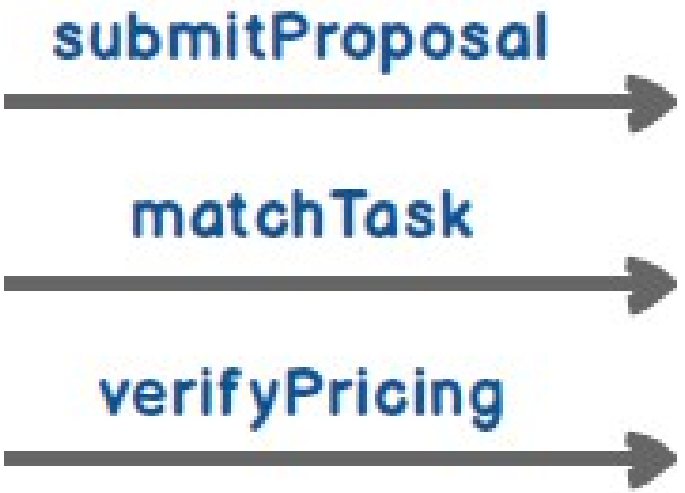
# Reactive DDD

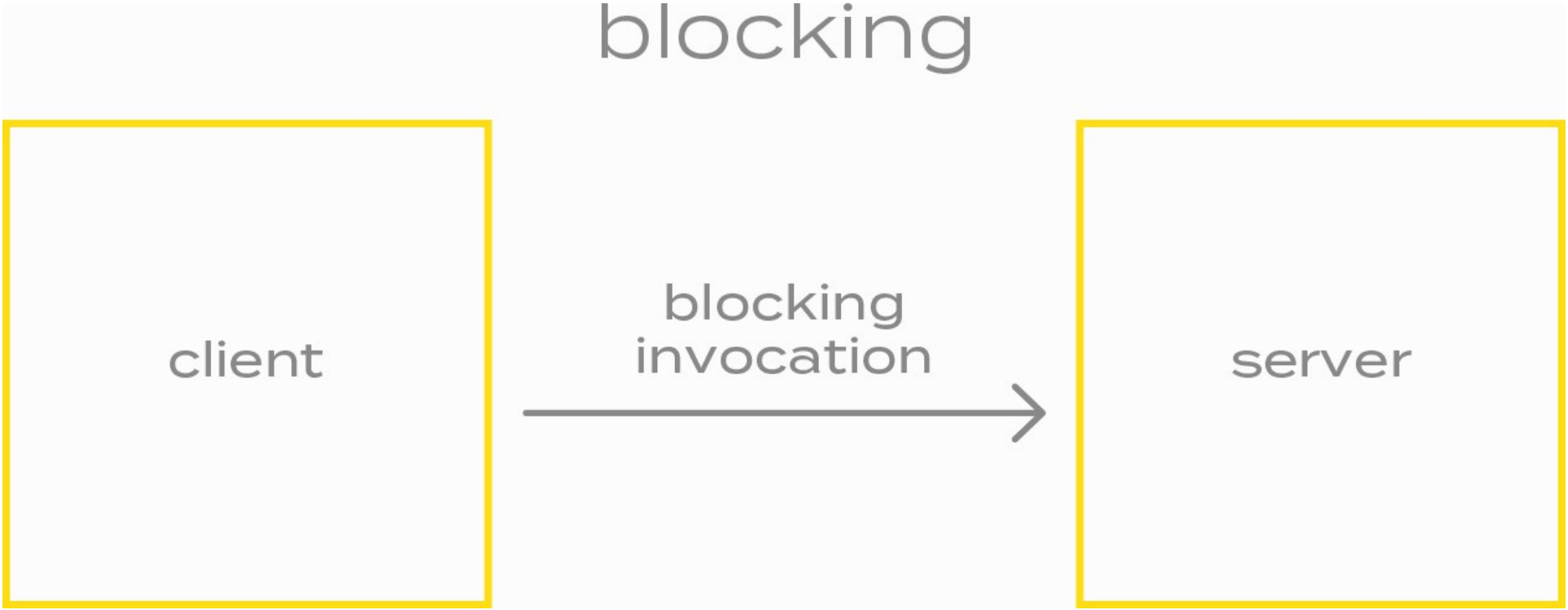
## When Concurrent Waxes Fluent




KALELE
LEARN. DESIGN. EXCEL.

vlingo

# Where We Are



# Blocking

Behavior
HTTP Request-Response
Database
...

# Anemic Domain Model

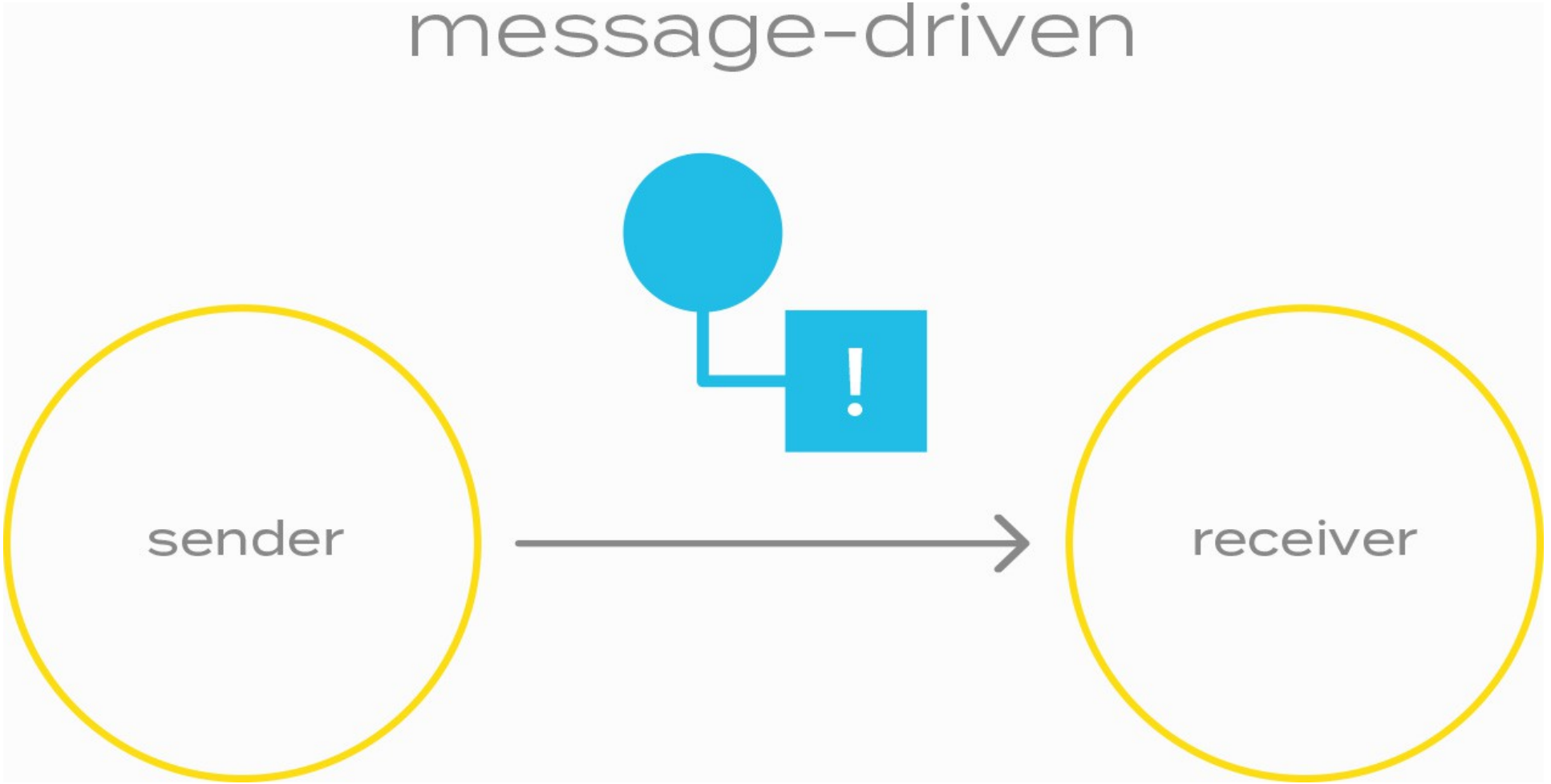
@VaughnVernon

# Anemic Domain Model

```
@Entity
public class Client {
   @Id
   private String id;
   @Column
   private String name;
   @Column
   private String address1;
   @Column
   private String address2;
   @Column
   private String city;
   @Column
   private String stateOrProv;
   ...
}
```

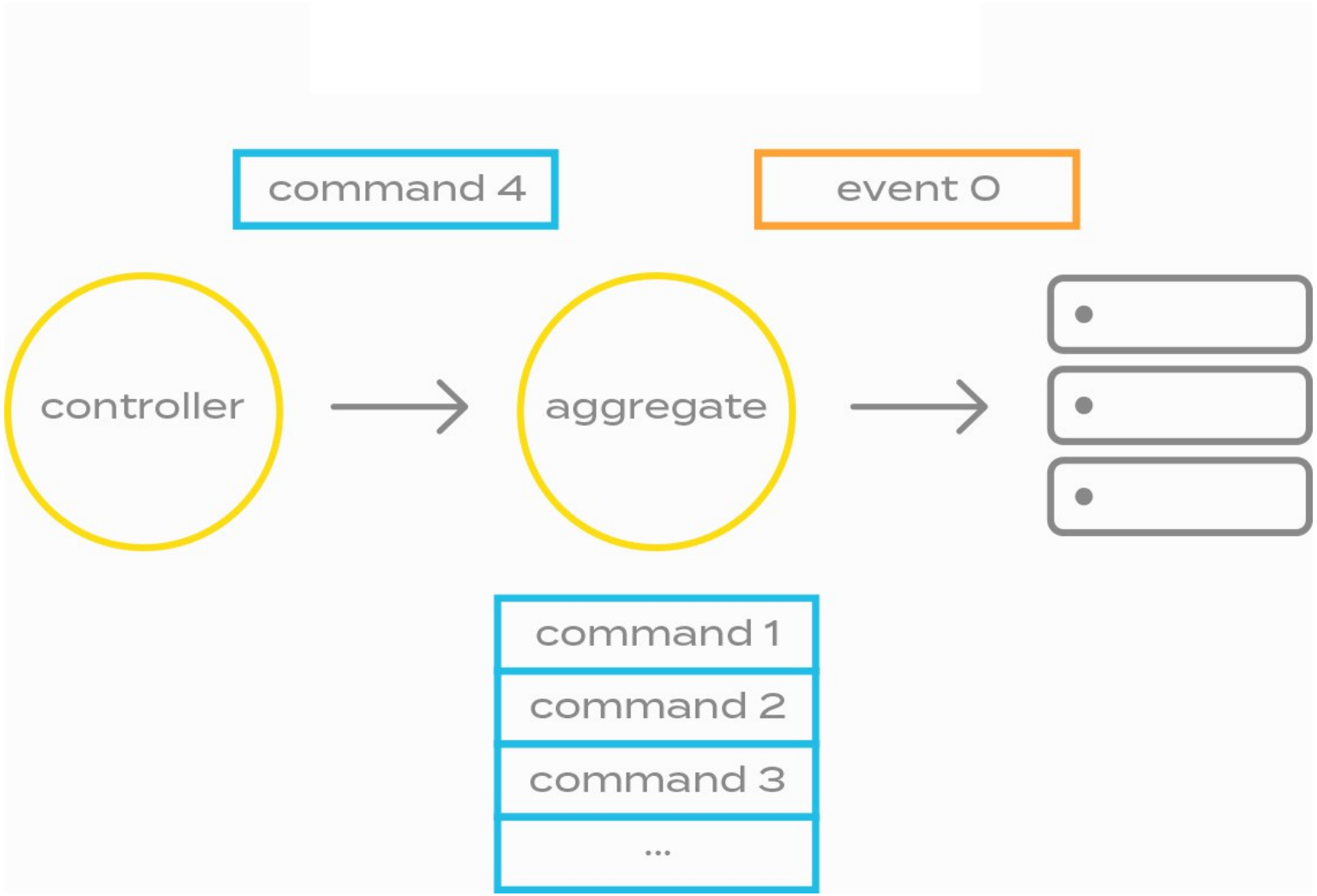
@VaughnVernon

# Going To




@VaughnVernon

# Event Driven



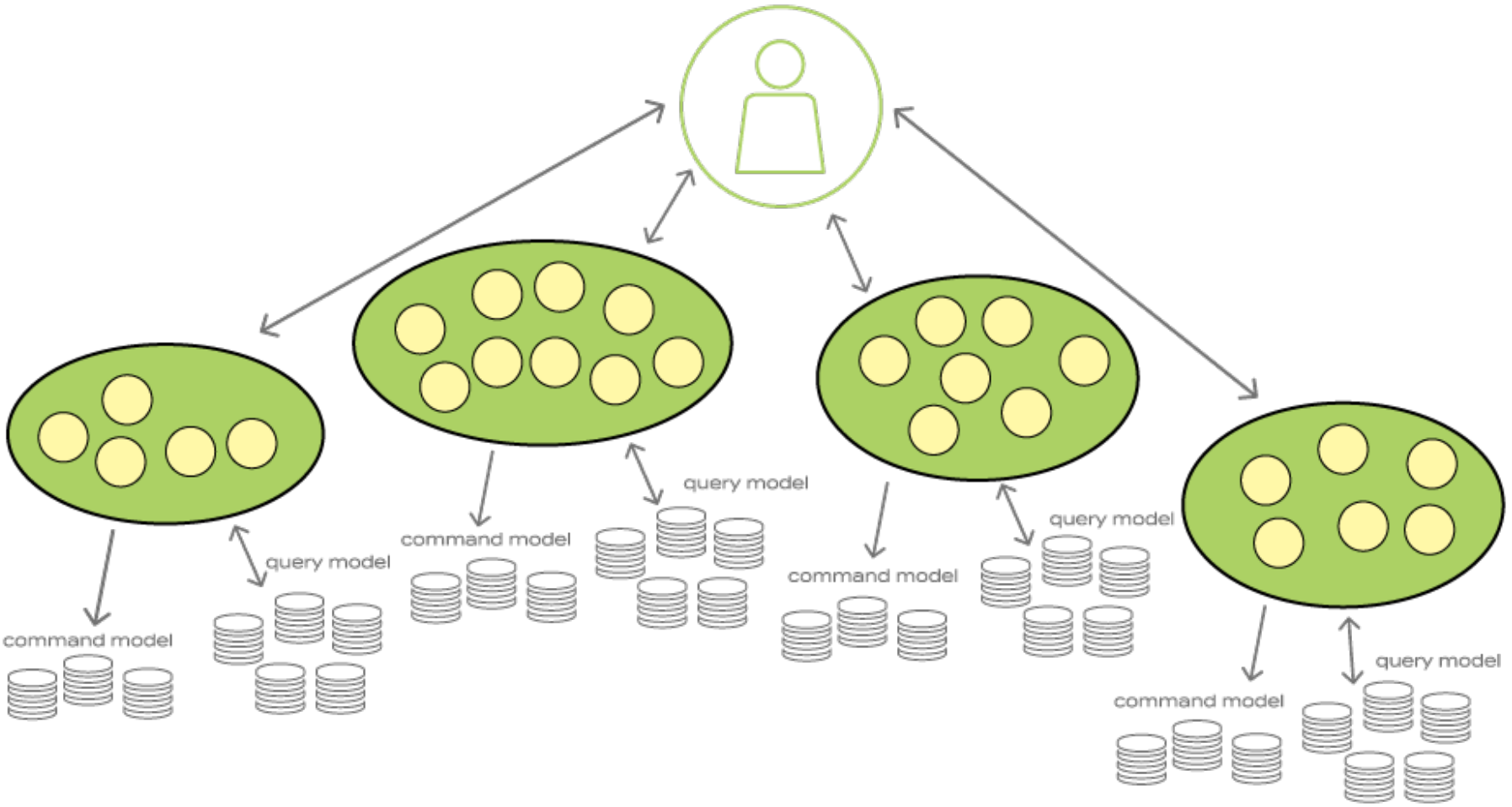

@VaughnVernon


Microsoft
OS/2
@VaughnVernon

# Reactive but...

Language Investment
Model Fluency
Type Safety
Testability

# Reactive DDD




@VaughnVernon

# Fluent?

flu·ent
/ˈflo͞oənt/

*adjective*

(of a person) able to express oneself easily and articulately.
"a fluent speaker and writer on technical subjects"
*synonyms:* articulate, eloquent, expressive, communicative, coherent, cogent, illuminating, vivid, well written/spoken
"a fluent campaign speech"

