



CherryChain®

Ithaca

The Clean and Hexagonal Architectural Island

Luca Guadagnini



Luca Guadagnini



@lucaguada



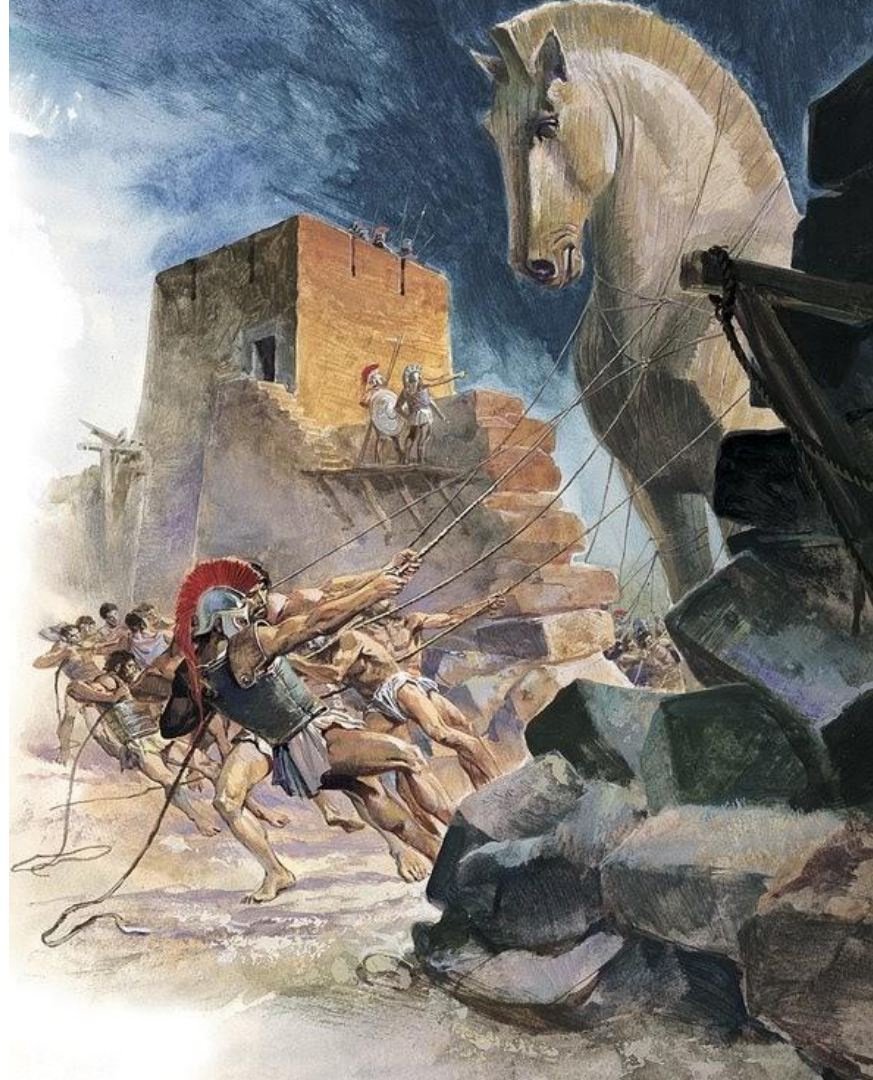
CherryChain[®]

<https://www.cherrychain.it>

OpenJDK



**Domain
Driven
Design**





Our Odyssey

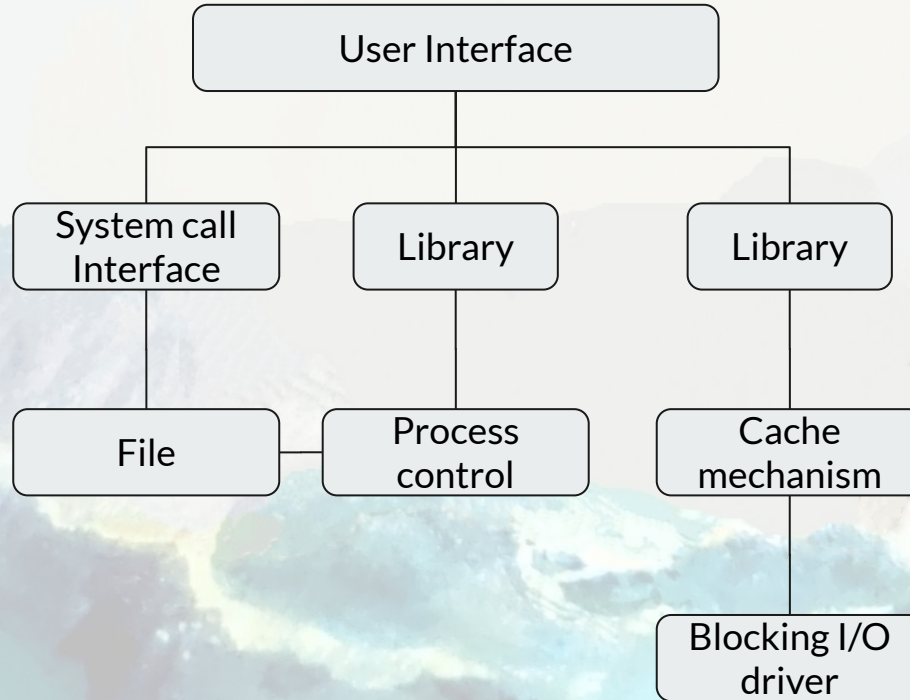
- ❖ Book 1: What is software architecture?
- ❖ Book 2: A common 3-layered approach
- ❖ Book 3: A clean and hexagonal solution
- ❖ Conclusions: Ithaca



