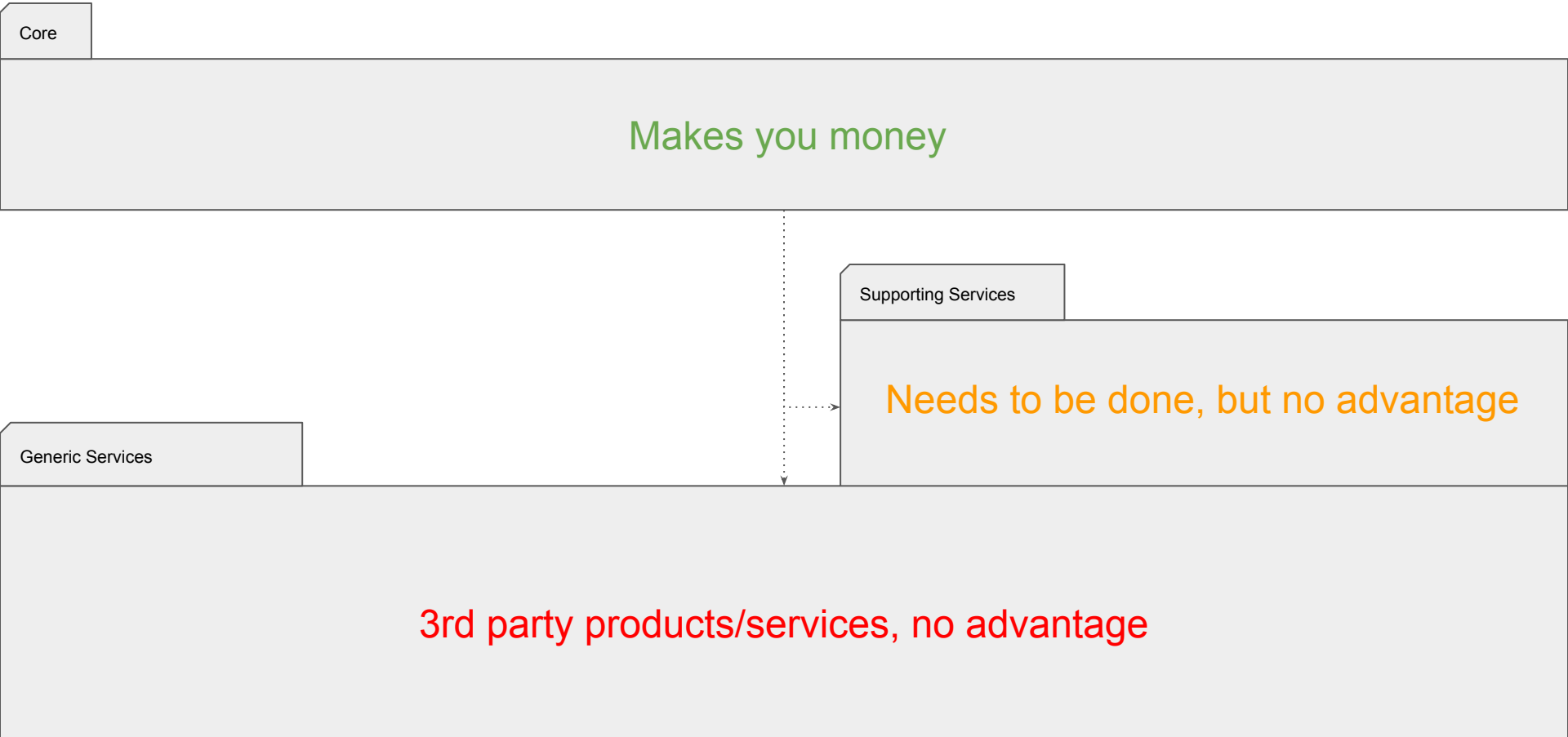
Nesting

- Boundaries exist at different levels of granularity
- Within DDD approach the following levels of granularity are considered:
  - Conceptual boundaries (sub-domains)
  - Application and Domain Services
  - Bounded Context
  - Stateless vs Stateful Elements
  - Aggregate
  - Entity
  - Value
- It’s possible to stop at any level and still have some value
- Not all parts of the system will be designed equally well!

Sub-domains



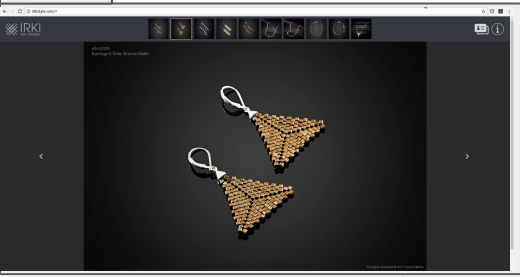
Sub-domains (popular version)



Example (my approach)

Strategic Impact: build, buy, contribute

Core



Essense of my business

push down
pull up

Supporting Services

Green Invoice

...

