

61. Spring Framework 8.0 출시!

- ① 'spring-webmvc' 확장 버전을 출시한 이후

✓ (spring 6.x → Jakarta EE 11 & 9.x jakarta.* Tomcat 10.x
 spring 5.x → JavaEE (8.x) = Jakarta EE (8.1) jaxws.* Tomcat 9.x)

- ② 스트링 애플리케이션으로 뷰를

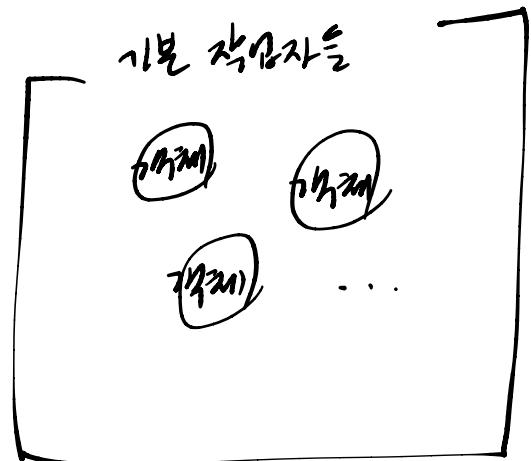
- ③ " 웹으로 뷰를

- ④ " Application 뷰로 AppConfig 123

- ⑤ " 콘트롤러로 뷰를

- ⑥ " IoC 컨테이너로 뷰를

* Spring Framework



이런 기능을 처리할 때
적용자가 등록되어 있으려면
(인증)
그 적용자를 실행 (마이그로우)하면
처리된다.
없으면 예외를 던져서 기능을 무시한다.

기능 추가?

그 기능을 수행할 적용자를 등록
(적용자)



- ① 적용 적용자를 사용해서 등록
- ② (액션레이아웃) 설정을 통해
(XML)

* 요청 자세히 봐서 요청 헤더의 자세히 봐

<form>

```
<input name="email"/>  
<input name="password"/>  
<button>로그인</button>
```

