NHN FORWARD

API를 프록시해 보기

다룰 내용

1. 이걸 왜 만들었을까
2. 어떻게 만들었을까
3. 실제 적용은 어땠을까
4. 더 해야 할 일은 무엇일까

이걸 왜 만들었을까

이걸 왜 만들었을까

아.. 보인다 암울한 미래가..

출처: https://screenrant.com/avengers-infinity-war-doctor-strange-die-futures/

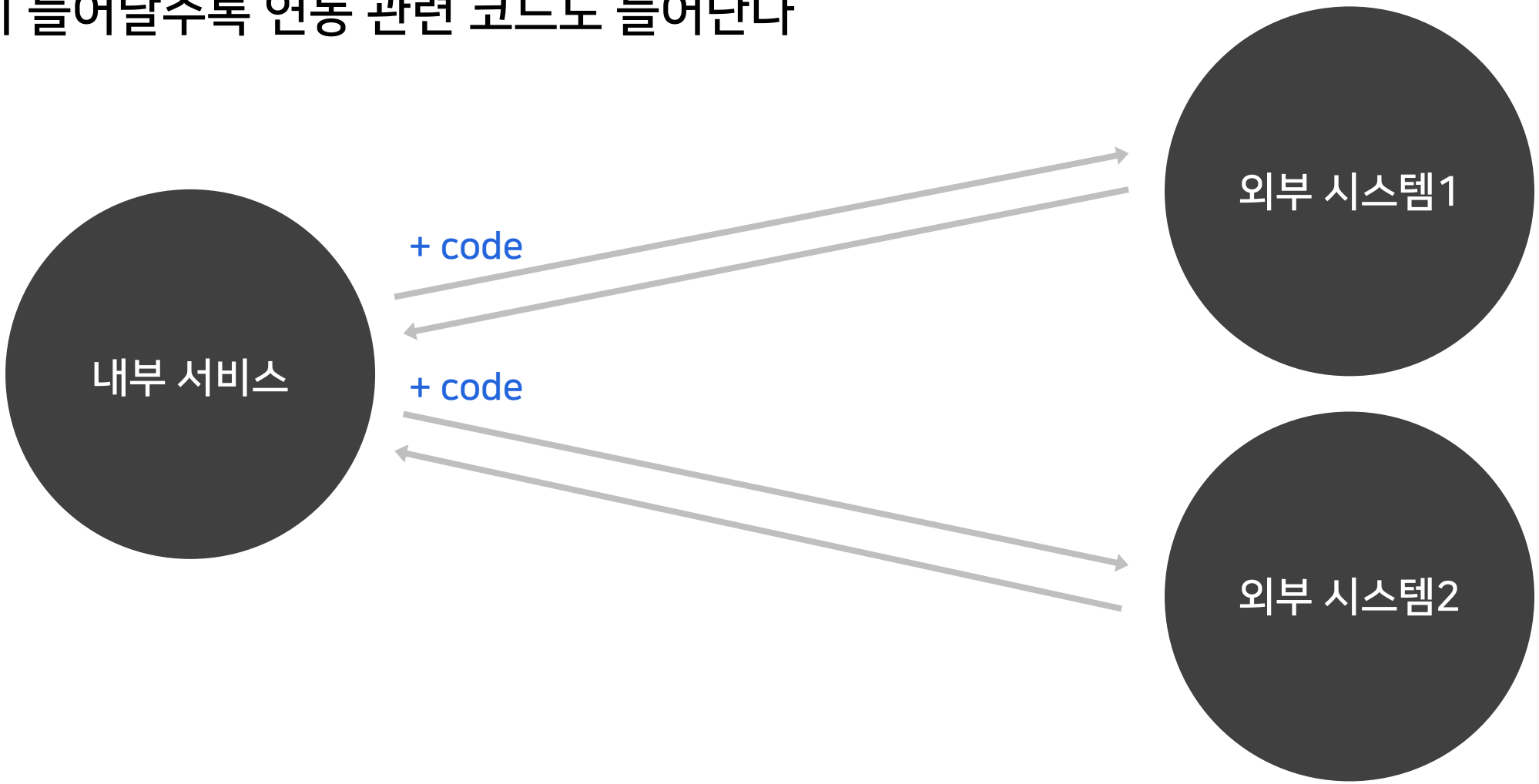
이걸 왜 만들었을까

암울한 미래가 보인다

- 추후 수많은 외부 시스템과의 연동이 필요함
- 고객(client)이 많아질수록 그 수는 계속해서 늘어나는 구조