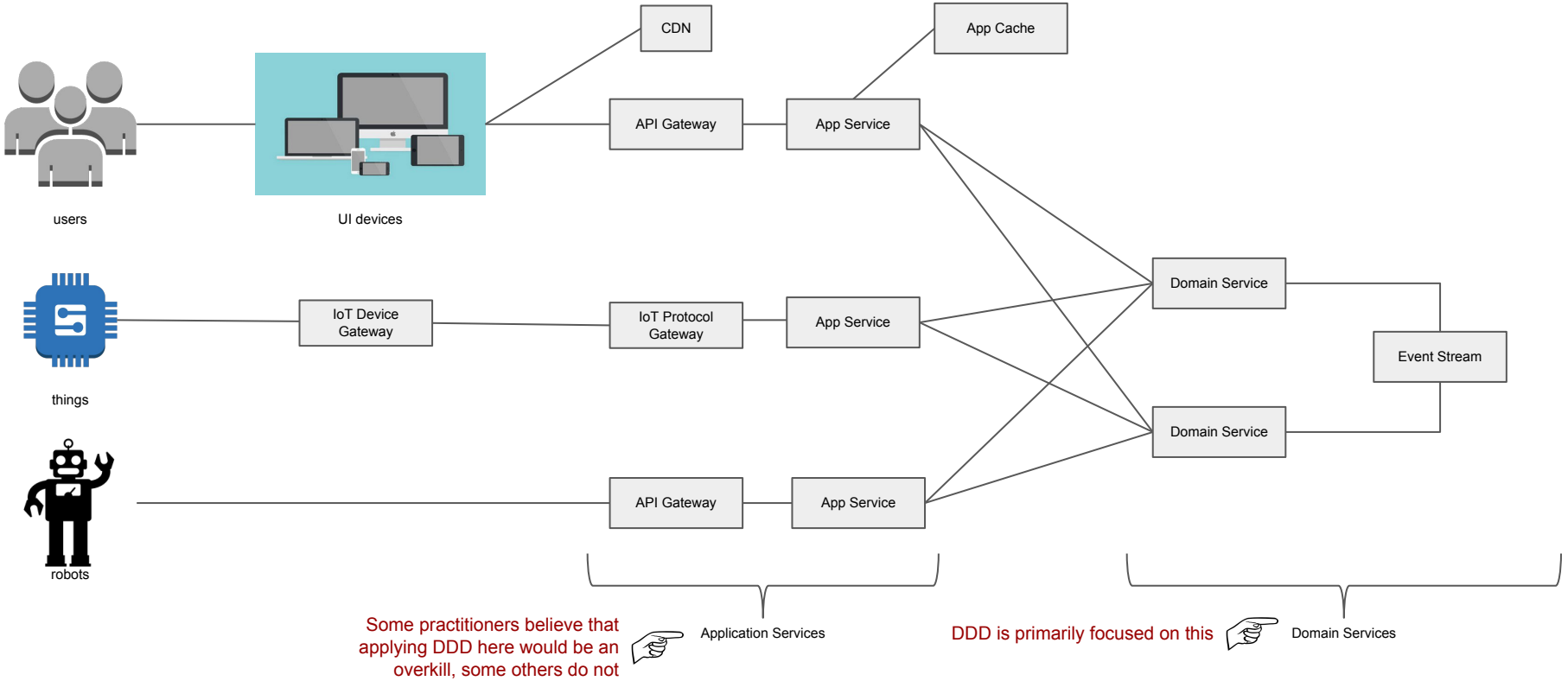
Common for group of domains

Generic Services

...

Generic for any domain

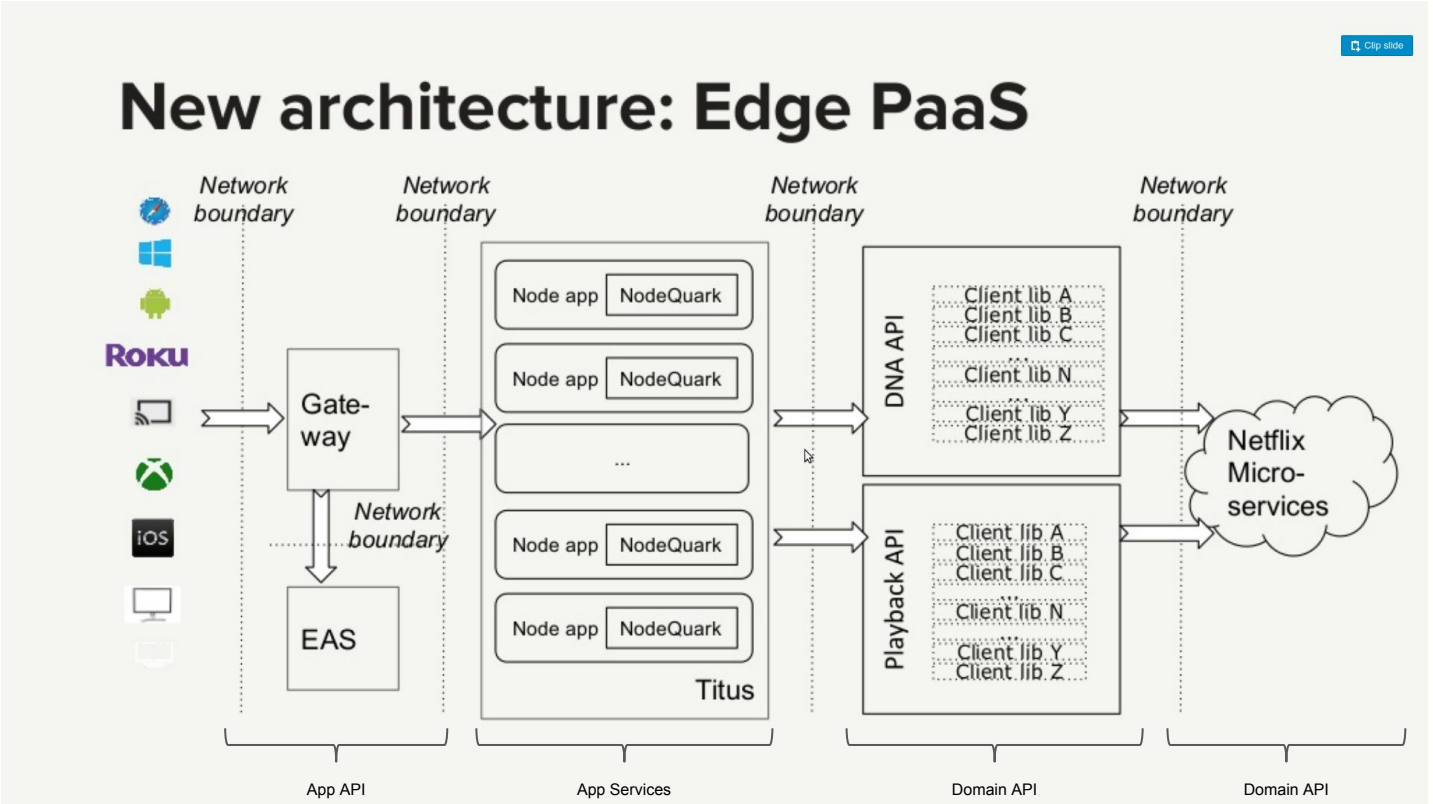
Application vs Domain Services (oversimplified)