POST로

</form>

요청 헤더

요청 자세히 봐

email=aaa@test.com & password=1111

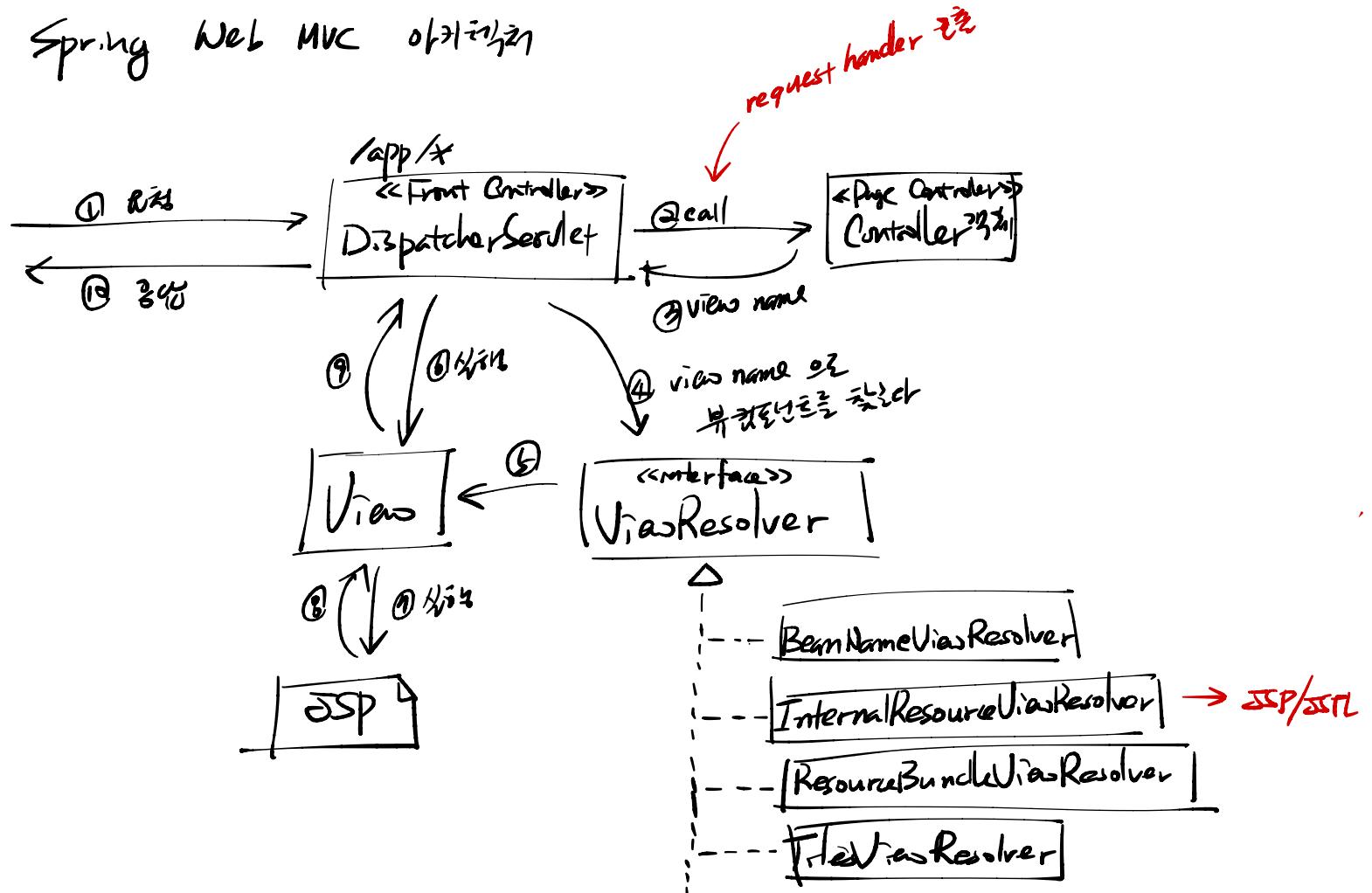
String

login(String email, String password) {

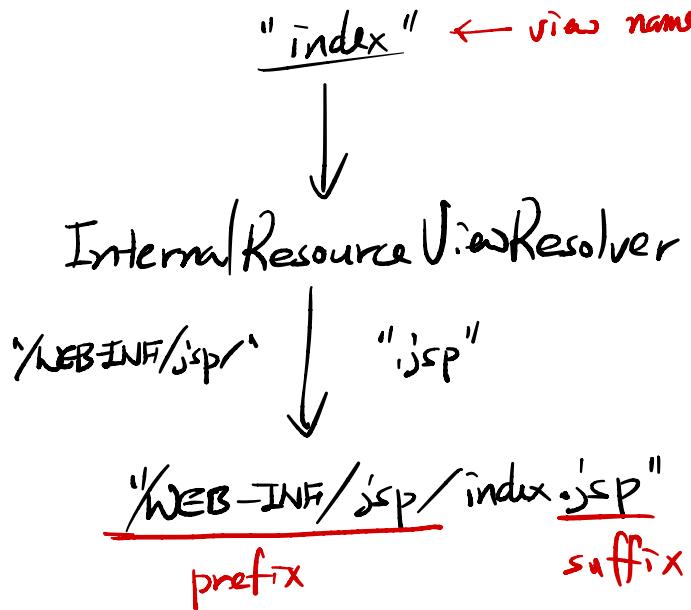
요청 헤더의 자세히 봐

}

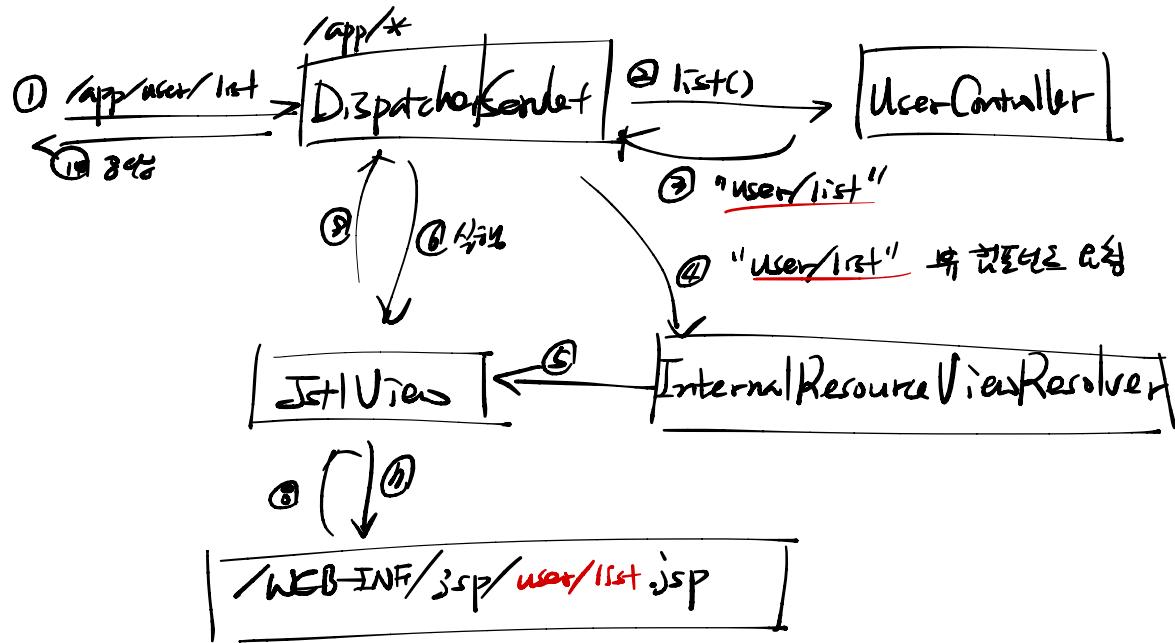
* Spring Web MVC 07/27/21



