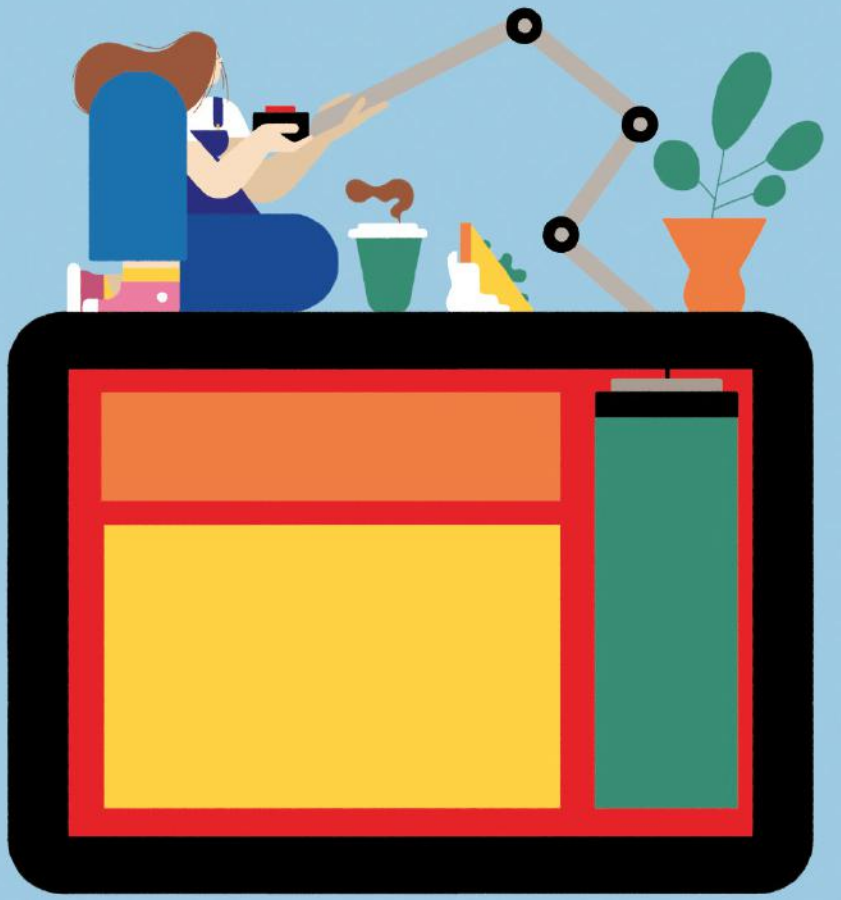
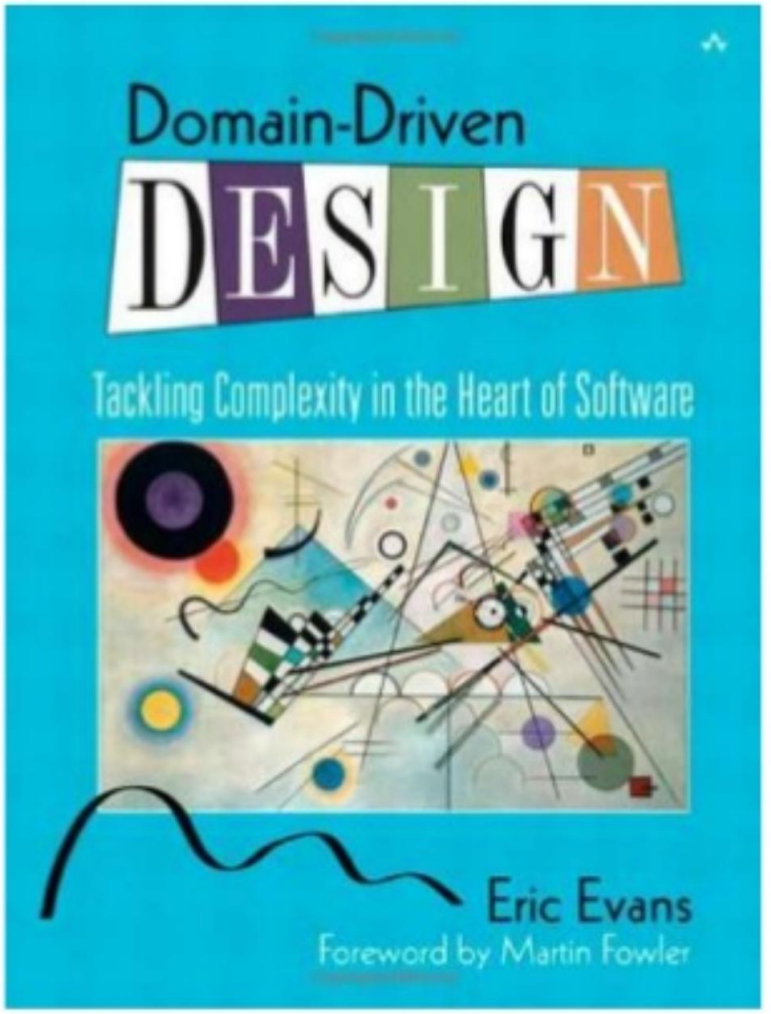
bit Computer Training
by Netmind

