

Java/Spring 테스트를 추가하고 싶은 개발자들의 오답노트

잘못된 테스트 사례 학습

아키텍처

헥사고날 아키텍처

INDEX

01

아키텍트

갑자기 아키텍처?
아키텍트의 목표

02

헥사고날

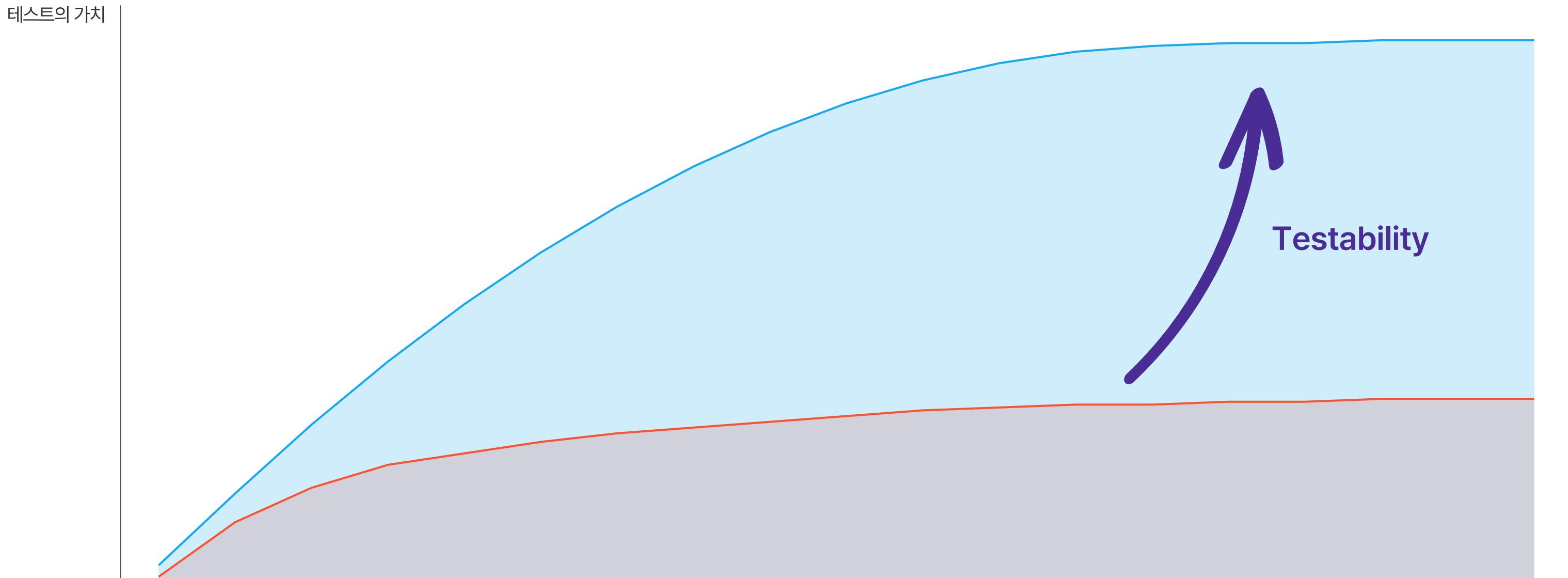
의존성 역전
진화하는 아키텍처

01. 아키텍트

- 갑자기 아키텍트를 왜 이야기 하는가?
- 아키텍처와 테스트의 상관관계
- 아키텍트의 목표

갑자기 아키텍처?

- 올바른 테스트의 가치 ref. Effective Unit Testing

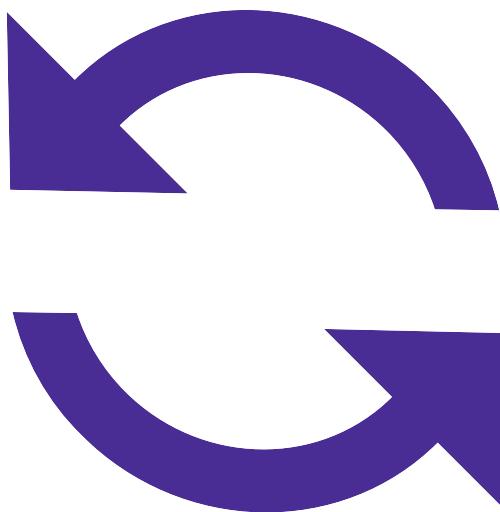


갑자기 아키텍처?

- 아키텍처와 테스트의 상호관계

테스트가 너무 힘든데?

테스트



아키텍처

뭐가 정답이지...?

갑자기 아키텍처?

- 어디에서든 통하는 강력한 무기



테스트하기 쉬운 코드가 좋은 코드일 확률이 높다

아키텍트의 정의

○ 아키텍처의 정의

“ 아키텍처란 어떤 비즈니스 문제를 해결하기 위해, **준수해야하는 제약**을 넣는 과정.

아키텍트의 정의

○ 아키텍처의 정의

“ 아키텍처는 종착지가 아니라 여정에 더 가까우며, 고정된 산출물이 아니라 계속된 탐구 과정에 더 가까움을 이해해야 좋은 아키텍처가 만들어진다.
- 케블린 헨니 (Kevlin Henney)

로버트 C. 마틴 저, 클린 아키텍처 소프트웨어 구조와 설계의 원칙, 송준이 옮김, (인사이트(insight), 2019-08-20), 추천사 xix

아키텍트의 목표

○ 아키텍트의 목표

“ 소프트웨어 아키텍처의 목표는 필요한 시스템을 만들고 유지보수하는데 투입되는 **인력을 최소화** 하는 데 있다.

로버트 C. 마틴 저, 클린 아키텍처 소프트웨어 구조와 설계의 원칙, 송준이 옮김, (인사이트(insight), 2019-08-20), 7p

“ 아키텍터의 목표는 시스템에서 정책을 가장 핵심적인 요소로 식별하고, 동시에 세부사항은 정책에 무관하게 만들 수 있는 형태의 시스템을 구축하는 데 있다. 이를 통해 세부사항을 결정하는 일은 미루거나 연기할 수 있게 된다.