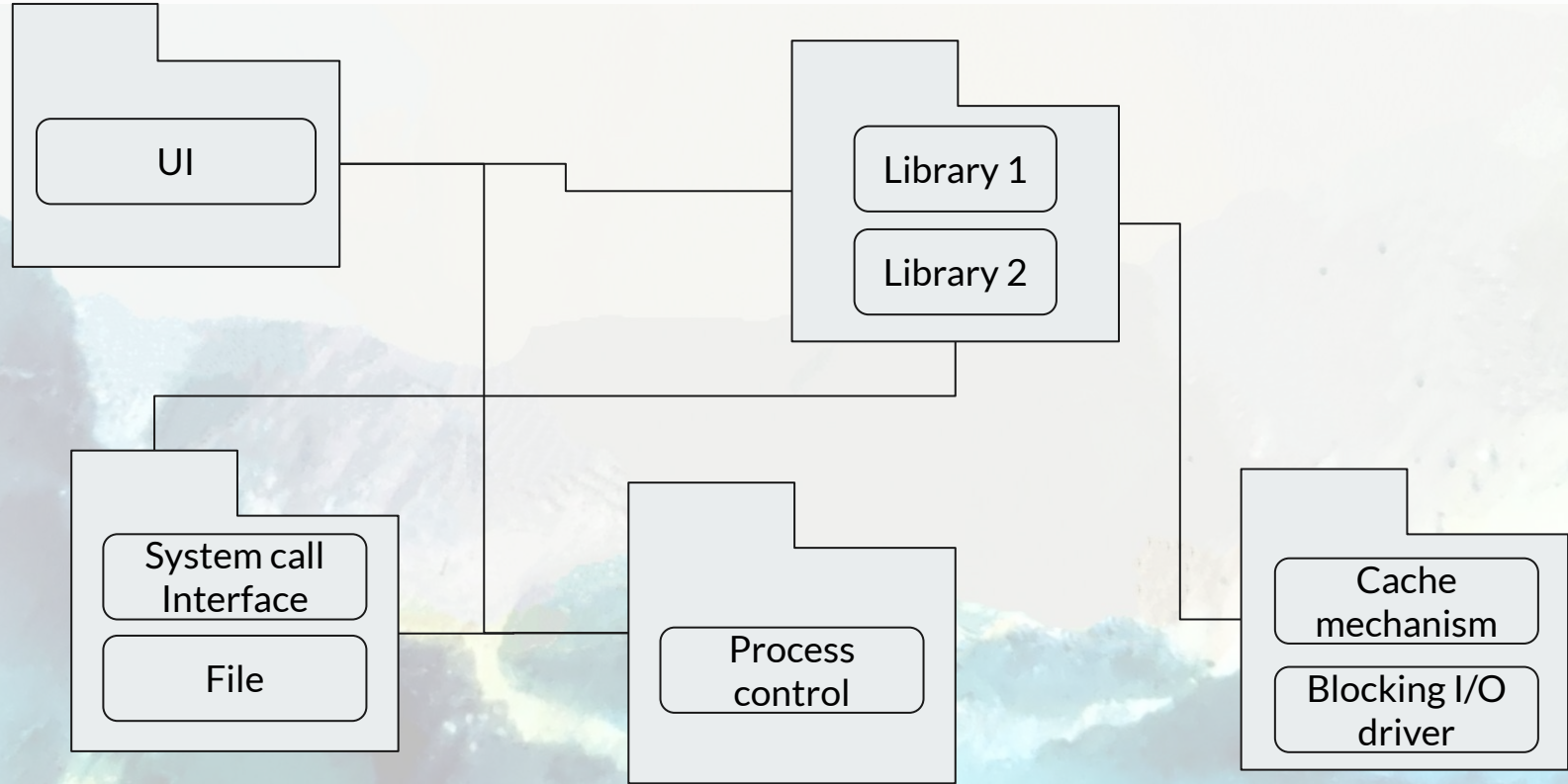
**What are we talking about, when we are talking about
Software Architecture?**

oh com'on!
building
software is
easy!