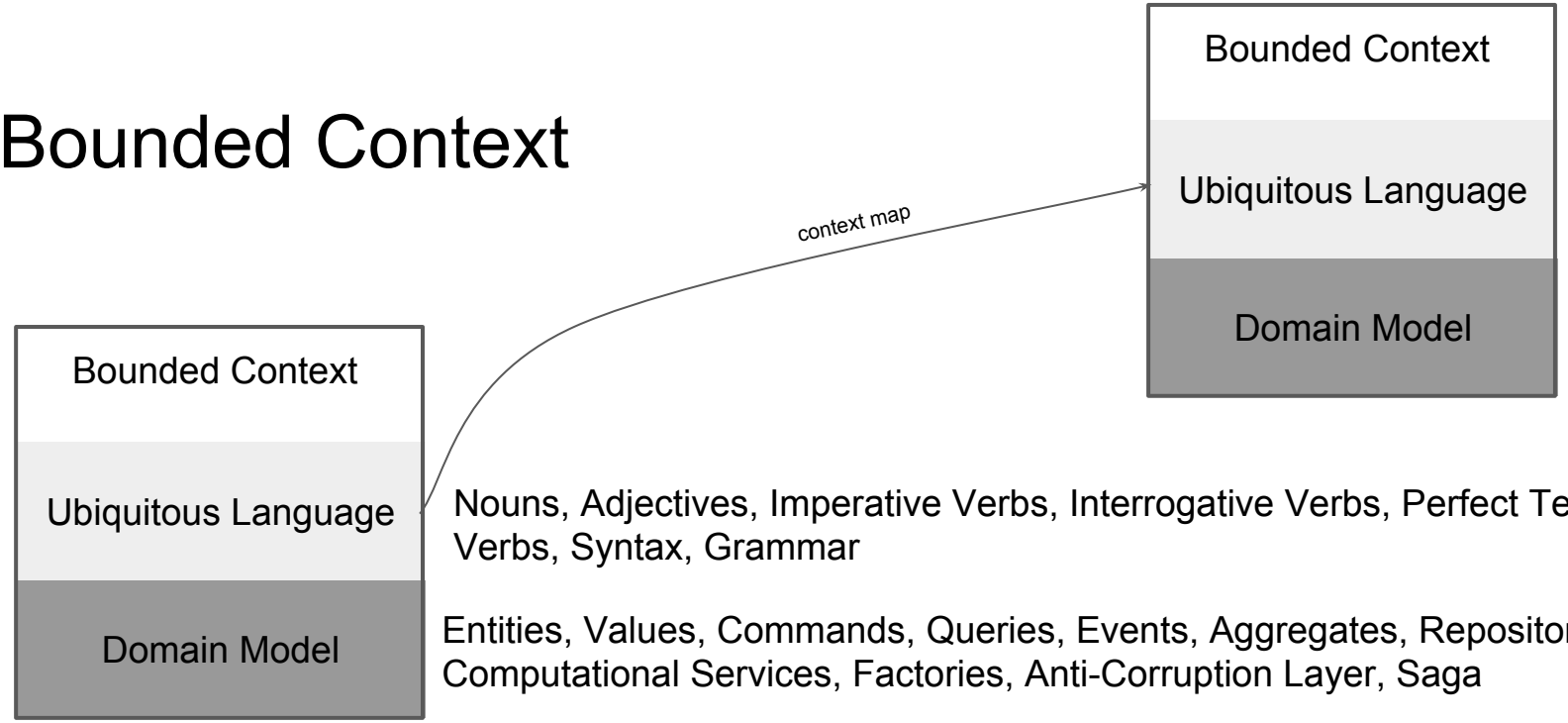
Example

Clients	App Services	Domain Services
?	?	?

Textbook Example: Netflix API



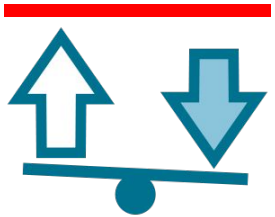
Bounded Context



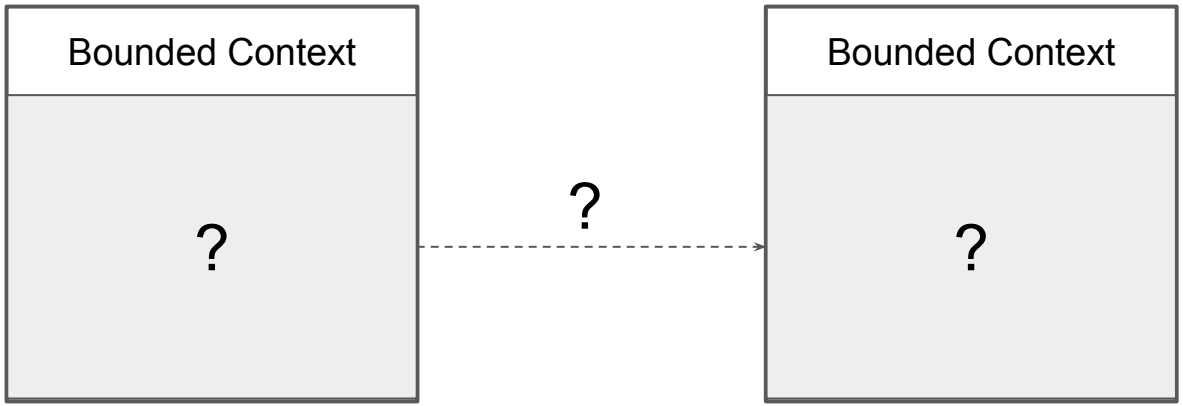
Nouns, Adjectives, Imperative Verbs, Interrogative Verbs, Perfect Tense Verbs, Syntax, Grammar

Entities, Values, Commands, Queries, Events, Aggregates, Repository, Computational Services, Factories, Anti-Corruption Layer, Saga

