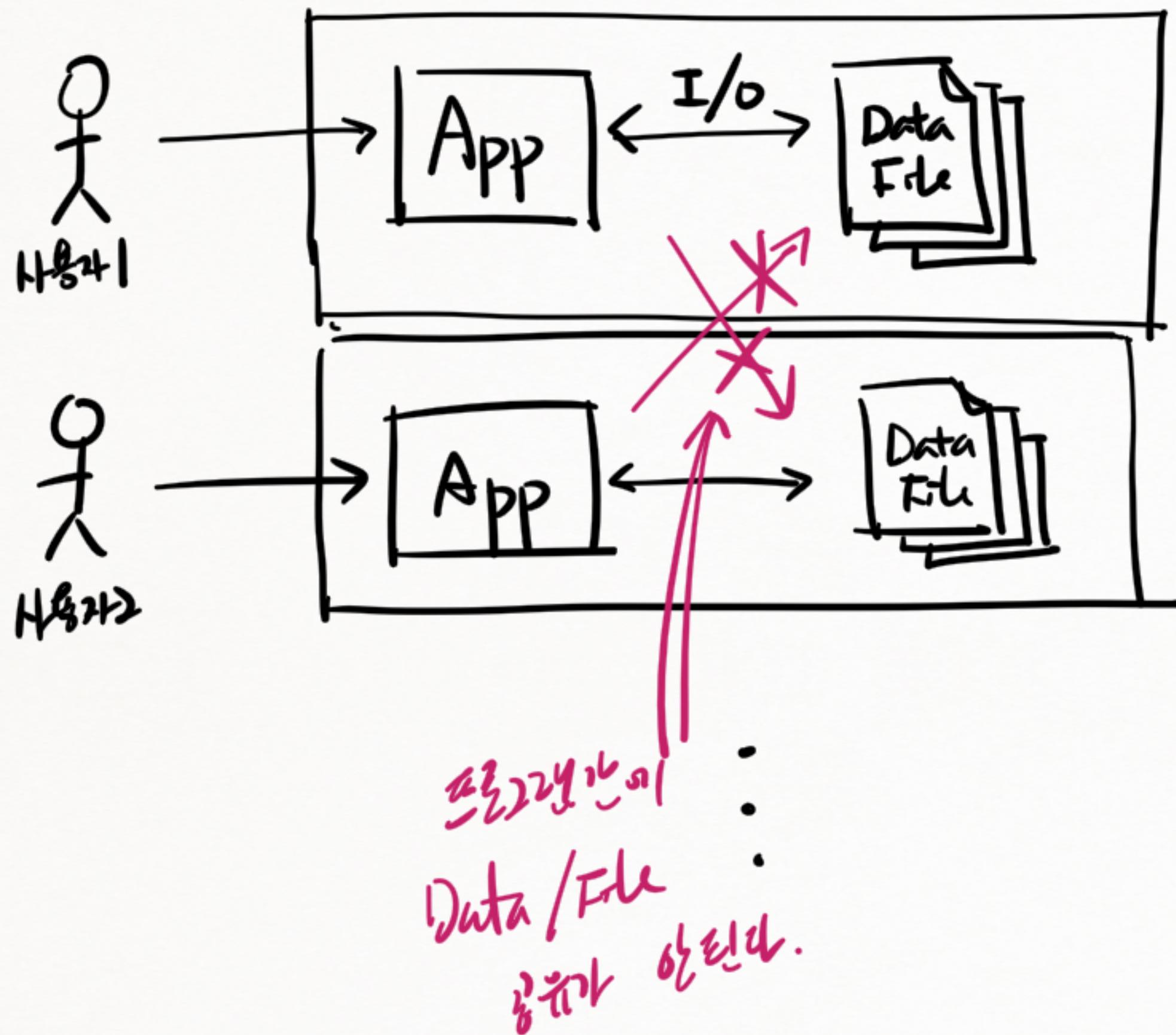


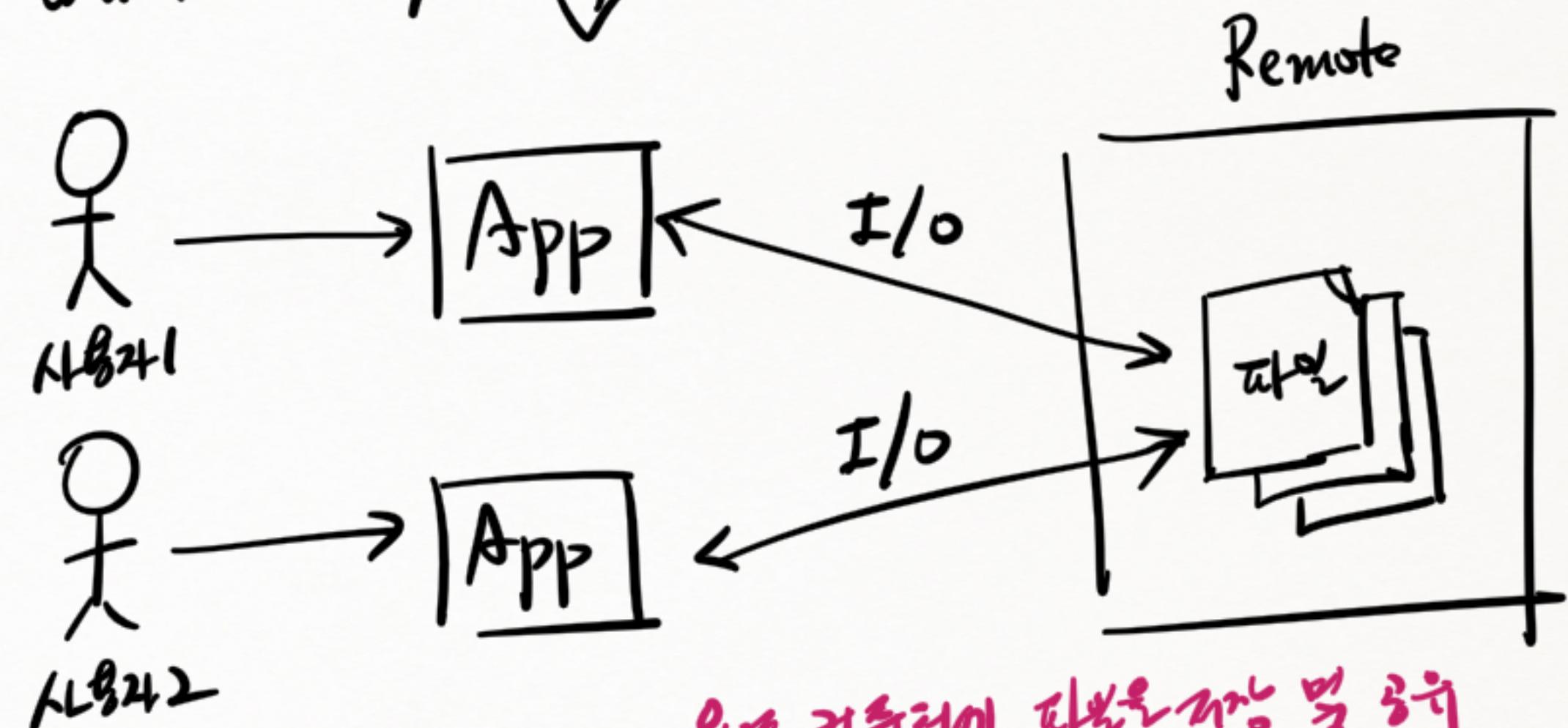
37. 바이오킹을 통한 데이터 공유

① 현황



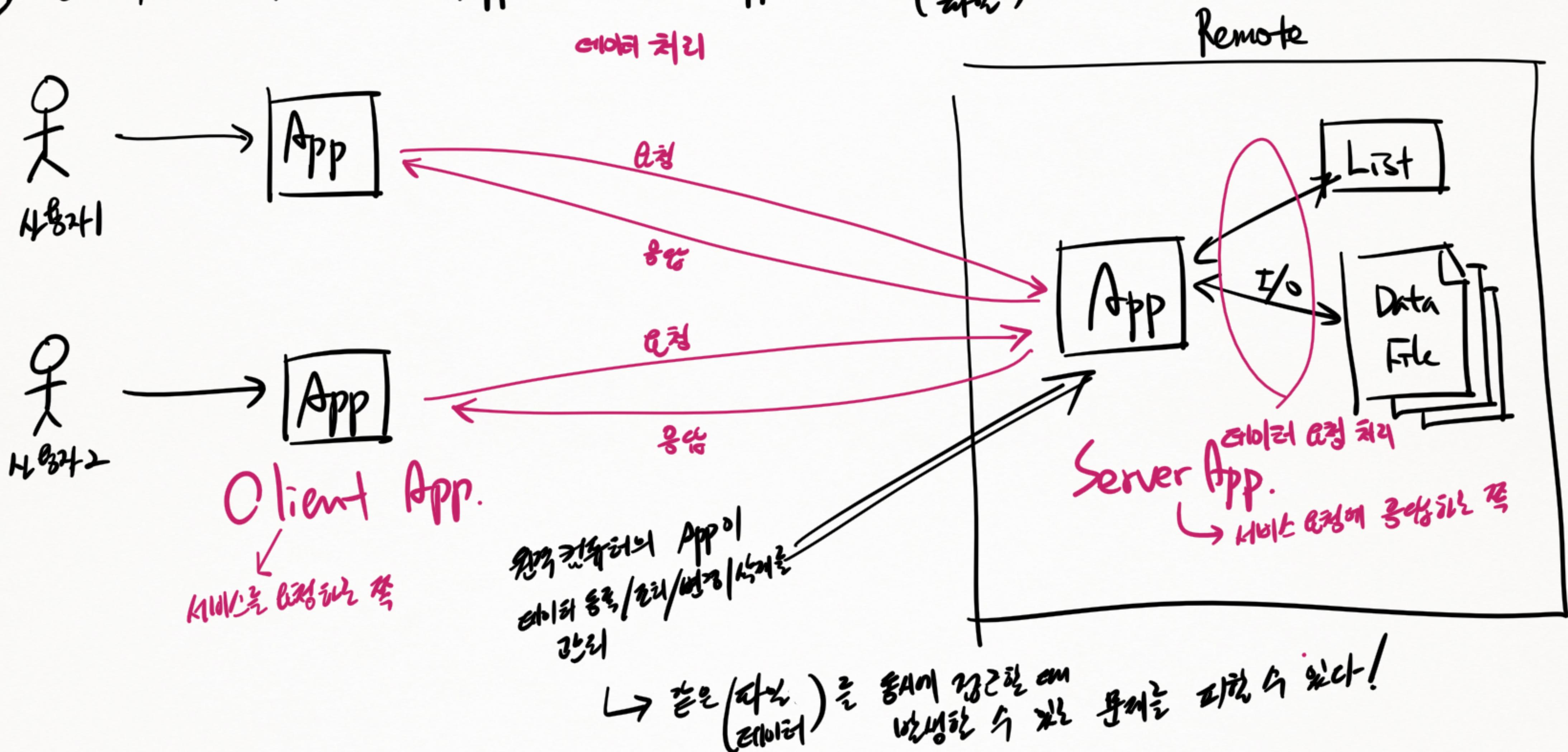
App을 실행하는 컴퓨터
데이터 파일이 로컬에 저장된다.