Software elements for reasoning



Software elements for reasoning



The eye of the beholder stakeholder



Hi little architect, I'm your stakeholder, is your software good enough?



Nobody knows



Developers

What can I do to make my architecture nice?

Quality attributes:

- Availability
- Deployability
- Energy Efficiency
- Integrability
- Modifiability
- Performance
- Safety
- Security
- Testability
- Usability

What can I do to make my architecture nice?

Quality attributes:

- Availability
- Deployability
- Energy Efficiency
- Integrability
- Modifiability
- Performance
- Safety
- Security
- Testability
- Usability



**We don't
need to
cover
them all**


Process guidelines:

- An architecture team and a CTO bound to developers
- Focus on a well-specified queue of QA's
- Docs!
- Evaluated by QA
- From a walking-skeleton with no integrations to a incrementally growing system

What can I do to make my architecture nice?

Quality attributes:

- Availability
- Deployability
- Energy Efficiency
- Integrability
- Modifiability
- Performance
- Safety
- Security
- Testability
- Usability



We don't
need to
cover
them all

Structural guidelines:

- Functional modularization
- QA's obtained with well-know architectural patterns
- Platform or tool independent
- Write and read sides segregation
- Design patterns are your friend (when you know the problem you want to solve)



oh com'on!
my
architecture
is fine!

A common architectural pattern: the 3-layered architecture

No really, how can I start?

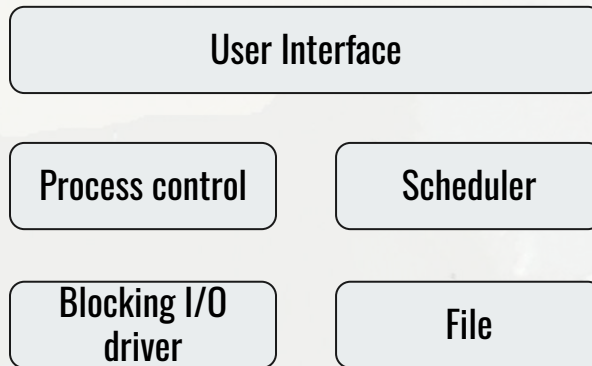
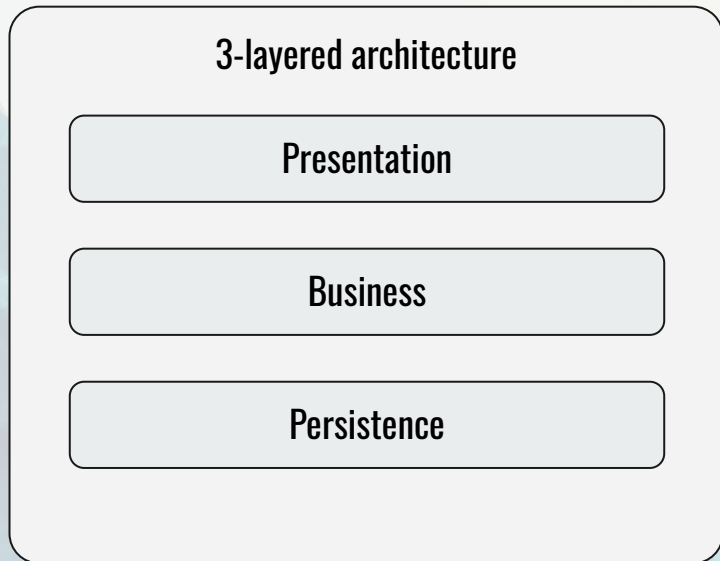
3-layered architecture

Presentation

Business

Persistence

No really, how can I start?



Request

Good enough?

Quality attributes:

- Deployability
- Modifiability
- Testability

n-layered architecture

Presentation

Web

Business

Persistence

Database

Good enough?