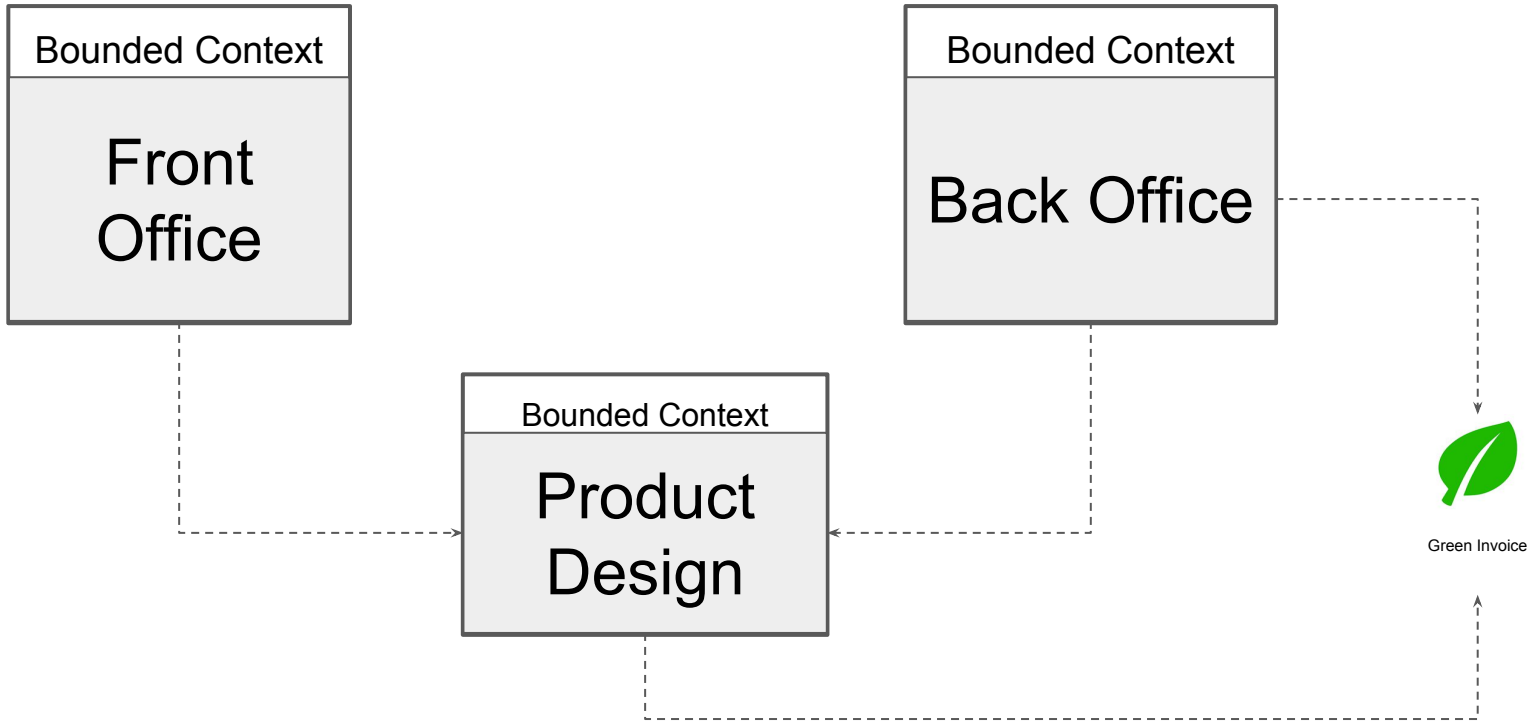
- Linguistic boundaries
- Reflect commitment to maintain strong cohesion inside and loose coupling outside
- Inside Bounded Context the model could evolve independently
- Defines a Minimal Meaningful scope with Maximum Acceptable Risk of not having semantic consistency across different parts of the system (not too much, not too little)



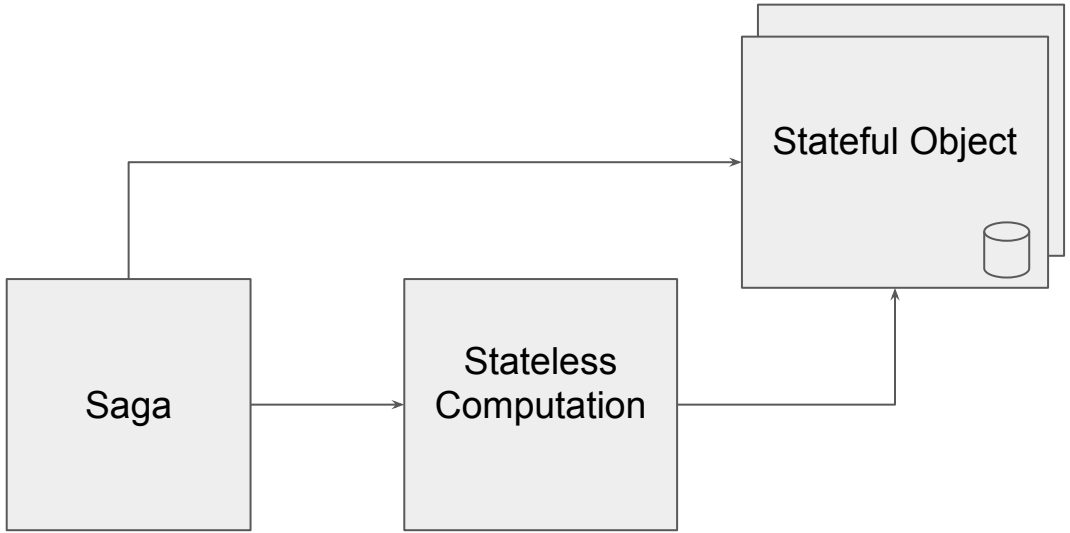
Example



Example (my system at the moment)



Stateless vs Stateful

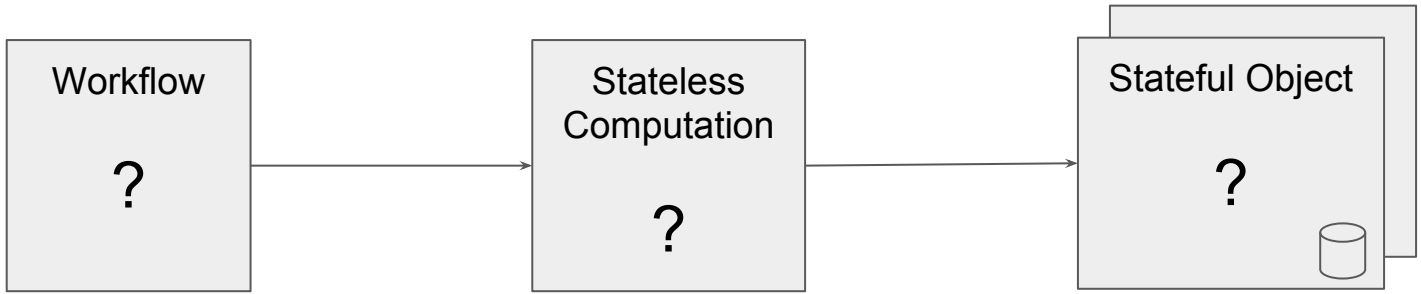


Stateful Objects introduced when managing State is a part of the Domain Service contract

