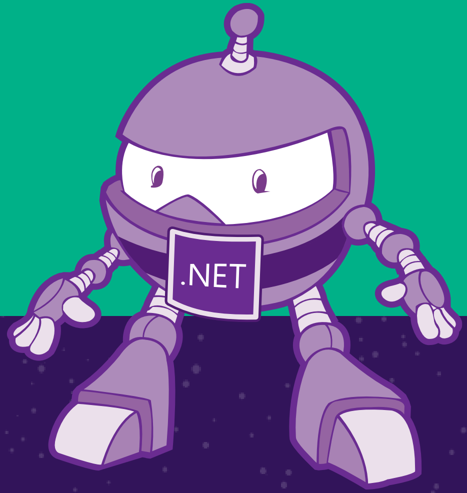
Clean Architecture in ASP.NET Core

Bogdan Shvets



Bogdan Shvets

Senior SE at SoftServe

I don’t have Twitter

I don’t have my own blog

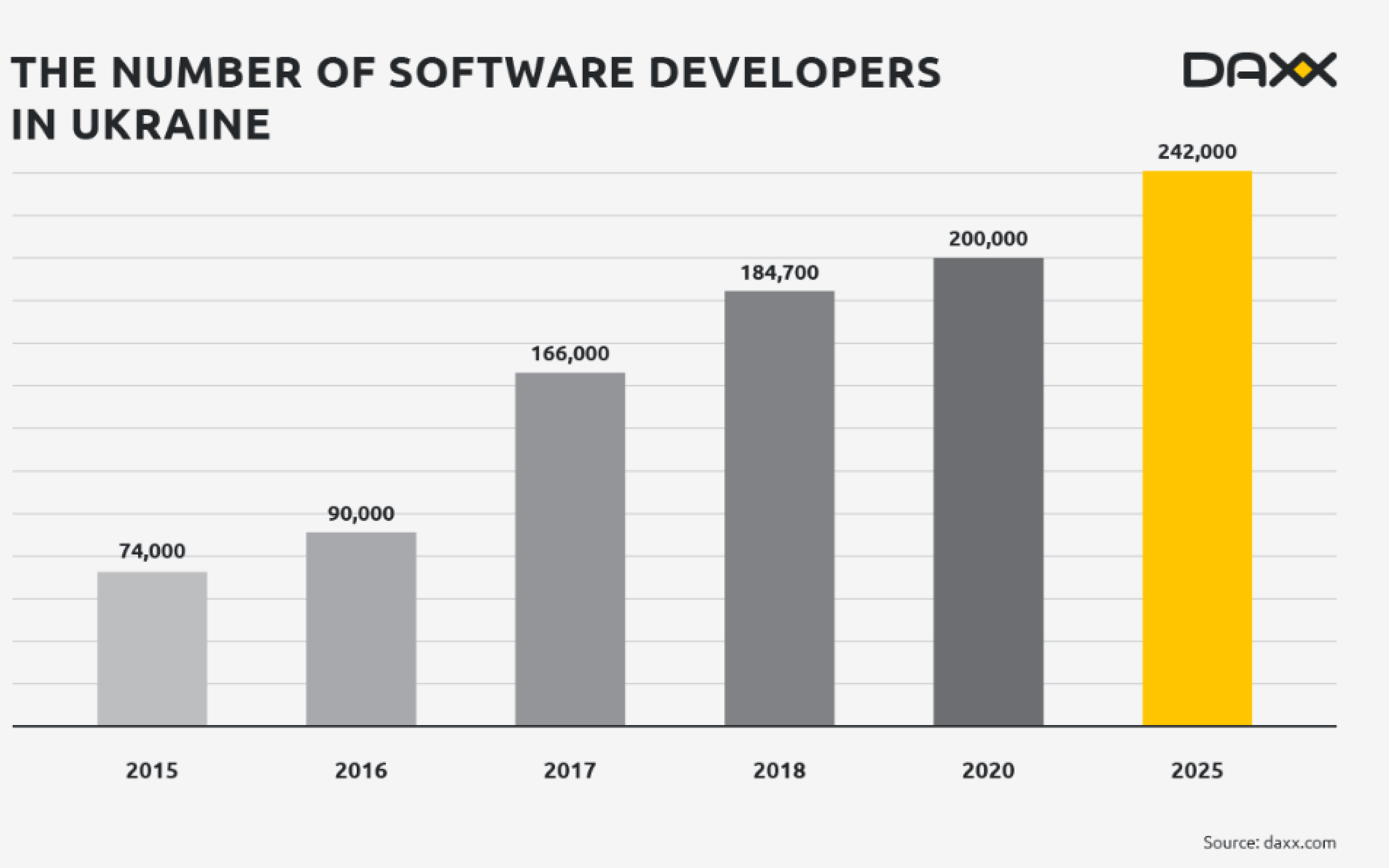
I don’t have my YouTube channel

But I have GitHub account, not too much there, I just have it…

AGENDA

- Why we need architecture?
- Basics, Business Rules, Layers and components
- Clean Architecture overview
- Ways to implement

Why bother with Architecture?