Quality attributes:

- ~~Deployability~~
- ~~Modifiability~~
- ~~Testability~~

n-layered architecture

Web Presentation

Mobile Presentation

Web HTTP/1.1

Business

Business

Web HTTP/2.0

Persistence

Database

Simplest solution: The Ram-Runaway Pattern



What really matters?

n^m -layered architecture

Presentation

Business

Persistence



oh com'on!
it's not done
yet?

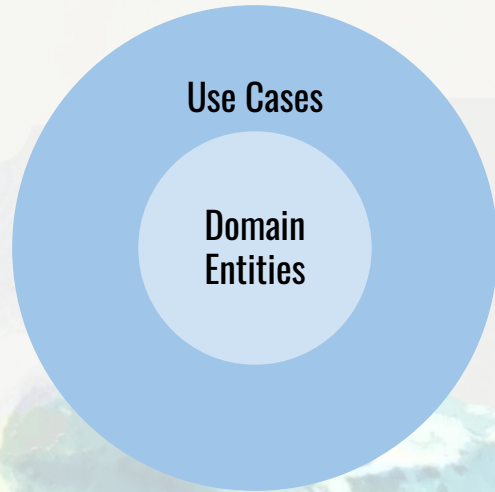
How to be clean and hexagonal with an architecture

Let's start to be clean: Domain Entities

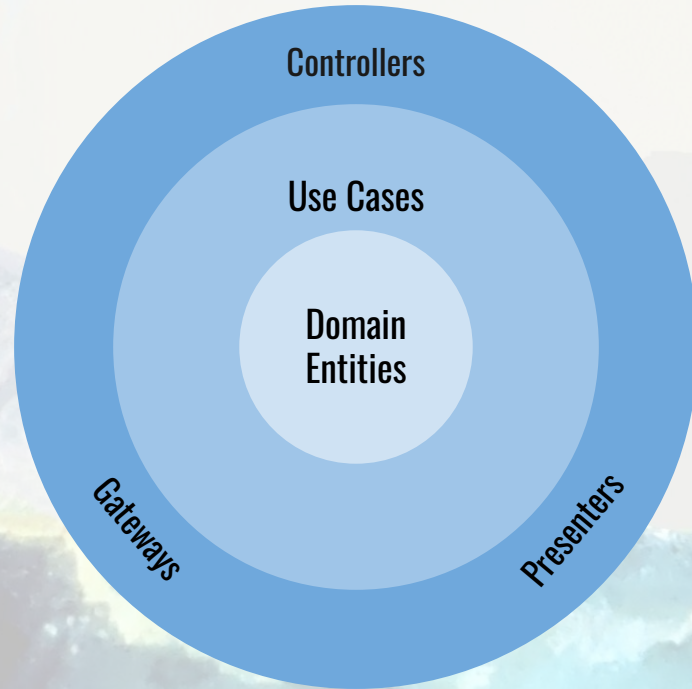


Domain
Entities

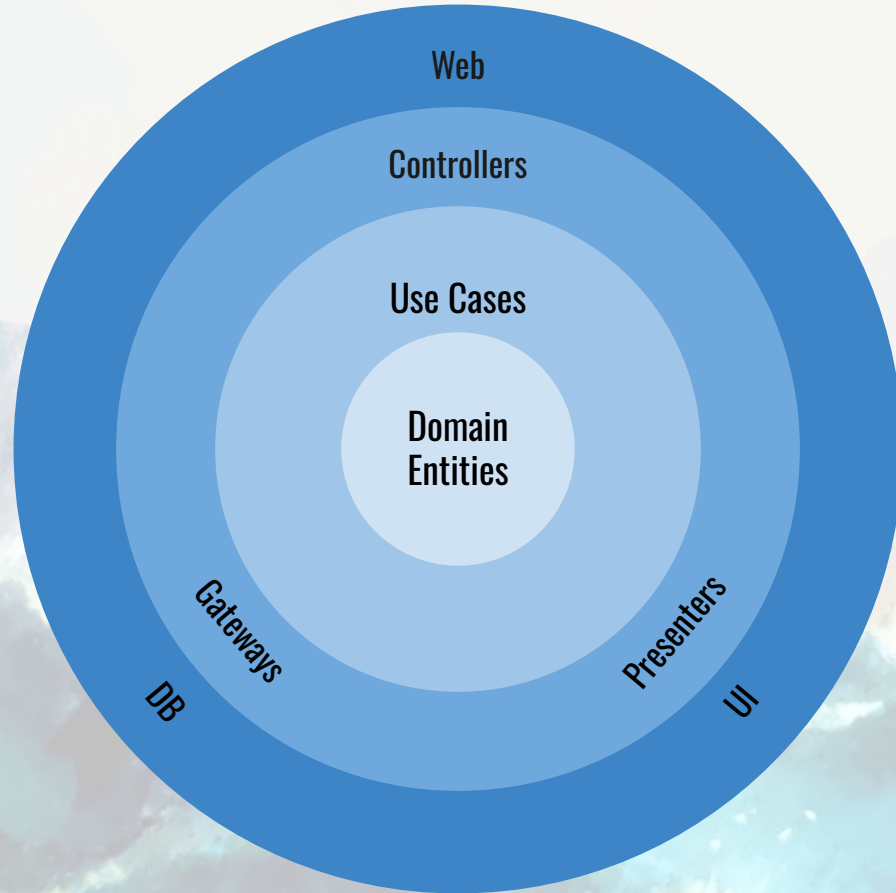
Let's start to be clean: Use Cases



Let's start to be clean: Services (i.e. Controllers, Gateways, etc...)



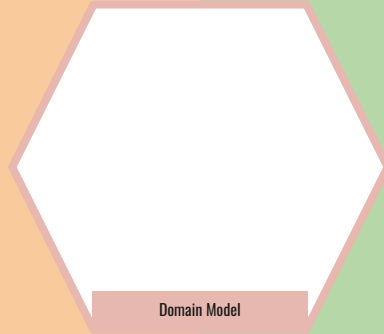
Let's start to be clean: External Services (i.e. Web, DB, UI, etc...)



