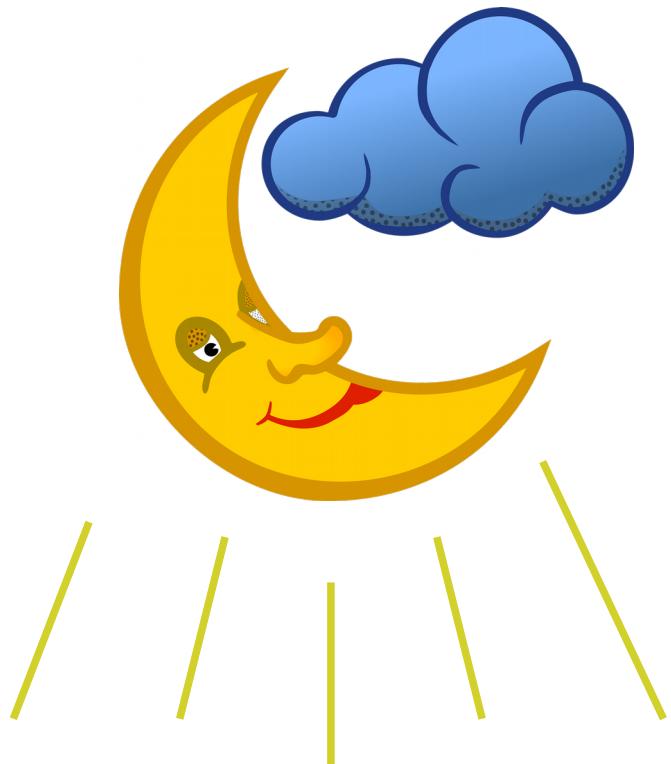


The Gospel of Uncle Bob and Kent Beck: Clean Architecture and TDD

“ ... he who never quotes will
never be quoted... ”

- Charles Spurgeon



New Things

“... everything I see under
the sun is not new.”

- from Ecclesiastes (paraphrased)

bad news

“Code as if the one who will inherit
your code base is a psychopathic
mass murderer who knows your
home address.”

- Anonymous

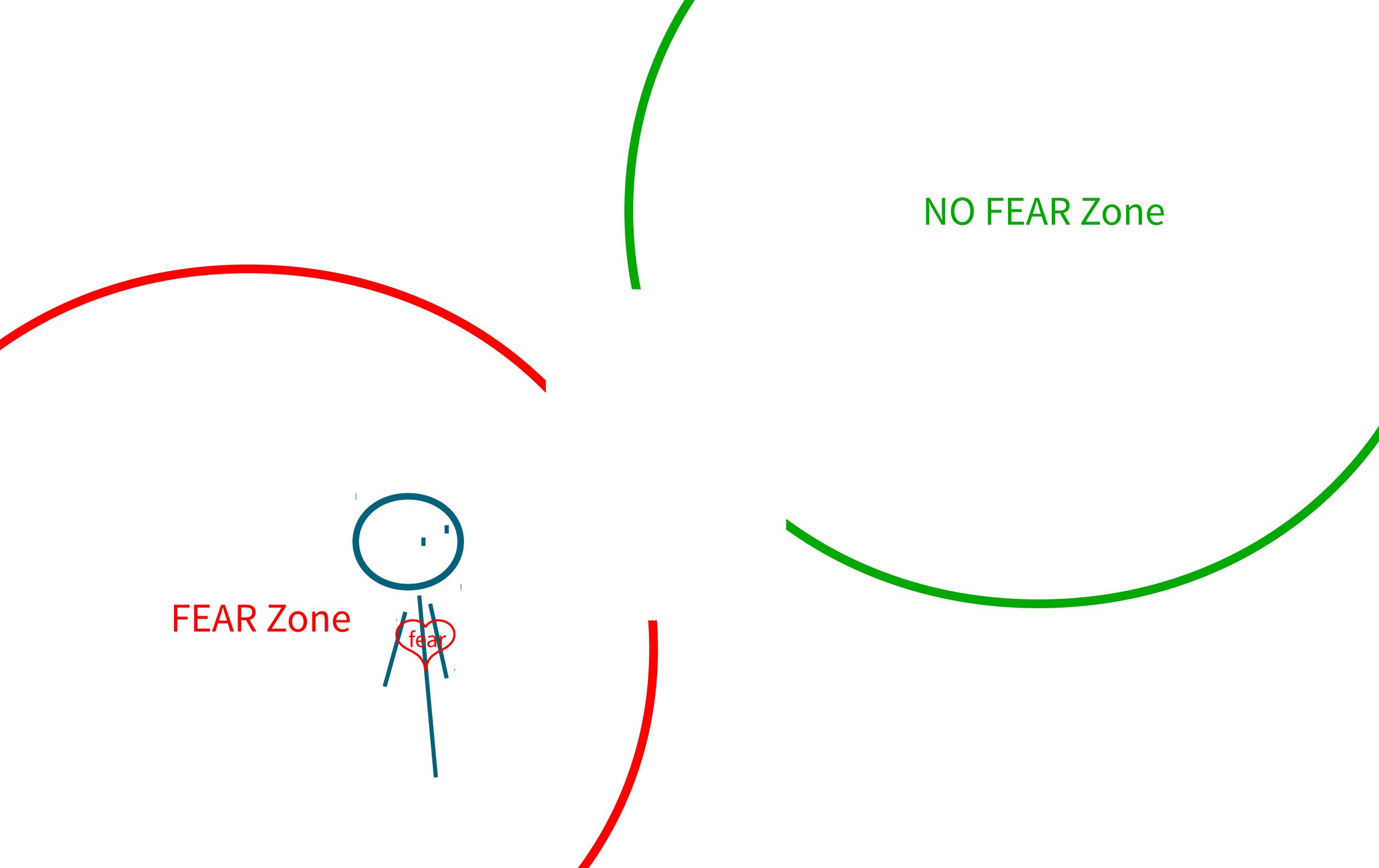
“Code as if the one who will inherit
your code base is a gossip
who has lots of gossip friends”

- Anonymous

Be conscientious when you write
code

We fear that we become slaves of
our creations

“There is a fear that produces
NO FEAR”



NO FEAR Zone

FEAR Zone

FEAR Zone

NO FEAR Zone



solution

“Nothing makes a system more flexible than a suite of tests. Nothing.

Good architecture and design are important; but the effect of a robust suite of tests is an order of magnitude greater. It's so much greater because those tests enable you to improve the design.”

- Uncle Bob Martin

“Code without tests is bad code.

It doesn't matter how well written it is; it doesn't matter how pretty or object-oriented or well-encapsulated it is.

With tests, we can change the behavior of our code quickly and verifiably.

Without them, we really don't know if our code is getting better or worse.”

- Michael Feathers

```
public class GreetingCreatorTests
{
    [Fact]
    public void should_create_correct_greeting()
    {
        // arrange
        var greetingCreator = new GreetingCreator();

        // act
        string greeting = greetingCreator.Greet("Name");

        // assert
        Assert.Equal("Hello, Name!", greeting);
    }
}
```



I wish I had a
CreatingCreator
class that can
create greetings

```
public class GreetingCreator
{
    public string Greet(string name)
    {
        return "Hello, Name!";
    }
}
```



My wish is my
command

```
public class GreetingCreatorTests
{
    [Fact]
    public void should_create_correct_greeting()
    {
        // arrange
        var greetingCreator = new GreetingCreator();

        // act
        string greeting = greetingCreator.Greet("Name");

        // assert
        Assert.Equal("Hello, Name!", greeting);
    }
}
```



```
public class GreetingCreator
{
    public string Greet(string name)
    {
        return "Hello, " + name;
    }

}
```



I wish I can
confidently change
my production code

```
public class GreetingCreatorTests
{
    [Fact]
    public void should_create_correct_greeting()
    {
        // arrange
        var greetingCreator = new GreetingCreator();

        // act
        string greeting = greetingCreator.Greet("Name");

        // assert
        Assert.Equal("Hello, Name!", greeting);
    }
}
```



```
public class GreetingCreator
{
    public string Greet(string name)
    {
        return "Hello, " + name + "!";
    }
}
```

```
public class GreetingCreatorTests
{
    [Fact]
    public void should_create_correct_greeting()
    {
        // arrange
        var greetingCreator = new GreetingCreator();

        // act
        string greeting = greetingCreator.Greet("Name");

        // assert
        Assert.Equal("Hello, Name!", greeting);
    }
}
```



```
public class GreetingCreator
{
    public string Greet(string name)
    {
        return string.Format("Hello, {0}!", name);
    }

}
```



```
public class GreetingCreatorTests
{
    [Fact]
    public void should_create_correct_greeting()
    {
        // arrange
        var greetingCreator = new GreetingCreator();

        // act
        string greeting = greetingCreator.Greet("Name");

        // assert
        Assert.Equal("Hello, Name!", greeting);
    }
}
```



“You give me an application that is beautifully designed but there's no tests... I'll be **afraid to touch it**. It will rot.”