↓
App 간에 데이터를 공유할 수 없다!

② 데이터 파일은
별도의 컴퓨터에 분리/공유
가능할까?

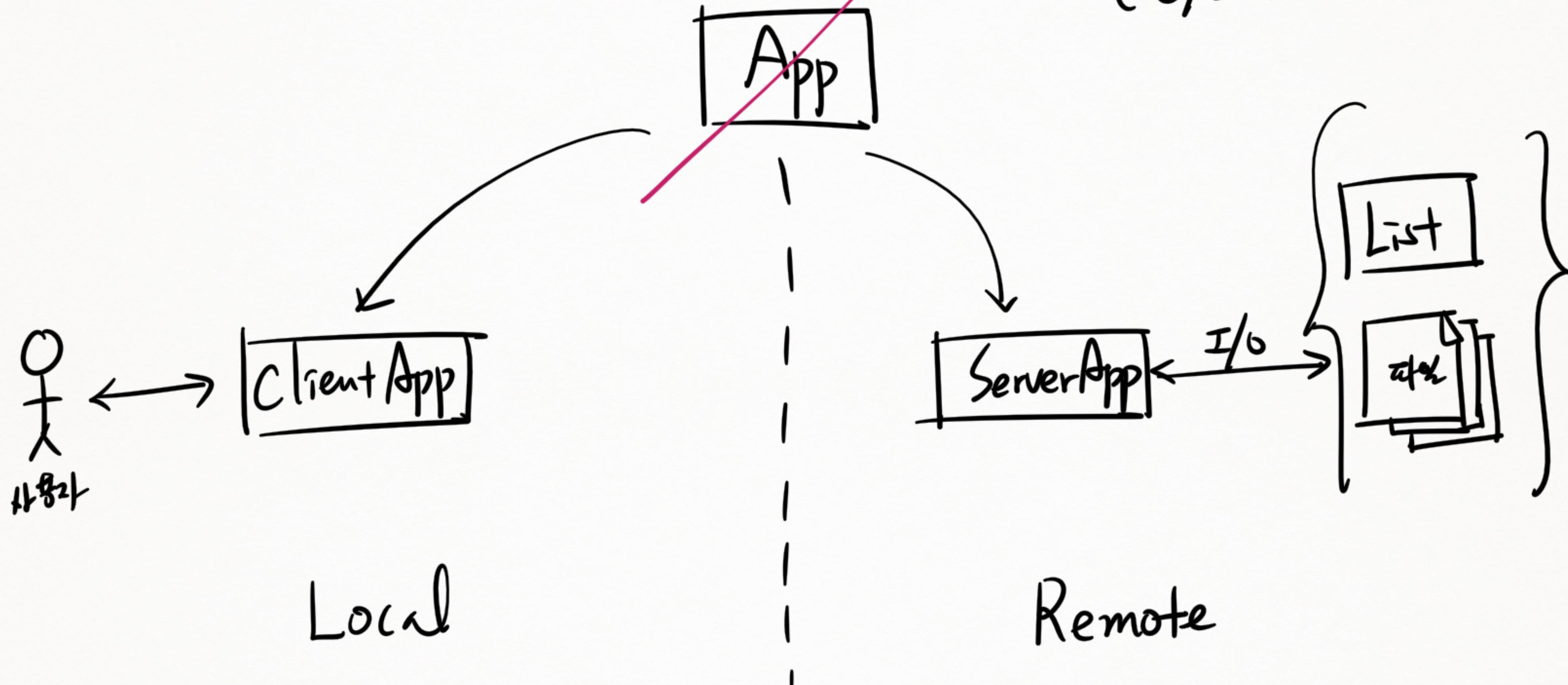


- 원격 컴퓨터에 파일을 저장 및 공유
- 동시에 여러 App이 같은 파일을 편집하다 보면
파일의 데이터가 깨질수 있다.