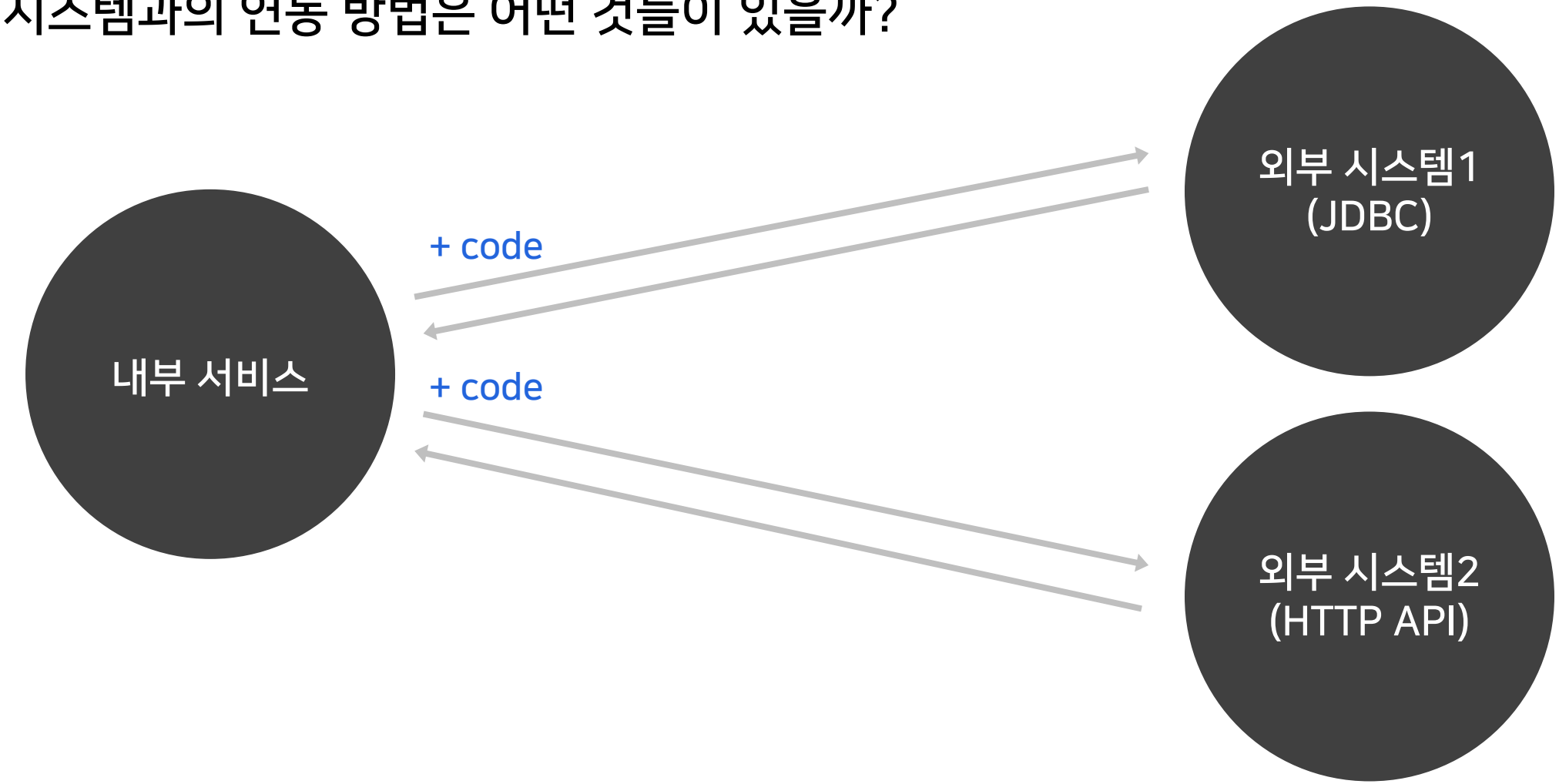
이걸 왜 만들었을까

고객이 늘어날수록 연동 관련 코드도 늘어난다



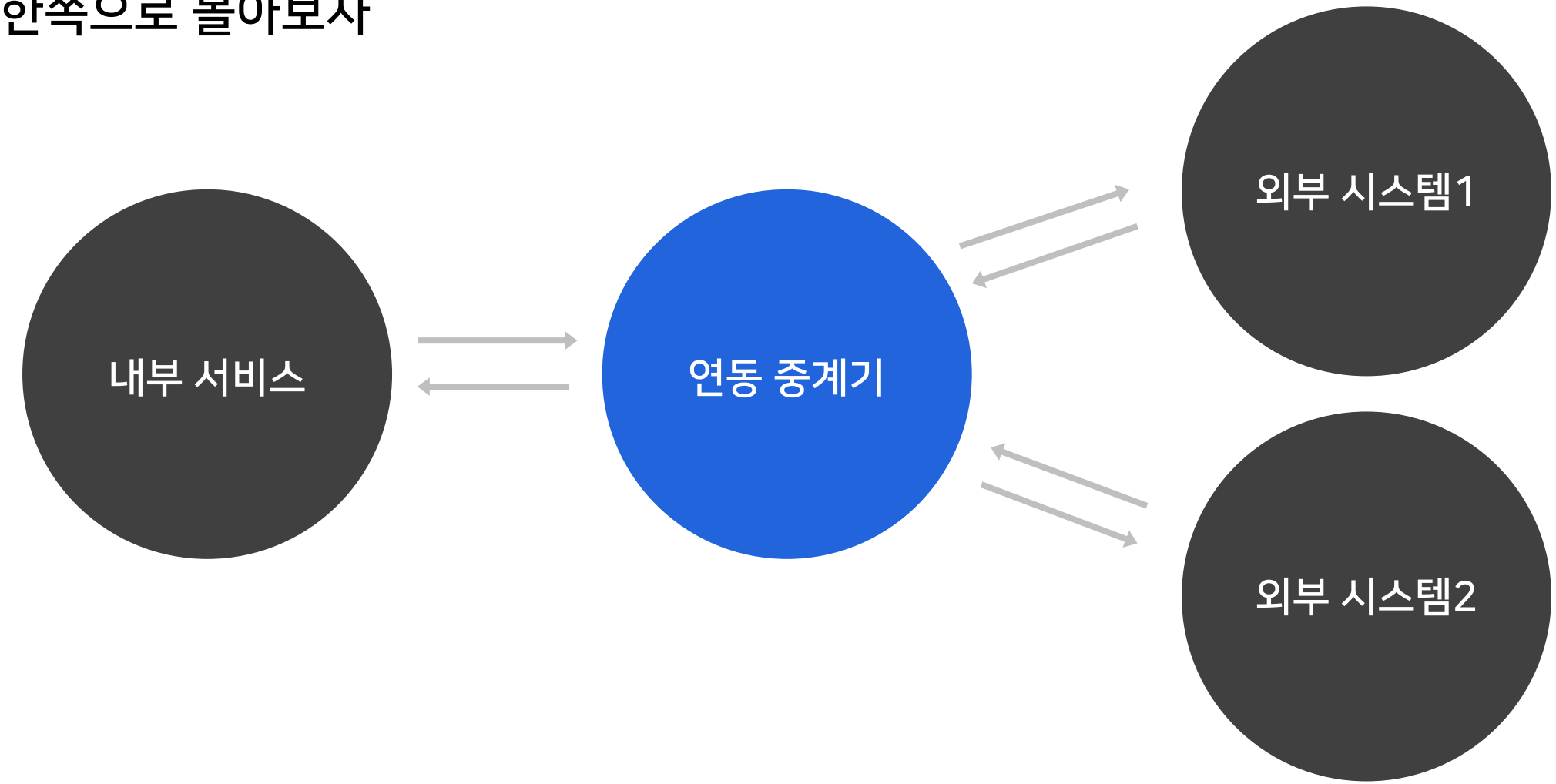
이걸 왜 만들었을까

외부 시스템과의 연동 방법은 어떤 것들이 있을까?



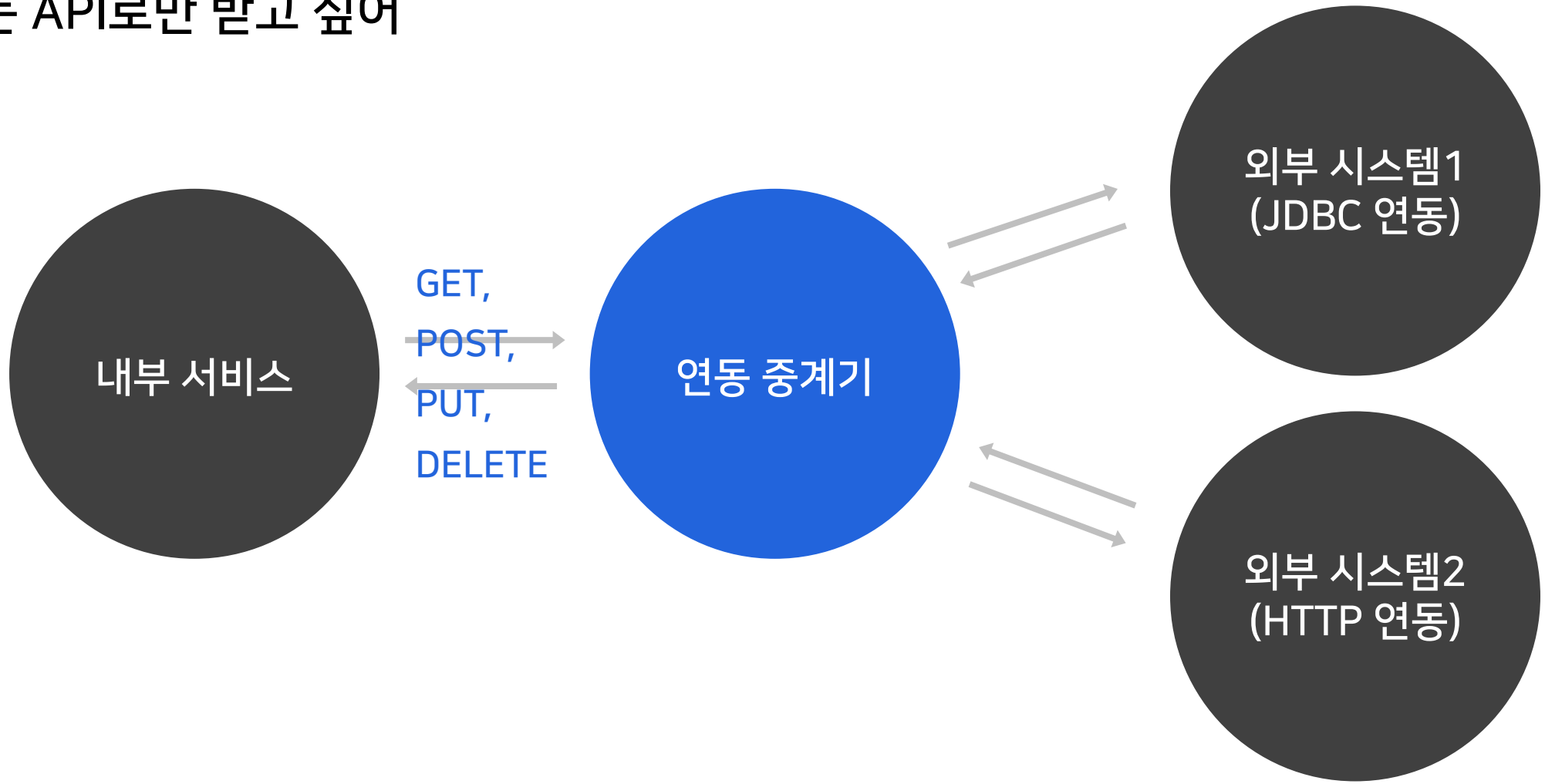
이걸 왜 만들었을까

일단 한쪽으로 몰아보자



이걸 왜 만들었을까

우리는 API로만 받고 싶어

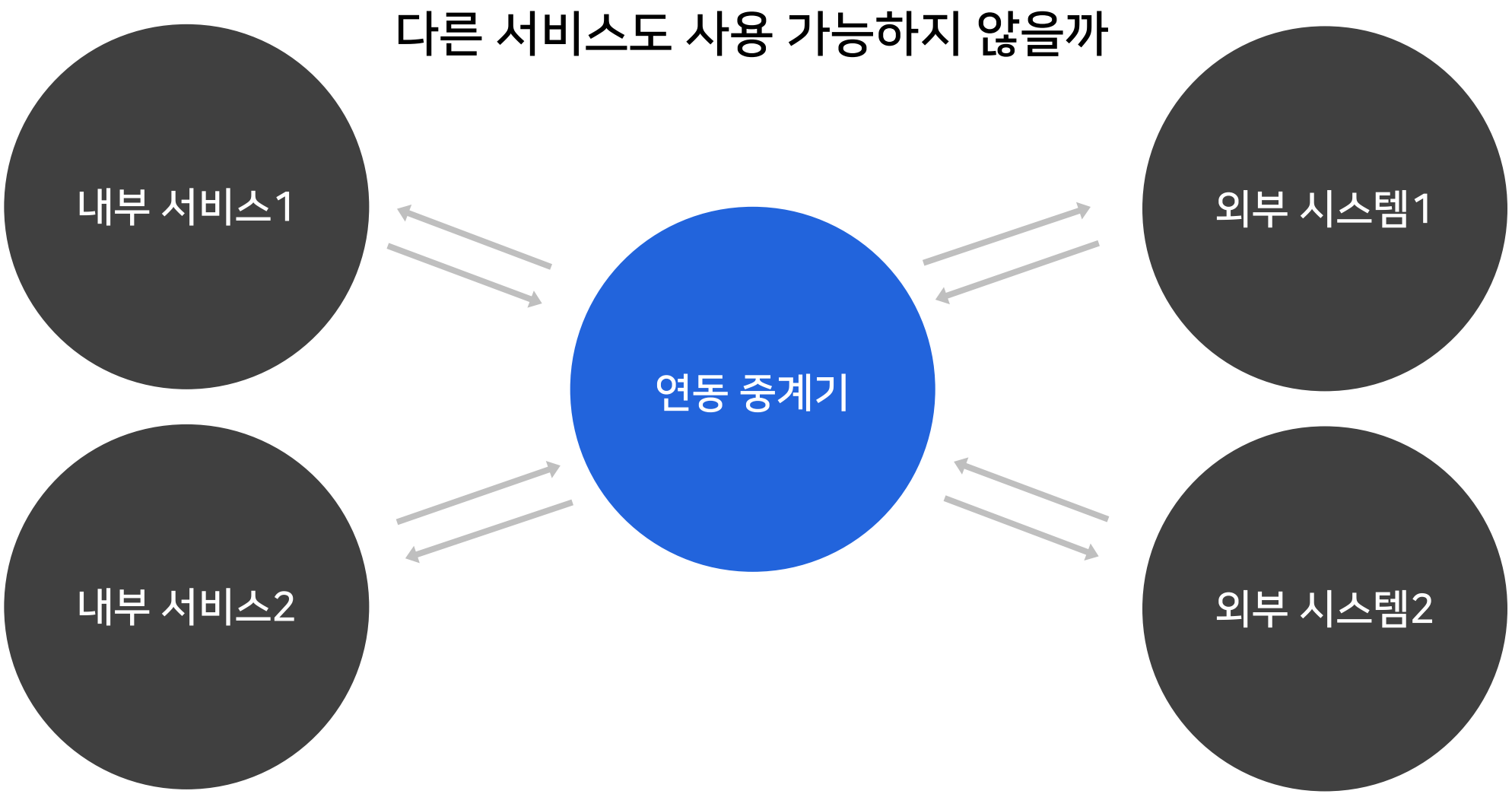


이걸 왜 만들었을까

내부 서비스의 경우 대응이 쉽다.