Keep going hexagonal: Domain Model

User Interface

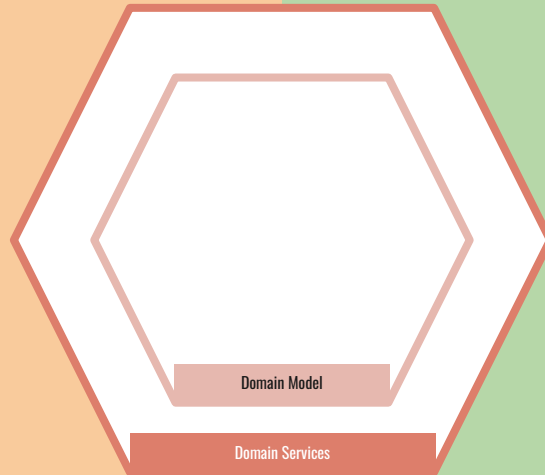
Infrastructure



Keep going hexagonal: Domain Services

User Interface

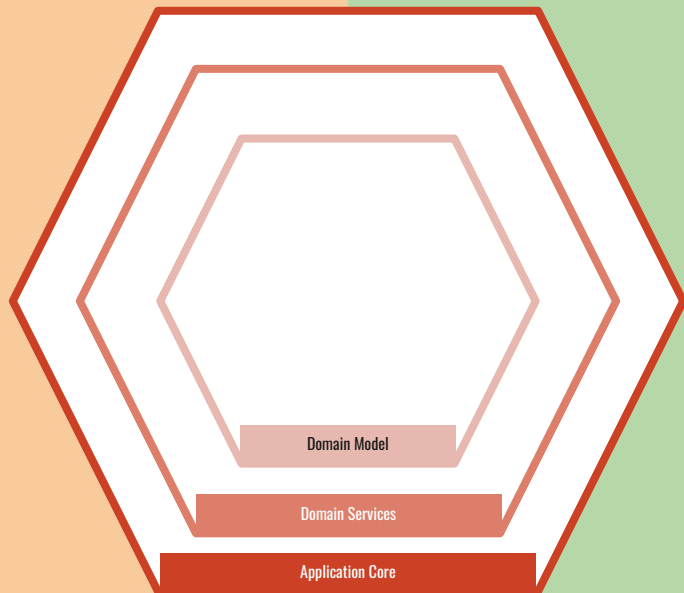
Infrastructure



Keep going hexagonal: Application Core

User Interface

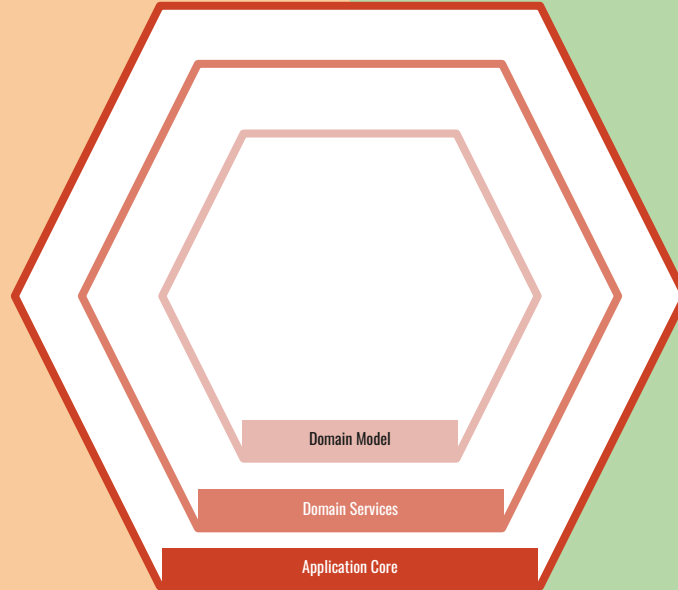
Infrastructure



Keep going hexagonal: Requests from outside?

