Domain Driven Design

die Reinigungskraft für Source Code


KEEP
CALM
AND DON'T
WASTE
MY TIME

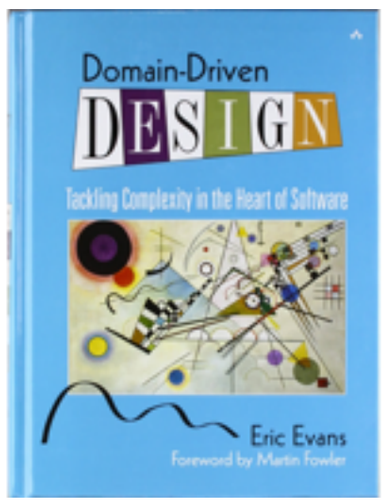

Auch Du
hältst den Code sauber,
Genosse!

Domain Driven Design

Eric Evans

Software Engineer

Buchautor



Ubiquitous Language

Domain

Bounded Context

Aggregate

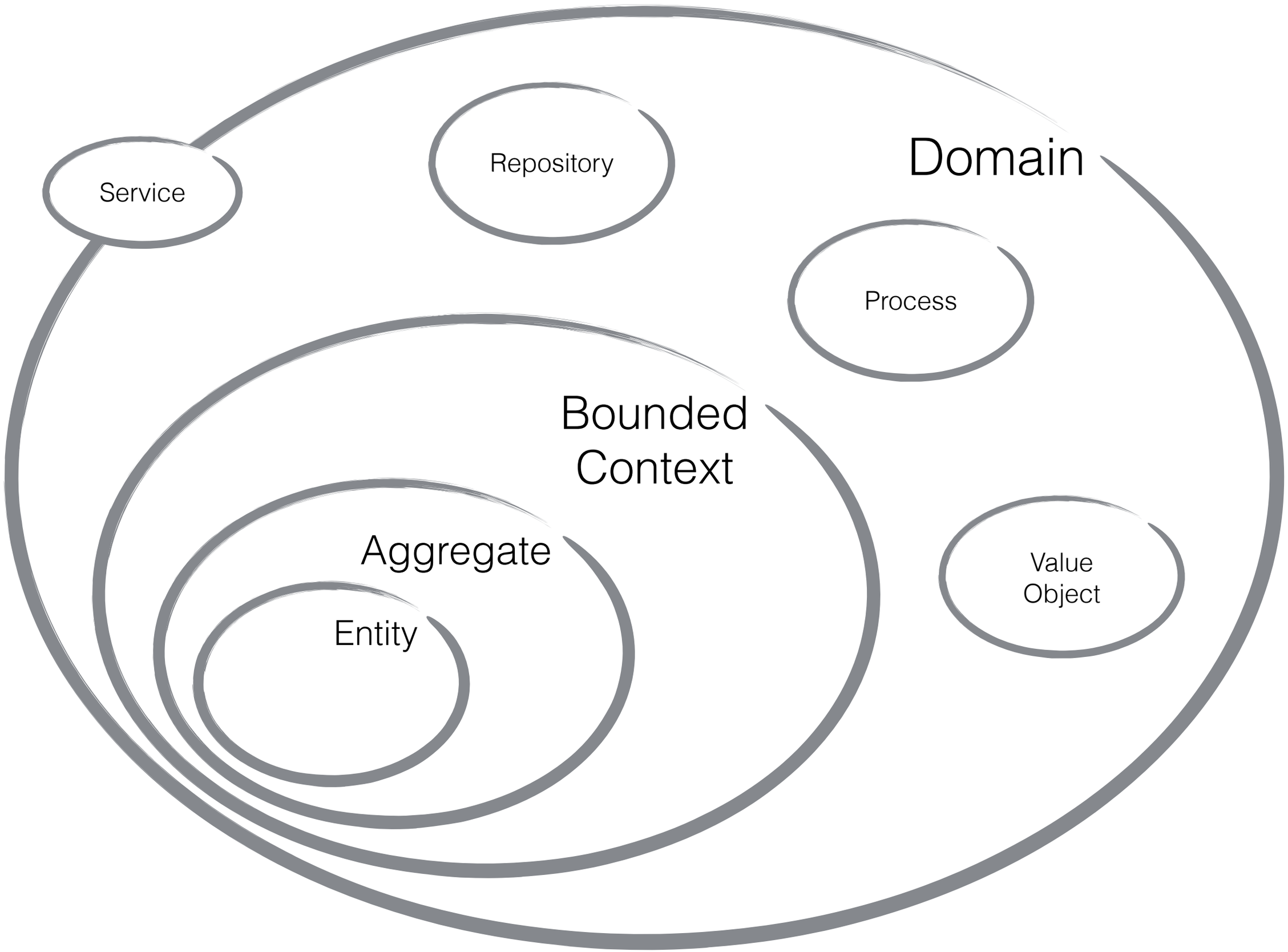