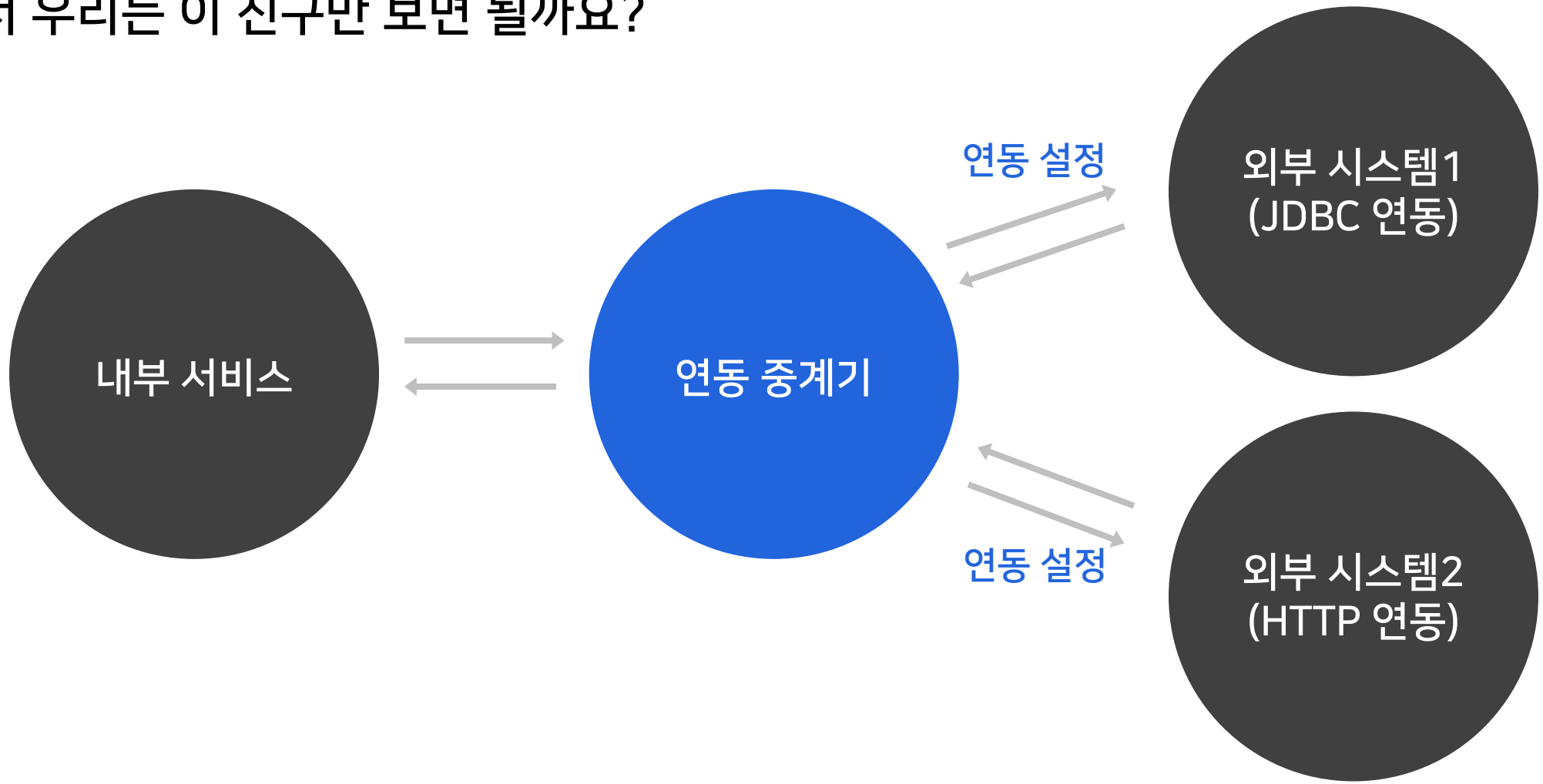
```
public class ProxyResponse {

  private ProxyResponseHeader header;
  private ProxyResponseResult result;
}

class ProxyResponseHeader {
…
}

class ProxyReponseResult {
…
}
```

이걸 왜 만들었을까



이걸 왜 만들었을까

그래서 우리는 이 친구만 보면 될까요?

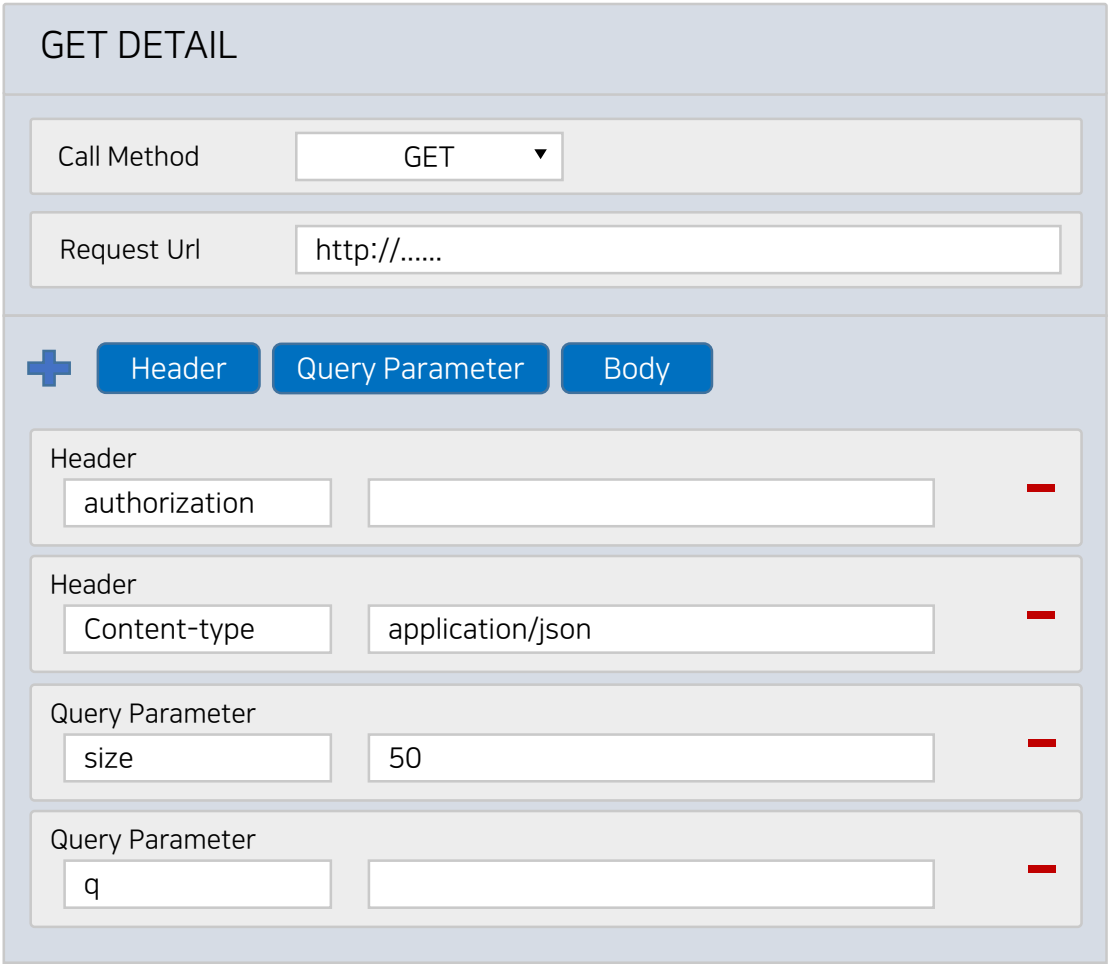


이걸 왜 만들었을까

양방향에 필요한 화면(view)을 제공

이걸 왜 만들었을까

외부 시스템



이걸 왜 만들었을까

내부 서비스

API SPEC

Request URL

http://.....

Request Method

GET

Header

name	required	default
authorization	true	
content-type	true	application/json

Query Parameter

name	required	default
size	false	50
q	false	

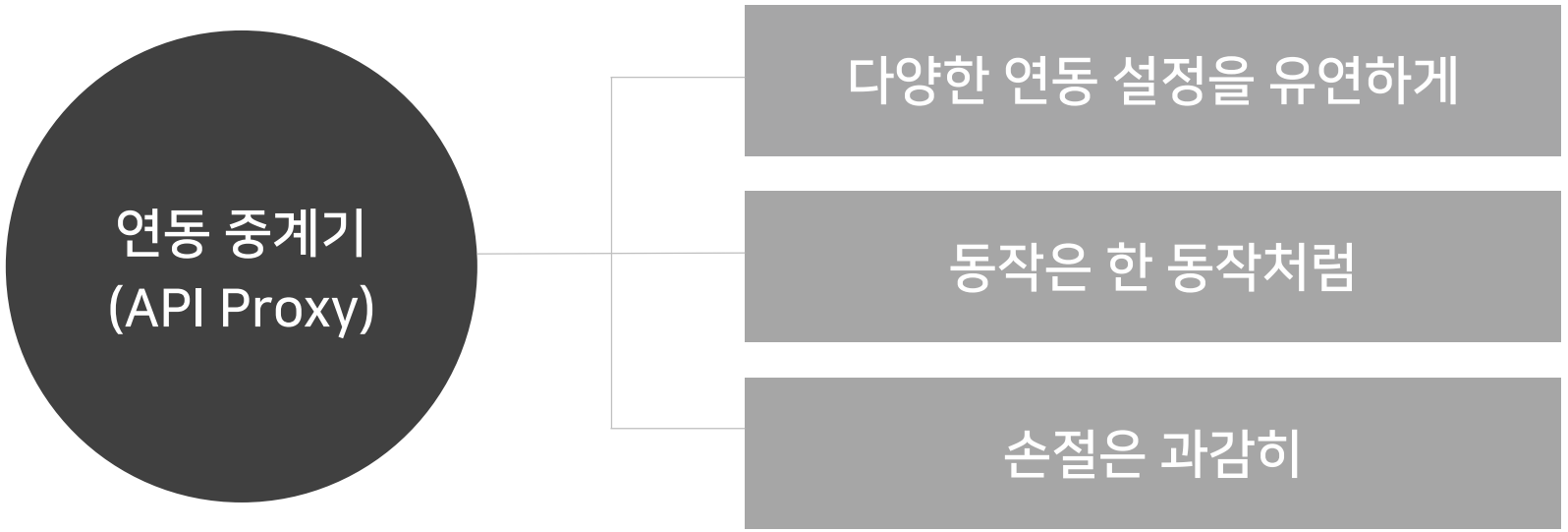
Response Body

```
{
    "header": {
        "resultCode": 200,
        "resultMessage": "",
        "isSuccessful": true
    },
    "result": {
        /*  */
    }
}
```