User Interface

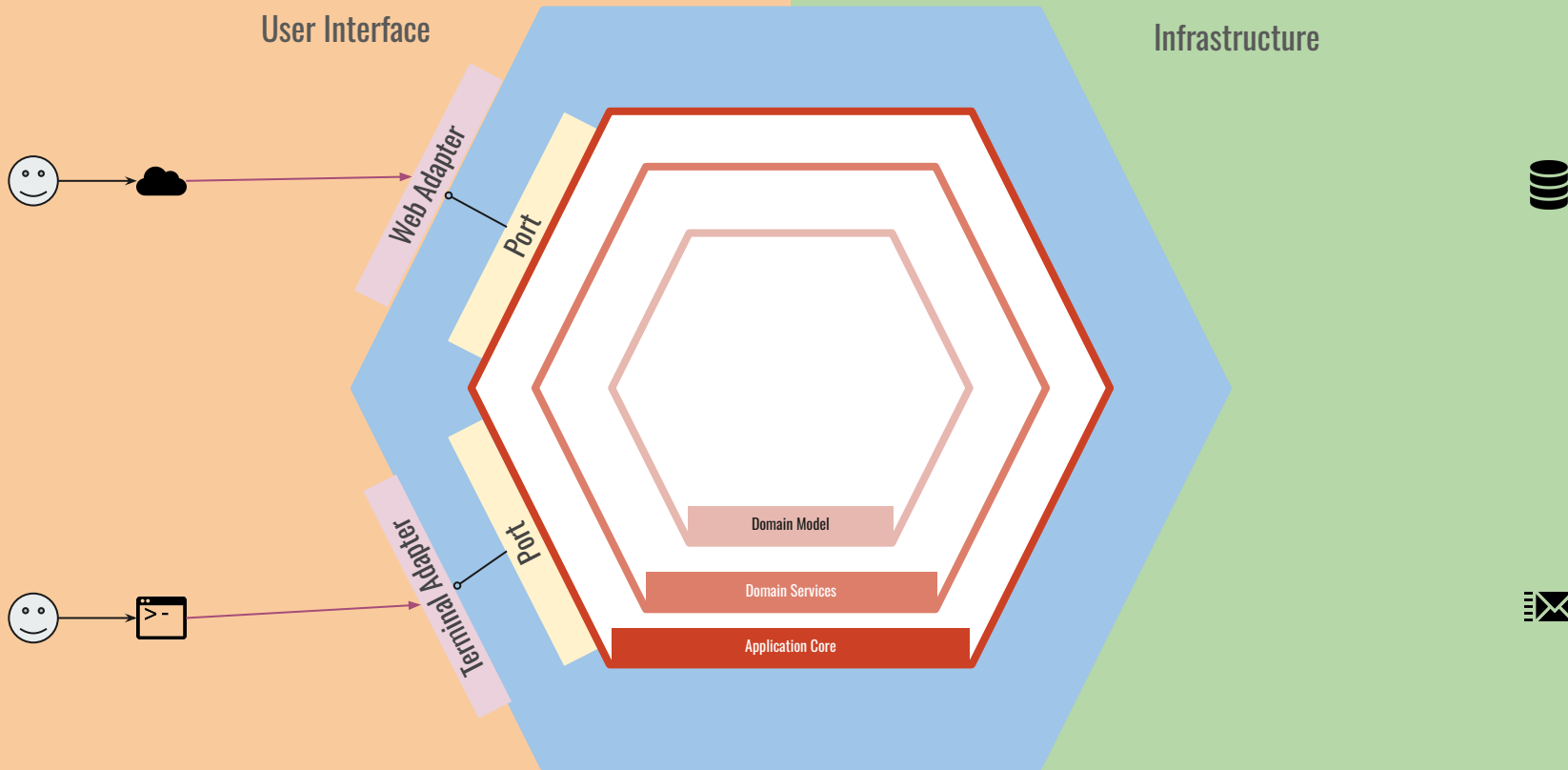
Infrastructure