로버트 C. 마틴 저, 클린 아키텍처 소프트웨어 구조와 설계의 원칙, 송준이 옮김, (인사이트(insight), 2019-08-20), 147p

아키텍트의 목표

○ 아키텍트의 목표

- 인적 자원 절감
- 정책을 만들고 세부사항을 미루는 시스템 개발

아키텍트의 목표

○ 아키텍트의 목표

- 인적 자원 절감
- 장애를 만들고 세부사항을 미루는 시스템 개발

동시 작업이 가능해야 한다

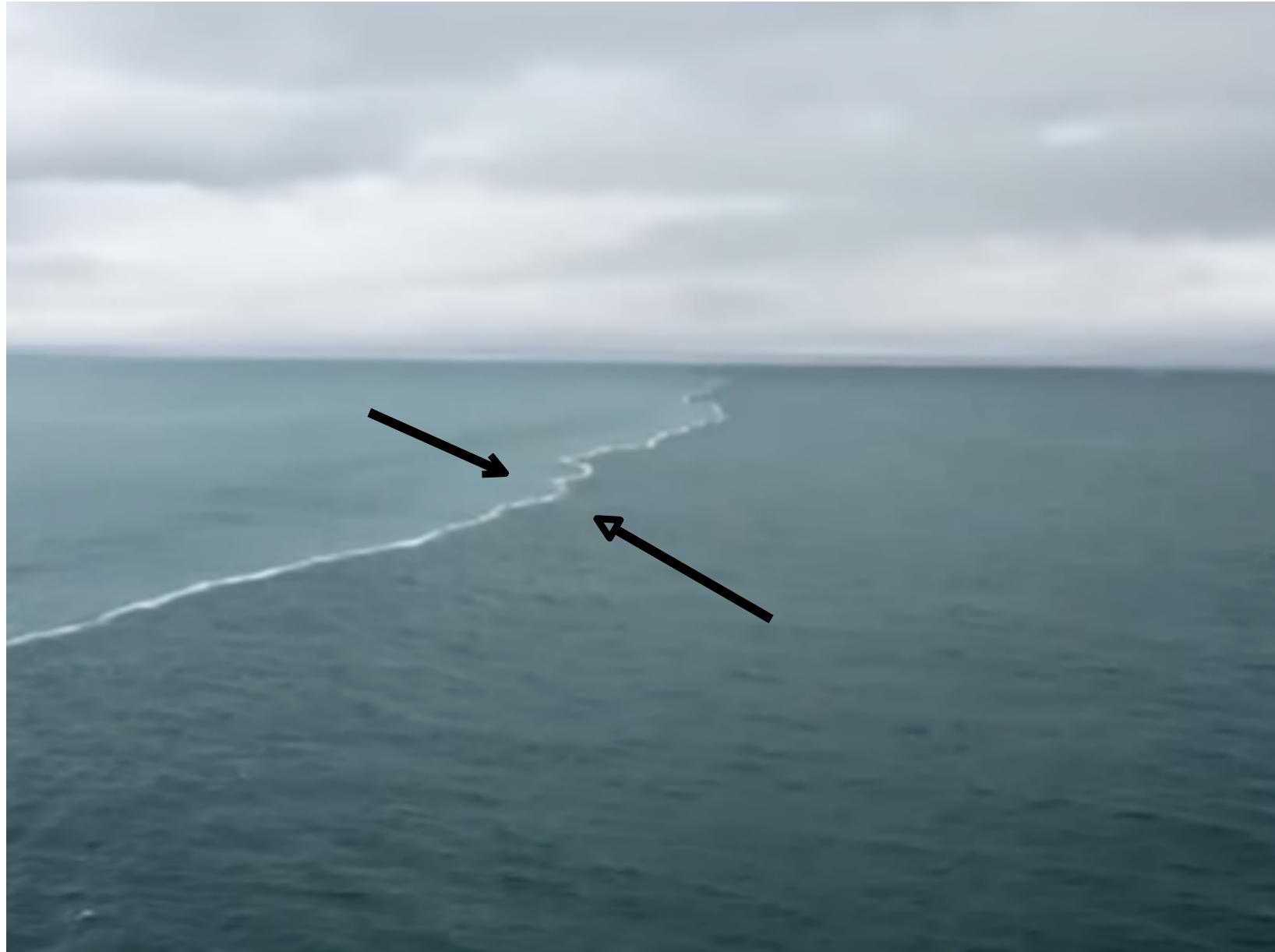
관심사를 분리해야 한다

경계를 나눠야 한다

아키텍트의 목표

○ 아키텍트의 목표

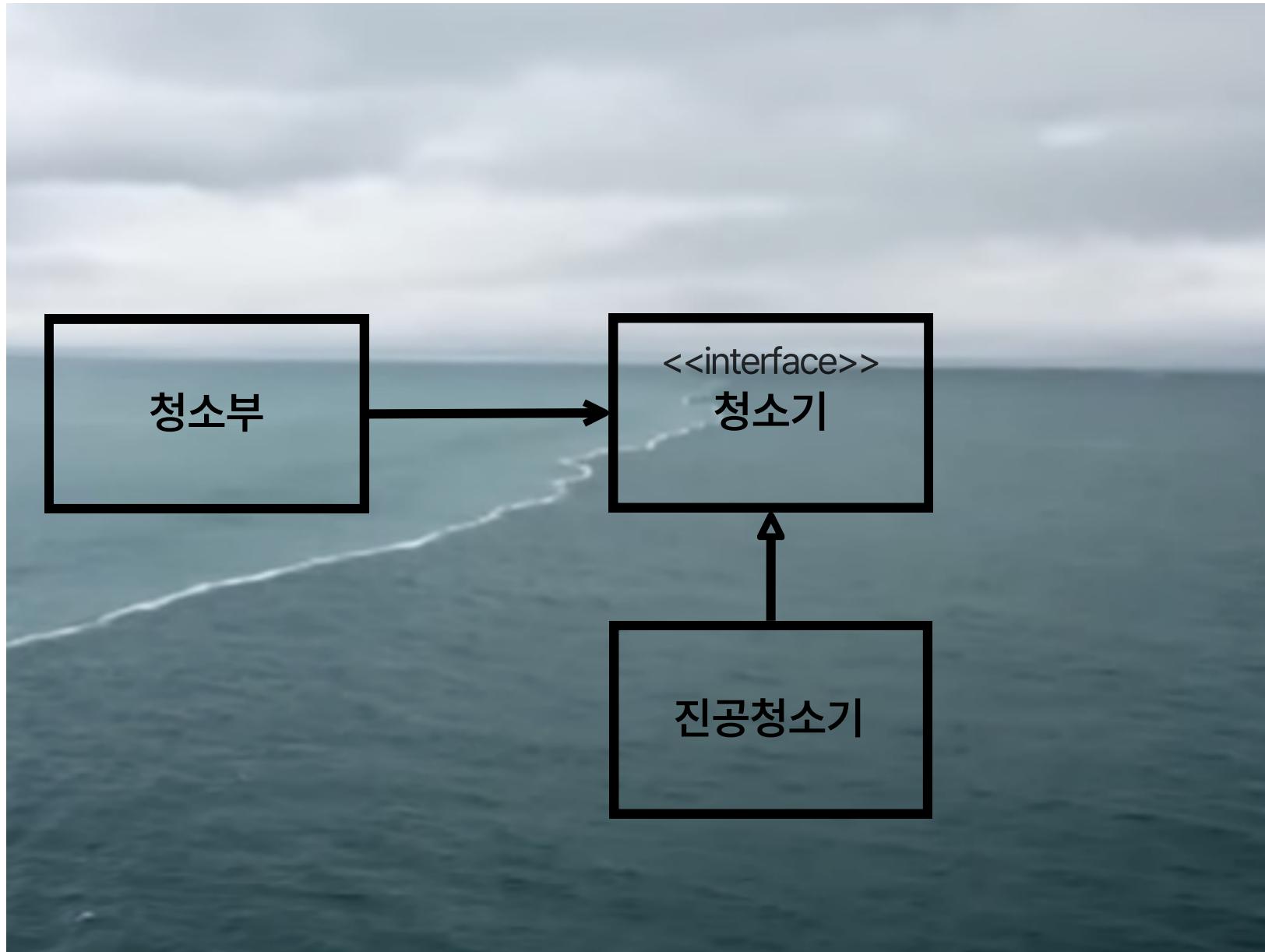
경계



아키텍트의 목표

○ 아키텍트의 목표

의존성 역전은 경계를 만드는 방법중 하나다.



아키텍트의 목표

○ 아키텍트의 목표

- 인적 자원 절감
- 장애를 만들고 세부사항을 미루는 시스템 개발

동시 작업이 가능하게 됨

관심사를 분리

의존성 역전 + 다른 여러 방법

아키텍트의 목표

○ 아키텍트의 목표

- 인적 자원 절감
- 정책을 만들고 세부사항을 미루는 시스템 개발

“개발 초기에는 데이터베이스 시스템을 선택할 필요가 없다. (중략)
개발 초기에는 웹 서버를 선택할 필요가 없다. (중략)
개발 초기에는 REST를 적용할 필요가 없다. (중략)
개발 초기에는 의존성 주입 프레임워크를 적용할 필요가 없다.”

로버트 C. 마틴 저, 클린 아키텍처 소프트웨어 구조와 설계의 원칙, 송준이 옮김, (인사이트(insight), 2019-08-20), 147p

아키텍트의 목표

○ 아키텍트의 목표

- 인적 자원 절감
- 정책을 만들고 세부사항을 미루는 시스템 개발

“개발 초기에는 Oracle인지 MySQL인지 Mongo인지 선택할 필요가 없다.
개발 초기에는 Nginx인지 Apache인지 선택할 필요가 없다.
개발 초기에는 REST인지 GraphQL인지 선택할 필요가 없다.
개발 초기에는 Spring을 적용할 필요가 없다.

도메인이 먼저 개발되어야 한다.

아키텍트의 목표

○ 아키텍트의 목표

- 인적 자원 절감
- 정책을 만들고 세부사항을 미루는 시스템 개발

“ 프레임워크와 결혼하지 말라!

로버트 C. 마틴 저, 클린 아키텍처 소프트웨어 구조와 설계의 원칙, 송준이 옮김, (인사이트(insight), 2019-08-20), 306p

“ 도메인이야 말로 소프트웨어의 핵심입니다. Spring / Jpa가 중요한 게 아닙니다.

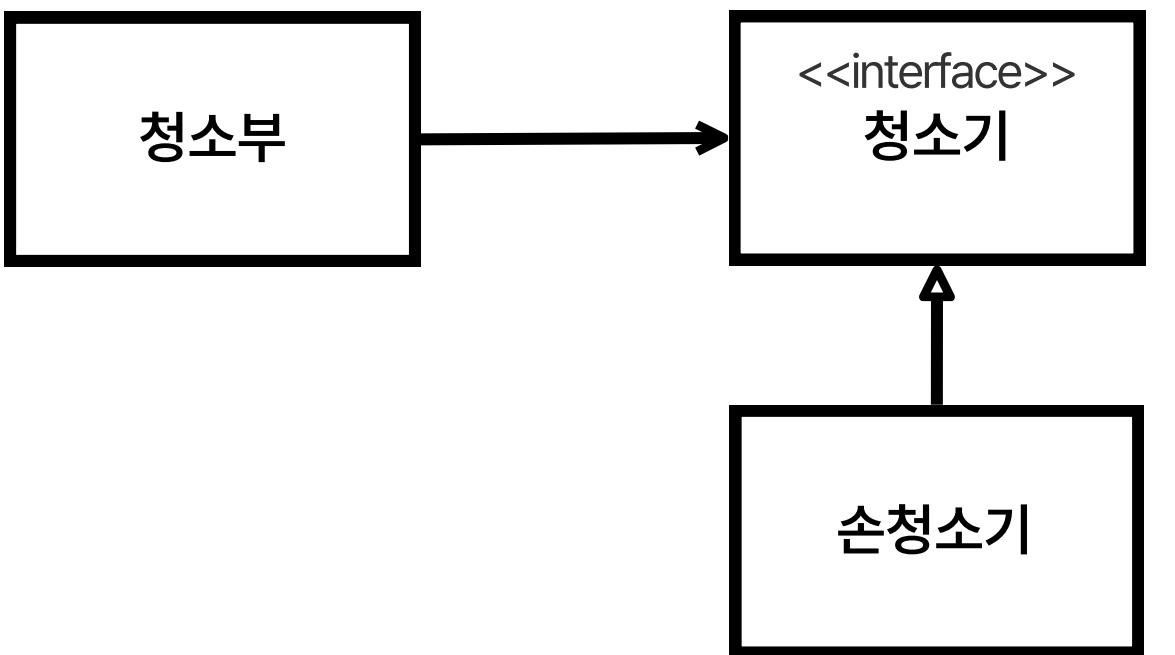
02. 헥사고날

- 의존성 역전과 경계
- 개선된 아키텍처를 진화시키기

헥사고날

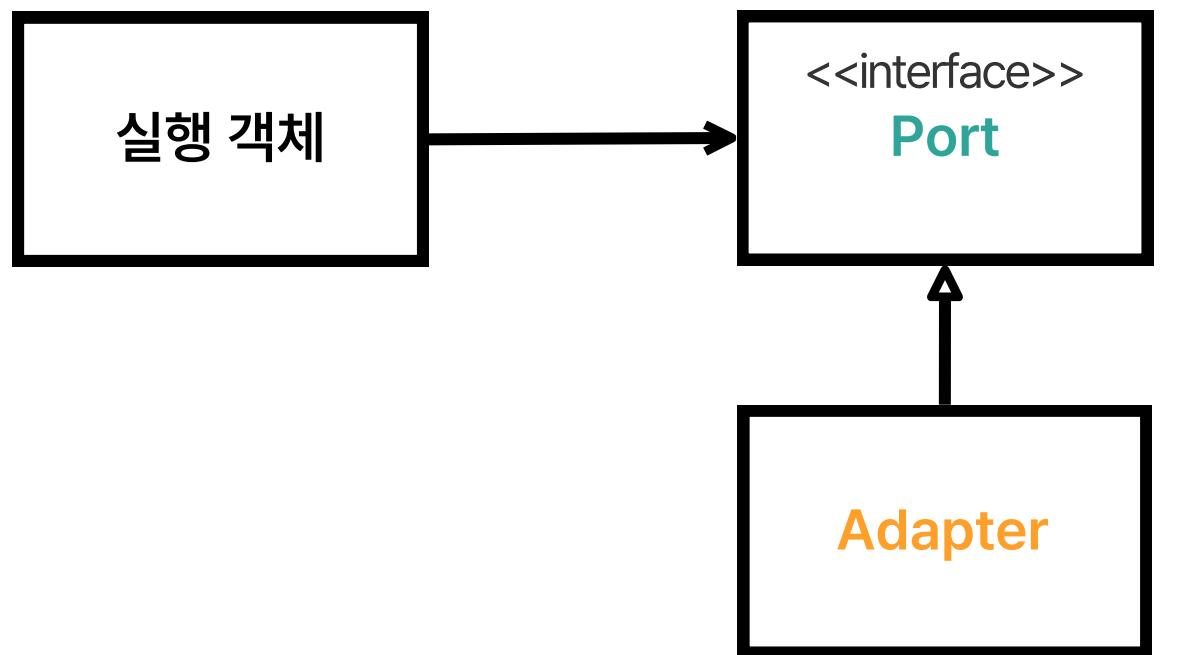
○ 의존성 역전

- = 관심사를 분리하고 싶다
- = 고립시키고 싶다
- = 서로에게 영향을 받고 싶지 않다



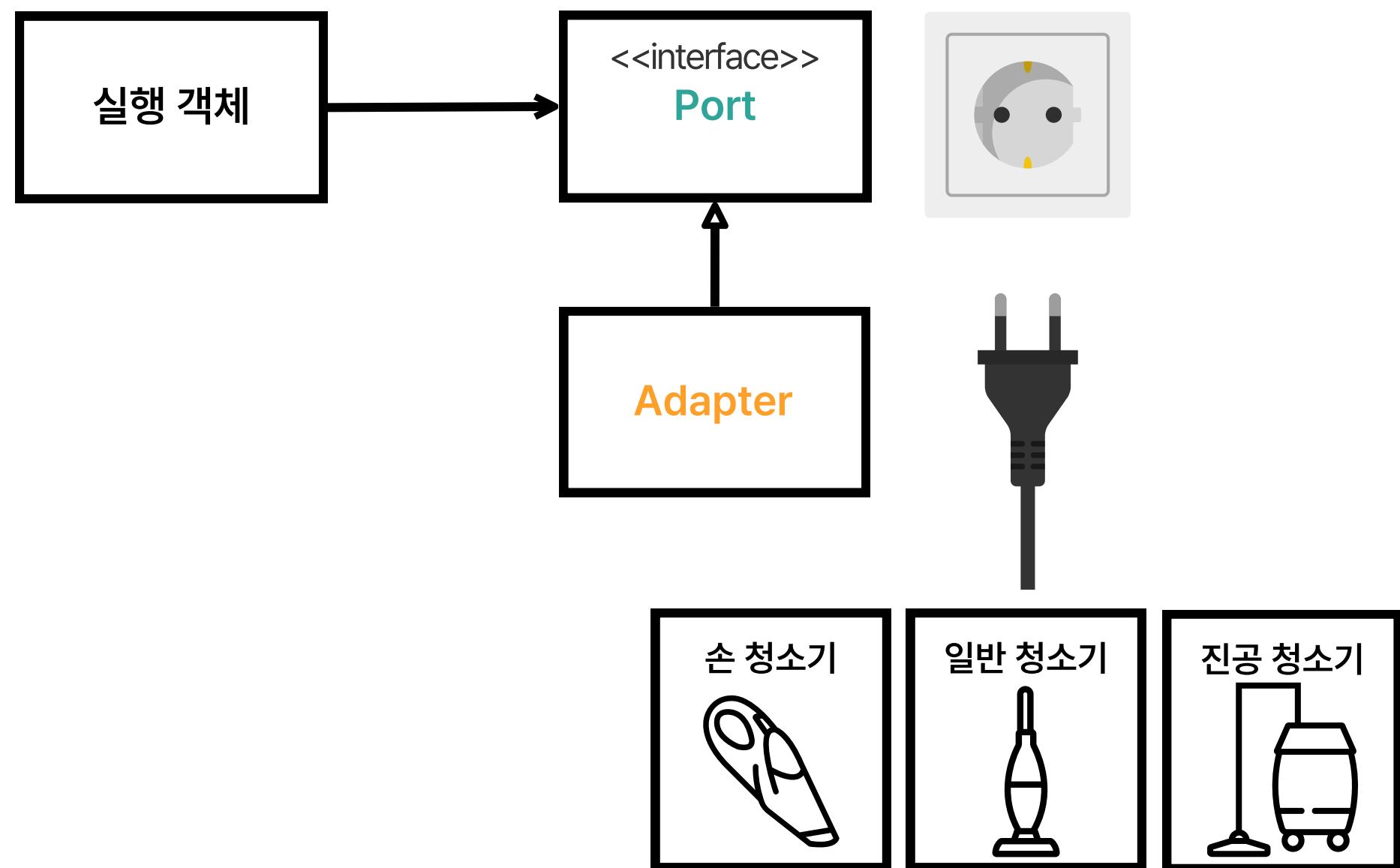
헥사고날

○ 의존성 역전 (포트-어댑터 패턴)



