웹 애플리케이션 이해

- 데스크톱 애플리케이션
- 클라이언트·서버 애플리케이션
- 다중 클라이언트 요청 처리
- 웹 애플리케이션
- 클라이언트·서버 아키텍처의 진화

1.1 데스크톱 애플리케이션

데스트톱 애플리케이션

특징

- **PC에 설치한 후 실행**
- 사용자 화면 출력, 업무 관련 작업 실행, 데이터 처리를 모두 PC에서 수행

PC

데스트톱 애플리케이션

특징

- PC에 설치한 후 실행
- **사용자 화면 출력**, 업무 관련 작업 실행, 데이터 처리를 모두 PC에서 수행

PC

✓ 프리젠테이션 로직

데스트톱 애플리케이션

특징

- PC에 설치한 후 실행
- 사용자 화면 출력, **업무 관련 작업 실행**, 데이터 처리를 모두 PC에서 수행

PC

✓ 프리젠테이션 로직

✓ 비즈니스 로직

데스트톱 애플리케이션

특징

- PC에 설치한 후 실행
- 사용자 화면 출력, 업무 관련 작업 실행, **데이터 처리**를 모두 PC에서 수행

PC

- ✓ 프리젠테이션 로직
- ✓ 비즈니스 로직
- ✓ 데이터 로직

데스트톱 애플리케이션

특징

- PC에 설치한 후 실행
- 사용자 화면 출력, 업무 관련 작업 실행, 데이터 처리를 모두 PC에서 수행

PC