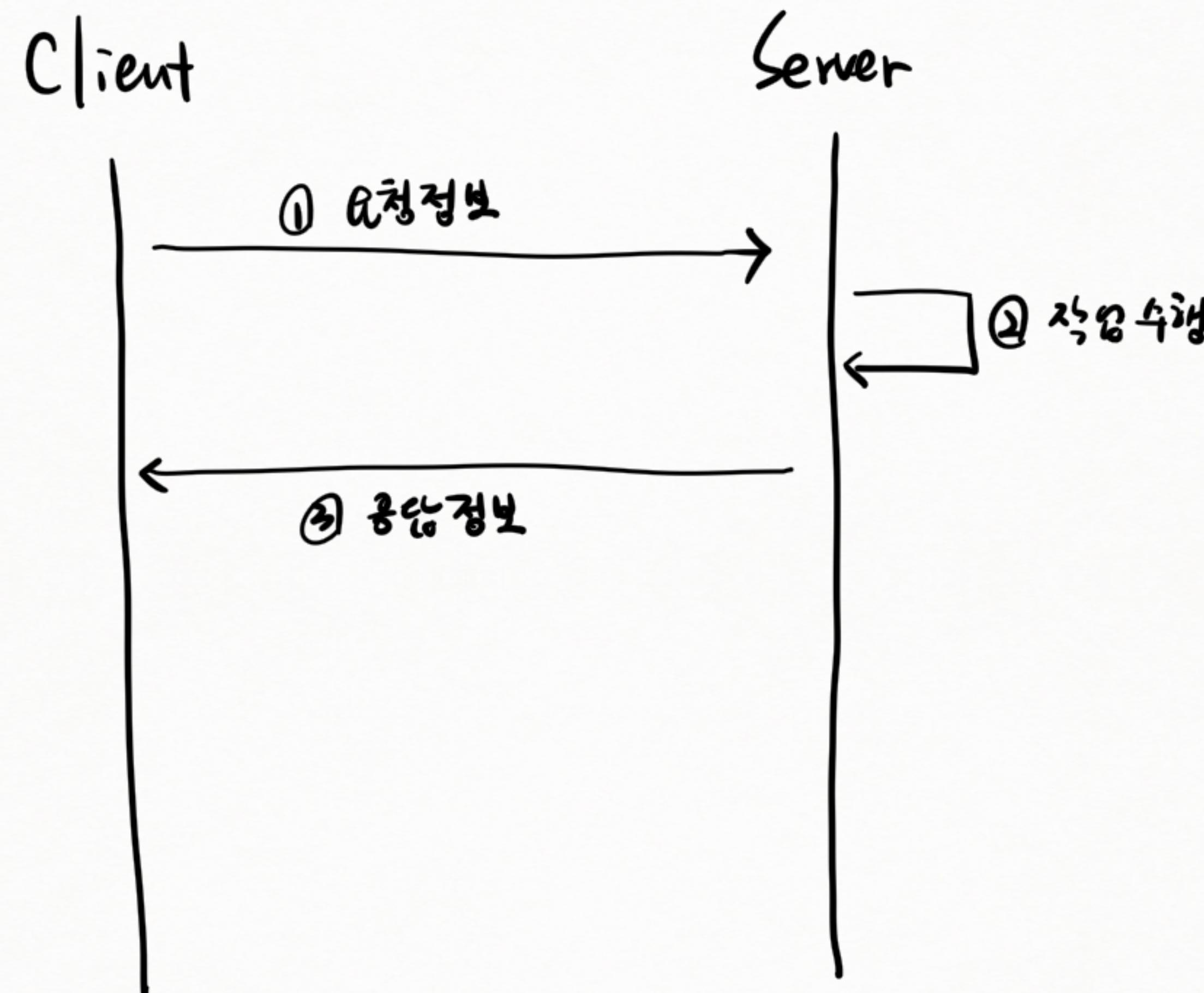
③ 데이터 관리 기능을 별도의 App. 으로 분리 - App이 직제 (데이터) 를 접근하는 것을 막는다.



* System Architecture : Client / Server Architecture
(C/S Architecture)



* client / server 통신 규칙(protocol)



* client / server 통신 규칙 (protocol)

① 요청 정보 (JSON 문자열)

```
{  
  "command": "테이터이름/명령",  
  "data": "JSON 문자열"  
}
```

↓ 예

```
{  
  "command": "board/insert",  
  "data": {"title": "...",  
           "content": "...",  
           ... :  
           }  
}
```

"테이터이름 / 명령"

기사로: board
회원: member
독서록: reading

서버의 DAO 메서드명.

JSON 문자열

command 값: 일반 문자열
data 값: JSON 문자열

- * client / server 통신 규칙 (protocol)

② 응답 정보 (JSON 문자열)

```
{  
    "status": "실행결과",  
    "result": "JSON 문자열"  
}
```

↓ 성공 예

```
{  
    "status": "success",  
    "result": "[{"no": 12, ... }]"  
}
```