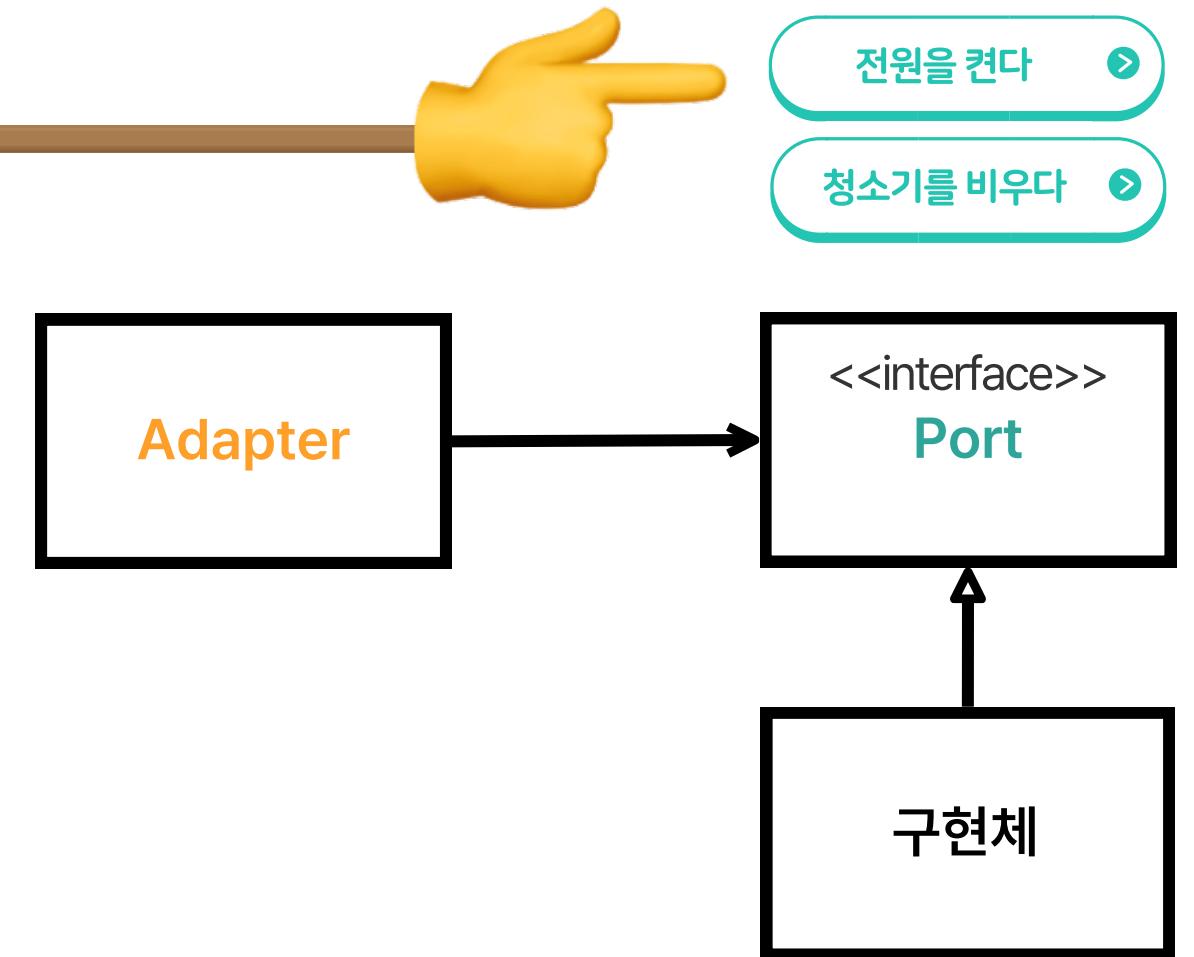
헥사고날

○ 의존성 역전 (포트-어댑터 패턴)



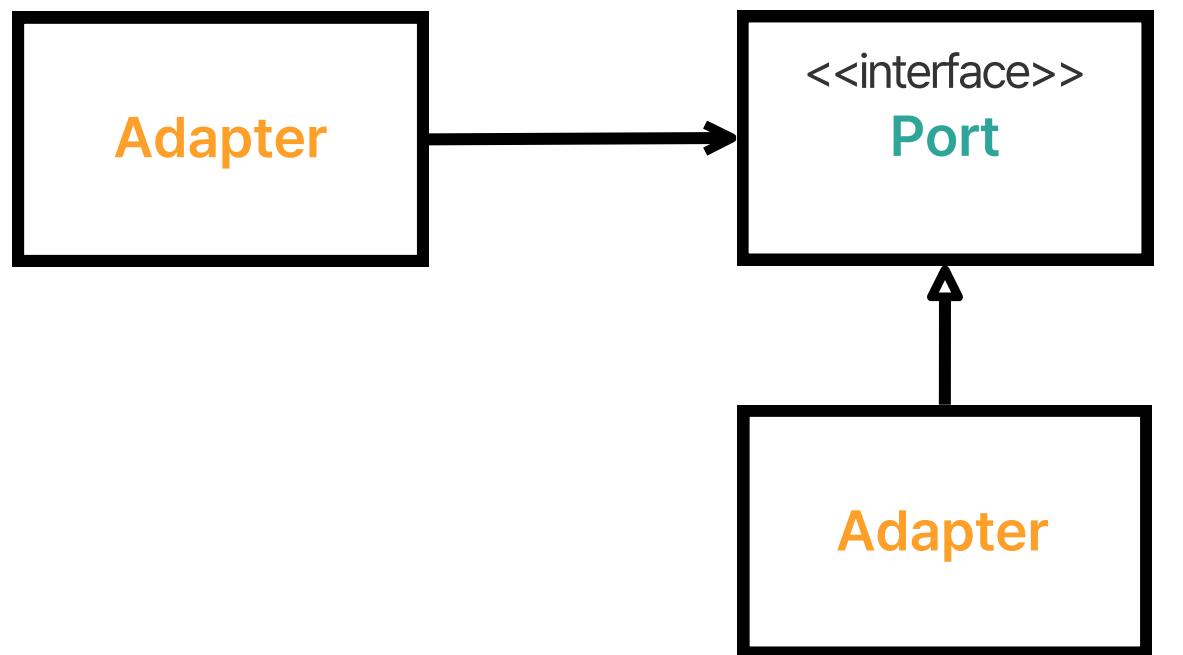
헥사고날

○ 의존성 역전 (포트-어댑터 패턴)



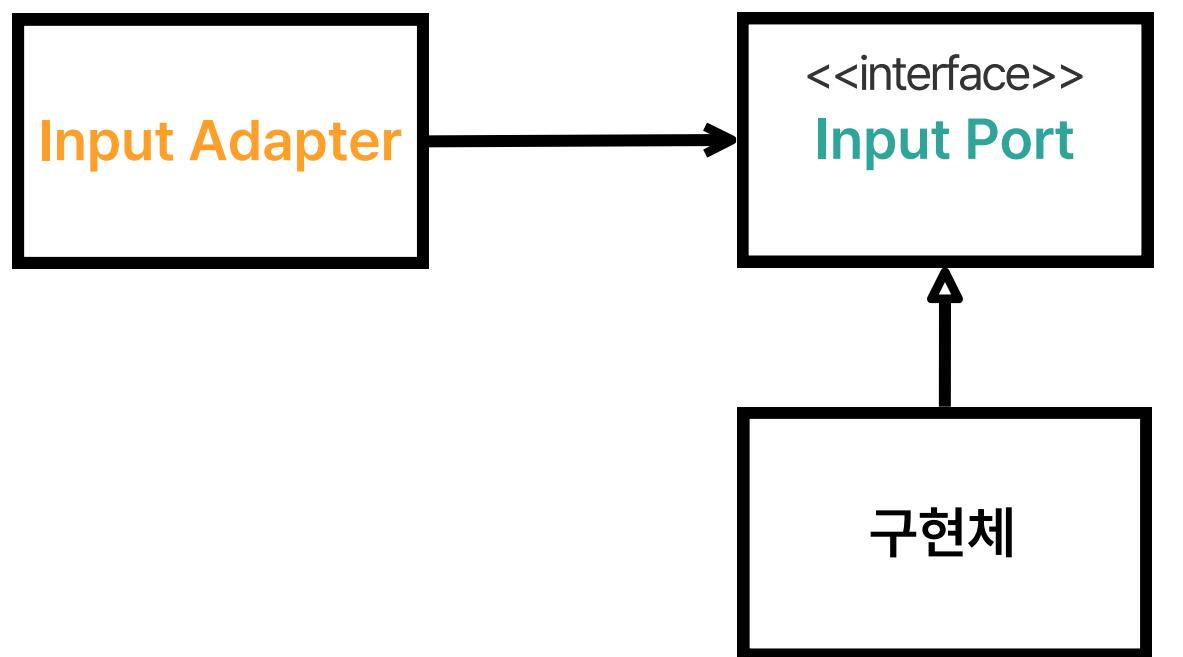
헥사고날

○ 의존성 역전 (포트-어댑터 패턴)



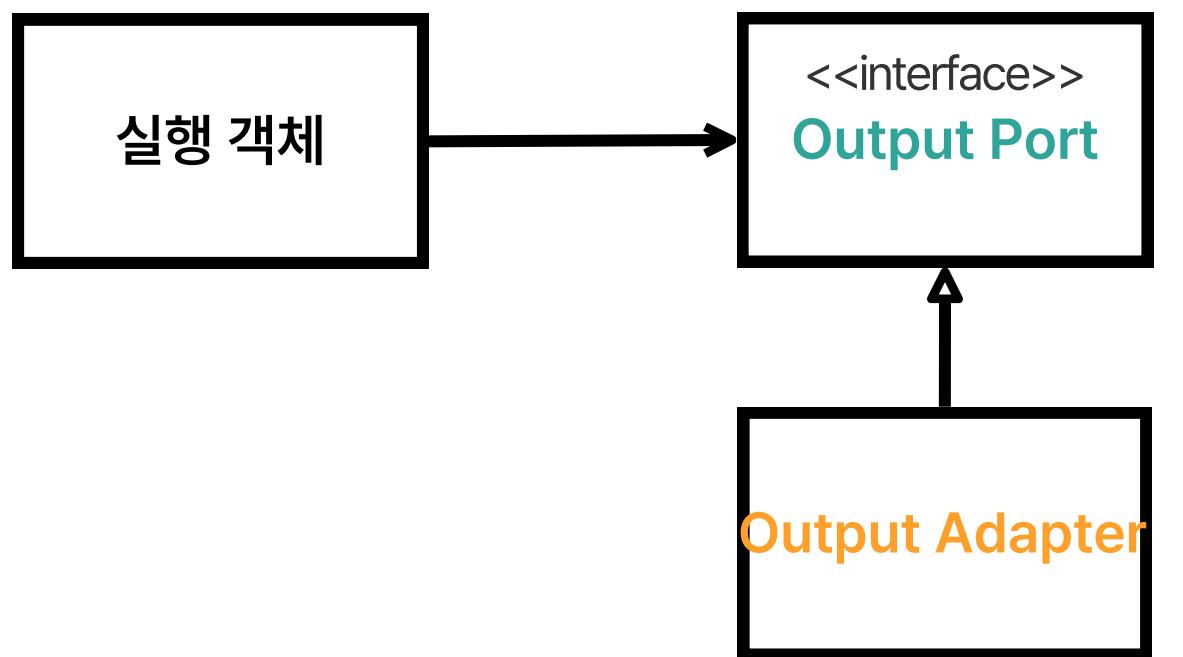
헥사고날

○ 의존성 역전 (포트-어댑터 패턴)