- ✓ 프리젠테이션 로직
- ✓ 비즈니스 로직
- ✓ 데이터 로직

문제점

- **배포가 번거롭다**
- 보안에 취약하다

데스트톱 애플리케이션

특징

- PC에 설치한 후 실행
- 사용자 화면 출력, 업무 관련 작업 실행, 데이터 처리를 모두 PC에서 수행

PC

- ✓ 프리젠테이션 로직
- ✓ 비즈니스 로직
- ✓ 데이터 로직

문제점

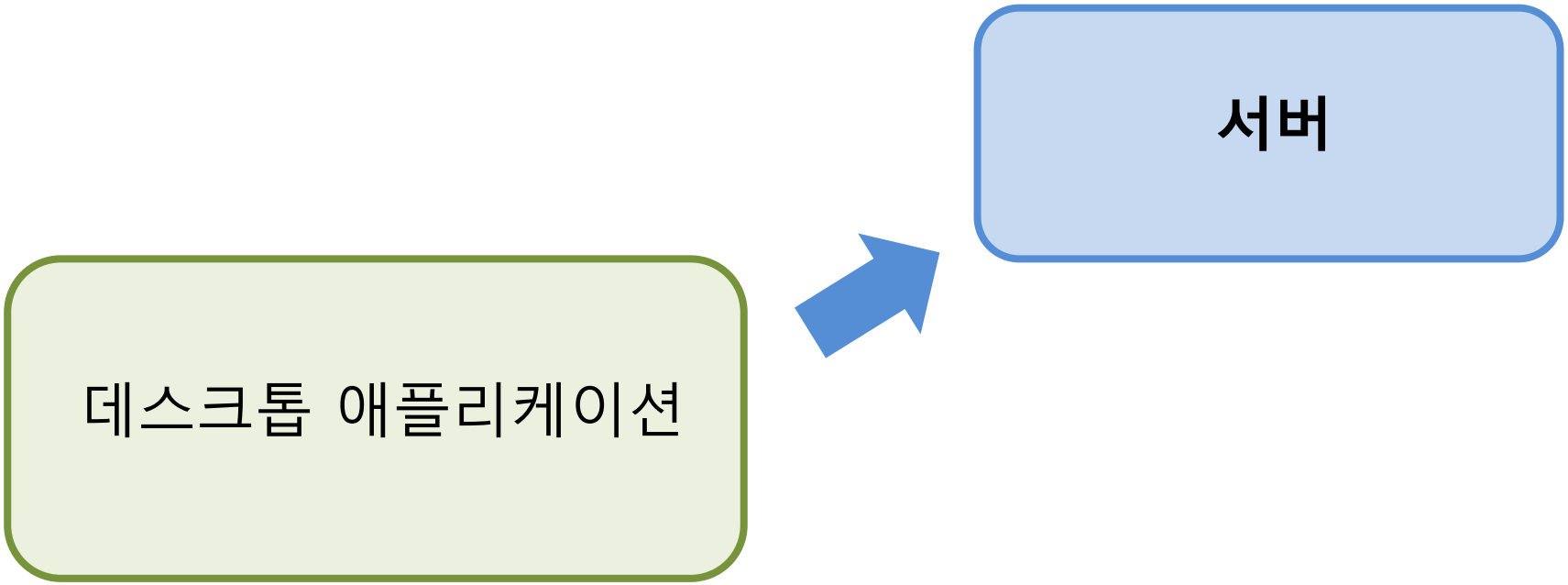
- 배포가 번거롭다
- **보안에 취약하다**

1.2 클라이언트·서버 애플리케이션

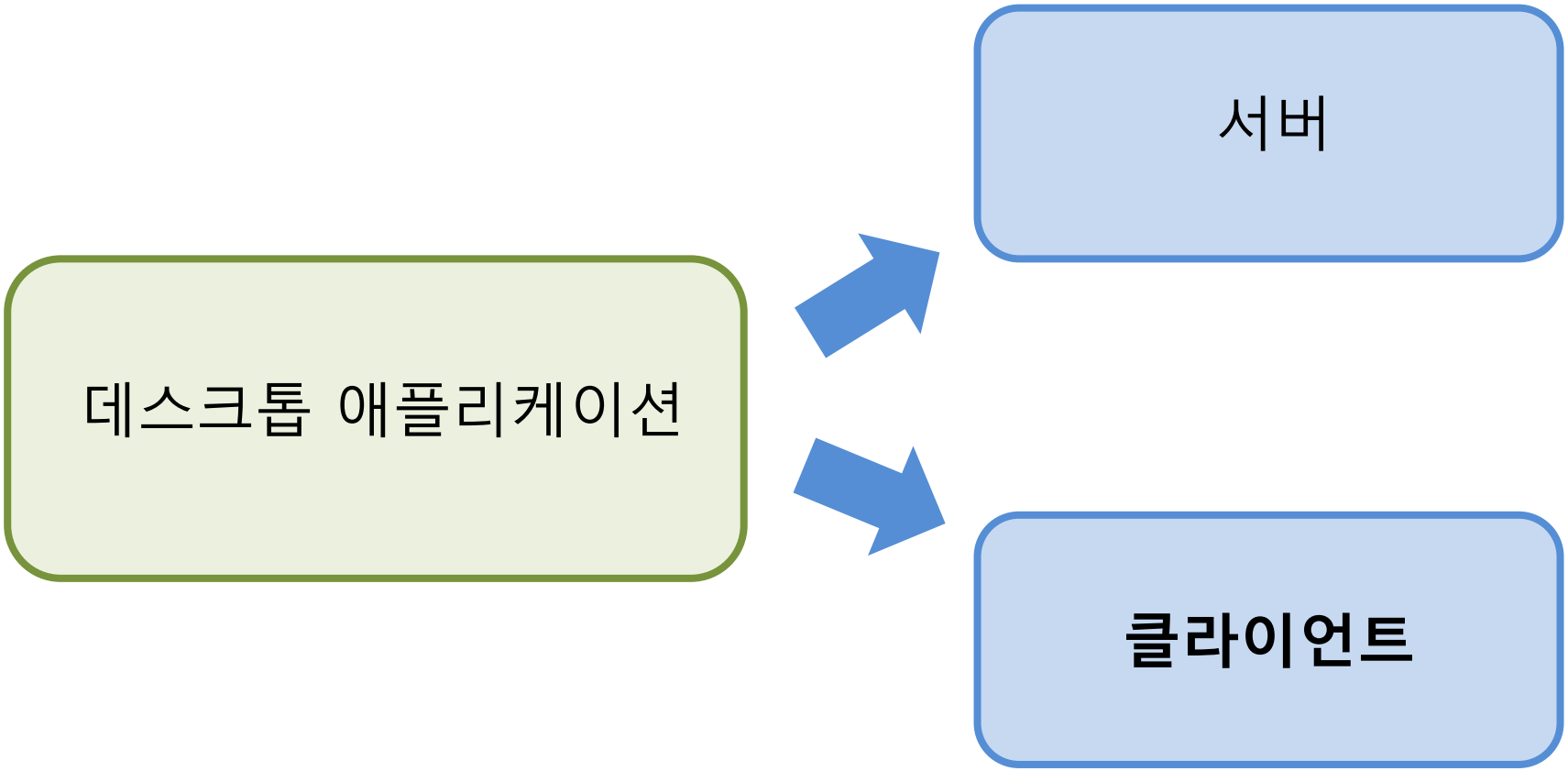
1.2 클라이언트·서버 애플리케이션

데스크톱 애플리케이션

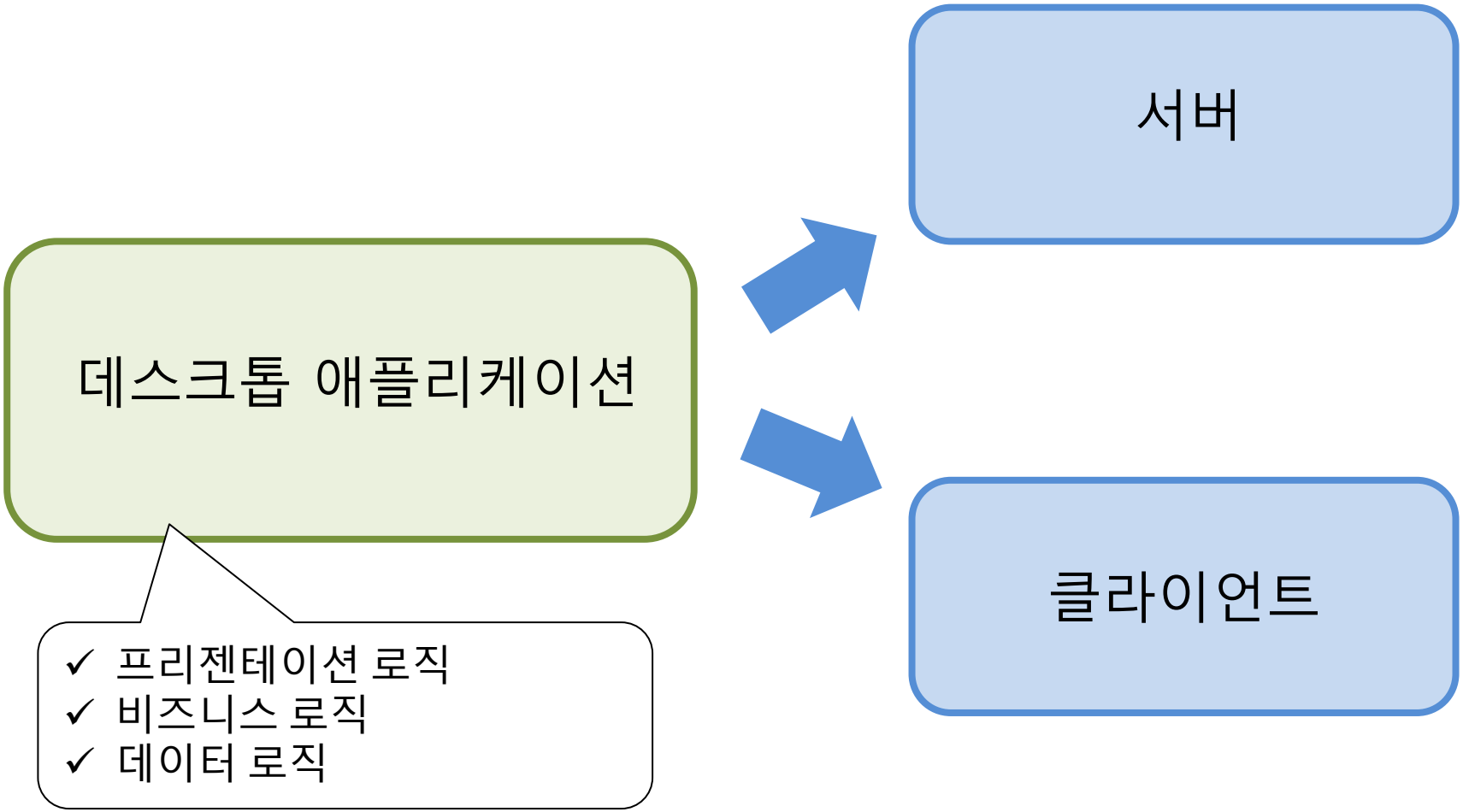
1.2 클라이언트·서버 애플리케이션



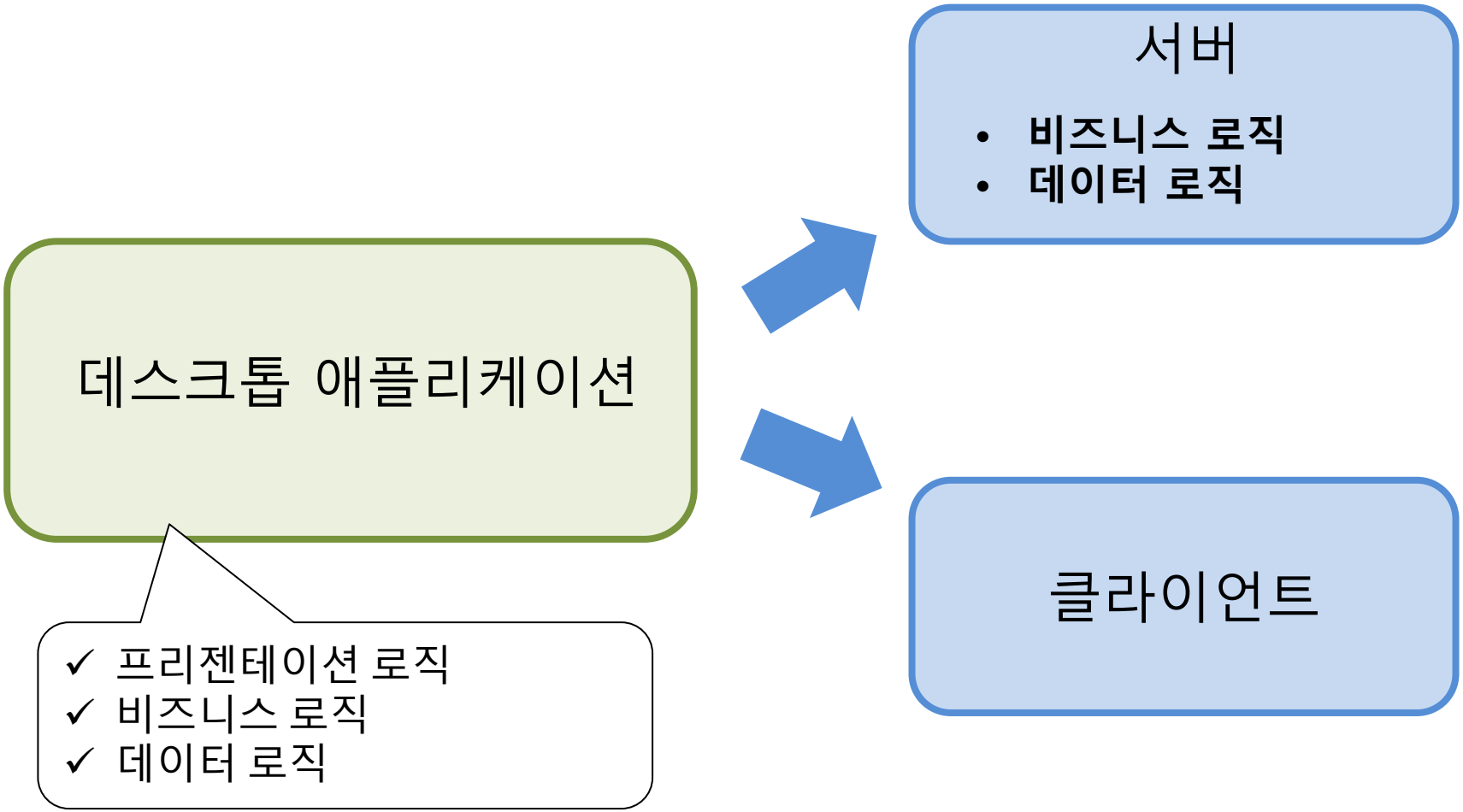
1.2 클라이언트·서버 애플리케이션



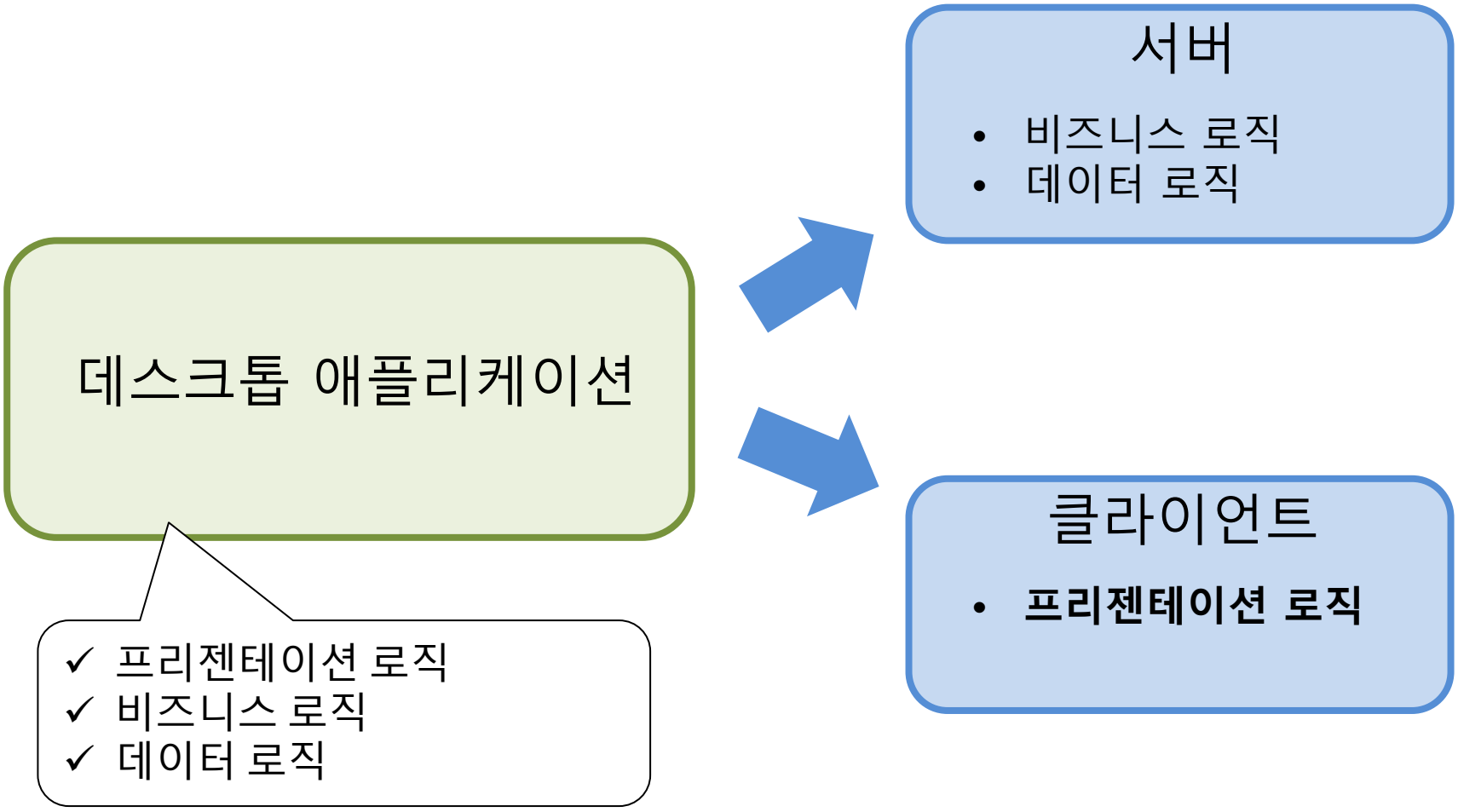
1.2 클라이언트·서버 애플리케이션



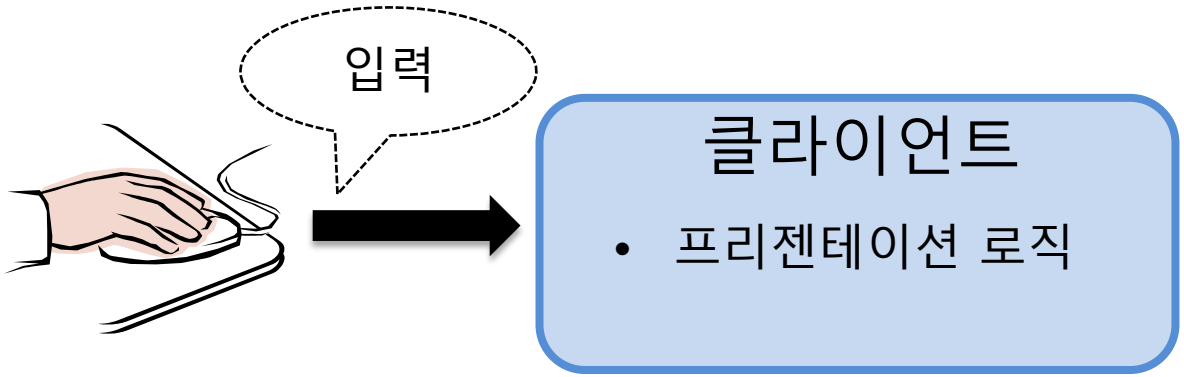
1.2 클라이언트·서버 애플리케이션



