서버 프로그램과 인터페이스 구현

모듈을 개발했을 때의 응집도와 결합도

응집도 (강할수록 좋음)

- 기능적 응집도
- ↓
- 순차적 응집도
- ↓
- 교환적 응집도
- ↓
- 절차적 응집도
- ↓
- 시간적 응집도
- ↓
- 논리적 응집도
- ↓
- 우연적 응집도

기순교절시논우

순차적: 출력이 다음코드의 입력으로 들어감

절차적: 관련된 기능이 특정 순서로 실행

결합도 (강할수록 안좋음)

- 내용 결합도
- ↓
- 공통 결합도
- ↓
- 외부 결합도
- ↓
- 제어 결합도
- ↓
- 스탬프 결합도
- ↓
- 자료 결합도

내공외제스자

공통 모듈 구현 절차

DTO/VO → SQL → DAO → Service → Controller → View

- **DTO/VO**: 데이터를 전송하고 저장을 위한 객체를 설계한다.
- **SQL**: 데이터베이스 접근을 위한 쿼리 작성.

- **DAO**: 데이터베이스와의 상호작용을 담당하는 객체를 설계.
- **서비스**: DAO를 호출하여 데이터 처리 후 DTO/VO를 사용하여 데이터를 주고 받는다.
- **컨트롤러**: View와 서비스의 중간 단계 역할 (HTTP 요청/응답 처리).
- **뷰(View)**: 사용자(고객)에게 보이는 화면을 구현.

FAN-IN과 FAN-OUT

소프트웨어에서 모듈이 많을 때 서로 IN/OUT을 표기

특정 모듈에서의 FI / FO?

- **Fan-in**: 모듈에서 들어오는 개수
- **Fan-out**: 모듈에서 나가는 개수

(여러 모듈이 특정 모듈을 사용하고, 특정 모듈은 다른 하나의 모듈을 호출)

그 외 용어를 배워보자

- **배치(Batch) 프로그램**: 정기적으로 반복 수행하거나, 규칙에 의해 일괄적으로 수행되는 것
- **미들웨어**: 서로 다른 소프트웨어 응용 프로그램이나 구성 요소를 연결하는 소프트웨어
  (예: JDBC, RabbitMQ, Apache Tomcat 등)

객체지향 설계원칙 5개 (SOLID)

1. **SRP (Single Responsibility Principle)** : 단일 책임 원칙
2. **OCP (Open/Closed Principle)** : 개방-폐쇄 원칙
3. **LSP (Liskov Substitution Principle)** : 리스코프 치환 원칙
4. **ISP (Interface Segregation Principle)** : 인터페이스 분리 원칙
5. **DIP (Dependency Inversion Principle)** : 의존 역전 원칙

인터페이스를 어떻게 설계할까?

데이터를 주고 받을 때, 어떤 식으로 노드를 구성할 것인가?

큰 규모의 회사에서 사용하는 방식:

EAI (Enterprise Application Integration)

1. **Point-to-Point**: 각 애플리케이션이 다른 애플리케이션과 연결되어 데이터를 주고 받는 방식
2. **Hub-and-Spoke**: 중앙 허브가 모든 데이터와 요청을 중개
3. **Message Bus**: 메시지 버스를 통해 애플리케이션 간 통신 관리
4. **Hybrid**: 위 방식들을 섞어 사용하는 방식 (내부: H&S, 외부: MB)

ESB (Enterprise Service Bus)

기업 애플리케이션 통합을 위한 아키텍처 패턴으로, 서비스 지향 아키텍처(SOA)를 지원.
통신을 효율적이고 조정 가능하도록 관리하기 위해 설계된 중간 계층 소프트웨어.

인터페이스 관련 용어

1. **JSON**: 데이터를 저장하고 전송하기 위한 경량 데이터 교환 형식. 속성-값 쌍으로 구성.
2. **REST**: HTTP 기반의 URI 자원 표현 + CRUD 연산 (GET, POST, PUT, DELETE 등).
3. **직렬화 / 역직렬화**: 객체를 전송할 수 있는 형식으로 변환 / 다시 객체로 복원.
4. **AJAX**: 웹페이지를 동적으로 갱신할 수 있는 비동기 통신 기술.
5. **XML**: 데이터를 구조적으로 표현하기 위한 마크업 언어.
6. **API**: 애플리케이션 간의 상호작용을 정의하는 인터페이스.