헥사고날

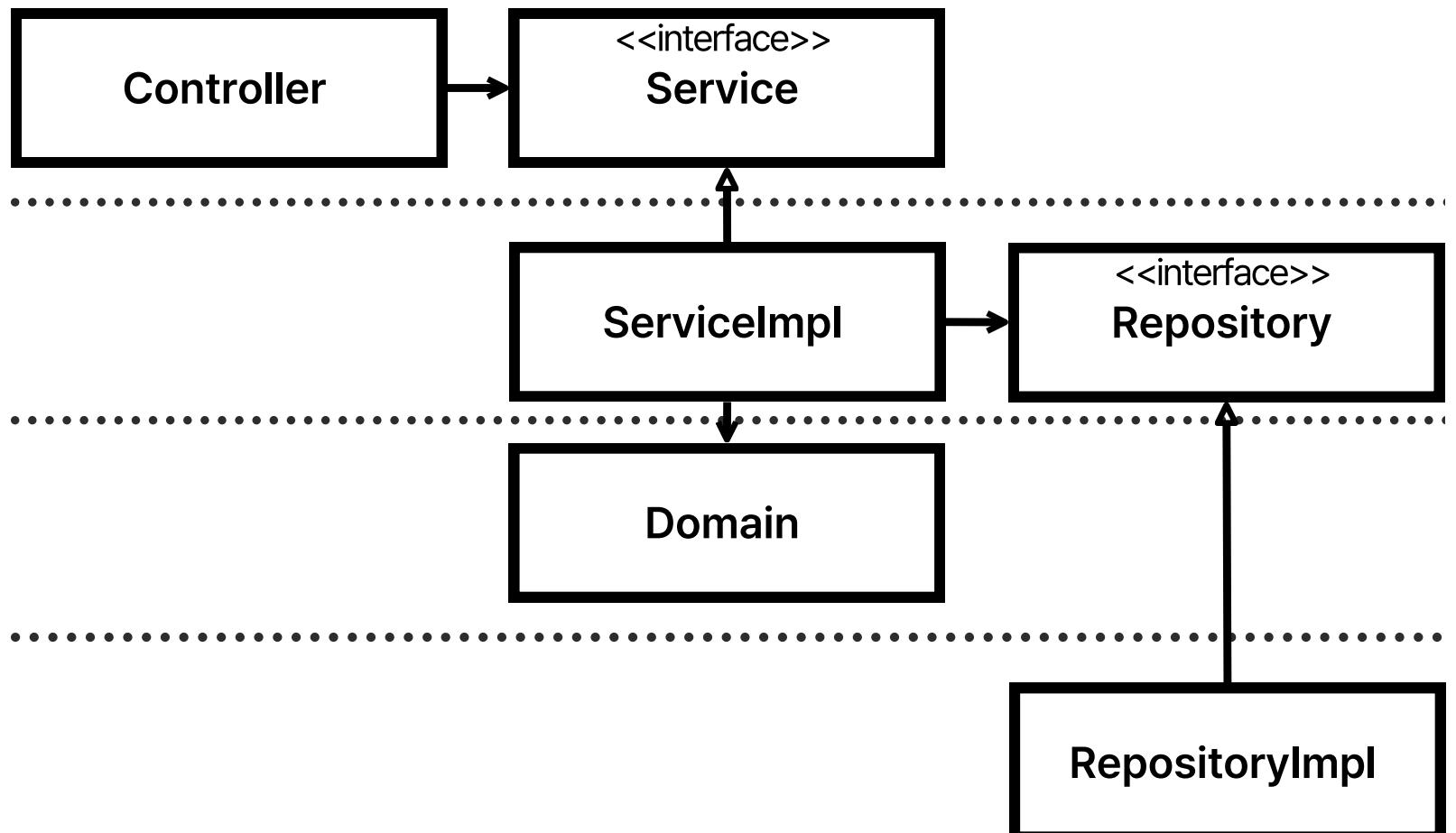
○ 의존성 역전 (포트-어댑터 패턴)



헥사고날

○ 진화하는 아키텍처

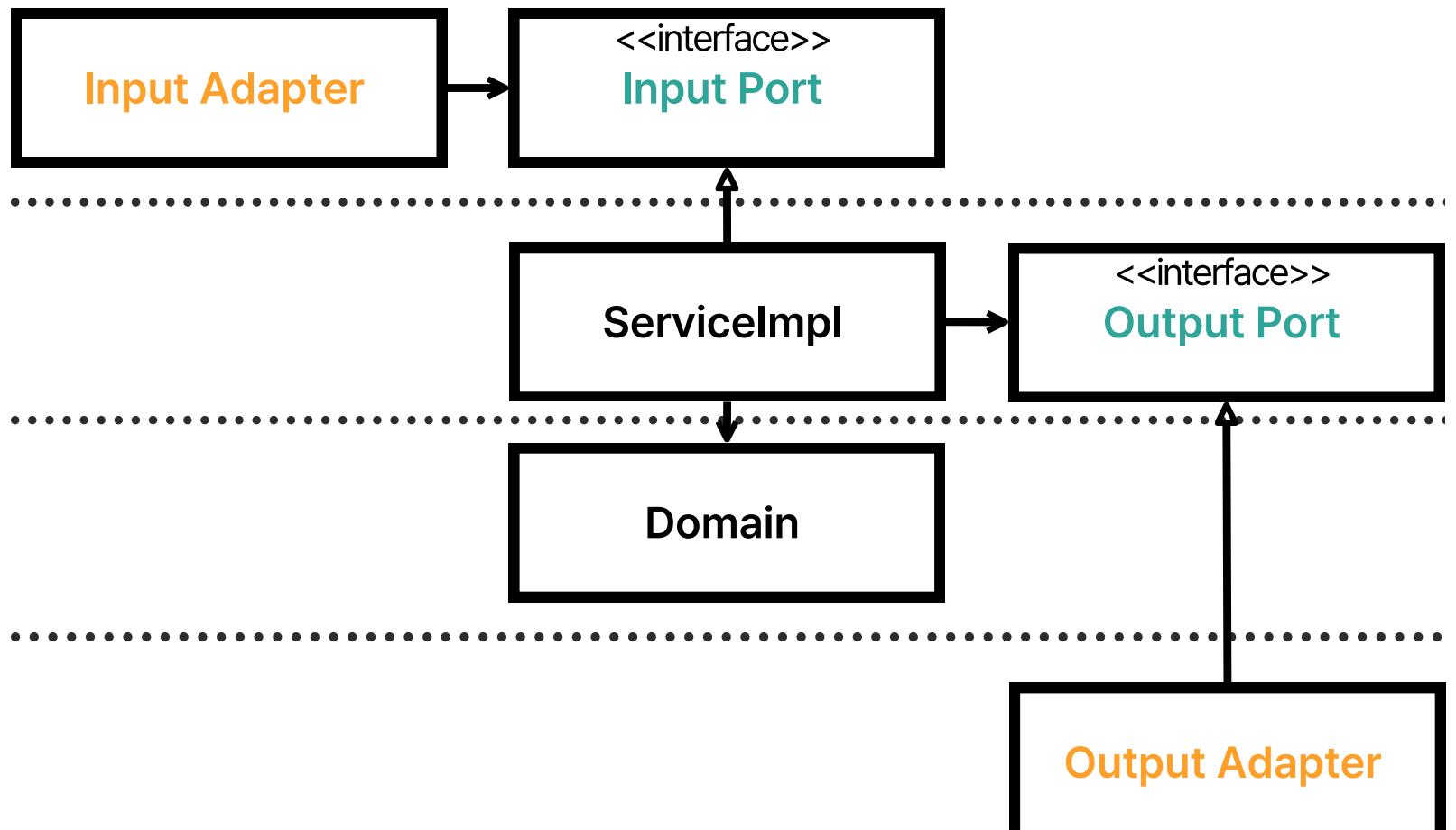
개선된 아키텍처를 해체해봅시다.