- (of a person) able to speak or write a particular foreign language easily and accurately.
  "she became **fluent in** French and German"
  *synonyms:* articulate; have a (good) command of
  "fluent in French"
- (of a foreign language) spoken accurately and with facility.
  "he spoke fluent Spanish"

@VaughnVernon

# Protocol

```
public interface Progress {
   ...
   void submitFor(Client client, ...);
   void denyPricing(Money suggestedPrice);
   void verifyPricing();
   ...
}
```

# Fluent

```
proposal.submitFor(client, expectations)
```

# Fluent

```
stage.actorOf(from(userId), User.class)
  .andThenInto(user -> user.with(new Contact(...)))
  .otherwiseConsume(noUser -> Response.of(NotFound))
  .andThenConsume(user -> Response.of(Ok, user));
```

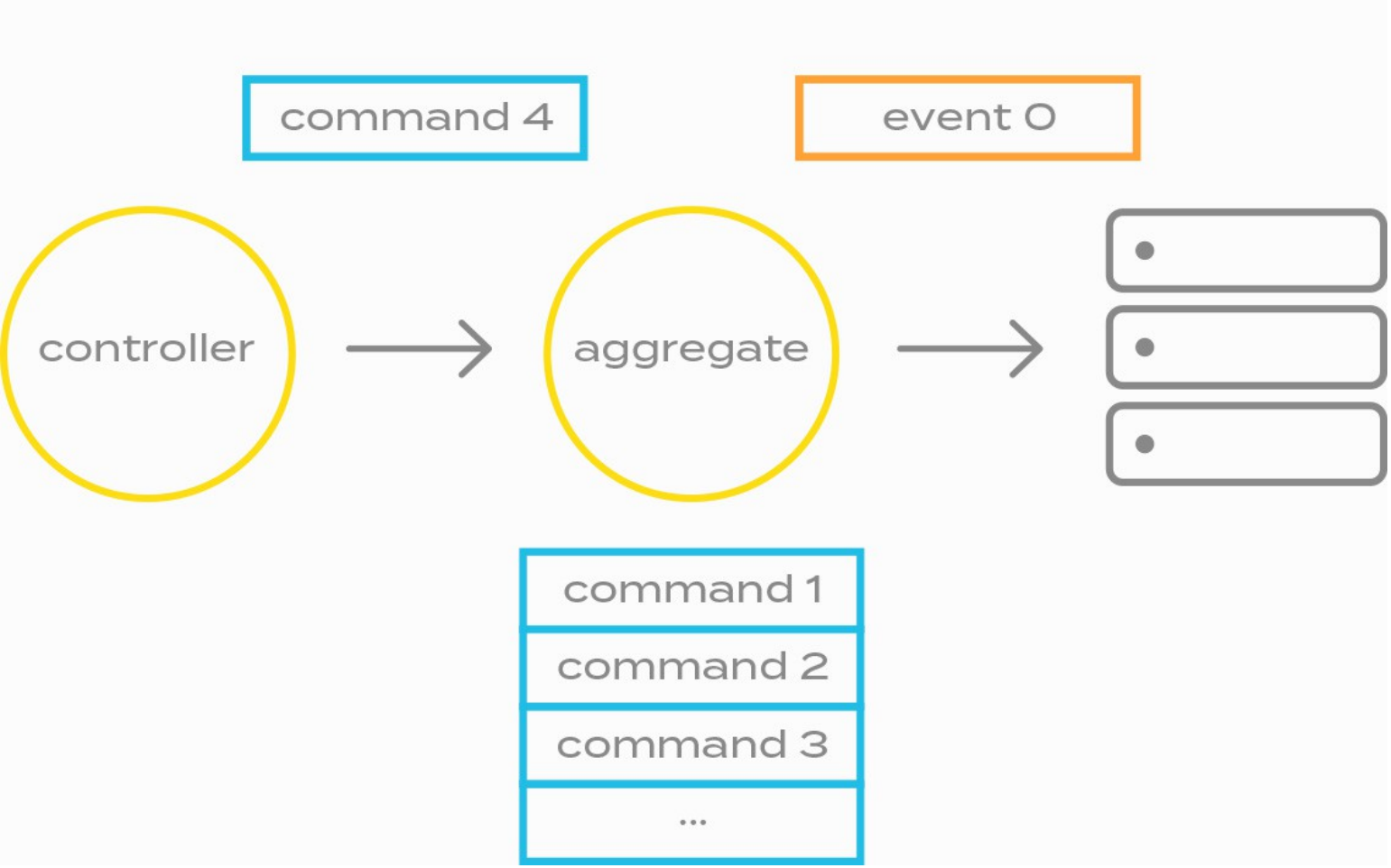
# Reactive?

```
proposal.submitFor(client, expectations)
```

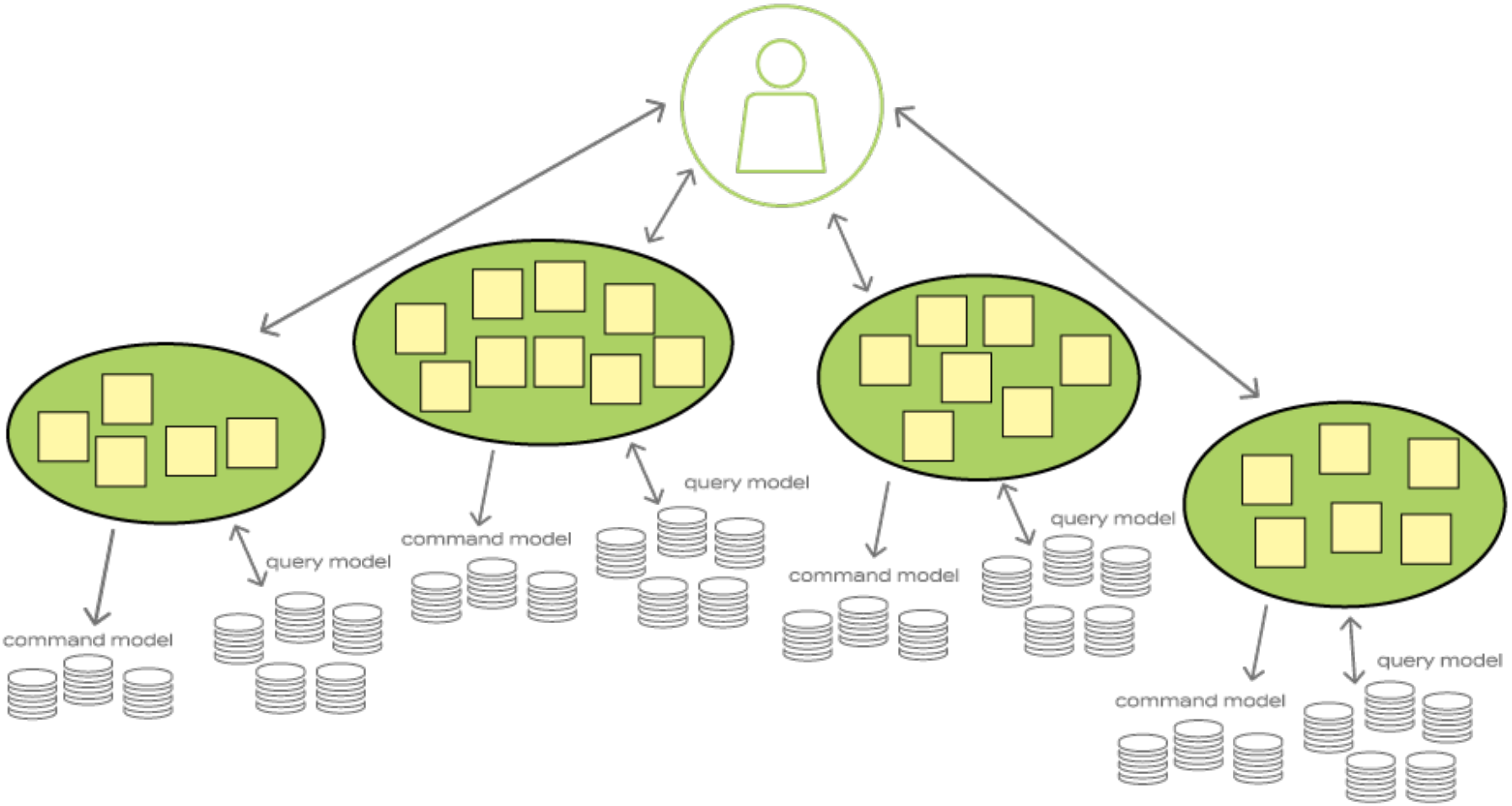
# Type Safe?

```
proposal.submitFor(
        Client.from(clientId),
        Expectations.of(
            Summary.has(summary),
            Description.has(description),
            Keywords.are(keywords),
            completedBy,
            Step.from(steps),
            price));
```

# Uncertainty (1)



# Uncertainty (2)




@VaughnVernon

# Model

```
public final class ProposalEntity
    extends EventSourced
    implements Proposal {

  public Client client;
  public Expectations expectations;
  ...
}
```

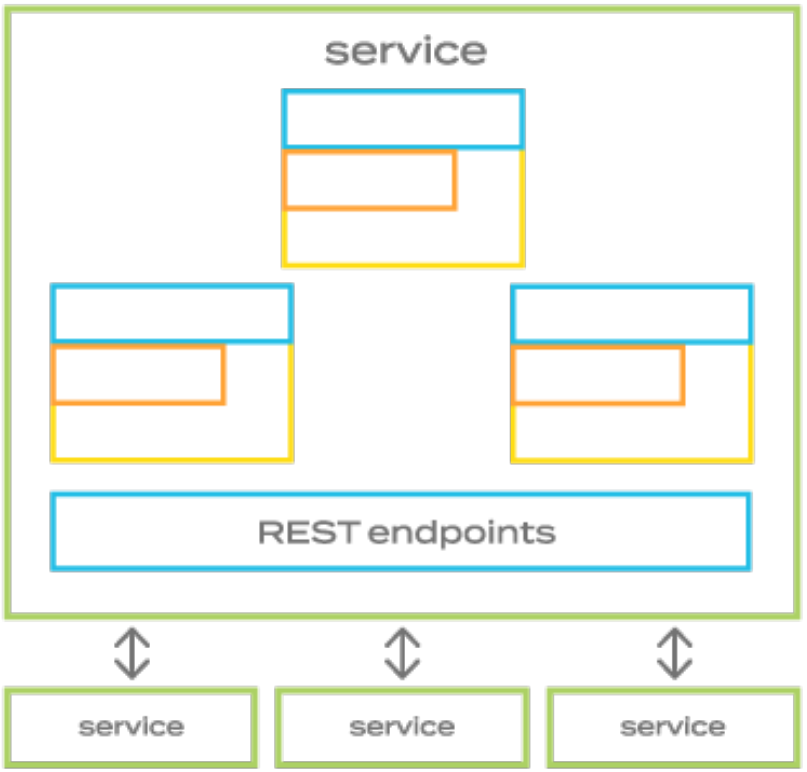
# Model

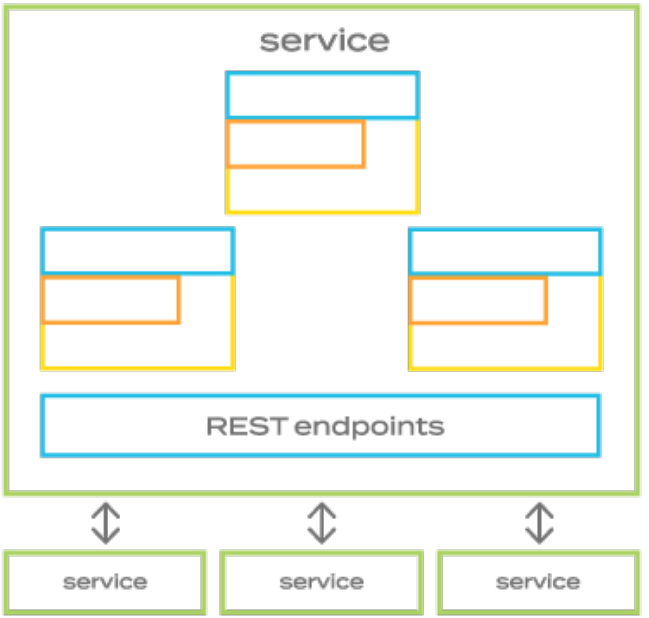
```
public final class ProposalEntity
    extends EventSourced
    implements Proposal {

  public Client client;
  public Expectations expectations;
  public Progress progress;
  ...
}
```

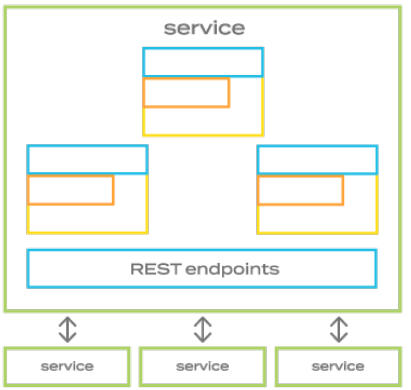
# Model It

```
public final class Progress {
  public static Progress None = ...
  public static Progress Submitted = ...

  private final Set<Spec> specs;

  protected Progress verifiedForPricing() {
    return newWith(Spec.PricingVerified);
  }
  ...
}
```