Entity

Value Object

Process

Service

Repository


Service
Repository
Domain
Process
Bounded
Context
Aggregate
Entity
Value
Object


Infrastructure
Domain
UI

FACHLICHKEIT

in die

DOMAIN

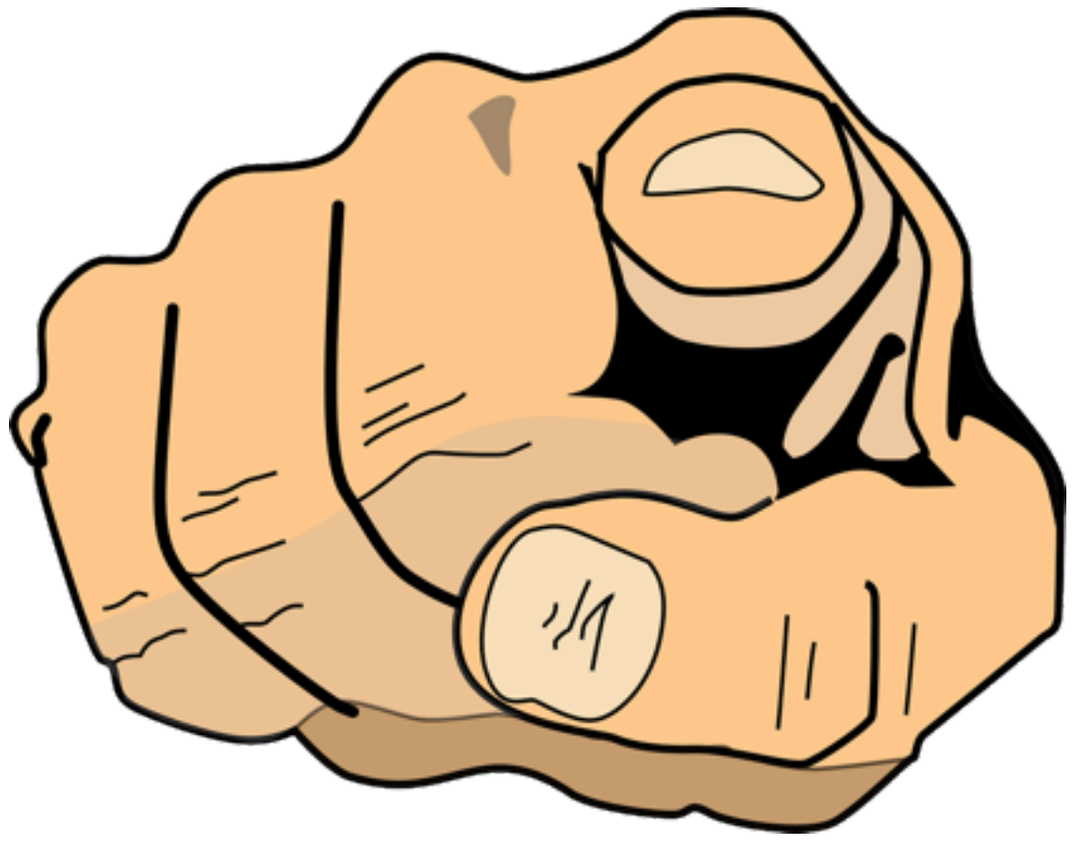
DER REST

in die

INFRASTRUCTURE

oder

UI



Darf es ein wenig Code sein?