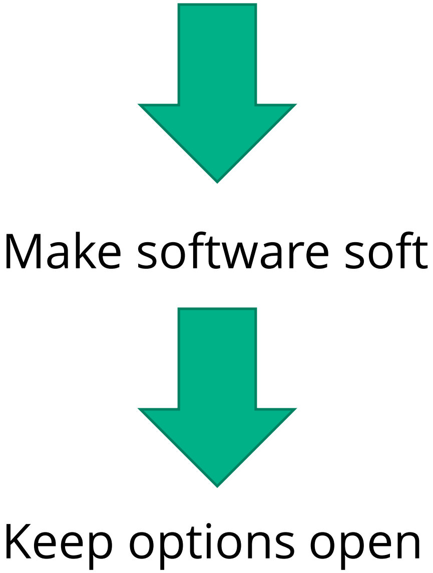


THE NUMBER OF SOFTWARE DEVELOPERS IN UKRAINE

Source: daxx.com

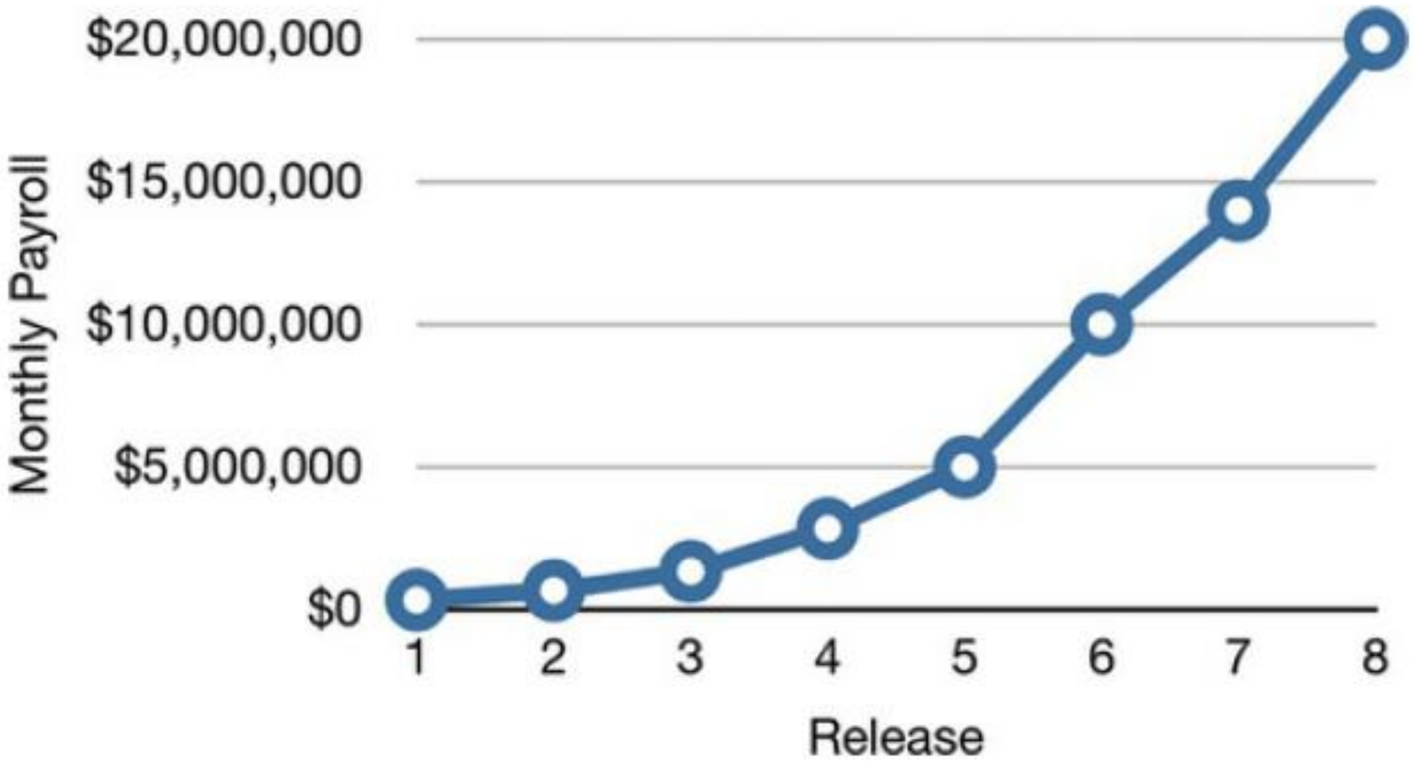
Goals of optimal architecture

- Facilitate development
- Facilitate deployment
- Facilitate maintenance



Make software soft

Keep options open



If you think good architecture is expensive, try bad architecture.
—Brian Foote and Joseph Yoder

Clean Architecture:
Standing on the shoulders of giants

Robert Martin

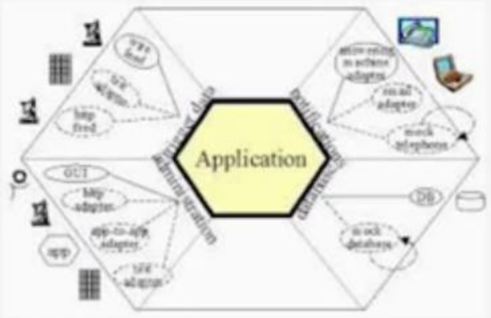
Robert C. Martin Series

PRENTICE HALL

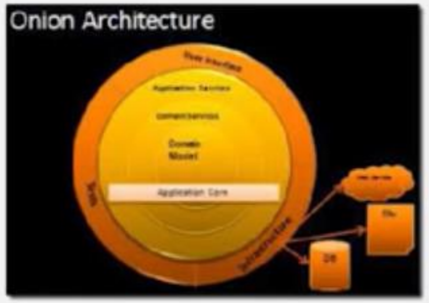
Clean Architecture

Robert C. Martin

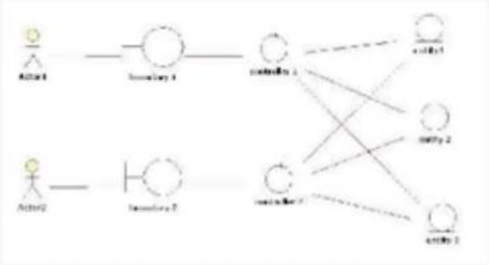
Alistair Cockburn



Jeffrey Palermo



Ivar Jacobson



How we will get to Clean Architecture

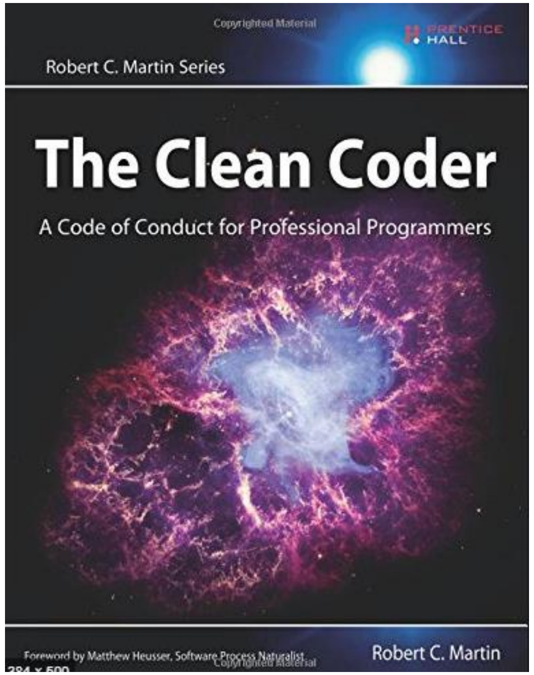
The Bricks

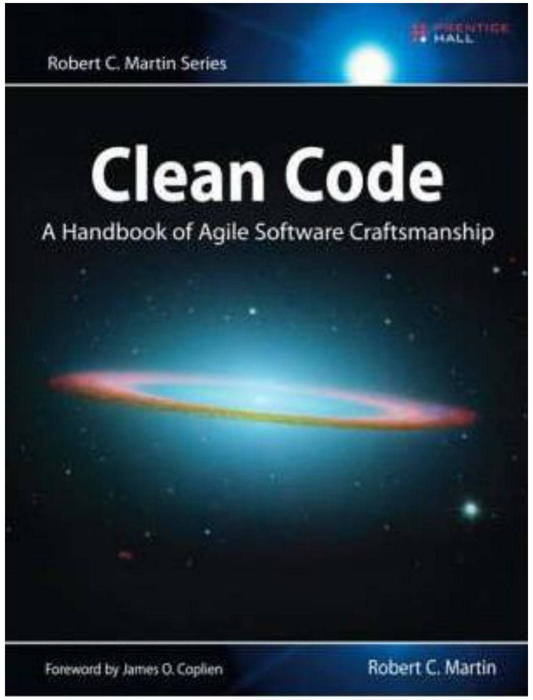
The Walls

The Building

Clean Code

- Not today
- Better read a book
- Or at least read an article

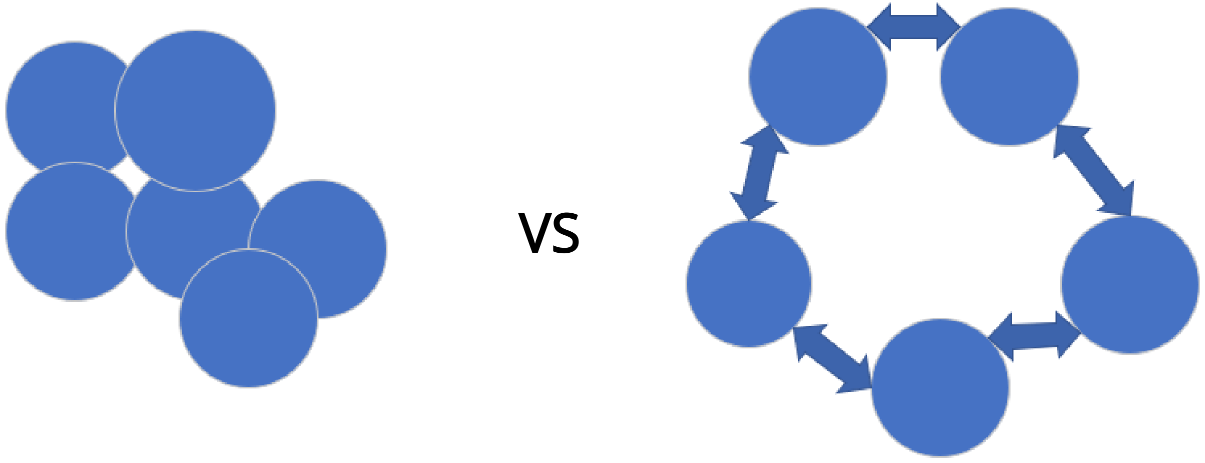

Copyrighted Material
PRENTICE HALL
Robert C. Martin Series
The Clean Coder
A Code of Conduct for Professional Programmers
Foreword by Matthew Heusser, Software Process Naturalist
Robert C. Martin


