CodeDesign

Drive predictability through Software Design

Monolith or Microservices?
Designing Deploy-Time Flexibility for Modular Systems

Florin Coroș

florin@onCodeDesign.com

linkedin.com/in/florincoros

CodeDesign

Drive predictability through Software Design

Florin Coroș

Software Architect Consultant

Technical Trainer

Founder of Code Design

enjoing playing GO

enjoing traveling

oncodedesign.com/monolith-or-microservices/

CodeDesign

Drive predictability through Software Design

Monolith or Microservices?
Designing Deploy-Time Flexibility for Modular Systems

Florin Coroș

florin@onCodeDesign.com

linkedin.com/in/florincoros

From Monolith to Micro-services

the MONOLITH

Not good at

Scalability	Maintainability	
Reliability	Testability	Security
Availability	Modernization	High Coupling
Resilience	Extensibility	

REDUNDANCY DECOMPOSITION

oncodedesign.com/monolith-or-microservices/

From Micro-services back to a Monolith

Not good because

Complex Communication

Performance

Complex Debugging & Diagnostics

Expensive Infrastructure

Maintainability

Testability

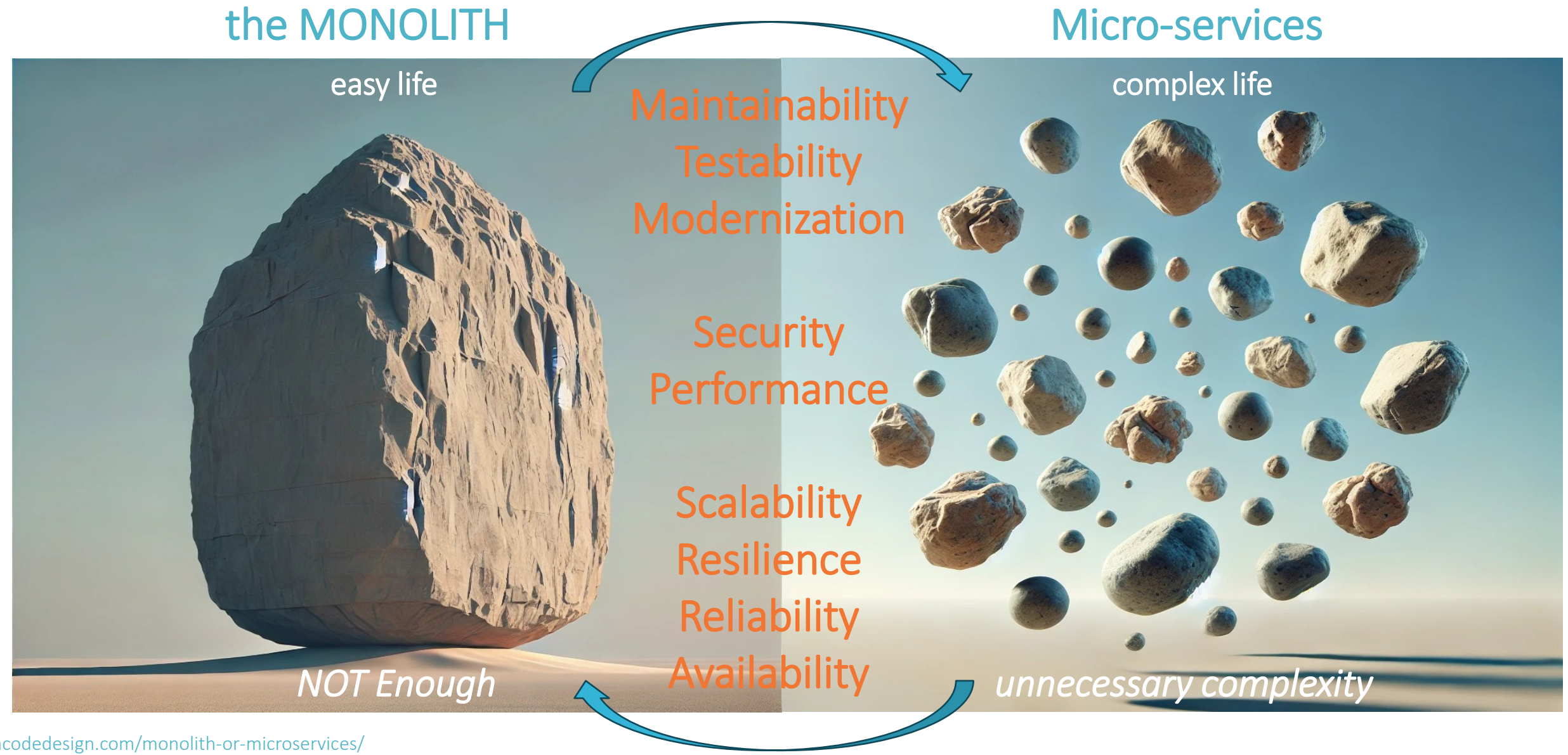
UNNECESARY COMPLEXITY

DECOMPOSITION

Micro-services

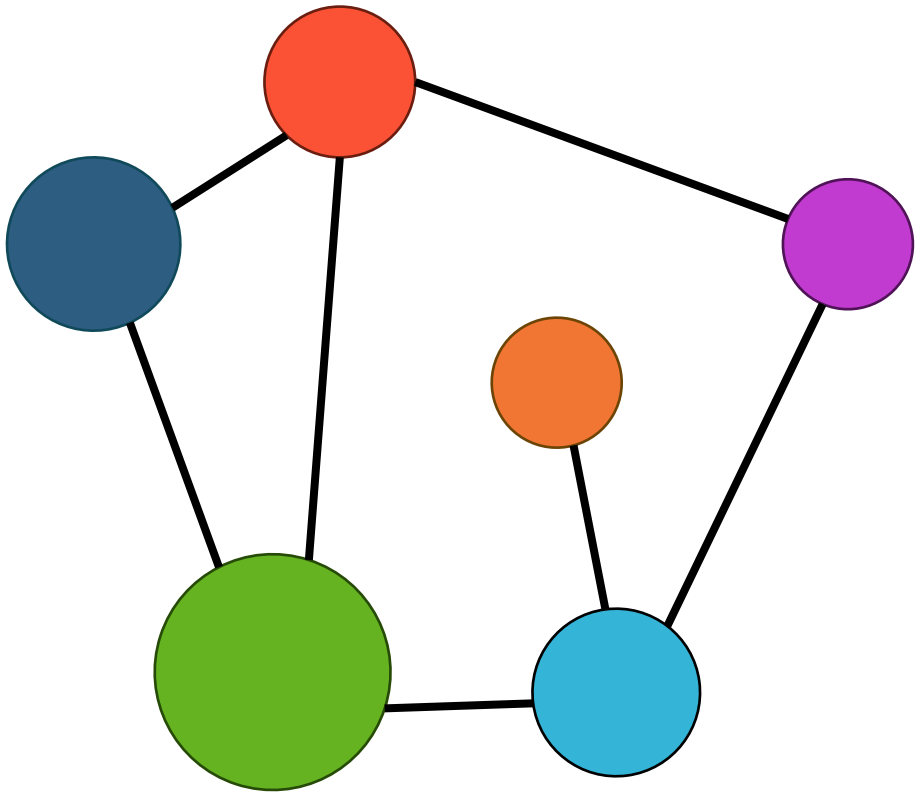
oncodedesign.com/monolith-or-microservices/