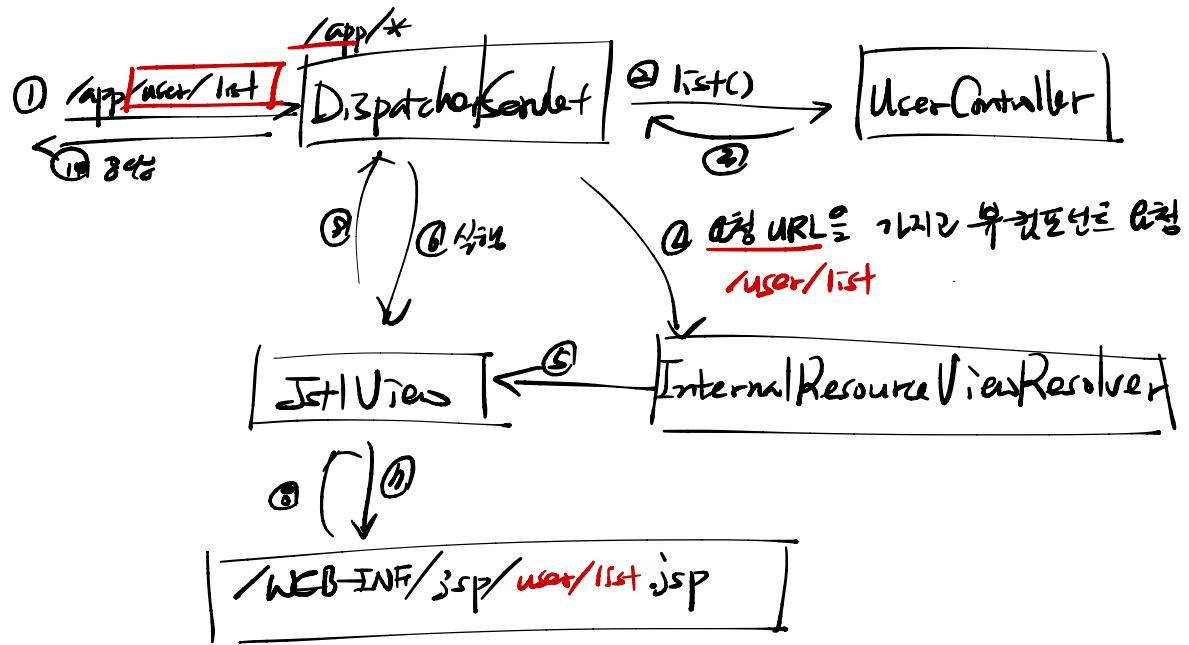
- * Internal/Resource ViewResolver



* view name %*% can

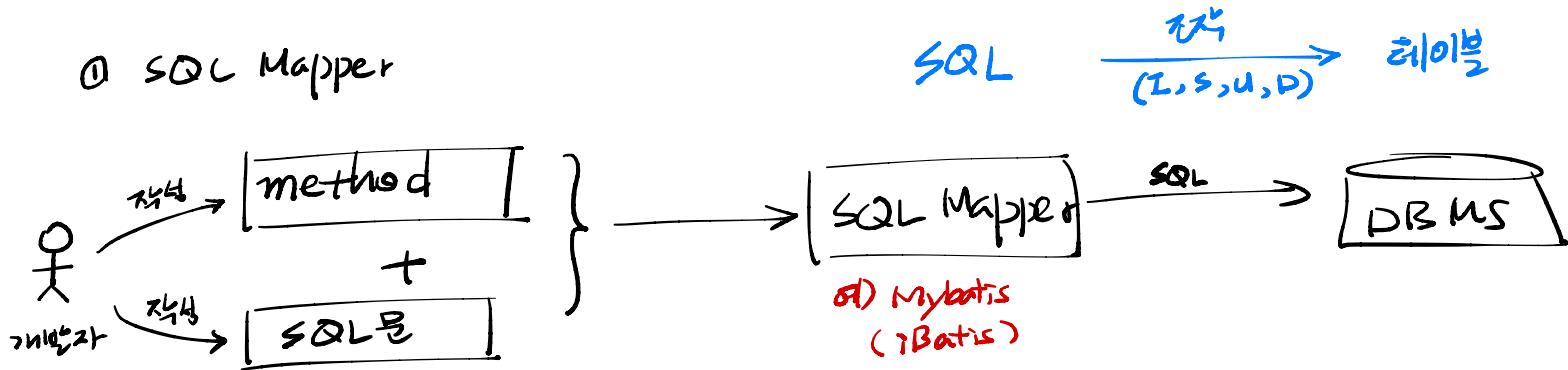


* view name $\stackrel{\text{보통}}{\rightarrow}$ controller

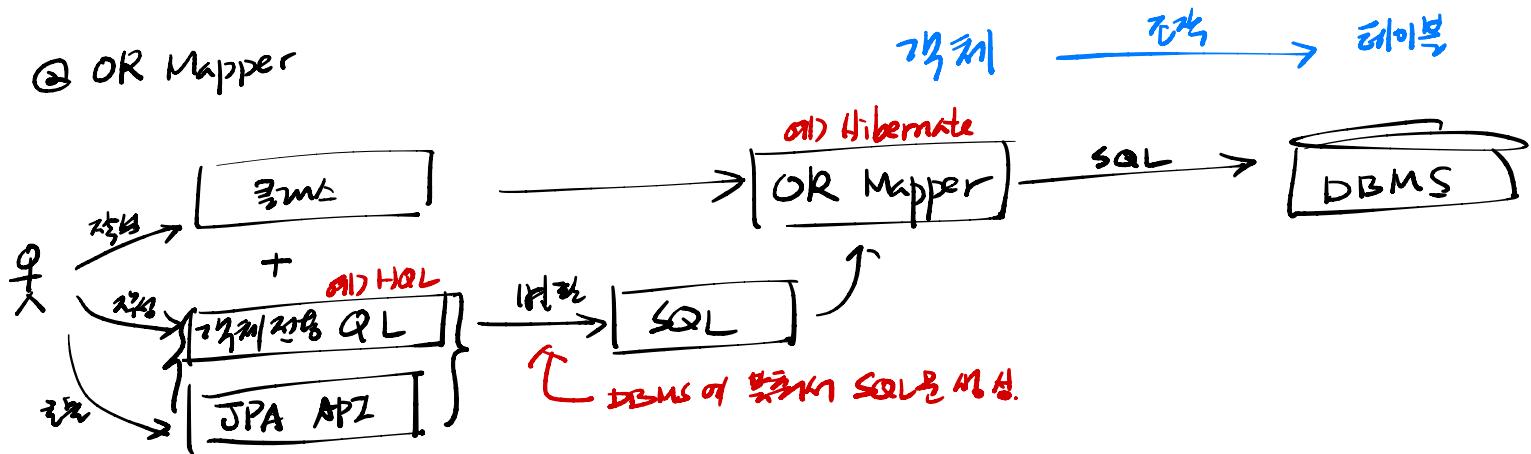


* SQL Mapper vs OR Mapper