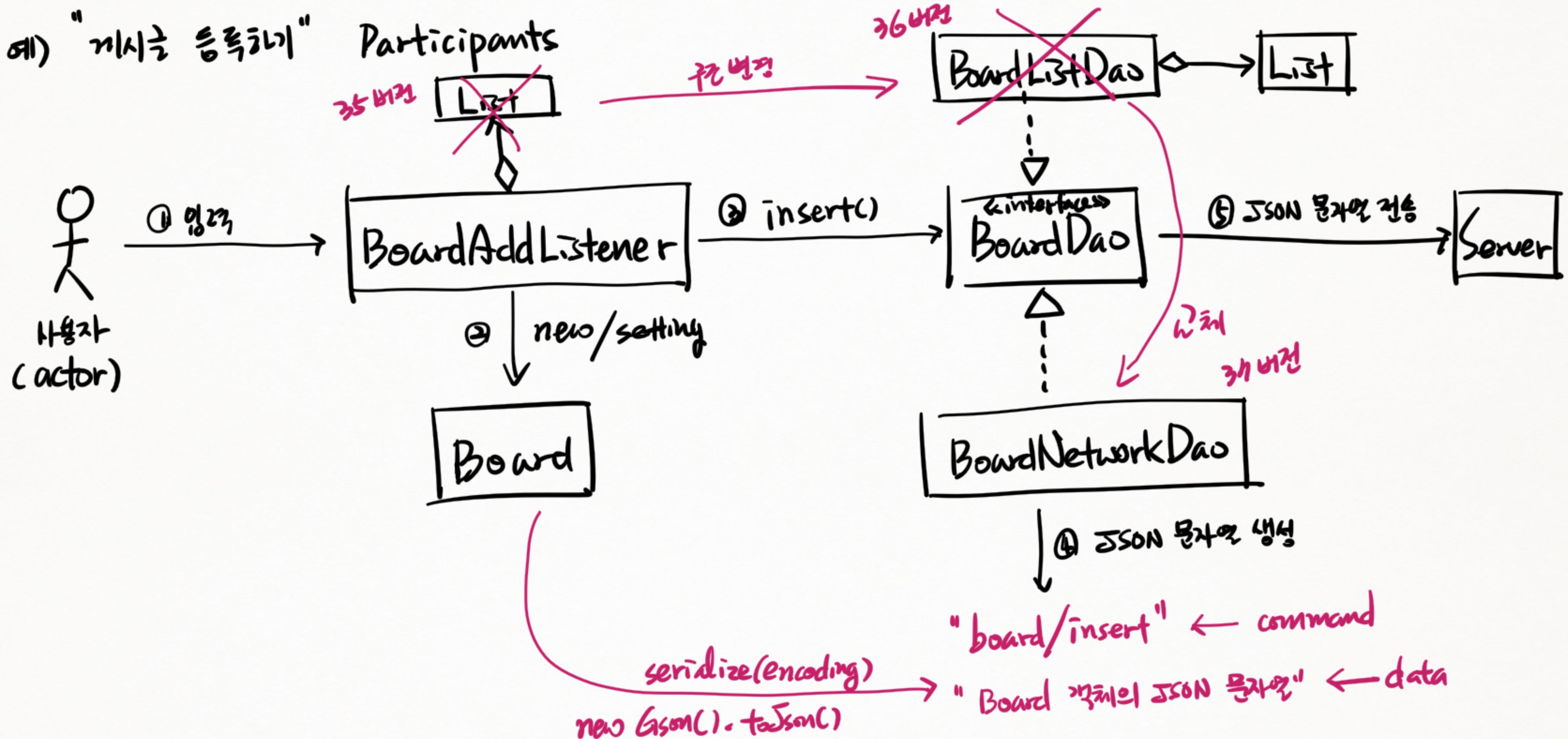
- * 실행 결과
"success": 성공
"failure": 실패

↓ 실패 예

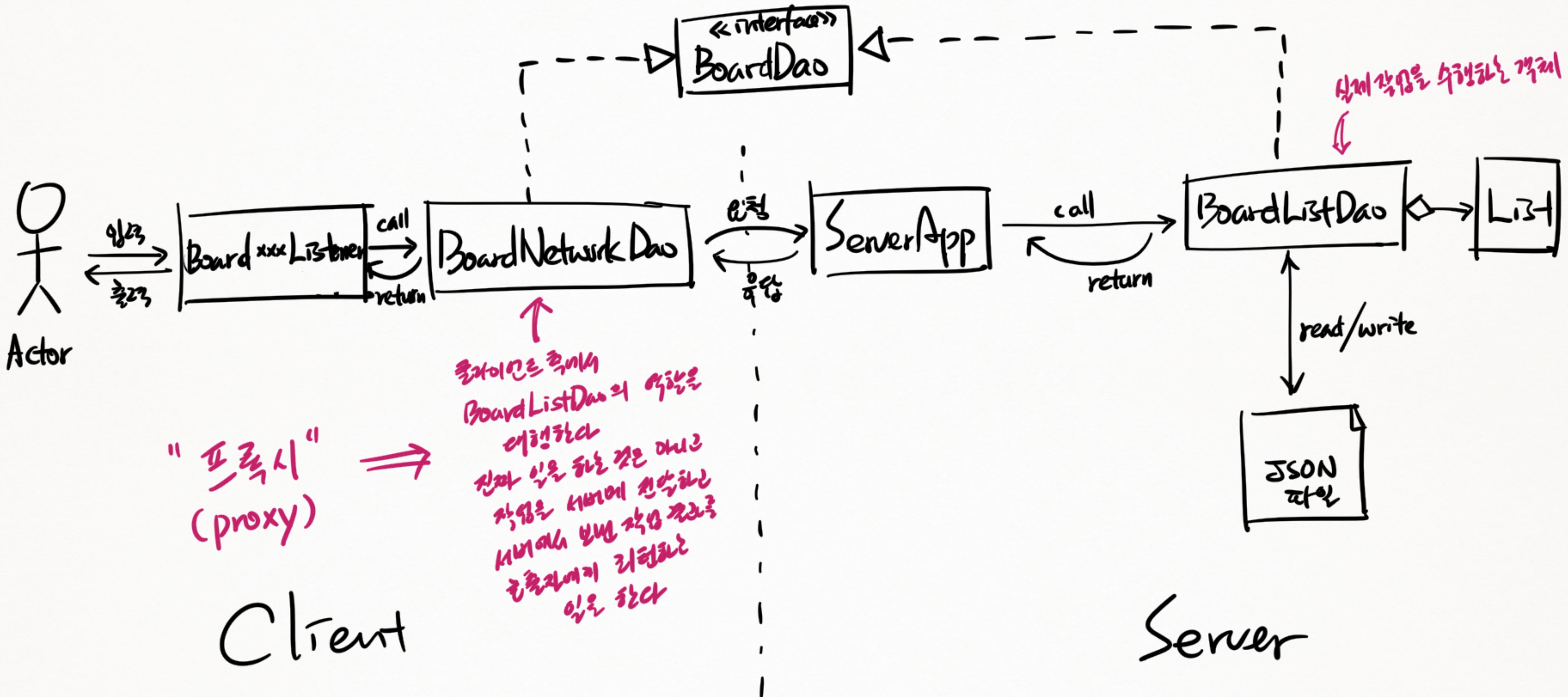
```
{  
    "status": "failure",  
    "result": "해당 번호의 데이터가 없습니다!"  
}
```

* 요청 정보 뷰티가 작업에 참여하는 객체들과 실행흐름

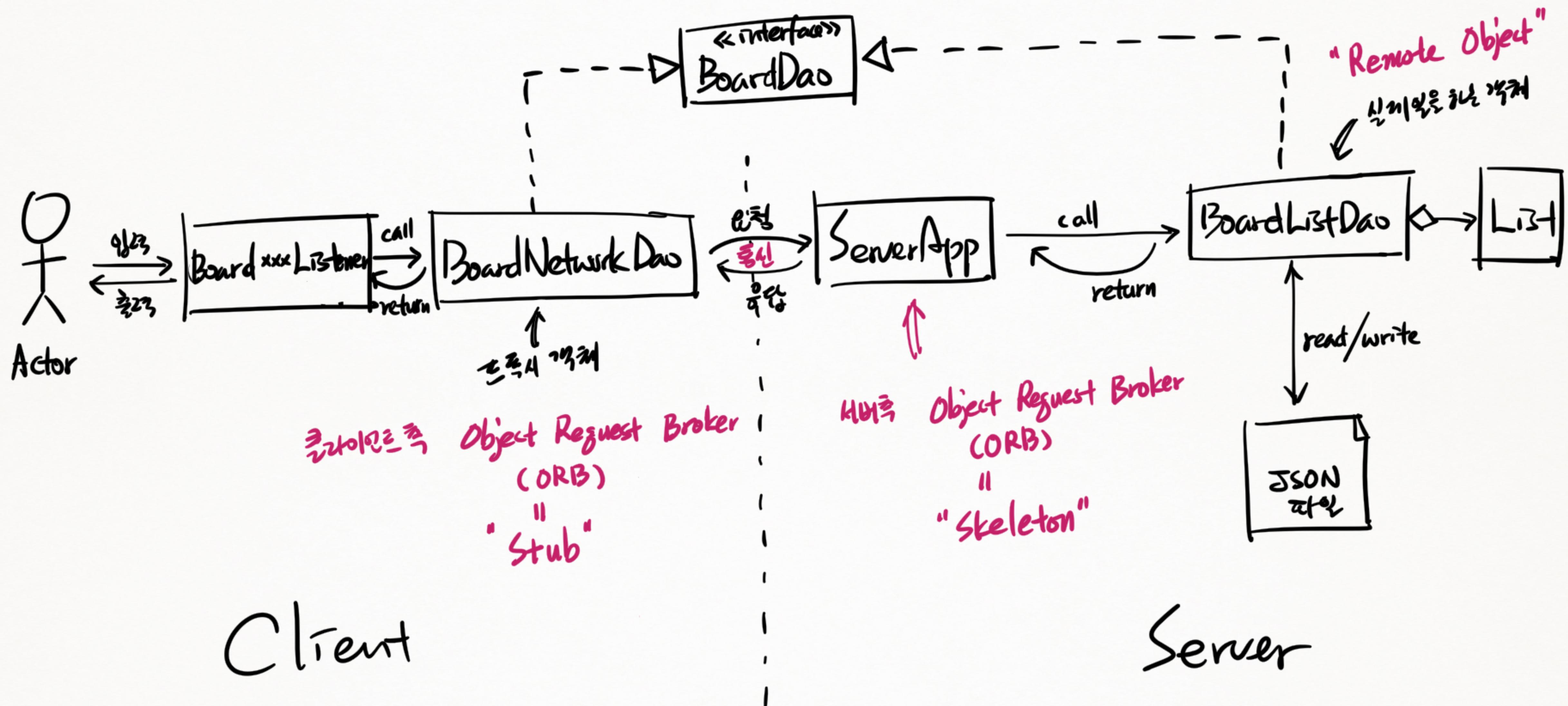
예) "게시글 등록하기" Participants



* DAO 와 Proxy 패턴 (GoF)

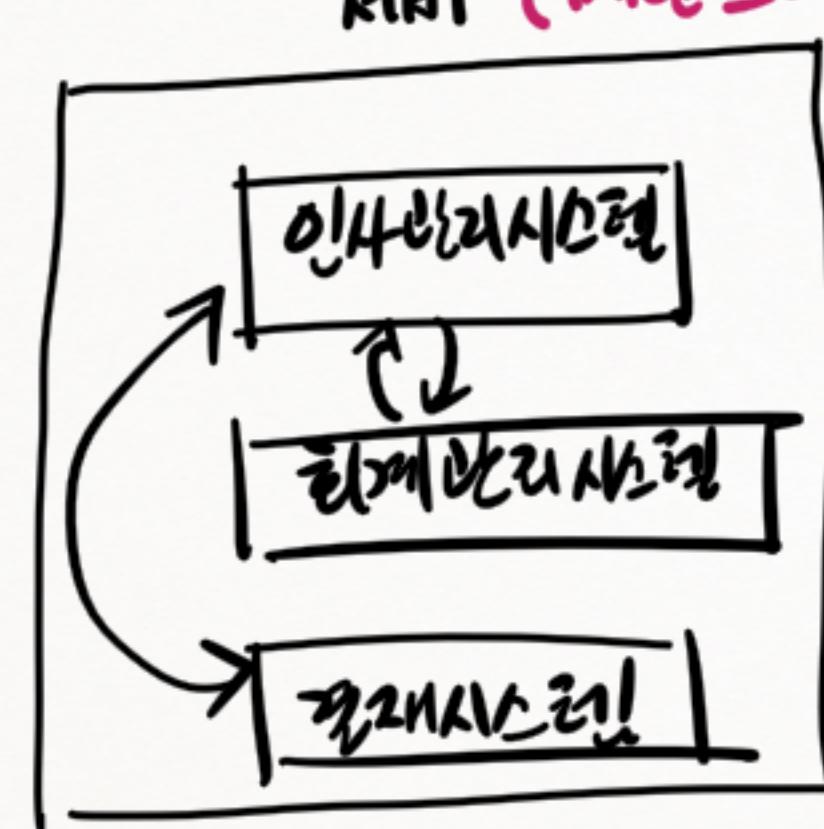


* DAO 와 Proxy 패턴 (GoF) ↗ 예술자의 명칭



* 온라인 컴퓨팅

① 중앙 집중식 컴퓨팅



문제점

② 온라인 컴퓨팅

⇒ Rest API (RESTful)

→ 크기↓ 가격↓ 용량↓ (down sizing)

(클라우드웨어) 서버

↳ Unix

↳ Solaris, HP-UX, IBM AIX, ...