“You give me an application that is badly designed but has a **comprehensive suite of tests**... I can make the design better because I have a suite of tests.

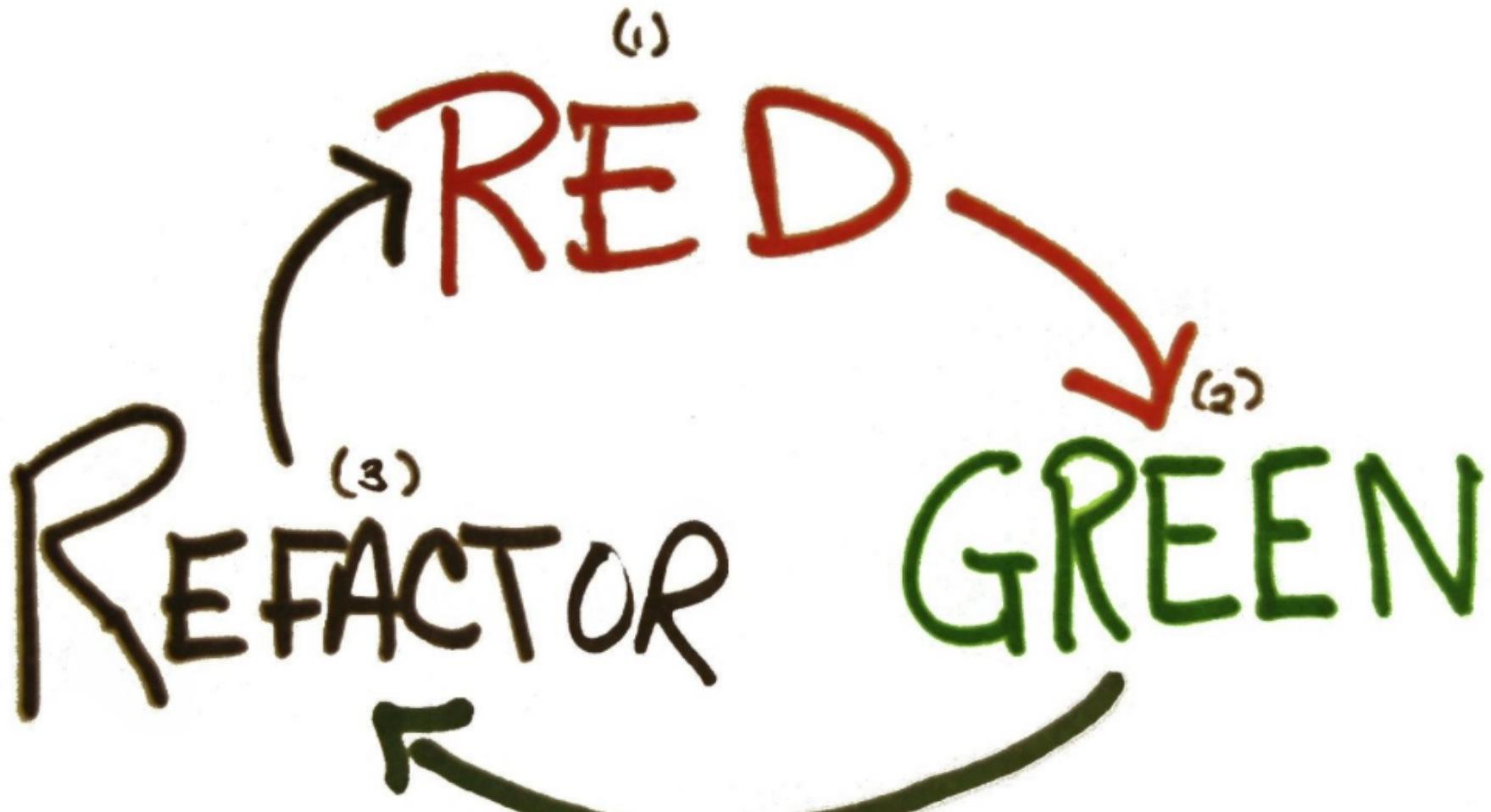
“A suite of tests is more important than a good design.”

- Uncle Bob Martin

A **comprehensive** suite of **tests**
can give us **NO FEAR**

TDD

Test-Driven Development



From <http://blog.cleancoder.com/uncle-bob/2014/12/17/TheCyclesOfTDD.html>

Why write tests first





Source: <https://tinyurl.com/y7rlue57>

“... [TDD] practitioners discovered
that **writing tests first made a**
significant improvement to the
design process.”

- Martin Fowler

<https://martinfowler.com/articles/mocksArentStubs.html>

Writing tests **first** forces us to
write a **comprehensive** suite of
tests which can give us **NO FEAR**

Some TDD history



TDD was developed by Kent Beck in the late 1990's as part of Extreme Programming.

In essence you follow three simple steps repeatedly:

1. Write a test for the next bit of functionality you want to add.
2. Write the functional code until the test passes.
3. Refactor both new and old code to make it well structured.

- Martin Fowler

<https://martinfowler.com/bliki/TestDrivenDevelopment.html>

Kent Beck

Source: <https://tinyurl.com/y7bb7knx>

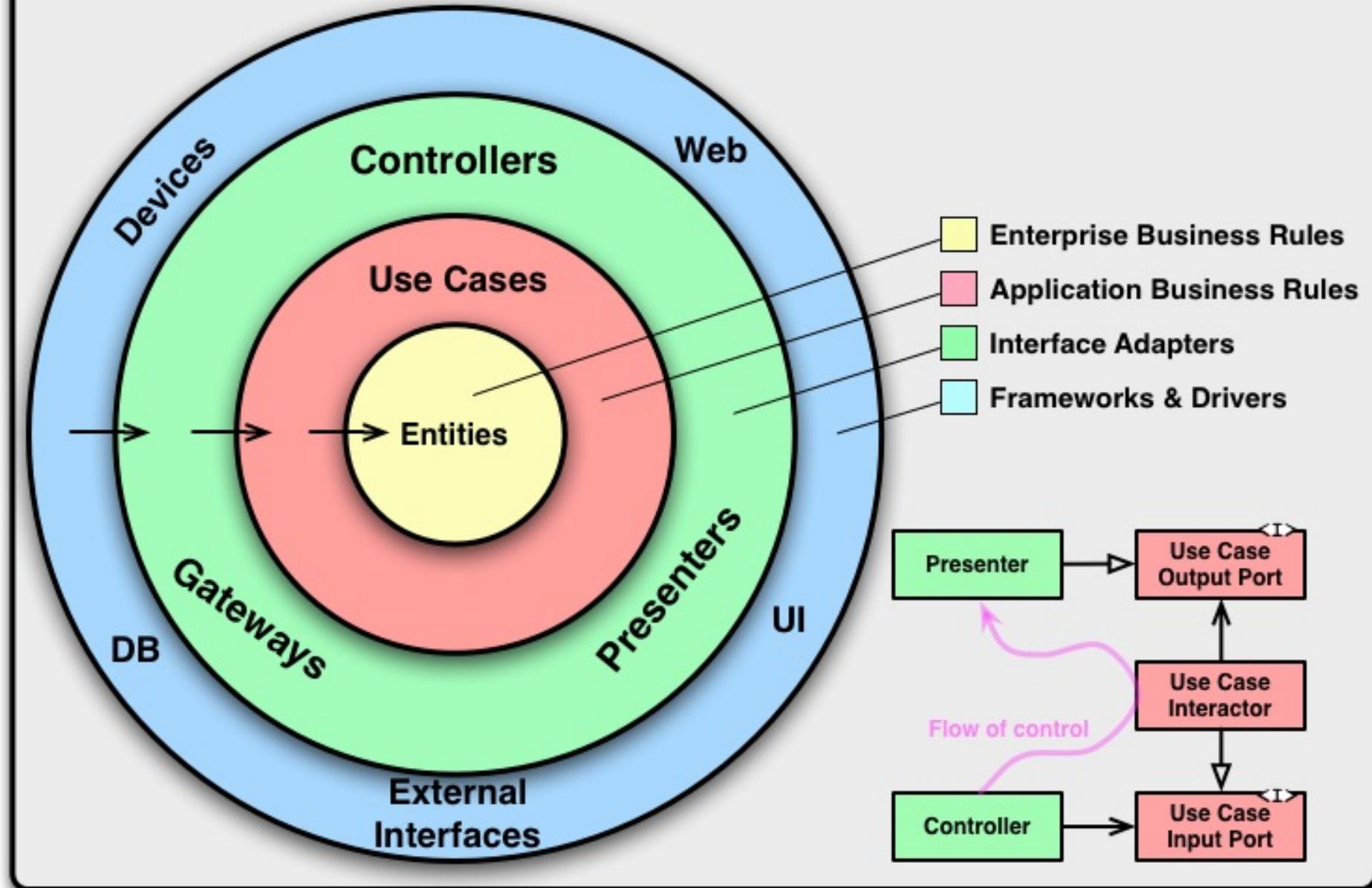
“Before you begin to code any software project,
you need to have some **architectural vision** in
place.