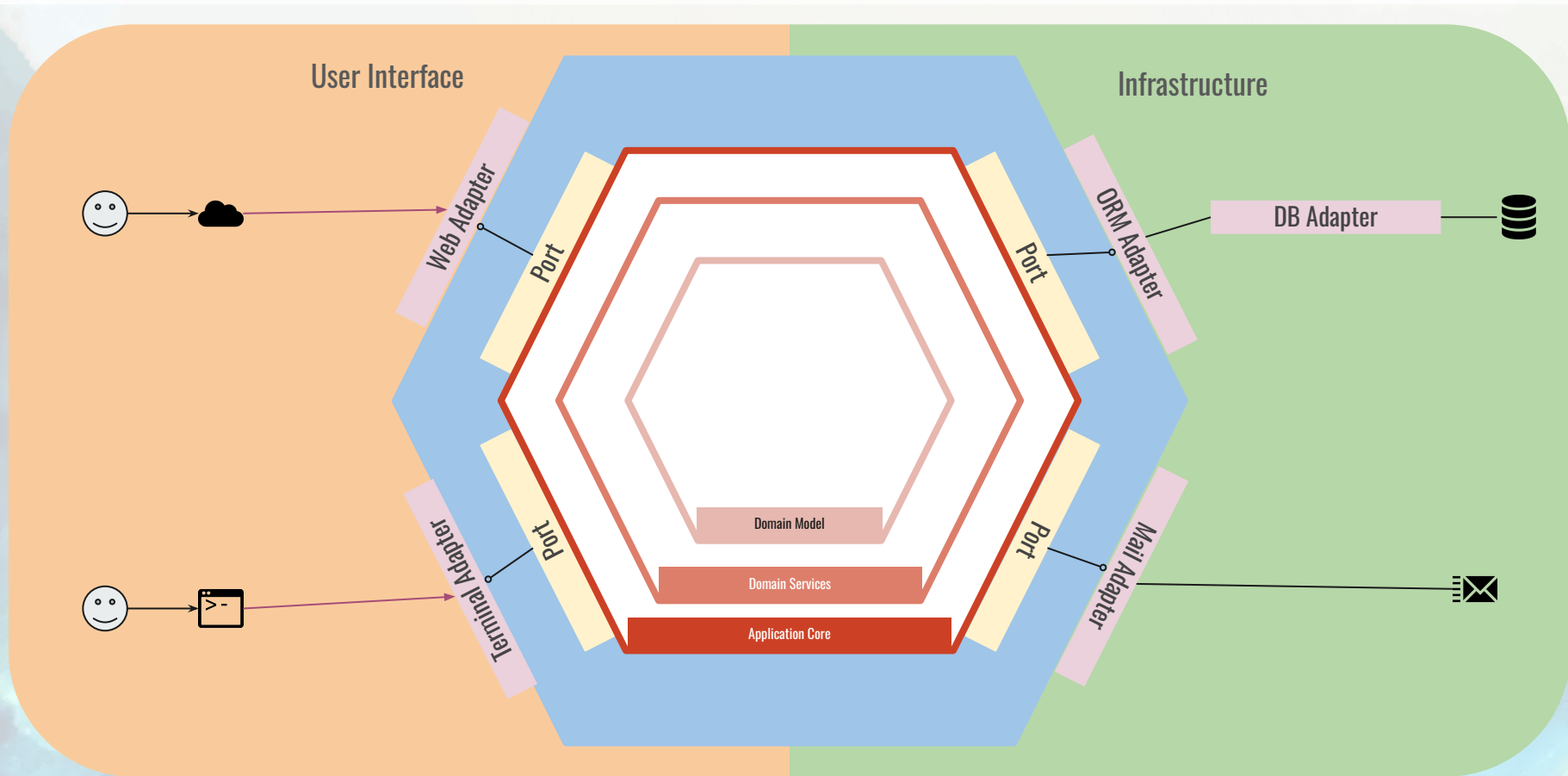
Keep going hexagonal: Ingress Ports and Adapters