(클라우드웨어) 서버 2

↳ Linux

(클라우드웨어) 서버 3

↳ Windows

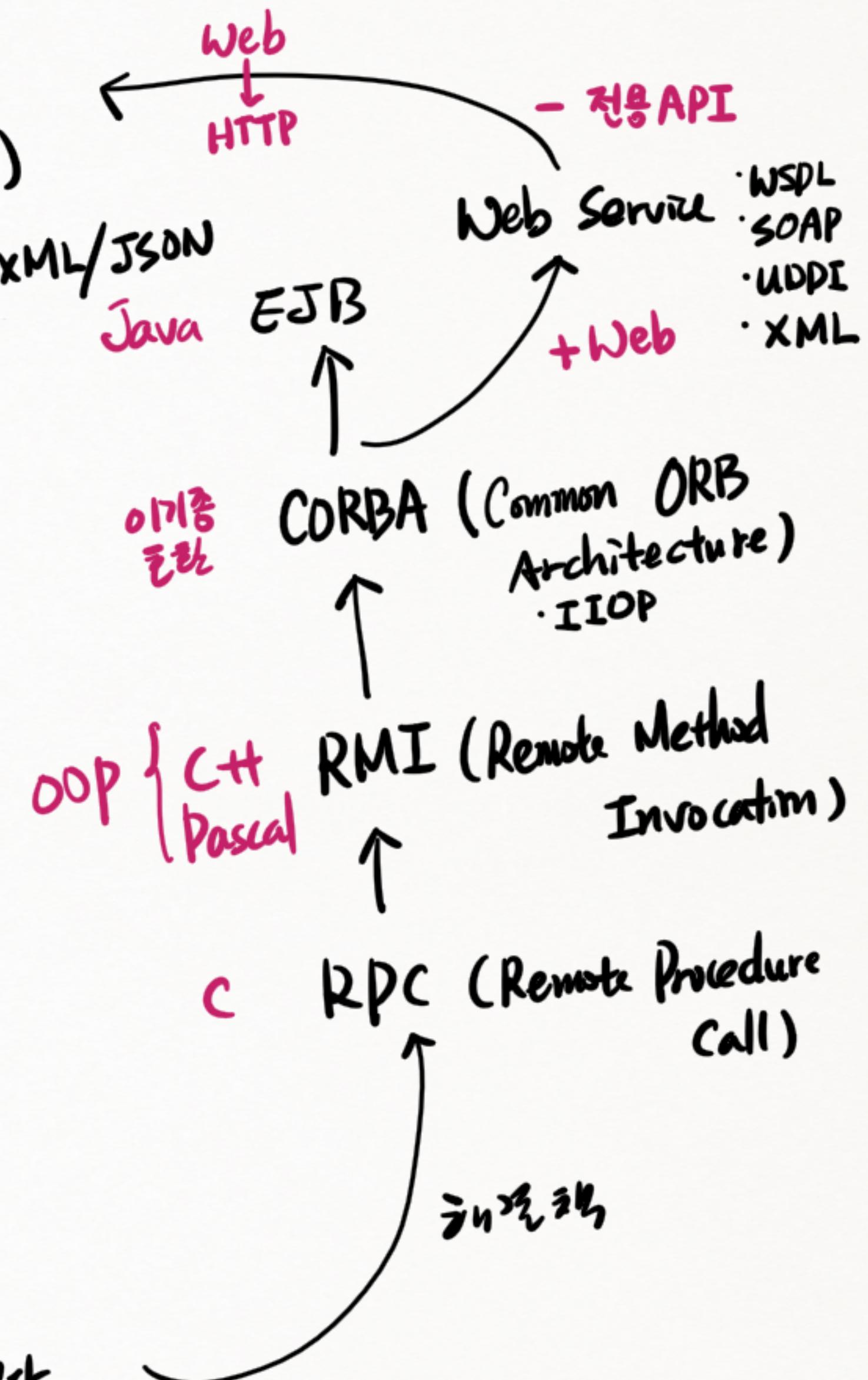
인내 관리 시스템

회계 관리 시스템

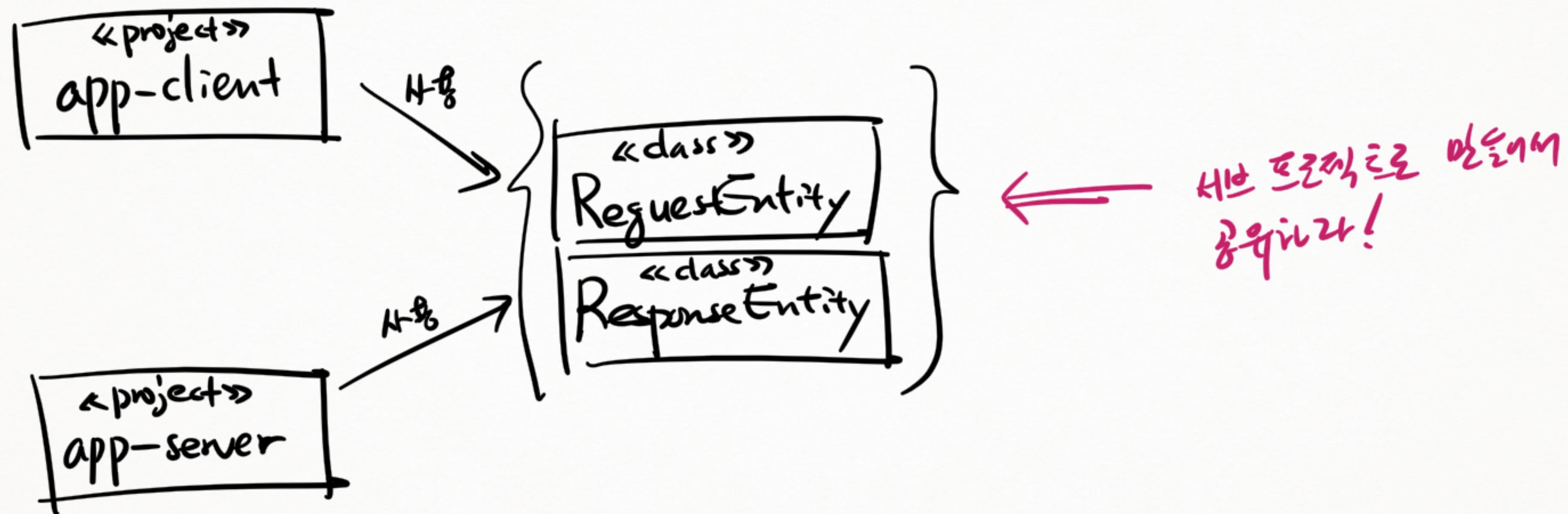
고객 관리 시스템



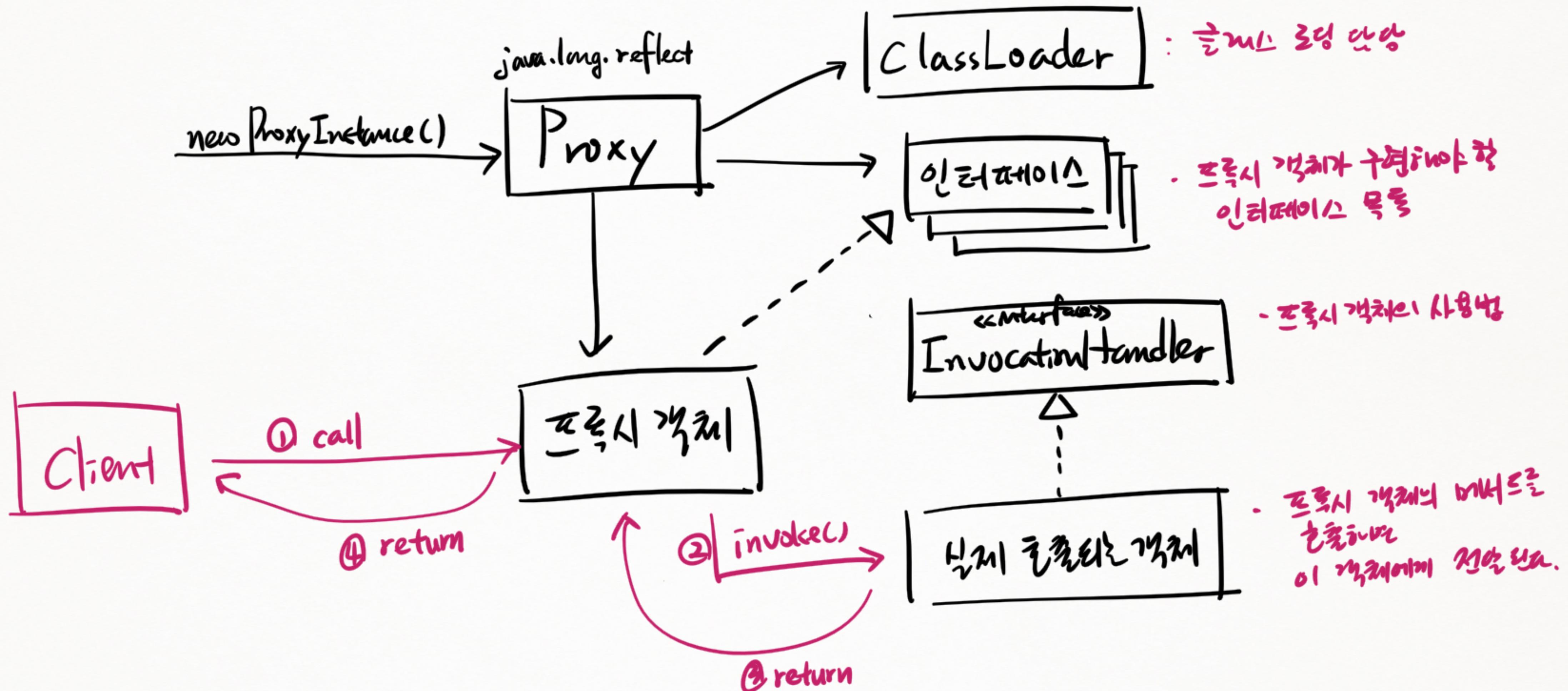
문제점



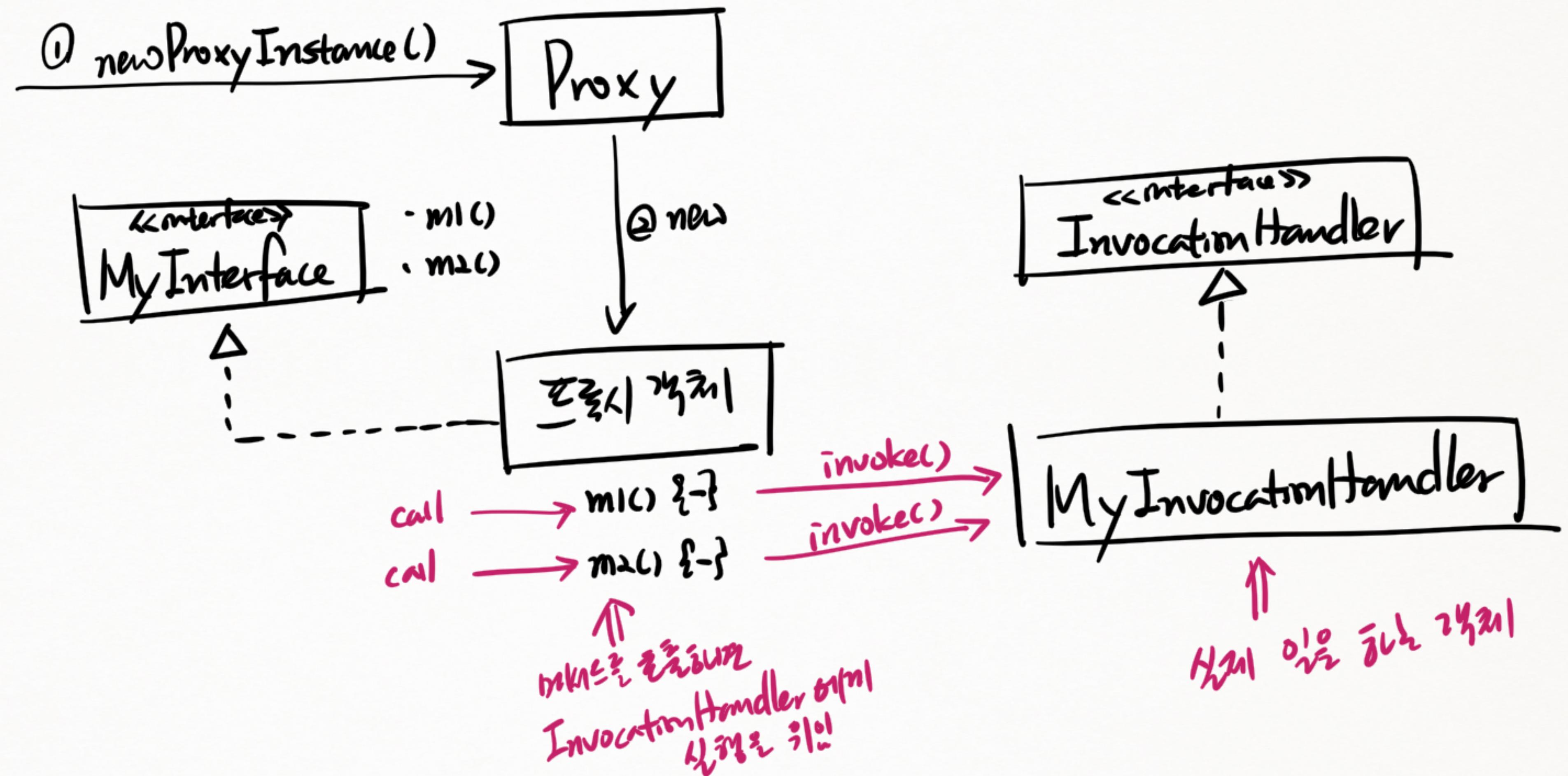
* Project 간에 헤더 공유하기