헥사고날

○ 진화하는 아키텍처

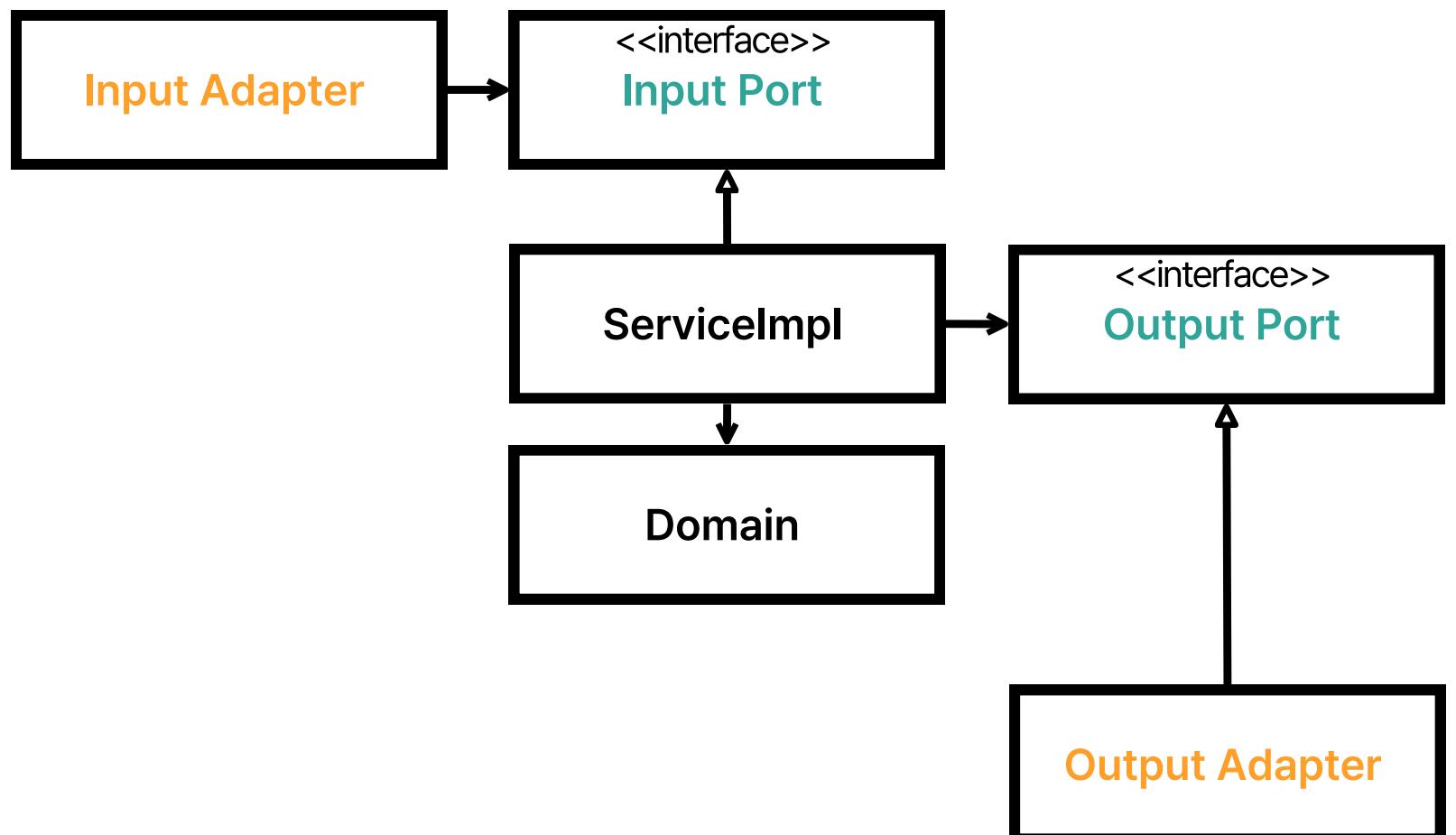
포트 - 어댑터로 구분



헥사고날

○ 진화하는 아키텍처

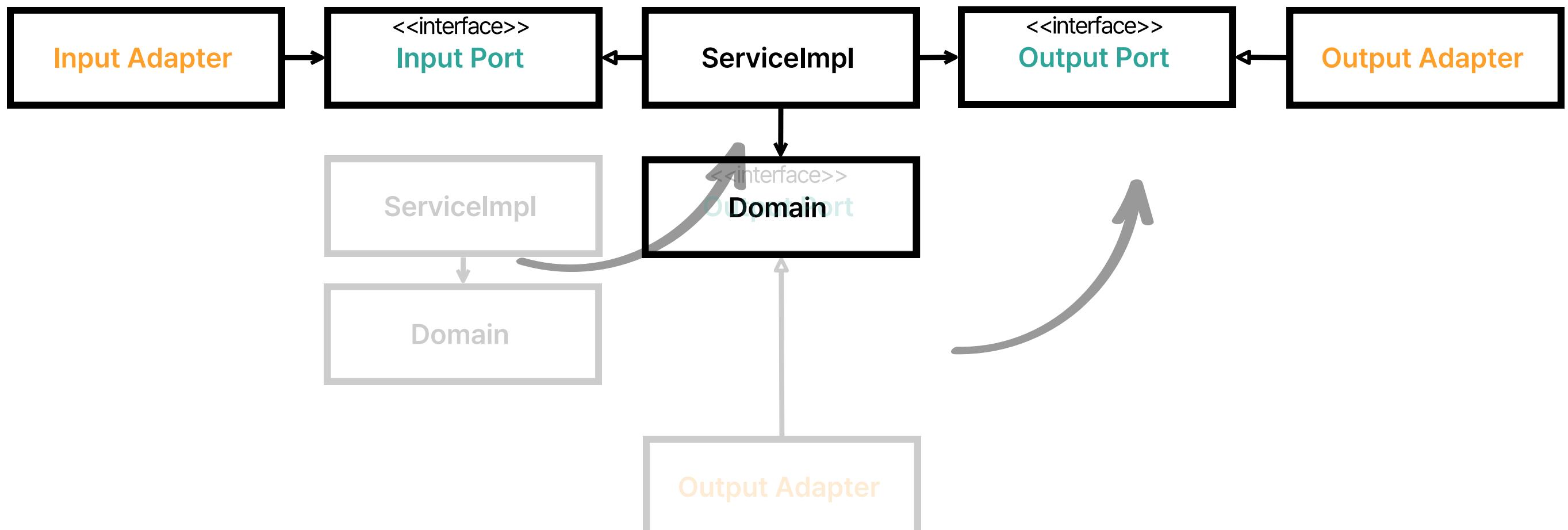
더 이상 의미 없어진 계층을 제거



헥사고날

○ 진화하는 아키텍처

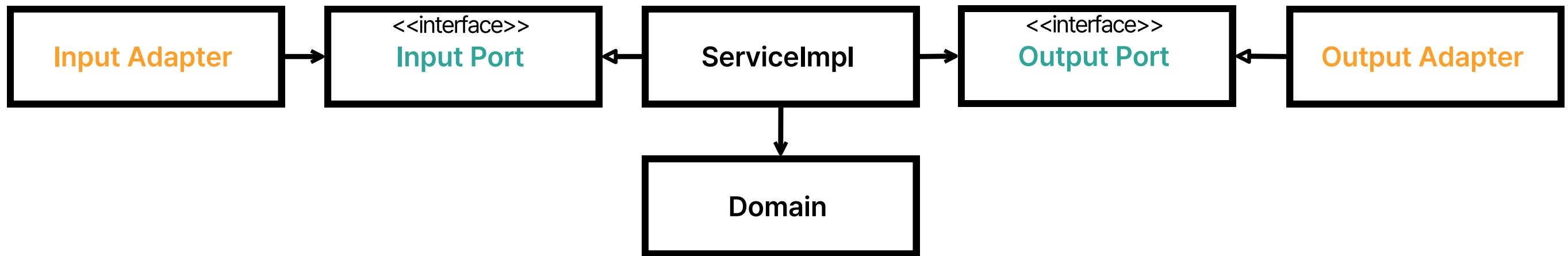
다이어그램을 재배치



헥사고날

○ 진화하는 아키텍처

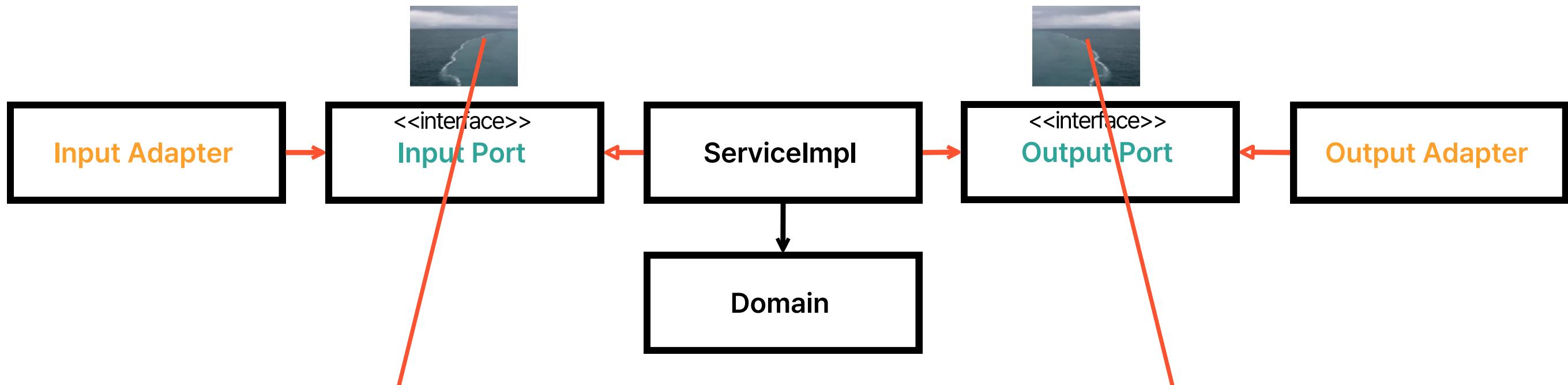
정리



헥사고날

○ 진화하는 아키텍처

경계를 파악

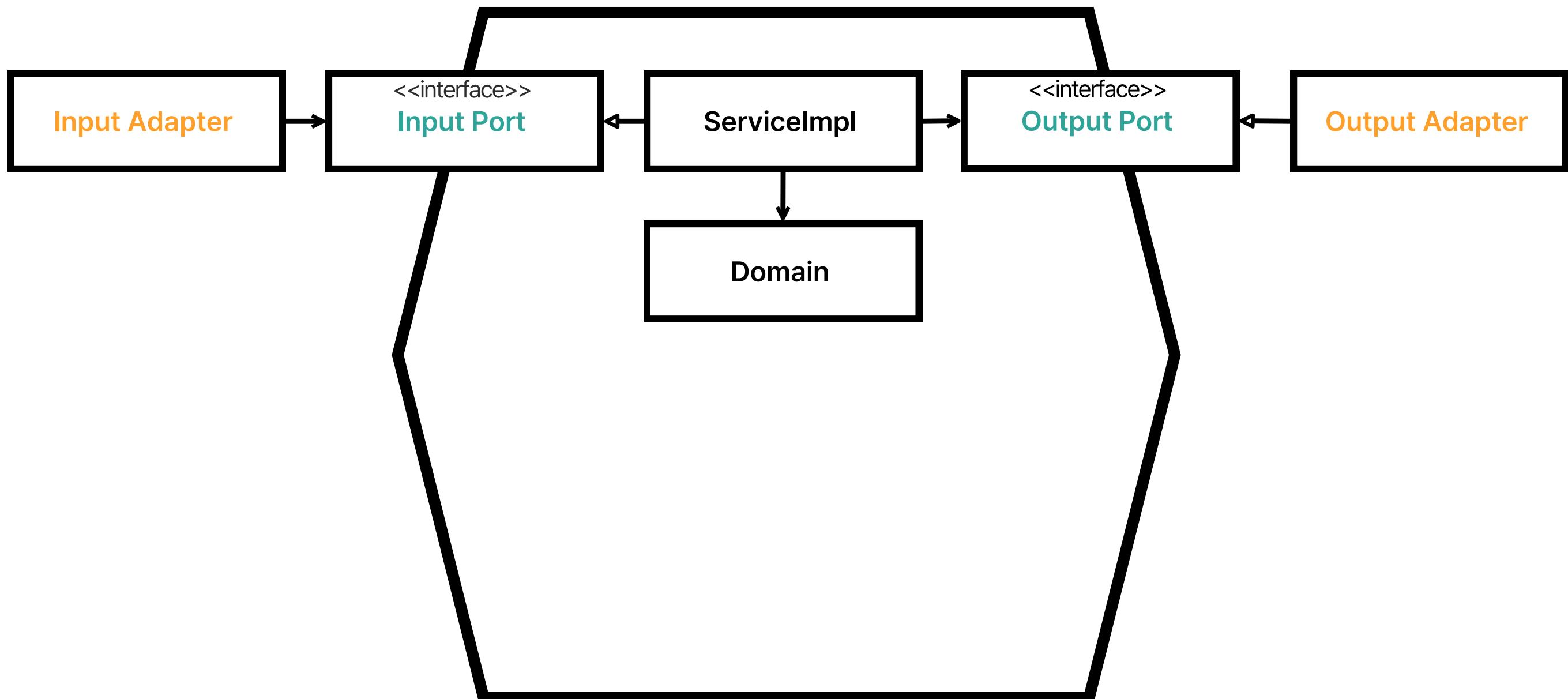