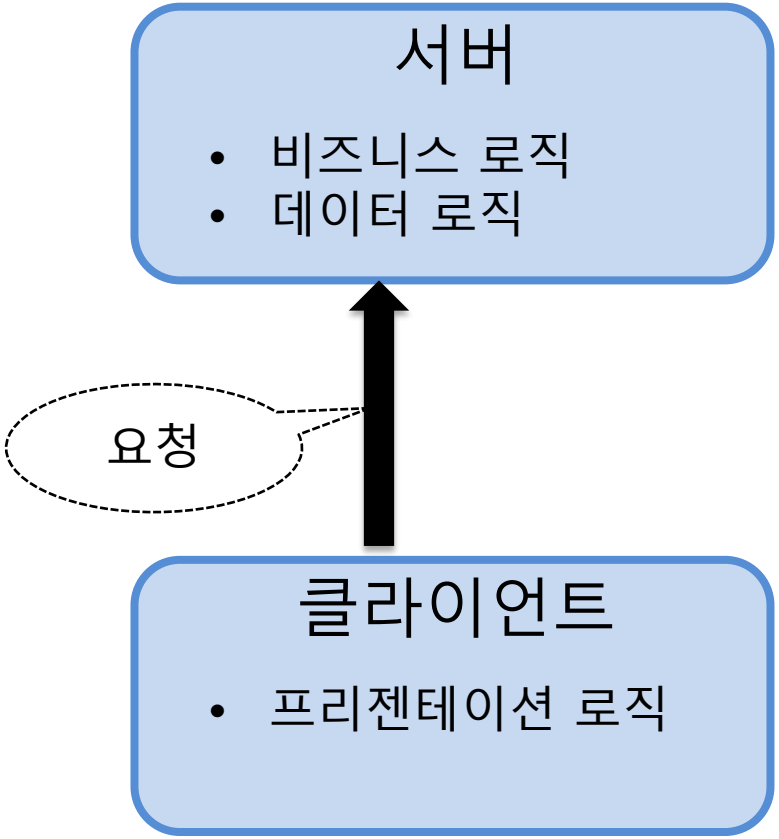
1.2 클라이언트·서버 애플리케이션



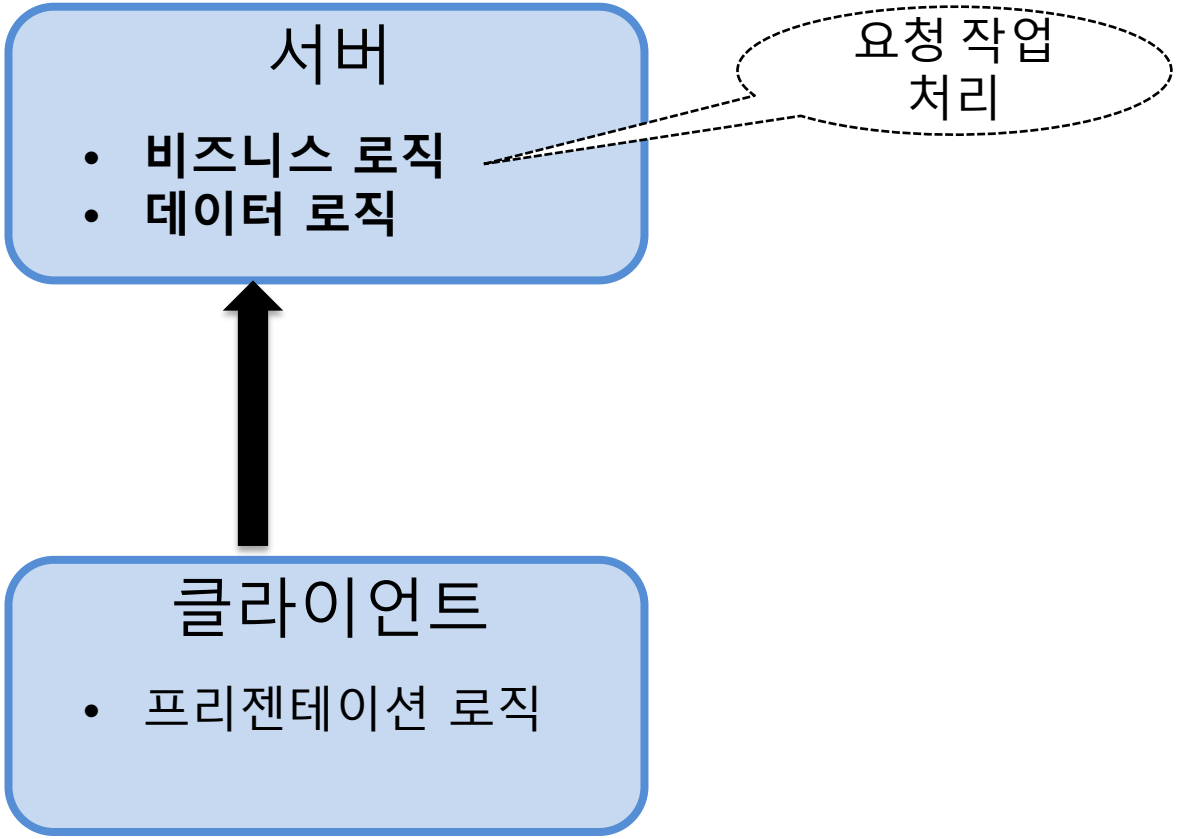
1.2 클라이언트·서버 애플리케이션



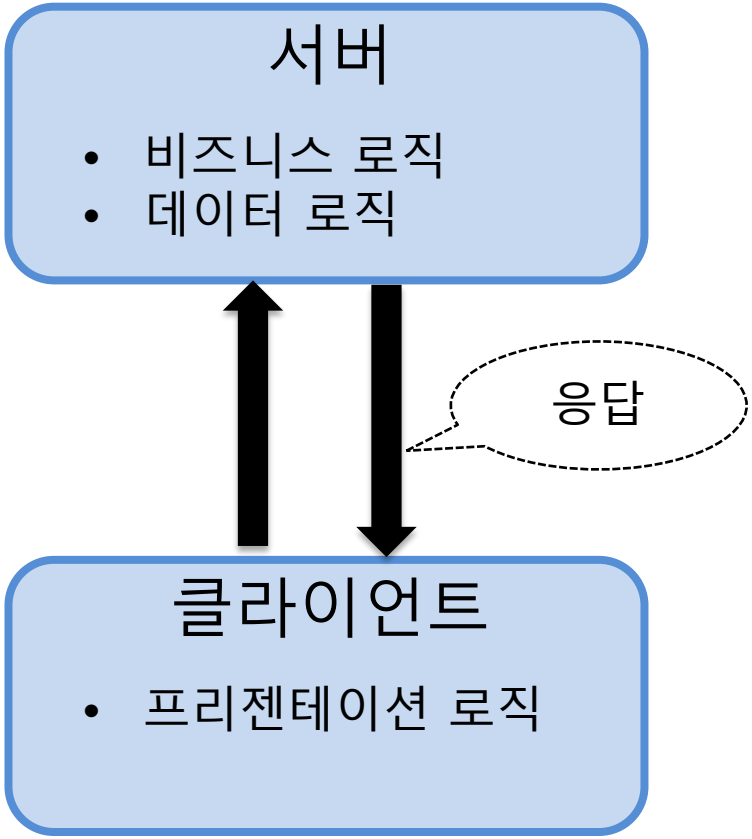
1.2 클라이언트·서버 애플리케이션



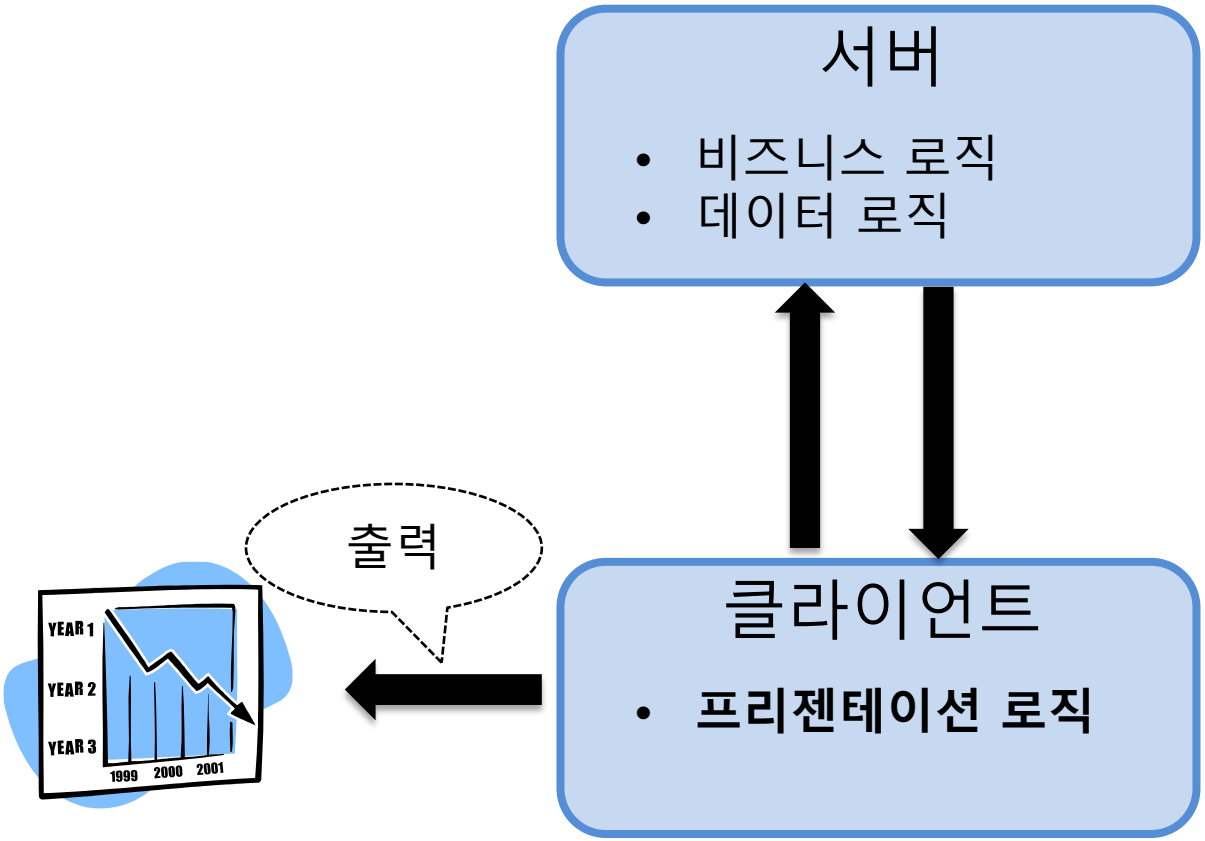
1.2 클라이언트·서버 애플리케이션



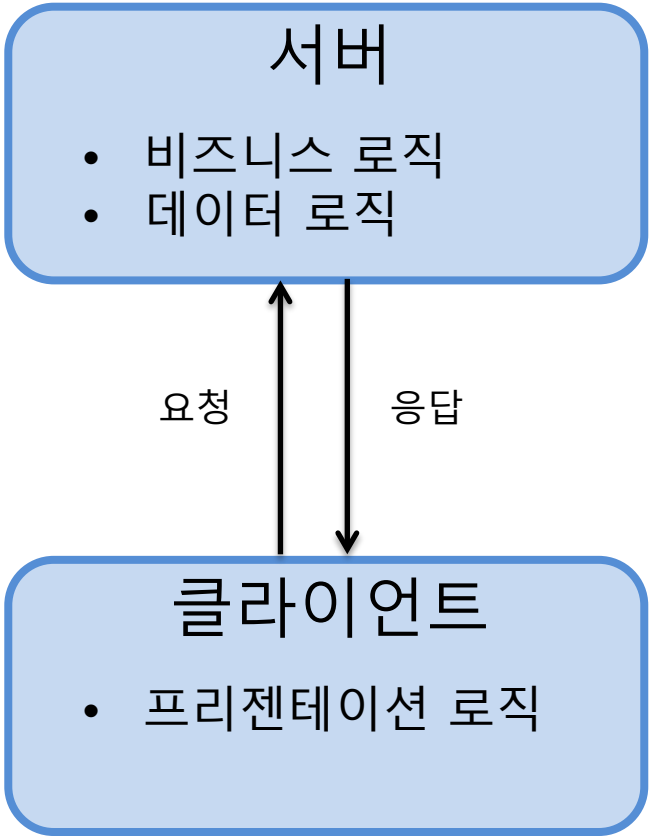
1.2 클라이언트·서버 애플리케이션



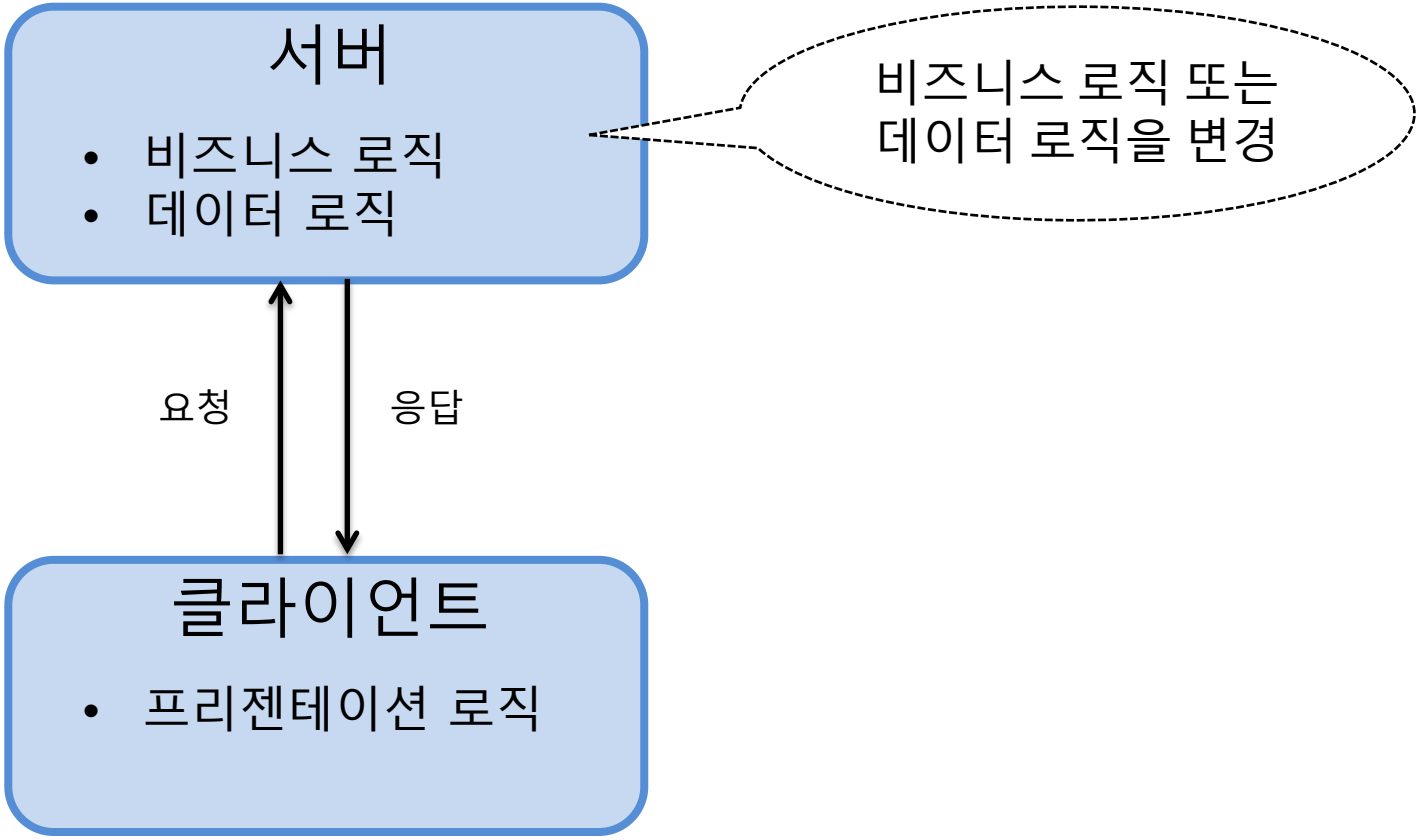
1.2 클라이언트·서버 애플리케이션



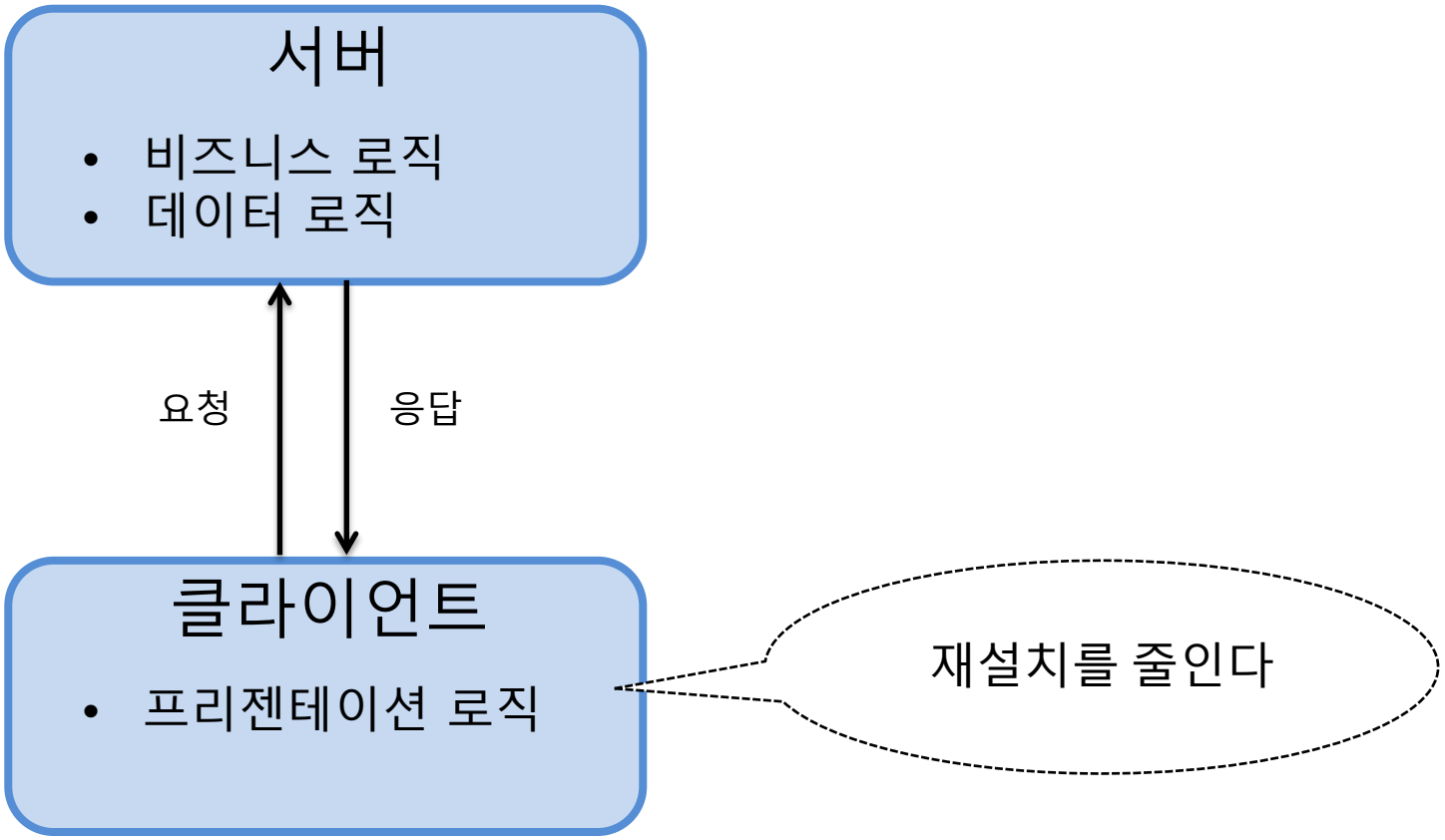
1.2 클라이언트·서버 애플리케이션



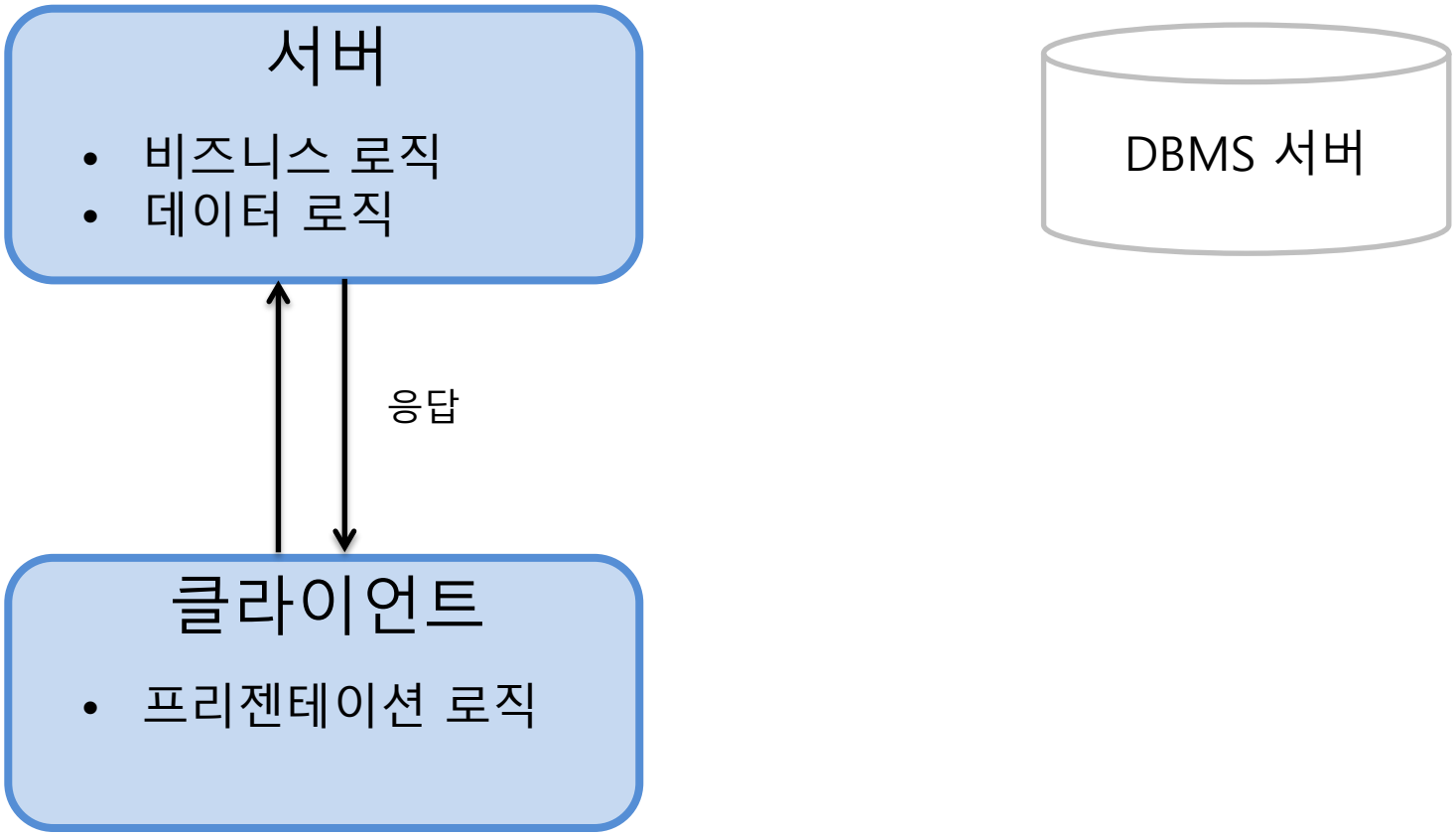
1.2 클라이언트·서버 애플리케이션



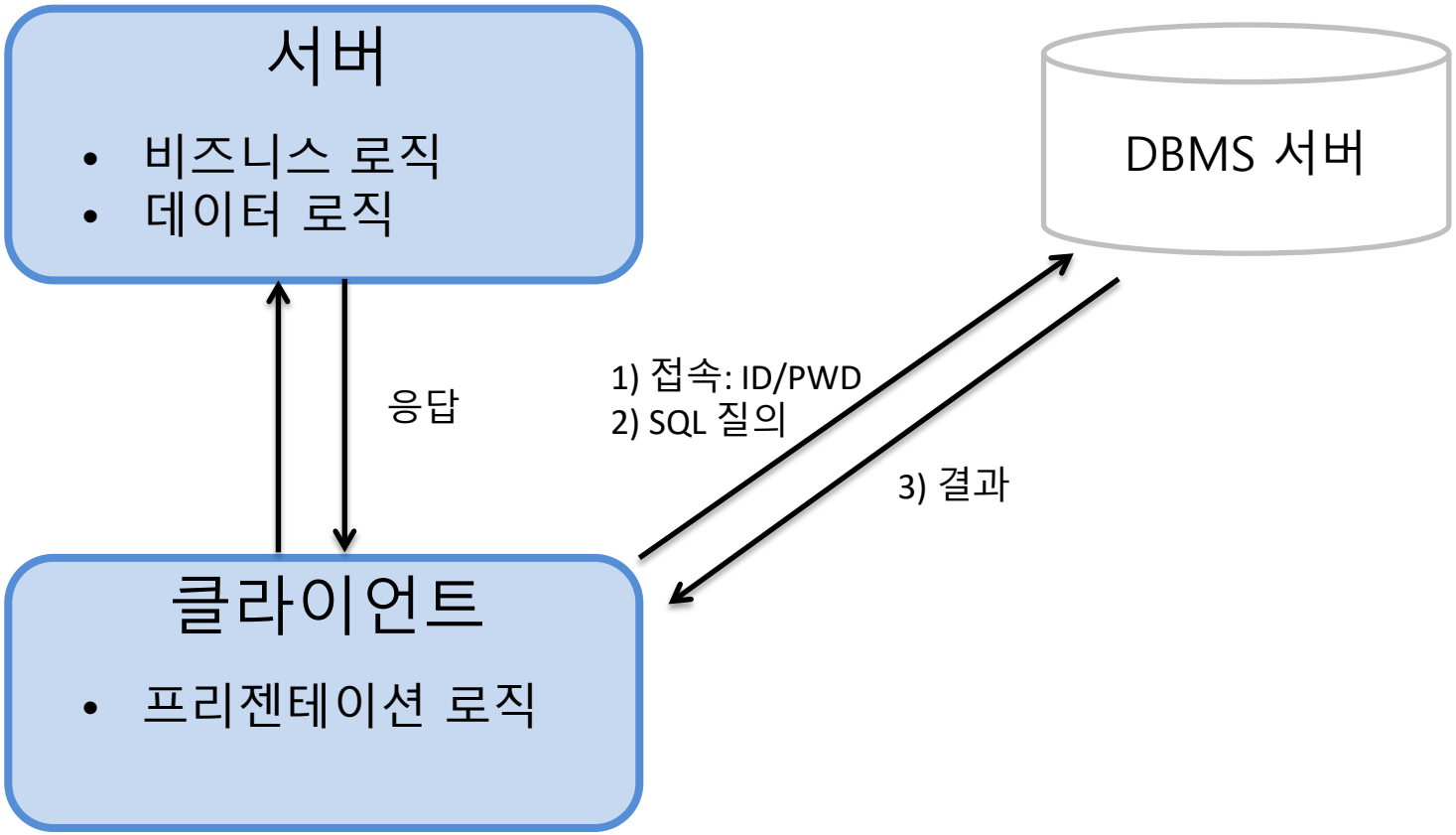
1.2 클라이언트·서버 애플리케이션



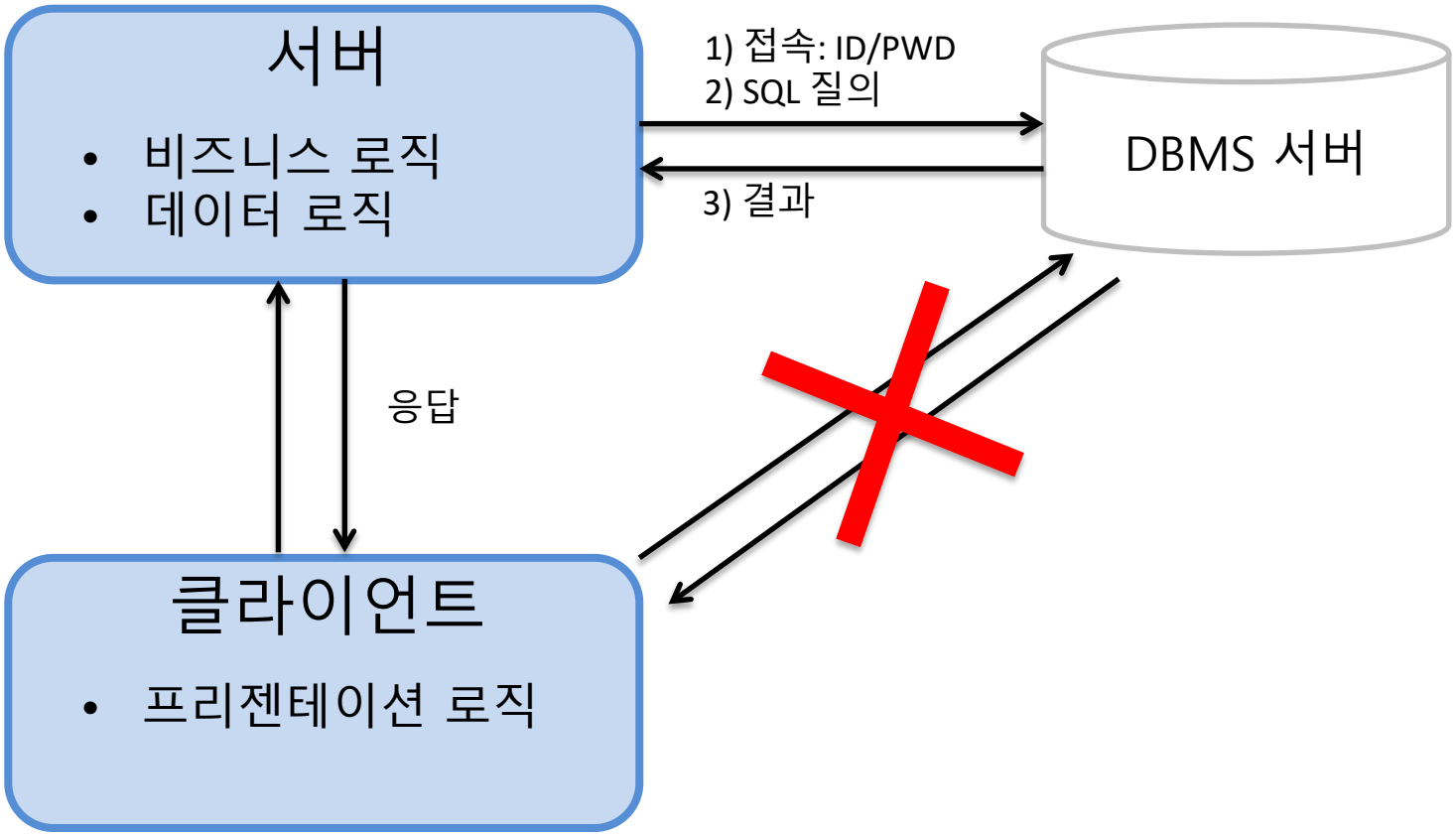