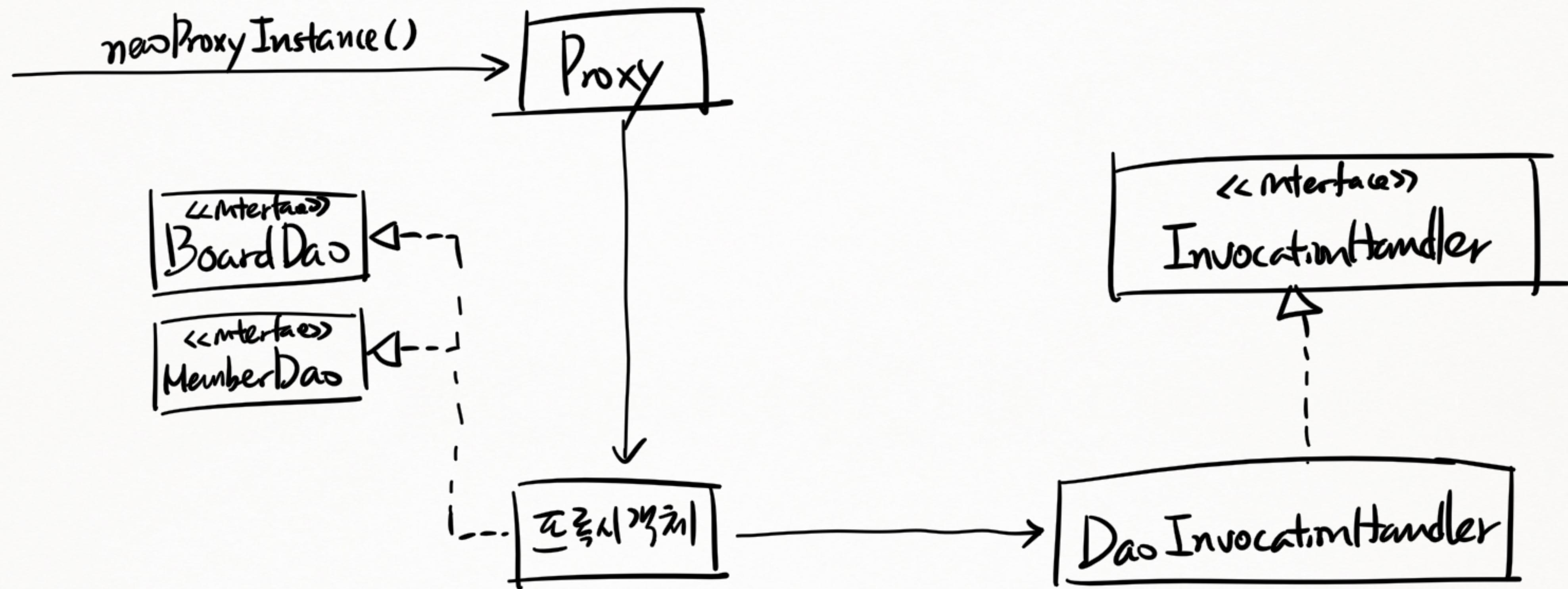
3.8. 프록시 객체 자동 생성



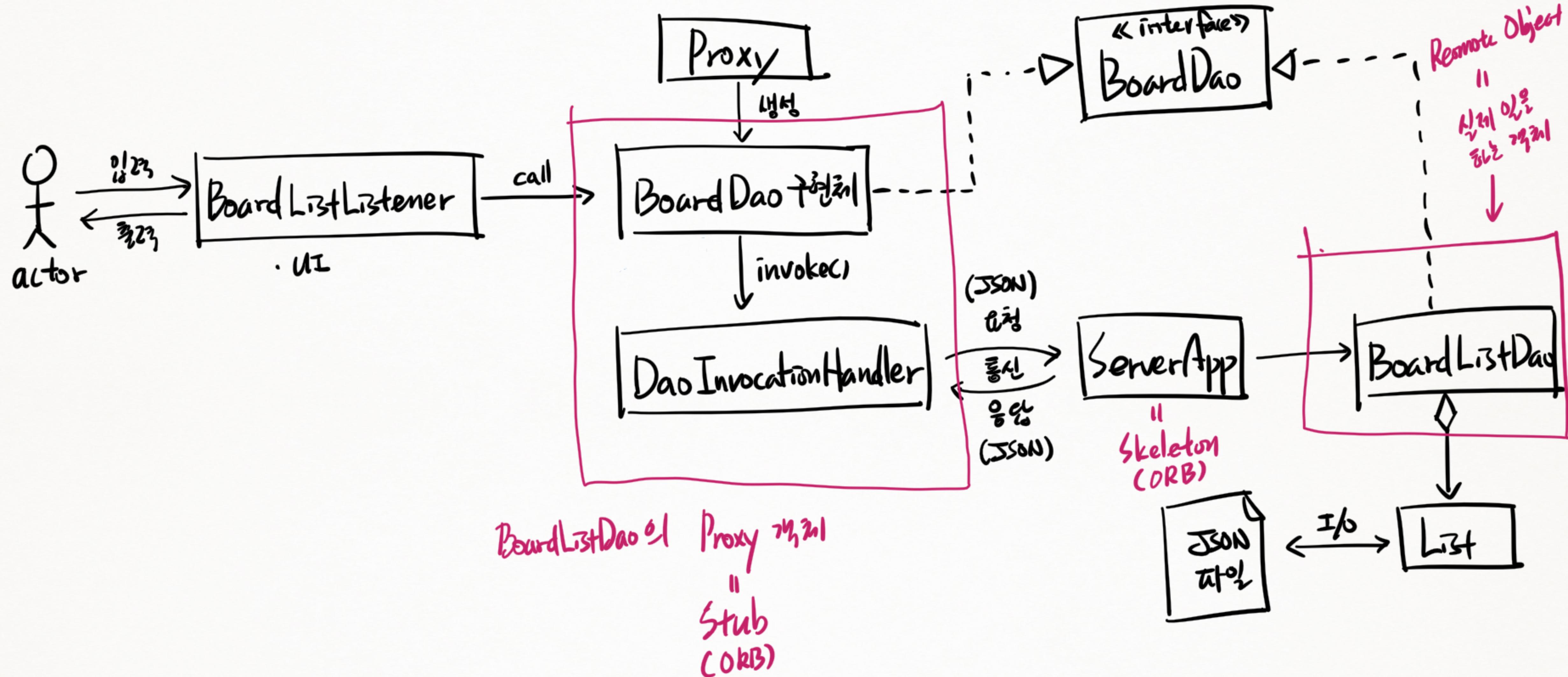
38. 프록시 객체 사용 예제 - ②



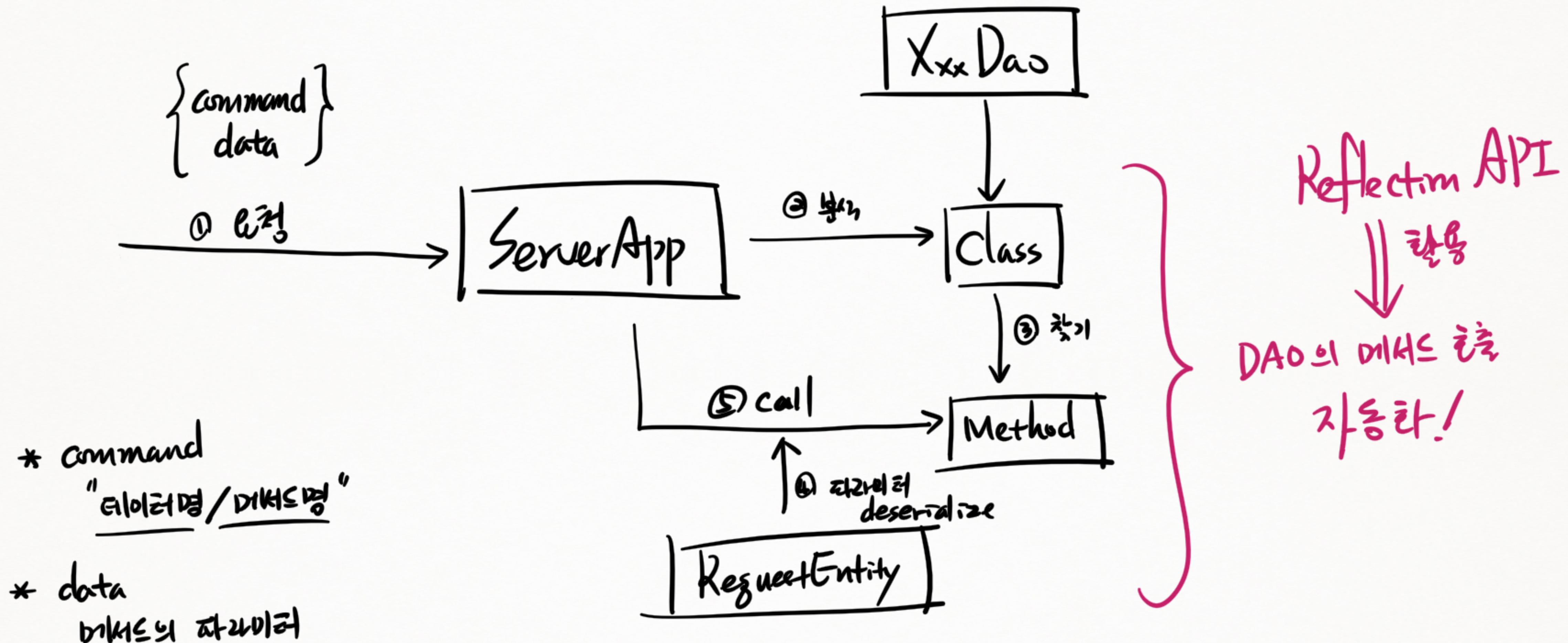
38. 프록시 패턴의 사용 예제 - Participants



3B. 프록시 개념 자동 생성 - Participants II



39. Reflection API를 사용하여 DAO 객체의 메서드 호출을 자동화하기