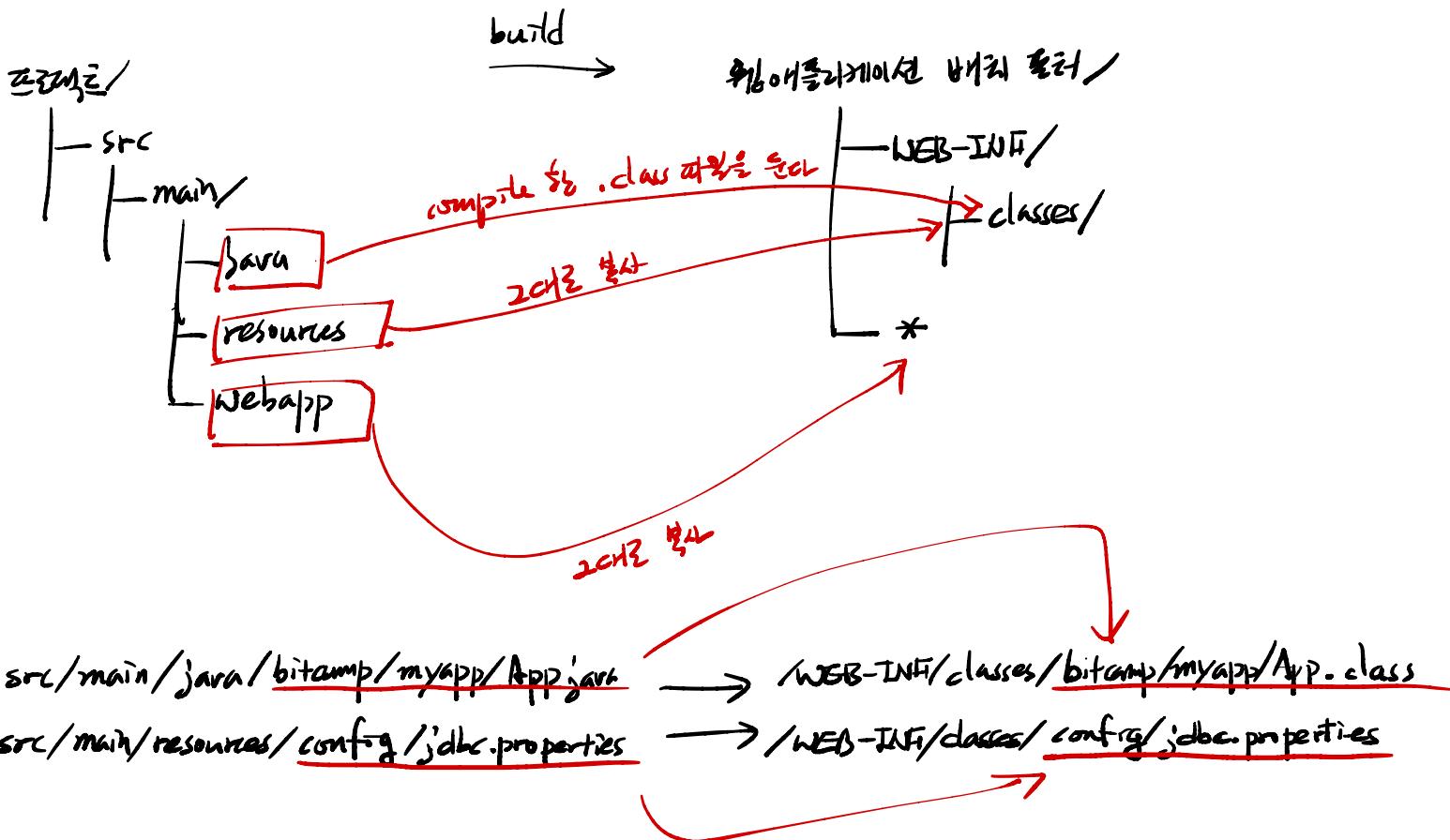
① SQL Mapper



② OR Mapper



* build et classpath



* DaoFactory

↑
생성
DAO

SqISessionTemplate

↑
Mybatis
가져온
DAO

[BoardDao]

insert()

[DaoFactory]

↳ sqISession.insert("BoardDao.insert", -);

[SqISessionTemplate]

