

6장. 점진주의: 조금씩, 조금씩, 앞으로

점진주의

- 점진적으로 일하는 것은 **가치를 점진적으로 구축하는 것**
 - 시스템의 모듈화 또는 컴포넌트화를 이용하는 방법이 있습니다.

점진주의의 정의 및 목표

점진주의는 **"점진적인 설계는 더 나은 성능을 보장하기 위해 개선될 경우 컴포넌트를 자유롭게 대체할 수 있는 모든 모듈식 설계 애플리케이션과 직접적으로 관련이 있다"**고 정의됩니다.

- 핵심 가치 전달** 점진적으로 일하는 것은 **가치를 점진적으로 구축하는 것을 의미합니다.**
- 반복과의 관계** 복잡한 시스템을 만들려면 **반복적인 접근 방식과 점진적인 접근 방식이 모두 필요합니다.** 반복적인 접근 방식이 일련의 반복을 통해 무언가를 다듬고 개선하는 것이라면, 점진적인 접근 방식은 시스템을 구축하고 서서히 릴리스하는 것입니다.
- 효율적인 진전** 점진적인 접근 방식을 사용하면 **업무를 세분화하고 단계별로 가치를 전달하여 더 빨리 가치에 도달하고 더 작고 간단한 단계로 가치를 전달할 수 있습니다.**

우주선 예시로 살펴보는 모듈성

- 아폴로 우주선의 4가지 주요 모듈
 - 서비스 모듈
 - 사령선 모듈
 - 달 탐사 모듈
 - 상승 모듈
- 모듈성의 장점
 - 개별 컴포넌트를 문제의 한 부분에 집중하도록 구축 가능합니다.
 - 각 모듈은 전체 문제의 더 큰 부분을 처리하도록 설계된 경우와 비교해 훨씬 더 단순해질 수 있습니다.
- 모듈성의 장점은 마이크로서비스나 서비스 지향 설계 같은 "컴포넌트" 기반 설계 접근 방식의 철학입니다.

효율 높은 조직 구성을 위한 비법

모듈성이 제공하는 큰 이점 중 하나는 "격리성"으로 한 모듈의 내부적인 세부 사항은 다른 모듈로부터 숨겨지고 다른 모듈과 관련이 없습니다.

모듈식 접근 방식은 팀이 훨씬 더 독립적으로 작업할 수 있게 만듭니다.

- 이는 최소한의 조율만으로 각 팀이 점진적으로 전진할 수 있습니다.
- 이러한 자유로움 덕에 모듈식 접근 방식을 완전히 수용한 조직은 전례 없는 속도로 전진하고 혁신할 수 있습니다.

결국 사람과 팀이 고품질의 창의적 작업을 실현할 수 있게 더 큰 자율성을 부여하여 분산적이고 점진적인 변화를 가져가는 것이 효율 높은 조직 구성의 비결입니다.

모듈성 (Modularity)을 통한 점진적 접근

점진주의를 가능하게 하는 가장 중요한 아이디어는 **모듈성**입니다.

- **설계의 기반** 점진적인 접근 방식은 시스템의 **모듈화** 또는 **컴포넌트화**를 이용하는 것입니다. 복잡한 시스템을 구축할 때 모듈성을 관리하는 데 매우 중요하며, 시스템을 더 작고 이해하기 쉬운 조각으로 나누어 집중할 수 있게 합니다.
- **아폴로 프로그램 사례** 아폴로 프로그램은 달에 사람을 보내기 위해 **우주선을 일련의 모듈로 나누는** '달 궤도 랑데부(LOR)'라는 임무 프로필을 만들었는데, 이는 각 모듈이 임무의 특정 부분에 집중하도록 했고, 다른 그룹이 각자의 모듈에 대해 거의 독립적으로 작업할 수 있게 했습니다.
- **소프트웨어에서의 적용** 마이크로서비스나 서비스 지향 설계 같은 컴포넌트 기반 설계 접근 방식의 철학도 문제의 한 부분을 해결하려는 목표로 문제를 여러 부분으로 나누는 모듈성에서 비롯됩니다.
- **조직 구성의 비결** 모듈식 접근 방식은 팀이 훨씬 더 **독립적으로 작업할 수 있게** 하며, 다른 팀과 최소한의 조율만으로 각 팀이 점진적으로 전진할 수 있는 자유를 부여합니다. 최적의 팀 규모는 일반적으로 **8명 이하**라는 생각이 일반적입니다.

점진주의를 적용하기 위한 실천 도구

리팩토링

- 코드를 개선하거나 최소한 안전하게 수정할 수 있는 작고 간단하며 통제된 단계로 변경할 수 있는 능력을 제공합니다.

버전제어

- 버전제어를 통해 쉽게 롤백할 수 있는 구조를 가져갈 수 있습니다.