# *Everyone* Is Going Microservices

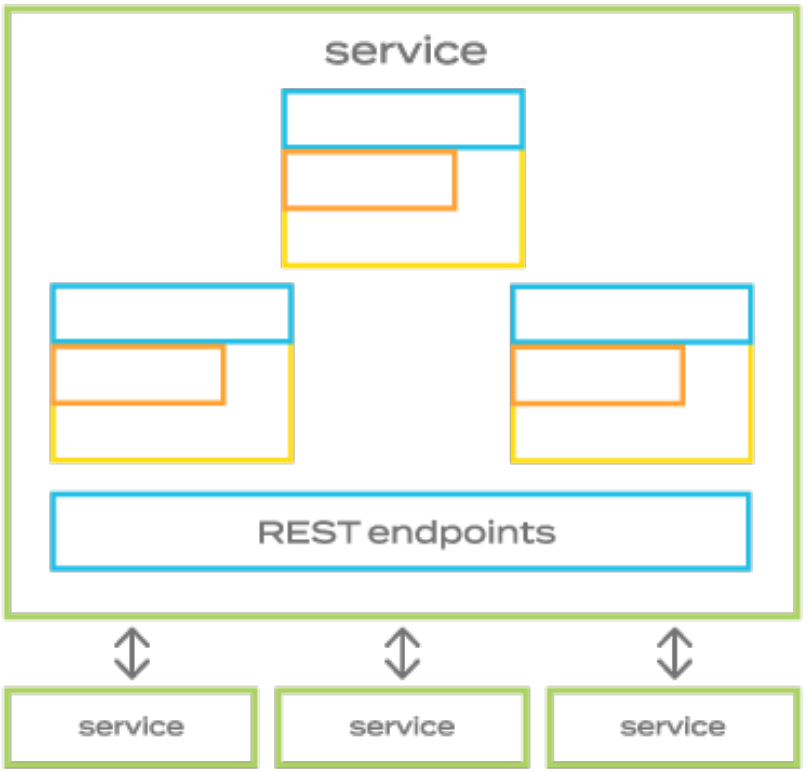


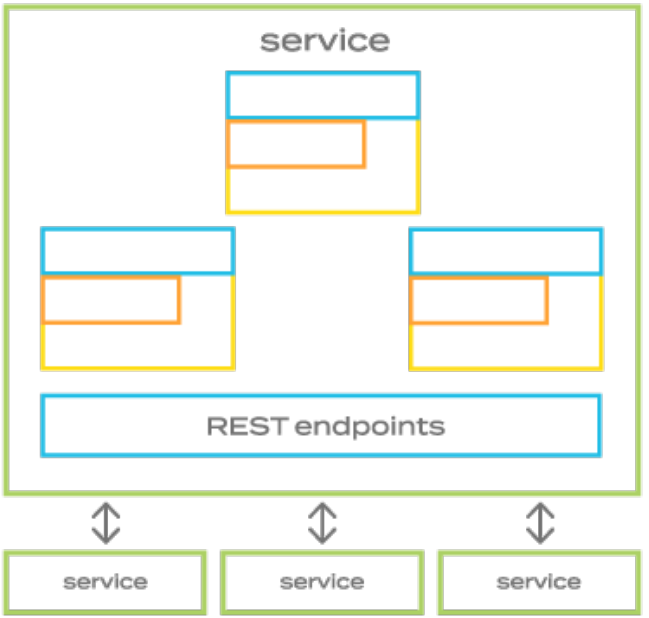



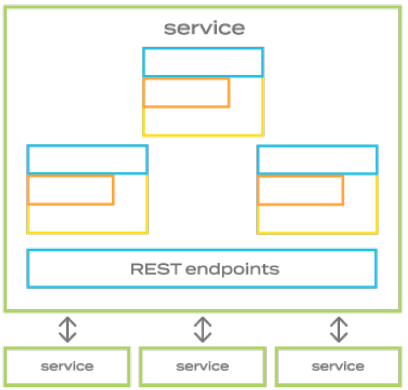




@VaughnVernon

# *So We Want to Go*
# Microservices




@VaughnVernon

# Business Wants

@VaughnVernon

# You Want

@VaughnVernon

# Definitions

**de•scrip•tion (dĭ-skrĭpˈshən)►**

**n. The act, process, or technique of describing.**

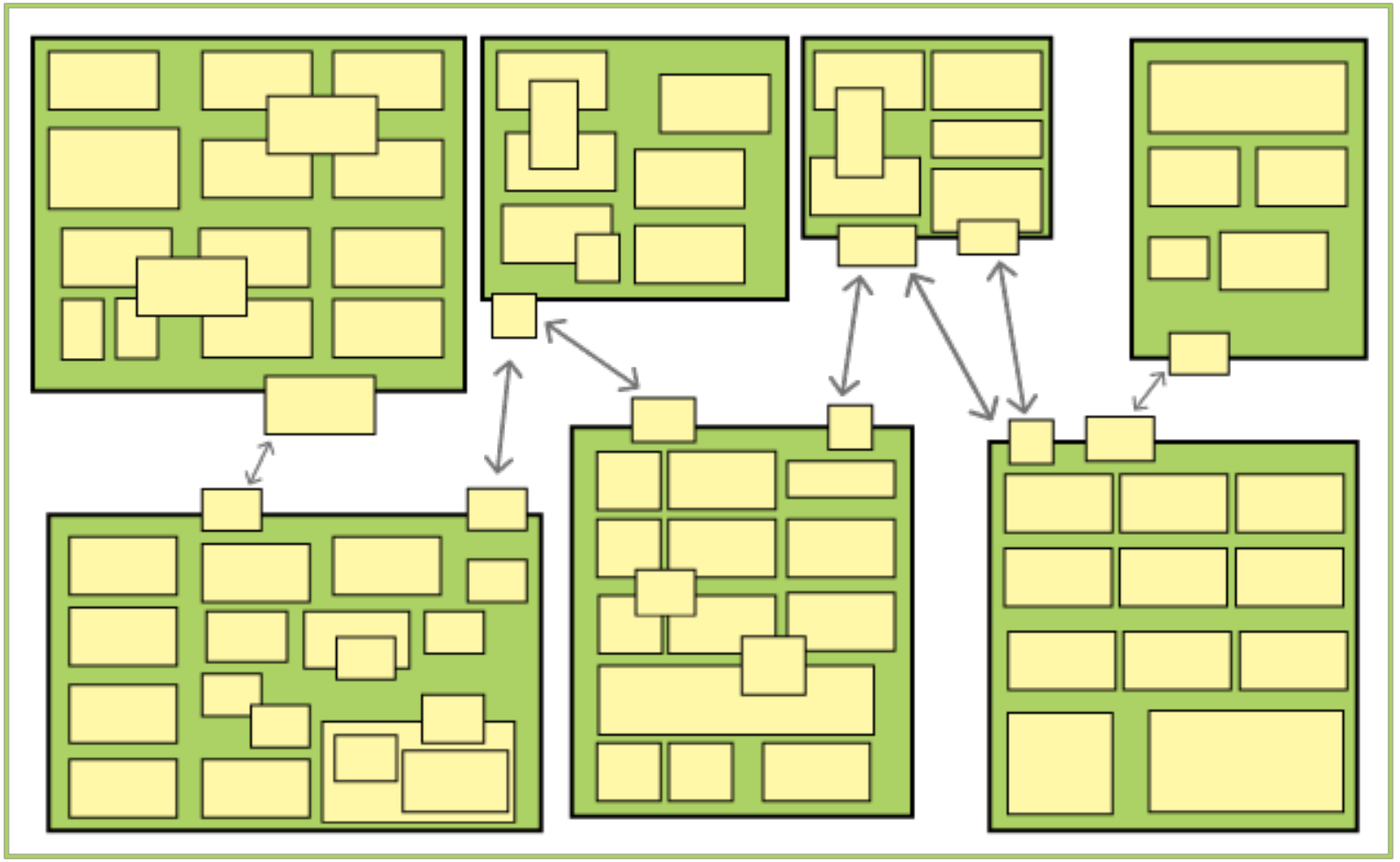
**n. A statement or an account describing something: published a description of the journey; gave a vivid description of the game.**

**n. A pictorial representation: Monet's ethereal descriptions of haystacks and water lilies.**

# Legacy

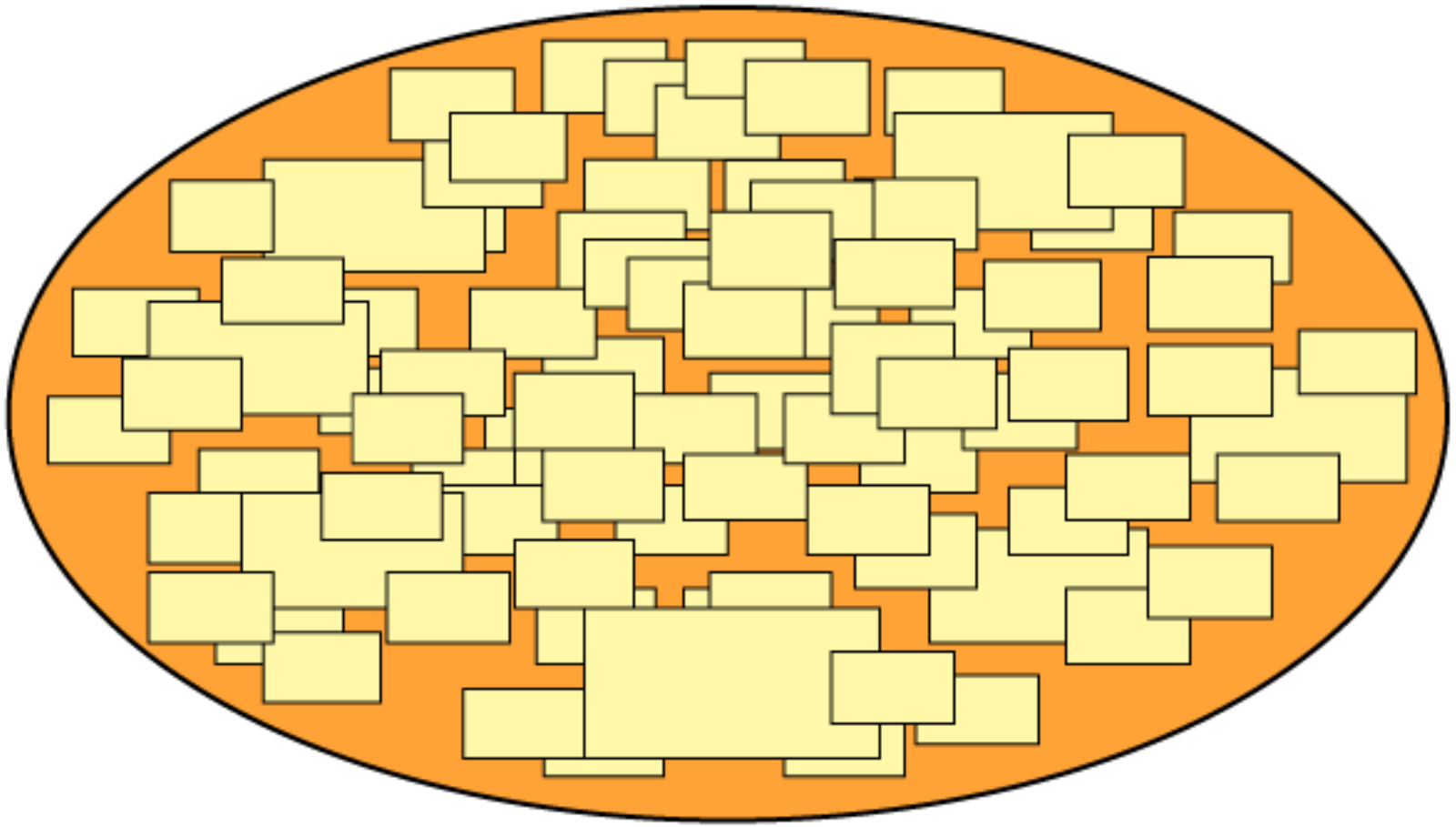
@VaughnVernon

# Legacy: Monolith



@VaughnVernon

# Legacy: BBoM



@VaughnVernon

# Microservice

?