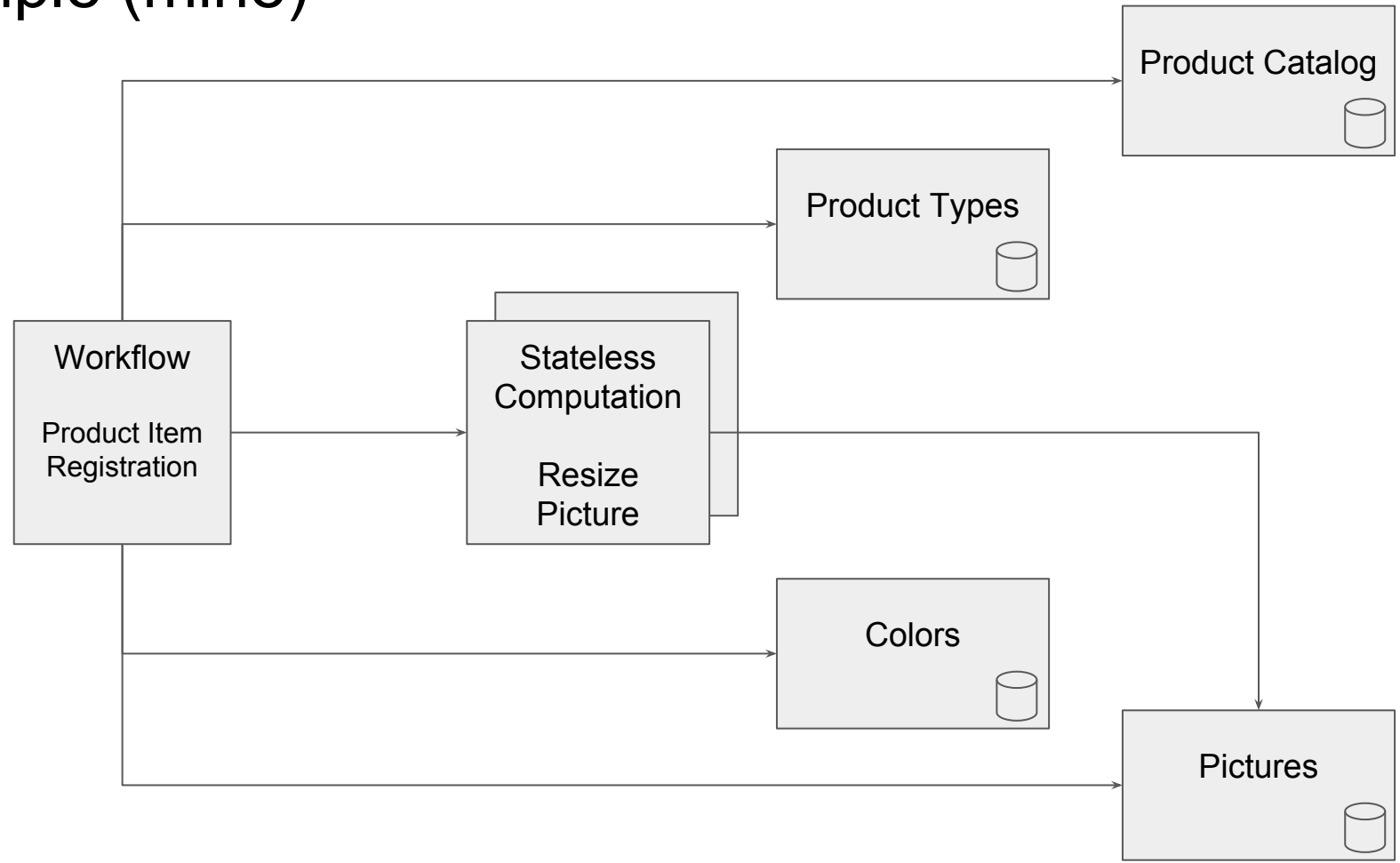
Implements a multi-step workflow. Needs state to keep track of progress and run compensating transactions in the case of failure

Performs computation which does not naturally fit into any stateful object. In particular could be a Factory for some Stateful Object with complex creation process

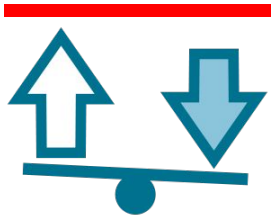
Example

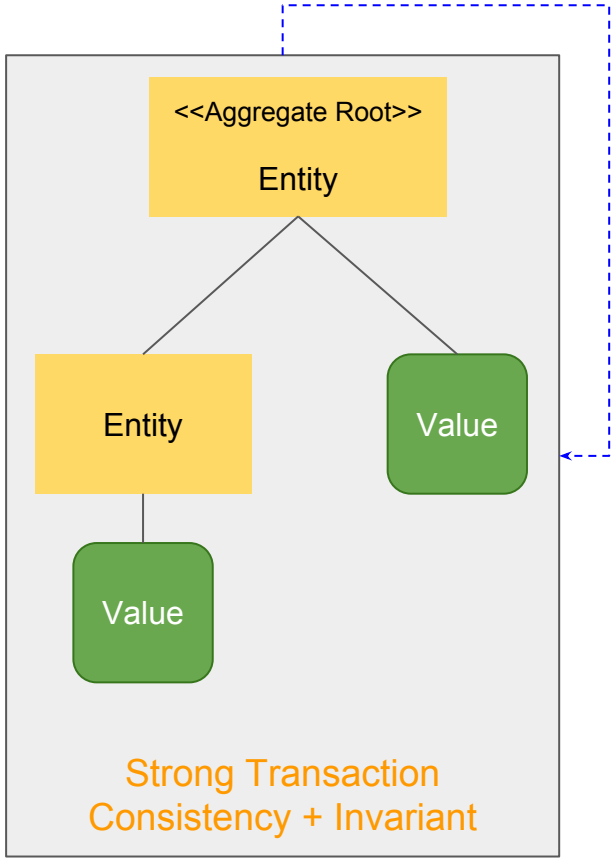


Example (mine)