HTTP 응답코드

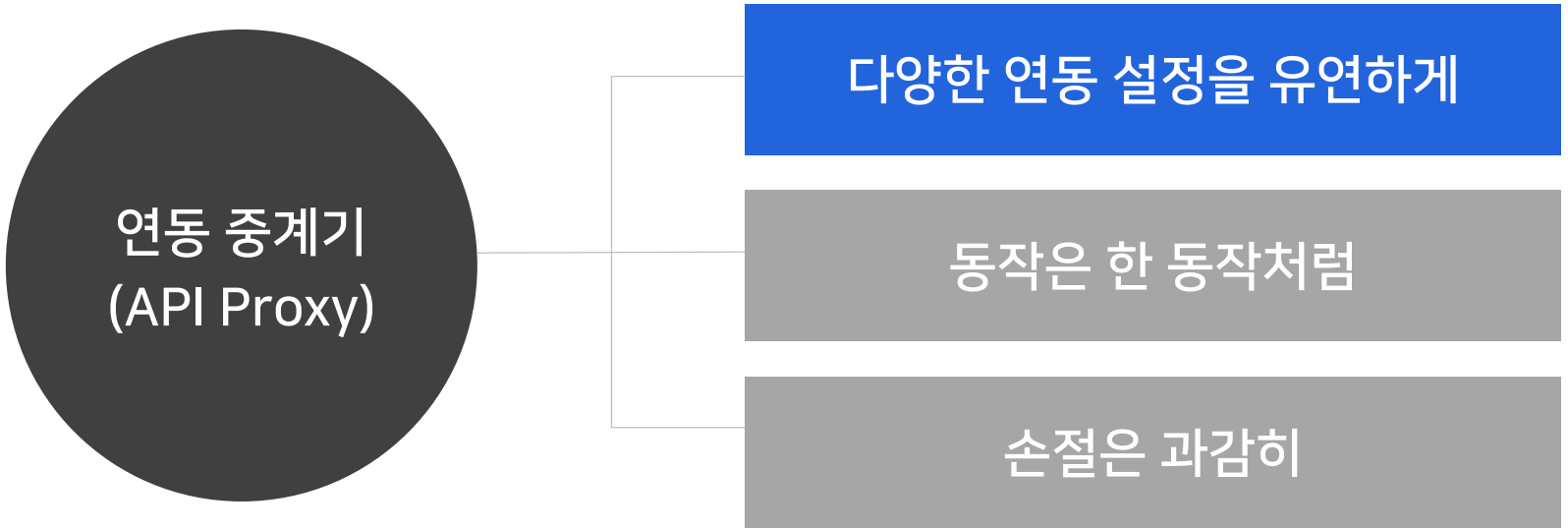
- 200 성공
- 400 Bad Request
- 401 Unauthorized
- 404 조회결과 없음
- 500 서버오류 (문의 부탁드립니다.)

이걸 왜 만들었을까



어떻게 만들었을까

어떻게 만들었을까



어떻게 만들었을까: 다양한 연동 설정을 유연하게

연동 방법을 하나의 모듈로 분류

어떻게 만들었을까: 다양한 연동 설정을 유연하게

모듈마다 입력받는 정보가 다름

```
Class.forName(driverClassName);
Connection conn =
DriverManager.getConnection(jdbcUrl,
username, password);

PreparedStatement ps =
conn.prepareStateMent(query);
ResultSet rs = ps.executeQuery();
```

```
WebClient.builder()
        .baseUrl(url)
        .defaultHeaders(headers)
        .build()
        .post()
        .uri(uriBuilder ->
uriBuilder.query(queryString).build())
        .bodyValue(bodyValue)
        .retrieve()
        .bodyToMono(Response.class)
        .flatMap(Mono::just)
        .block();
```

어떻게 만들었을까: 다양한 연동 설정을 유연하게

모듈에 구애받지 않는 입력 정보가 필요

항목	값
database	MySQL, Oracle, Cub..
jdbcUrl	jdbc:mysql://…
username	……
password	*********
queryType	SELECT, UPDATE, ..
query	SELECT … FROM …
param	[…, …]

항목	값
METHOD	GET, POST, PUT, ..
url	https://...
queryString	[…, …]
header	[…, …]
body	{ … }

어떻게 만들었을까: 다양한 연동 설정을 유연하게

모듈에 구애받지 않는 입력 정보가 필요

항목	값	유형
database	MySQL, Oracle, Cub..	SELECTION
jdbcUrl	jdbc:mysql://…	RAW
username	.....	RAW
password	*********	RAW
queryType	SELECT, UPDATE, ..	SELECTION
query	SELECT … FROM …	RAW
param	[…, …]	KEY_VALUE

항목	값	유형
METHOD	GET, POST, PUT, ..	SELECTION
url	https://...	RAW
queryString	[…, …]	KEY_VALUE
header	[…, …]	KEY_VALUE
body	{ … }	RAW

어떻게 만들었을까: 다양한 연동 설정을 유연하게

모듈에 구애받지 않는 입력 정보가 필요

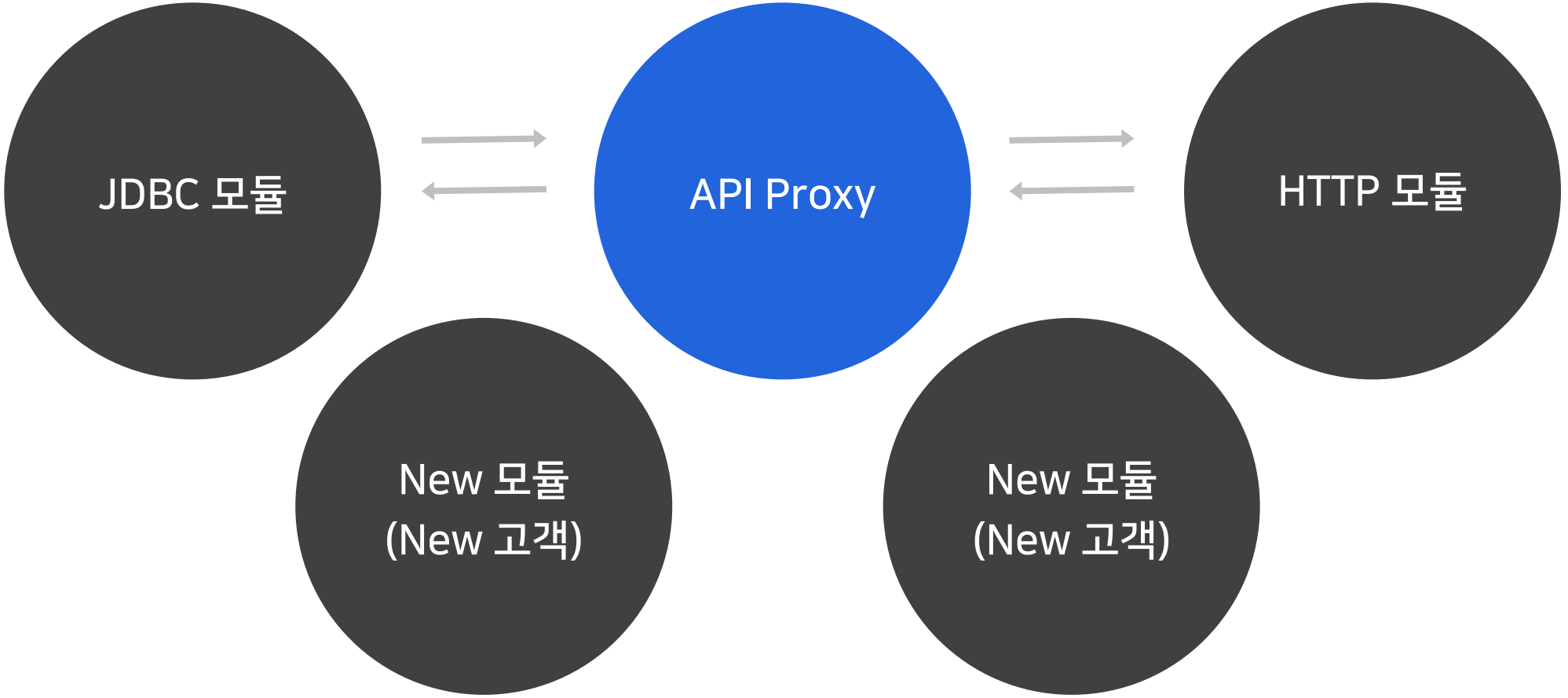
ITEM	
NAME	STRING
TYPE	SELECTION, RAW, KEY_VALUE(MAP)
RAW	STRING
SEL_CD	ENUM
KEY	STRING
VALUE	STRING

어떻게 만들었을까: 다양한 연동 설정을 유연하게

추후 다른 모듈이 추가되면?

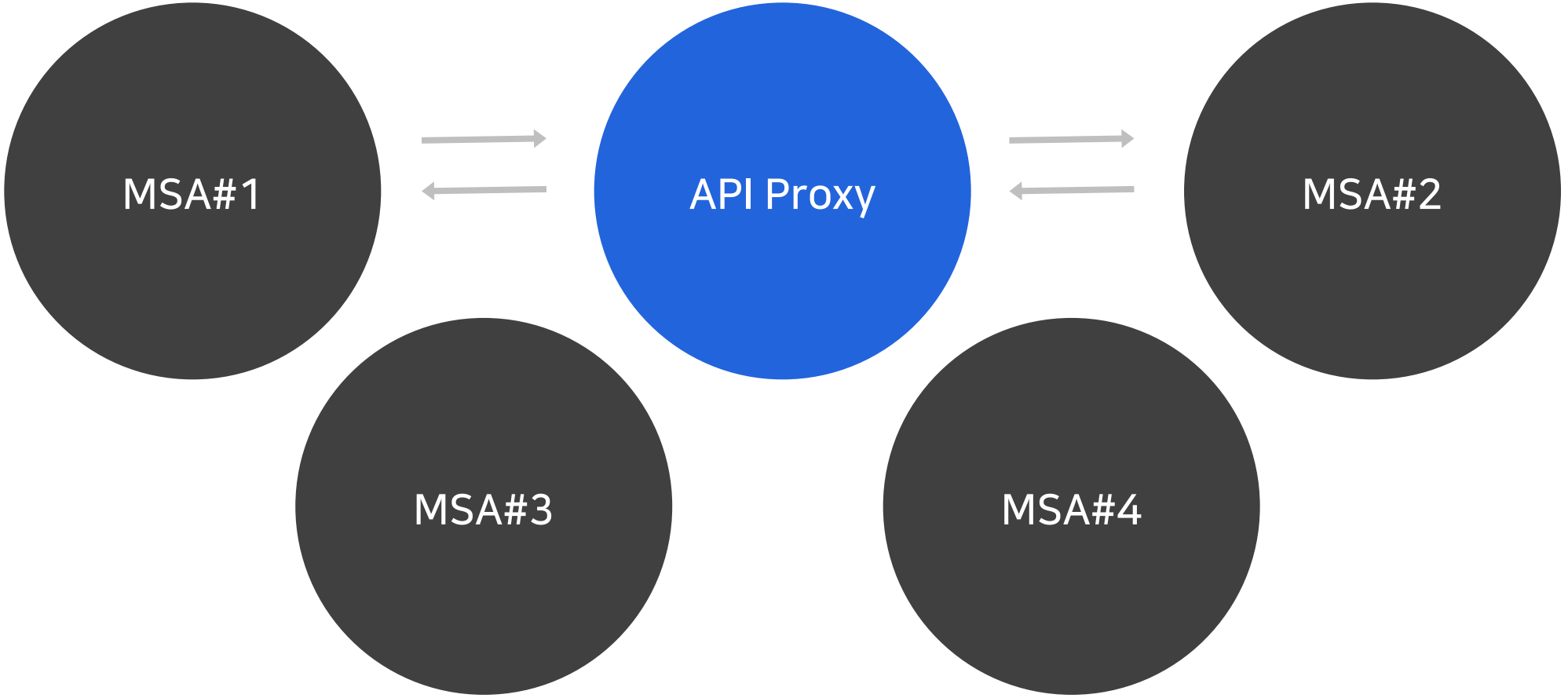
어떻게 만들었을까: 다양한 연동 설정을 유연하게

추후 다른 모듈이 추가되면?