History Repeats Itself



oncodedesign.com/monolith-or-microservices/

Modular System - Concept

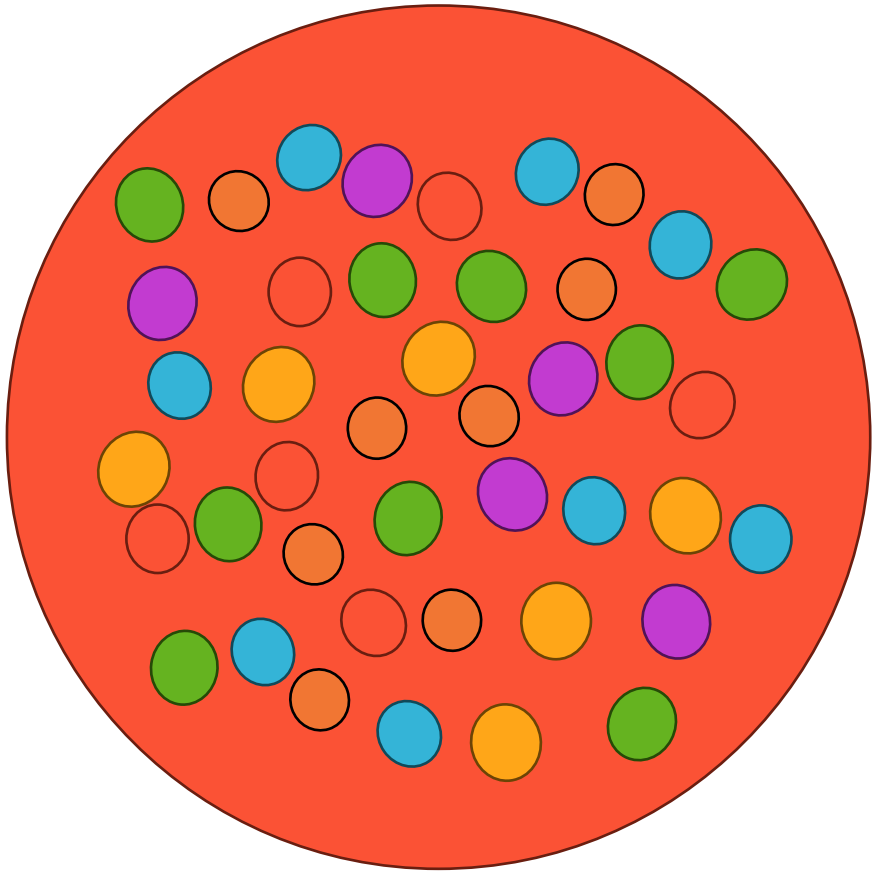


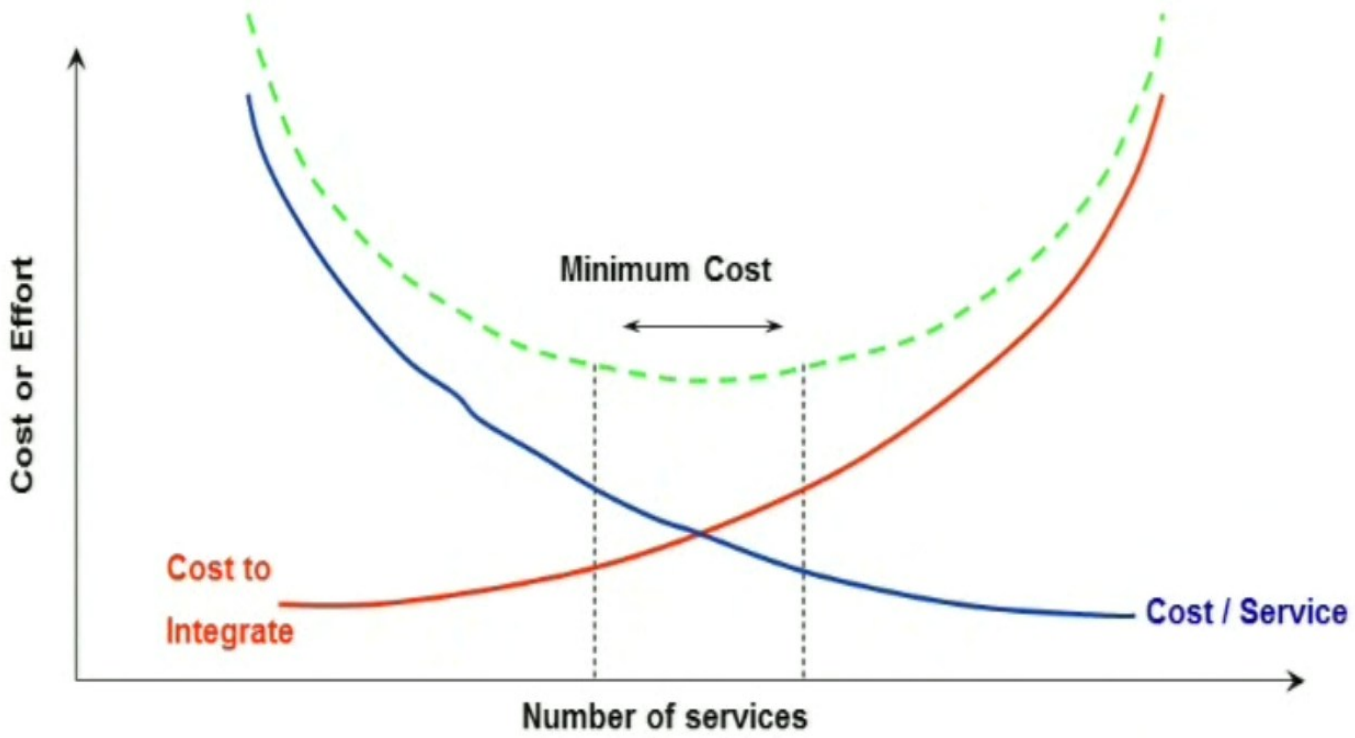
- Maintainability
- Extensibility
- Reusability