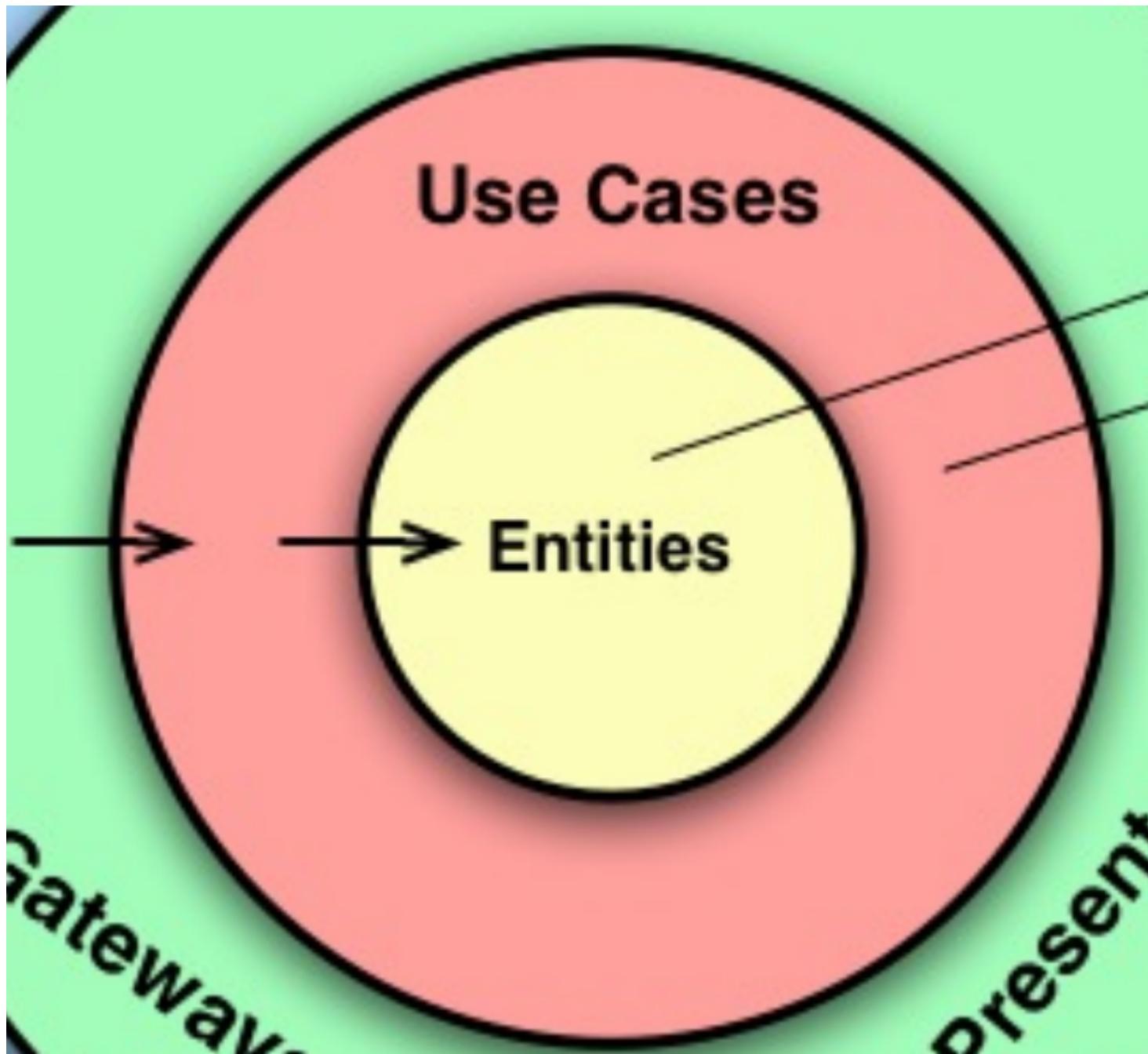
TDD will not, and can not, provide this vision.
That is not TDD’s role.”

- Uncle Bob Martin

clean Architecture

The Clean Architecture





Create Cat Use Case

Data:

<Name>

<Title>

Primary Course:

1. User issues “Create Cat” command with above data.
2. System validates all data.
3. System creates new Cat and determines cat-id
4. System delivers cat-id to user

Exception Course: Validation Error

1. System delivers error message

```
public class CreateCatUseCaseInteractor
{
    ...
    public CreateCatResponse execute(CreateCatRequest request)
    {
        // User issues “Create Cat” command with above data.

        // System validates all data.

        // System creates new Cat and determines cat - id

        // System delivers cat-id to user

        return response;
    }
}
```

Cat

name
title

grandioseName()