이미지 출처 : 유튜브 채널(Video Lucu dan Unik) <https://youtu.be/goC9eE1rcz8>

헥사고날

○ 진화하는 아키텍처

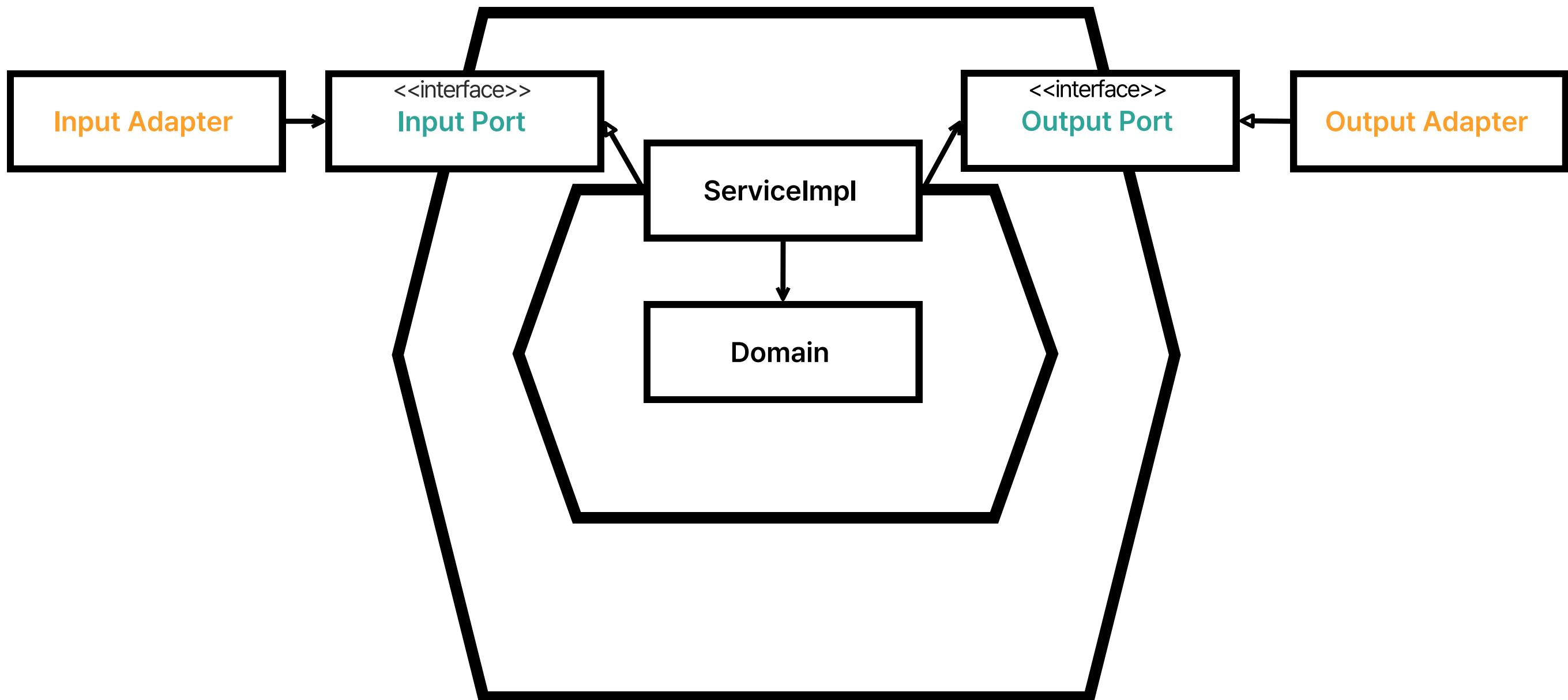
경계를 그립니다



헥사고날

○ 진화하는 아키텍처

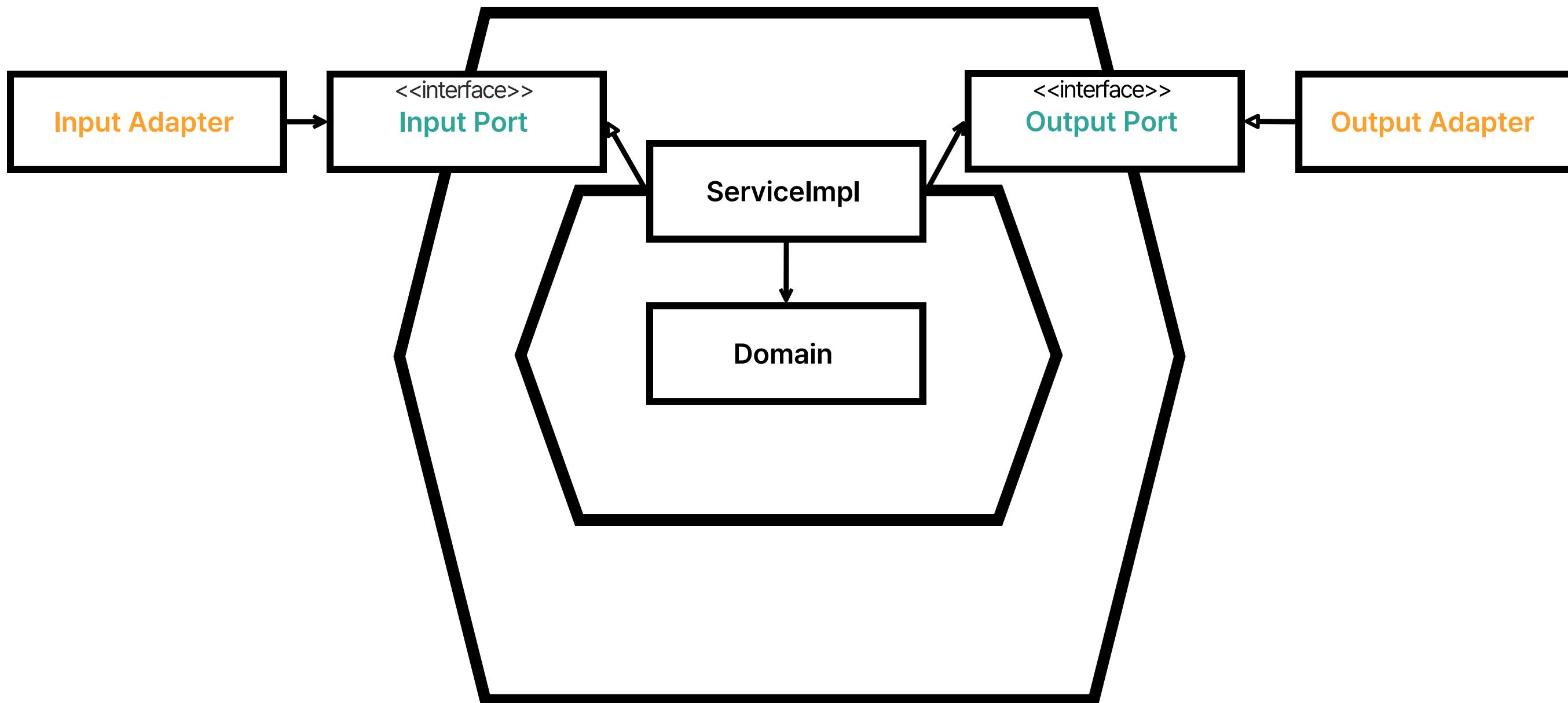
도메인 영역을 고립시킵니다



헥사고날

○ 헥사고날 아키텍처

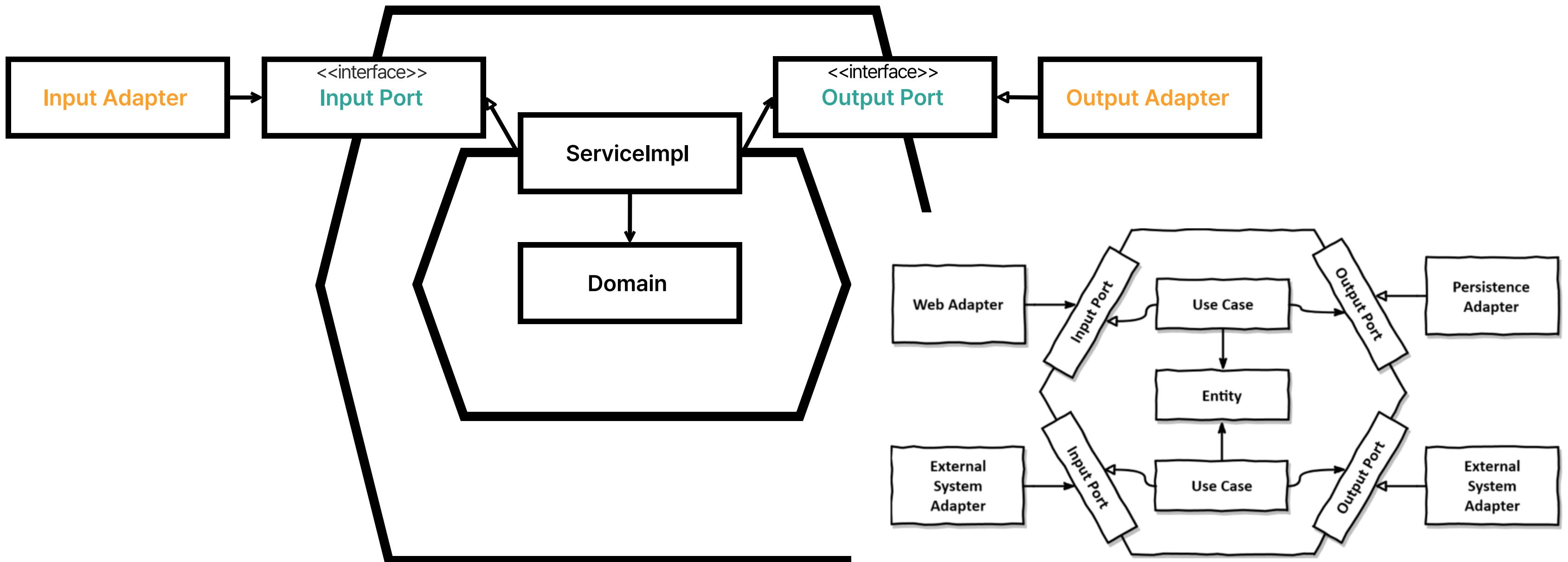
끝입니다



헥사고날

○ 헥사고날 아키텍처

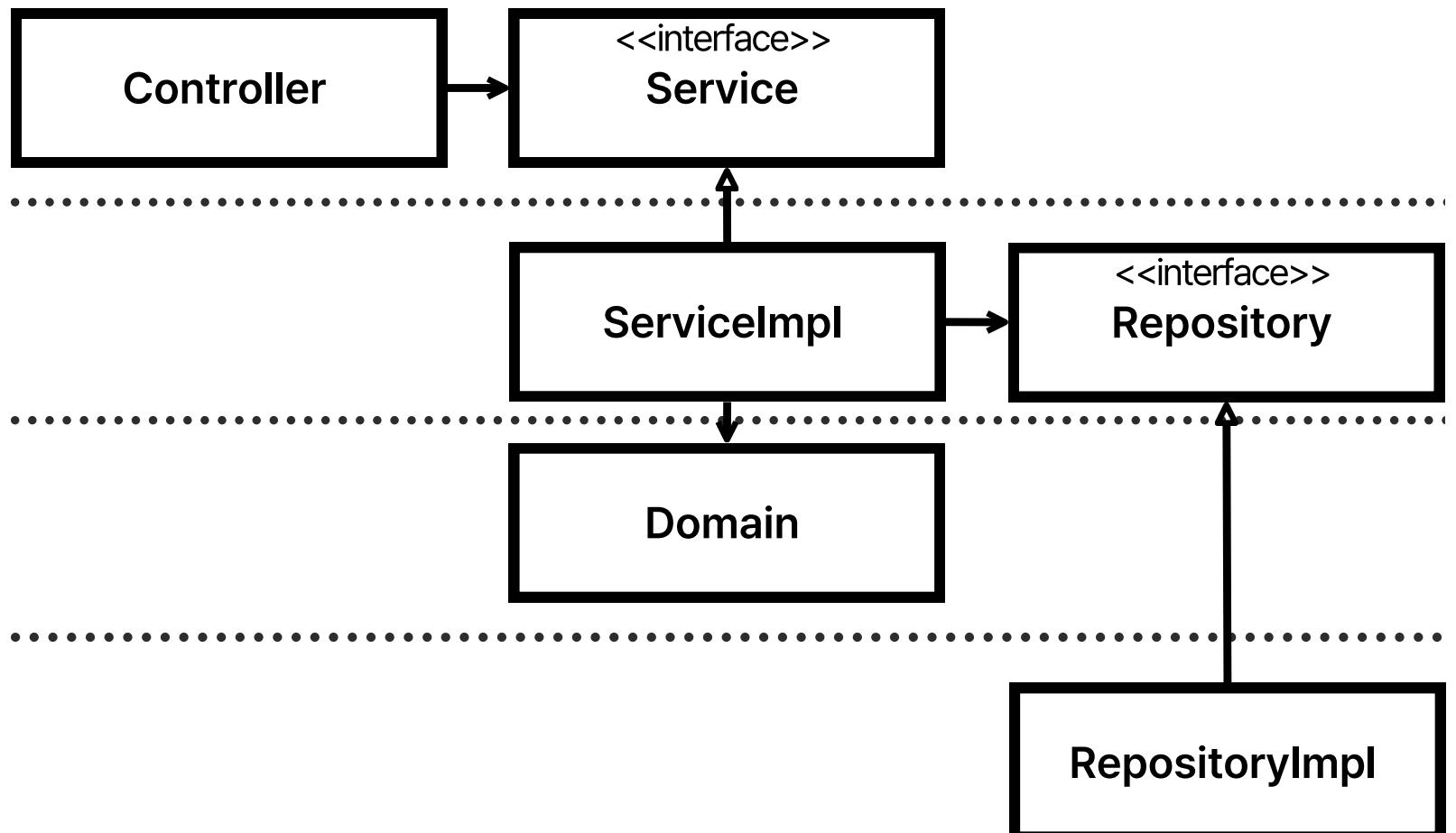
끝입니다



헥사고날

○ 헥사고날 아키텍처