Separate the Communication Concern

oncodedesign.com/monolith-or-microservices/

How many services?





oncodedesign.com/monolith-or-microservices/

from *Righting Software* by Juval Lowy

Contracts – Are Key in Modular Systems

Services communicate through Explicit Contracts

- Abstract the functions it provides
- Encapsulate (hide) the implementation details

Contracts described with language constructs:

- Operation Contracts – functions the interfaces
- Data Contracts – DTOs (the in/out params)
- Fault Contracts – Exceptions

Synchronous Communication – function calls

Asynchronous Communication – message based

Target: Design for Deploy-Time Flexibility

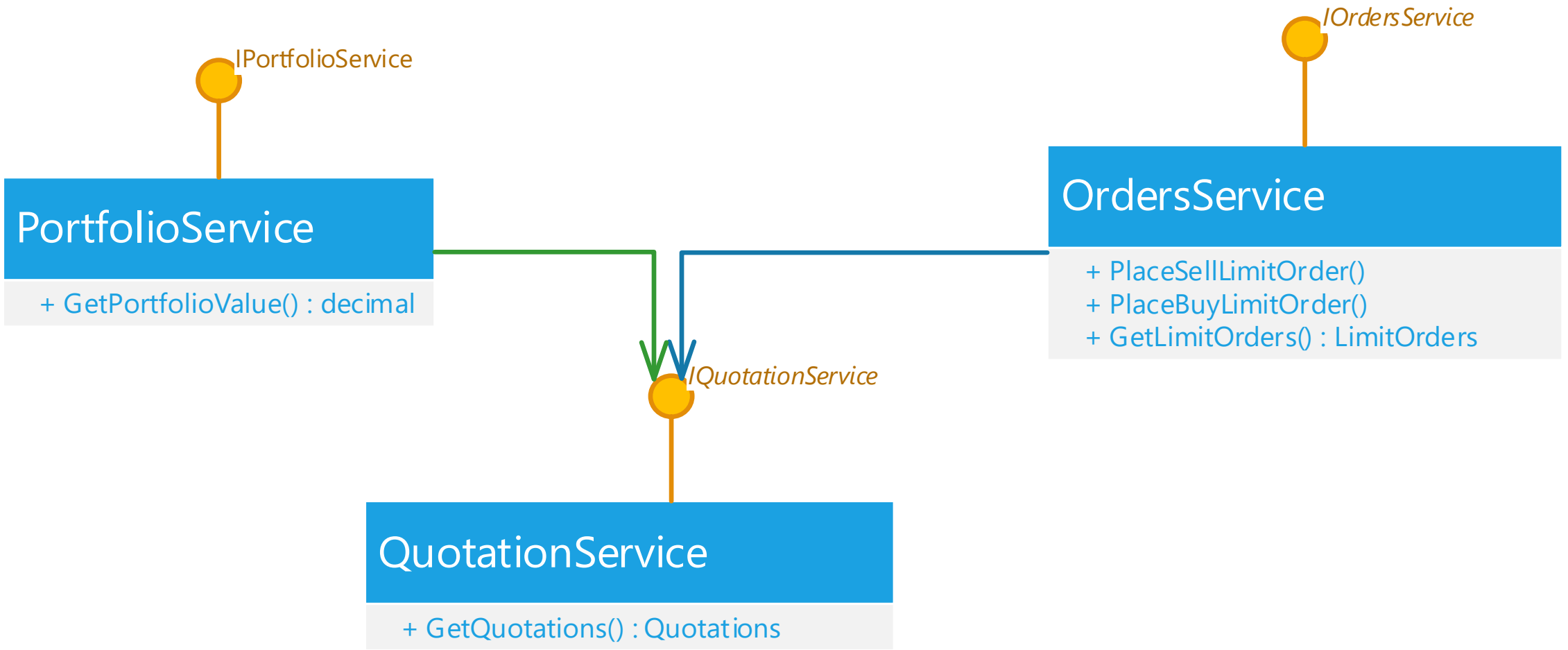
Decide ONLY at Deploy-Time if

deploy as a Monolith or

deploy as a Distributed System

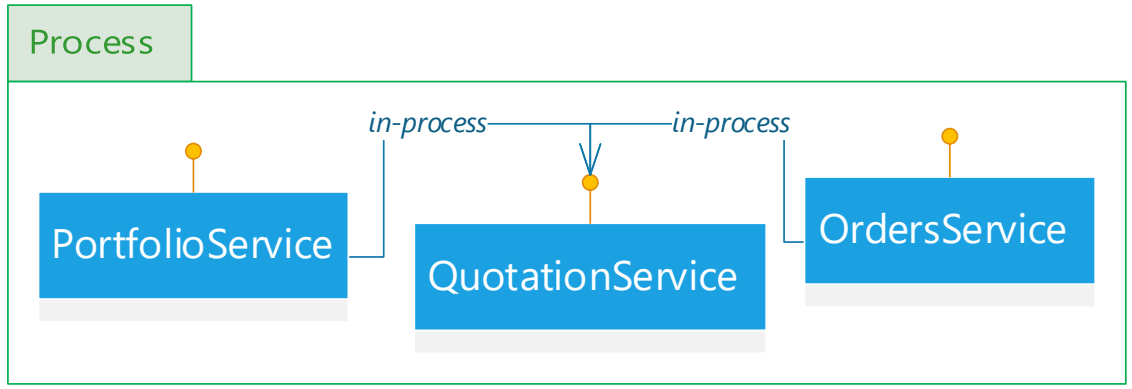
without changing the code
without recompile the code