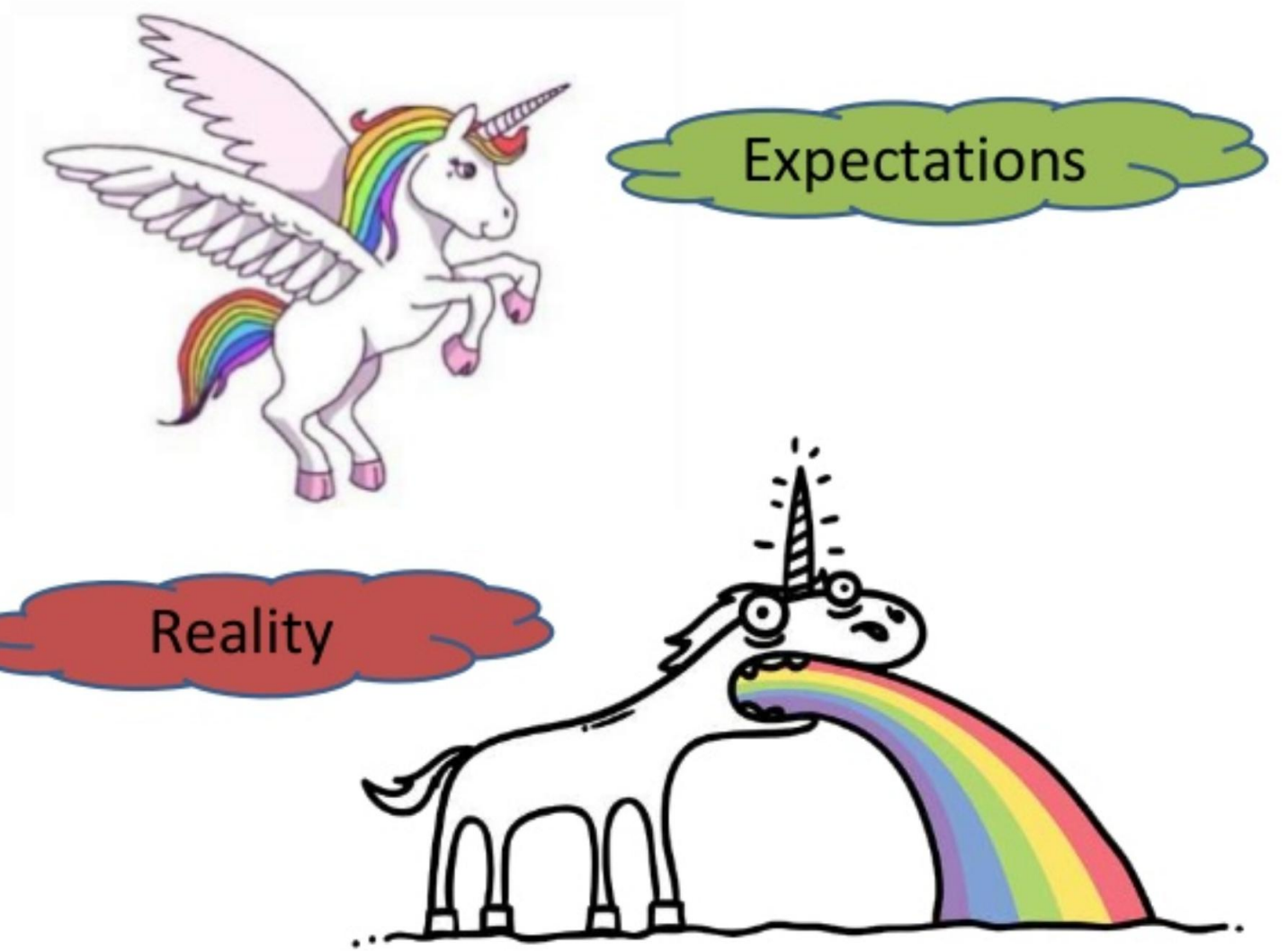
Web Development
Hexagonal Architecture



© 2018 BIT SAU, Barcelona

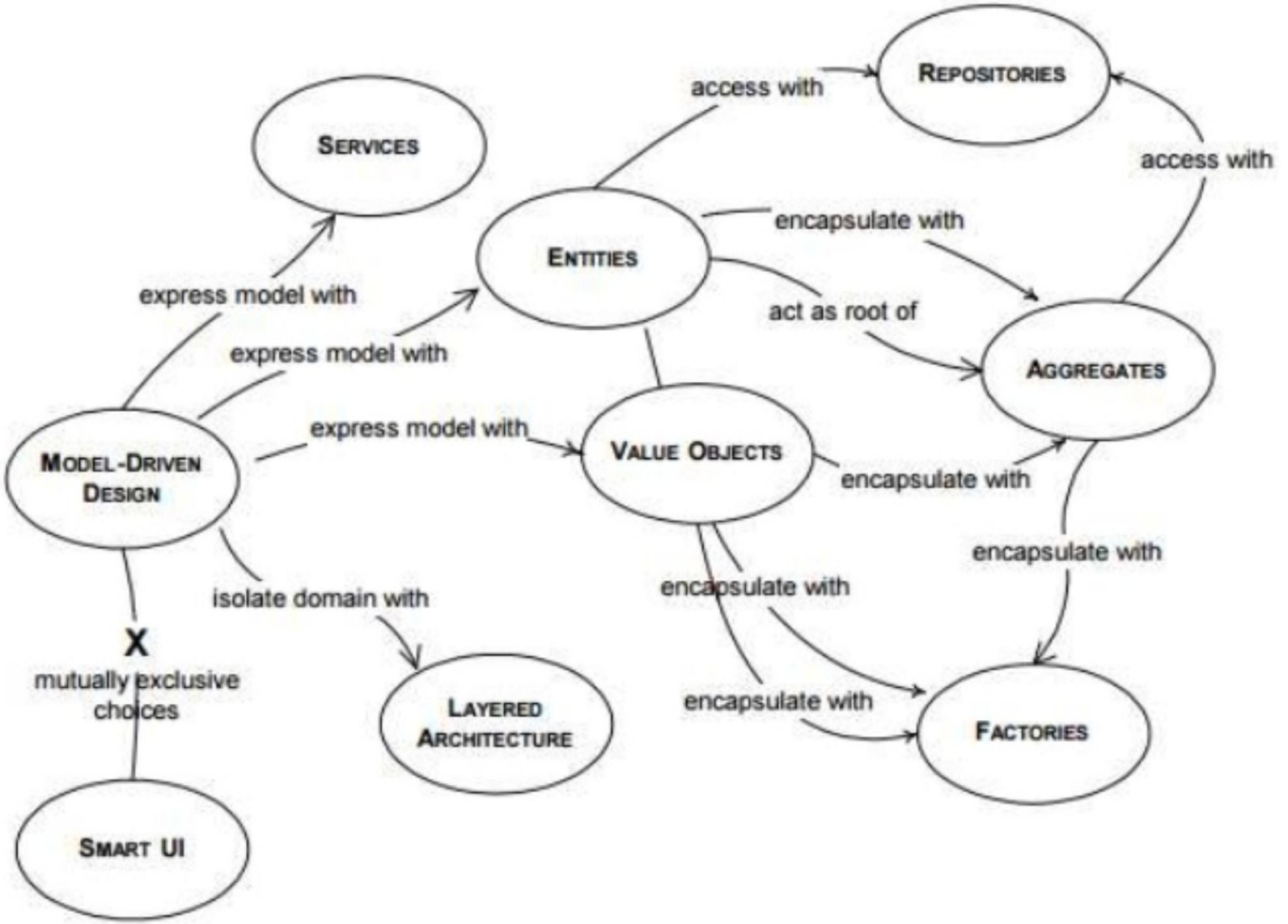
Domain Driven Design





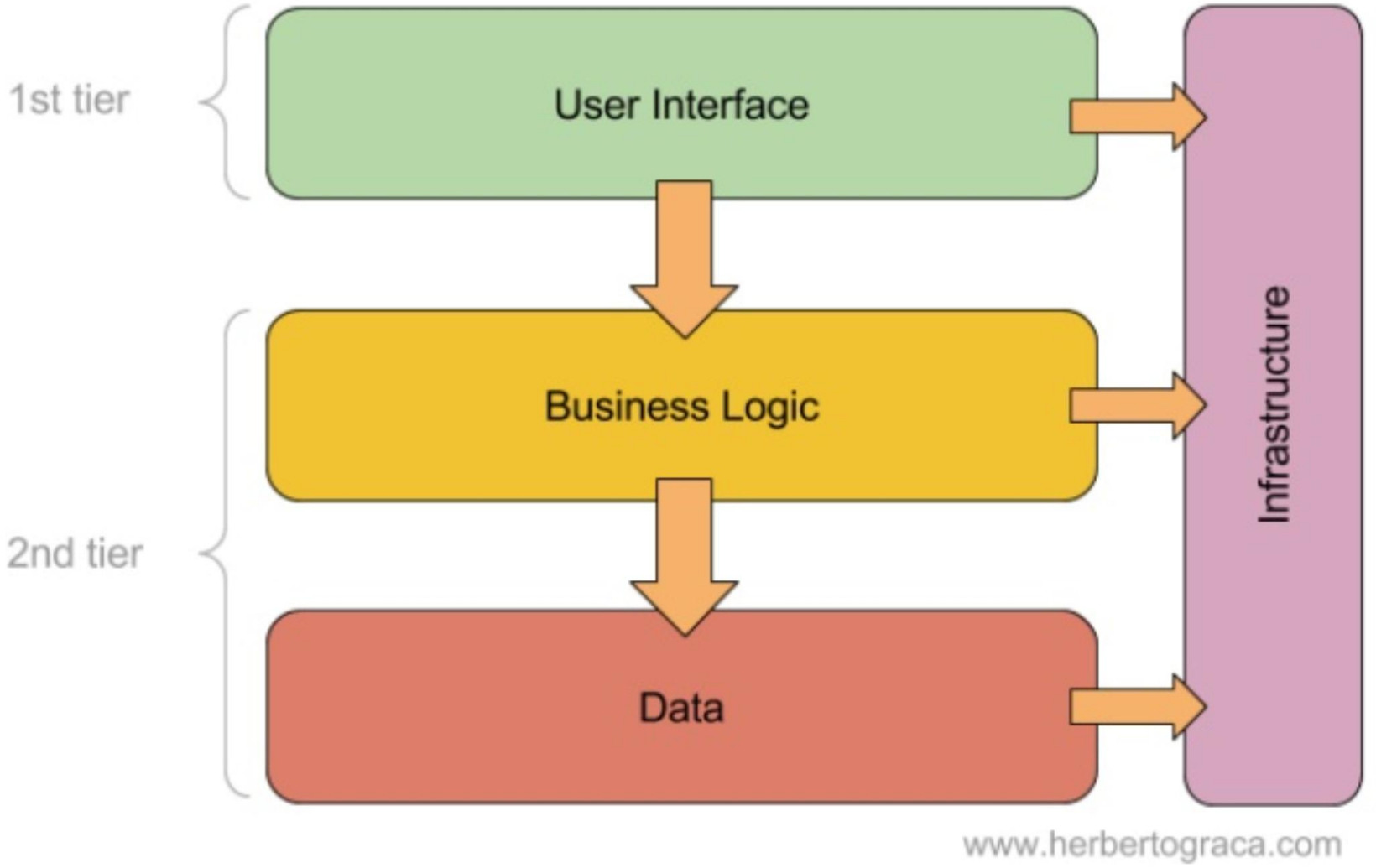
Everything is simple!