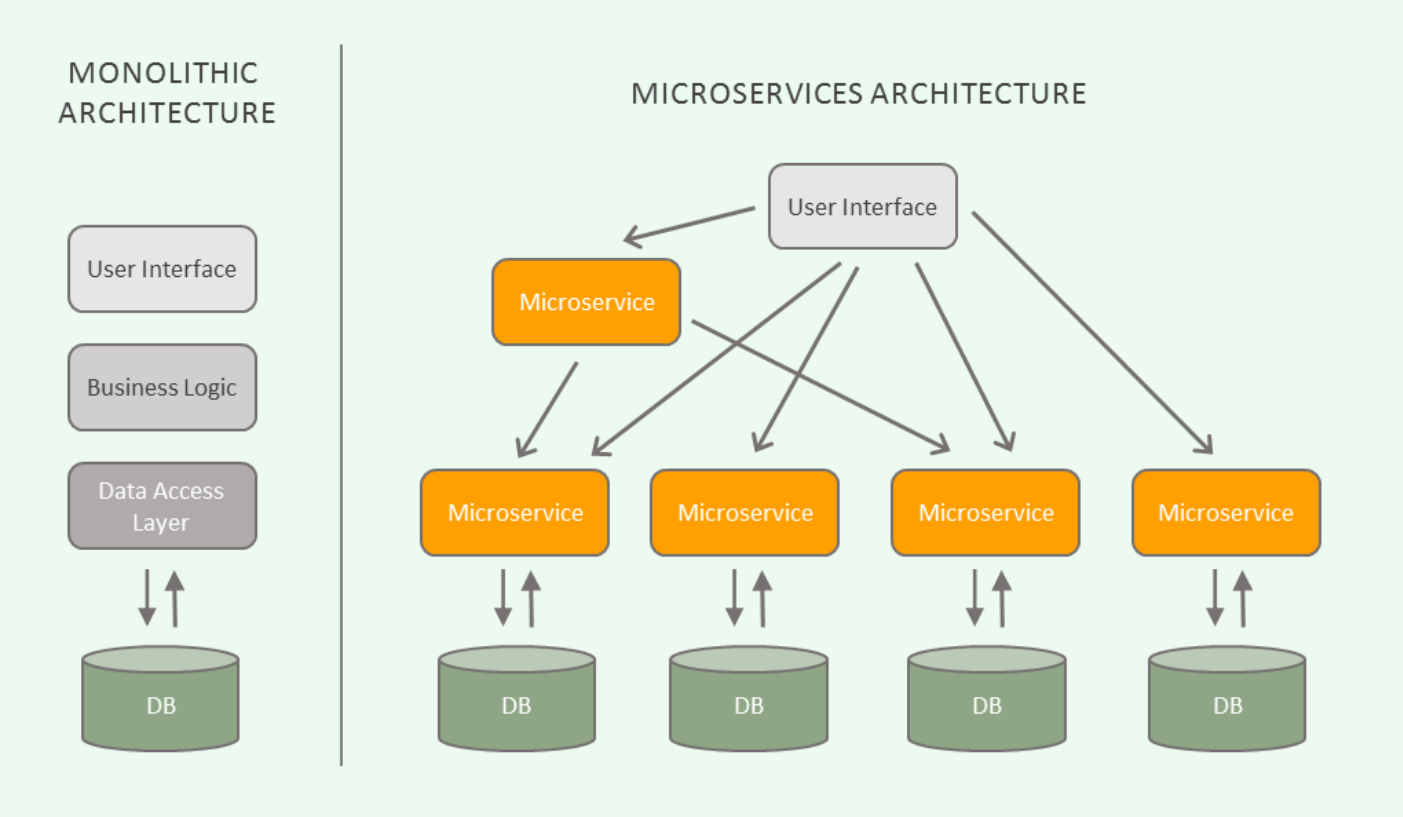


어떻게 만들었을까: 다양한 연동 설정을 유연하게

MSA(Microservice Architecture) 형태로 구현



어떻게 만들었을까: 다양한 연동 설정을 유연하게



출처: https://dzone.com/articles/what-are-microservices-actually

어떻게 만들었을까: 다양한 연동 설정을 유연하게

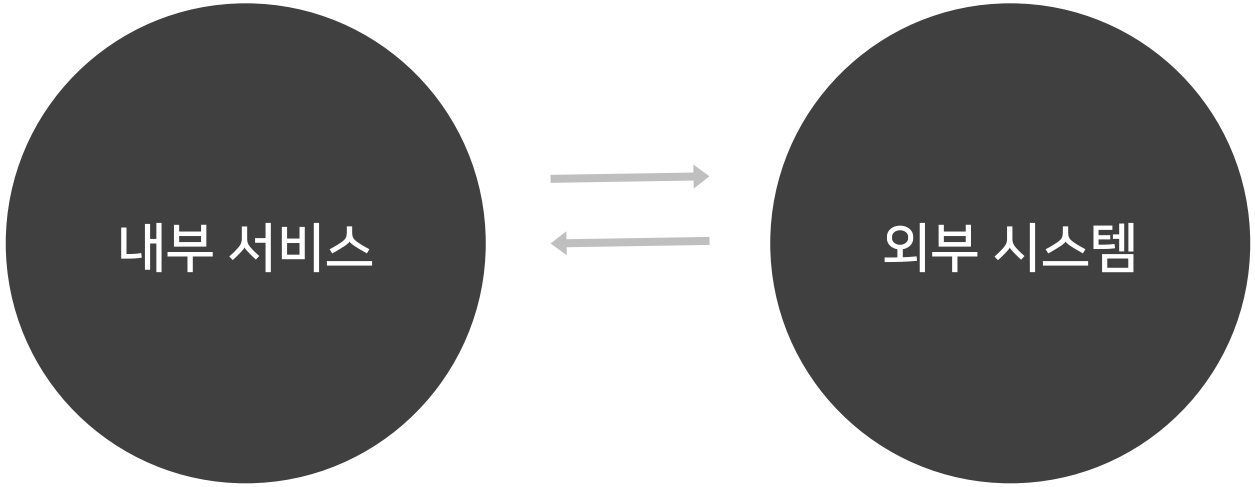
MSA 장점(vs Monolithic)

- 모듈별로 개발 및 형상관리가 가능하여 생산성이 증가
- 독립적인 빌드, 배포가 가능
- 기능 확장이 유연
- 가벼움

어떻게 만들었을까

어떻게 만들었을까: 동작은 한 동작처럼

동작도 유연할까?

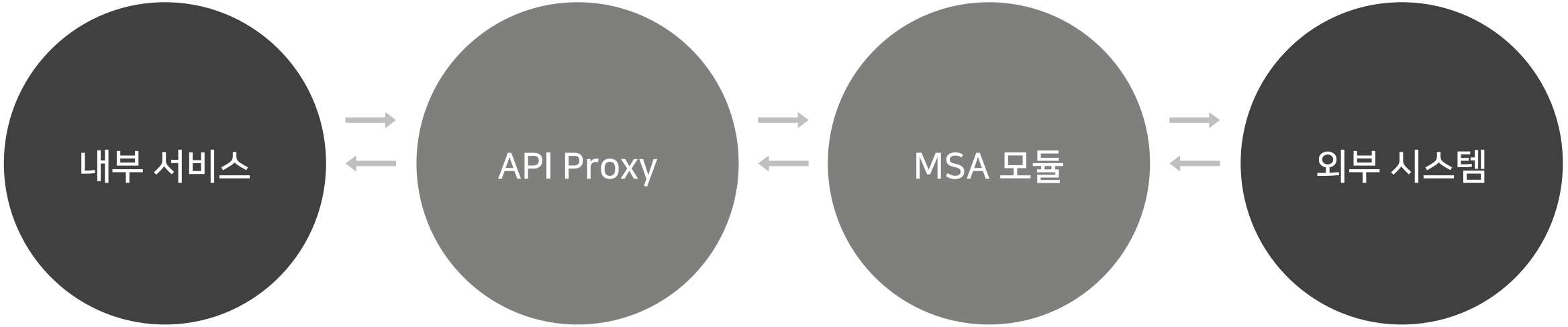


어떻게 만들었을까: 동작은 한 동작처럼

처음 구상한 그림

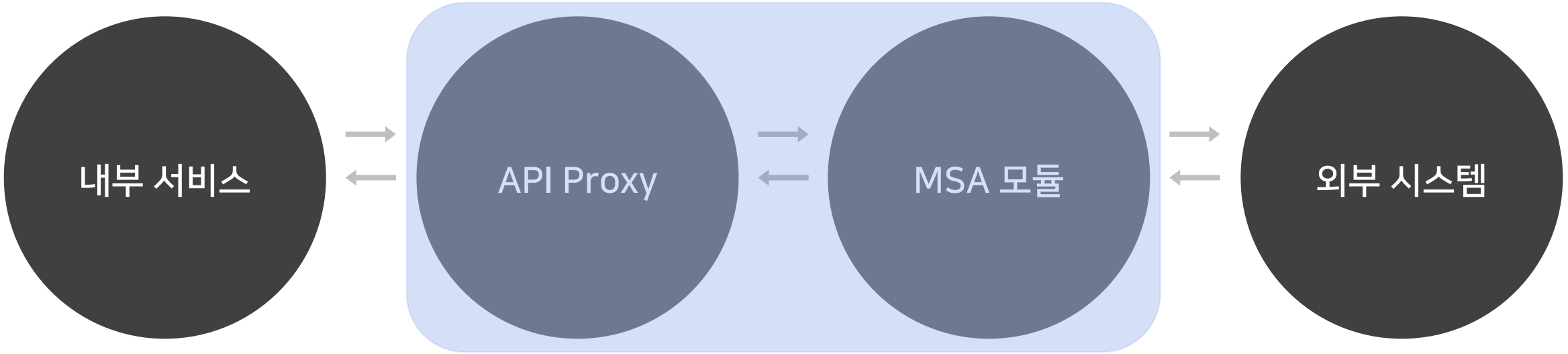
어떻게 만들었을까: 동작은 한 동작처럼

의도야 어찌 됐던 늘어져 버렸다.