DEMO: Simplified Example of Dependent Services

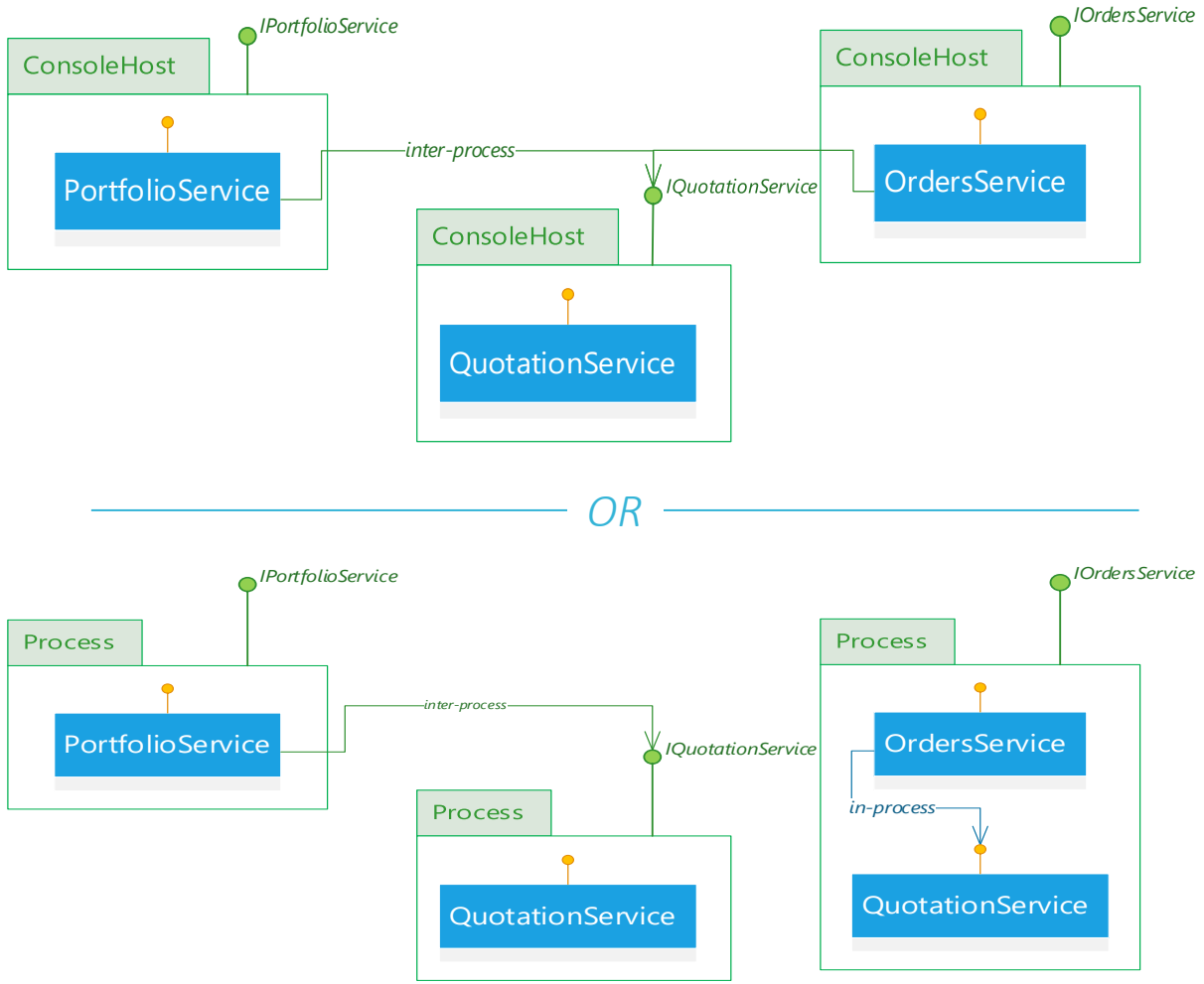


Decide at Deployment between Monolith or Micro-services

Monolith



Micro-services

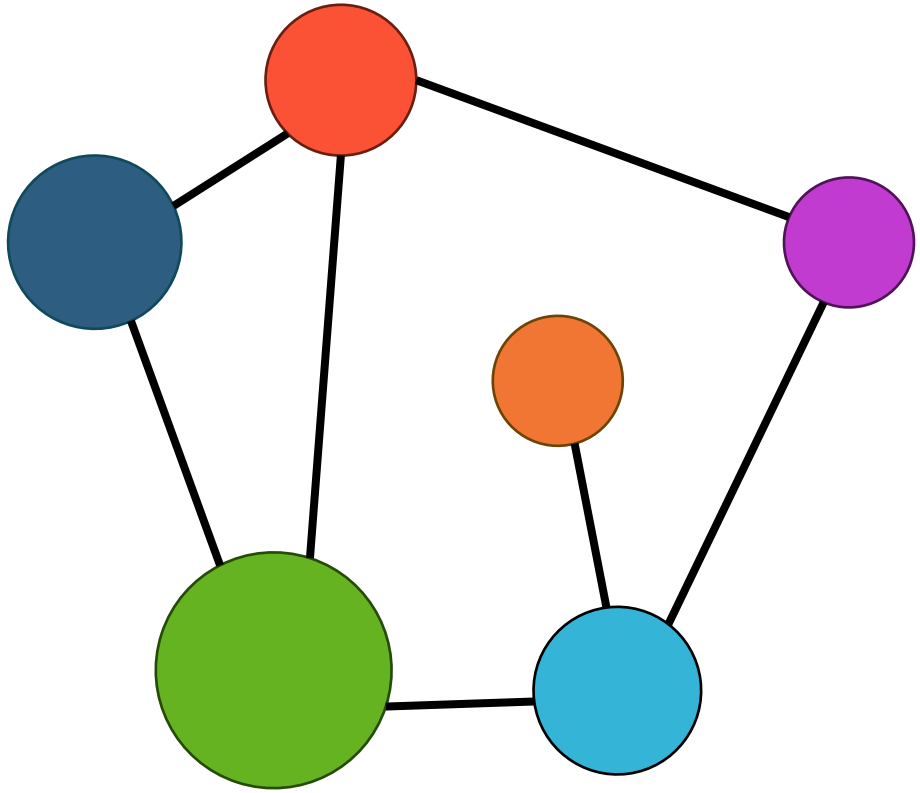


Solution for Design for Deploy-Time Flexibility

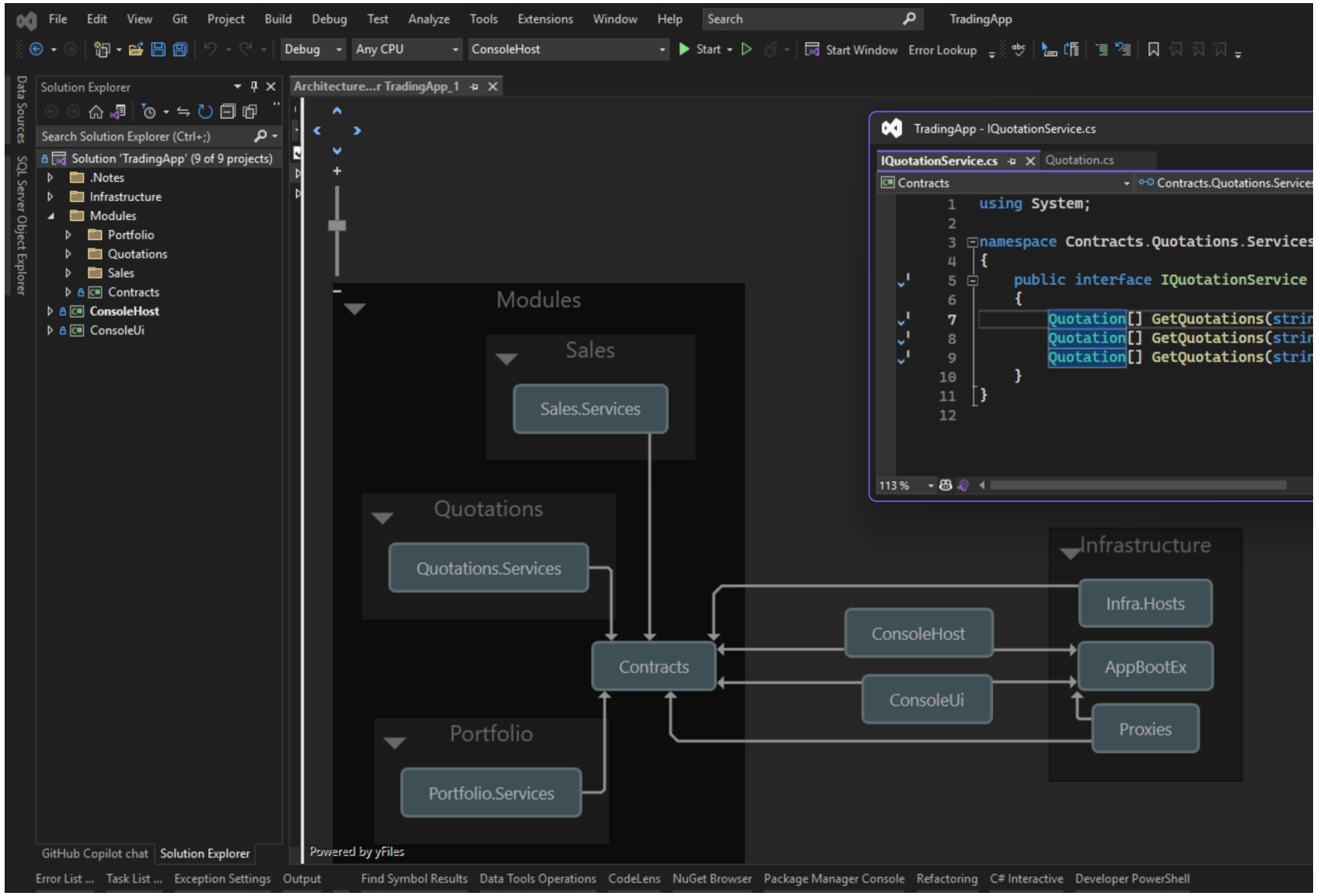
The solution stands on Three Pillars

1. **Depend only on CONTRACTS**
   written with abstract types
2. Use **Proxies** to forward the calls to the actual implementation
3. Generic Hosts with **Type Discovery**

Modular System



Coding Demo



Github Repo:

[Code-Design-Training / InterProcessCommunication / TradingApp]

Demo build up blog posts:

[oncodedesign.com/tag/communication]

CodeDesign

Drive predictability through Software Design

Designing Deploy-Time Flexibility for Modular Systems

florin@onCodeDesign.com

linkedin.com/in/florincoros

oncodedesing.com/training

oncodedesing.com/monolith-or-microservices

calendly.com/florin-oncodedesign/short-call

Florin Coroș

Software Architect Consultant

Technical Trainer