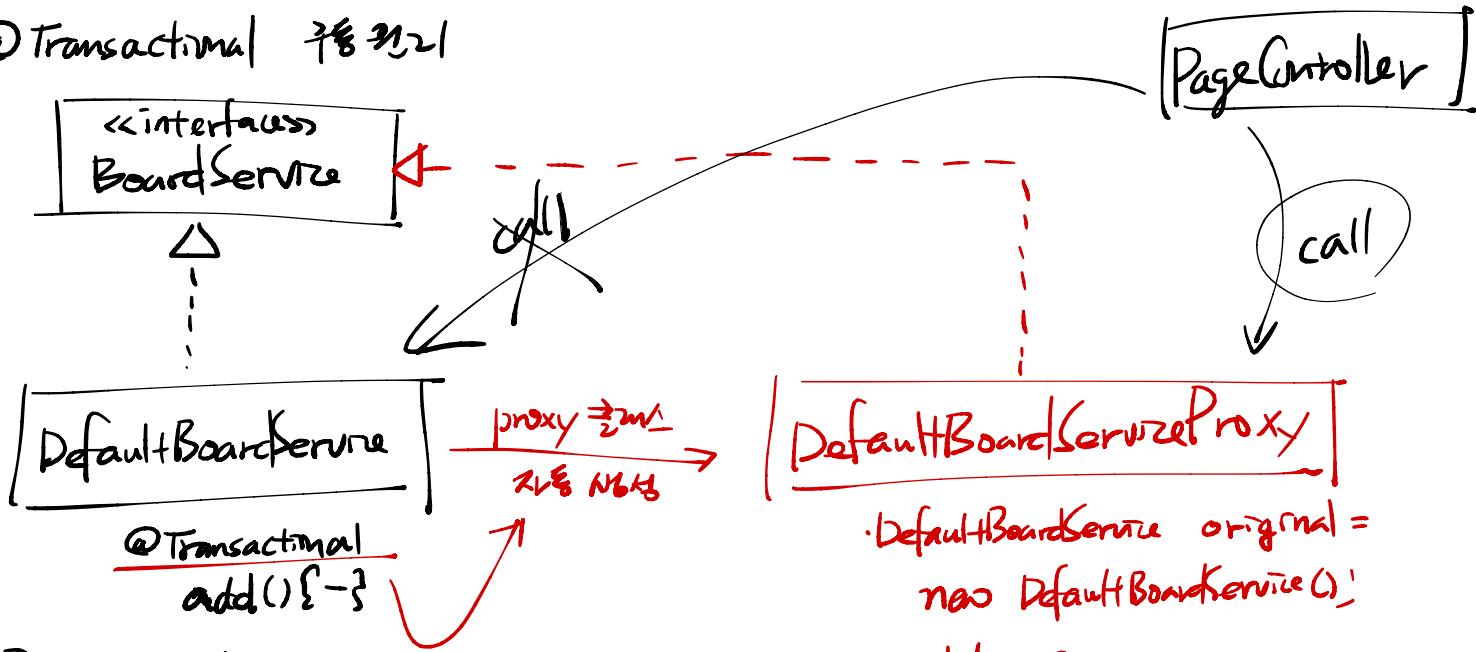
↳ sqISession.insert(

"bifrap-myapp.dao.BoardDao.insert",
-);

SQL
실행
하기위한
SQL문
생성
하기위한
Template

MyBatis
Template.

* @Transactional 78 31, 21



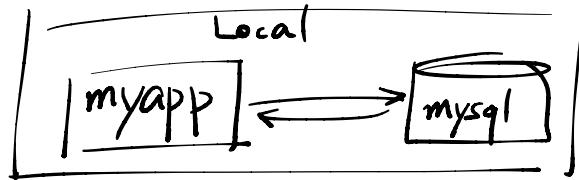
* AOP의 특징, 예제

↳ 기존 코드를 대체하지 않고
추가 기능을 적용하는 방법
↳ proxy 구현은 어렵지만 ("proxy를 만들기 어렵다")

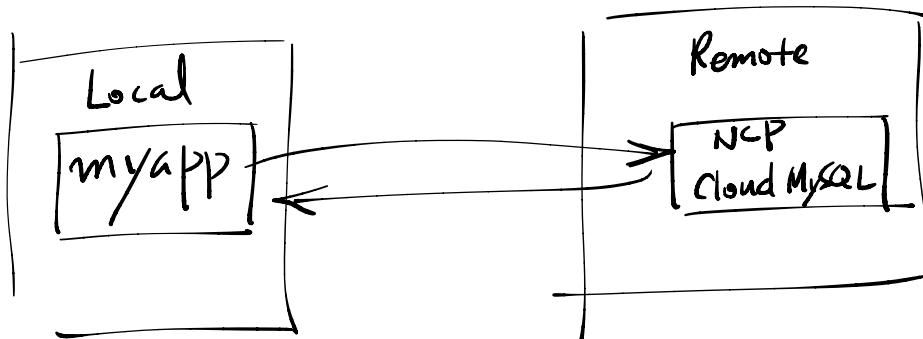
. **add () {**
 try {
 original.add();
 txManager.commit();
 } catch {
 txManager.rollback();
 }