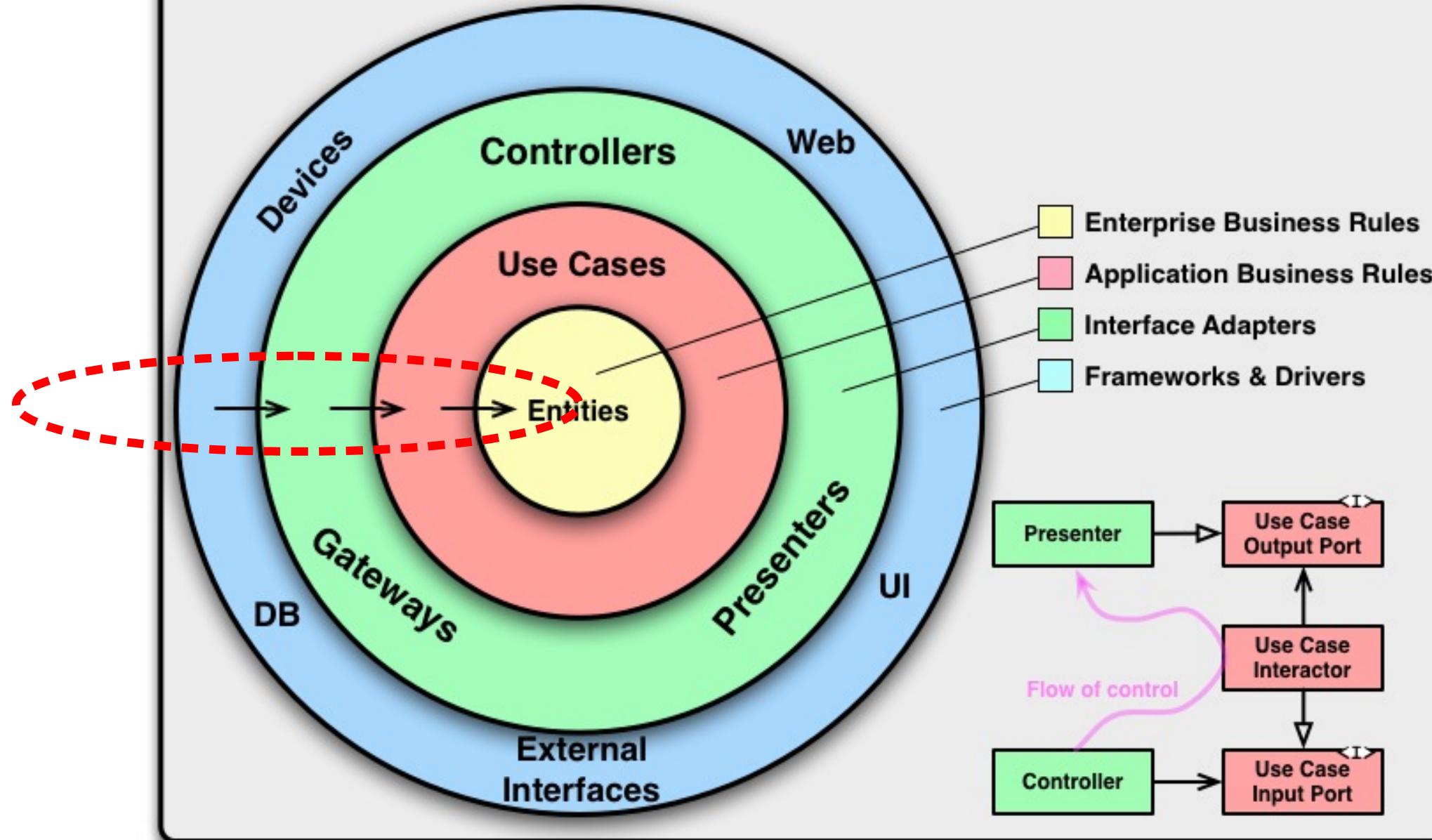
“Architecture is
about Intent, not
frameworks”

“Architecture should reveal operation.

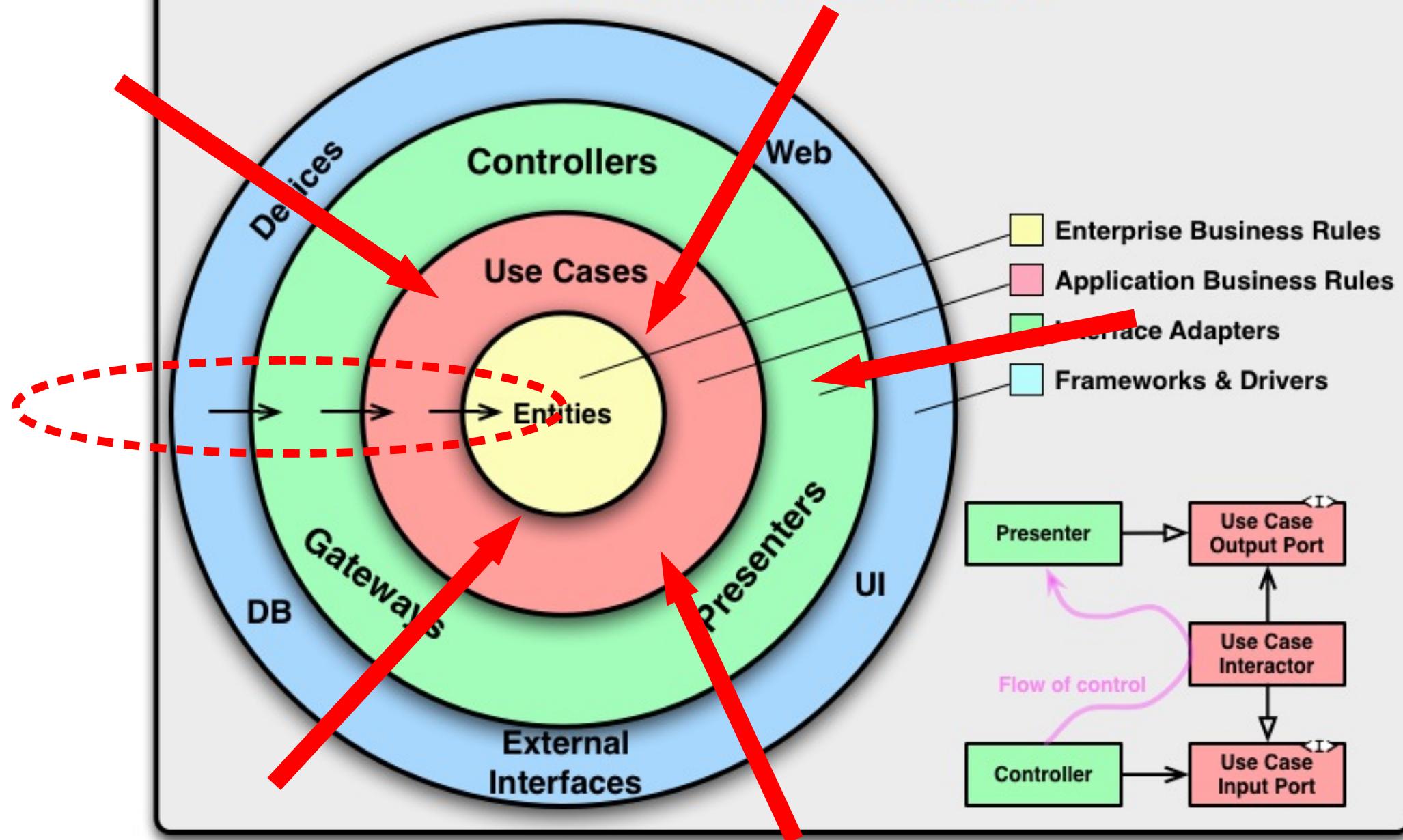
The architecture of the system should elevate the use-cases, the features, the required behaviors of the system, to first class entities that are visible landmarks for the developers.”