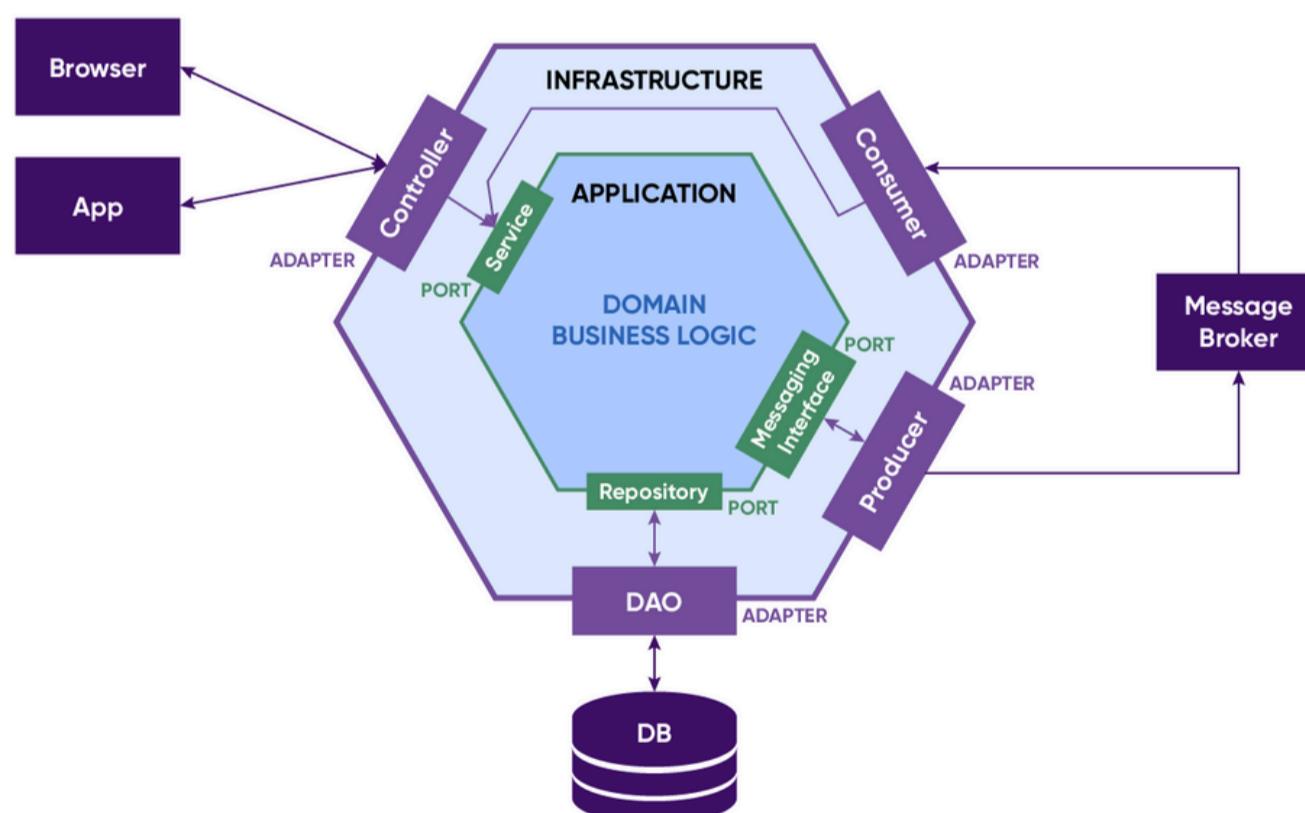
테스트

- 자동화된 테스트는 훨씬 더 강한 확신을 품고 점진적으로 앞으로 나아가도록 합니다.