Clip slide

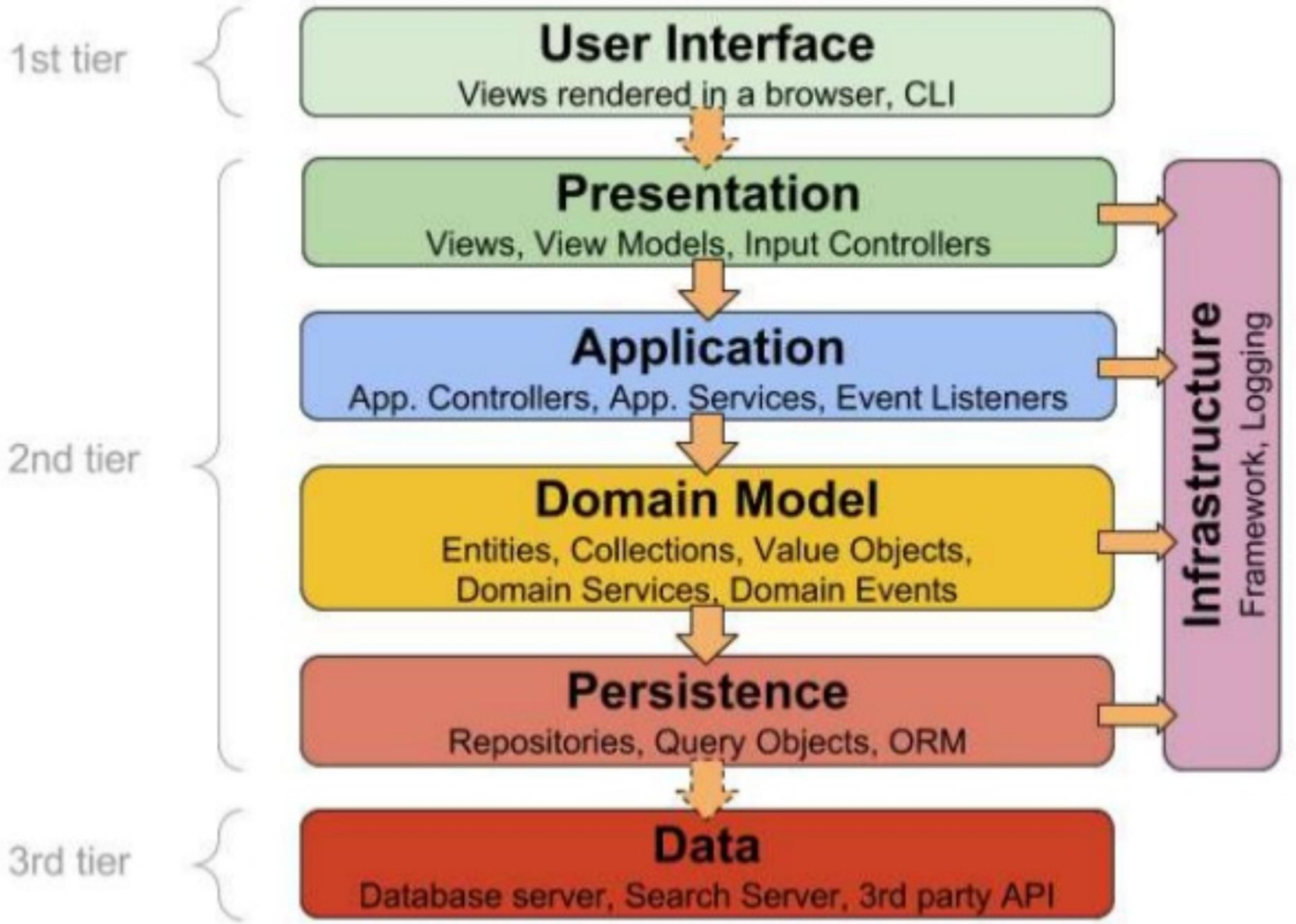


Layered architecture for kids

Clip slide



More serious view

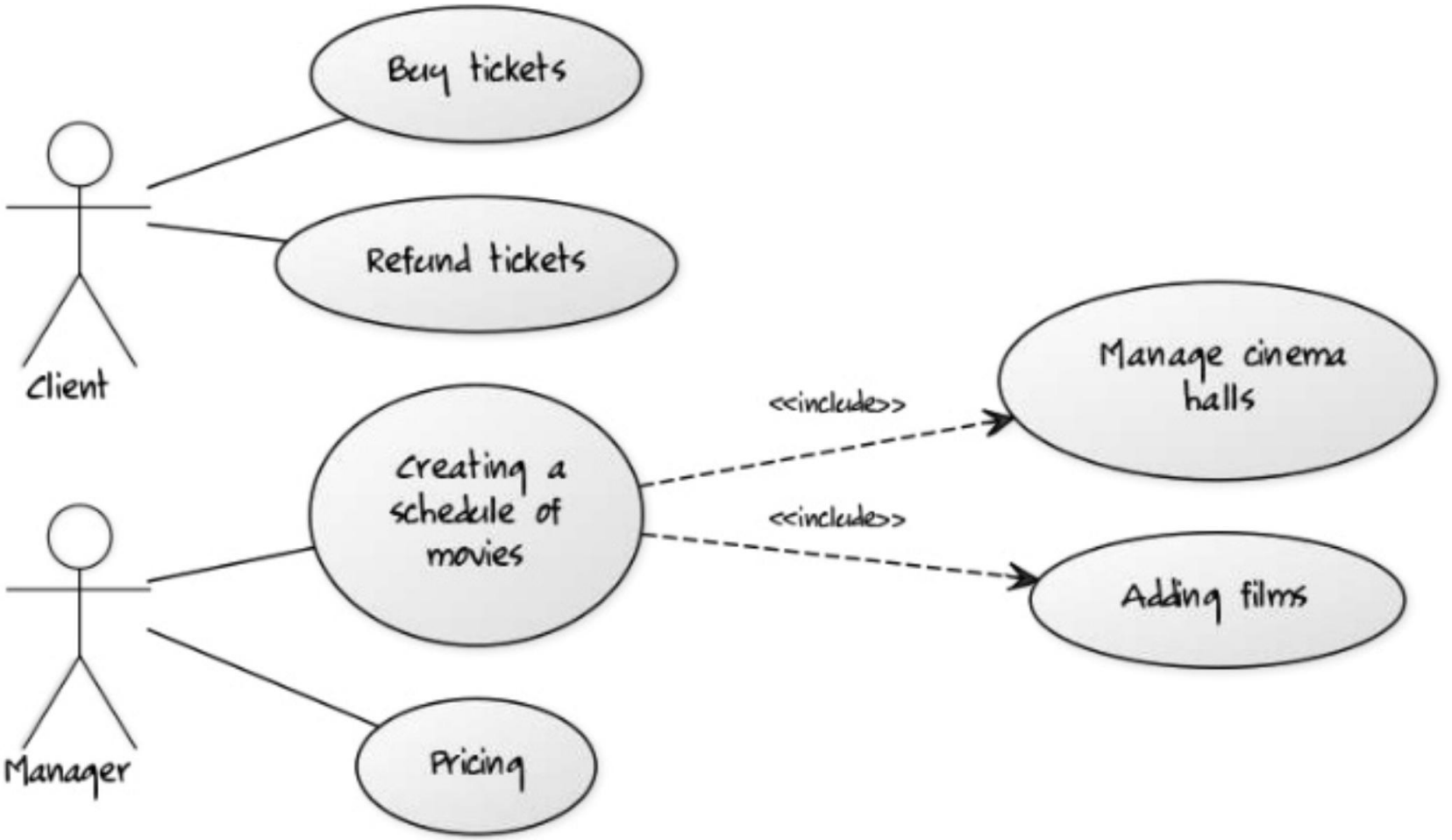


Help Dasha to find business logic

Clip slide

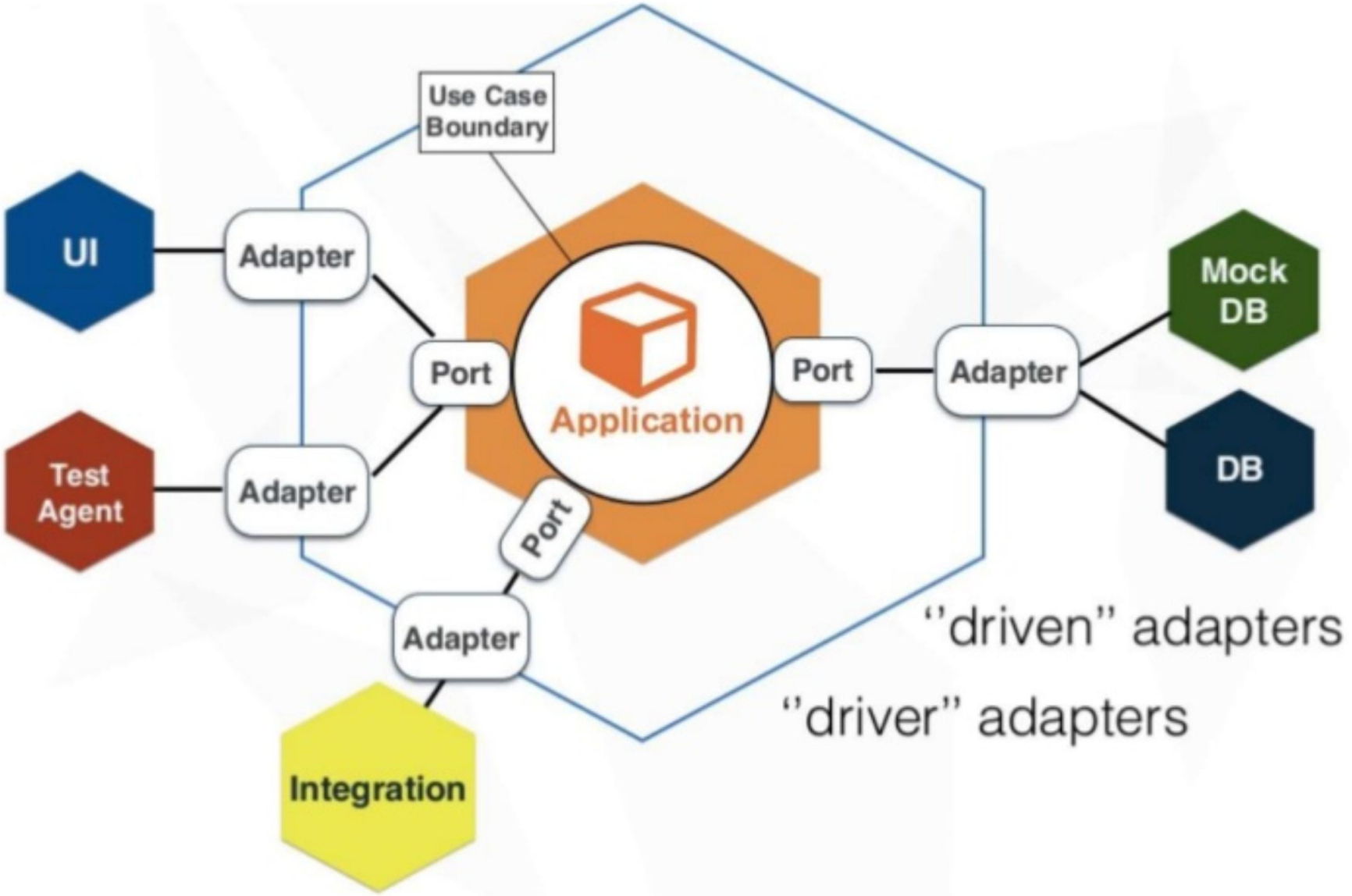
Everything should be Use Case driven

Clip slide



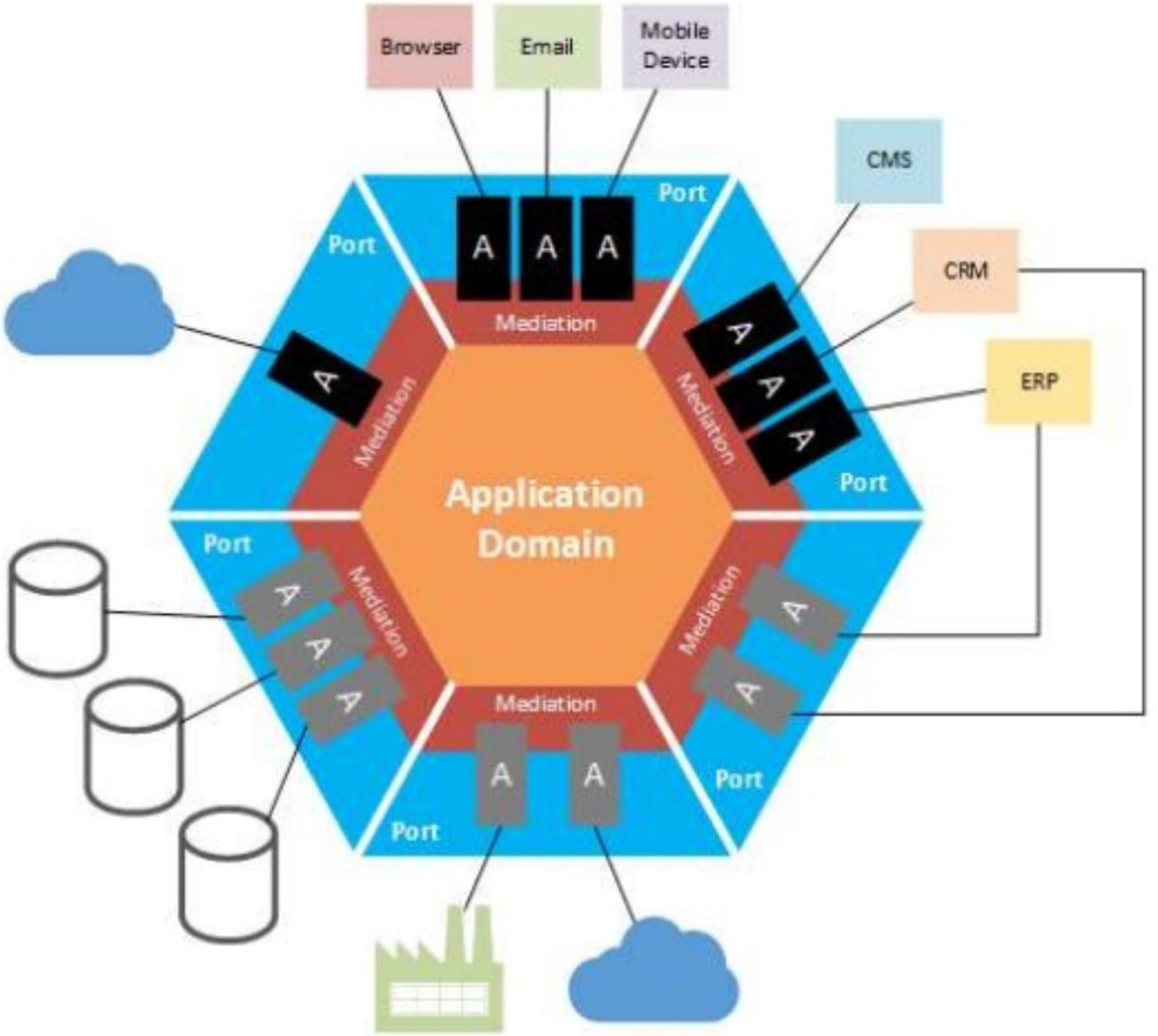
Alternative world

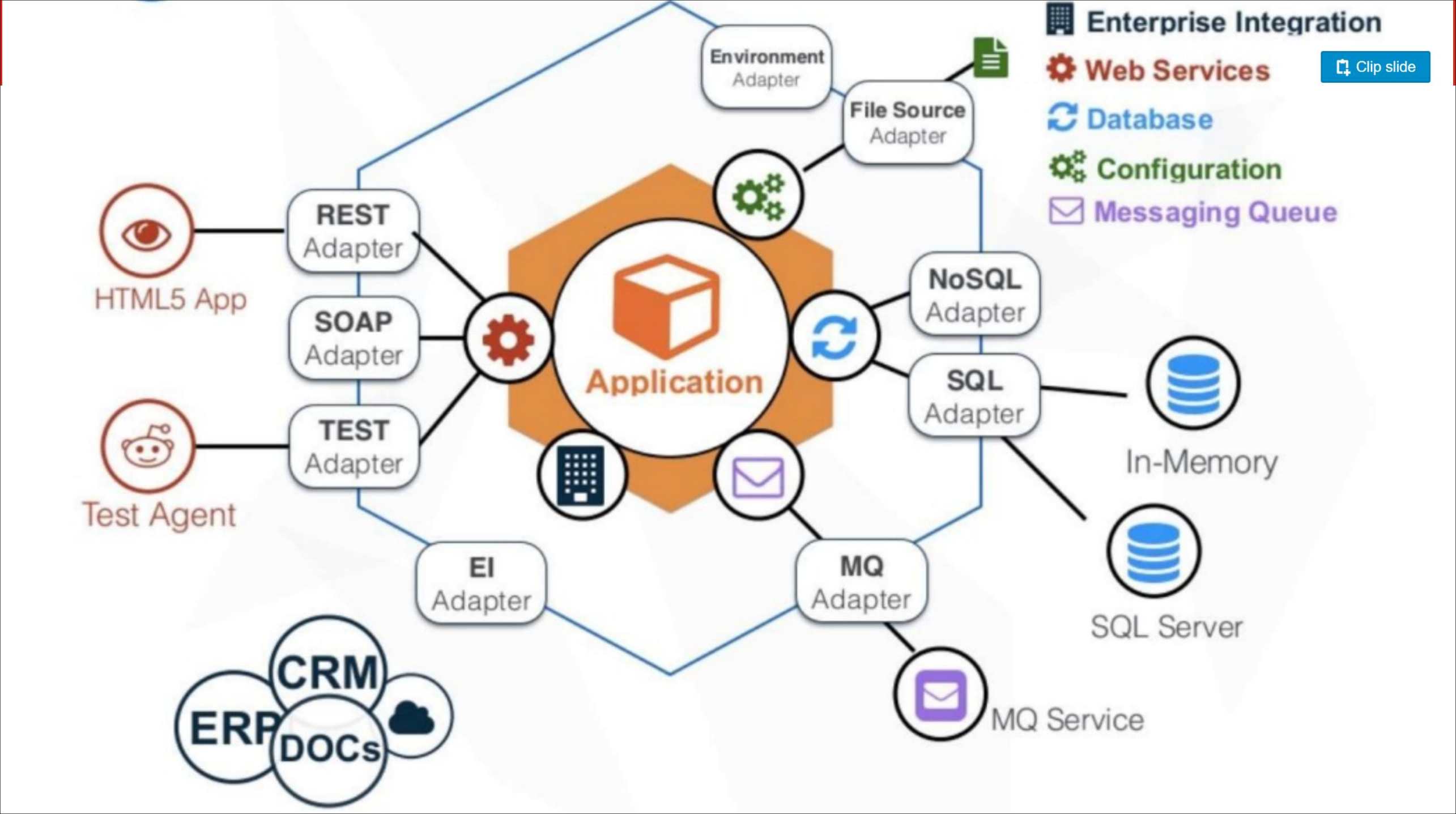
Clip slide

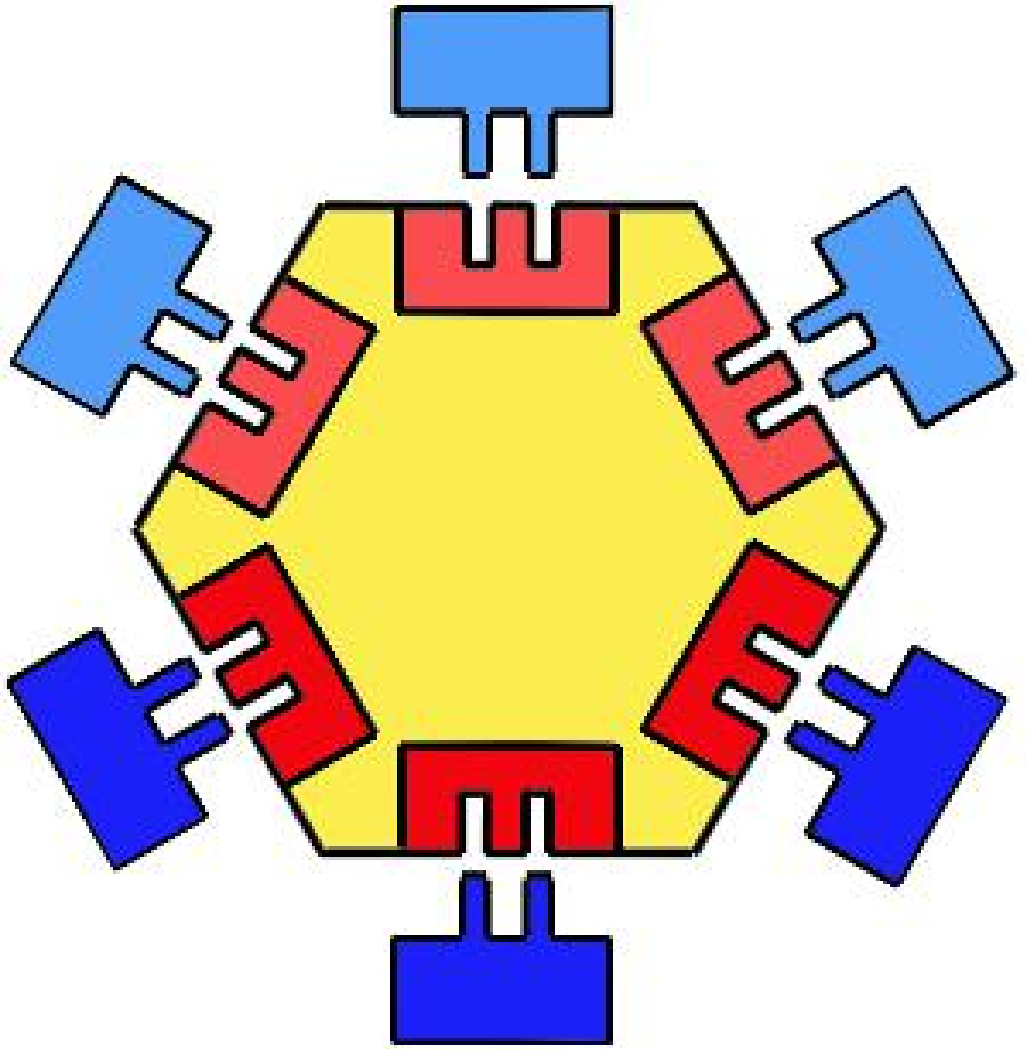