- Uncle Bob Martin

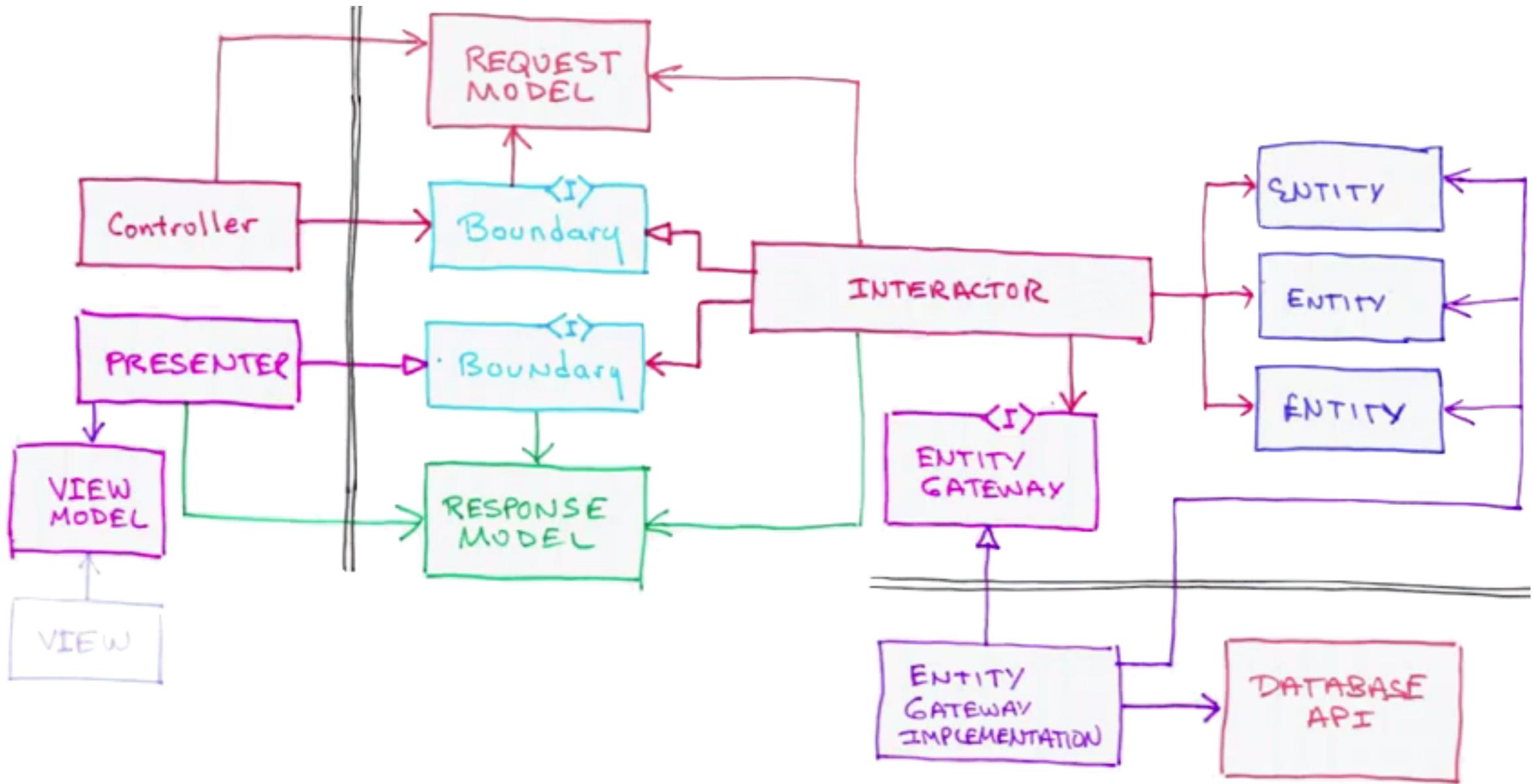
The Clean Architecture



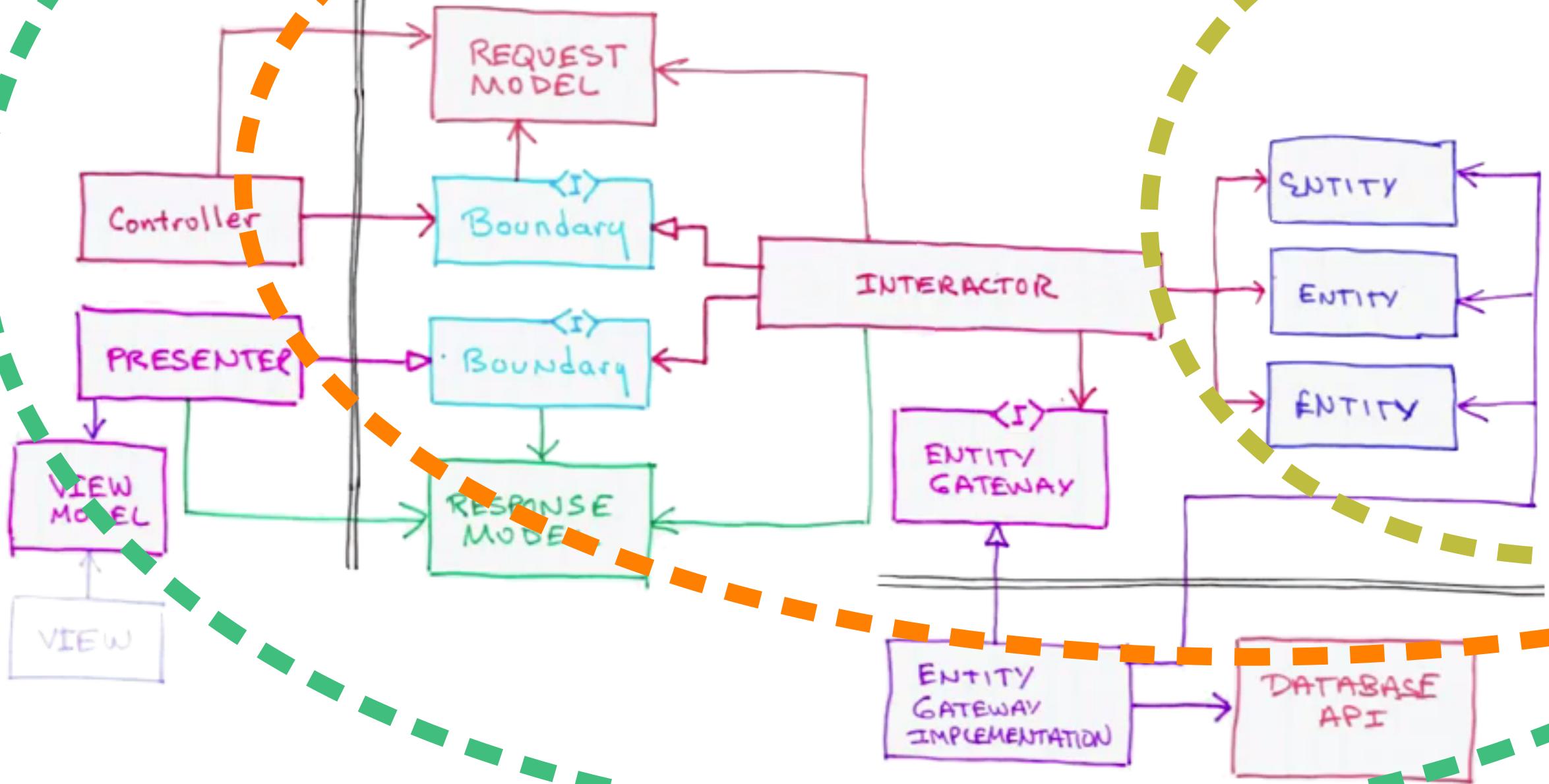
The Clean Architecture



Example

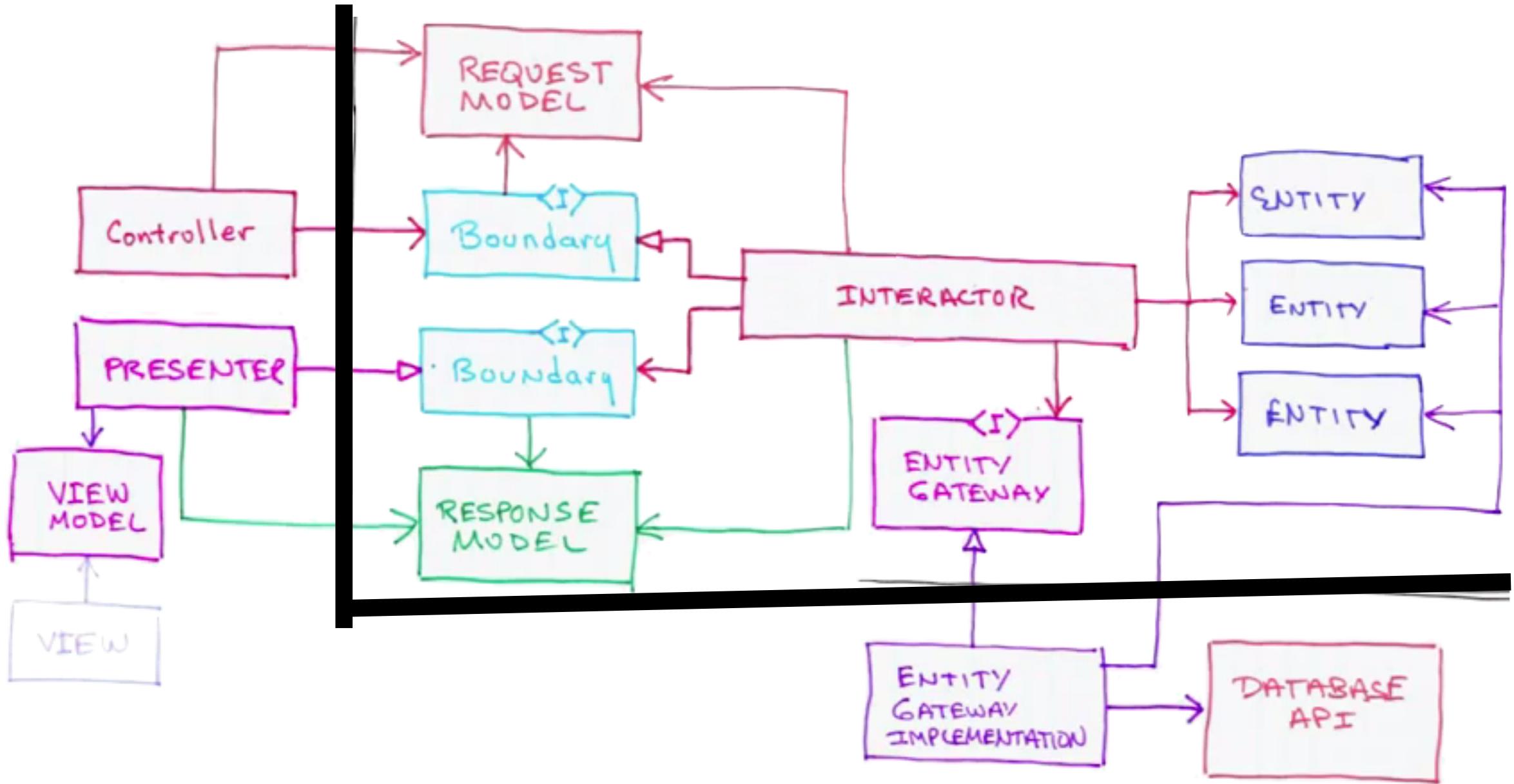


From Uncle Bob's slides of his talks on Clean Architecture