1.2 클라이언트·서버 애플리케이션



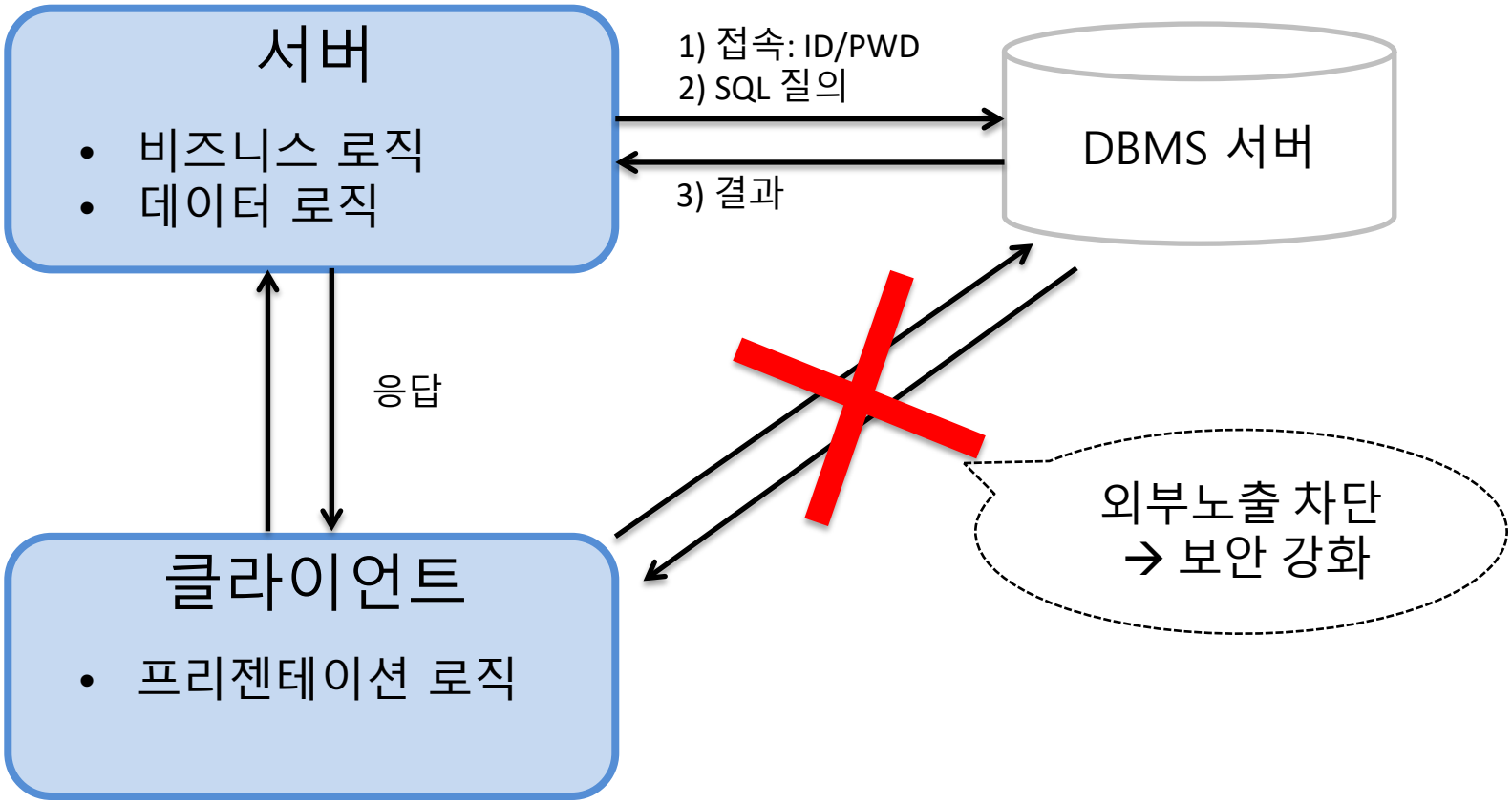
1.2 클라이언트·서버 애플리케이션



1.2 클라이언트·서버 애플리케이션



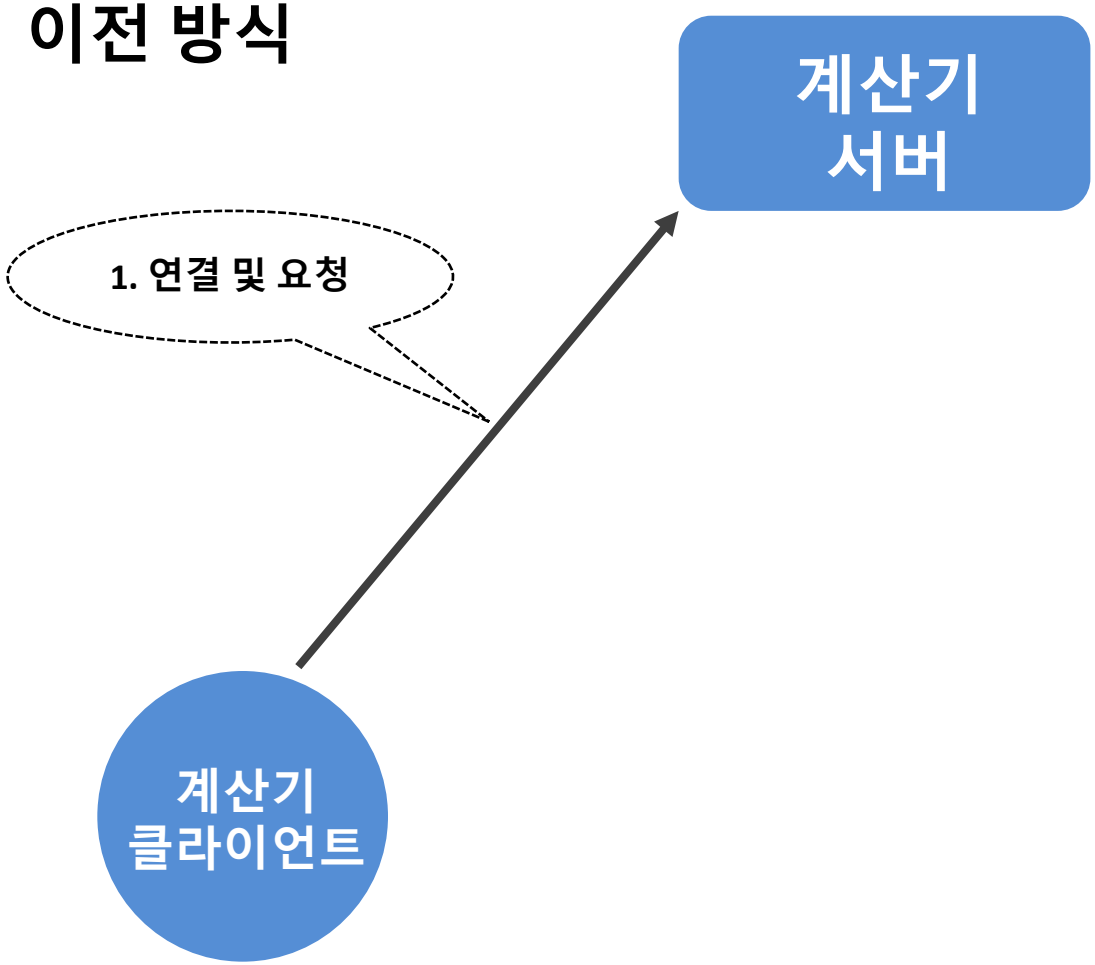
1.2 클라이언트·서버 애플리케이션



1.3 다중 클라이언트 요청처리

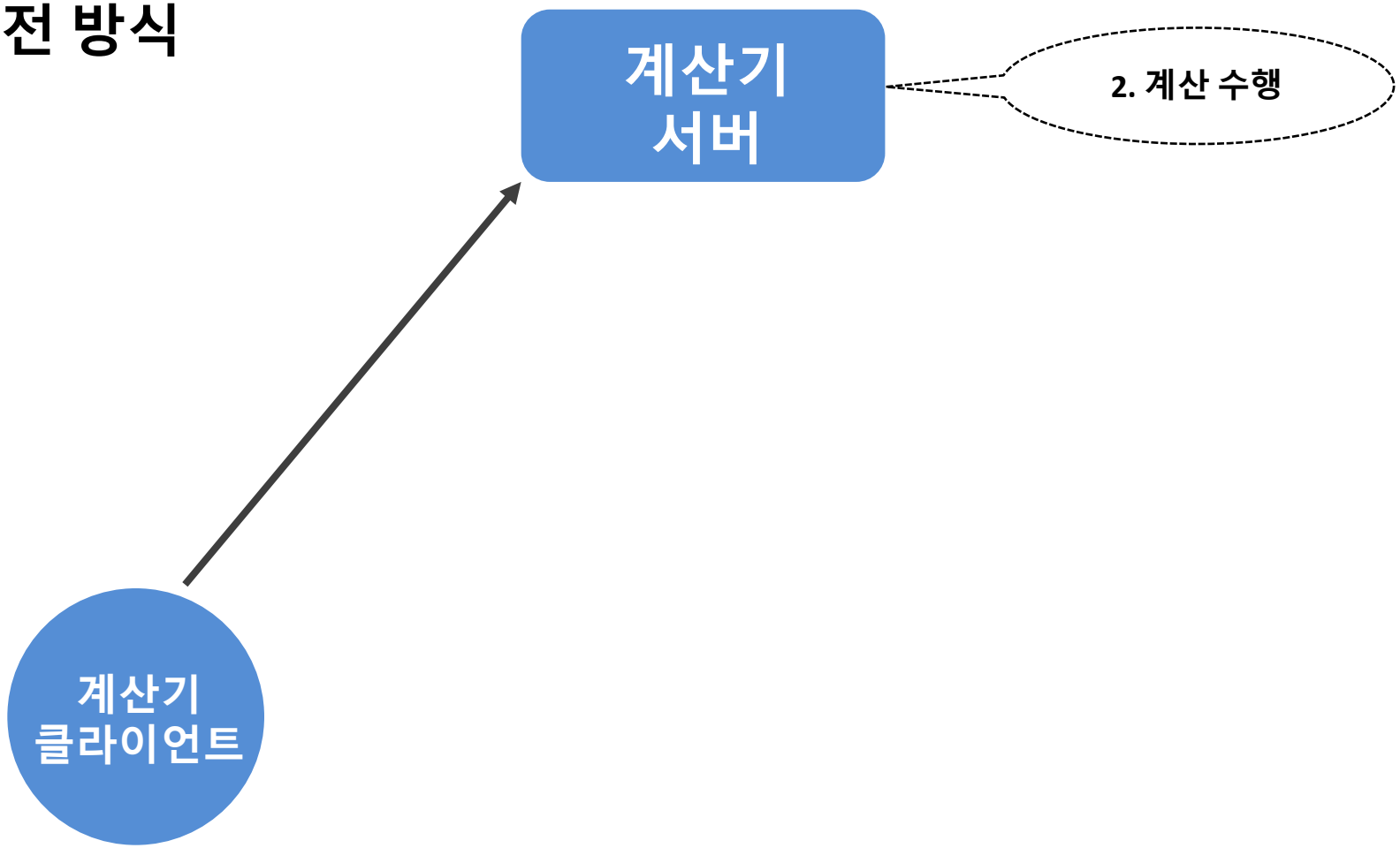
1.3 다중 클라이언트 요청처리

이전 방식



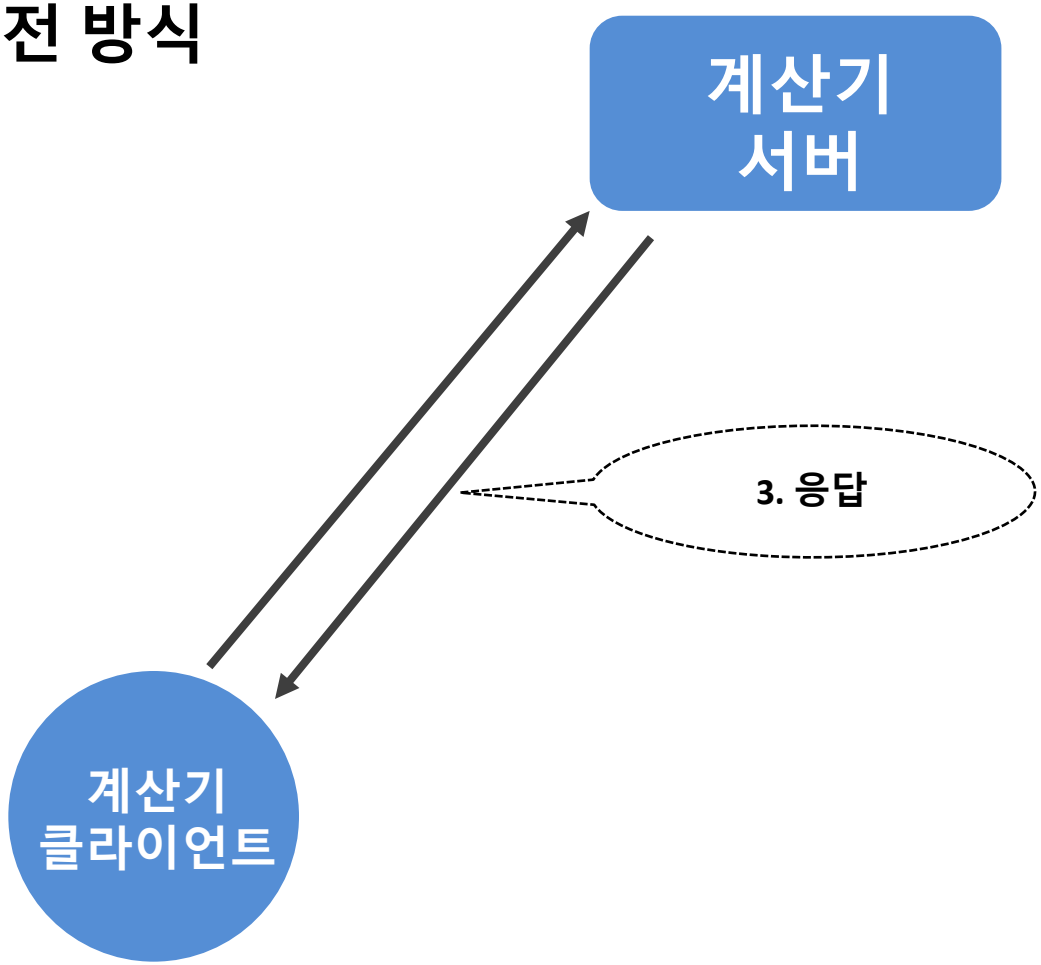
1.3 다중 클라이언트 요청처리

이전 방식



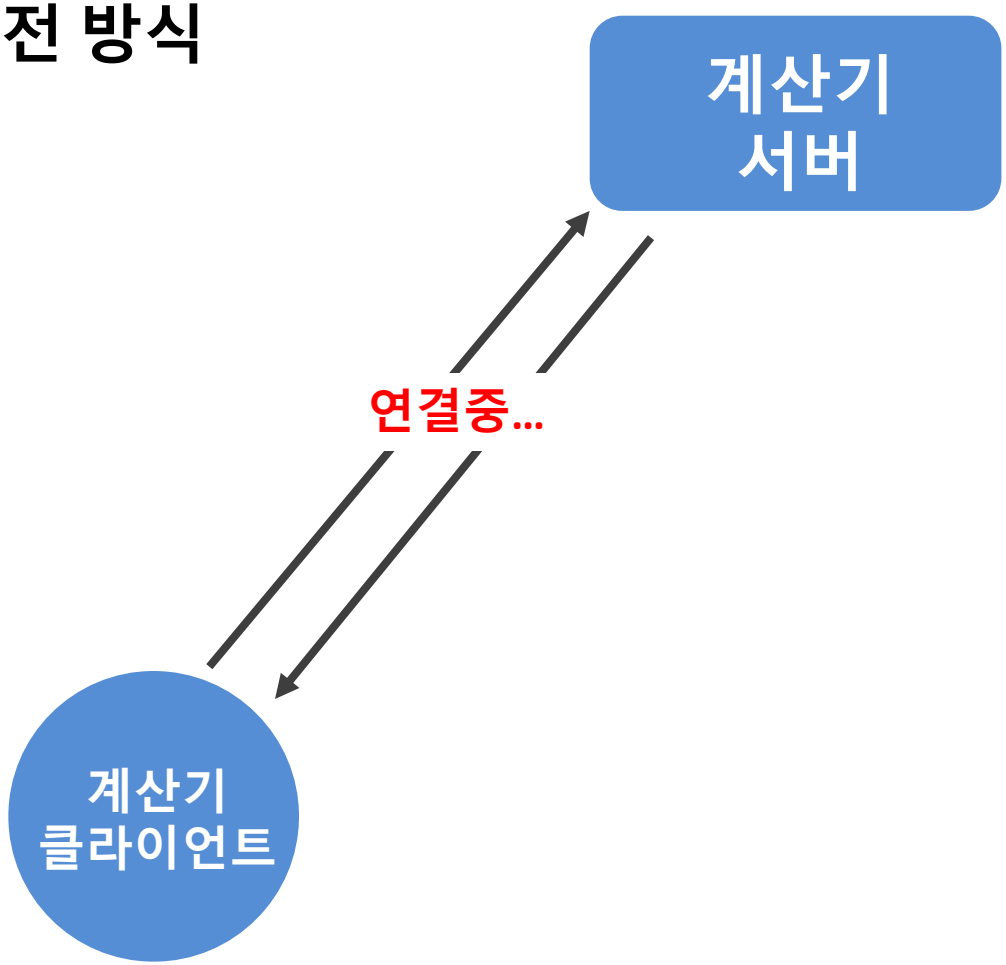
1.3 다중 클라이언트 요청처리

이전 방식



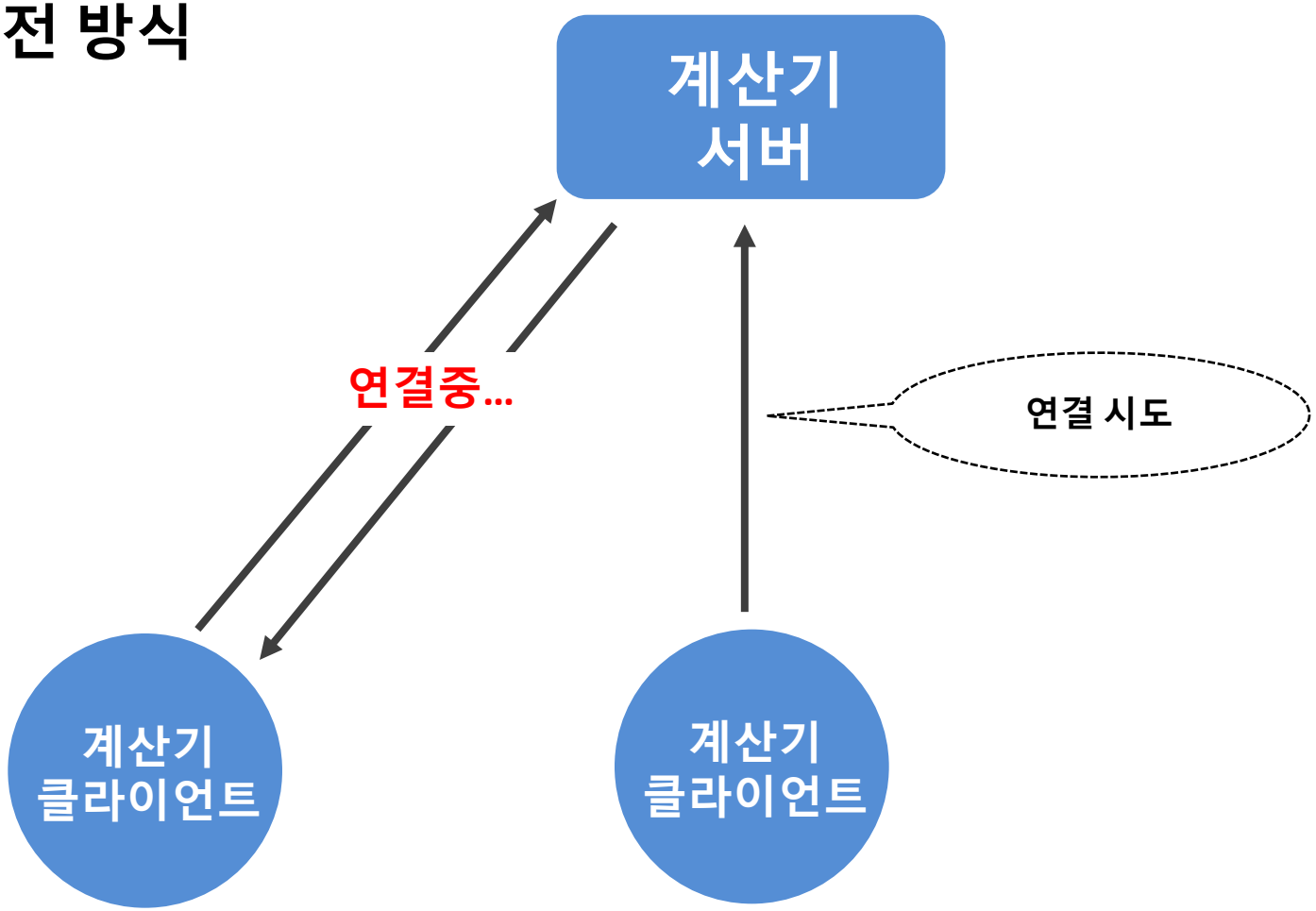
1.3 다중 클라이언트 요청처리

이전 방식



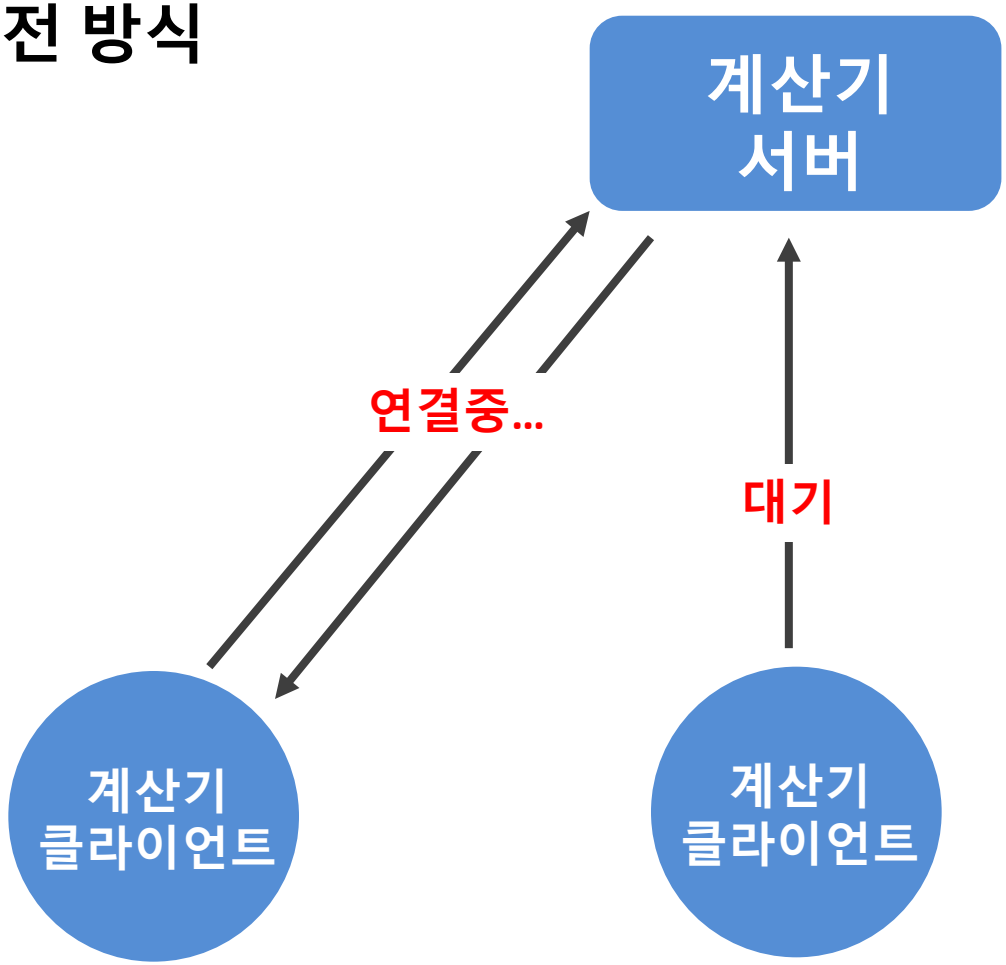
1.3 다중 클라이언트 요청처리

이전 방식



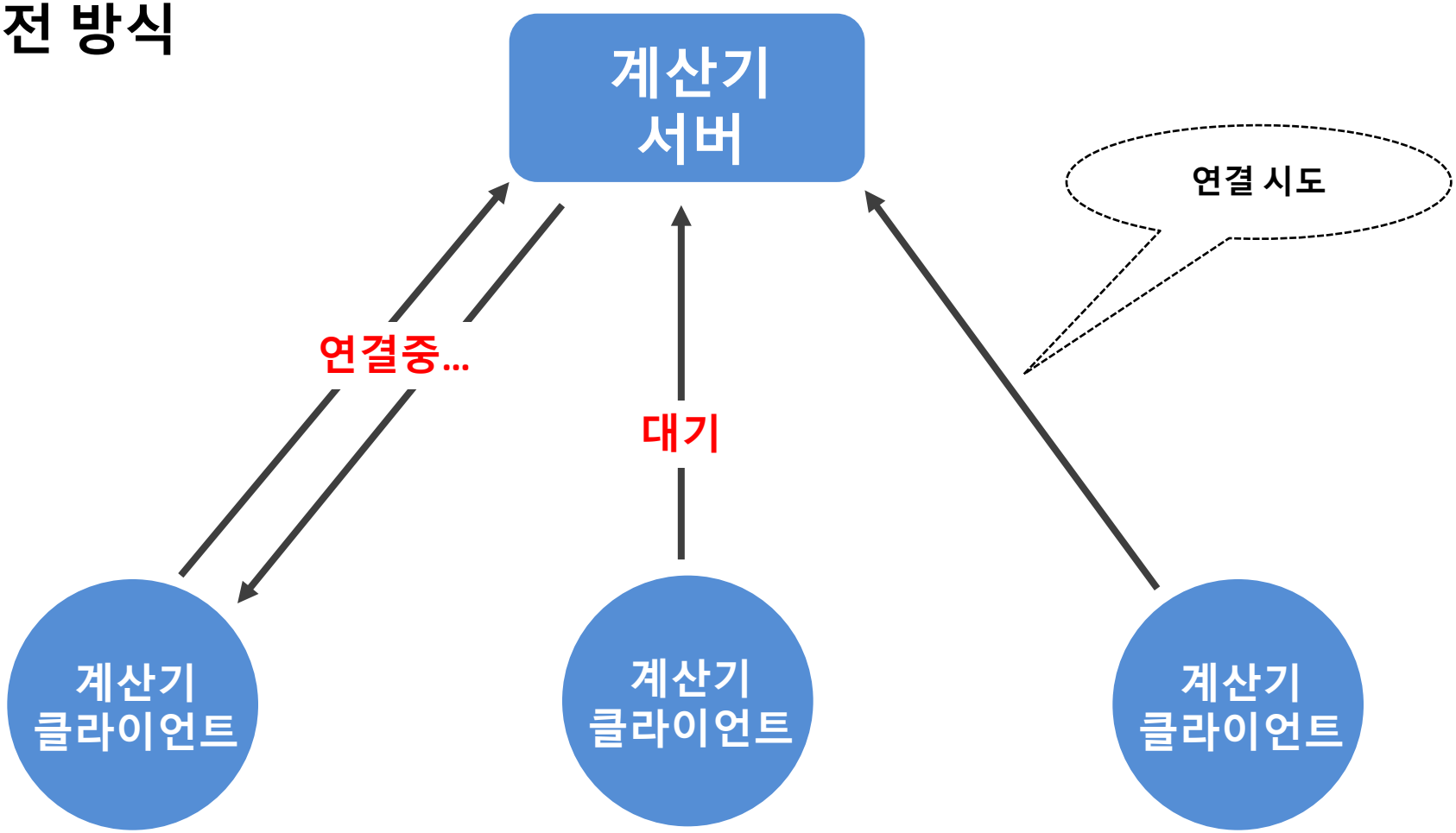
1.3 다중 클라이언트 요청처리