어떻게 만들었을까: 동작은 한 동작처럼

의도야 어찌 됐던 늘어져 버렸다.



어떻게 만들었을까: 동작은 한 동작처럼

출처: https://grpc.io/

어떻게 만들었을까: 동작은 한 동작처럼

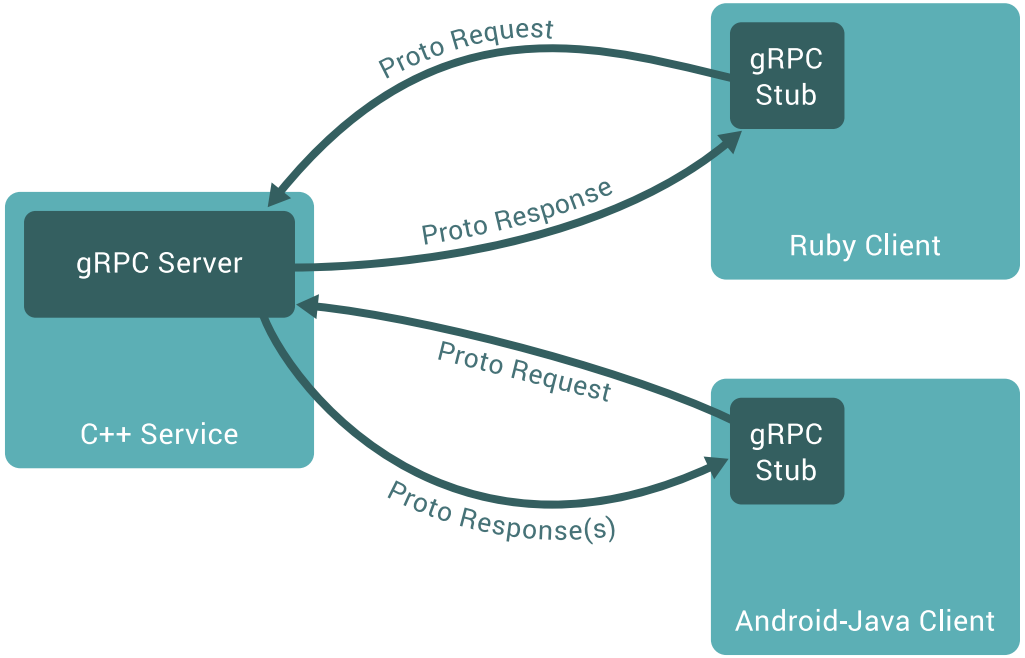
왜 gRPC인가?

- 일단, HTTP API보다는 빨라야 함

어떻게 만들었을까: 동작은 한 동작처럼

왜 gRPC인가?

- 다양한 프로그래밍 언어의 양방향 스텁(stub)을 지원



출처: https://grpc.io/

어떻게 만들었을까: 동작은 한 동작처럼

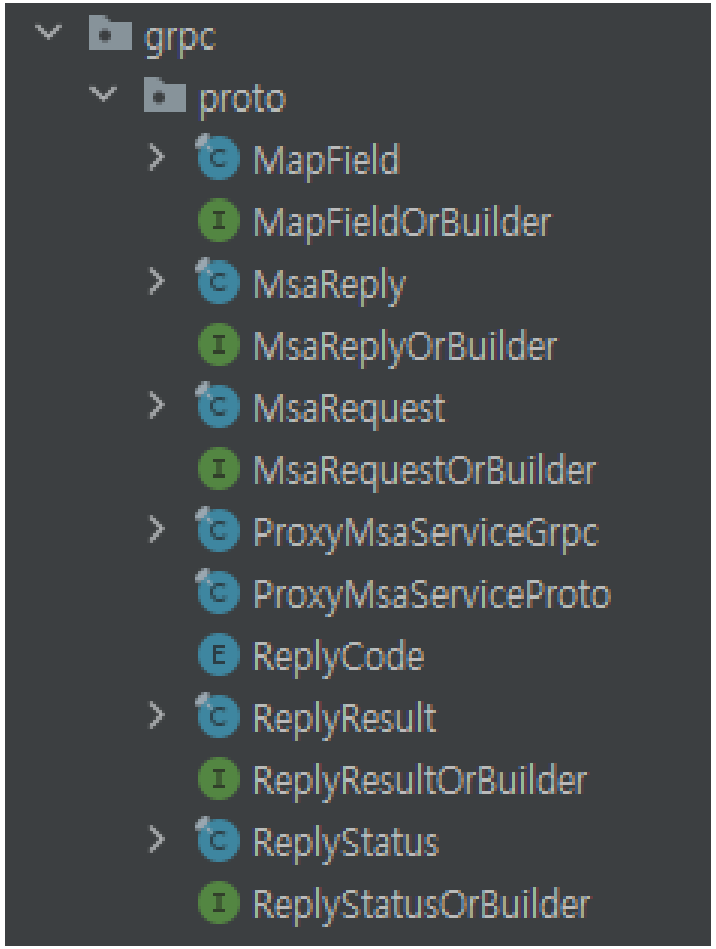
왜 gRPC인가?

```
message MsaRequest {
  int64 apiId = 1; // field number
  repeated string stringParams = 2;
  repeated MapField mapParams = 3;
}

message MsaReply {
  ReplyStatus status = 1;
  ReplyResult result = 2;
}

service ProxyMsaService {
  rpc callMsa(MsaRequest) returns (MsaReply) {}
}
```

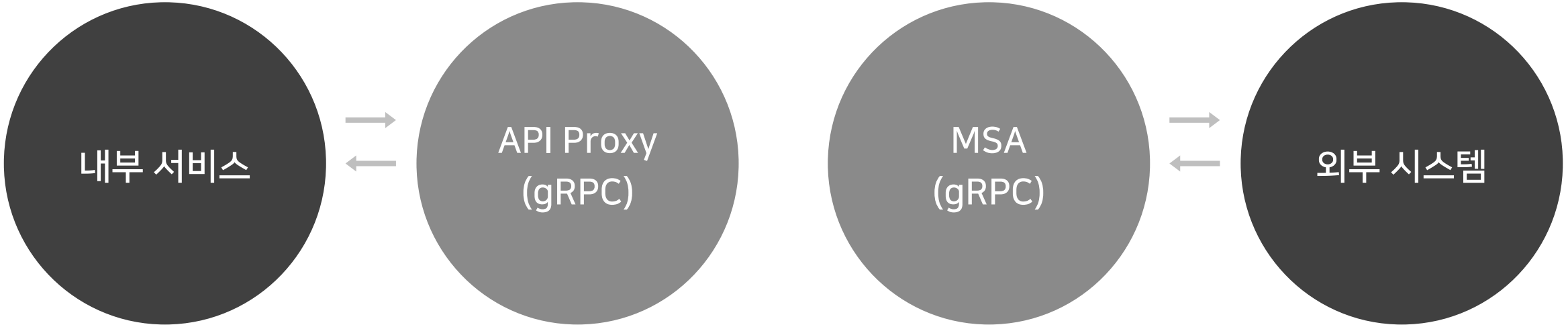
어떻게 만들었을까: 동작은 한 동작처럼

왜 gRPC인가?

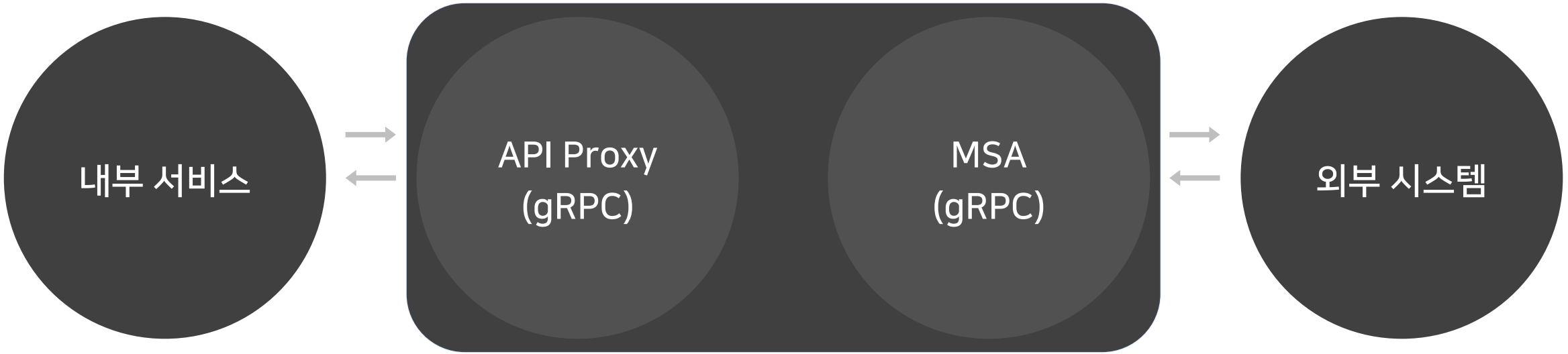
```
message MsaRequest {
  int64 apiId = 1; // field number
  repeated string stringParams = 2;
  repeated MapField mapParams = 3;
}

message MsaReply {
  ReplyStatus status = 1;
  ReplyResult result = 2;
}

service ProxyMsaService {
  rpc callMsa(MsaRequest) returns (MsaReply) {}
}
```



어떻게 만들었을까: 동작은 한 동작처럼



어떻게 만들었을까: 동작은 한 동작처럼



실제 적용은 어땠을까

실제 적용은 어땠을까

실제 적용은 어땠을까

역시 순탄하지 않다.

- Stored Procedure를 어떻게 작성했는지에 따라 결과가 다르다.
- 예측이 어렵고, 범용적인 처리가 어렵다.
- 결과가 있거나 없거나 한다.

실제 적용은 어땠을까

결과가 있거나 없거나 한다

- 아래 코드는 결과가 없으면 에러가 발생

```
CallableStatement cs = ...
...
ResultSet rs = cs.executeQuery();
```

출처: https://grpc.io/

실제 적용은 어땠을까

결과 유형을 관리

API DETAIL	
API ID	...
MODULE ID	JDBC
...	...
...	...
RESULT TYPE	EMPTY REQUIRED OPTIONAL_SUCCESS OPTIONAL_FAILURE

실제 적용은 어땠을까

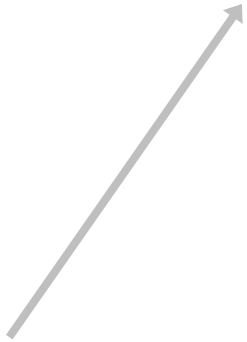
저희 상태 코드는 다릅니다.

내부 서비스	외부 시스템
A	1
B	2
C	3
D	4
E	5
F	6

실제 적용은 어땠을까

저희 상태 코드는 다릅니다.