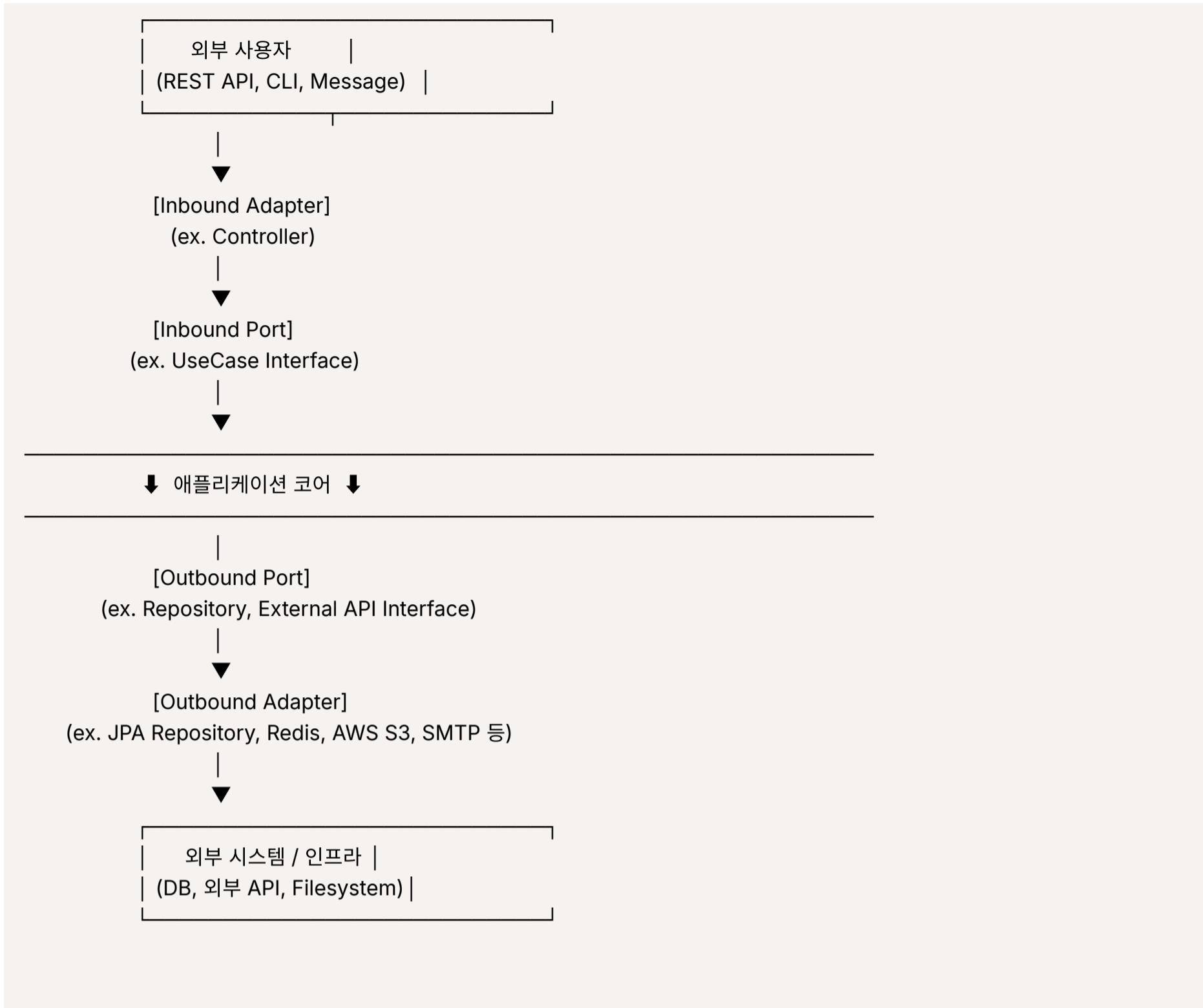
변경 부작용 최소화

포트 & 어댑터 패턴 (헥사고날 패턴)

- 결합을 떨어뜨리고 싶은 시스템의 두 컴포넌트 사이의 인터페이스 지점인 포트에서 입/출력 변환하는 분리된 코드 조각인 어댑터를 정의



 헥사고날 아키텍처 구조 (Ports & Adapters)



🛠 점진주의를 위한 실천 도구

점진주의를 성공적으로 적용하기 위해서는 기술적, 조직적인 도구들이 필요합니다.

- **리팩토링 (Refactoring)** 리팩토링은 변경의 부작용을 최소화하는 핵심 기술로, 코드를 안전하게 수정할 수 있는 **작고 간단하며 통제된 단계**로 변경할 수 있는 능력을 제공합니다.
- **테스트 (Testing)** 자동화된 테스트는 확신을 가지고 점진적으로 앞으로 나아가도록 우리를 보호하며, 더 나은 모듈식 시스템을 설계하고 실수의 폭발 반경을 제한하는 긍정적인 피드백 루프를 생성합니다.
- **포트와 어댑터 (Ports & Adapters)** 이 접근 방식은 시스템 컴포넌트 간의 경계를 고려하며, 다른 컴포넌트에 변경을 강요하지 않고 어댑터 뒤의 코드를 변경할 수 있는 자유를 얻게 해줍니다. 이는 변경의 영향을 관리하는 전략을 제공합니다.

🧠 점진적인 설계의 마인드셋

복잡한 시스템은 어떤 천재적인 창조자의 머리에서 온전히 만들어지지 않으며, 탐구하고 발견하는 과정을 통해 점진적으로 설계해야 합니다.

- **변화에 대한 수용** 좋은 공학 원칙의 기본은 **이해도가 깊어짐에 따라 코드를 자유롭게 변경하고 생각을 바꿀 수 있는 방식**으로 작업하는 것입니다. 이는 **실행 취소가 쉬운 작은 단계**로 작업할 필요가 있다는 것을 의미합니다.
- **방어적 접근 방식** 점진적인 작업은 **복잡성을 관리하는 기술**을 사용해 너무 큰 피해를 주는 실수를 하지 않도록 스스로를 보호하는 방어적 접근 방식입니다.

점진주의는 **문제를 해결하는 과정에서 본질적으로 제한된 접근 방식인 폭포수 방식**과는 달리, 모든 답을 알기 전에 작업을 시작하고 점진적으로 배우고 발전할 수 있는 **개방적이고 무한한 프로세스**를 가능하게 합니다.