# 100

# LoC

@VaughnVernon

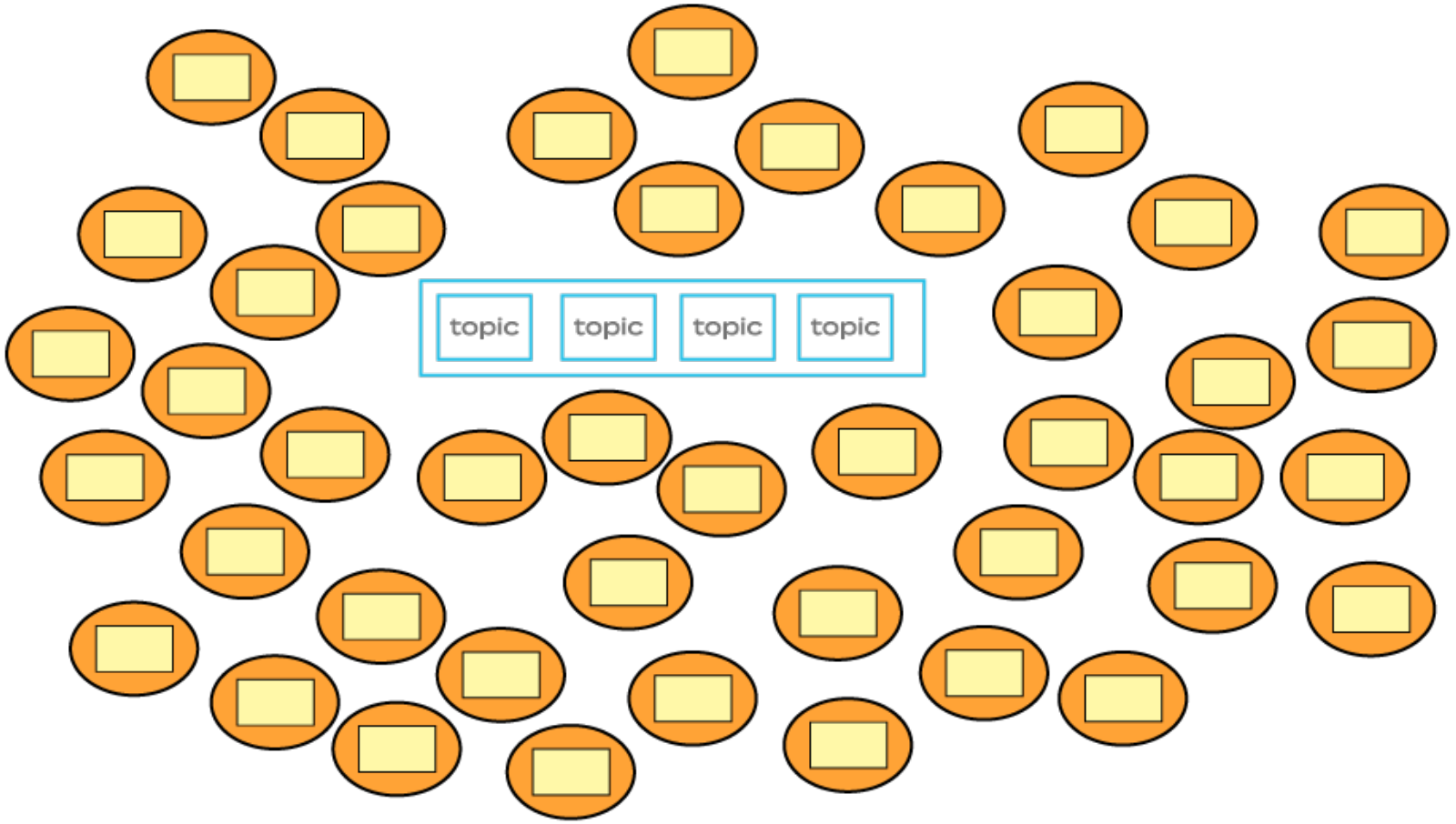
# 400

# LoC

@VaughnVernon

# 1000?

# 100 LoC Now




@VaughnVernon

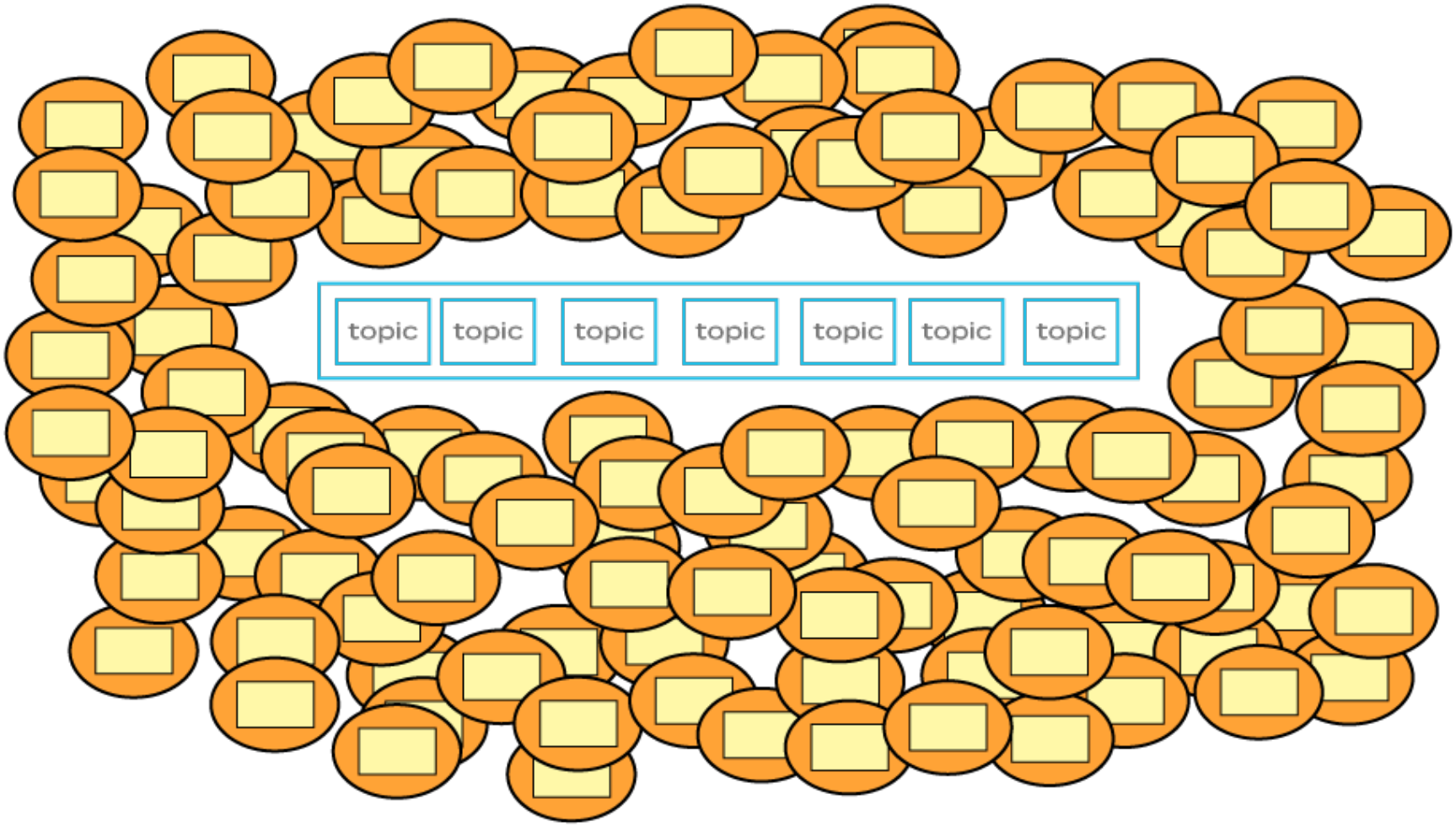
# Dependencies?

@VaughnVernon

# ONLY $400/mo!

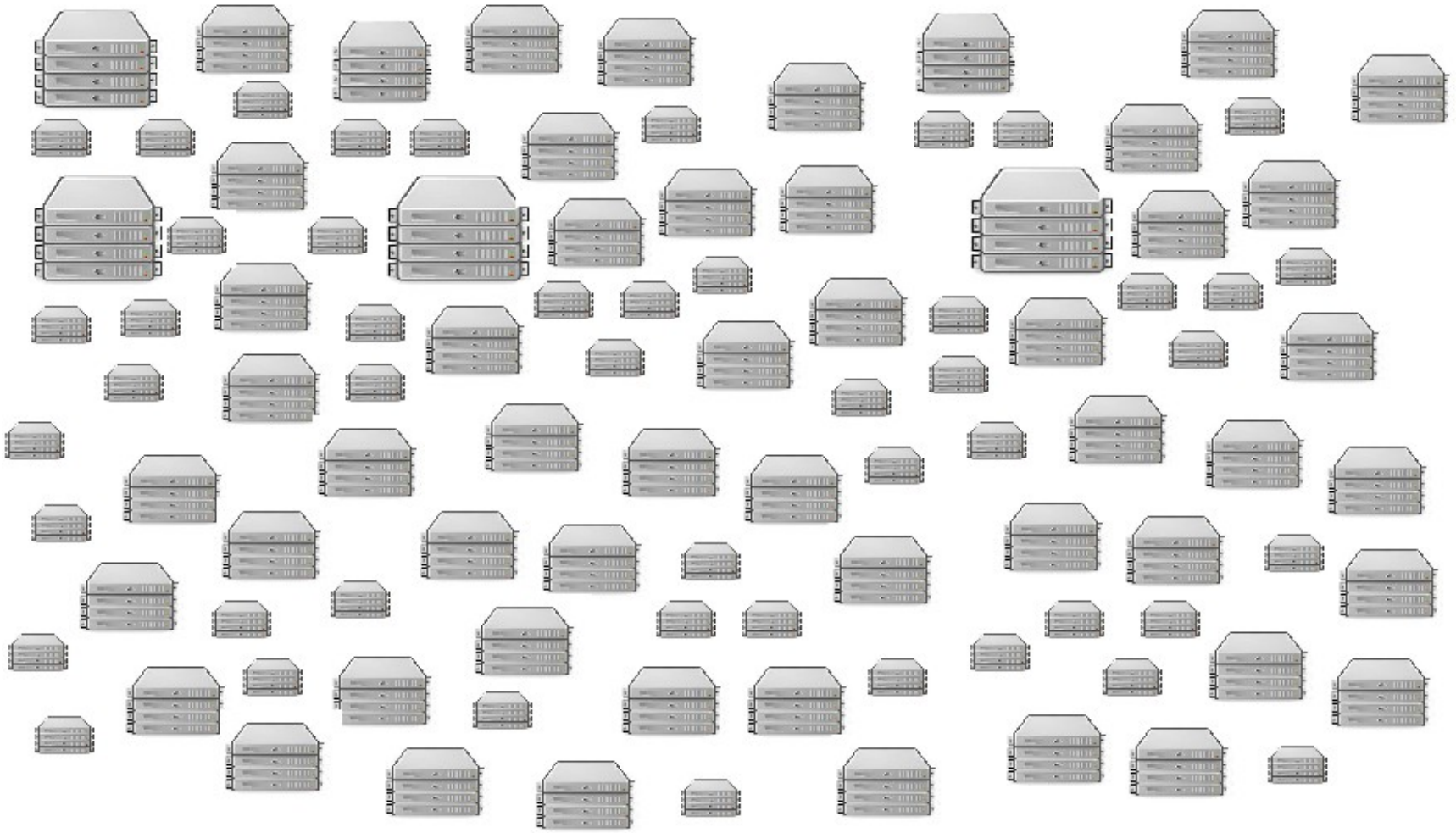
@VaughnVernon

# 100 Then * $400/mo




@VaughnVernon

# And...



@VaughnVernon

# Complex System

## 2,000,000 LoC?

## 5,000,000 LoC?

# 2,000,000 LoC

## 2MM / 100 = 20,000 MS

## 20K * $400/mo = $8MM/mo

## $8MM * 12 = $96MM/yr

@VaughnVernon

# 5,000,000 LoC

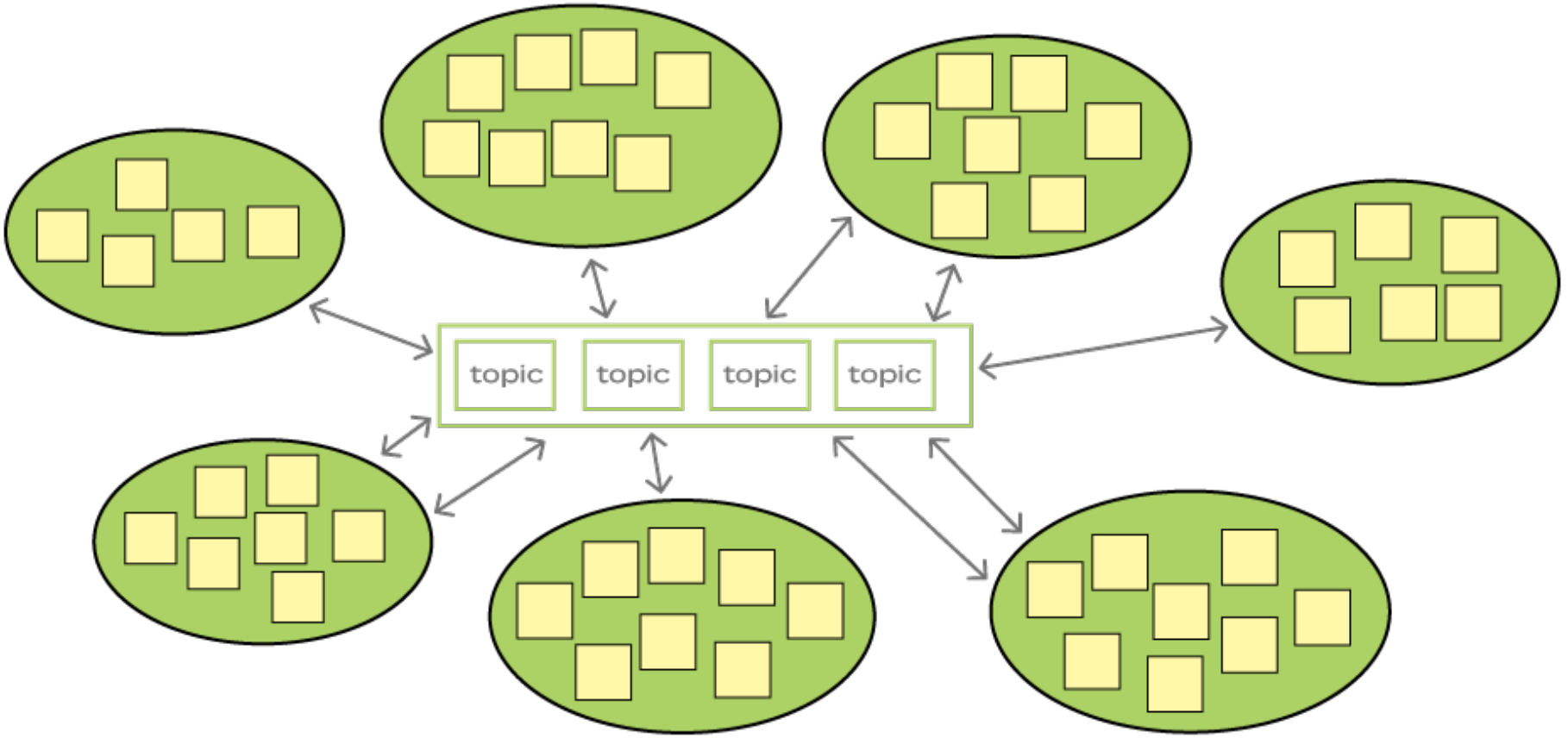
## 5MM / 100 = 50,000 MS

## 50K * $400/mo = $20MM/mo

## $20MM * 12 = $240MM/yr

@VaughnVernon

# Bounded Context




@VaughnVernon

# Identify Strategic Drivers

"Our problem is that our software implements ordinary business process but stops short of seeking strategic advantage."