항목	유형	변환 세트
database	SELECTION	
jdbcUrl	RAW	
username	RAW	
password	RAW	
queryType	SELECTION	
query	RAW	
param	KEY_VALUE	REPLACE_SET



FROM	TO
A	1
B	2
C	3
D	4
E	5
F	6

실제 적용은 어땠을까

"거기 잘 지내고 계신가요? 대답 좀.."

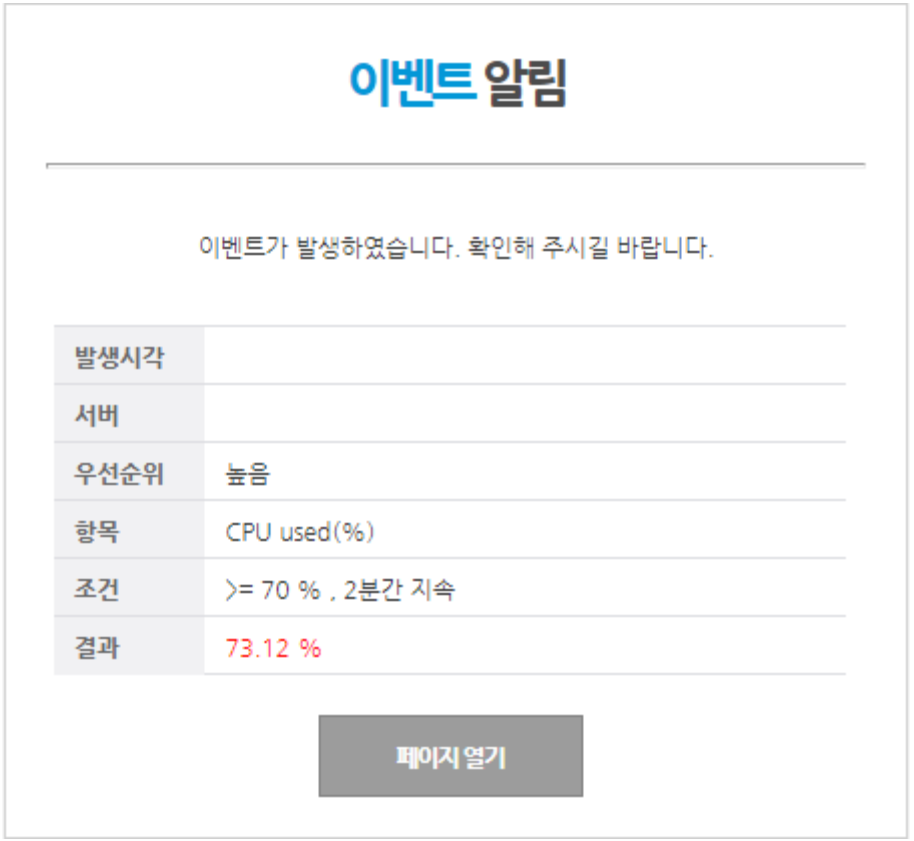
출처: https://movie.naver.com/movie/bi/mi/photoView.naver?code=19031

실제 적용은 어땠을까

응답이 오지 않는다..

- 원인 모를 지독한 타임아웃이 지속된다.
- 오류 메시지도 지속적으로 보내온다.

실제 적용은 어땠을까


이벤트 알림
이벤트가 발생하였습니다. 확인해 주시길 바랍니다.
발생시각
서버
우선순위
높음
항목
CPU used(%)
조건
>= 70 % , 2분간 지속
결과
73.12 %
페이지 열기

실제 적용은 어땠을까

조치를 마냥 기다릴 수 없다.

- 사용자들의 지속된 요청을 멈출 수 없다.
- 오류와 타임아웃 스레드로 인해 시스템 부하가 쌓여간다.
- 그냥 시스템 내리고 싶다.

실제 적용은 어땠을까

장애가 확산될 수 있다.

실제 적용은 어땠을까

장애가 확산될 수 있다.

실제 적용은 어땠을까

실제 적용은 어땠을까: 손절은 과감히

실제 적용은 어땠을까: 손절은 과감히

회로 차단기(circuit breaker) 패턴

- 말 그대로 문제가 되는 회로를 차단해서 시스템을 보호
- 장애 발생 시 FALLBACK 기능을 수행

실제 적용은 어땠을까: 손절은 과감히

NETFLIX

실제 적용은 어땠을까: 손절은 과감히

Hystrix Status

Hystrix is no longer in active development, and is currently in maintenance mode.

Hystrix (at version 1.5.18) is stable enough to meet the needs of Netflix for our existing applications. Meanwhile, our focus has shifted towards more adaptive implementations that react to an application's real time performance rather than pre-configured settings (for example, through adaptive concurrency limits). For the cases where something like Hystrix makes sense, we intend to continue using Hystrix for existing applications, and to leverage open and active projects like resilience4j for new internal projects. We are beginning to recommend others do the same.

실제 적용은 어땠을까: 손절은 과감히

resilience4j

- 여러 코어 모듈을 지원
  - Circuit Breaker, Bulkhead, Retry…
- Apache License 2.0

실제 적용은 어땠을까: 손절은 과감히

resilience4j 설정

```
resilience4j.circuitbreaker:
  configs:
    default:
      minimumNumberOfCalls: 10
      failureRateThreshold: 30
…
```

실제 적용은 어땠을까: 손절은 과감히

resilience4j 설정

```
@CircuitBreaker(fallbackMethod = "fallback")
public ProxyResponse testCircuitBreaker() {
…
}

public ProxyResponse fallback(Exception e) {
…
}

public ProxyResponse fallback(CallNotPermittedException e) {
…
}
```

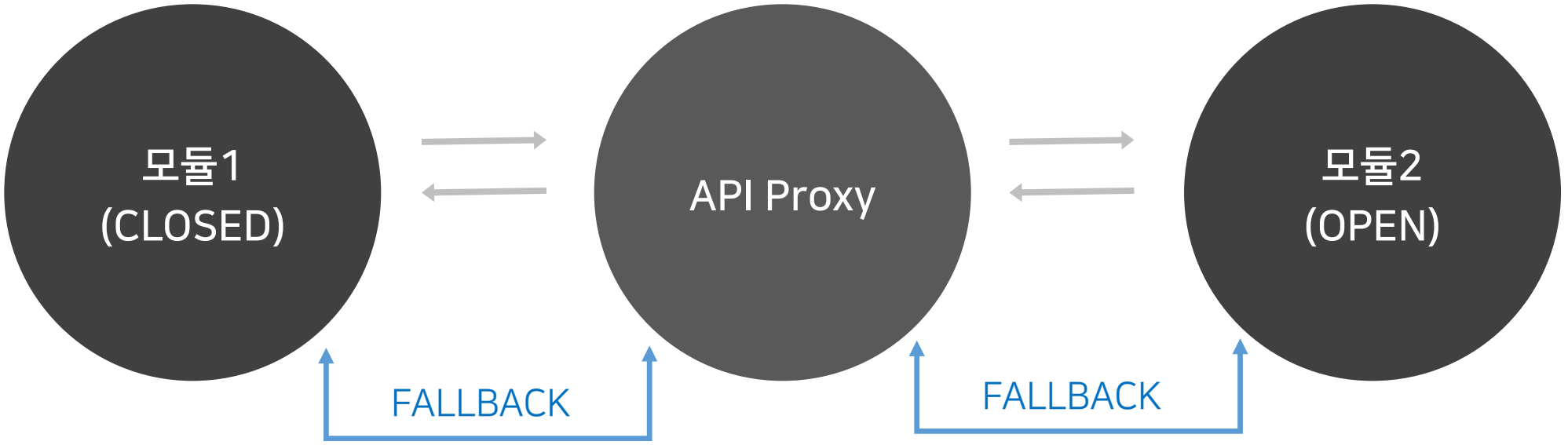
실제 적용은 어땠을까: 손절은 과감히

resilience4j 설정

```
@CircuitBreaker(fallbackMethod = "fallback")
public ProxyResponse testCircuitBreaker() {
…
}

public ProxyResponse fallback(Exception e) {   서킷이 CLOSED 일 때
…
}

public ProxyResponse fallback(CallNotPermittedException e) {   서킷이 OPEN 일 때
…
}
```