PRENTICE HALL
Robert C. Martin Series
Clean Code
A Handbook of Agile Software Craftsmanship
Foreword by James O. Coplien
Robert C. Martin

Loosely coupling

Coupling is the property that one module is forced to change because another does

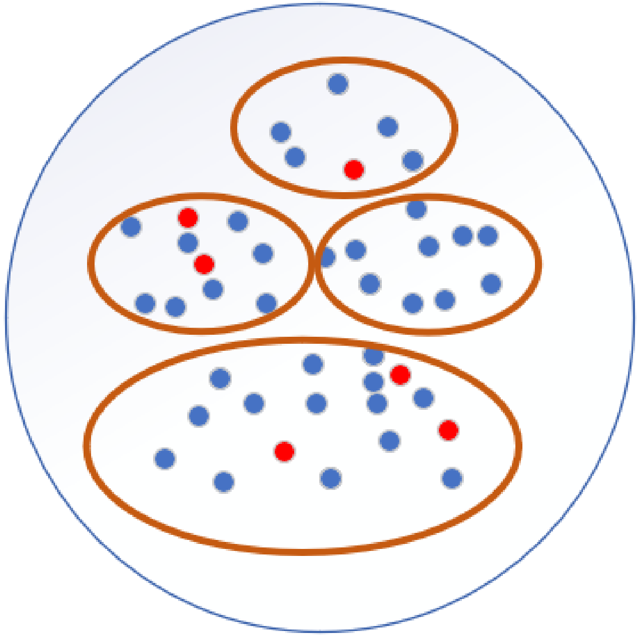
Two modules can be said to be **loosely coupled** if changes to the implementation details of one do not impact the implementation of the other



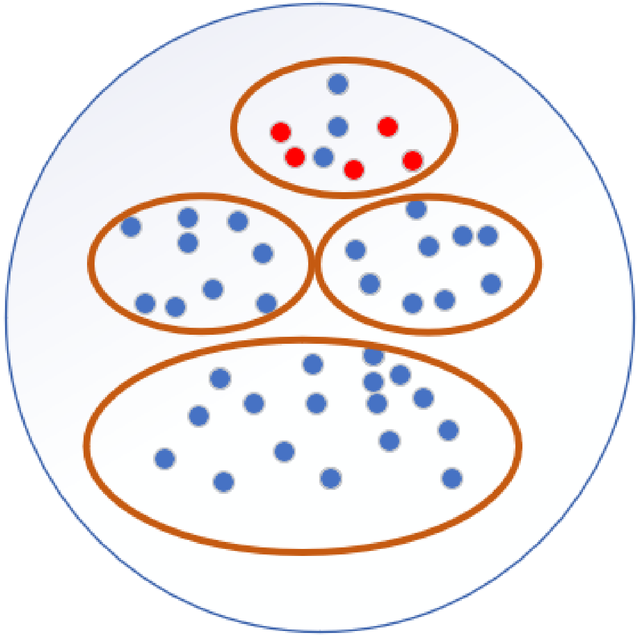
VS

High cohesion

Cohesion is the property that a module is subject to the same forces of change



Low Cohesion



High Cohesion

softserve

Single-Responsibility Principle

Open-Closed Principle

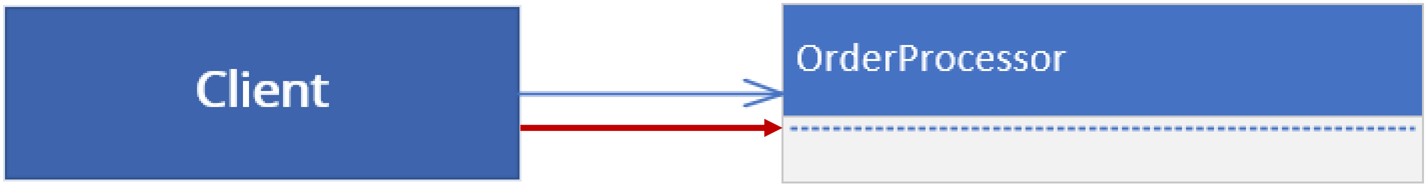
Liskov Substitution Principle

Interface Segregation Principle

Dependency Inversion Principle

Dependency Inversion Principle

High level modules should not depend on low level modules;
both should depend on abstractions.

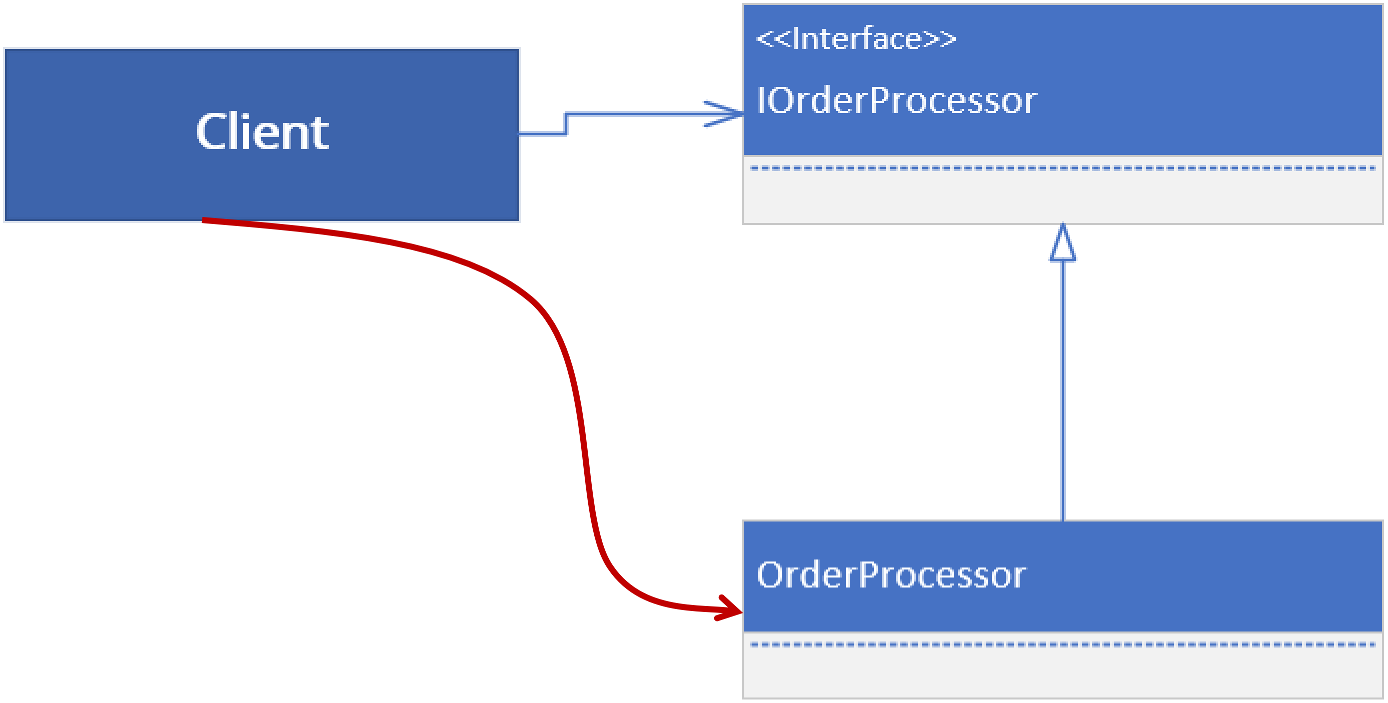


Client

OrderProcessor

⟶ - using

→ - flow



We can choose the direction of any code dependency!