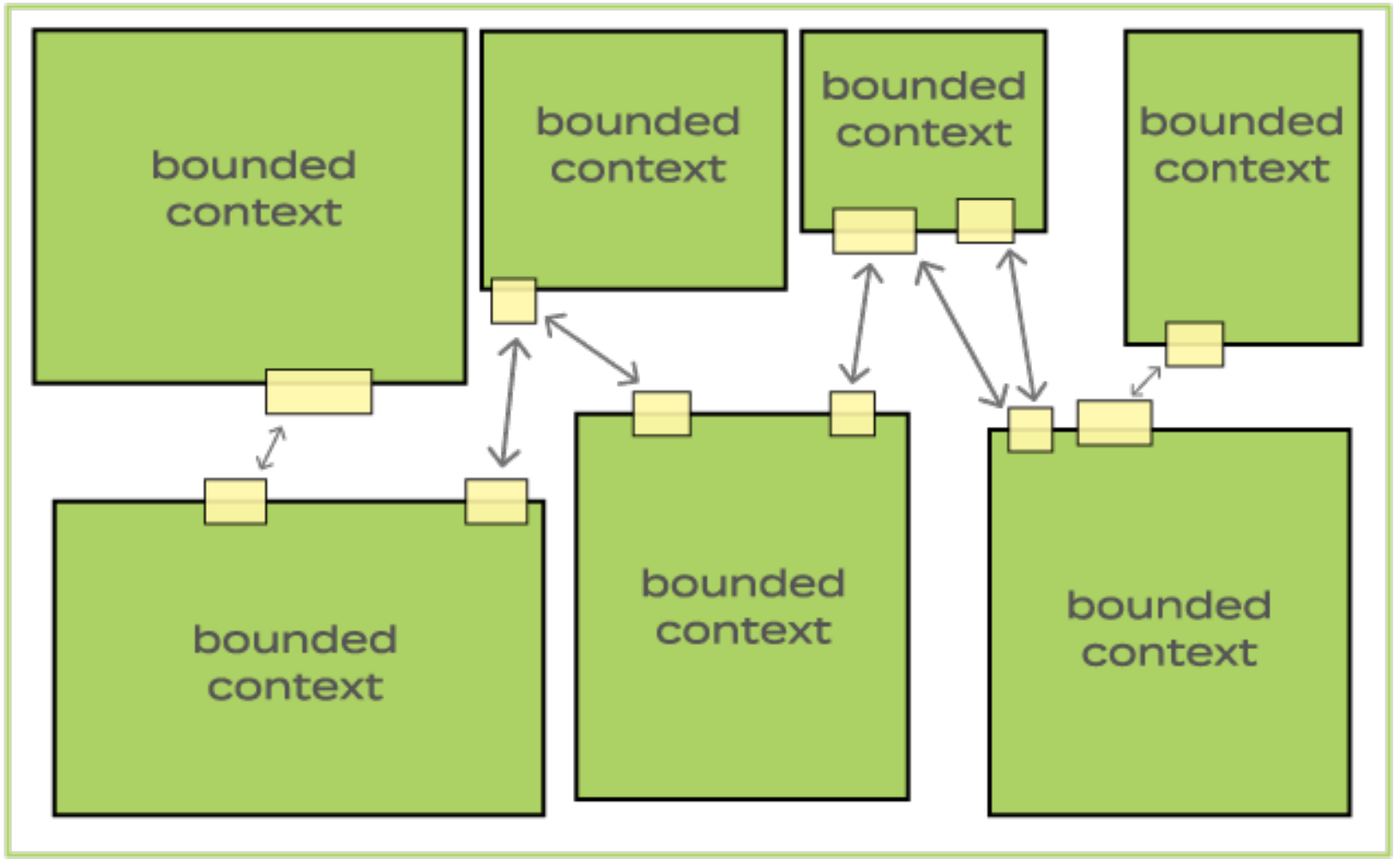
# Tell!

```
public class Client extends Entity {
   public void name(String name) {
      ...
   }
   public void address(PostalAddress address) {
      ...
   }
   public void relocateTo(PostalAddress address) {
      ...
   }
   public void telephone(Telephone telephone) {
      ...
   }
   ...
}
```

# Explicit, Testable, Less Code

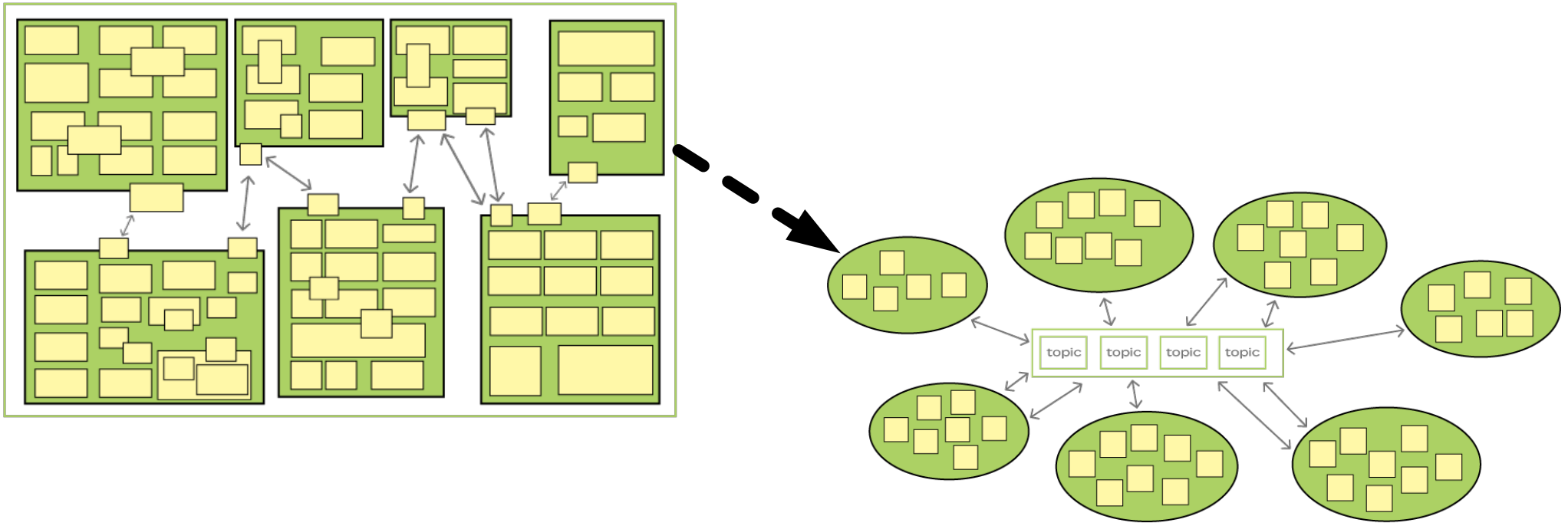
```
public class Client {
  public void relocateTo(PostalAddress address) {
    this.postalAddress = address;
    emit(new ClientRelocated(...));
  }
}

public class ClientTests {
  public void testThatClientRelocates() {
    ...
  }
}
```

# Bounded Context



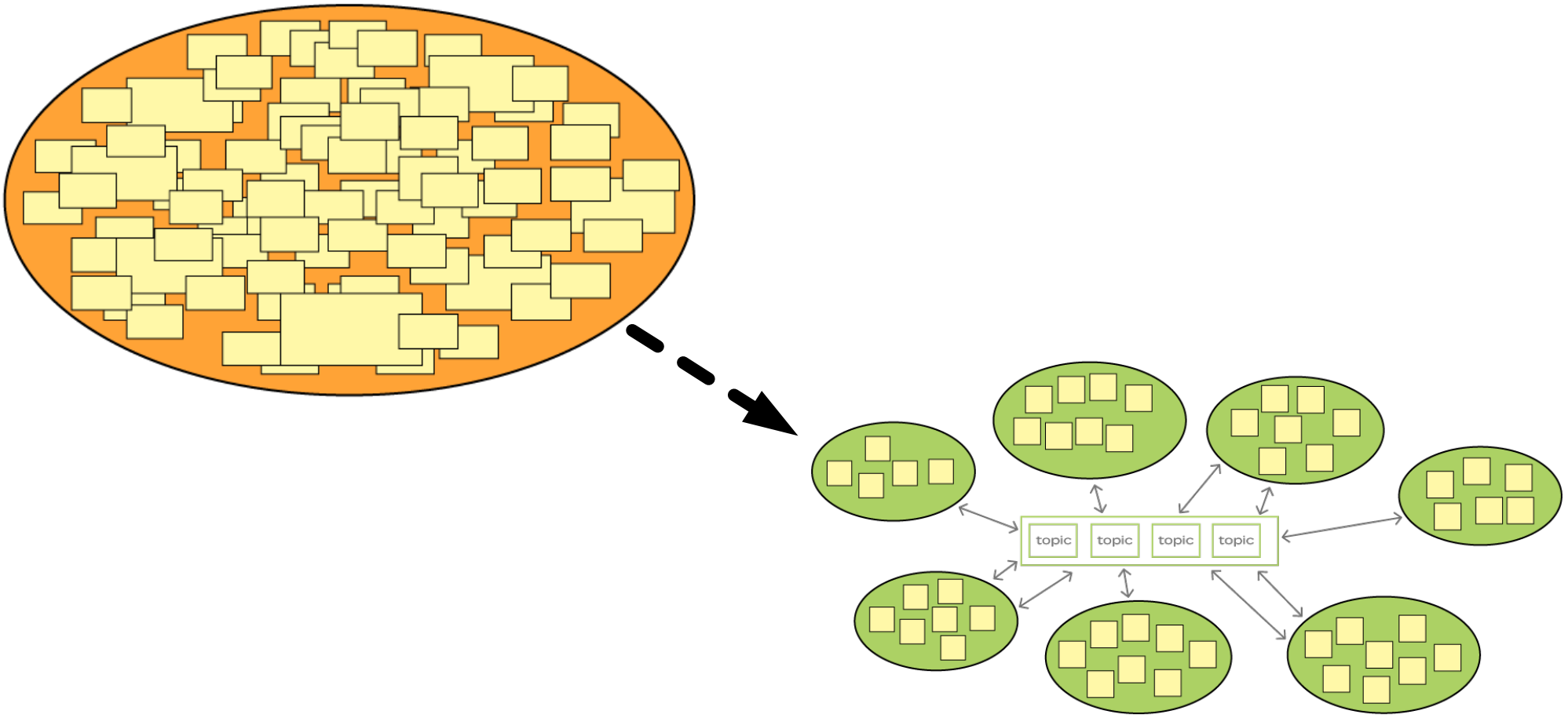

@VaughnVernon

# Monolith to Microservices




@VaughnVernon

# BBoM To BC—How?



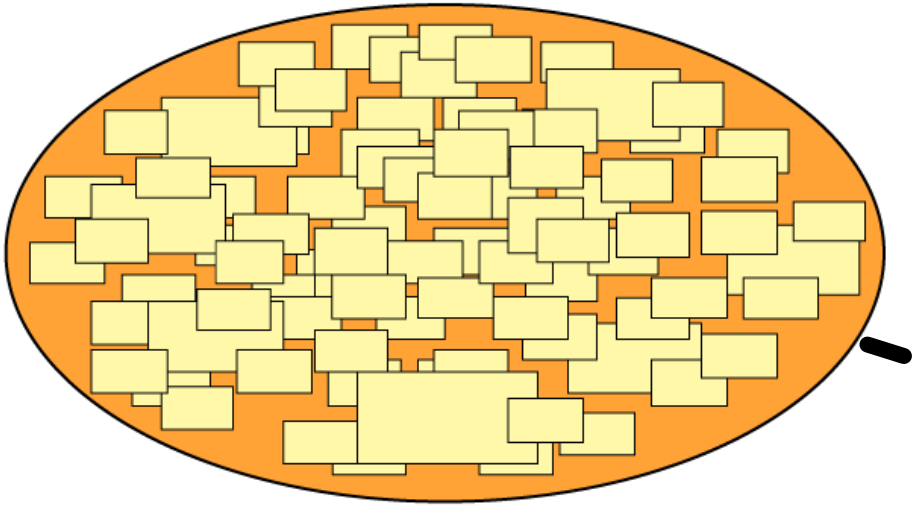

@VaughnVernon

# Rewrite

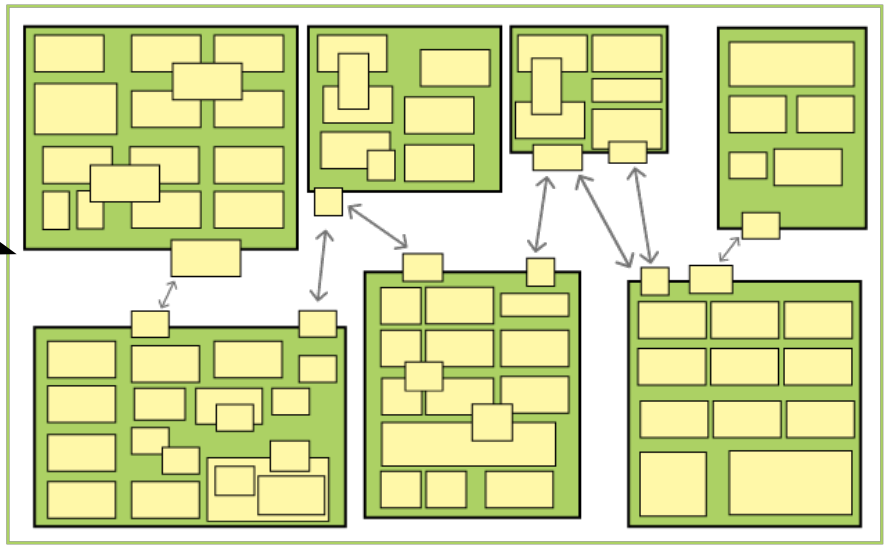
```
000024
000025      PROCEDURE DIVISION.
000026      0001-MAIN.
000027          INSPECT FUNCTION REVERSE(STR-1)
000028                  TALLYING WS-LEN1 FOR LEADING SPACES.
000029          COMPUTE  WS-LEN = LENGTH OF STR-1 - WS-LEN1.
000030          DISPLAY WS-LEN.
000031          MOVE 1 TO I.
000032          MOVE WS-LEN TO J.
000033          PERFORM REV-PARA WS-LEN TIMES.
000034          DISPLAY STR-1.
000035          DISPLAY STR-2.
000036          GOBACK.
000037       REV-PARA.
000038          MOVE STR-1(J:1) TO STR-2(I:1).
000039          SUBTRACT 1 FROM J.
000040          ADD 1 TO I.
000041          EXIT.
****** ********************************** Bottom of Data ********************************
```

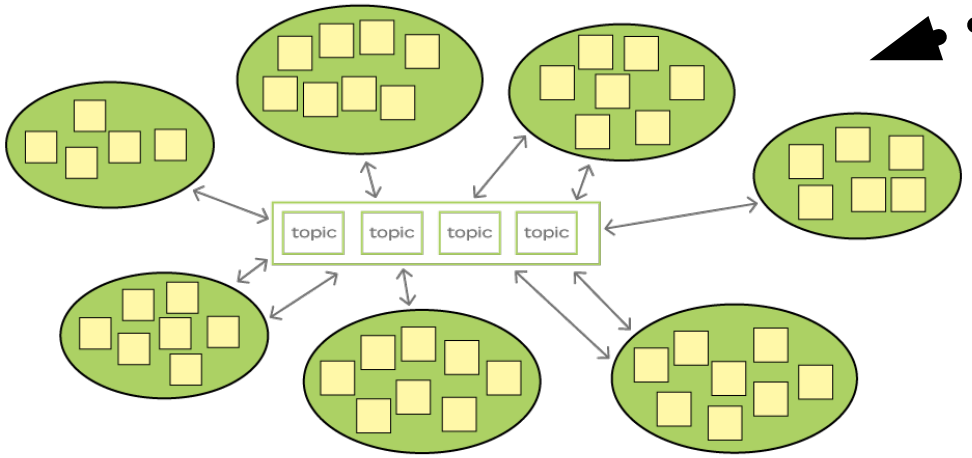
@VaughnVernon

# Eat an Elephant



One bite at a time.