이전 방식



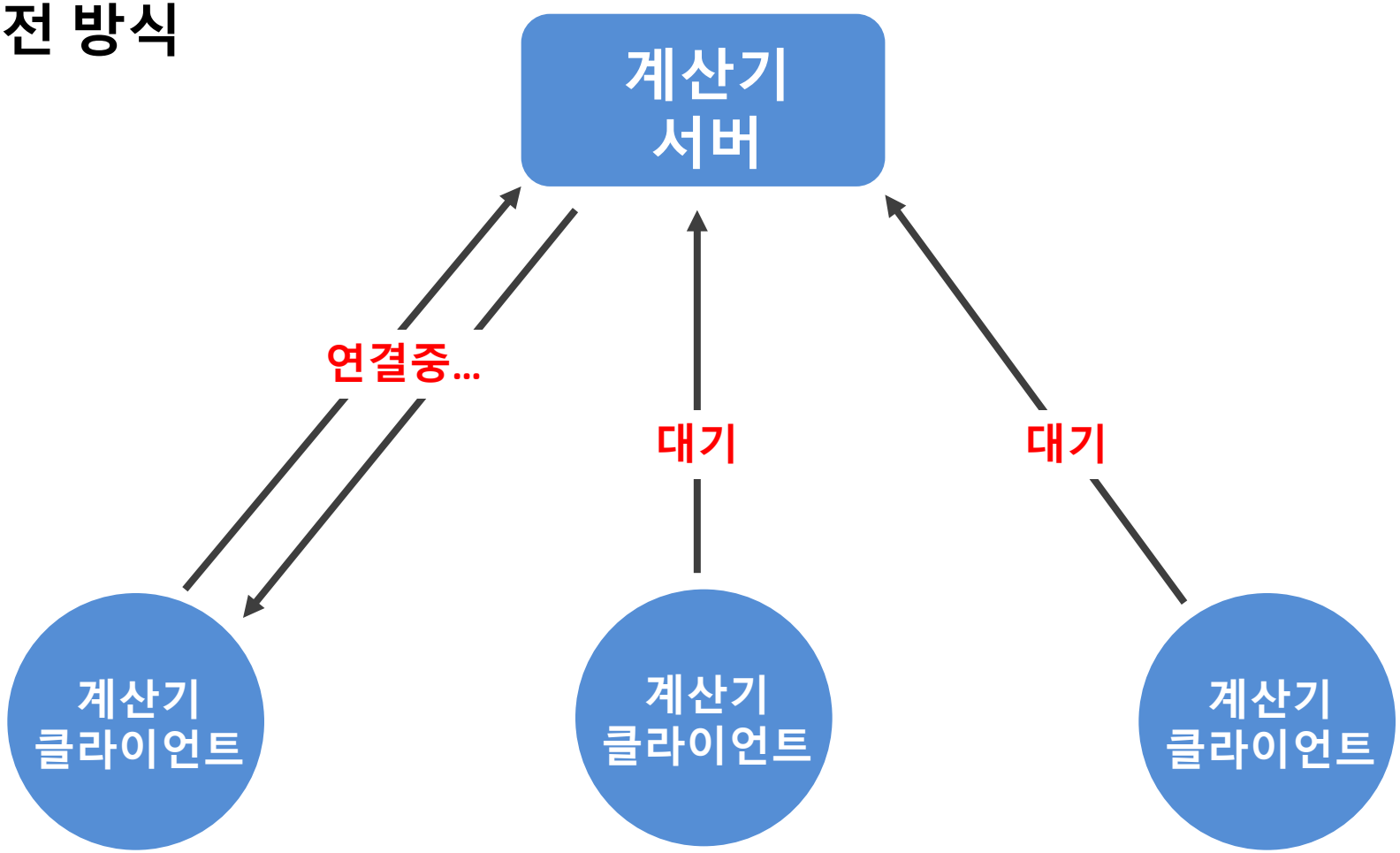
1.3 다중 클라이언트 요청처리

이전 방식



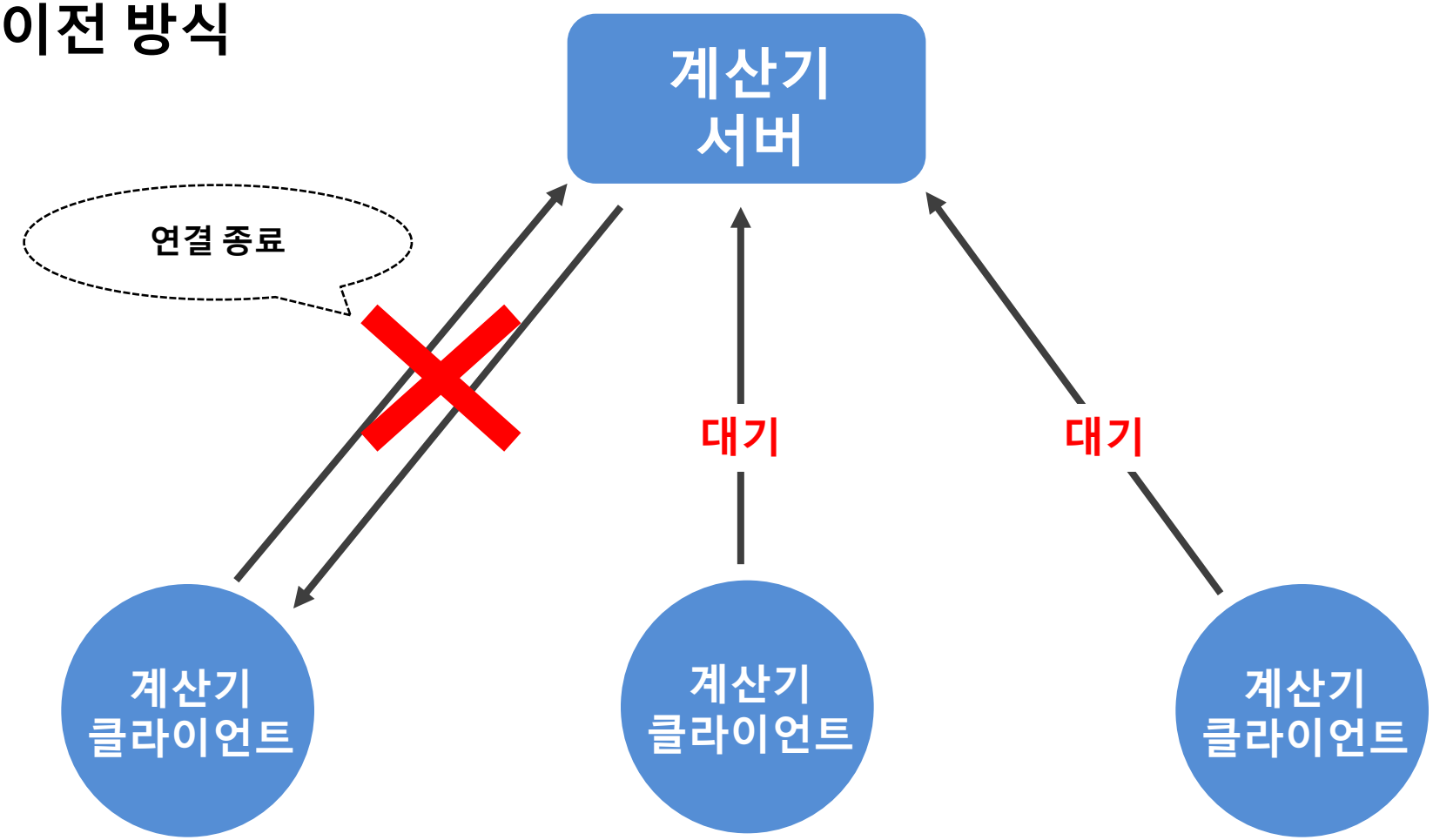
1.3 다중 클라이언트 요청처리

이전 방식



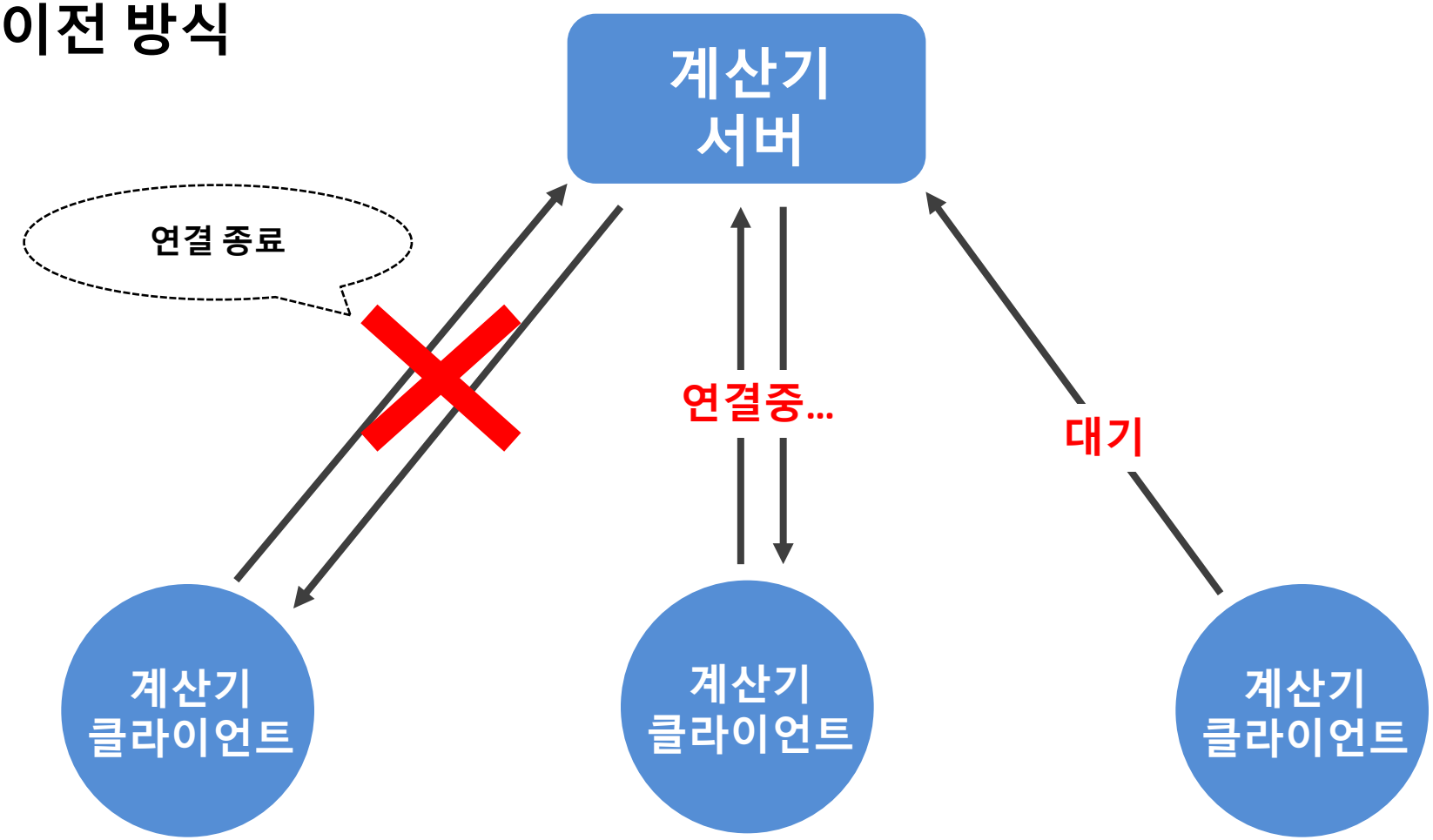
1.3 다중 클라이언트 요청처리

이전 방식



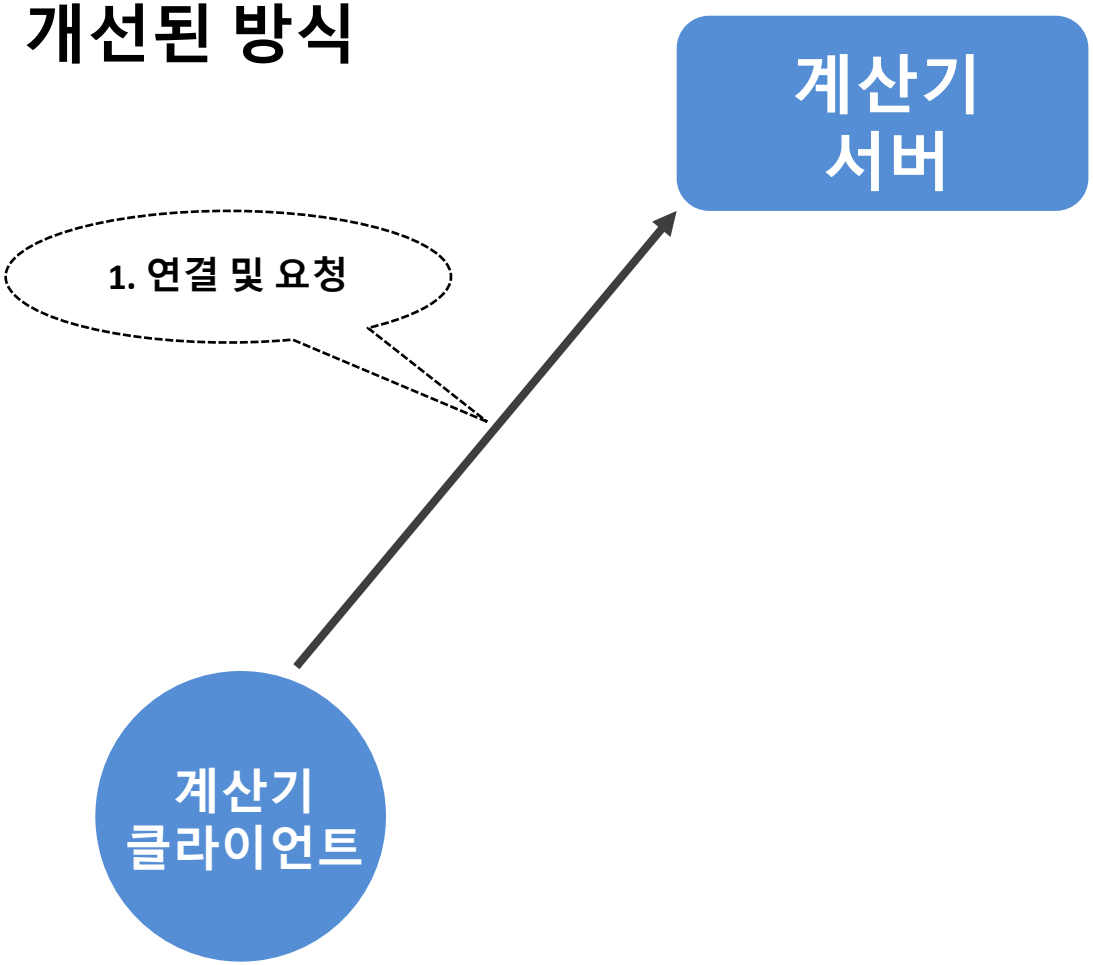
1.3 다중 클라이언트 요청처리

이전 방식



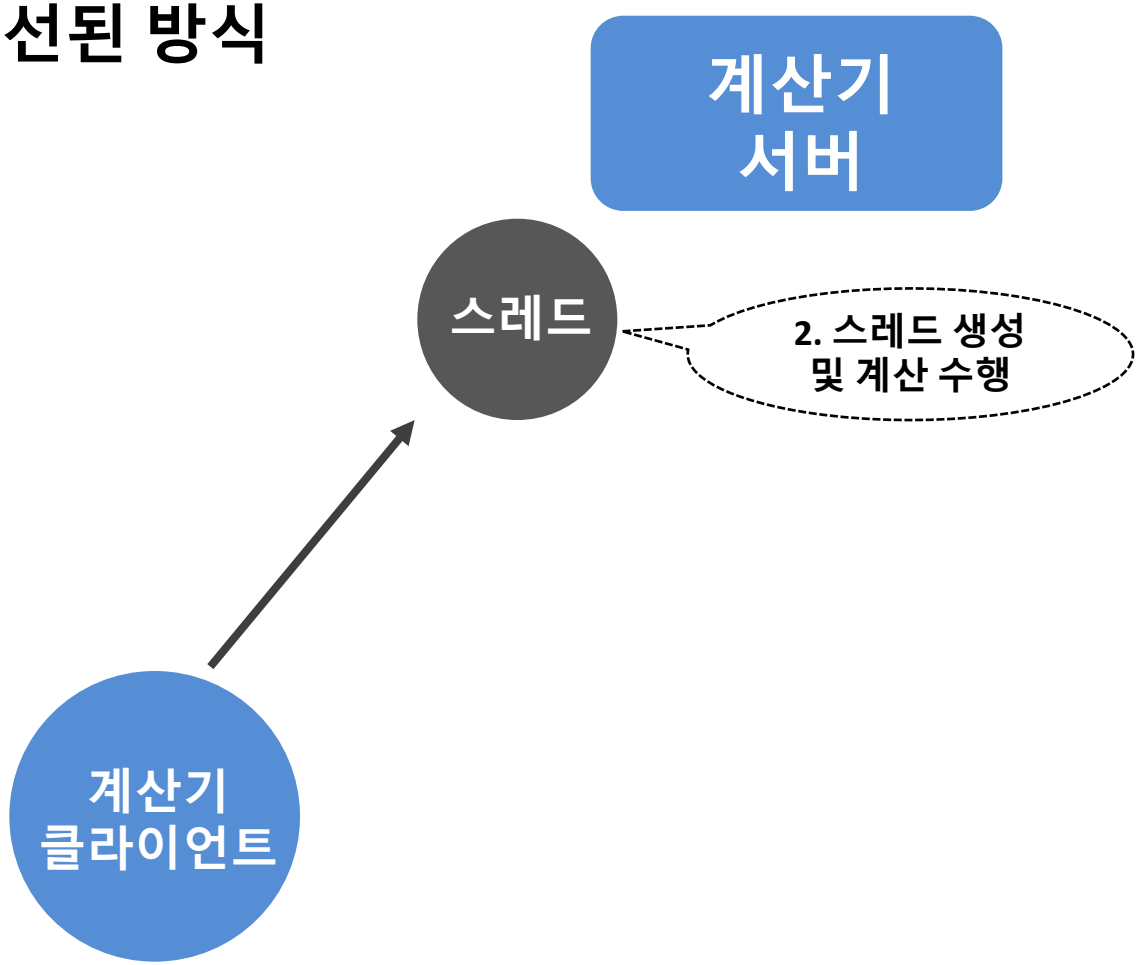
1.3 다중 클라이언트 요청처리

개선된 방식



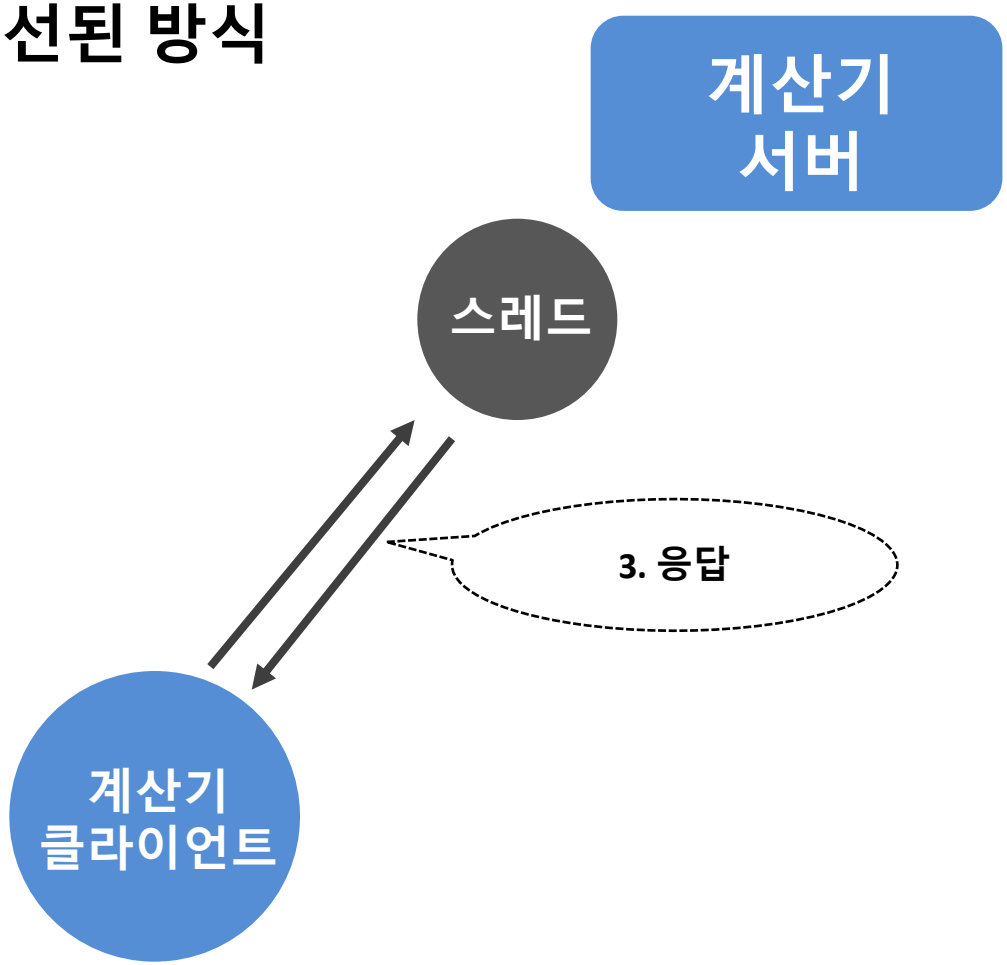
1.3 다중 클라이언트 요청처리

개선된 방식



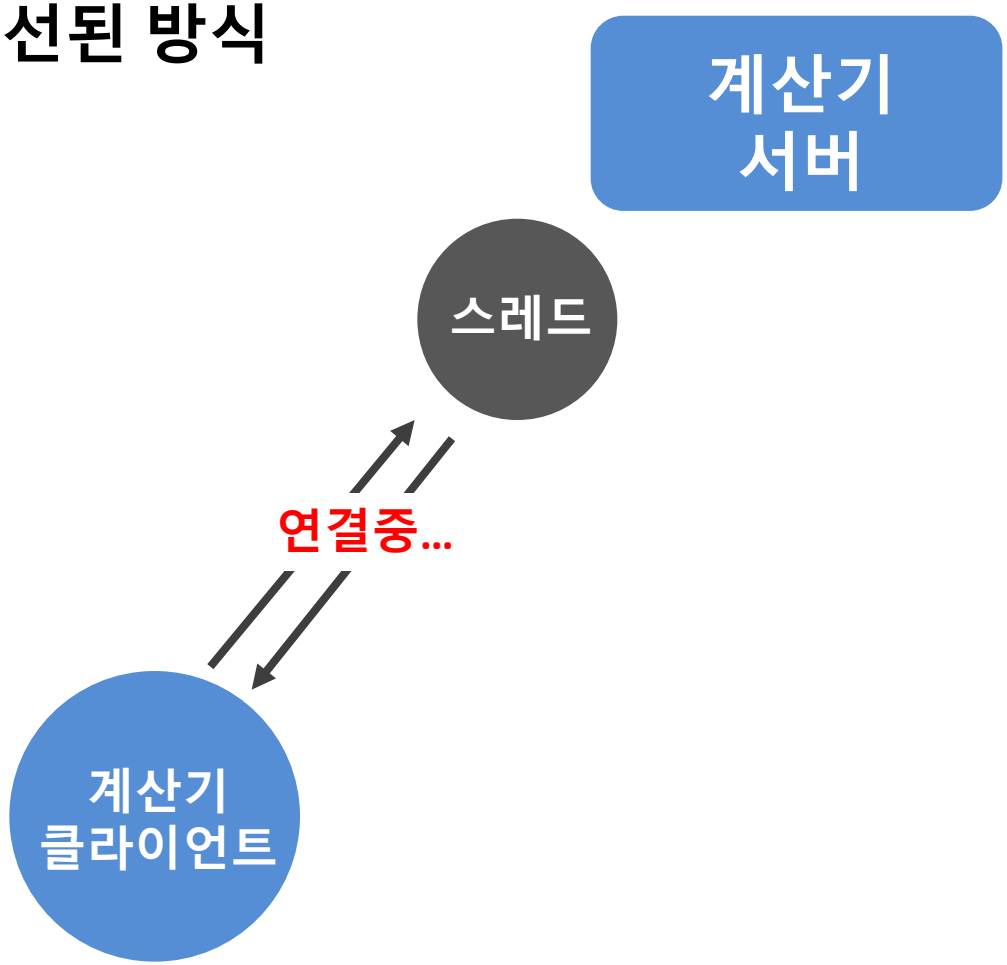
1.3 다중 클라이언트 요청처리

개선된 방식



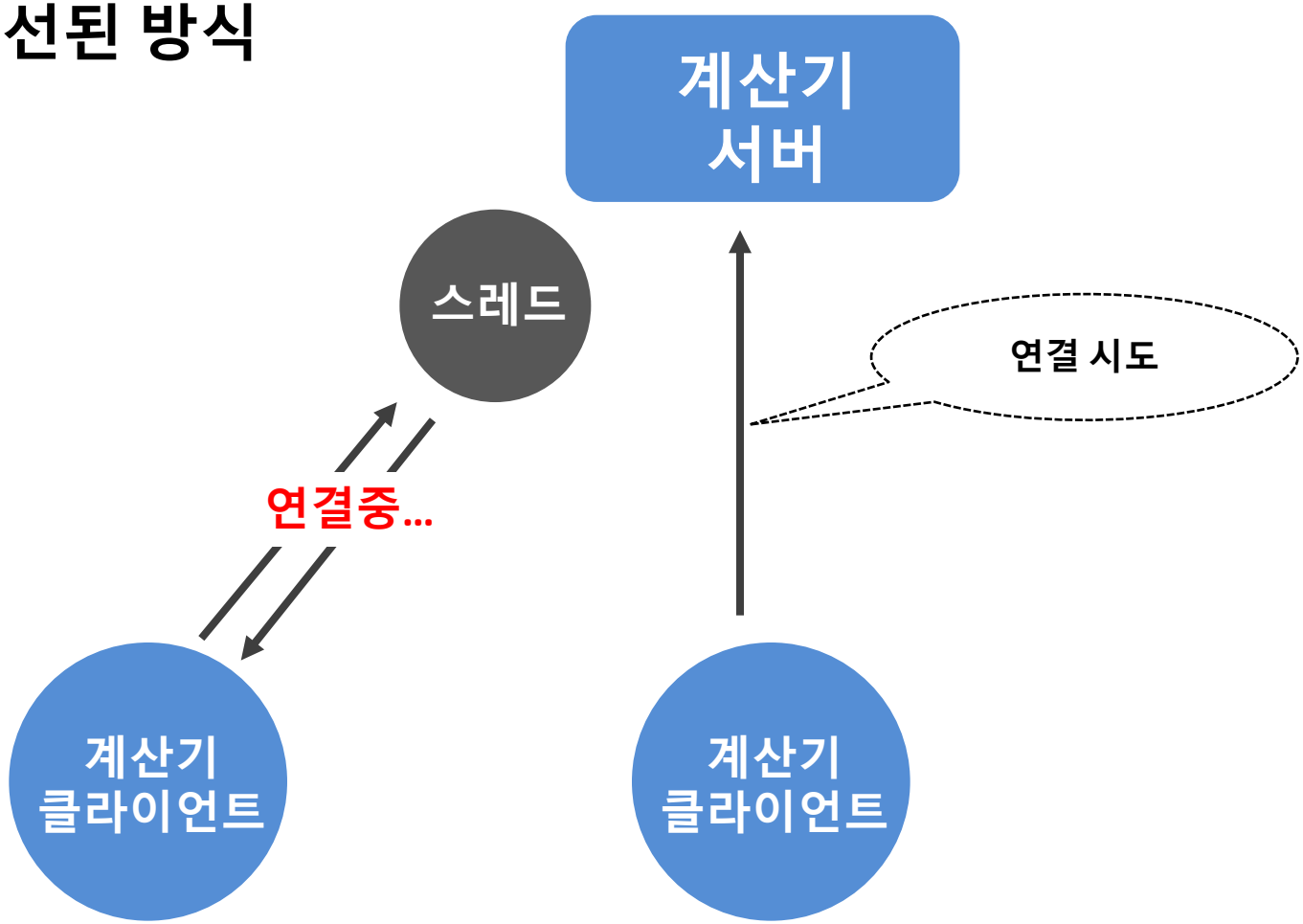
1.3 다중 클라이언트 요청처리

개선된 방식



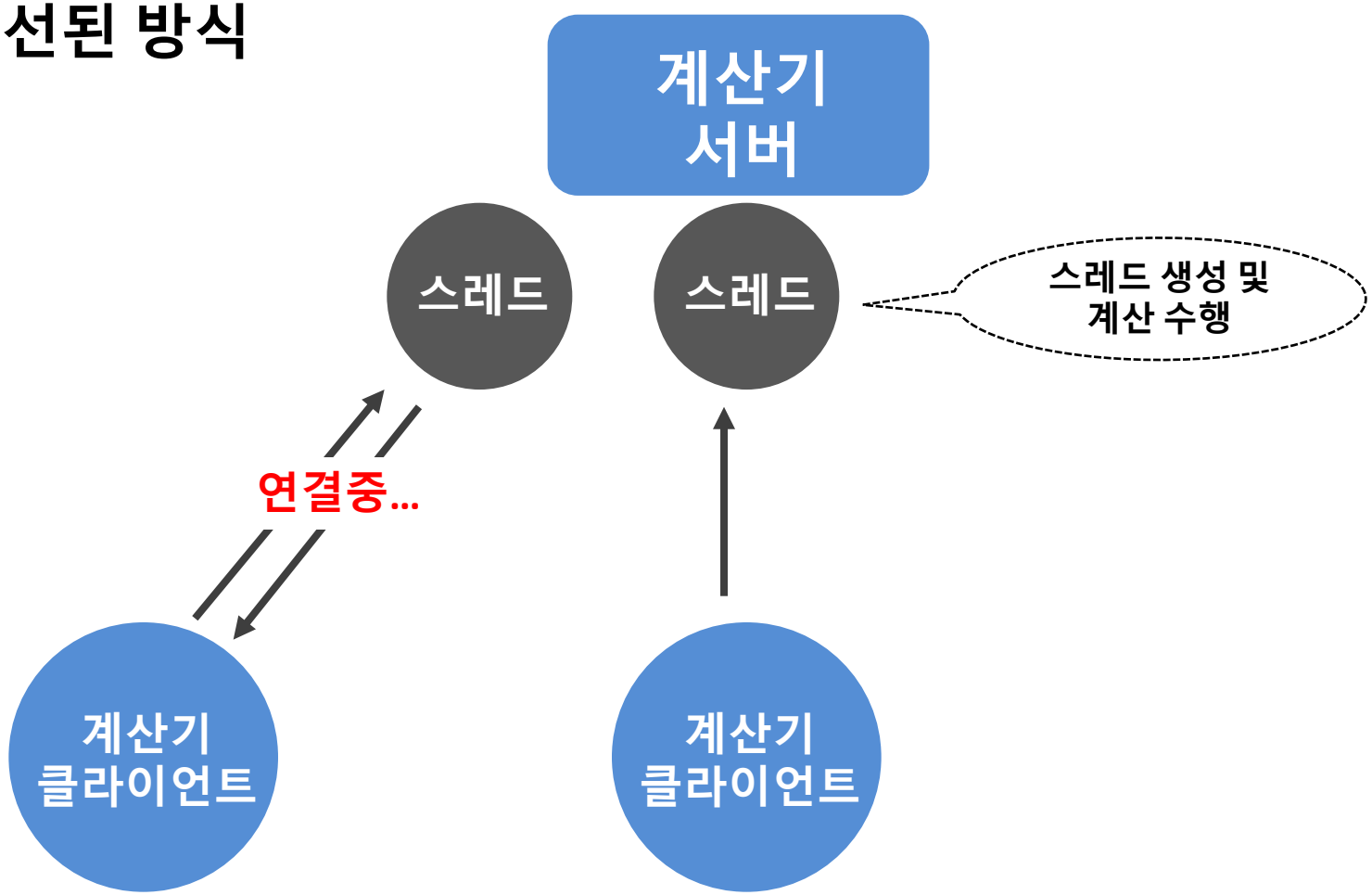