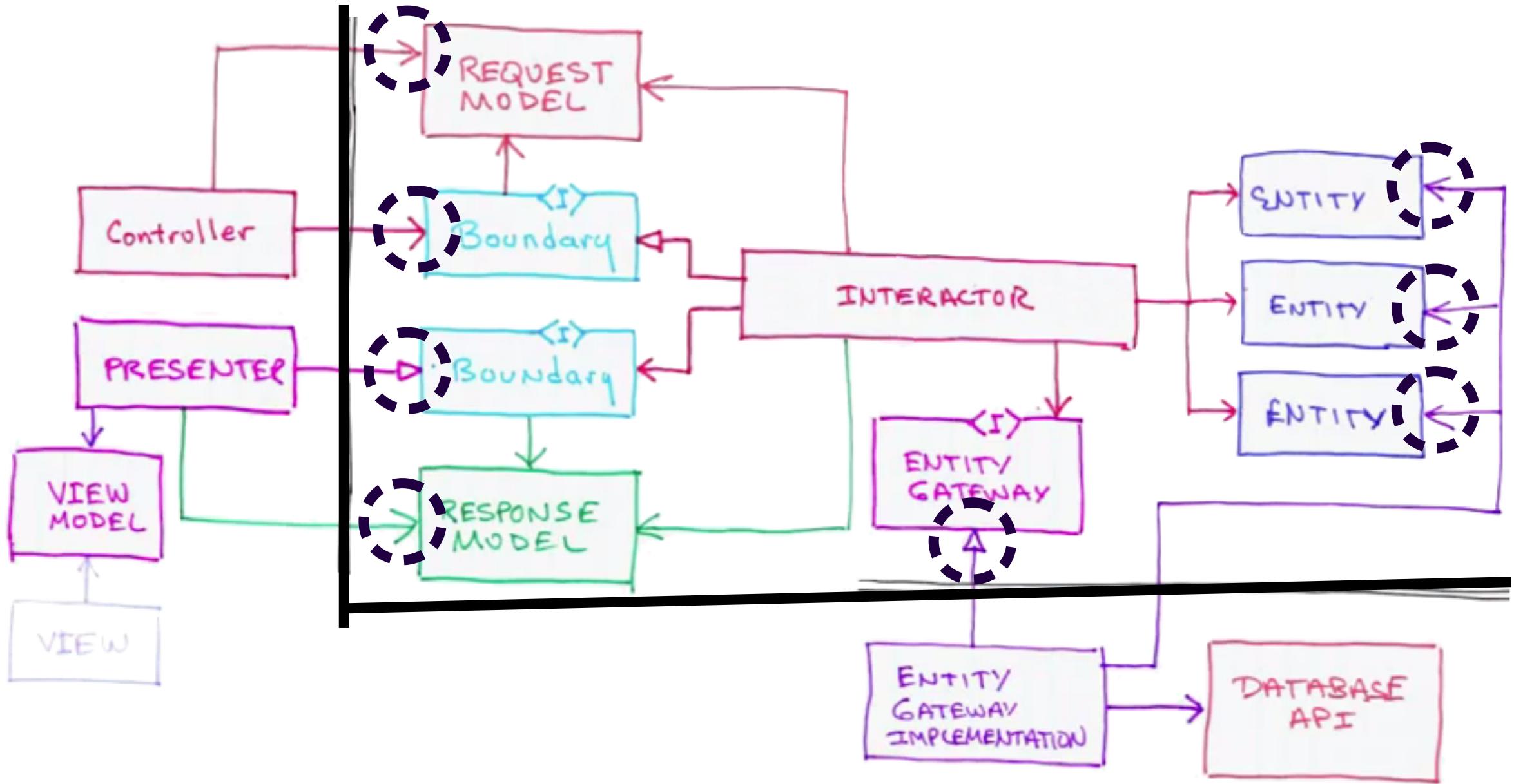


From Uncle Bob's slides of his talks on Clean Architecture

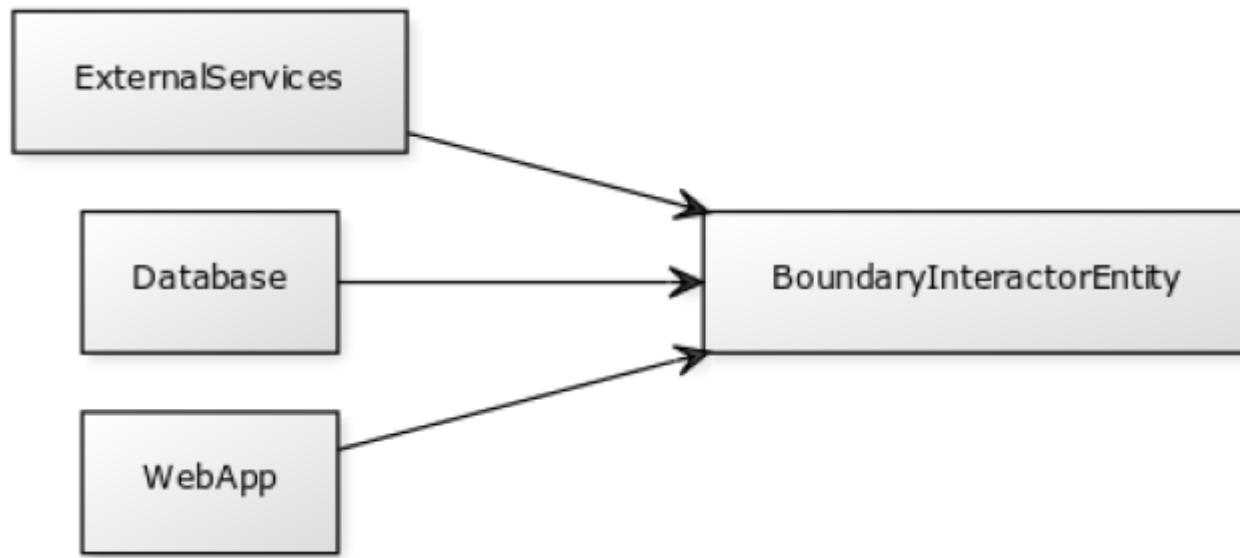
“Architecture is the art of drawing lines”
- Uncle Bob Martin