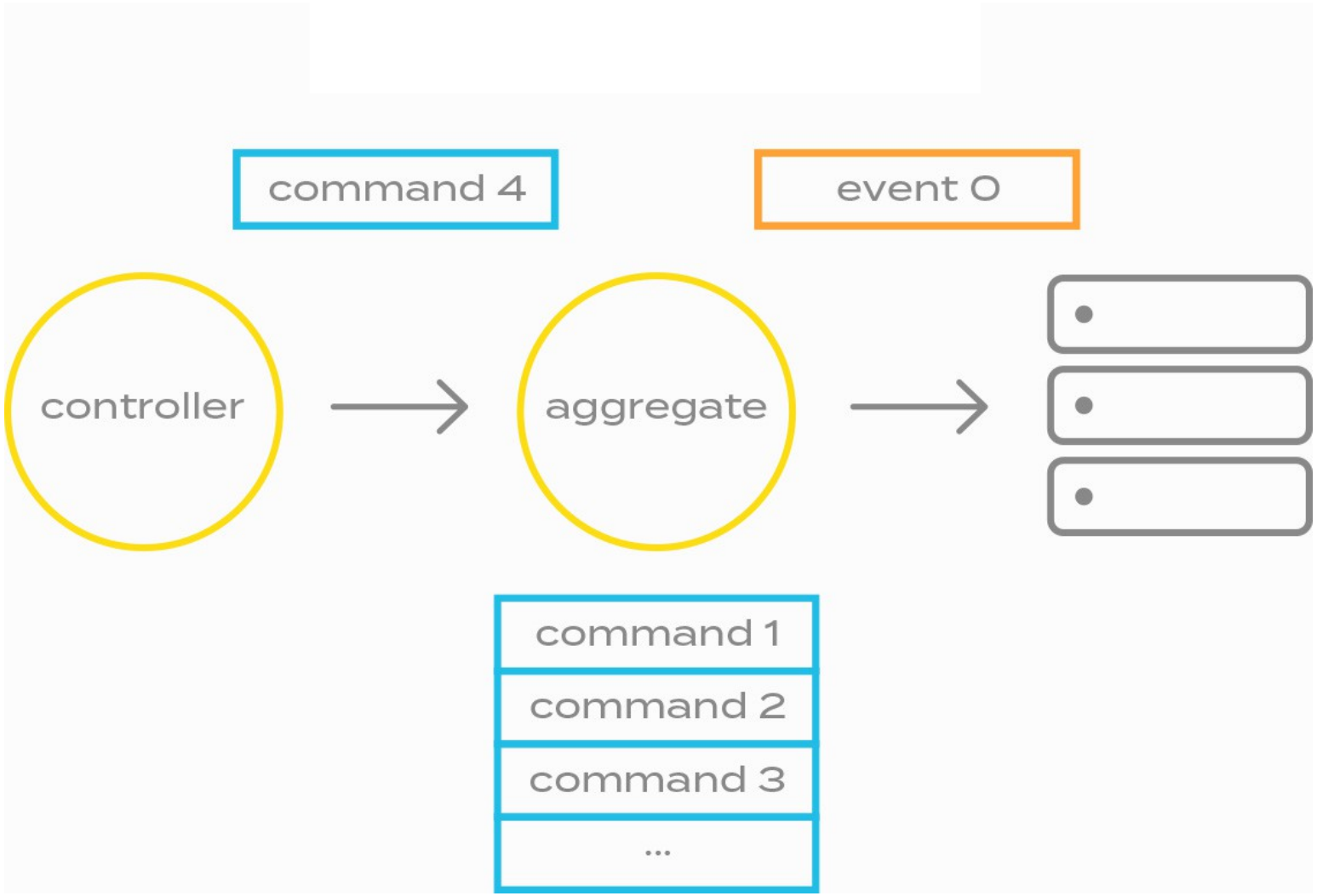





- Change-driven
- Value-driven
- Test-driven

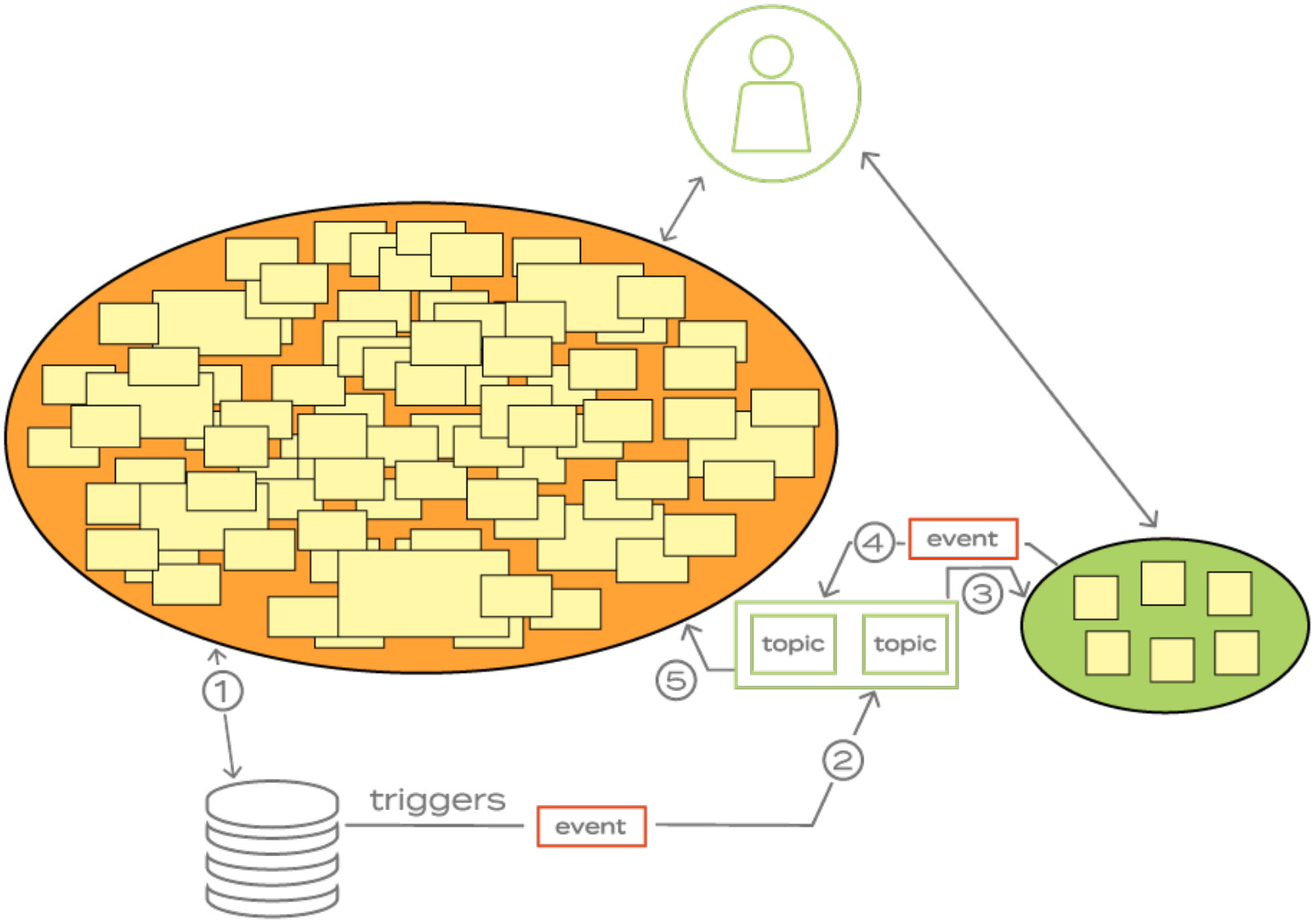
@VaughnVernon

# Event Driven




@VaughnVernon

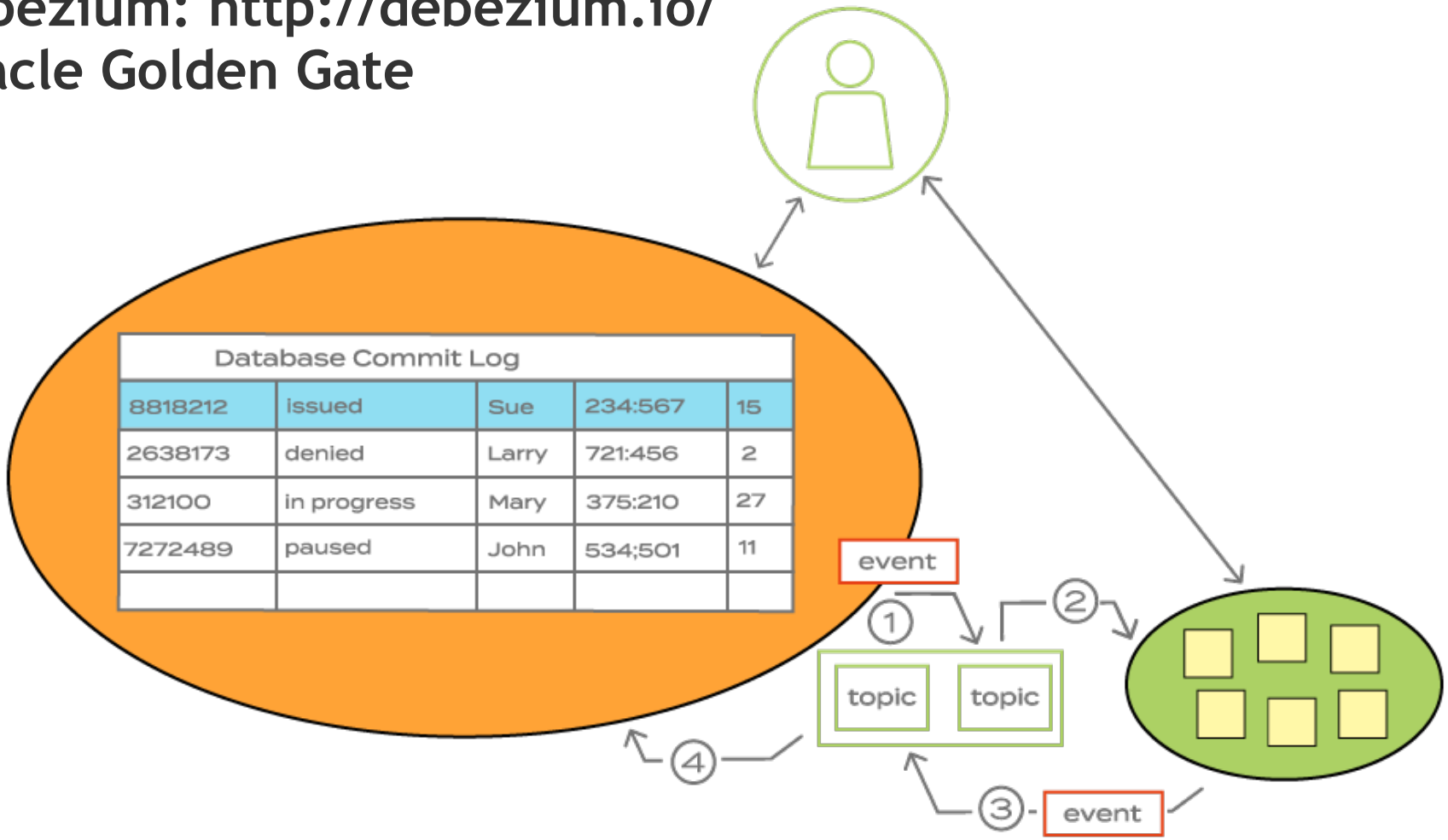
# Strangler (1)