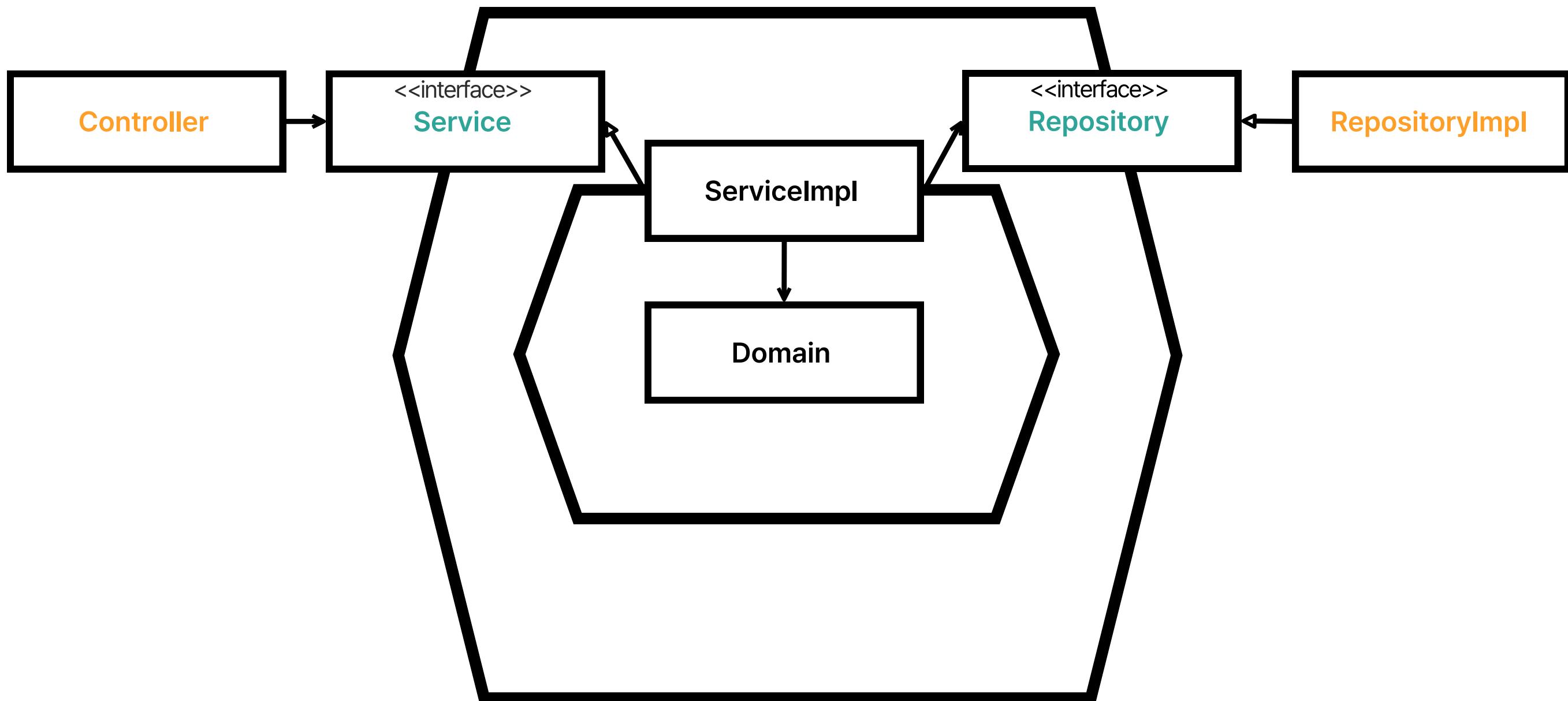
끝입니다



헥사고날

○ 헥사고날 아키텍처

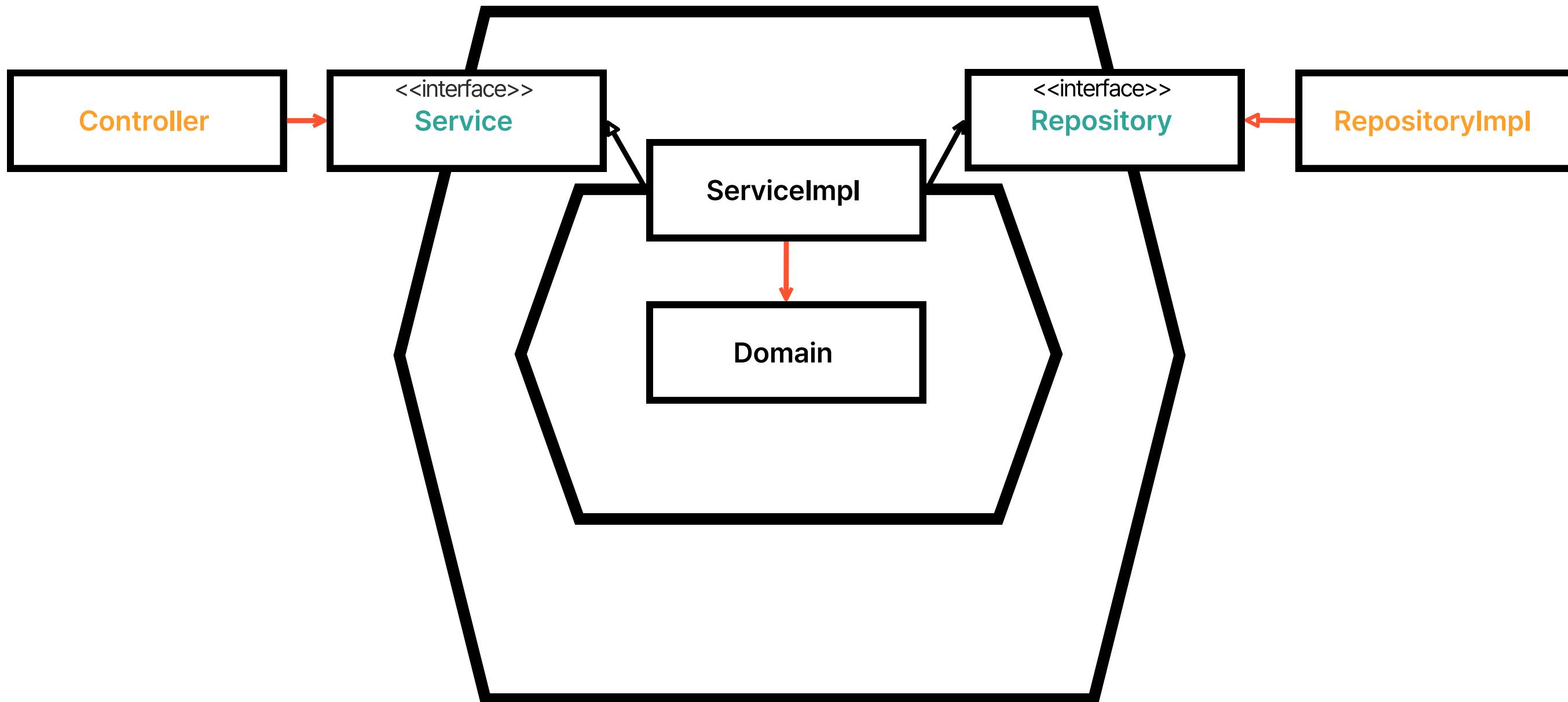
사실 지금까지 얘기한 내용들은 모두 클린 아키텍처나 헥사고날 아키텍처로 진화하는 방향을 설명 드린 것



헥사고날

○ 헥사고날 아키텍처의 장점

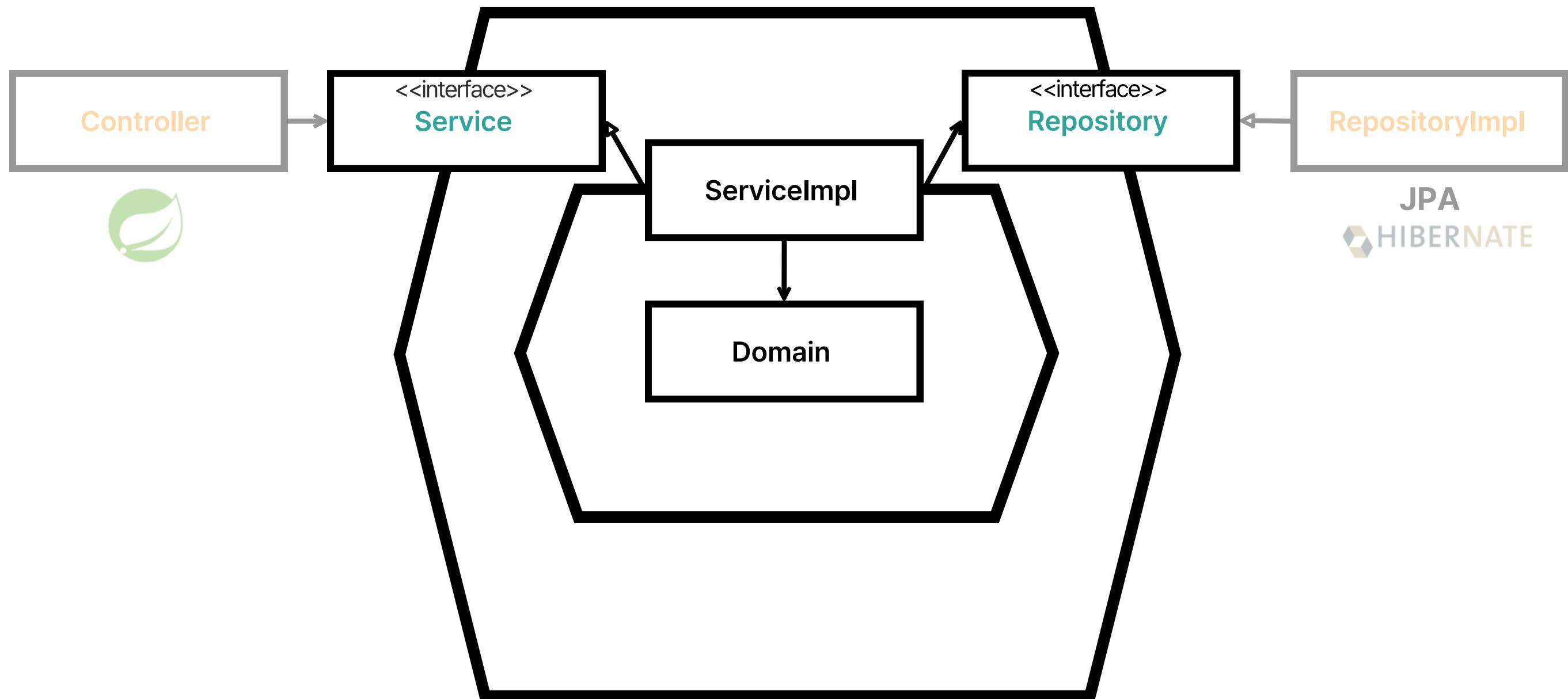
- 외부에서 도메인으로 향하는 방향이 단방향으로 유지된다



헥사고날

○ 헥사고날 아키텍처의 장점

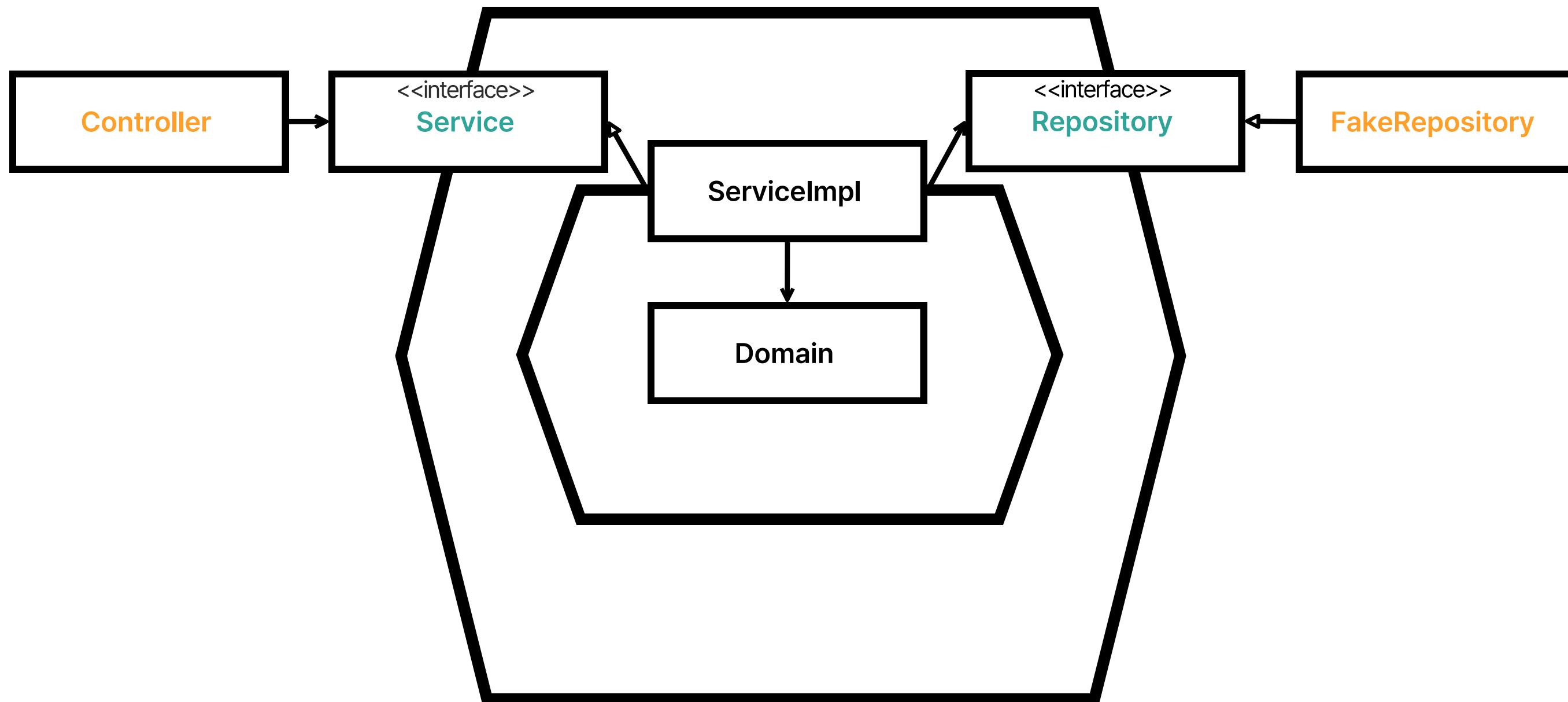
2. 외부 세계는 관심없습니다. 소프트웨어는 오직 도메인에 충실합니다.