62. Naver Cloud Platform 사용 예

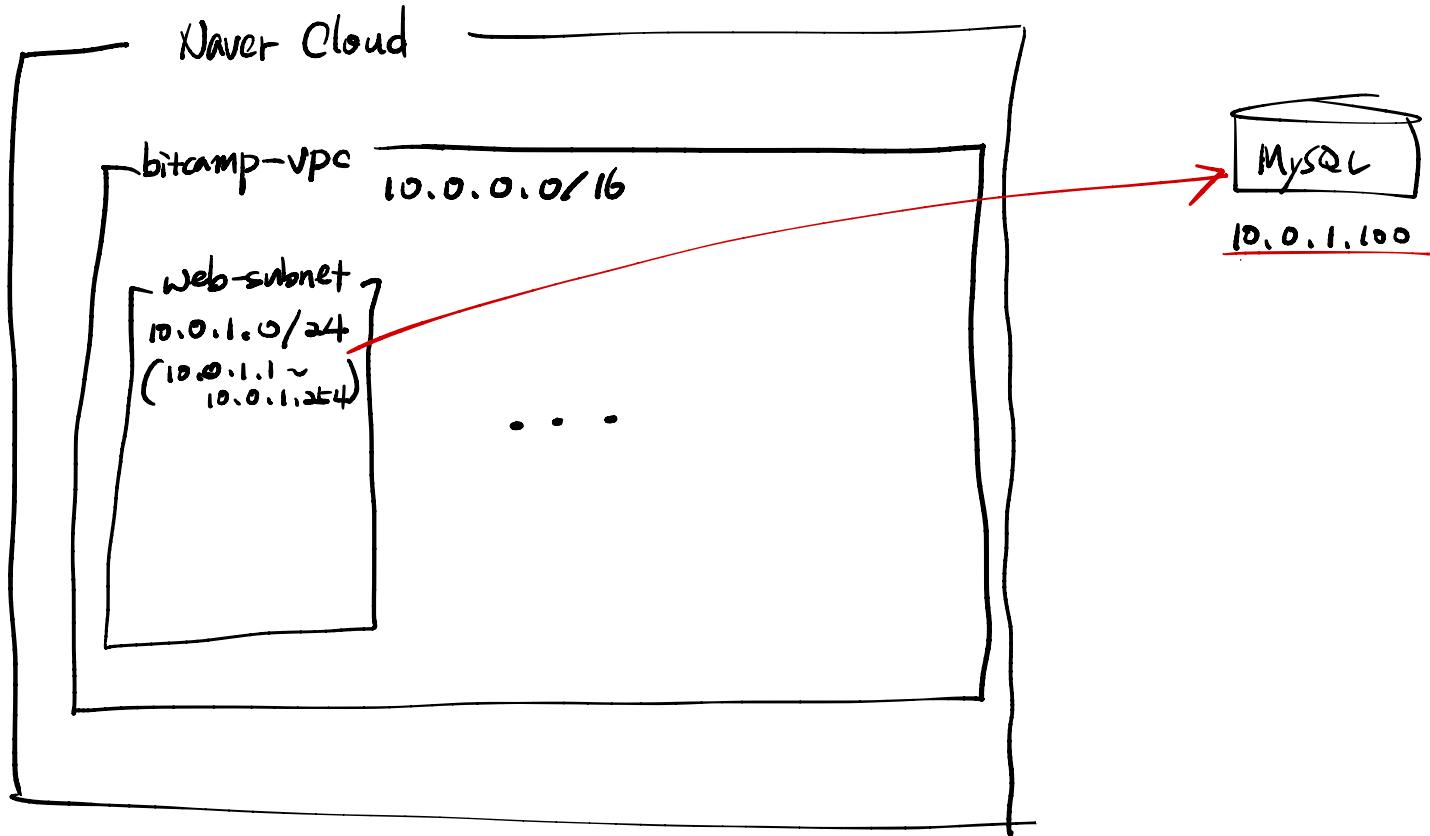
* ORI



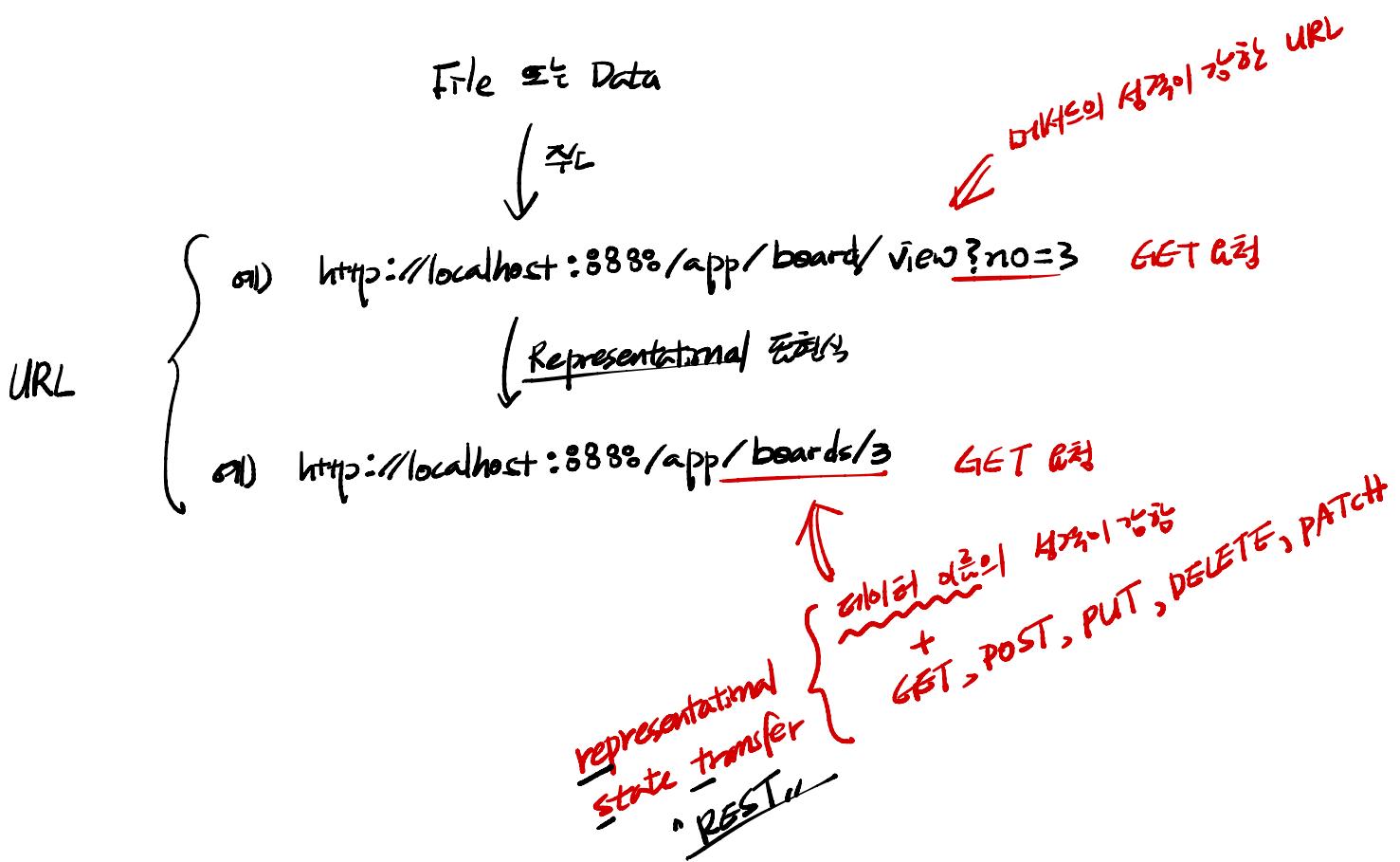
* 대입



* VPC (Virtual Private Cloud) - 퍼블릭 네트워크 대신 퍼비전 네트워크를 사용하는 자체적인 네트워크 환경



* RESTful API



* function → REST API

다른 프로토콜

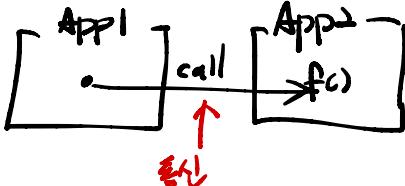
① function

(RPC)

② Remote Procedure Call

(RMI)

③ Remote Method Invocation



C 등 프로그래밍 언어

기법의 암호화



App의 행동을 분석 (분산 컴퓨팅)



한 App이 하던 일을