1.3 다중 클라이언트 요청처리

개선된 방식



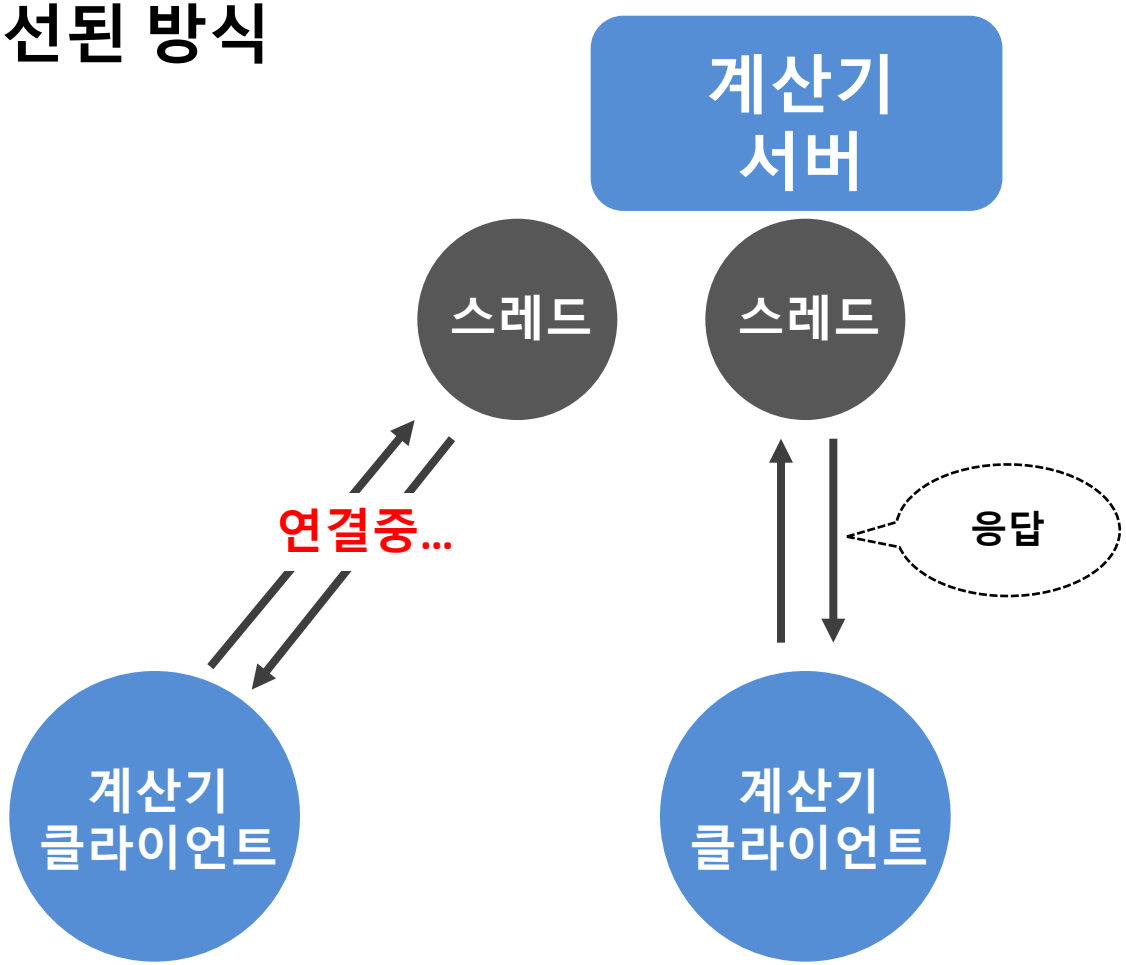
1.3 다중 클라이언트 요청처리

개선된 방식



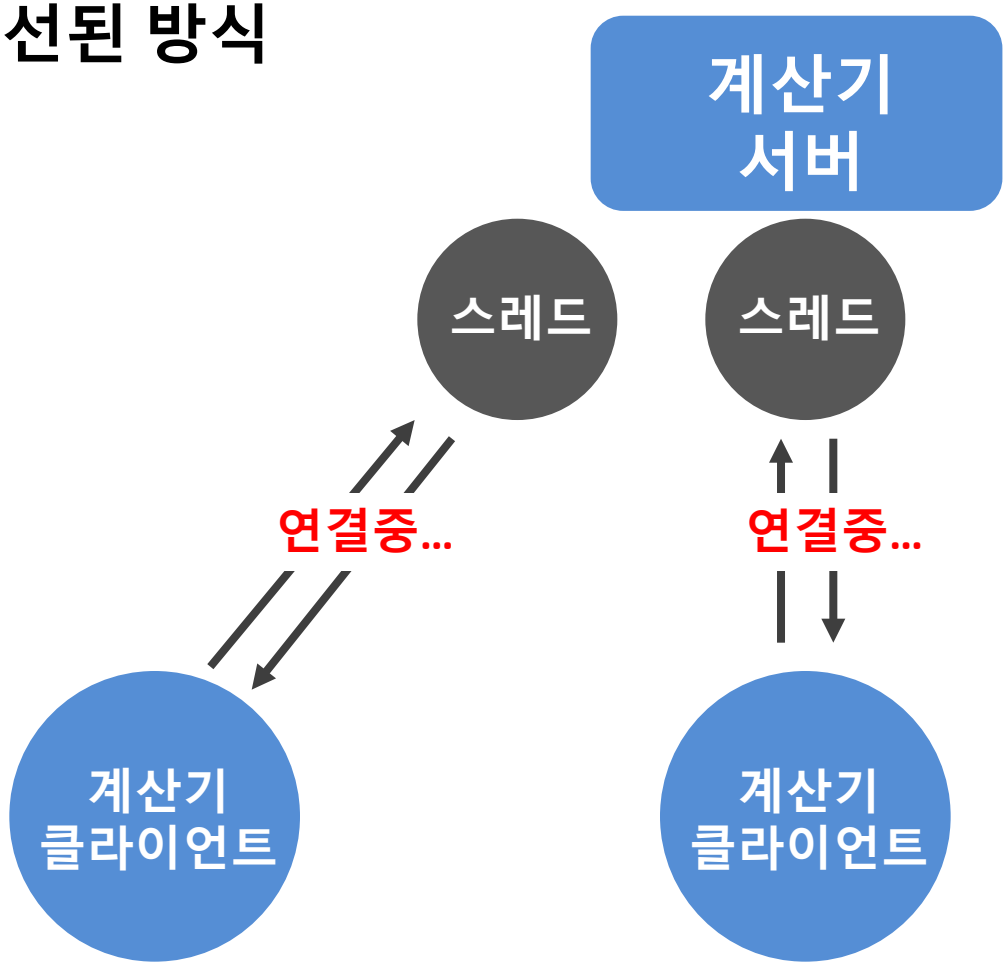
1.3 다중 클라이언트 요청처리

개선된 방식



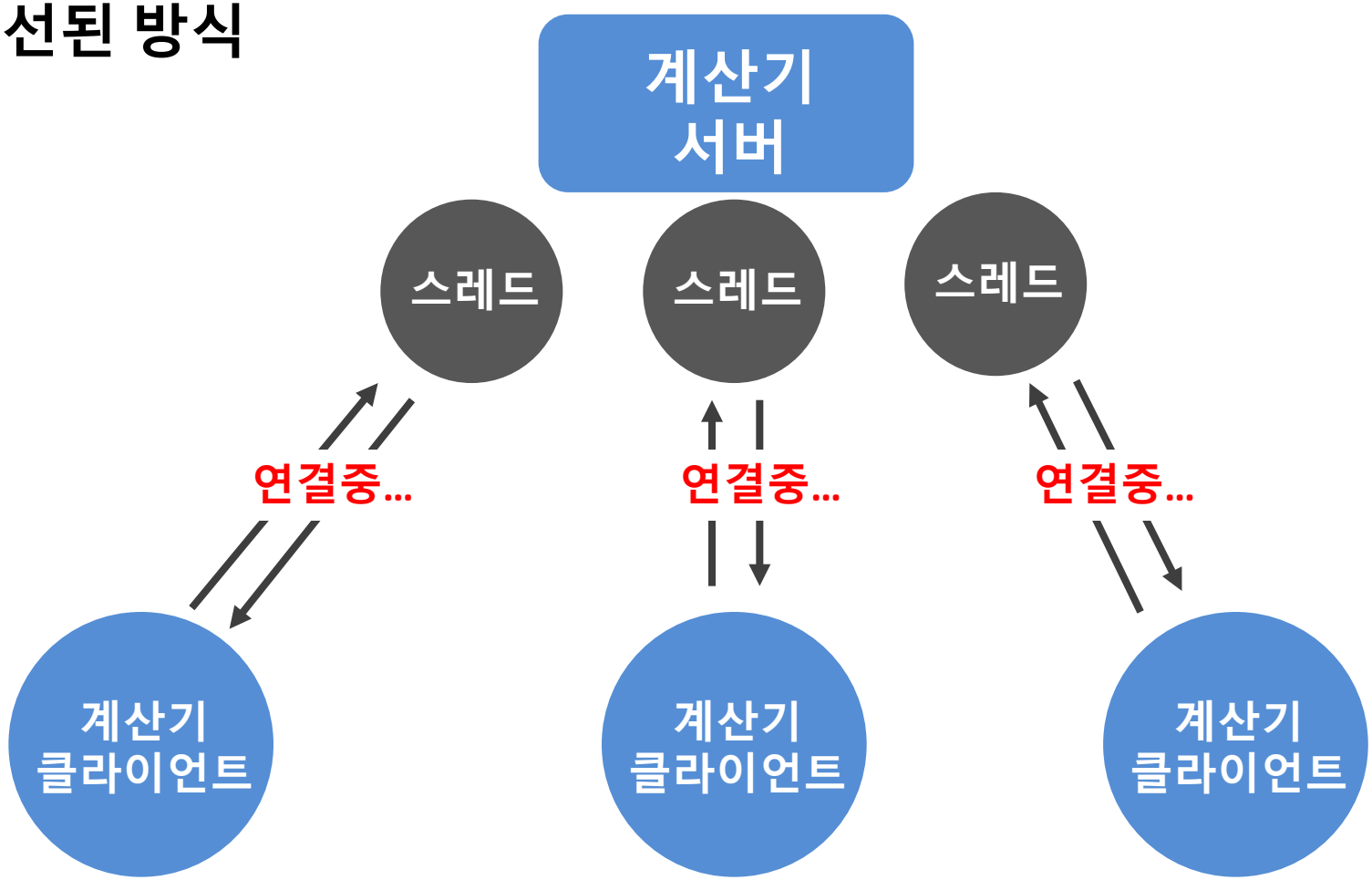
1.3 다중 클라이언트 요청처리

개선된 방식



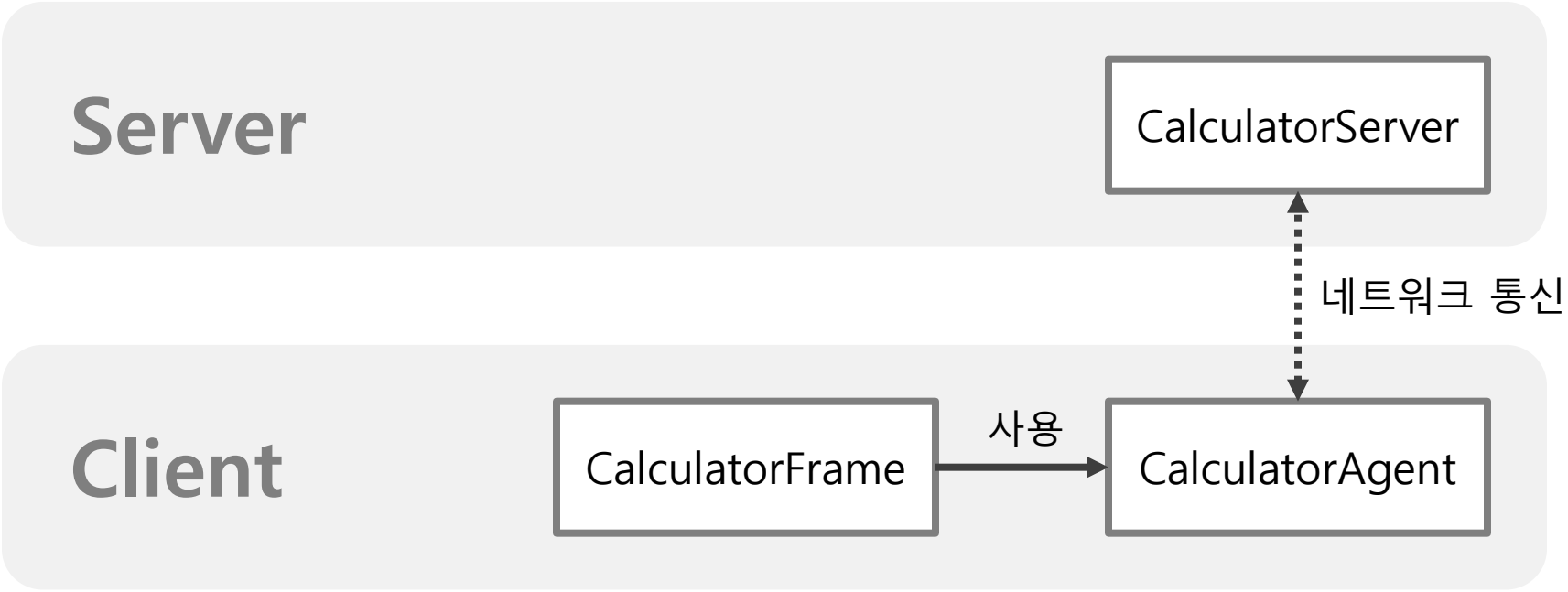
1.3 다중 클라이언트 요청처리

개선된 방식



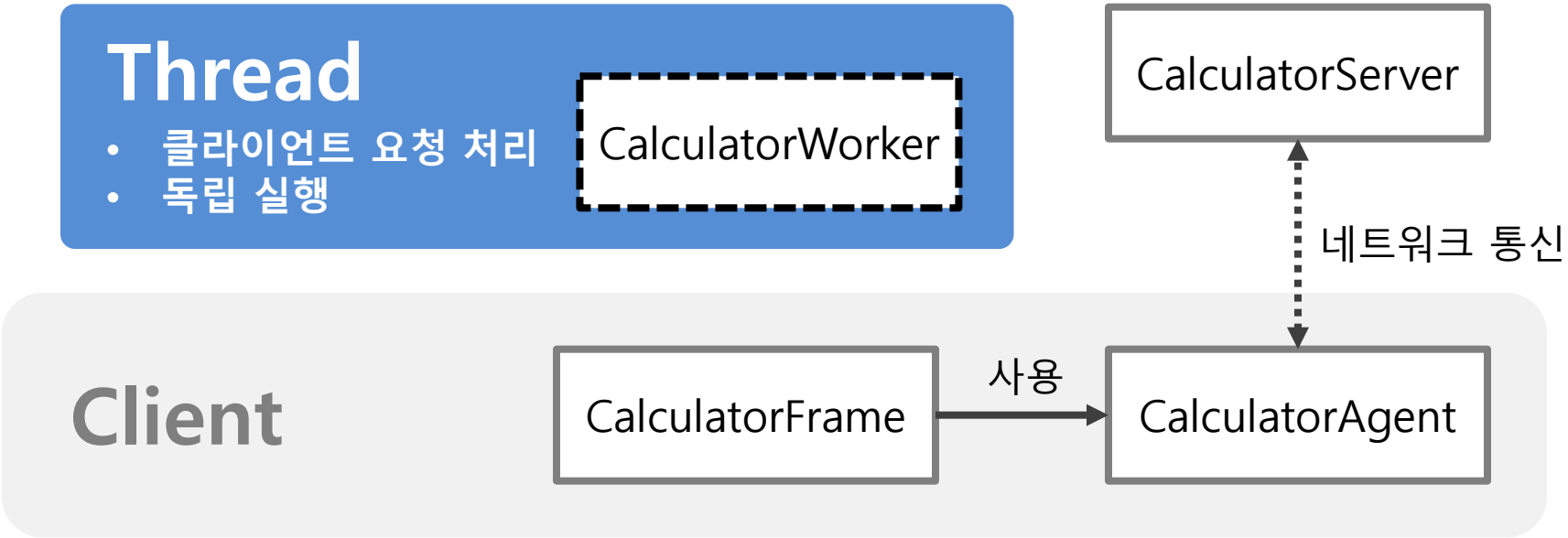
1.3 다중 클라이언트 요청처리

클라이언트의 요청 처리 부분을 별도의 작업으로 분리



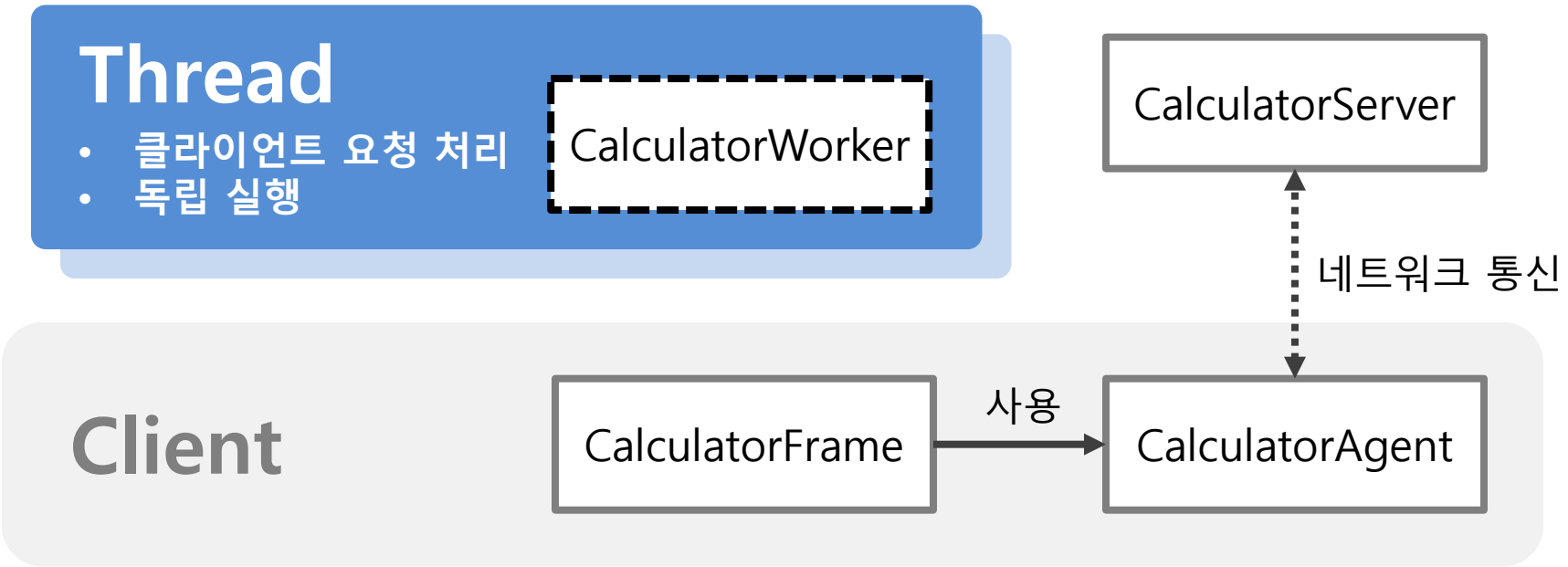
1.3 다중 클라이언트 요청처리

클라이언트의 요청 처리 부분을 별도의 작업으로 분리



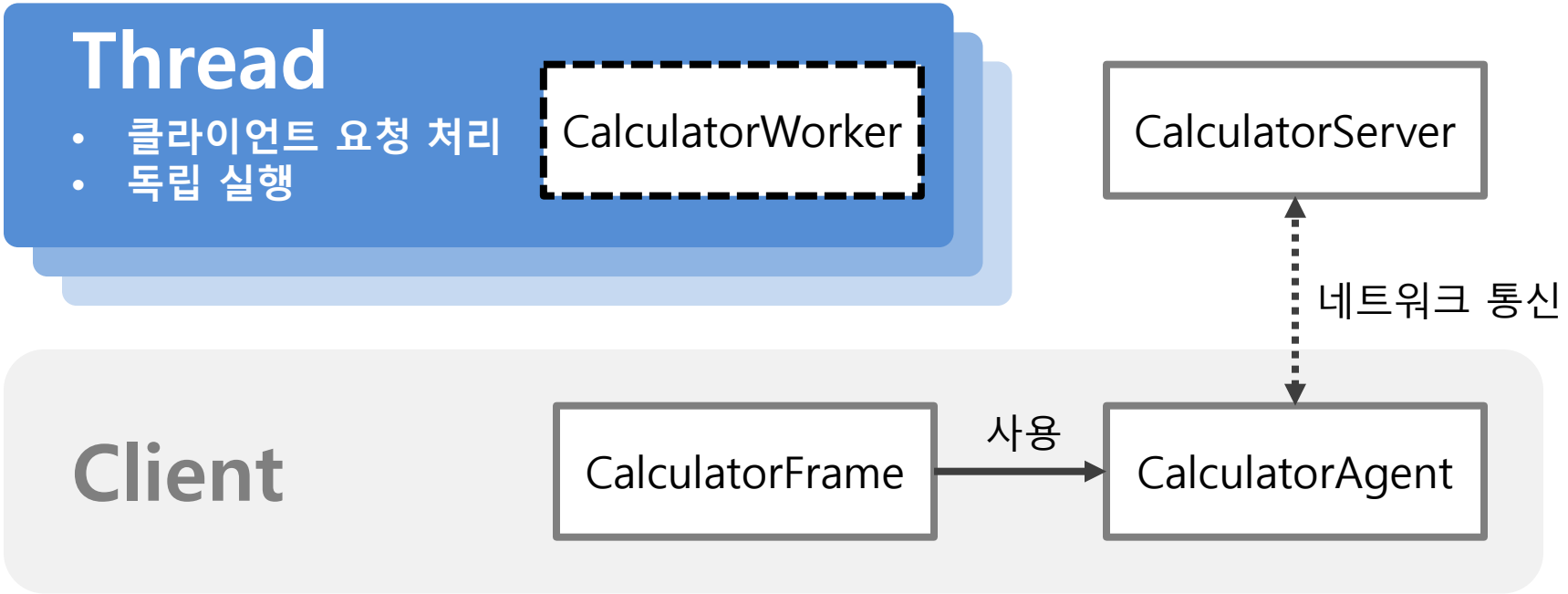
1.3 다중 클라이언트 요청처리

클라이언트의 요청 처리 부분을 별도의 작업으로 분리



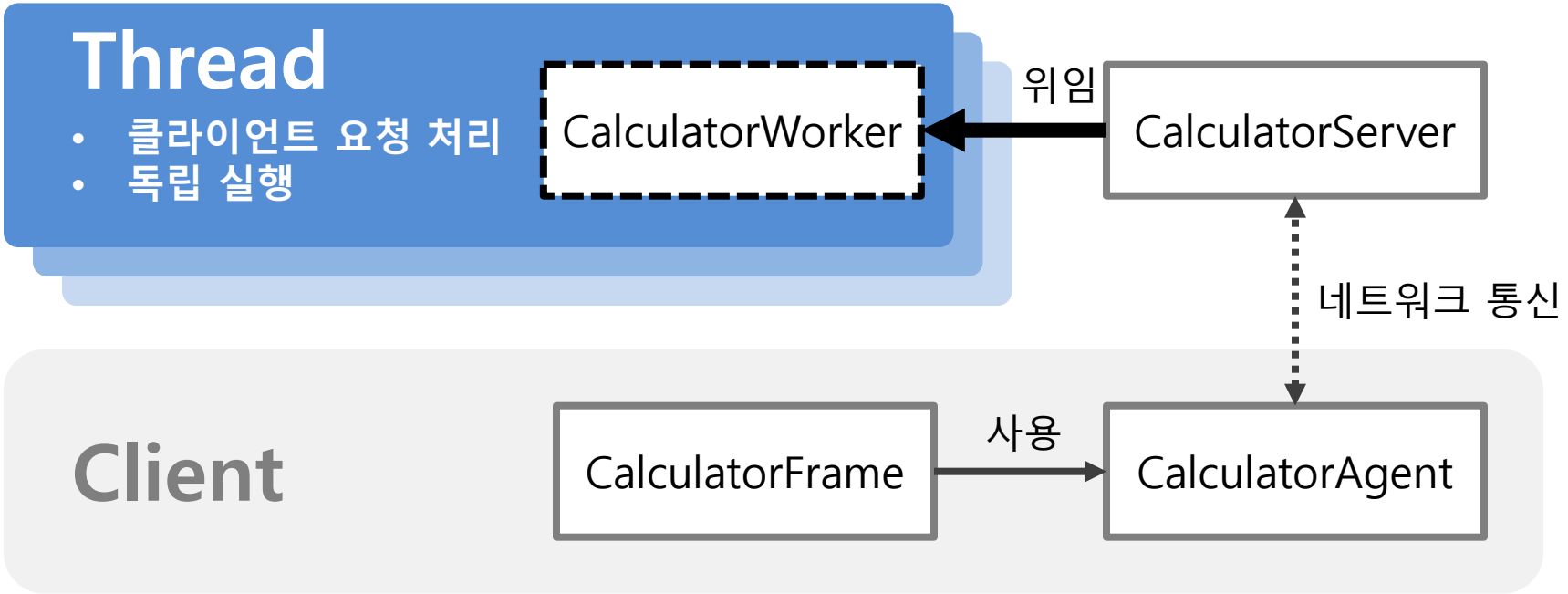