Single Responsibility Principle

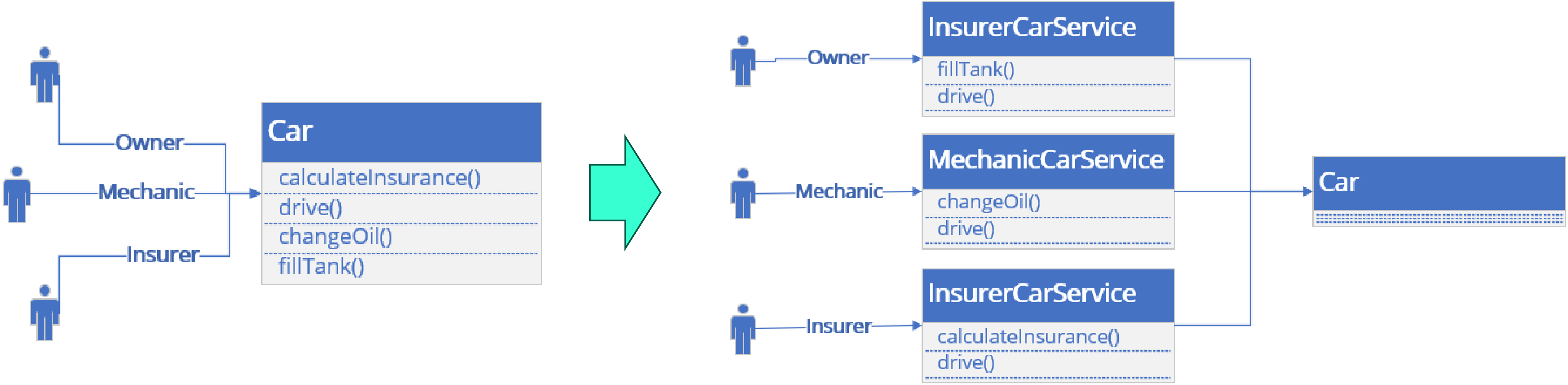
A module should only do one thing.

Single Responsibility Principle

~~A module should only do one thing.~~

A module should only have one reason to change.

Single Responsibility Principle



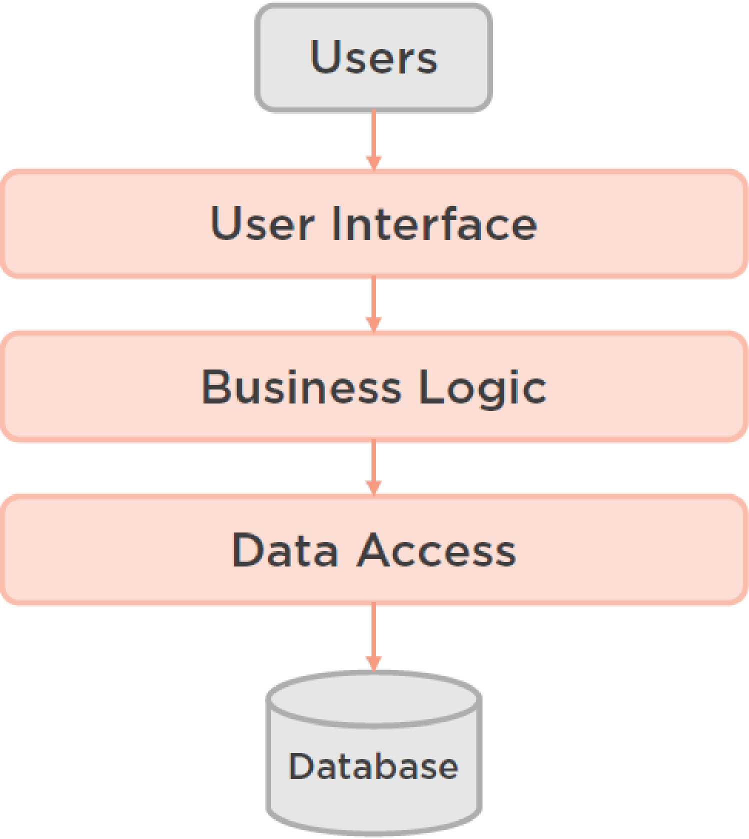
Separate the code that different actors depend on.