헥사고날

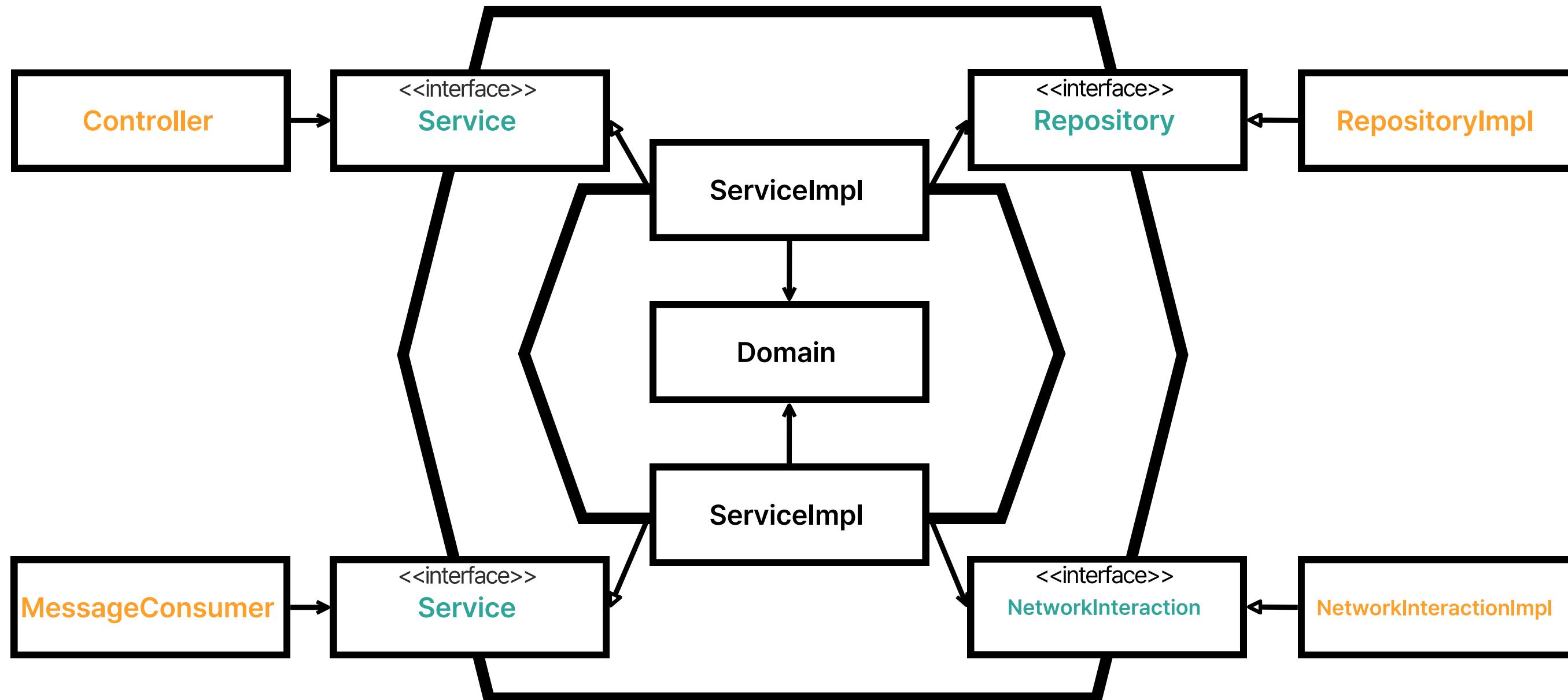
○ 헥사고날 아키텍처의 장점

3. 테스트에 유리합니다. (이유: 방향성 탐색 파트)



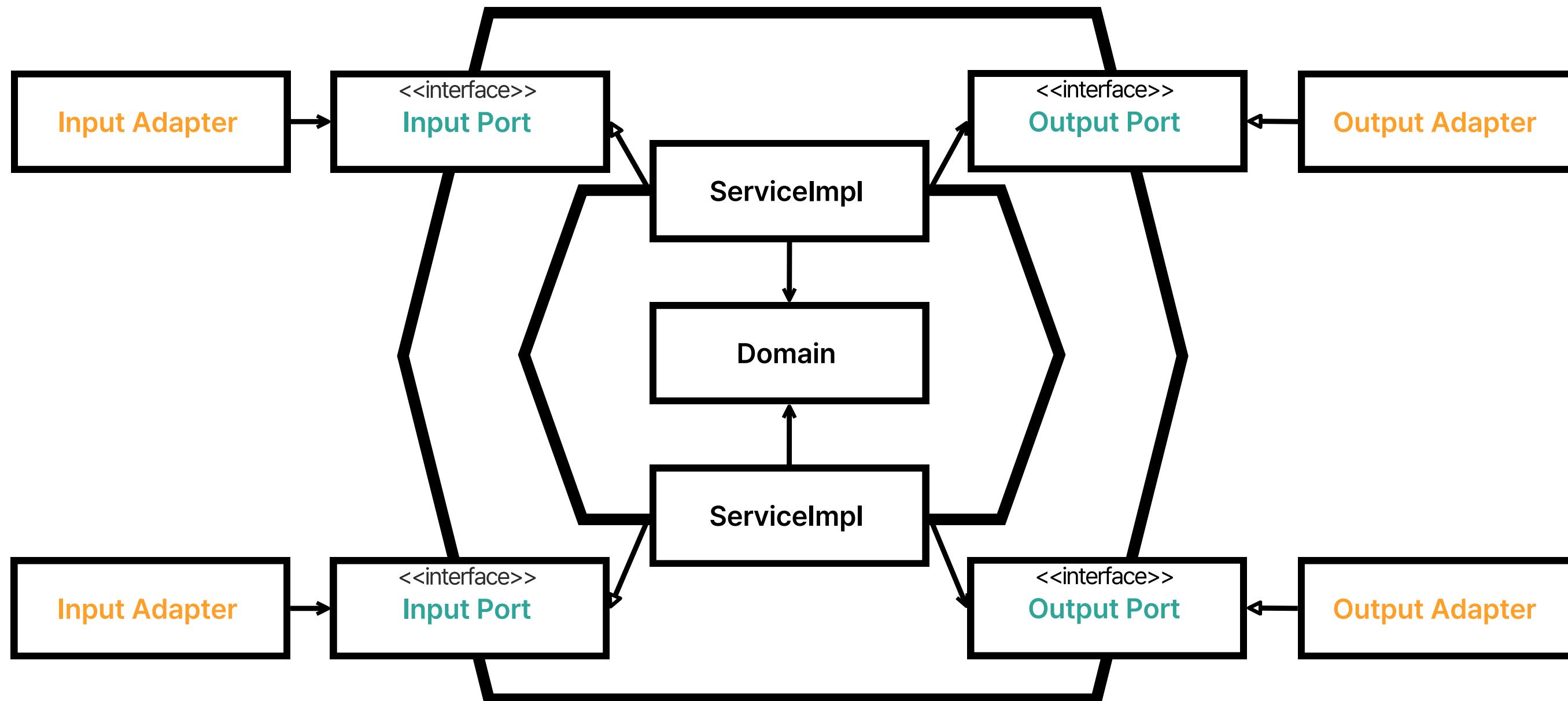
헥사고날

○ 상하 대칭으로 표현



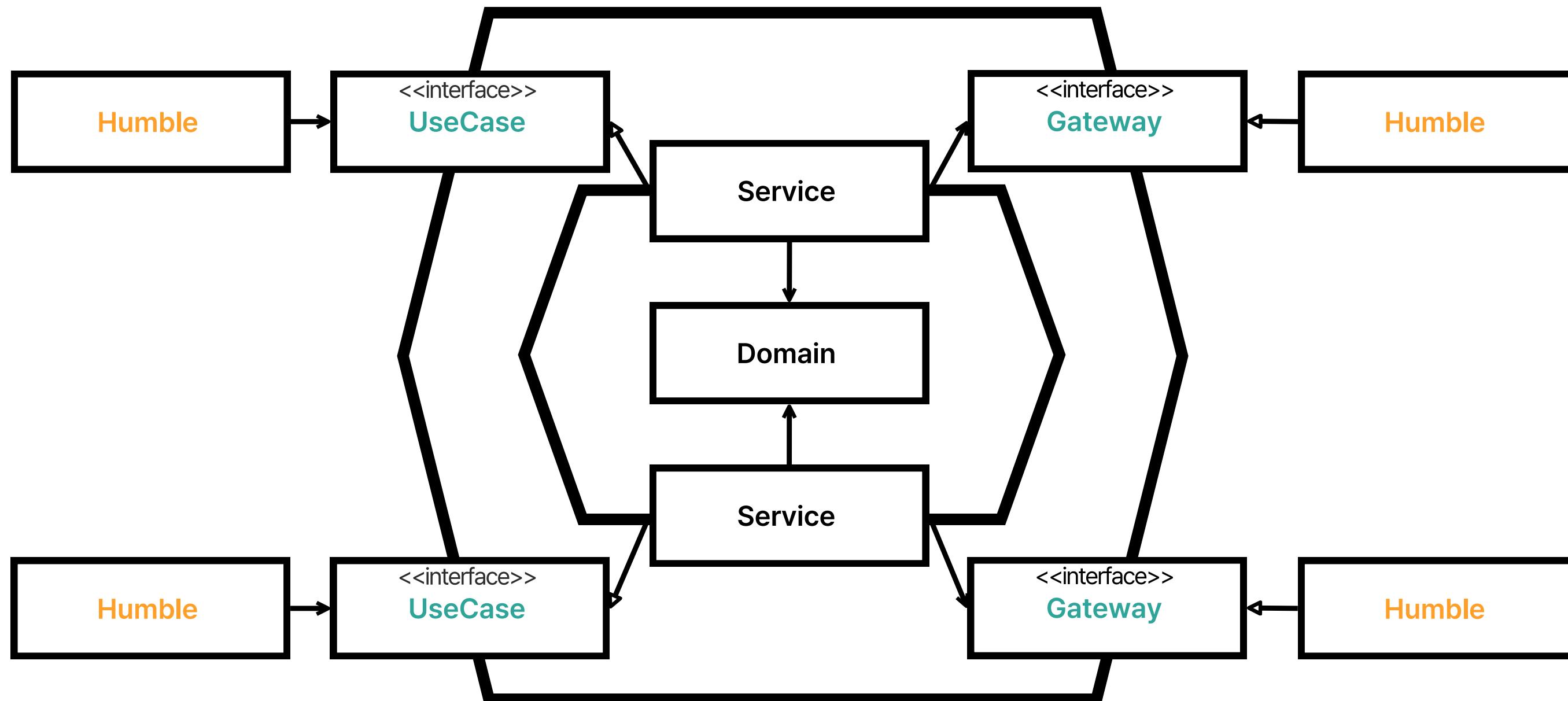
헥사고날

○ 헥사고날 아키텍처 정리



헥사고날

○ 클린 아키텍처



RETURN;

아키텍처