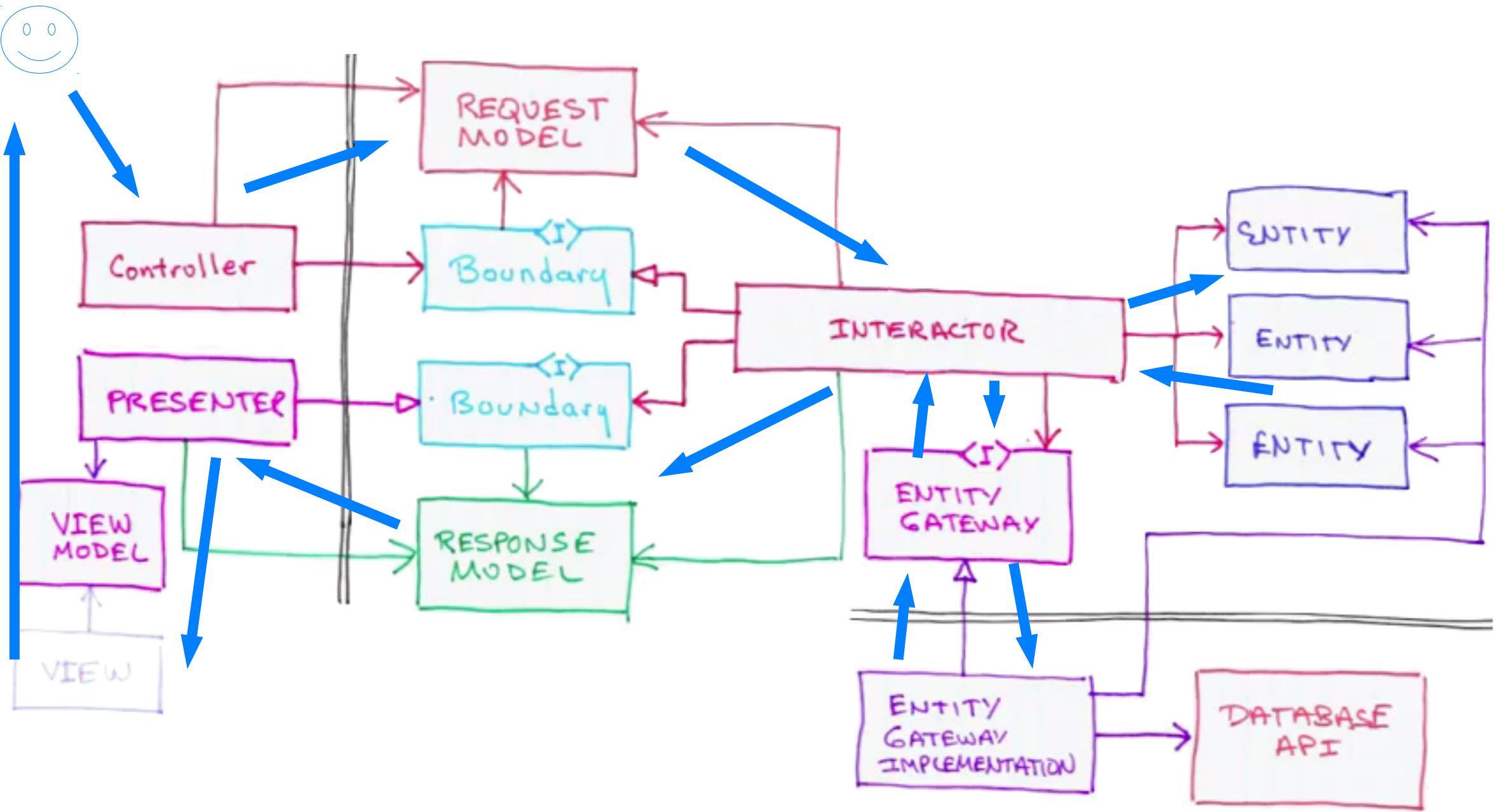
From Uncle Bob's slides of his talks on Clean Architecture



From Uncle Bob's slides of his talks on Clean Architecture



<http://craftsmanshipcounts.com/clean-architecture-compare-critique-java/>



From Uncle Bob's slides of his talks on Clean Architecture

```
public class CreateCatUseCaseInteractor
{
    CatCreatedPresenter presenter;

    ...

    public void execute(CreateCatRequest request)
    {
        ...

        CreateCatResponse response = ...;

        ...

        presenter.present(response);
    }
}
```

Goals

Write fast tests

Writing tests **first**, with the clean architecture as our **goal**, can give us a **comprehensive** suite of **fast**-running **tests** that can make us have **NO FEAR**

Make the lives of programmers easier

“Good architecture makes the system easy to understand, easy to develop, easy to maintain, and easy to deploy.

The ultimate goal is to minimize the lifetime cost of the system; and maximize programmer productivity.”

- Uncle Bob Martin

Some history

In the early 90s, we were in the verge of learning architecture

In the early 90s, we were in the verge of learning architecture

... then the web happened and we thought everything changed

In the early 90s, we were in the verge of learning architecture

... then the web happened and we thought everything changed

... then about 10 years later, we realized that the web is just a delivery mechanism

In the early 90s, we were in the verge of learning architecture

... then the web happened and we thought everything changed

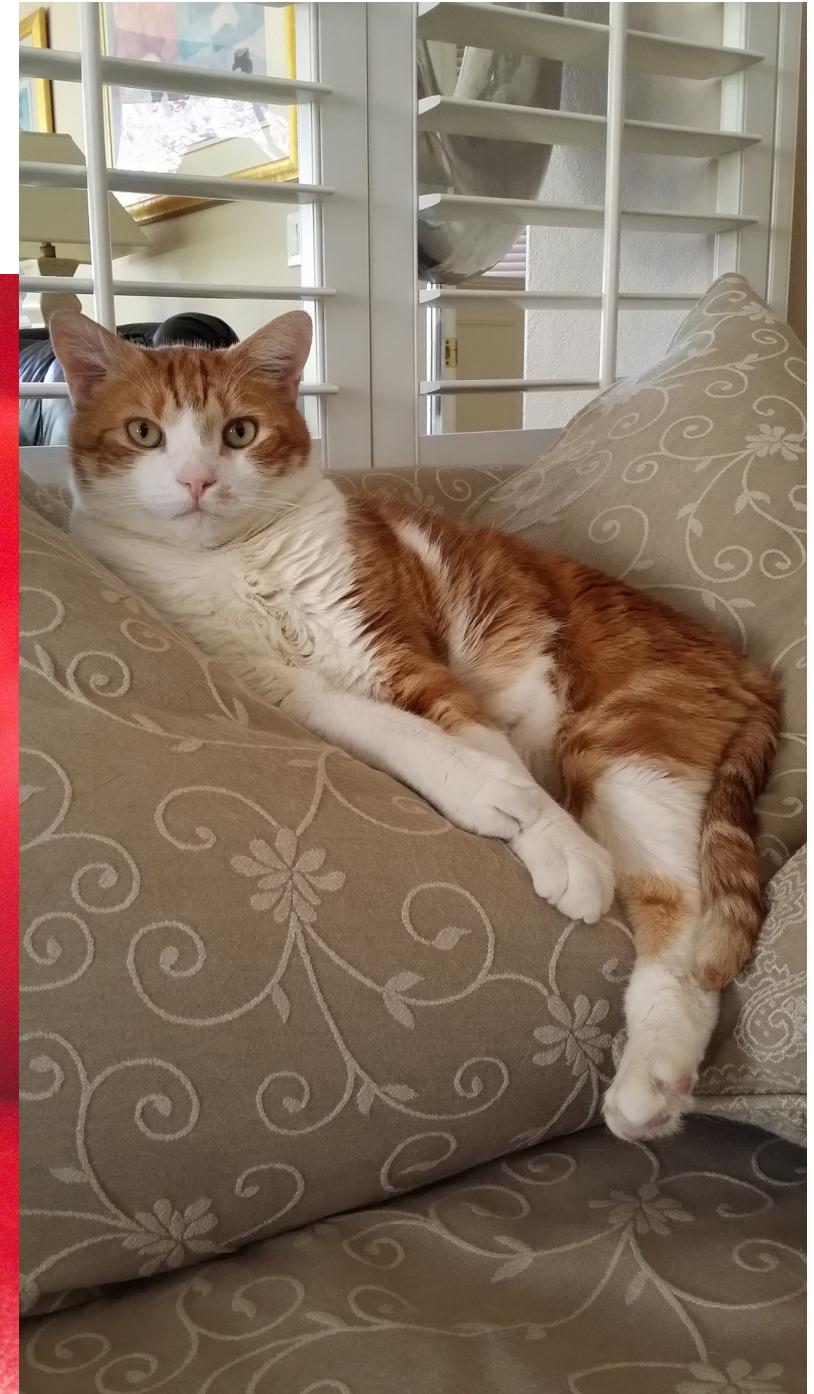
... then about 10 years later, we realized that the web is just a delivery mechanism