softserve

Business Rules

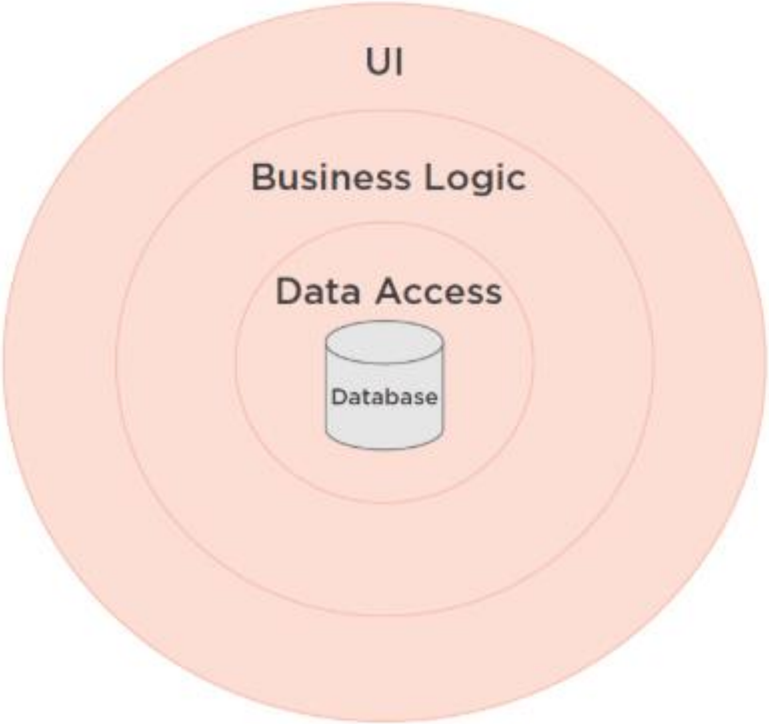
- Domain Rules
- Application Rules

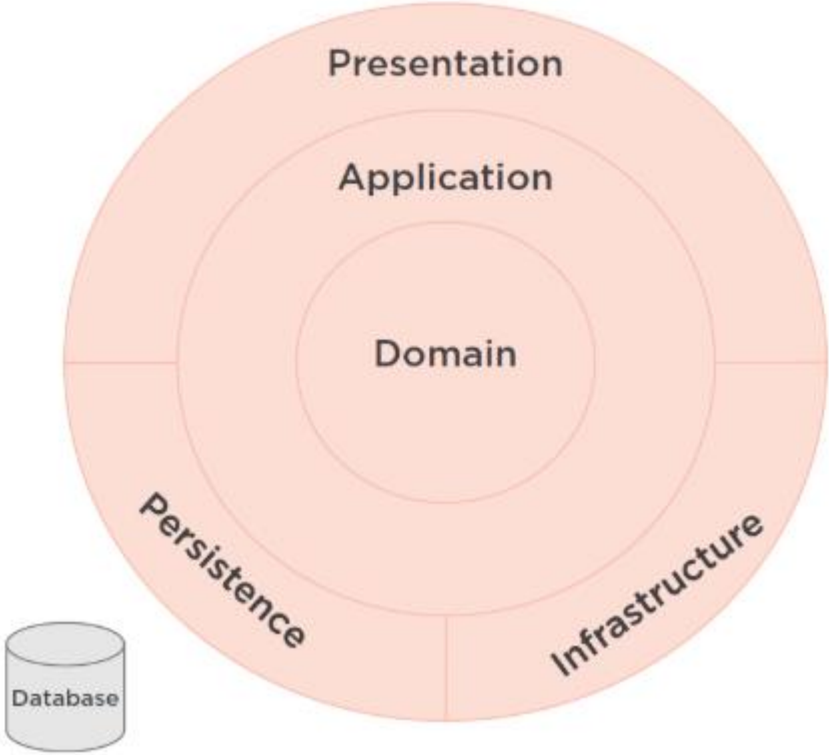
Layers and components



softserve

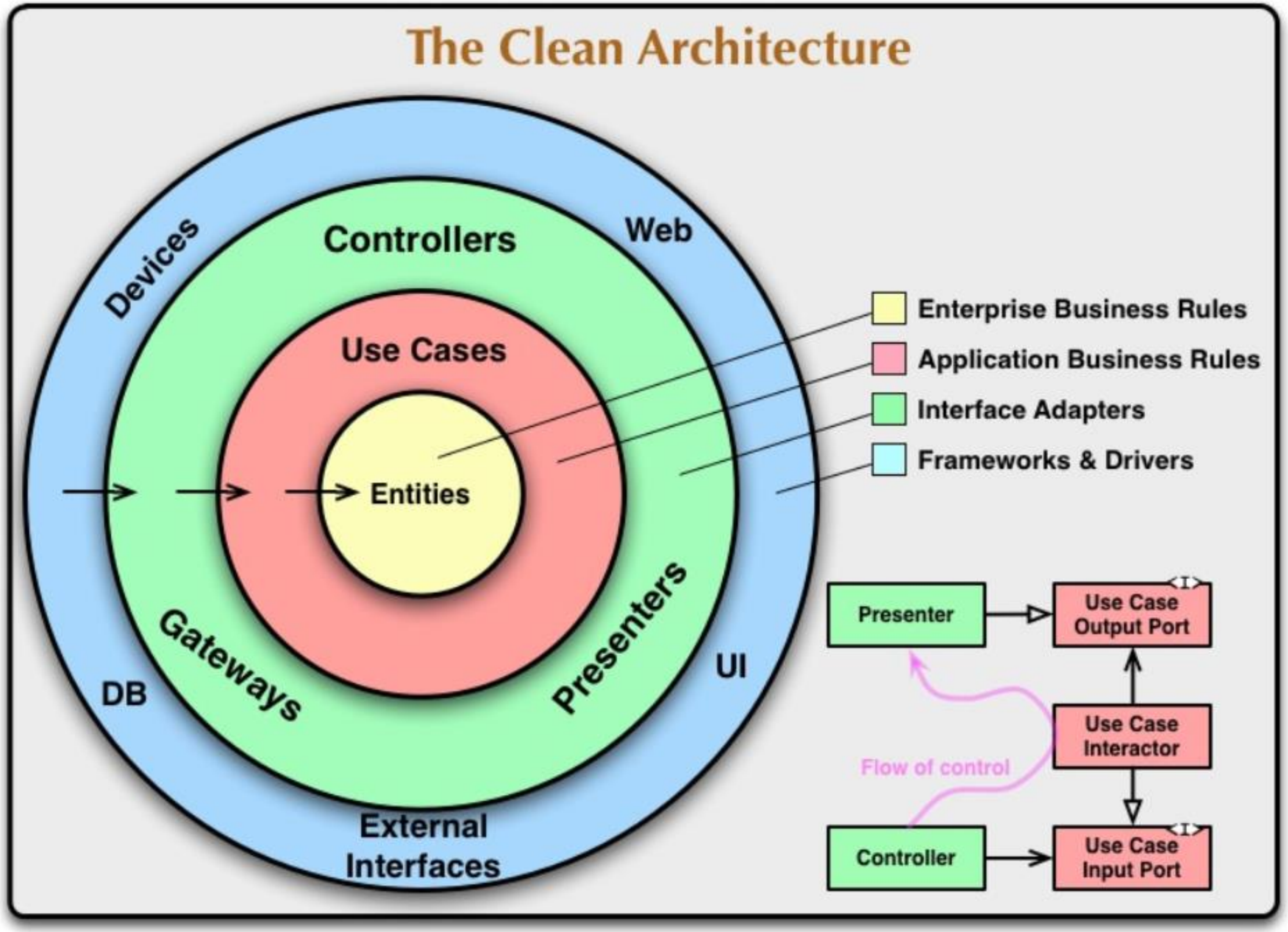
Database-Centric vs Domain-Centric



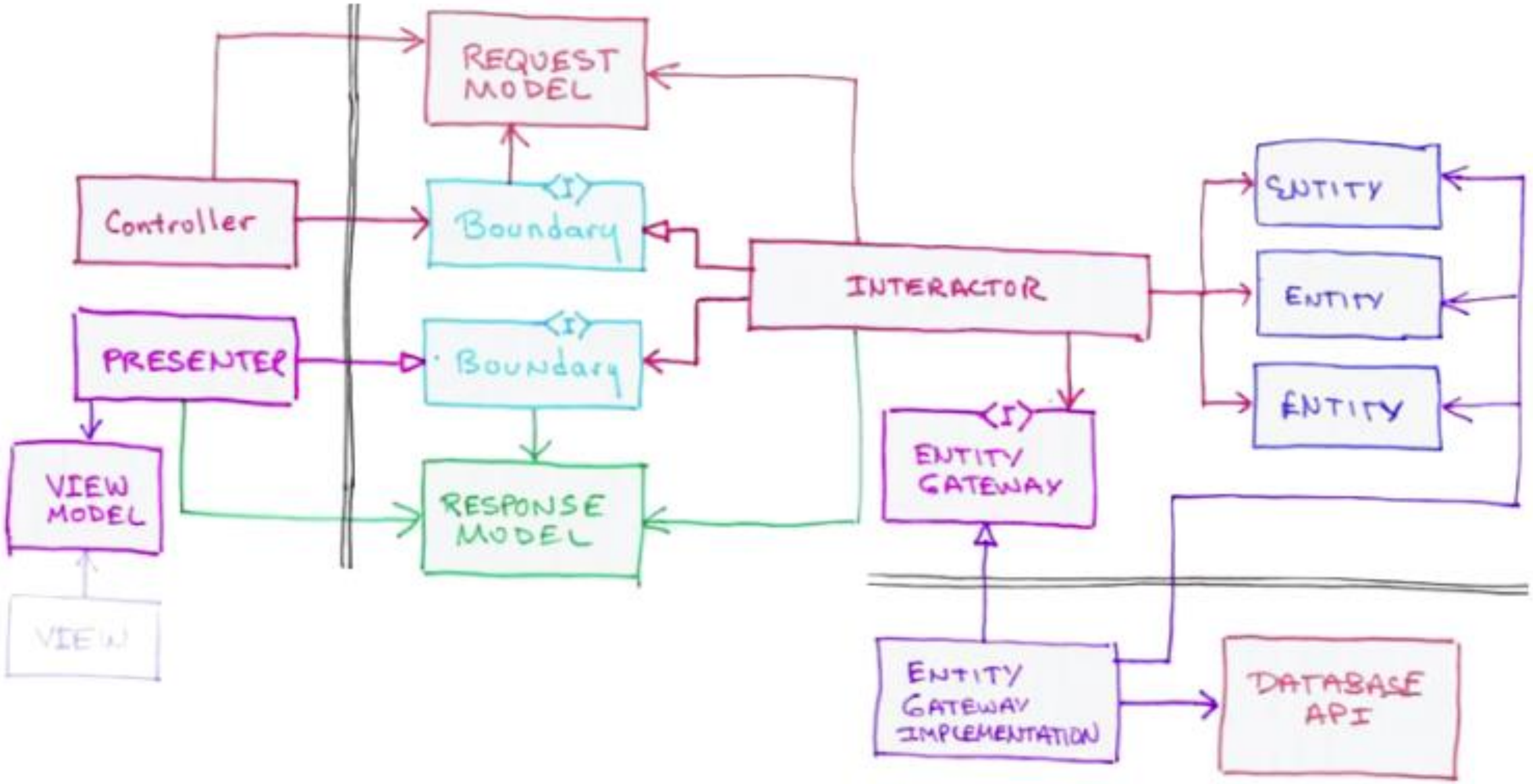


The First concern of the architect is to make sure that the house is usable, it is not to ensure that the house is made of brick.