다른 App으로 풀어서 실행

App 간의 의존성이 줄어듦



다른 App의
function을
호출할 수 있는
기술이 등장하게 되었다.



C++ 등 프로그래밍 언어



ODP 프로그래밍의 특징이 빠져

RPC를 개선

* function → REST API

③ RMI → ④ CORBA

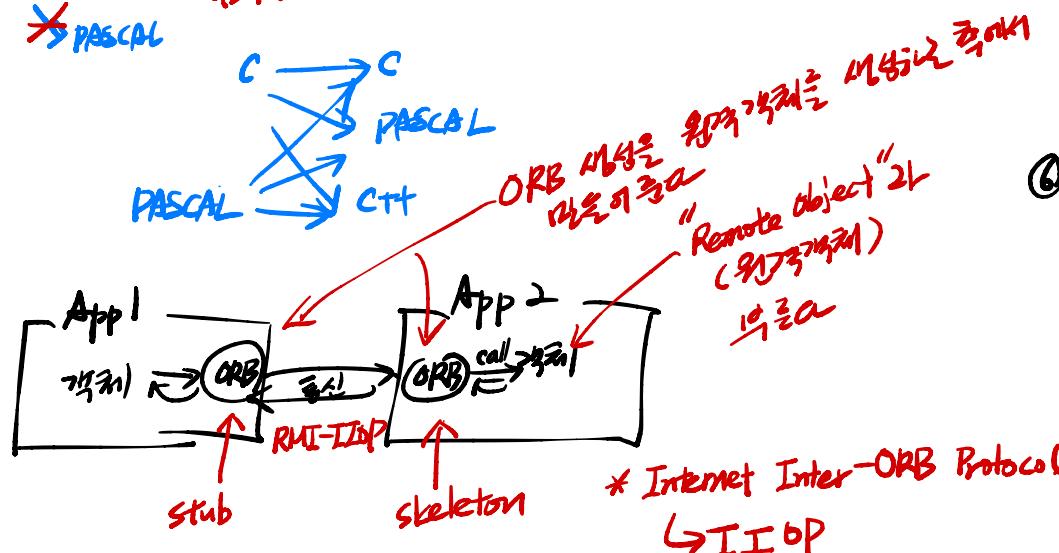
같은 인터프리터
만든 App끼리
함수를 호출하는
방식

C → C
✖ PASCAL

common
Object Request Broker
Architecture

자신의 언어로 만든 App끼리
함수를 호출하는 방식

C → C
✖ PASCAL
✖ C++



⑤ Web-service

✓ 파일과 HTTP 프로토콜을 이용
✓ ORB를 통과해 인터넷에 맞춰
설정되었음

↓
인터넷 규칙에 맞도록 개발된
작품을 찾기!