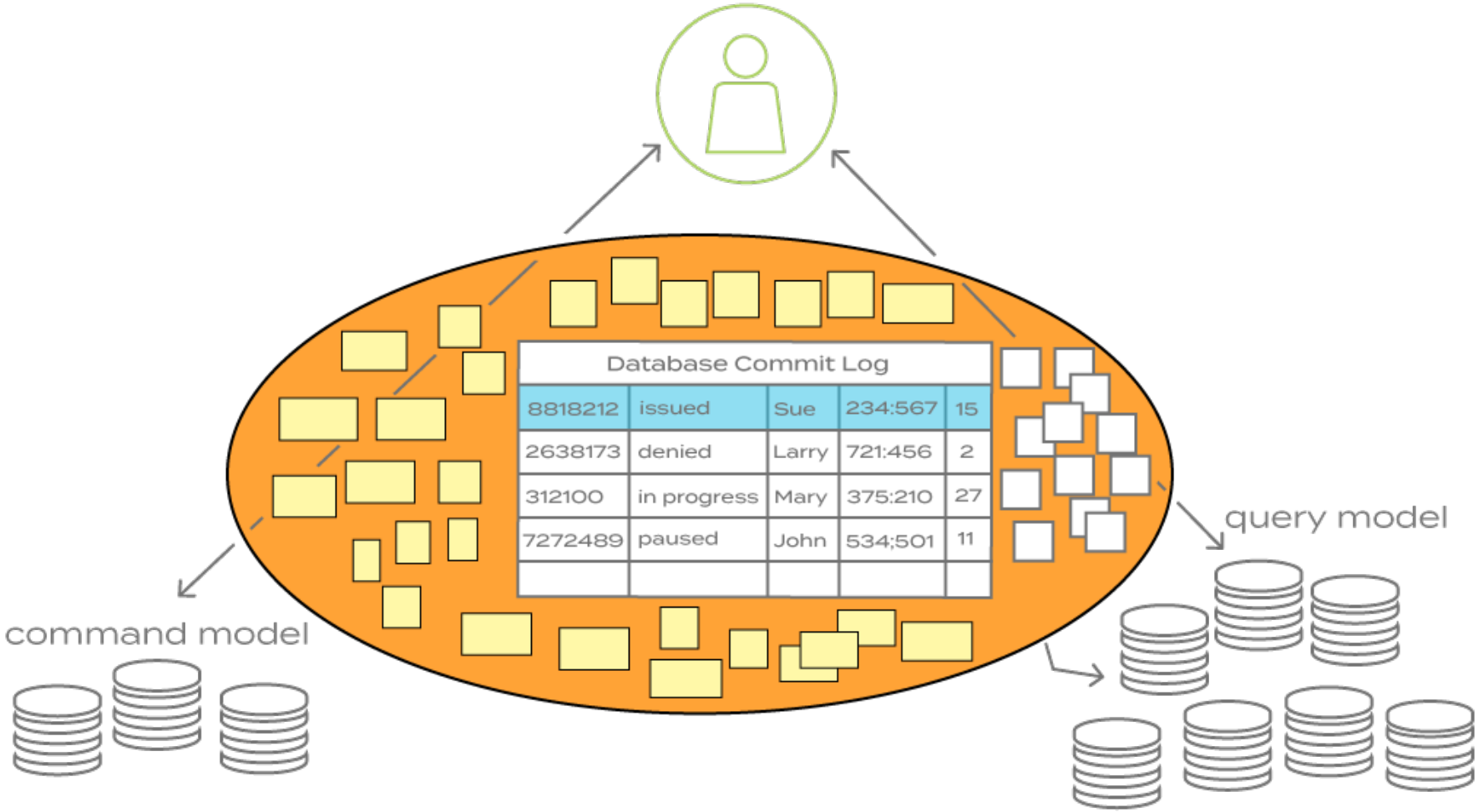

@VaughnVernon

# Strangler (2)

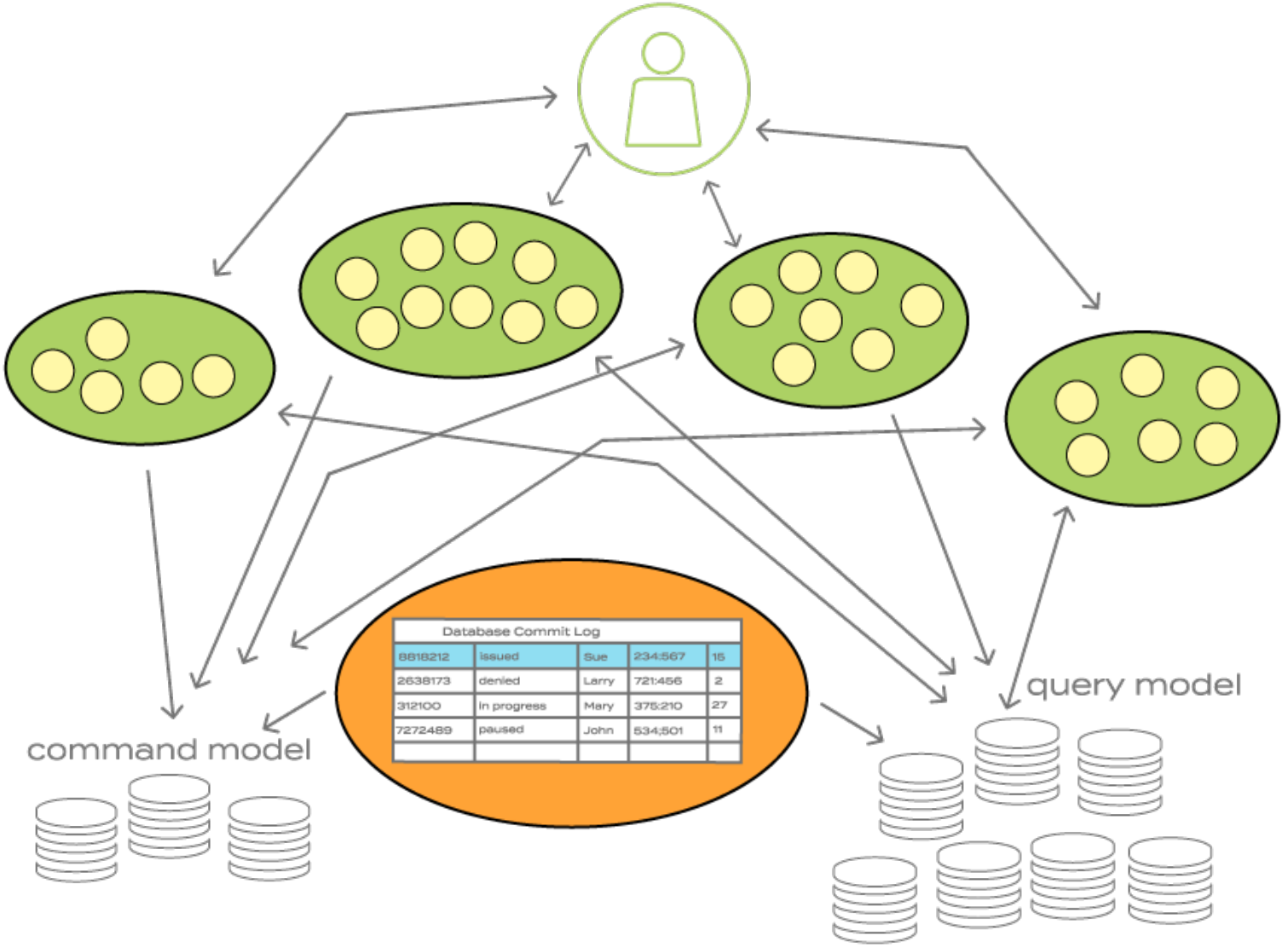
- Debezium: http://debezium.io/
- Oracle Golden Gate



# Restructured (1)

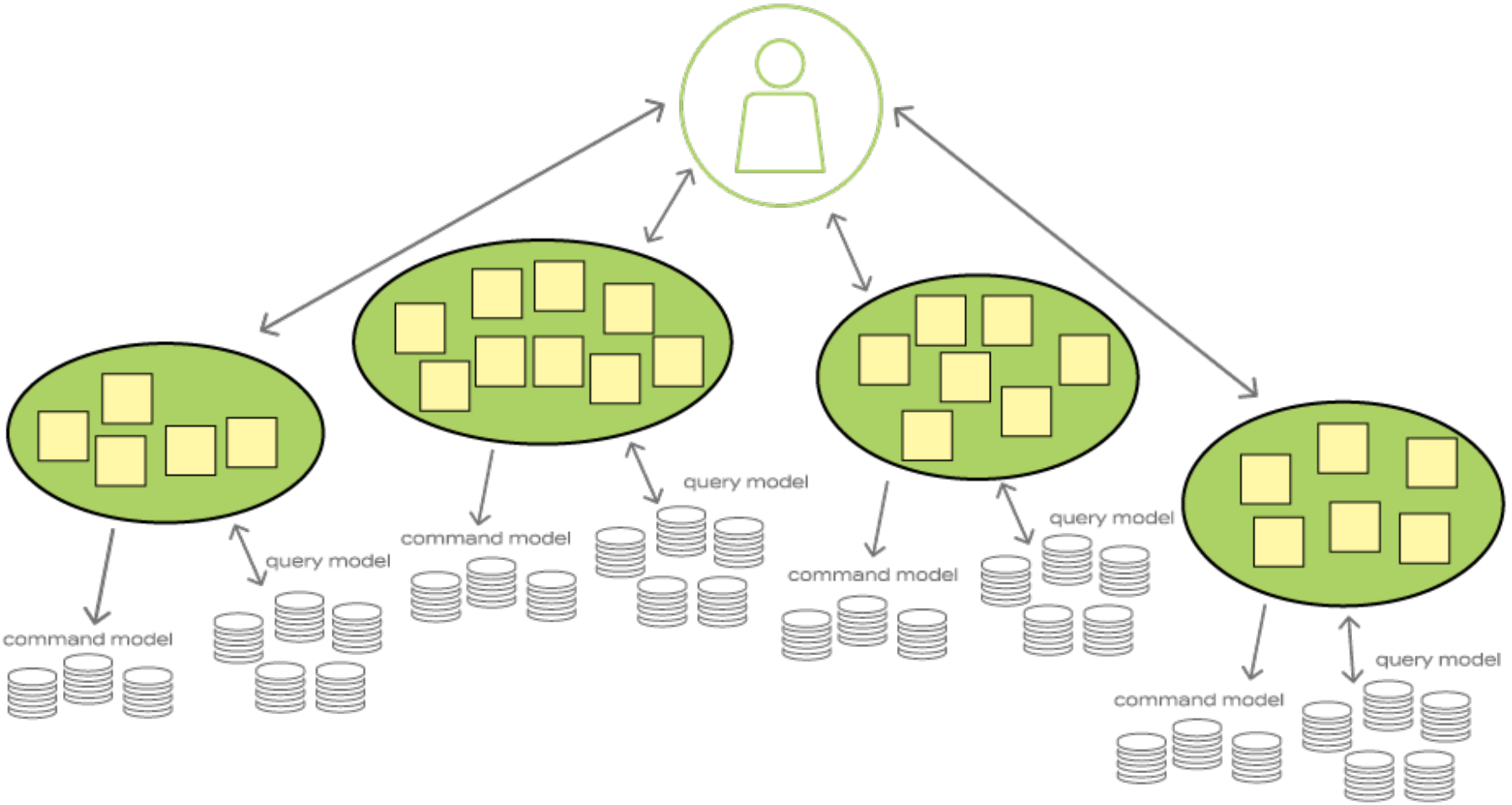


# Restructured (2)




@VaughnVernon

# Restructured (3)




@VaughnVernon

**Dave Farley**
@davefarley77

Following

True, but I can't think of many other fields of human endeavour where a 50x loss of efficiency is acceptable for "ease of use".

Our attitudes to efficiency in software are weird!

> **Ken Mijime** @KeNaCo666
> Replying to @mjpt777
>
> I presume this is a common knowledge.
> If you choose python, it's because of development ease and agility. Not for performance!

12:20 PM - 18 Sep 2018

**5** Retweets **26** Likes

6 5 26

@VaughnVernon

**vlingo**
@vlingo_io

"People who are more than casually interested in computers should have at least some idea of what the underlying hardware is like. Otherwise the programs they write will be pretty weird." -- Donald Knuth

> **Dave Farley** @davefarley77
> True, but I can't think of many other fields of human endeavour where a 50x loss of efficiency is acceptable for "ease of use".
>
> Our attitudes to efficiency in software are weird! twitter.com/KeNaCo666/stat…

4:15 PM - 21 Sep 2018

**10** Retweets **23** Likes

10 23

@VaughnVernon

**Dave Farley**
@davefarley77

Following

Replying to @tom_a_r_johnson

Except that, if the runtime of your program is 1 second and you can complete it in 1ms and don't, you are producing 1000x as much CO2 as you need to. Tiny for your application, but significant on a global scale.

(I know this is an over-simplification, but valid in principle)

1:53 AM - 19 Sep 2018

1

@VaughnVernon

**Dave Farley**
@davefarley77

Following

Replying to @tom_a_r_johnson

Except that, if the runtime of your program is 1 second and you can complete it in 1ms and don't, you are producing 1000x as much CO2 as you need to. Tiny for your application, but significant on a global scale.

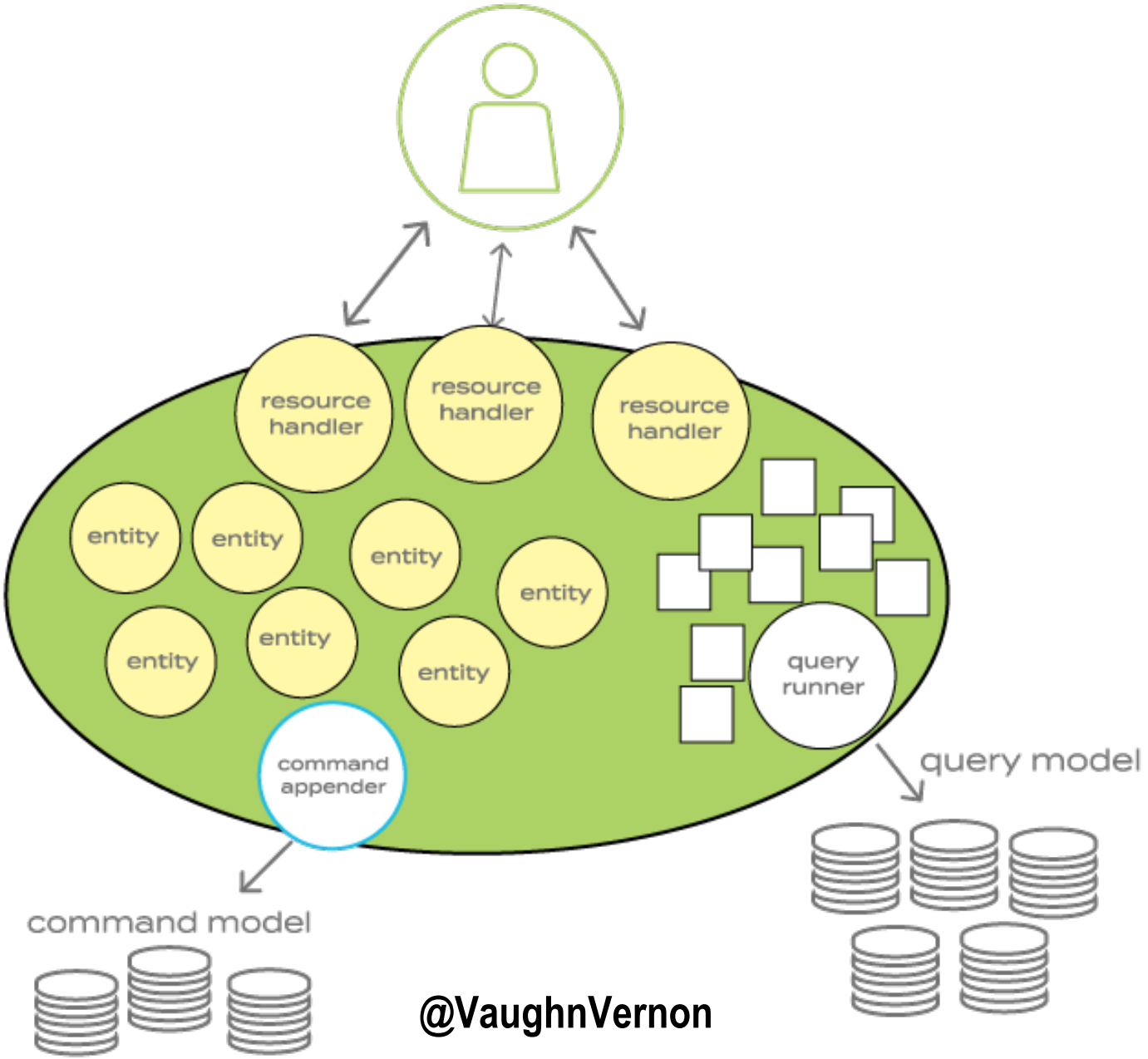
(I know this is an over-simplification, but valid in principle)