- Uncle Bob

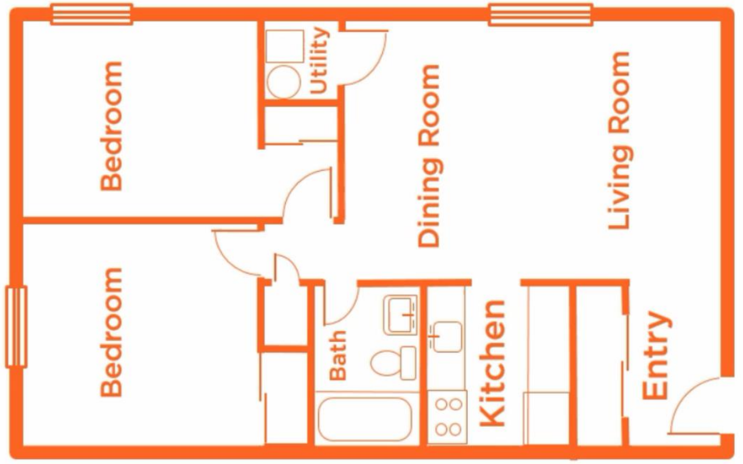
Clean Architecture



softserve


REQUEST MODEL
Controller
<I>
Boundary
INTERACTOR
ENTITY
ENTITY
ENTITY
PRESENTER
<I>
Boundary
<I>
ENTITY GATEWAY
VIEW MODEL
RESPONSE MODEL
VIEW
ENTITY GATEWAY IMPLEMENTATION
DATABASE API

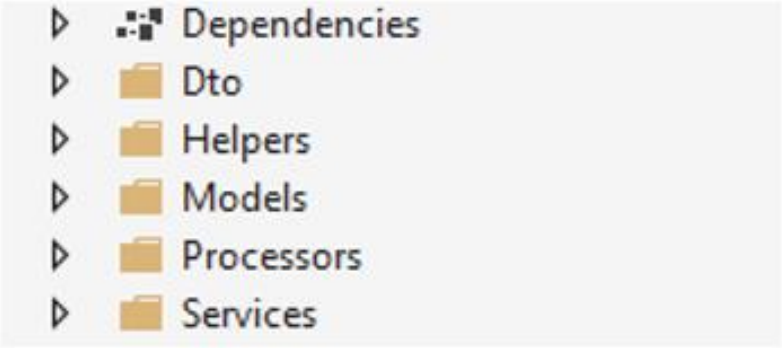
Screaming Architecture



The Architecture should scream the intent of the system!