User Interface

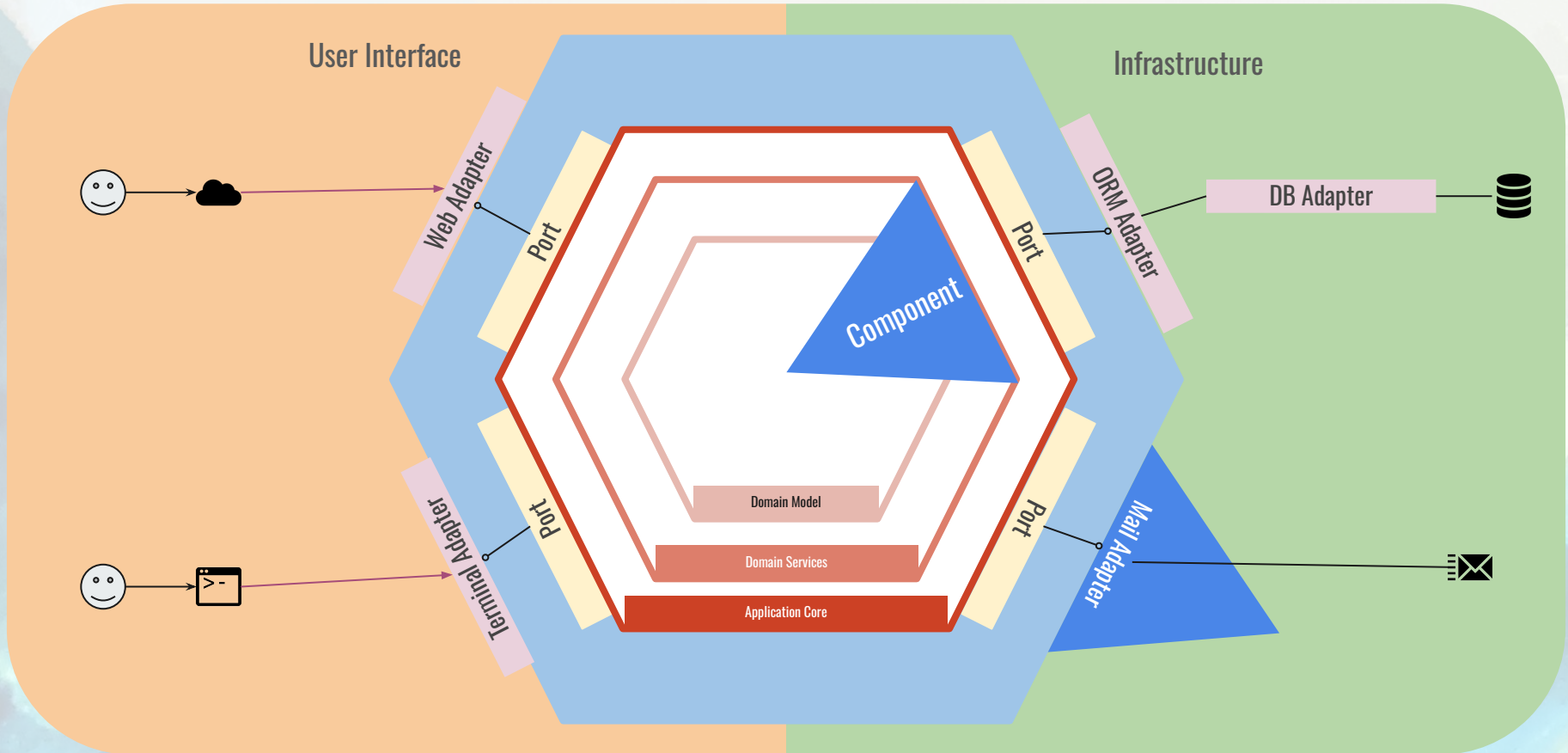
Infrastructure



Keep going hexagonal: Egress Ports and Adapters



Keep going hexagonal: Slice and dice!





Conclusions

What we learnt:

- Software architecture is not trivial
- Process and structural steps are essential
- Team communication is part of architectural tactics
- Business domain always rules
- Component isolations for the better
- Greek mythology is awesome



Conclusions

You can choose how to end the Odysseus you got in yourself:

- in homeric Odyssey the hero died by age
- in the tragedy Telegonia the hero is killed by his son
- in Odyssey written by Nikos Kazantzakis the hero started a new journey of dreams and discoveries
- in Dante's Inferno the hero died because he wanted to know too much
- write your own story



Please Nobody, let
me know if
everything works!

References:

- **Software Architecture in Practice**
by Len Bass, Paul Clements, Rick Kazman
- **Fundamentals of Software Architecture**
by Mark Richards, Neal Ford
- **Clean Architecture**
by Robert C. Martin
- **Get Your Hands Dirty on Clean Architecture**
by Tom Hombergs
- **Designing Hexagonal Architecture with Java**
by Davi Vieira



Thanks
for
listening!