⑥ Enterprise JavaBeans (EJB)

Java에서만 가능 RMI 가짐

있음!

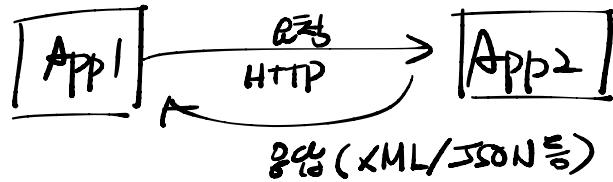
2000년 초반에는 PC를
대상으로 디바이스를 위한 EJB가
제작되었지만 초기에는 제한적이다.

* function → REST API

(Web-Service)
EJB

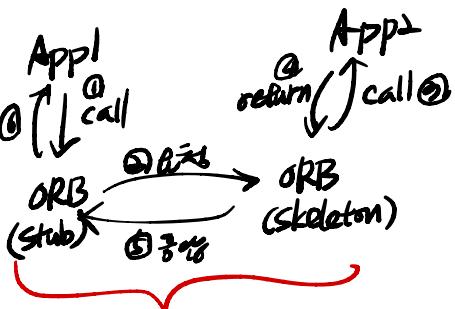
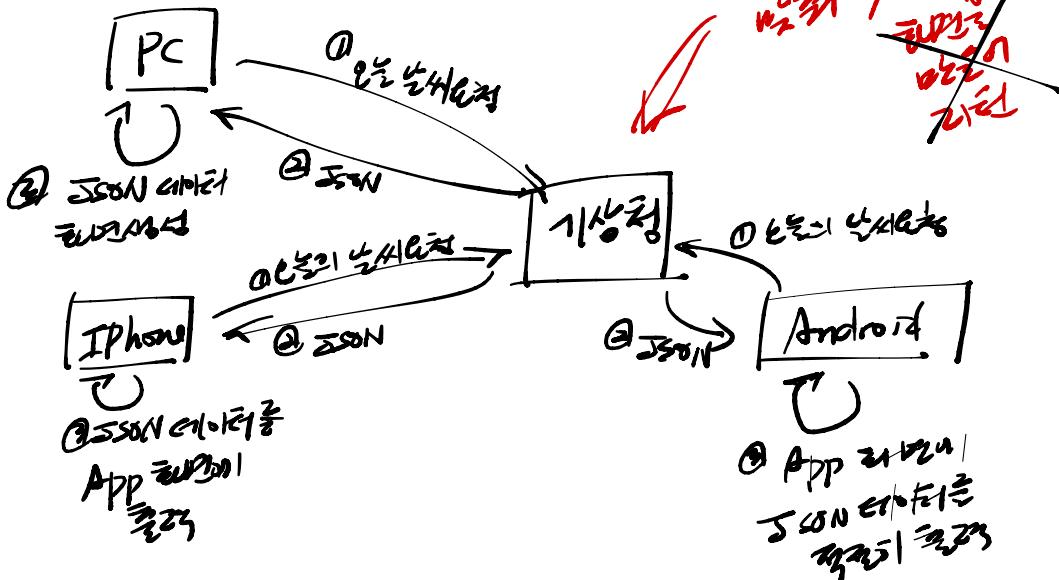
→ REST API

인터넷으로
ORB를 통한
서비스 제공



Response (XML/JSON 등)

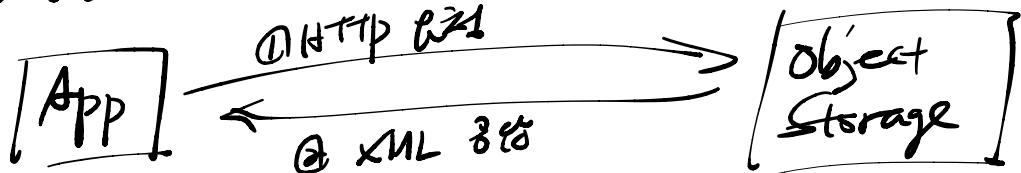
인터넷으로
ORB를 통한
서비스 제공



인터넷으로
ORB를 통한
서비스 제공

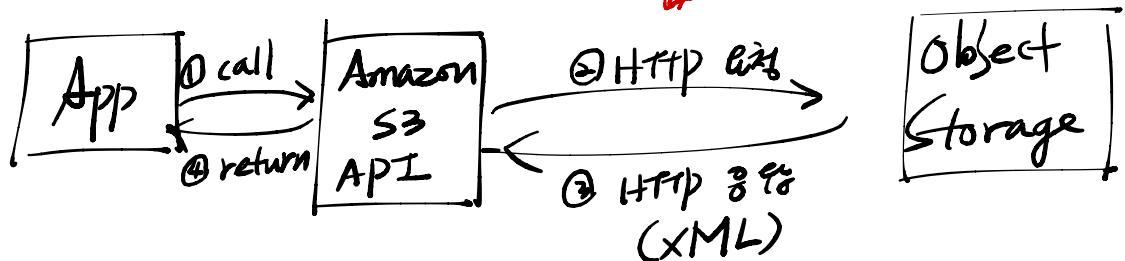
* Object Storage REST API API
↳ HTTP 퀼/값, 키!

① 직접 접근



HTTP 퀼/값은 직접 접근하기 편리하다
HTTP 응답을 직접 접근하기 편리하다

② 경유 애플리케이션



* 내부로 첨부파일은 NCP의 Object Storage에 저장