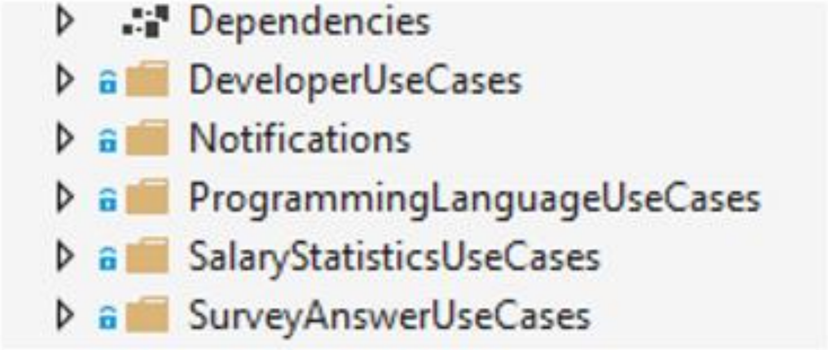
- Uncle Bob

softserve

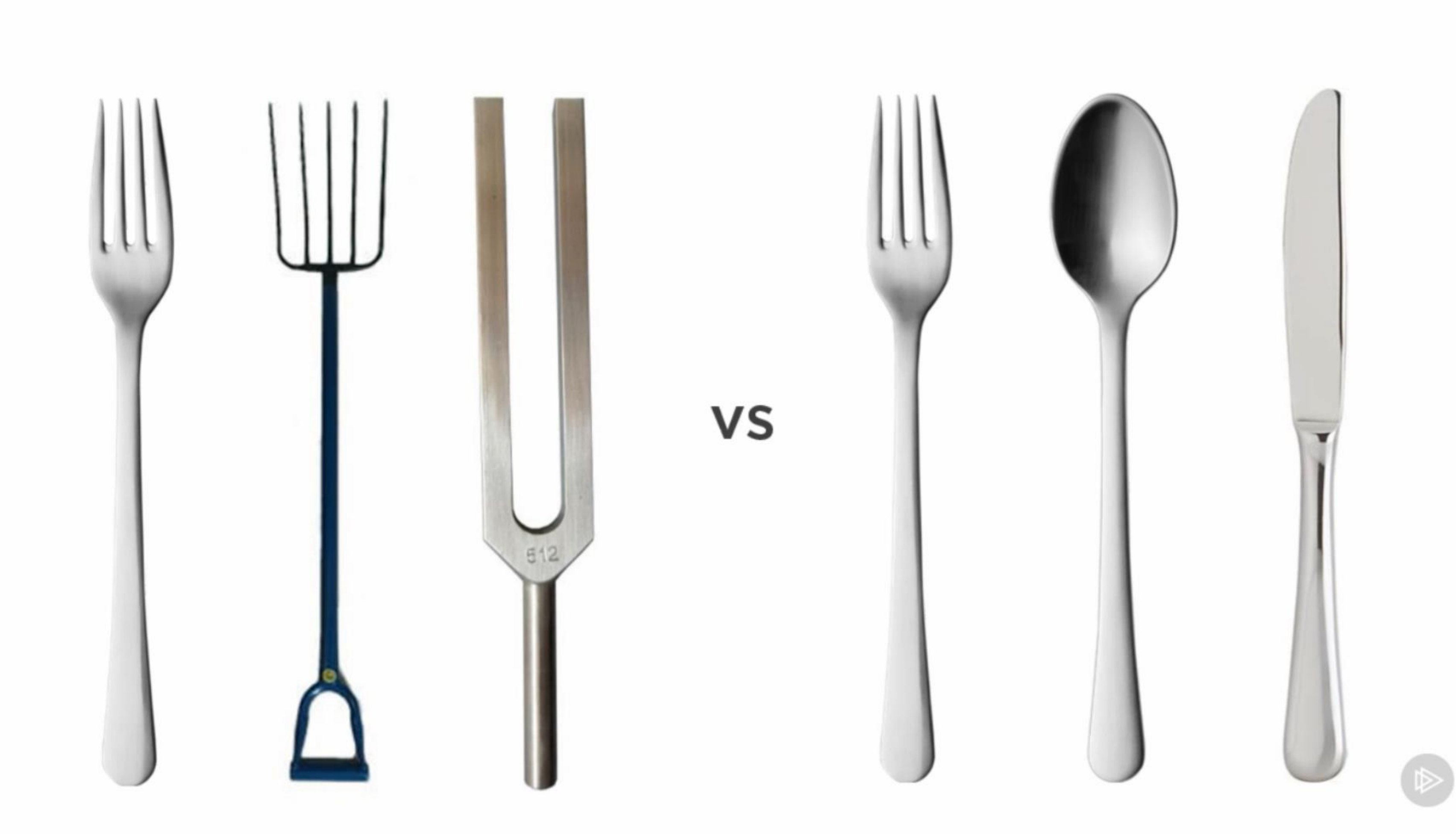


- Dependencies
- Dto
- Helpers
- Models
- Processors
- Services

VS

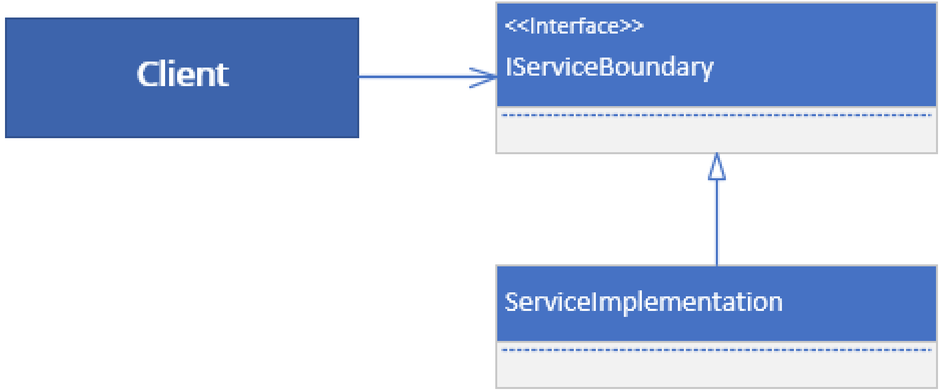


- Dependencies
- DeveloperUseCases
- Notifications
- ProgrammingLanguageUseCases
- SalaryStatisticsUseCases
- SurveyAnswerUseCases

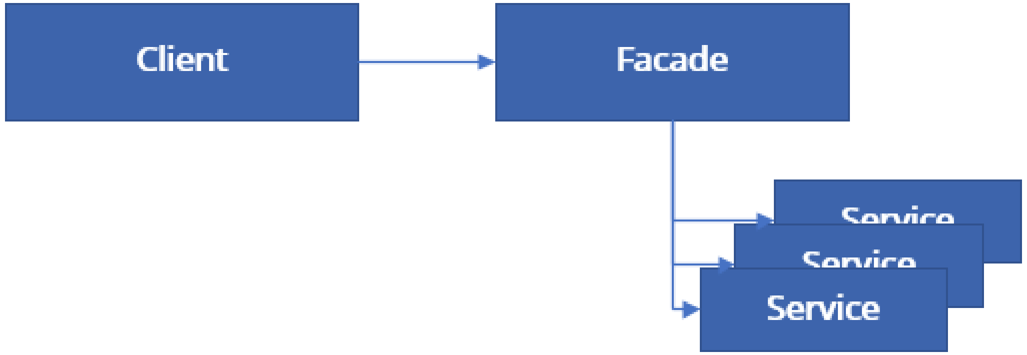

512
vs

Boundaries

1.



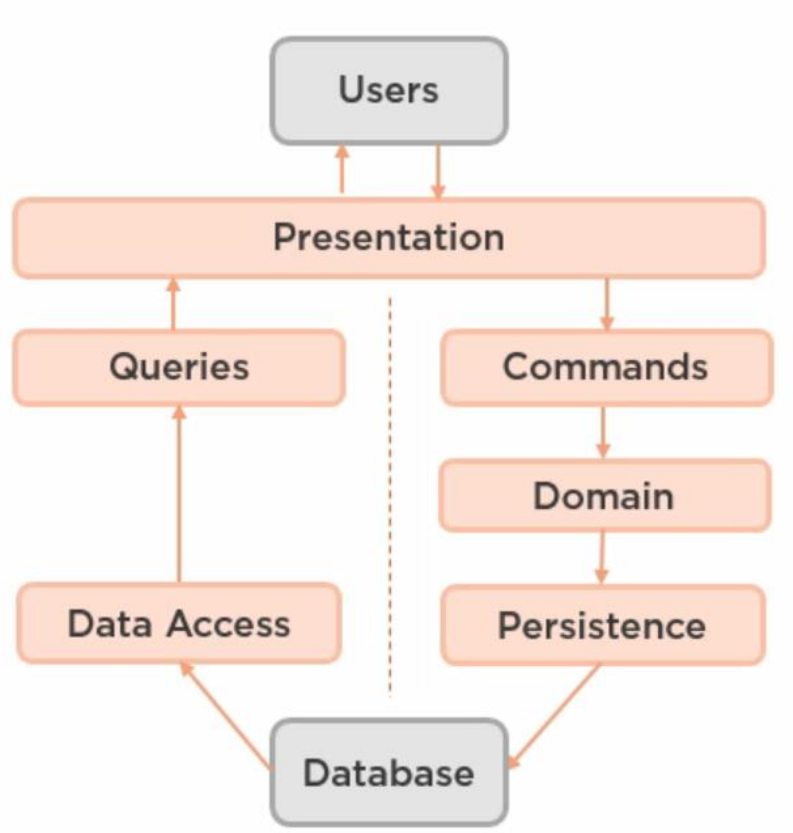
2.