Ports and adapters thinking

Clip slide




Enterprise Integration
Web Services
Database
Configuration
Messaging Queue
Clip slide
Environment Adapter
File Source Adapter
REST Adapter
SOAP Adapter
TEST Adapter
HTML5 App
Test Agent
Application
NoSQL Adapter
SQL Adapter
In-Memory
SQL Server
EI Adapter
MQ Adapter
MQ Service
CRM
ERP
DOCs



yellow: core logic
light red: primary ports
light blue: primary adapters
dark red: secondary ports
dark blue: secondary adapters

A **port** is an entry point, provided by the core logic. It defines a set of functions.

Primary ports are the main API of the application. They are called by the primary adapters that form the user side of the application. Examples of primary ports are functions that allow you to change objects, attributes, and relations in the core logic.

Secondary ports are the interfaces for the secondary adapters. They are called by the core logic. An example of a secondary port is an interface to store single objects. This interface simply specifies that an object be created, retrieved, updated, and deleted. It tells you nothing about the way the object is stored.

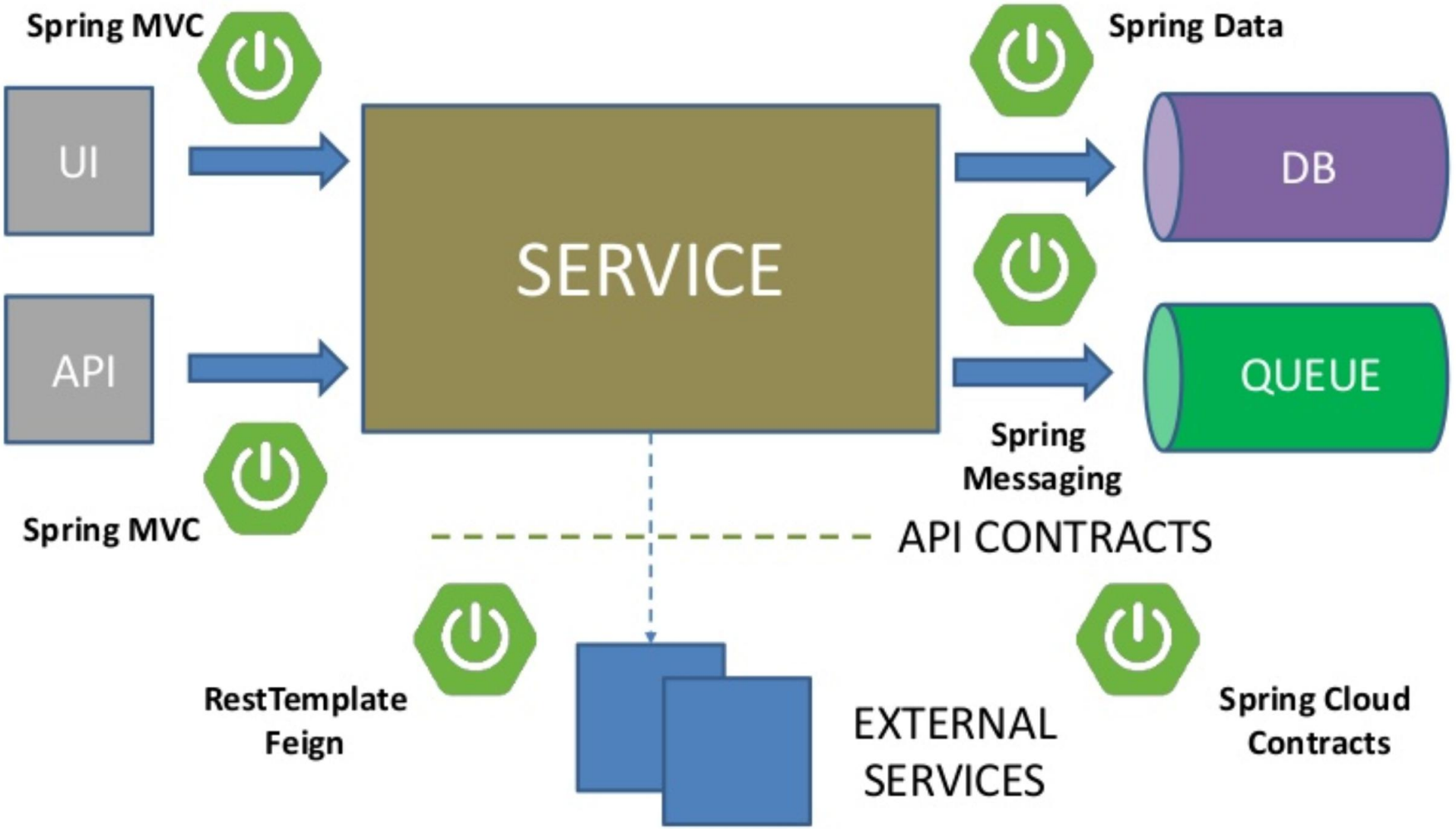
An **adapter** is a bridge between the application and the service that is needed by the application. It fits a specific port.