1.3 다중 클라이언트 요청처리

클라이언트의 요청 처리 부분을 별도의 작업으로 분리



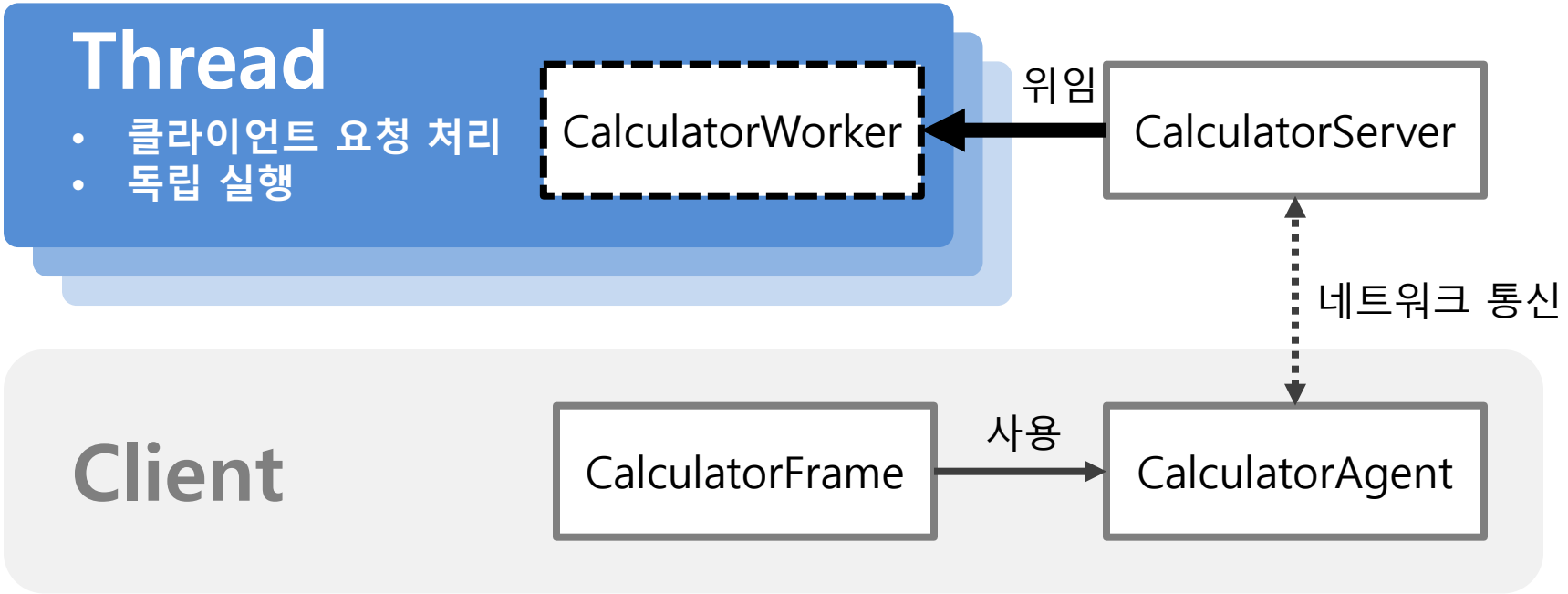
1.3 다중 클라이언트 요청처리

클라이언트의 요청 처리 부분을 별도의 작업으로 분리



1.3 다중 클라이언트 요청처리

클라이언트의 요청 처리 부분을 별도의 작업으로 분리

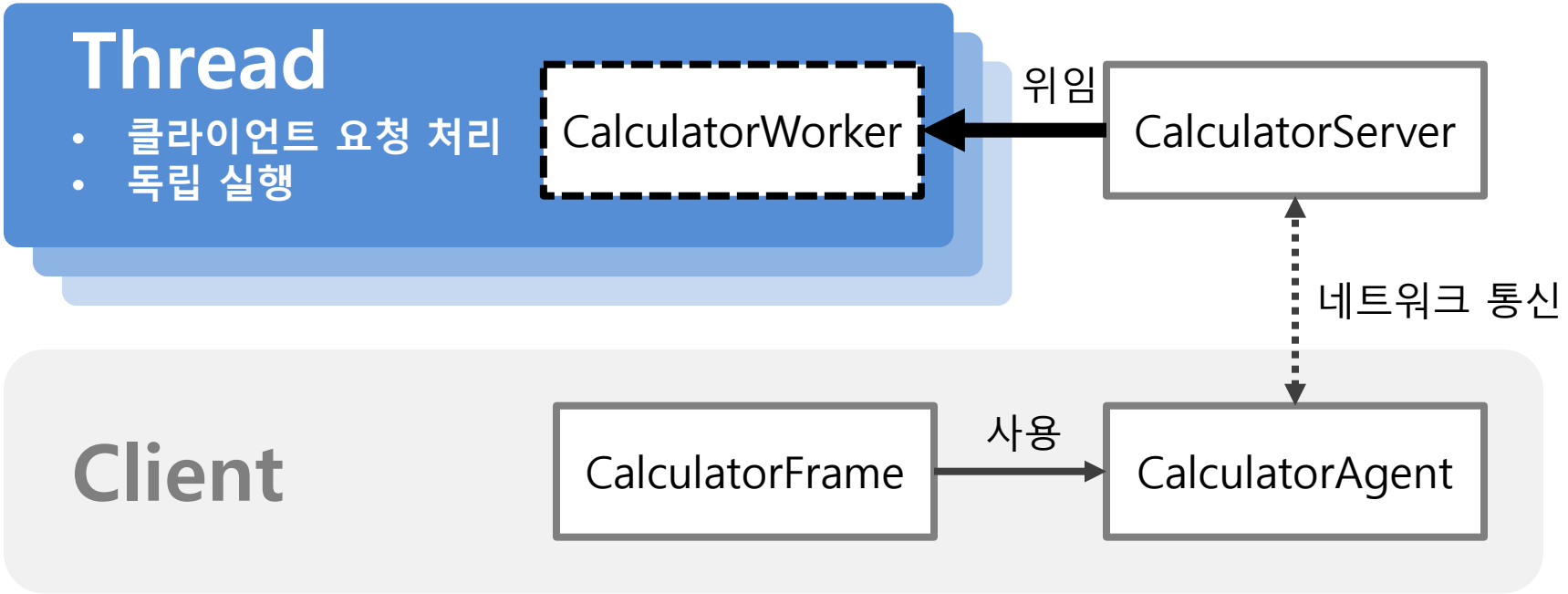


특징

- **각 클라이언트 요청처리를 개별 스레드가 담당**

1.3 다중 클라이언트 요청처리

클라이언트의 요청 처리 부분을 별도의 작업으로 분리

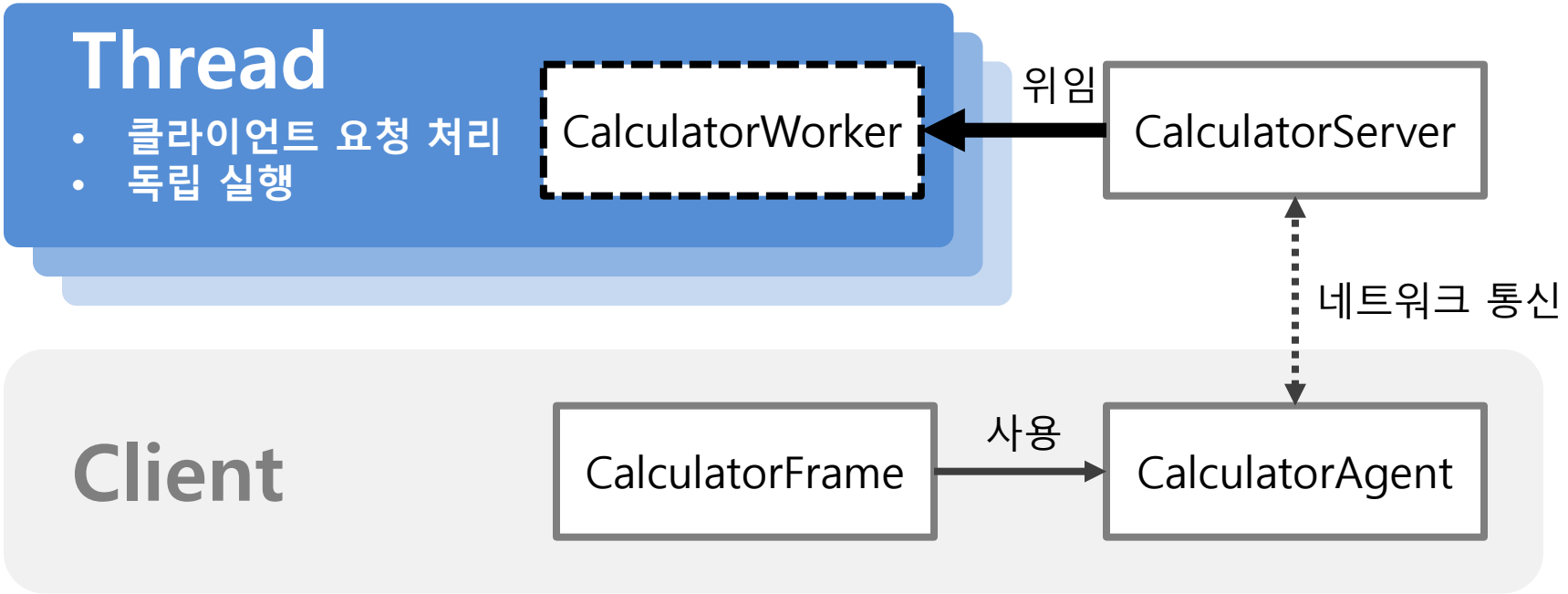


특징

- 각 클라이언트 요청처리를 개별 스레드가 담당
- **다중 클라이언트의 요청이 병행 처리**

1.3 다중 클라이언트 요청처리

클라이언트의 요청 처리 부분을 별도의 작업으로 분리

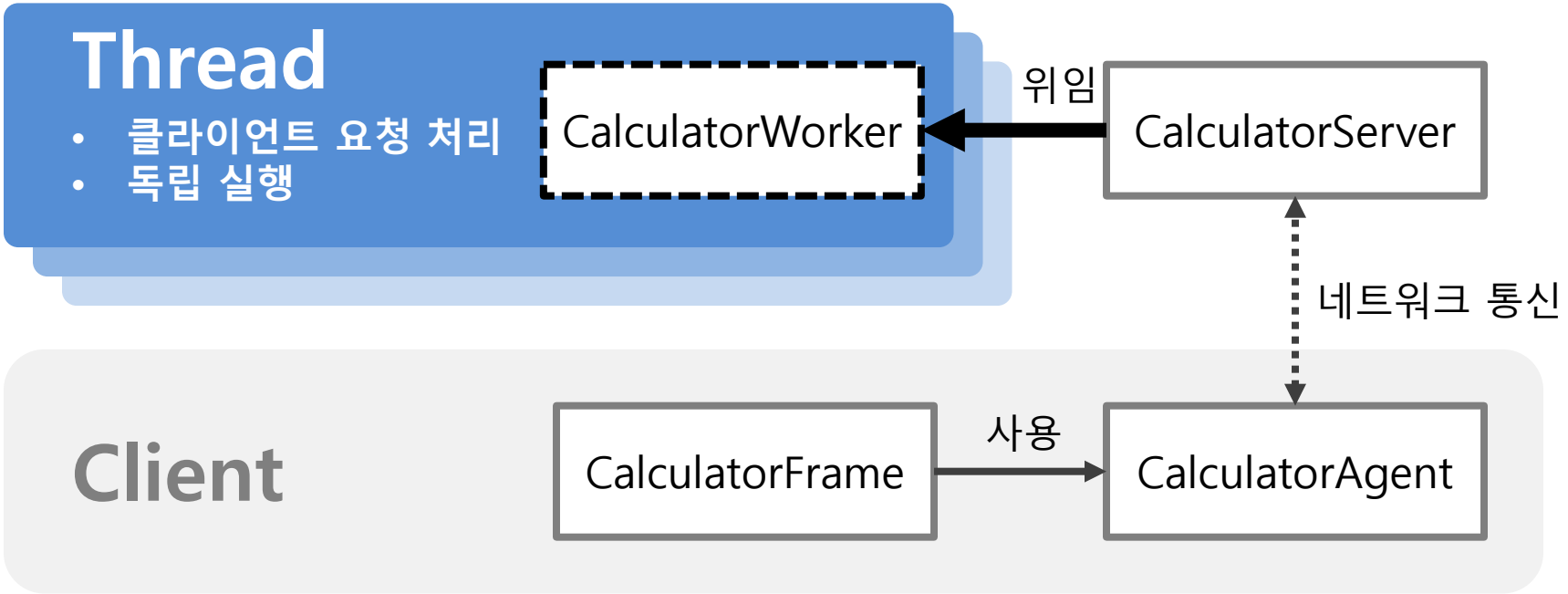


특징

- 각 클라이언트 요청처리를 개별 스레드가 담당
- 다중 클라이언트의 요청이 병행 처리 ➔ **동시 작업 가능**

1.3 다중 클라이언트 요청처리