3. Just skip boundaries

CQRS

Command Query Responsibility Segregation



DEMO

softserve

Way #1. Almost Ideal

I tried so hard
And got so far
But in the end
I got tired

Way #2. Using MediatoR

Jason Taylor: Clean Architecture with ASP.NET Core 2.2

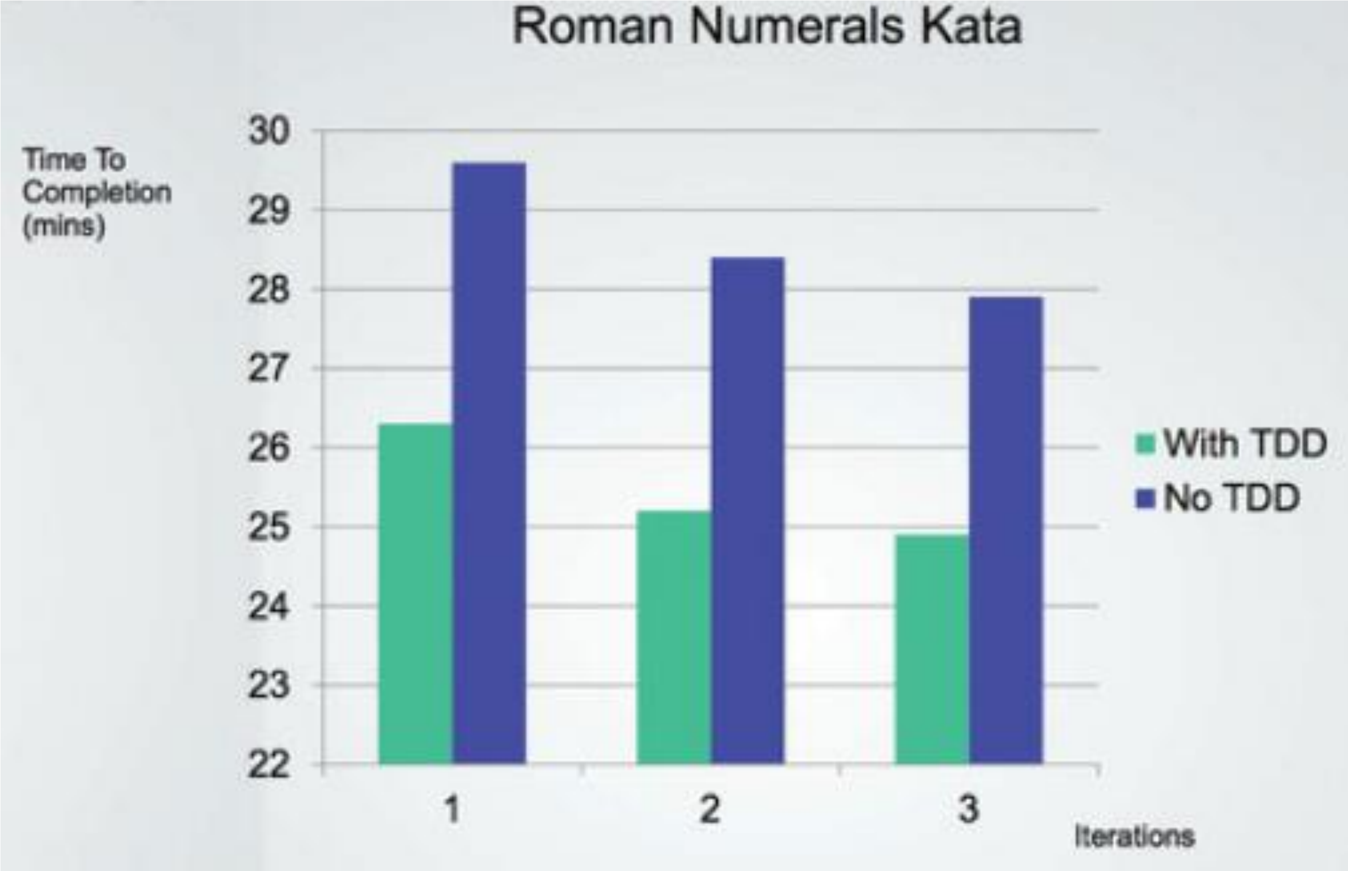
https://bit.ly/northwind-traders

Way #3. Quick

- Less boundaries
- Less CQRS

TDD

Test-Driven Development by **Jason Gorman**



softserve

Source: http://www.codemanship.co.uk/parlezuml/blog/?postid=1021

Summary

Clean Architecture (CA) just puts the business domain at the center

Compromises can be made. Only Four Circles?

CA is expensive

Do not use if you have only CRUD operation

Useful links and sources

YouTube

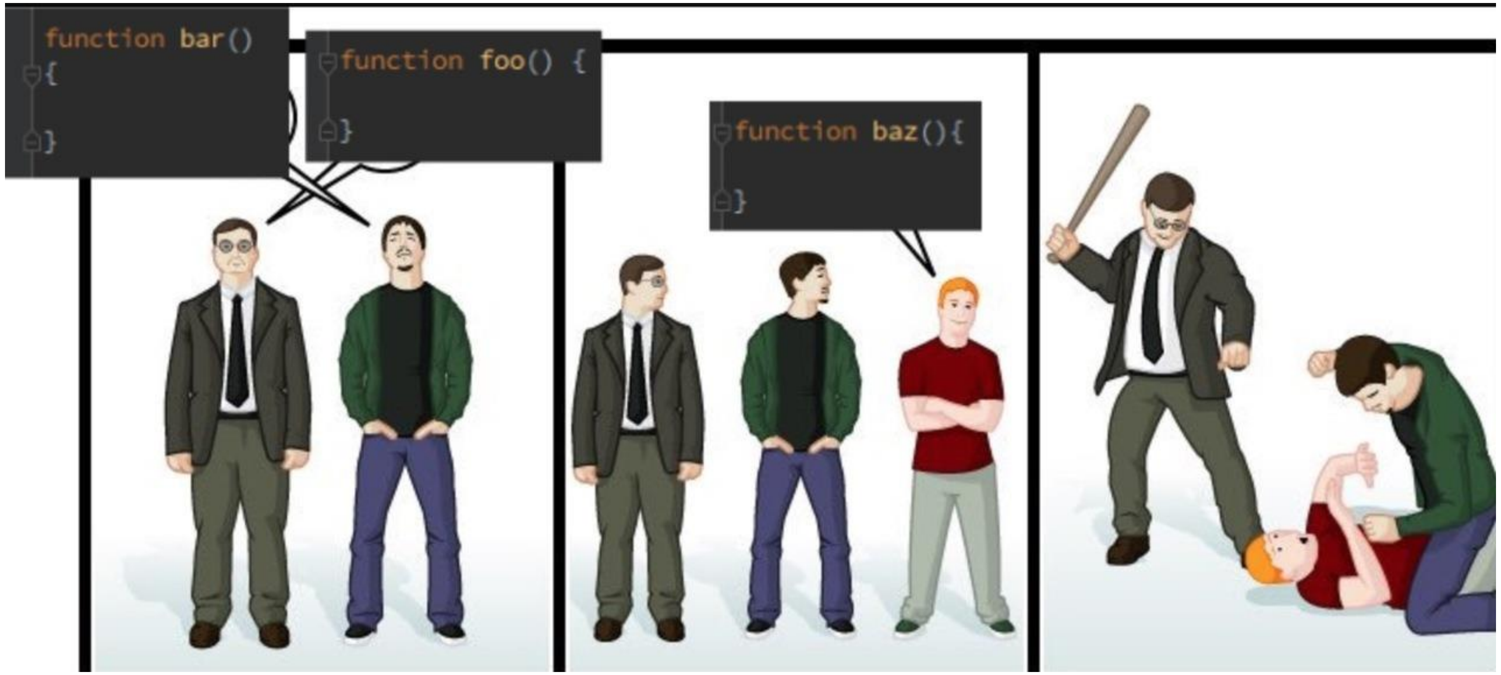
1. Uncle Bob about CA - https://www.youtube.com/watch?v=2dKZ-dWaCiU
2. Ian Cooper about CA https://www.youtube.com/watch?v=IAcxetnsiCQ

Articles and GitHub

1. CA: Standing on the shoulders of giants - https://herbertograca.com/2017/09/28/clean-architecture-standing-on-the-shoulders-of-giants/
2. Matthew Renze GitHub. Implemention and Pluralsight training - https://github.com/phongnguyend/Practical.CleanArchitecture
3. My GitHub with presentation and demos - https://github.com/bshvets/CleanArchitecture
4. https://fullstackmark.com/post/18/building-aspnet-core-web-apis-with-clean-architecture
5. Hexagon Architecture - https://blog.ndepend.com/hexagonal-architecture/
6. One more CA - https://pusher.com/tutorials/clean-architecture-introduction

softserve

FIGHT FOR THE ARCHITECTURE



Thank you!