1:53 AM - 19 Sep 2018

1

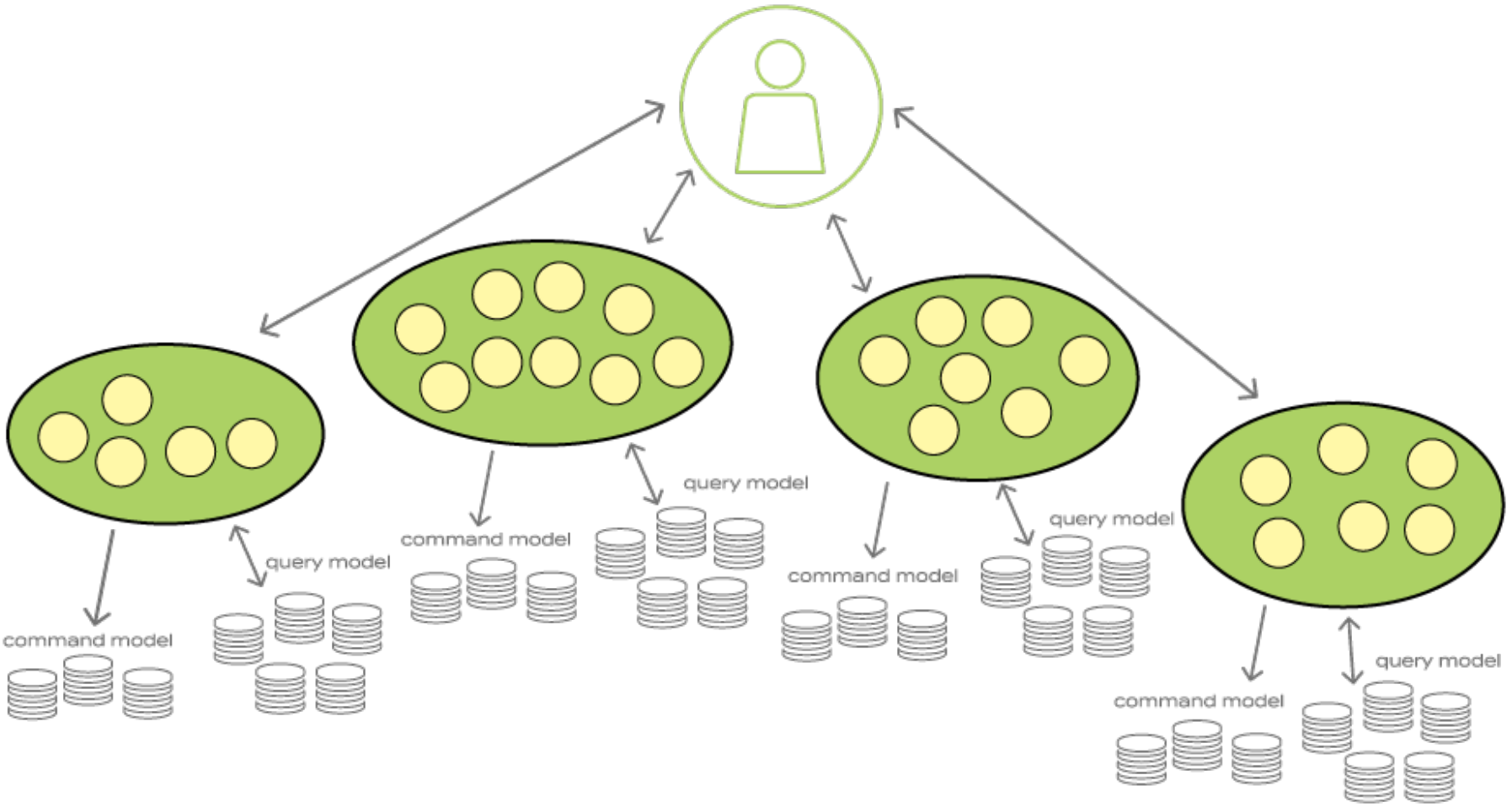
@VaughnVernon

# Reactive Rework



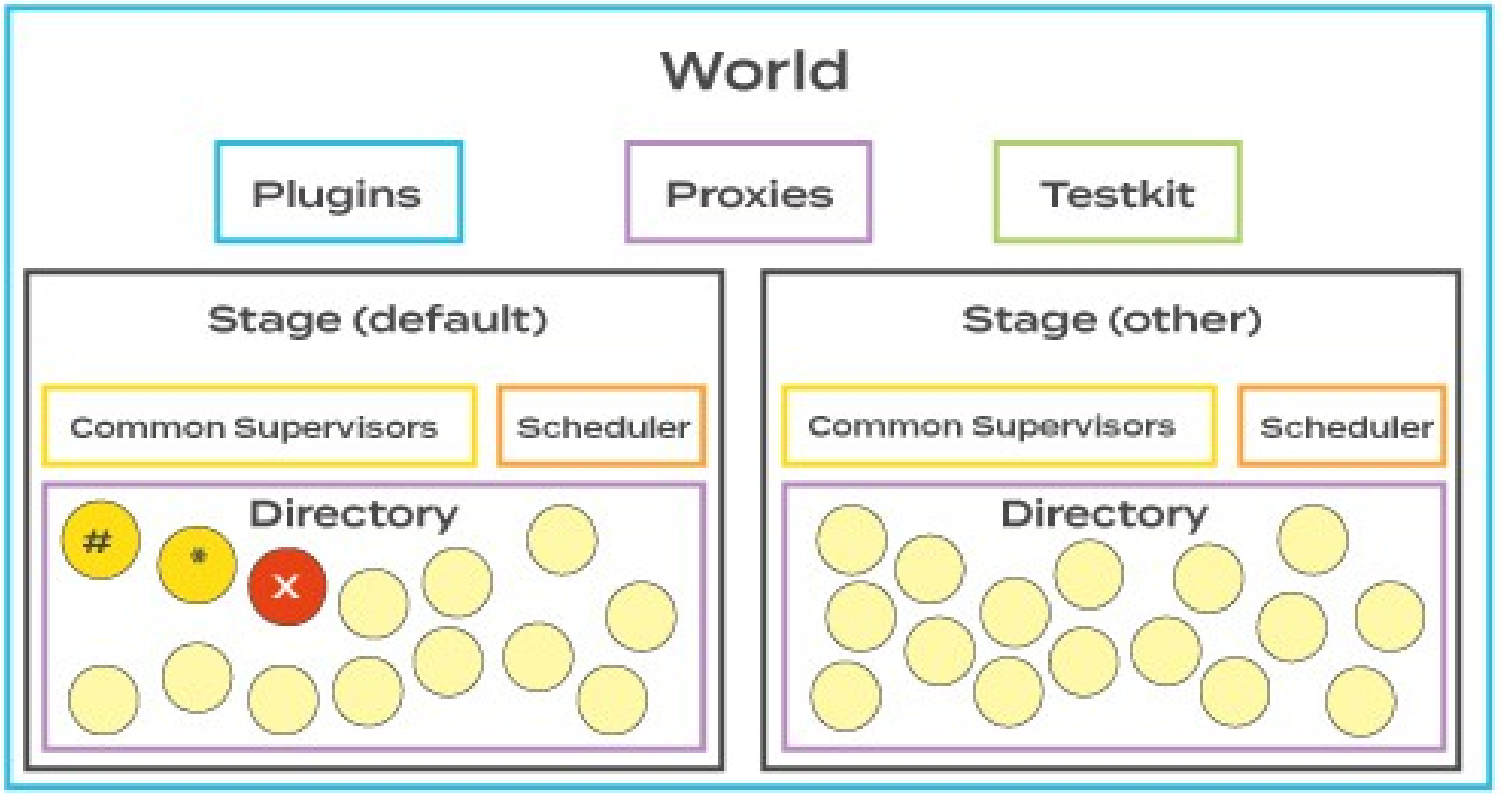

@VaughnVernon

# Reactive DDD

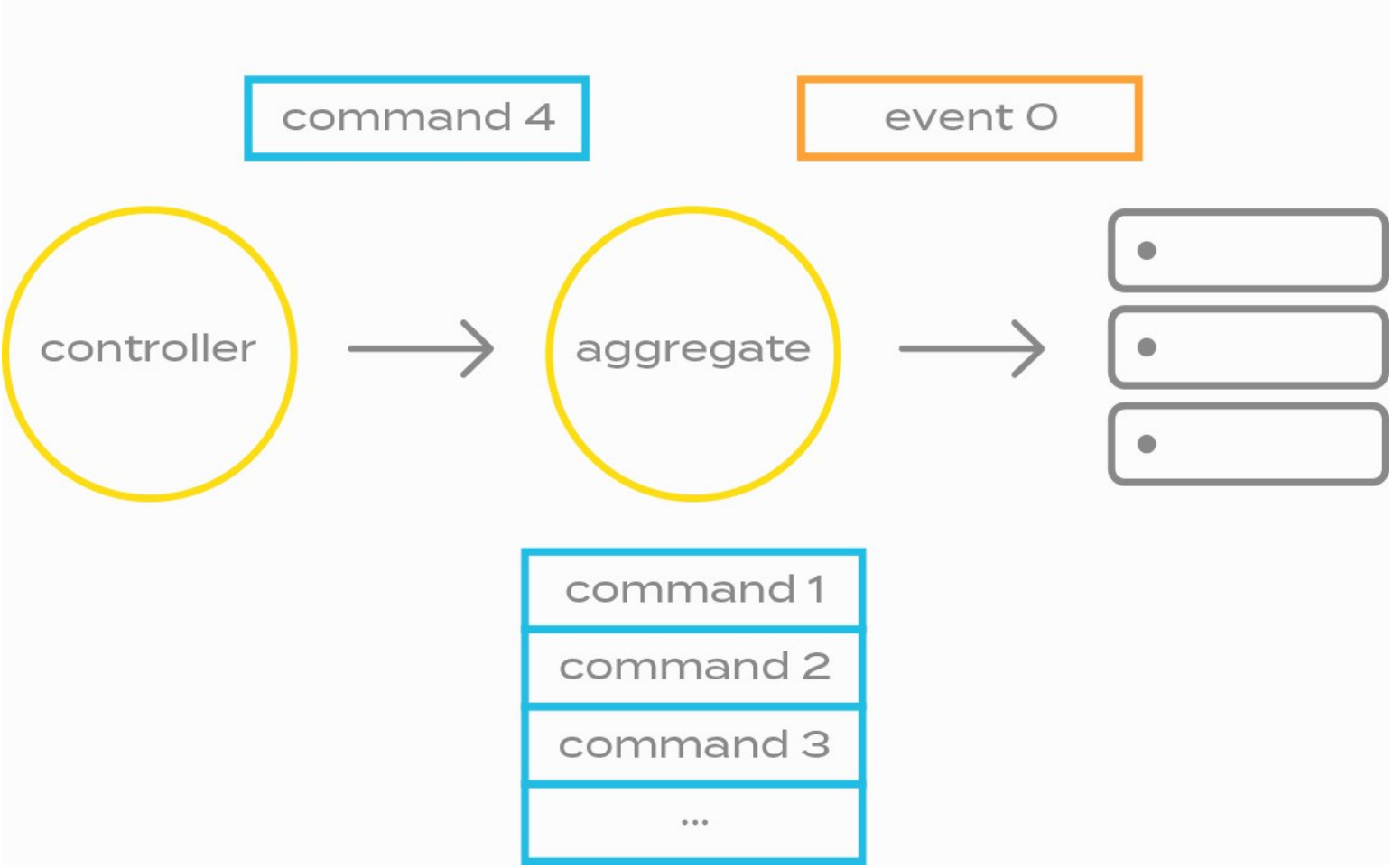



@VaughnVernon

# Architecture

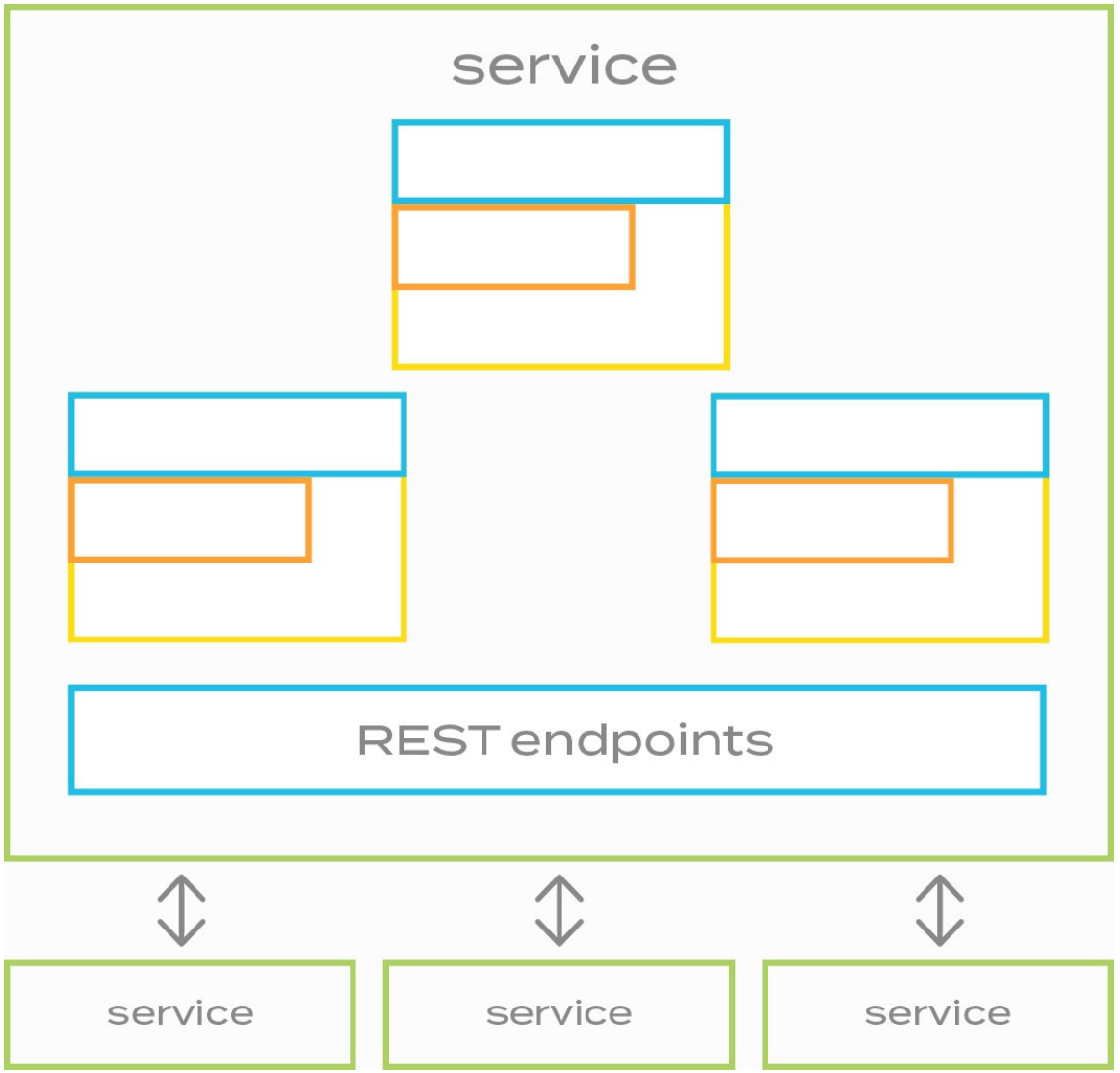


# vlingo/actors




@VaughnVernon

# vlingo/http




@VaughnVernon

# vlingo/lattice


@VaughnVernon

# vlingo/streams


@VaughnVernon

# Open Source Reactive Platform


Contact: vaughn@kalele.io

Web: vlingo.io

**@VaughnVernon**