A **primary adapter** is a piece of code between the user and the core logic. One adapter could be a unit test function for the core logic. Another could be a controller-like function that interacts both with the graphical user interface and the core logic. The primary adapter calls the API functions of the core logic.

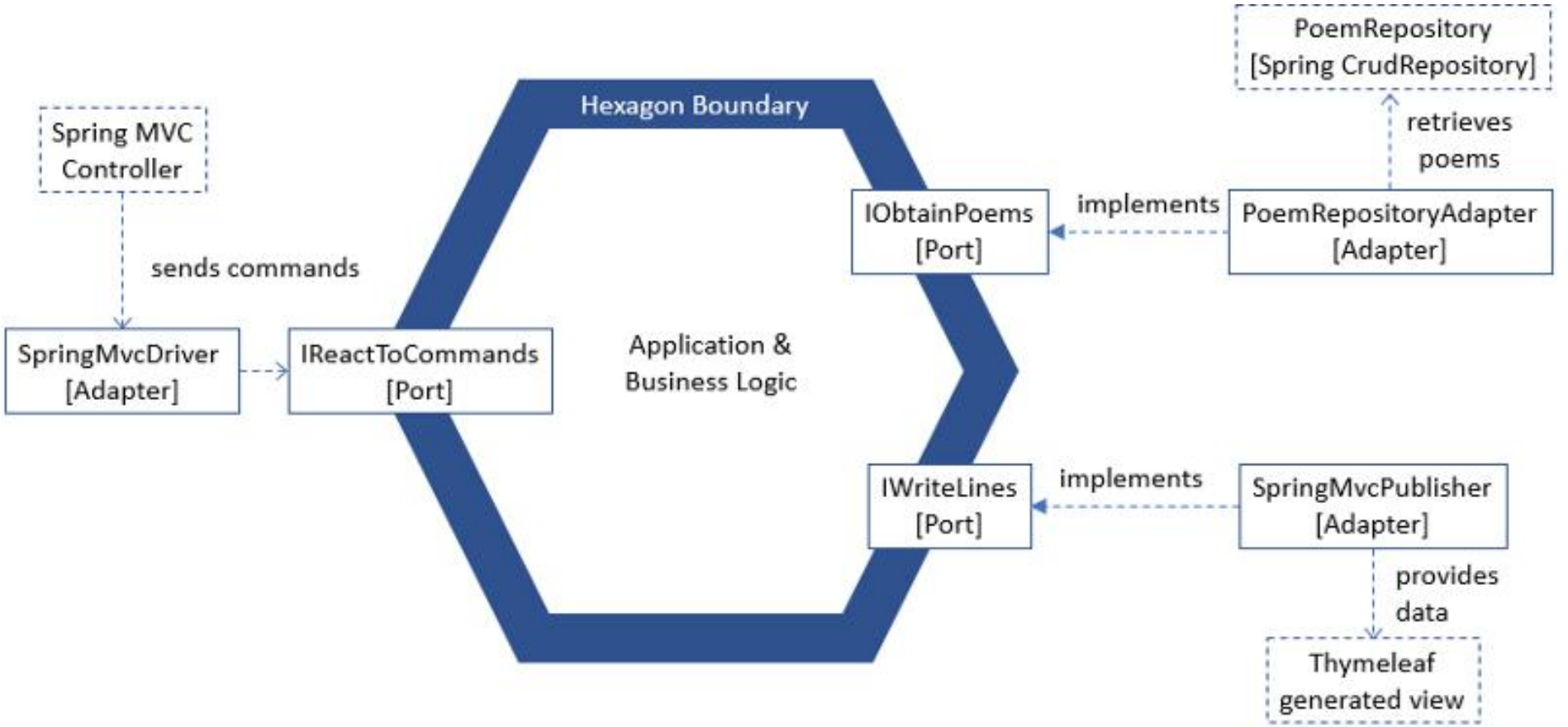
A **secondary adapter** is an implementation of the secondary port (which is an interface). For instance, it can be a small class that converts application storage requests to a given database, and return the results of the database in a format requested by the secondary port. It can also be a mock database object needed to unit tests certain parts of the core logic. The core logic calls the functions of the secondary adapter.