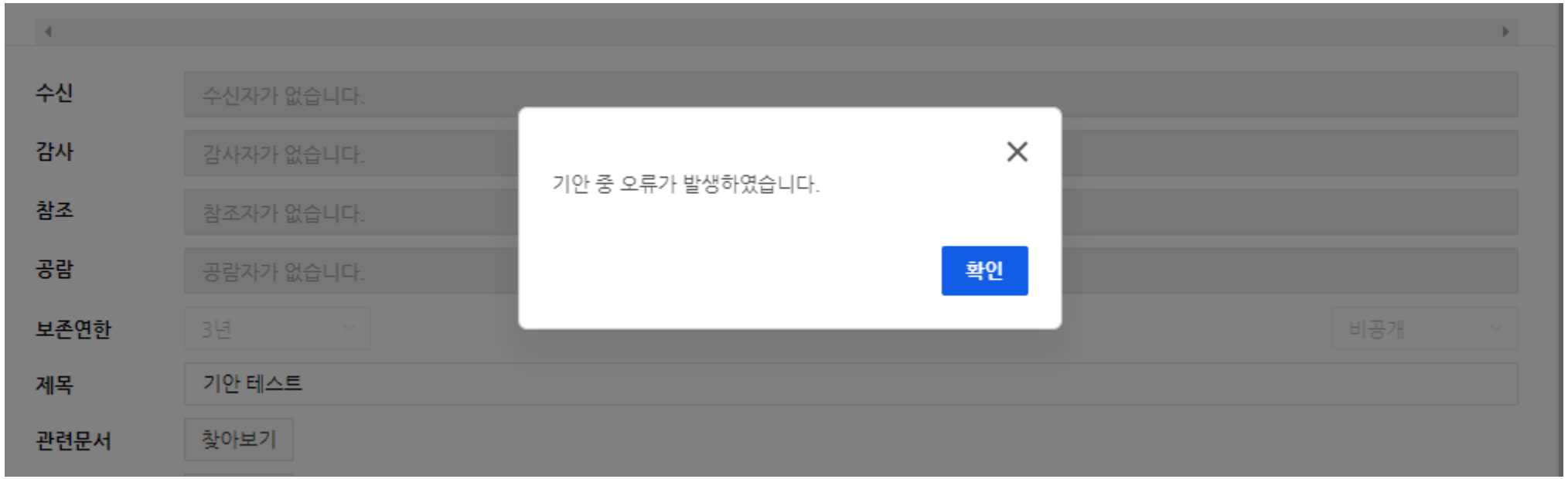
실제 적용은 어땠을까: 손절은 과감히

둘 다 오류가 난다고 가정했을 때



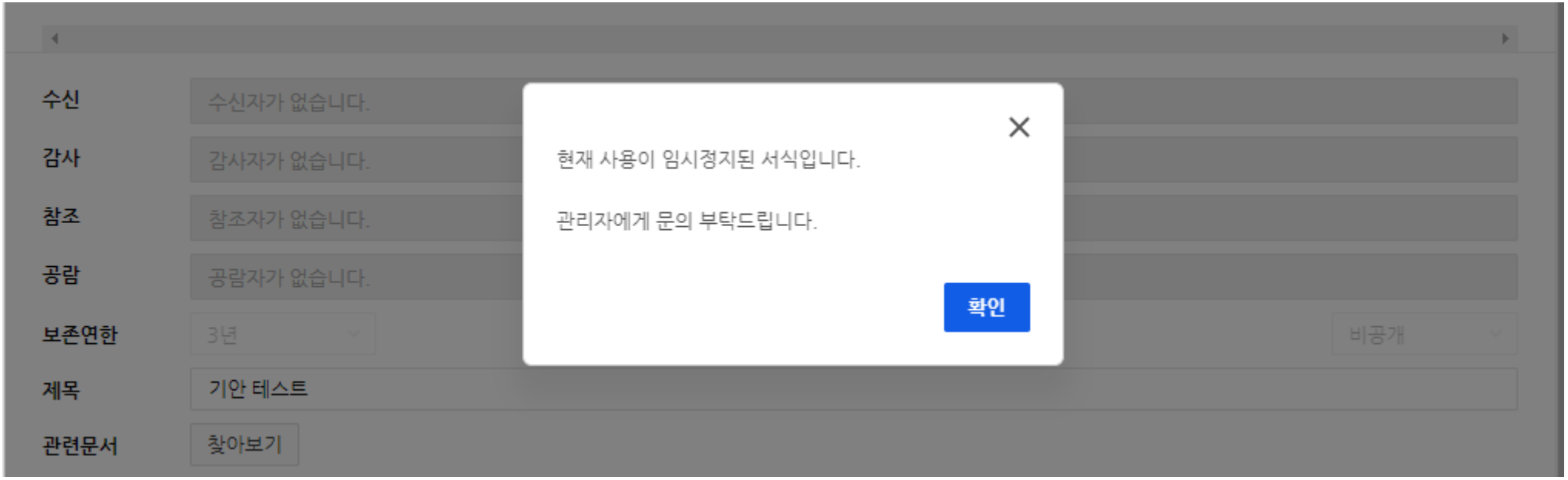
실제 적용은 어땠을까: 손절은 과감히

CLOSED



실제 적용은 어땠을까: 손절은 과감히

OPEN



더 해야 할 일은 무엇일까

더 해야 할 일은 무엇일까

MSA 단점(vs Monolithic)

- 성능
- 로직 추적
- 통합 테스트

더 해야 할 일은 무엇일까

로직 추적

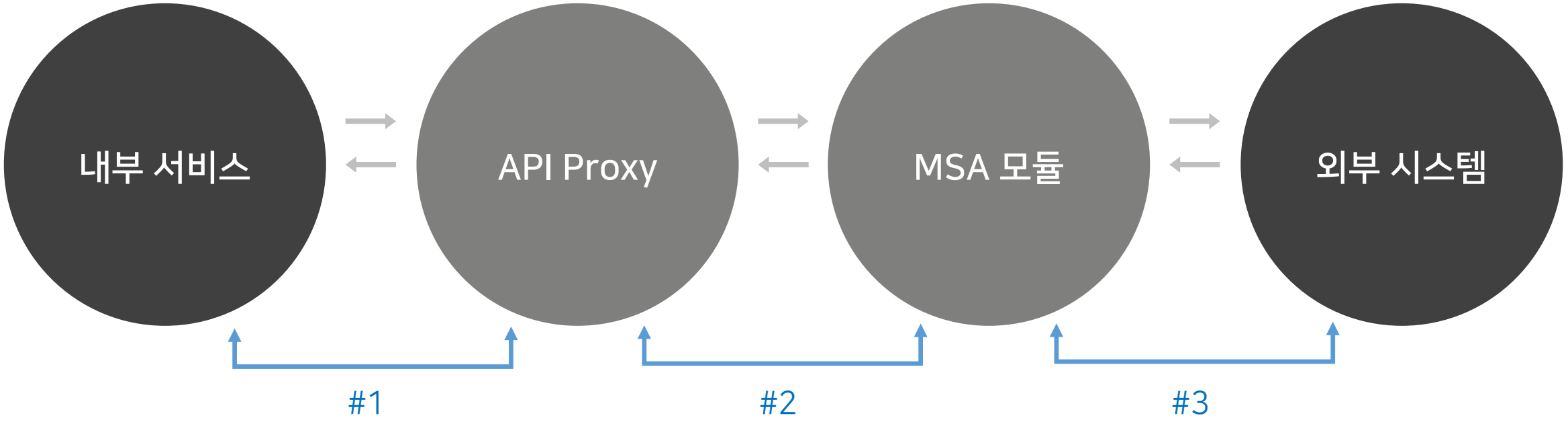
```
message MsaRequest {
  int64 apiId = 1; // field number
  repeated string stringParams = 2;
  repeated MapField mapParams = 3;
  string requestId = 4;
}
```

- 요청마다 고유의 ID를 부여해서 추적
- 해당 ID별 로그 관리

더 해야 할 일은 무엇일까

통합 테스트

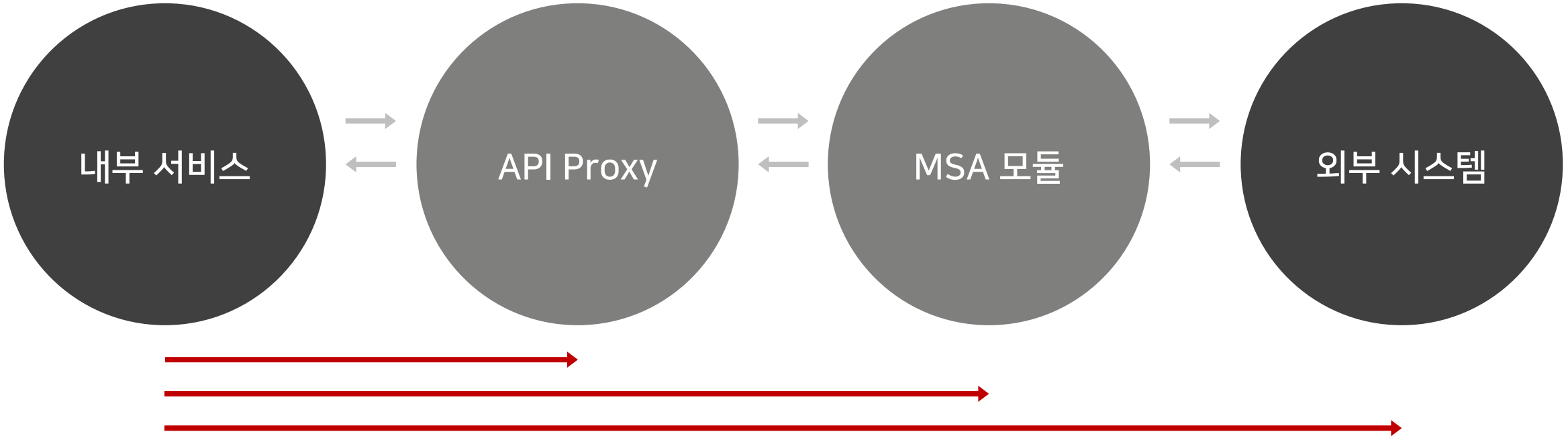
- 특정 단계까지 테스트하기 위한 리소스가 제법 크다.



더 해야 할 일은 무엇일까

통합 테스트

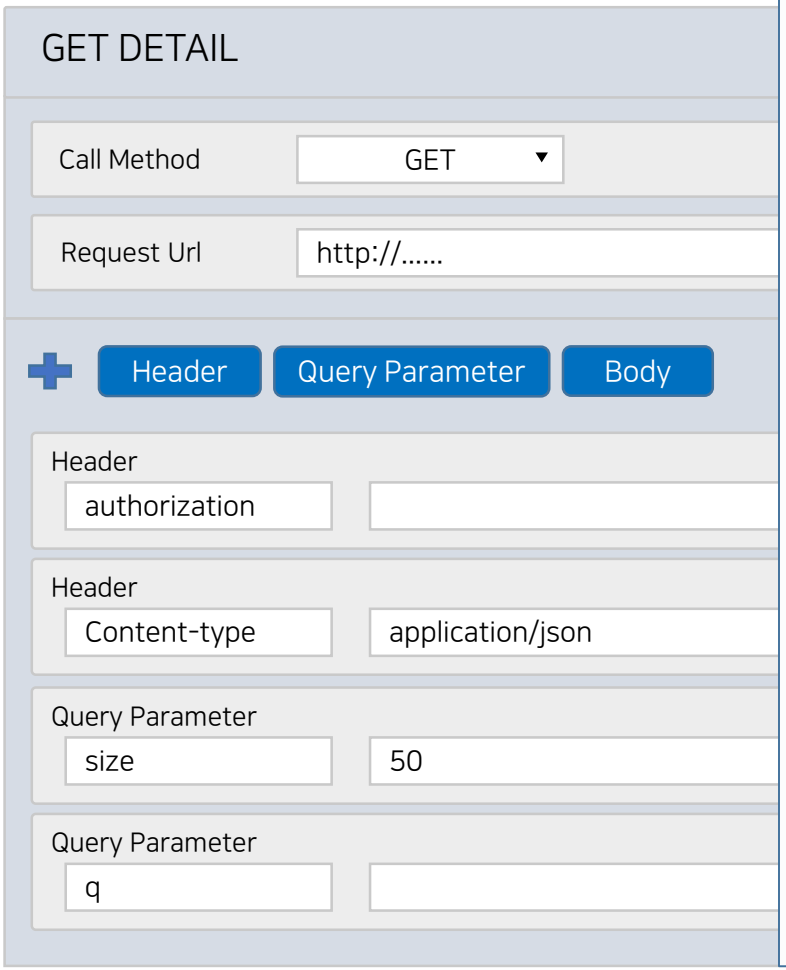
- 단계별로 테스트가 가능한 더미 데이터 생성



더 해야 할 일은 무엇일까

사용자가 화면 제공

- 연동 설정 화면
- API 명세 화면



마치며

마치며

이 시간에도 API Proxy는 열일 중

마치며

원 코인만 더..

NHN FORWARD