... then we started learning about architecture again

- Uncle Bob (paraphrased)

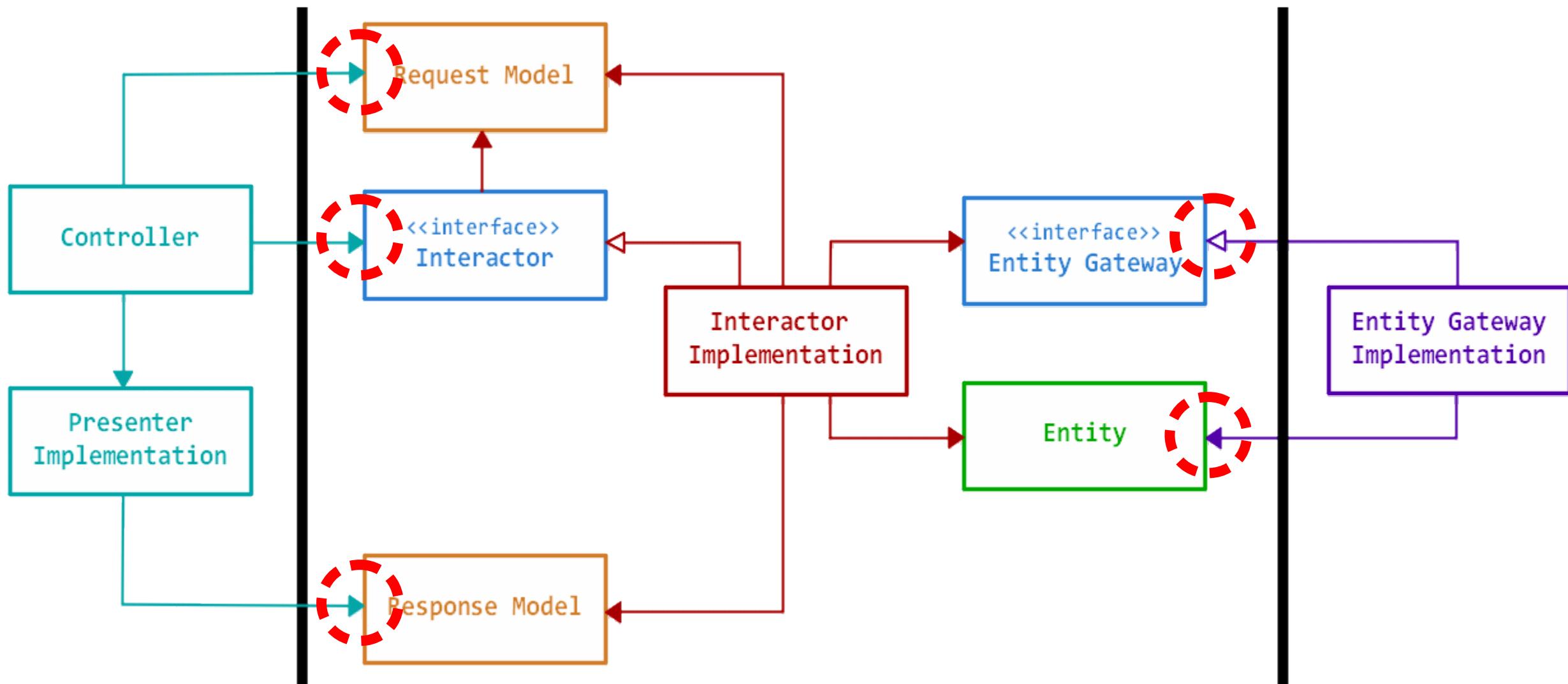
Demo

“In ancient times, cats were worshiped as gods. They have never forgotten this.”



“In ancient times, cats were worshiped as gods. They have never forgotten this.”





```
using Xunit;

namespace Interactors.Tests
{
    public class CreateCatInteractorTests
    {
    }
}
```

```
using Xunit;

namespace Interactors.Tests
{
    public class CreateCatInteractorTests
    {

        [Fact]
        public void test_that_the_tests_are_working()
        {
            Assert.Equal(4, 2 + 2);

        }
    }
}
```



```
namespace Interactors.Tests
{
    public class CreateCatInteractorTests
    {
        [Fact]
        public void ShouldReturnTheCorrectResponse()
        {
            var interactor = new CreateCatInteractor();
        }
    }
}
```



I wish I had an
interactor object

```
namespace Interactors
{
    public class CreateCatInteractor
    {
    }
}
```



```
namespace Interactors.Tests
{
    public class CreateCatInteractorTests
    {
        [Fact]
        public void ShouldReturnTheCorrectResponse()
        {
            var interactor = new CreateCatInteractor();
        }
    }
}
```



```
namespace Interactors.Tests
{
    public class CreateCatInteractorTests
    {
        [Fact]
        public void ShouldReturnTheCorrectResponse()
        {
            // arrange
            var interactor = new CreateCatInteractor();

            // act
            CreateCatResponse response = interactor.Execute();
        }
    }
}
```

I wish I had an Execute method in my interactor...

.. which returns a response object



```
namespace Interactors
{
    public class CreateCatReponse
    {
    }
}
```



```
namespace Interactors.Tests
{
    public class CreateCatInteractorTests
    {
        [Fact]
        public void ShouldReturnTheCorrectResponse()
        {
            // arrange
            var interactor = new CreateCatInteractor();

            // act
            CreateCatResponse response = interactor.Execute();
        }
    }
}
```



```
namespace Interactors
{
    public class CreateCatInteractor
    {
        public CreateCatReponse Execute()
        {
        }
    }
}
```



```
namespace Interactors.Tests
{
    public class CreateCatInteractorTests
    {
        [Fact]
        public void ShouldReturnTheCorrectResponse()
        {
            // arrange
            var interactor = new CreateCatInteractor();

            // act
            CreateCatResponse response = interactor.Execute();

        }
    }
}
```



Why TDD and Clean Architecture?

**They can save us from becoming
slaves of our creations.**

They can give us NO FEAR

FEAR Zone

NO FEAR Zone



**“There are no silver
bullets”**

“The only way to go fast is to go well”

- Uncle Bob Martin

Just in case you fear that the “gossip joke” (from the introduction slides) will leak from the joke world to the real world...

there already exists practices that can prevent that from happening:

Pair Programming or Mob Programming

...

I asked: “Jerry, do you do this with your own code too?”

Jerry scowled: “We work as a team around here, so there is no code I call my own.

Do you consider this code yours now?”

“Not anymore.” I said, meekly. “You’ve had a big influence on it.”

“We both have.” he said, “And that’s the way that Mr. C likes it. **He doesn’t want any single person owning code...**”

- from “The Craftsman #3: Clarity and Collaboration” of Uncle Bob Martin

Where to go from here...

Blog posts of Uncle Bob on Clean Architecture and TDD

Screaming Architecture

Clean Architecture

The Clean Architecture

A Little Architecture

The Cycles of TDD

TDD Harms Architecture

Etc.

Talks of Uncle Bob on Clean Architecture

Clean Architecture and Design

Architecture: The Lost Years

Etc.

Book

Clean Architecture: A Craftsman's Guide to Software Structure and Design

About me

Jeremiah M. Flaga

flaga.jeremiah@gmail.com