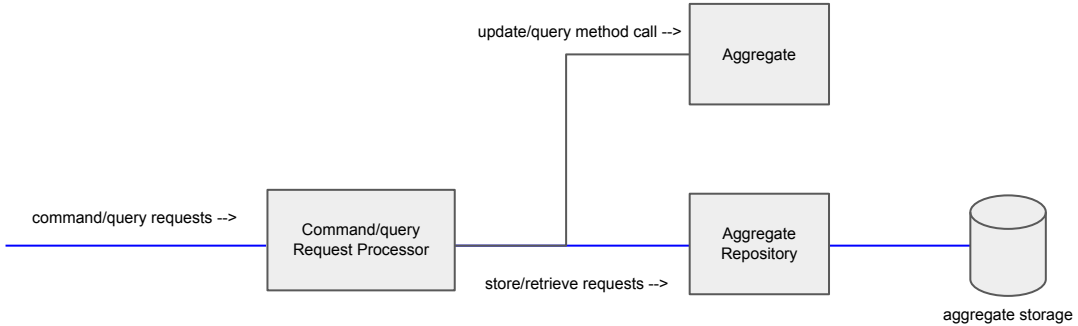
DDD Aggregate and Repository



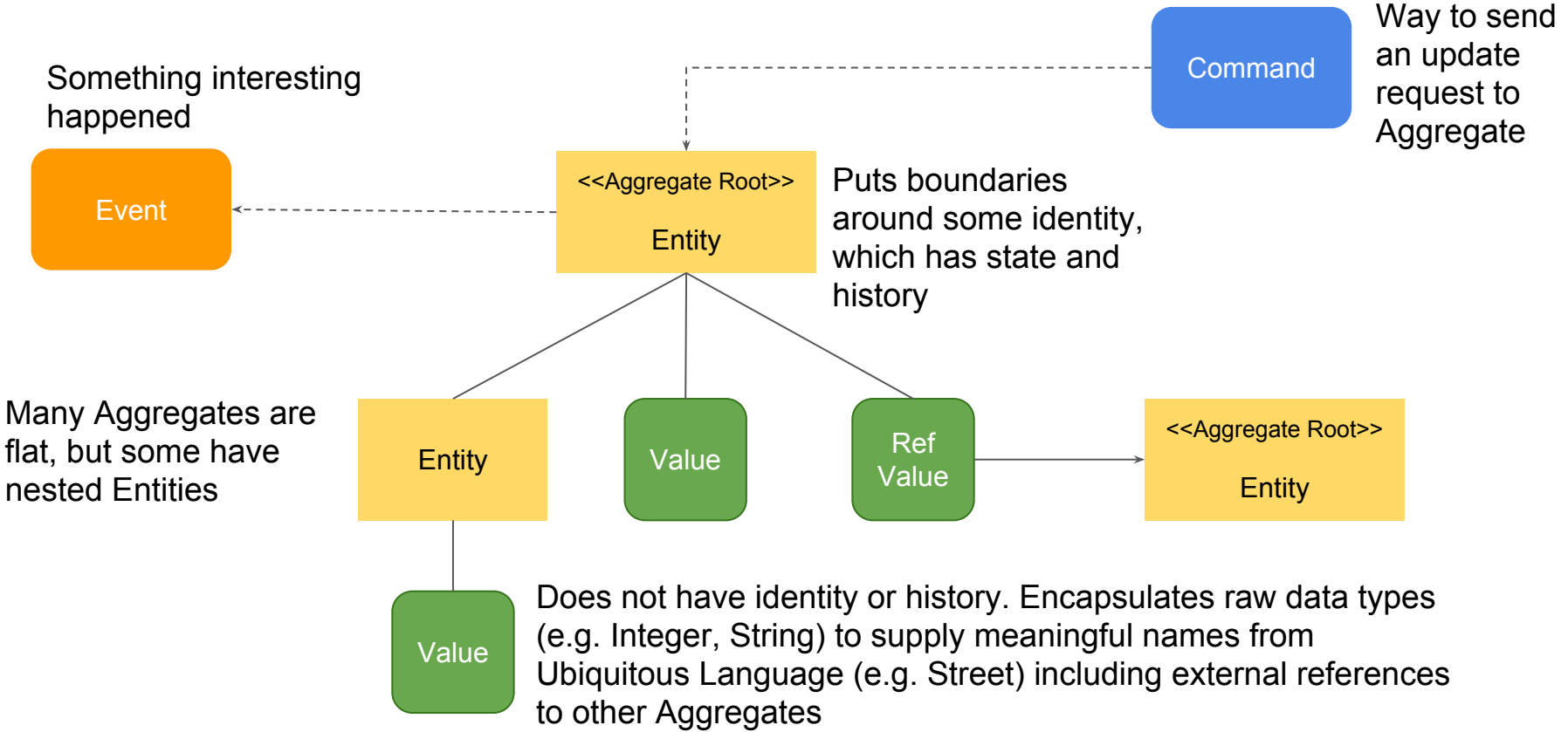


Eventual
Transaction
Consistency

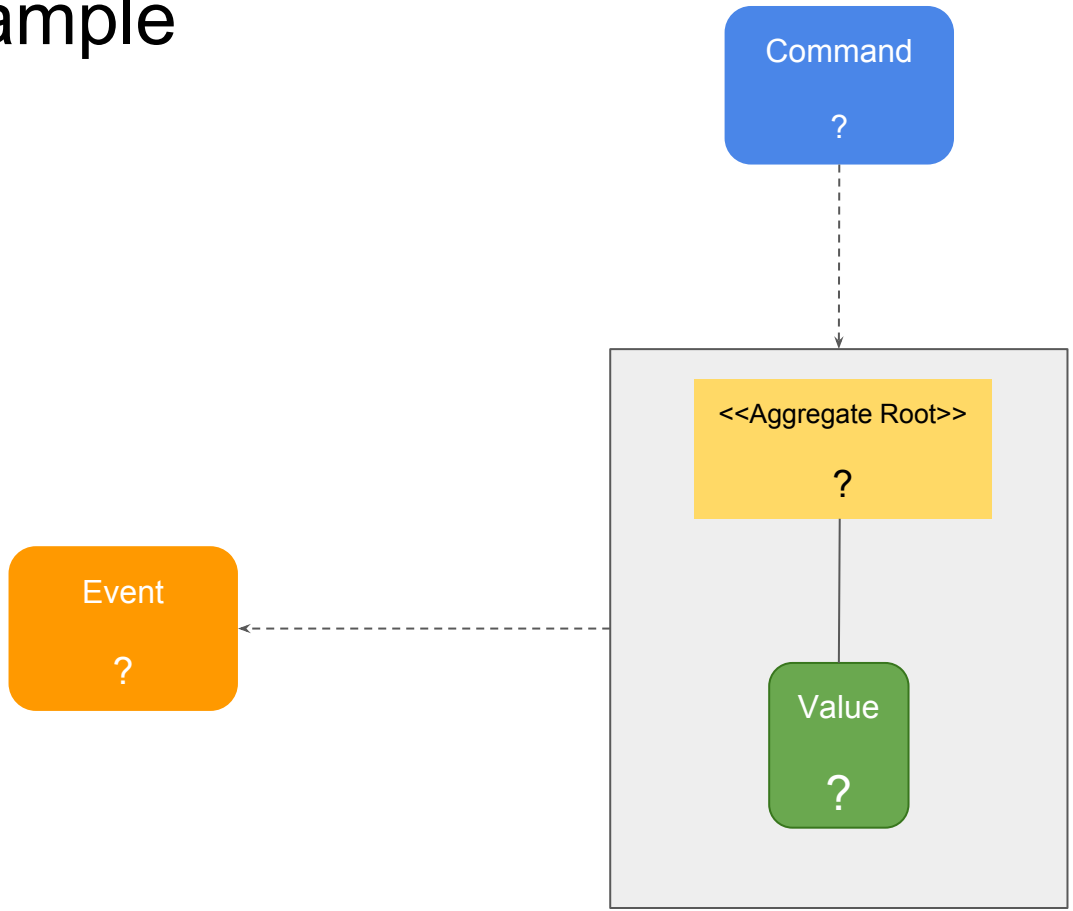
- Data integrity boundaries
- Reflects commitment to maintain strong transactional consistency inside and eventual consistency outside
- Inner parts are accessible only via Aggregate Root
- Inside Aggregate some form of invariant is maintained
- Aggregates of the same type are stored within Repository
- Defines a Minimal Meaningful transactional scope with Maximum Acceptable Risk level of not having immediate data consistency across different parts of the system (not too much, not too little)