Each service has similar structure

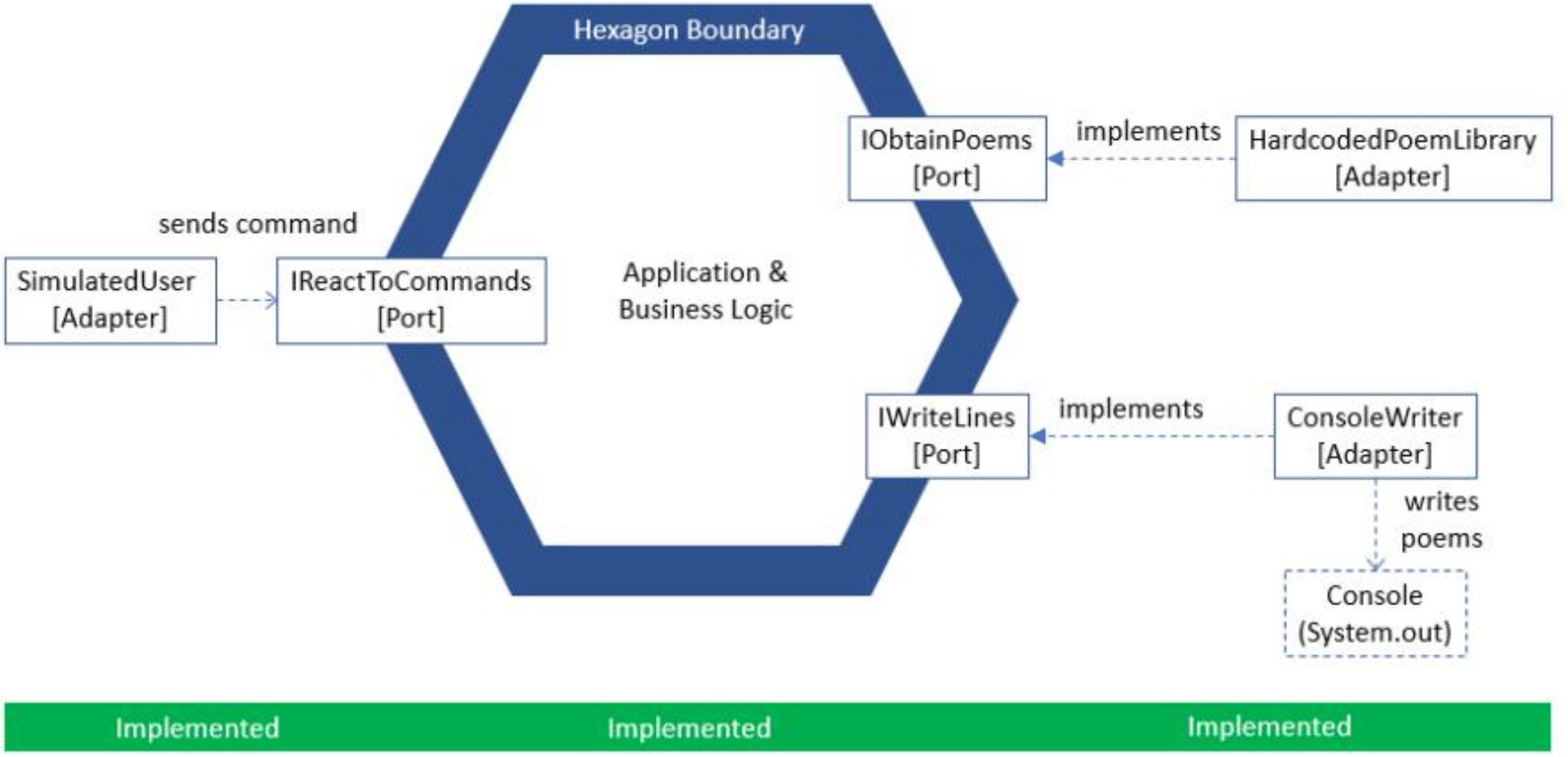
Clip slide

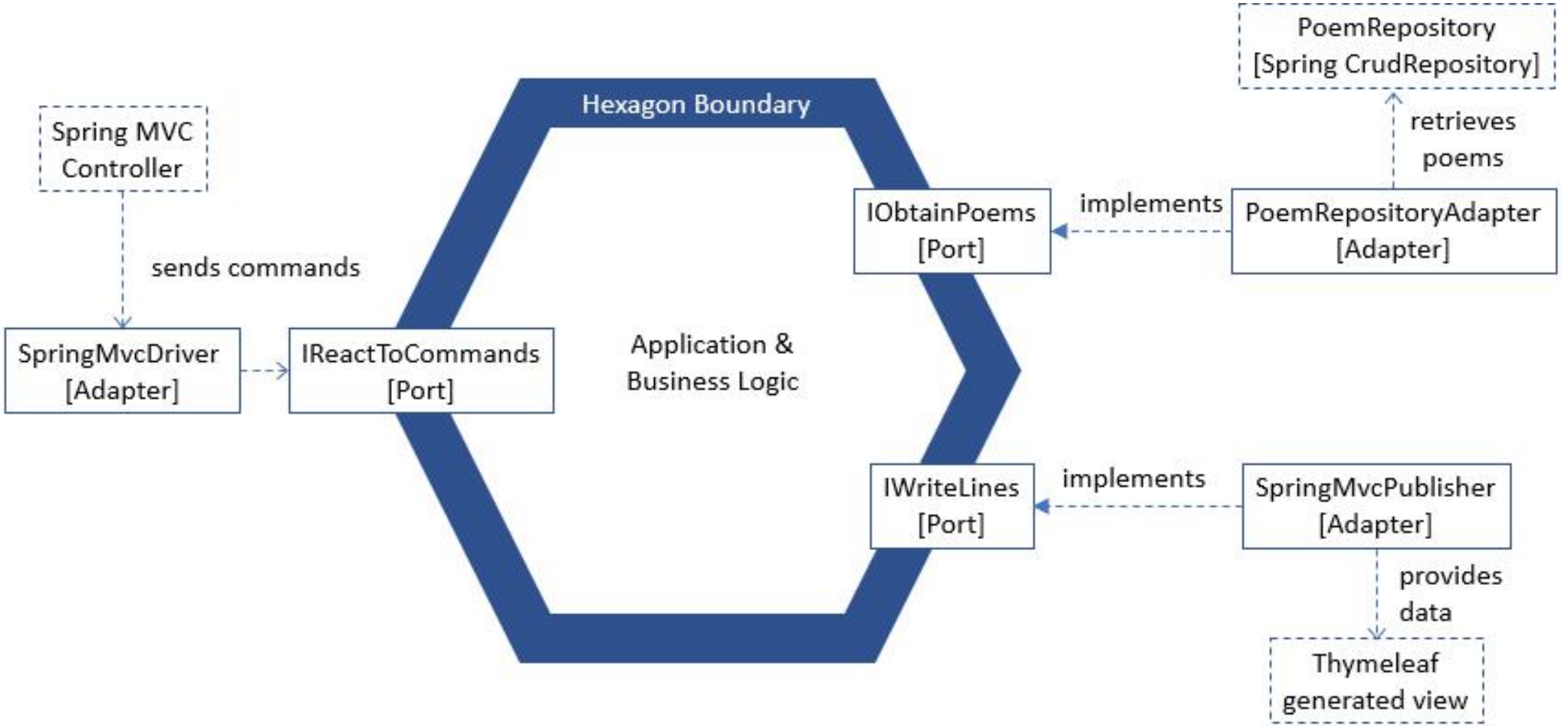


Example



Extracted from: https://www.freecodecamp.org/news/implementing-a-hexagonal-architecture/
Code: https://github.com/bertilmuth/poem-hexagon


Hexagon Boundary
IObtainPoems
[Port]
implements
HardcodedPoemLibrary
[Adapter]
sends command
SimulatedUser
[Adapter]
IReactToCommands
[Port]
Application &
Business Logic
IWriteLines
[Port]
implements
ConsoleWriter
[Adapter]
writes
poems
Console
(System.out)
Implemented
Implemented
Implemented


PoemRepository
[Spring CrudRepository]
retrieves
poems
Hexagon Boundary
Spring MVC
Controller
IObtainPoems
[Port]
implements
PoemRepositoryAdapter
[Adapter]
sends commands
SpringMvcDriver
[Adapter]
IReactToCommands
[Port]
Application &
Business Logic
IWriteLines
[Port]
implements
SpringMvcPublisher
[Adapter]
provides
data
Thymeleaf
generated view

bit Computer Training by Netmind

Boosting Your Digital Skills

www.bit.es

Barcelona
C/Almogàvers 123
08018 Barcelona
T +34 933 041 720
info@bit.es

Madrid
Plaza Carlos Trías Bertrán 7
Edificio Sollube 1ª Planta
28020 Madrid
T +34 914 427 703
info@bit.es

Bilbao
C/ San Vicente 8
Edificio Albia I, 6ª Planta
48001 Bilbao
T +34 944 354 982
info@bit.es

© 2018 BIT SAU, Barcelona