40. 예외 처리 예제

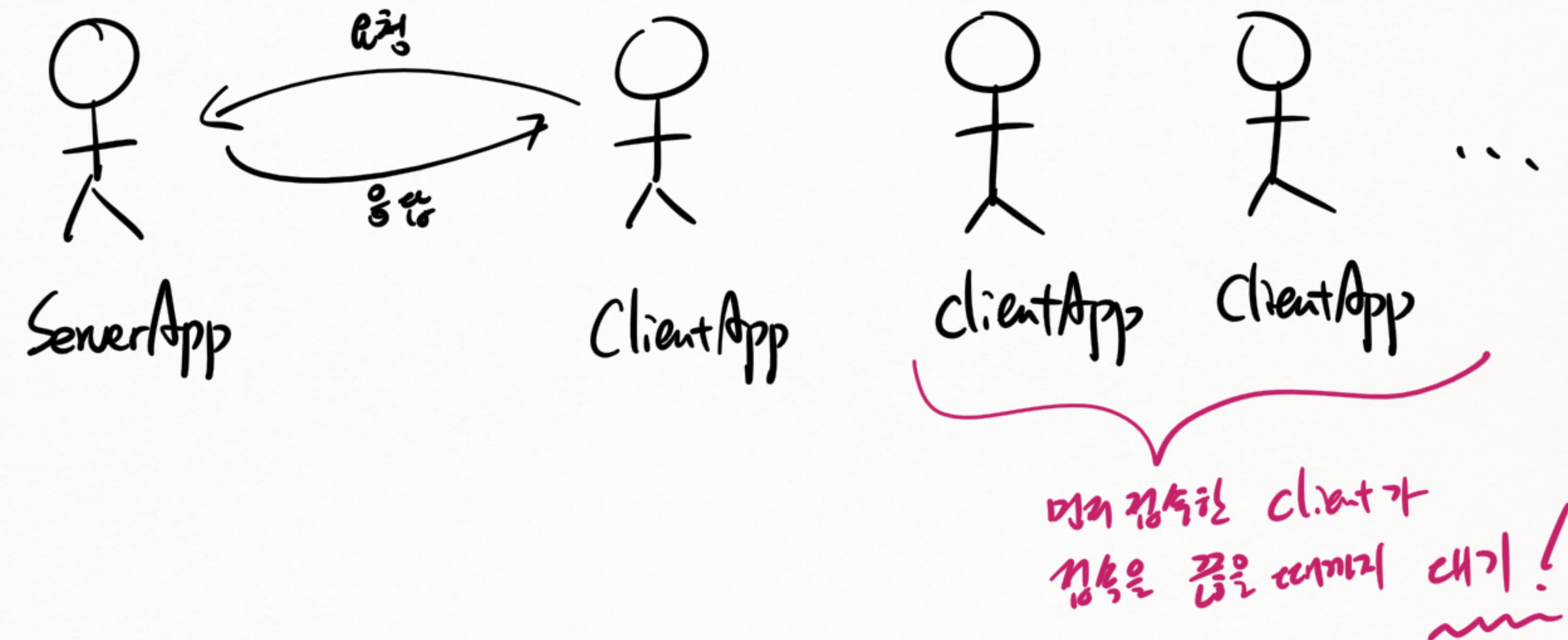
① client 편

```
try {  
    ==  
    menu.execute();  
    ==  
} catch (Exception e) {  
    ==  
}
```

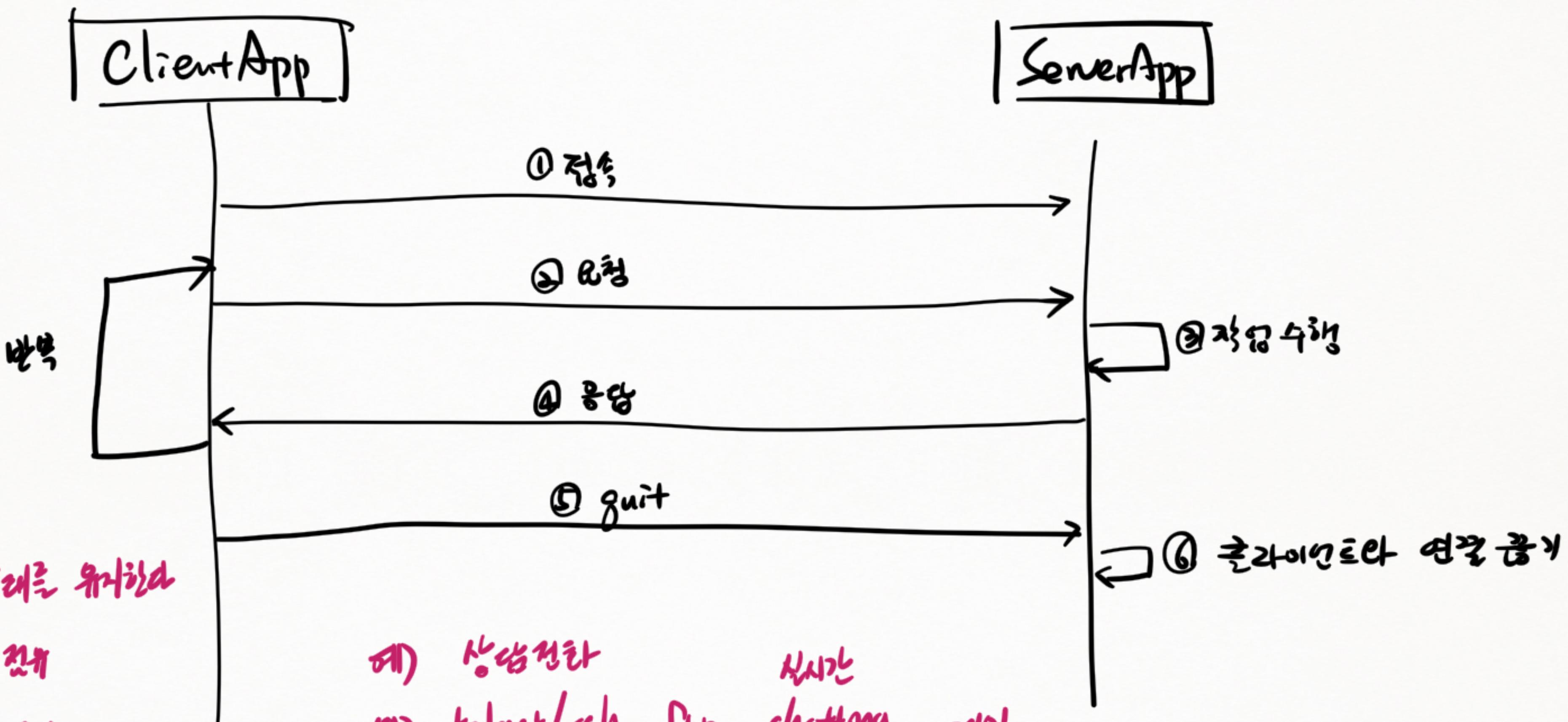
② server 편

```
try {  
    ==  
    dao.invoke(...);  
    ==  
} catch (Exception e) {  
    ==  
}
```

41. 예전 카카오로 페치를 스터디북으로 만들기 - Stateful 방식



41. 여러 주체들로 퍼진을 관리하고 처리하기 - Stateful 방식



* 특징
① 접속과 연결된 상태를 유지한다

- 새끼 자리를 찾기
 - 동시에 많은 클리어언트 접속을 유지 할 수 있다

② 초과이면은 정보유지 → 작업결과유지 → 연계작업을 처리하기 수월.

42. 예제 클라이언트 페처를 Stateless 방식