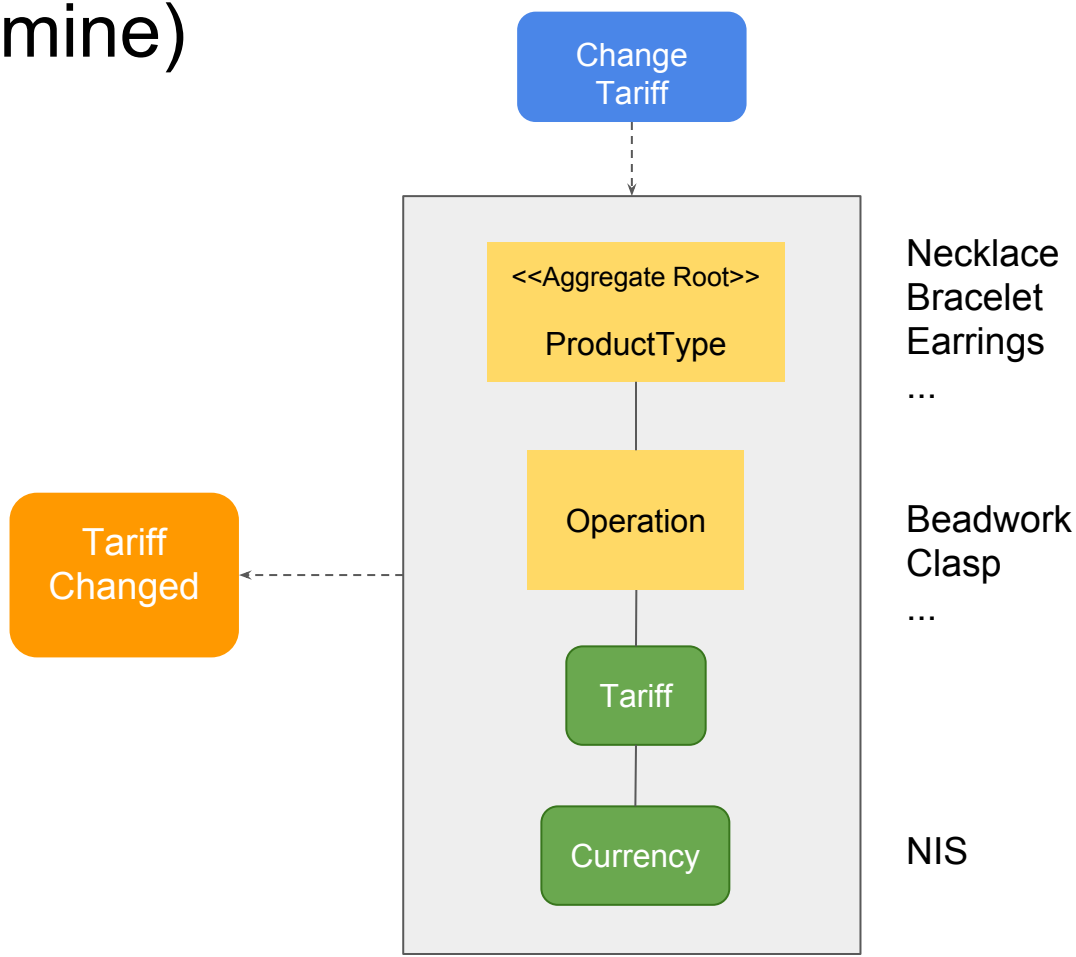
DDD Entity, Value, Event, and Command



Example



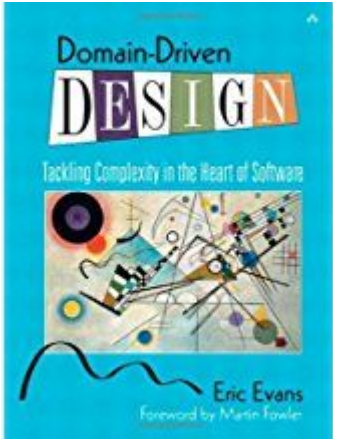
Example (mine)

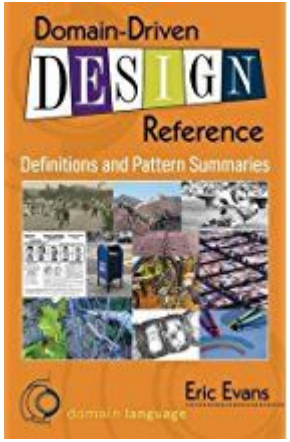


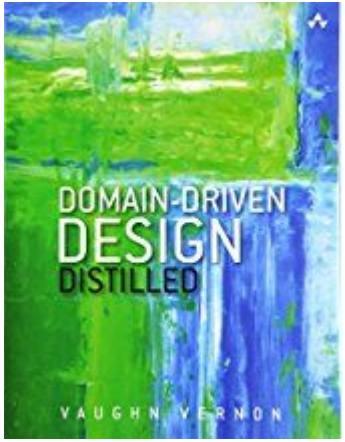
Advanced Topics not Covered Here

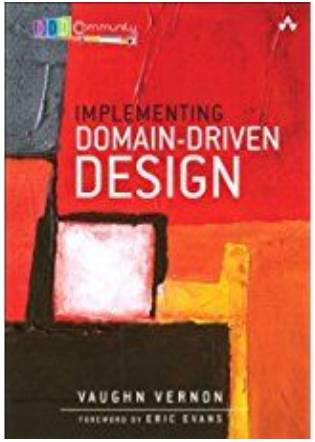
- Strategic DDD Patterns
- Context Mapping
- Command-Query Request Segregation (CQRS)
- Event Sourcing (ES)
- Dealing with Legacy Systems
- Event Storming
- DDD and Microservices (Erik’s talk)
- Cloud-native and Serverless DDD (area of my research)
- DDD and Machine/Deep Learning
- DDD and IoT
- DDD and VR/AR
- ...