Service

```
public class Mietvertrag {

    public void Vorbereiten(ID, Adresse){
        …
    }

    …
}
```

Aggregate

```
internal class Mietvertrag {
    ...
}
```

Entity

```
internal class Vertrag {
   …
}
```

```
internal class Mieter {
   …
}
```

Value Object

```
internal struct Adresse {
    ...
}
```

Repository

```
public class Mietverträge {
    public IEnumerable<Vertrag> Alle(){
        …
    }
    …
}
```

Projektstruktur

```
Solution
  * Contracts
    * Models
  * Domain
    * Bounded Context
      * Services
      * Repositories
      * Aggregates
    * ValueObjects
    * Processes
  * UI
    * Views
    * Controller
```

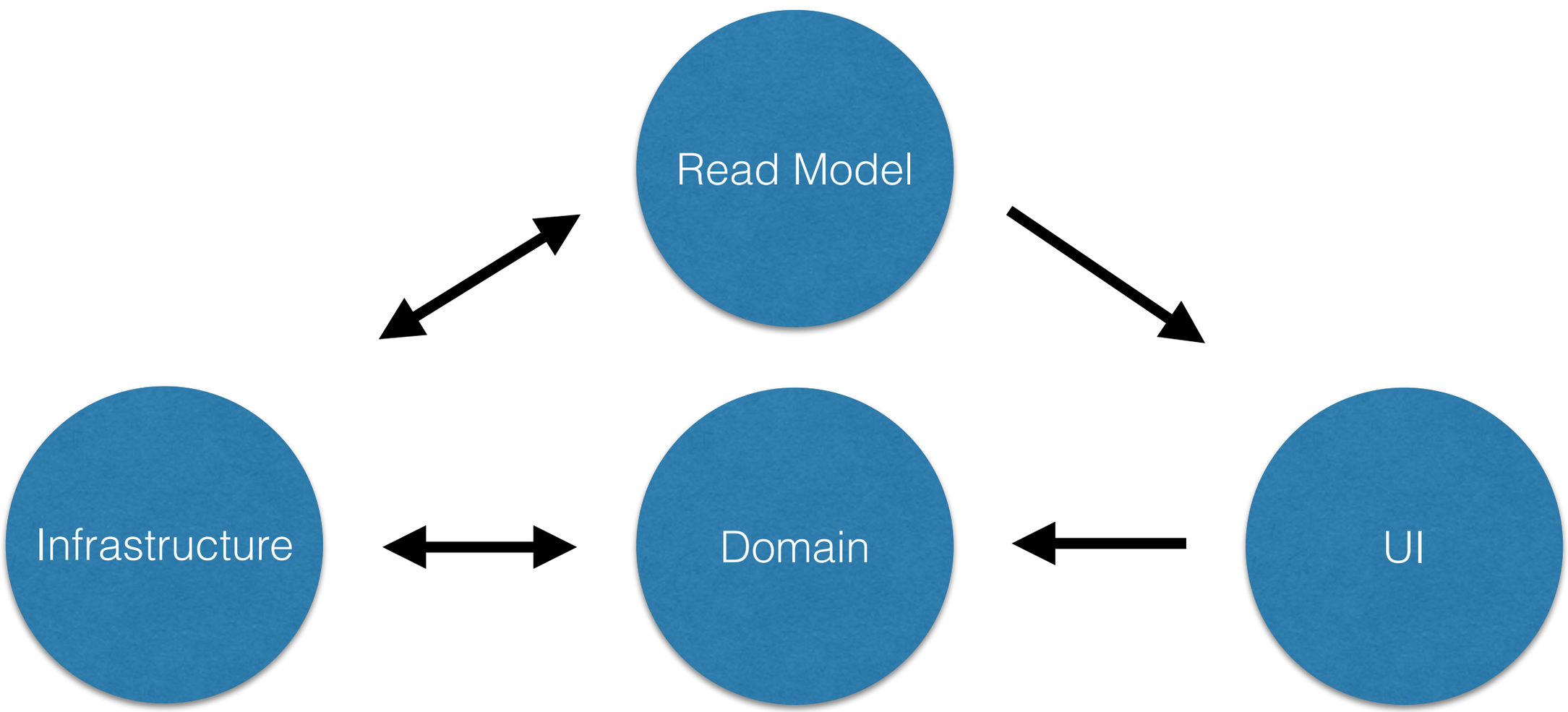
Enterprise Software

Enterprise Software im DDD Style

Was kommt danach?

EventSourcing

CQRS


Read Model
Infrastructure
Domain
UI

Command

Event

Event Store

Wenn ich mich entscheiden sollte, ich würde nur noch DDD verwenden.

Vielen Dank für die Aufmerksamkeit

Ich bin erreichbar unter
Twitter: @janekf
EMail: jan.fellien@devcrowd.de