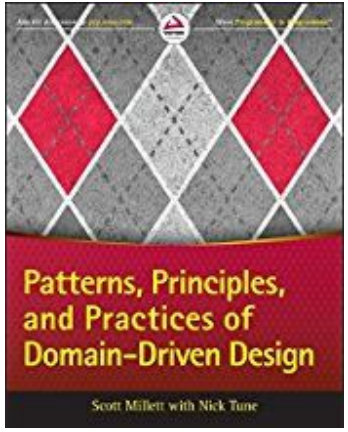
DDD Bookshelf

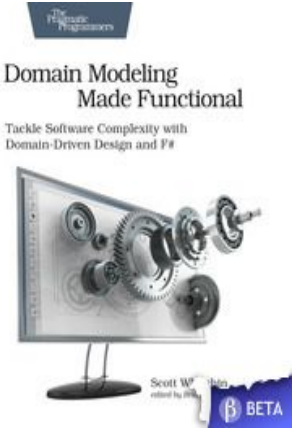


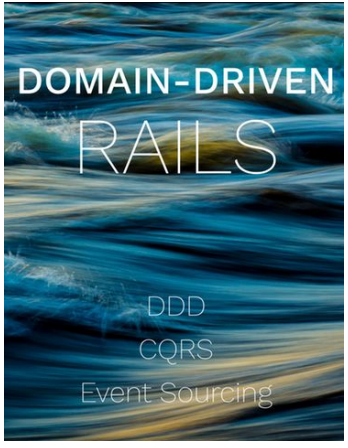


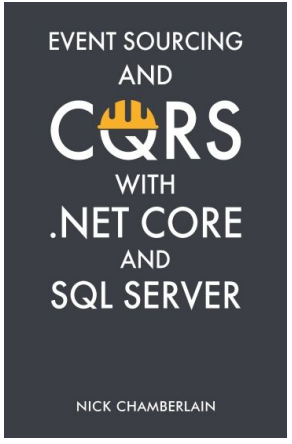












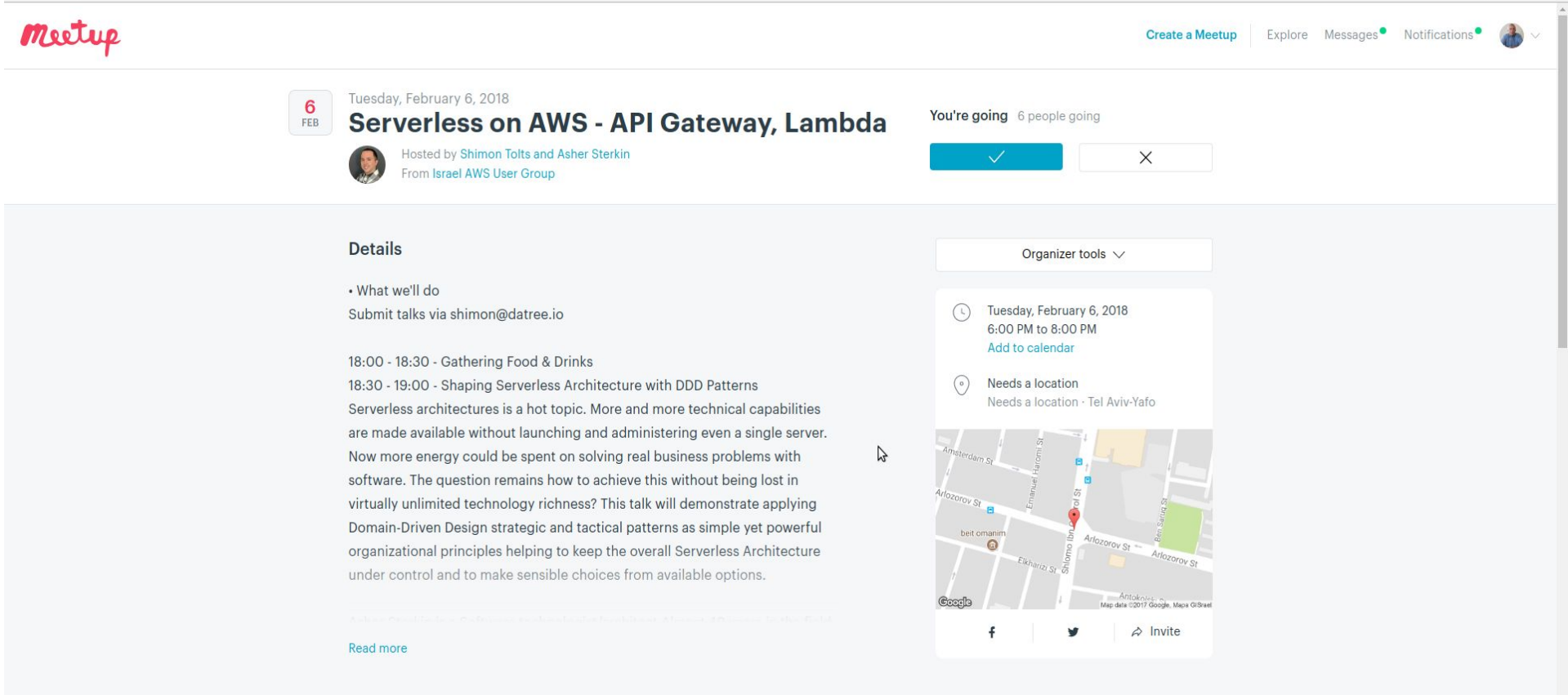
DDD Video Library

- Eric Evans: Tackling Complexity in the Heart of Software
- Eric Evans: Good Design is Imperfect Design
- Eric Evans: DDD & Microservices: At Last, Some Boundaries!
- Eric Evans: Acknowledging CAP at the Root -- in the Domain Model
- Many, many more on YouTube and Vimeo
-

DDD Online

- Eric Evans' site: https://domainlanguage.com/
- Domain-Driven Design on InfoQ: https://www.infoq.com/domain-driven-design
- Long curated list at https://github.com/heynickc/awesome-ddd
- Vladik's Blog at http://vladikk.com/
-

A Shameless Promotion




ANY
QUESTIONS
?