클라이언트의 요청 처리 부분을 별도의 작업으로 분리



특징

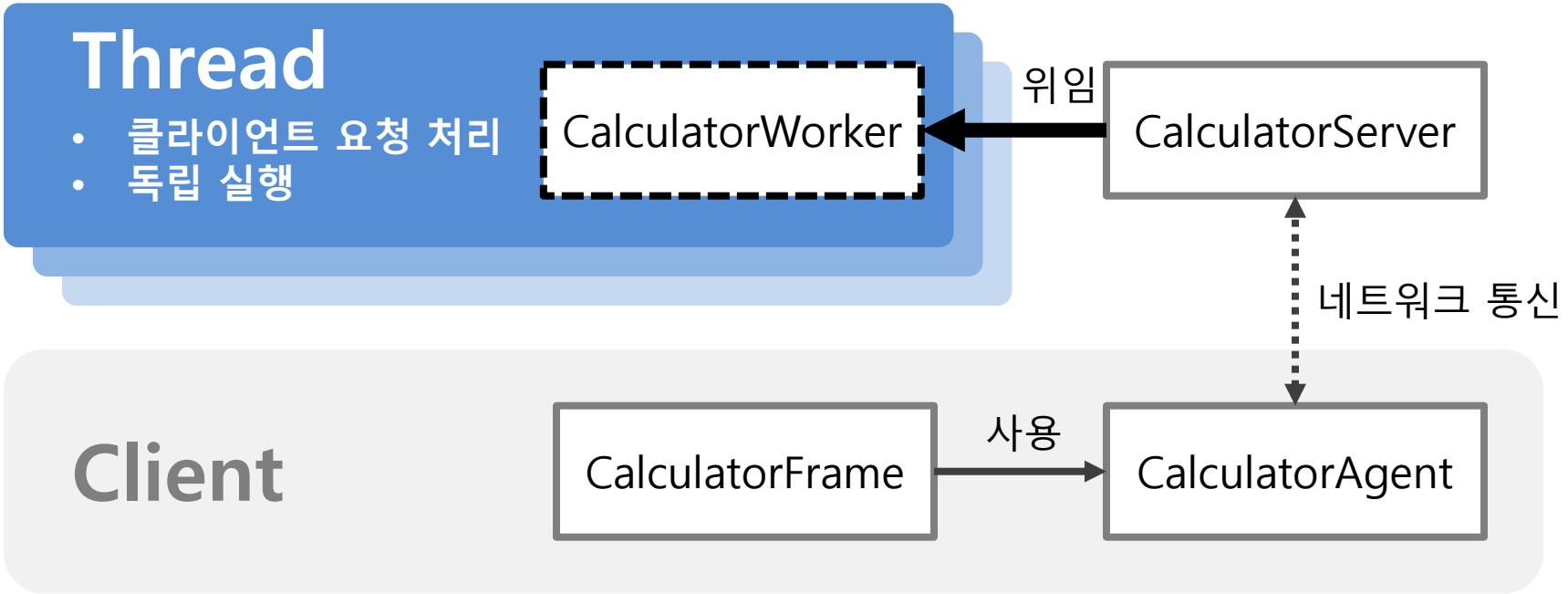
- 각 클라이언트 요청처리를 개별 스레드가 담당
- 다중 클라이언트의 요청이 병행 처리 ➔ 동시 작업 가능

문제점

- **소켓 및 스레드 도입**

1.3 다중 클라이언트 요청처리

클라이언트의 요청 처리 부분을 별도의 작업으로 분리



특징

- 각 클라이언트 요청처리를 개별 스레드가 담당
- 다중 클라이언트의 요청이 병행 처리 ➔ 동시 작업 가능

문제점

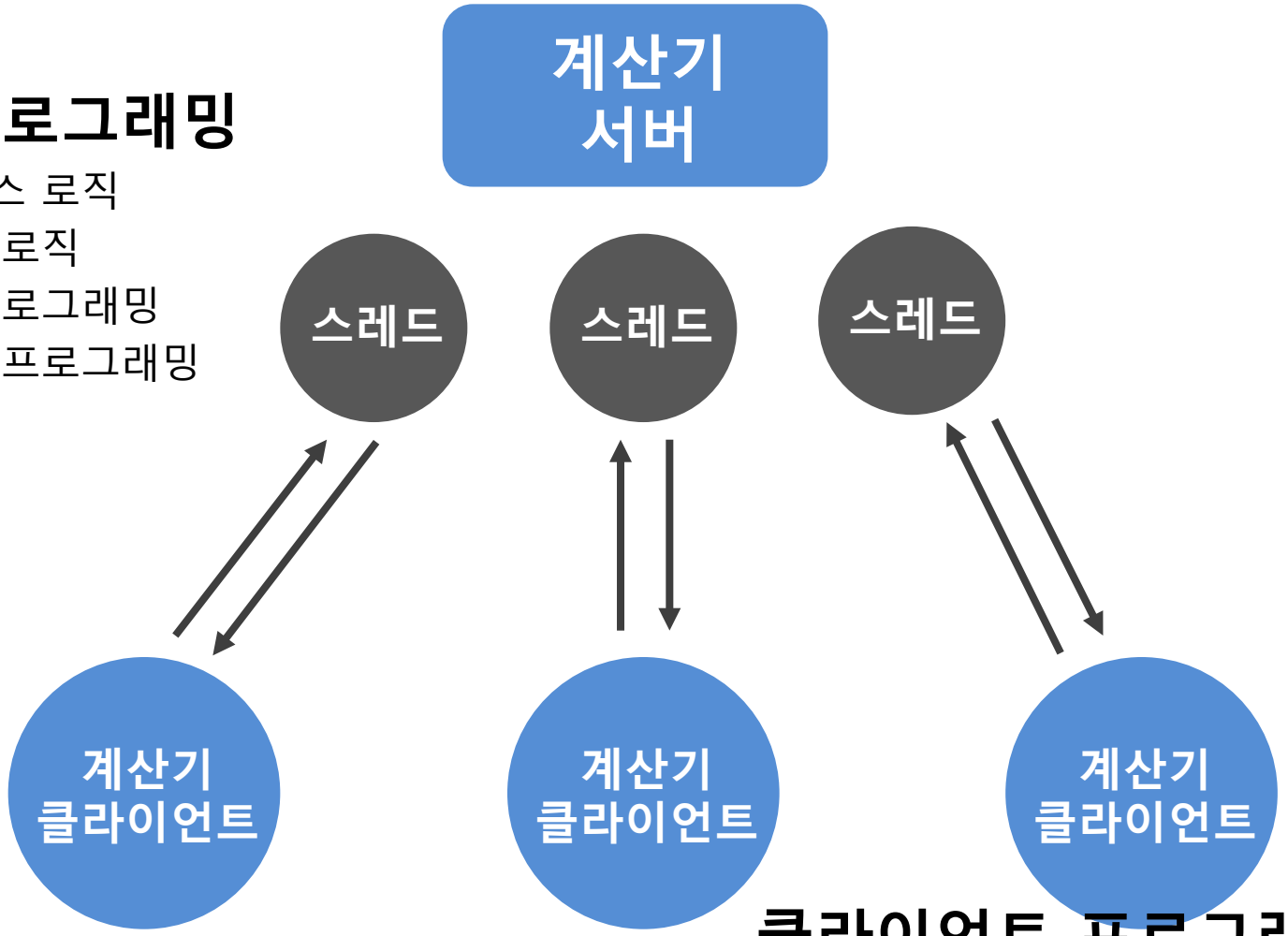
- 소켓 및 스레드 도입 ➔ **프로그래밍이 복잡**

1.5 웹 애플리케이션 아키텍처의 특징

1.5 웹 애플리케이션 아키텍처의 특징

서버 프로그래밍

- 비즈니스 로직
- 데이터 로직
- 소켓 프로그래밍
- 스레드 프로그래밍



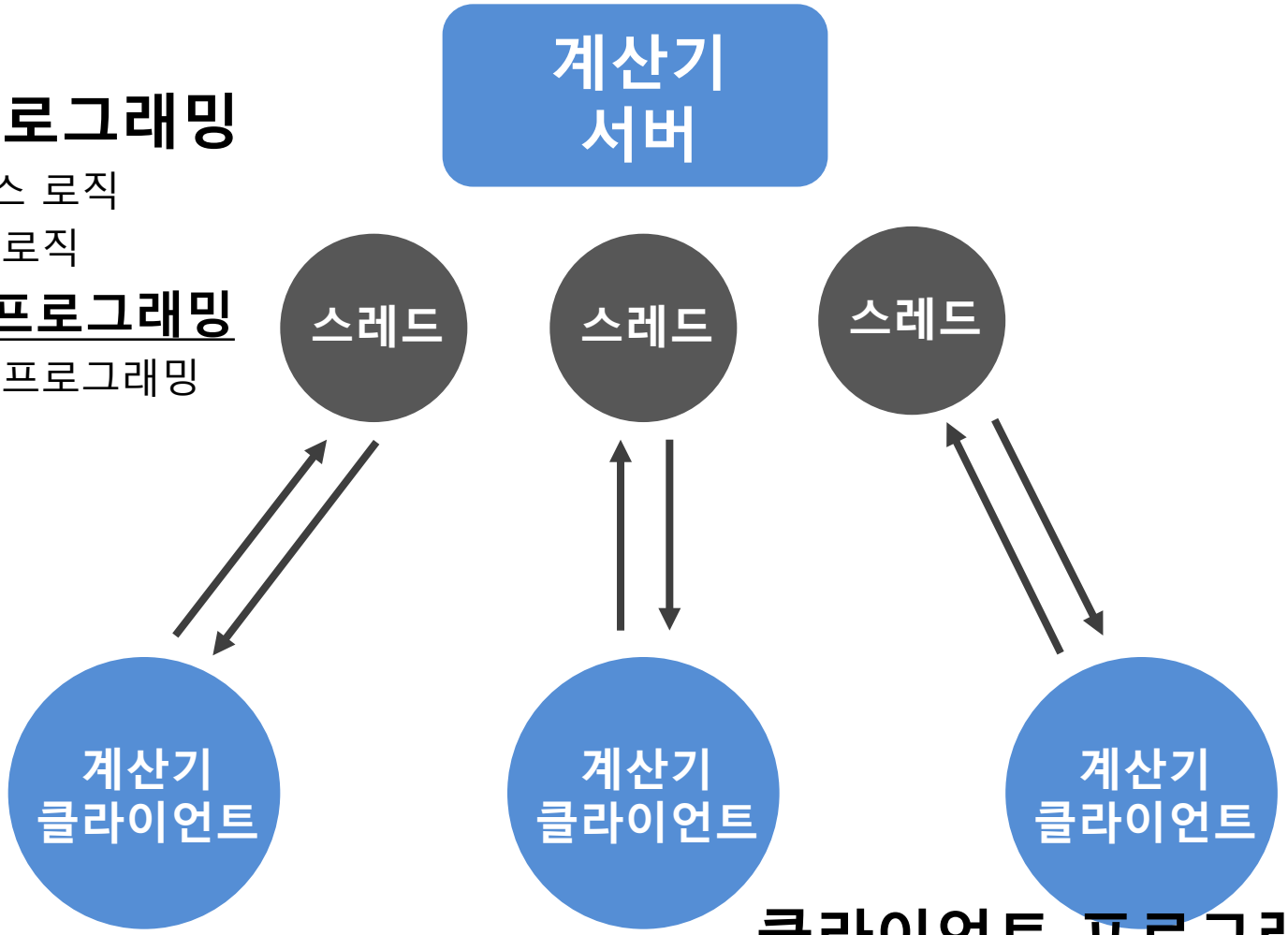
클라이언트 프로그래밍

- 프리젠테이션 로직
- 소켓 프로그래밍

1.5 웹 애플리케이션 아키텍처의 특징

서버 프로그래밍

- 비즈니스 로직
- 데이터 로직
- **소켓 프로그래밍**
- 스레드 프로그래밍



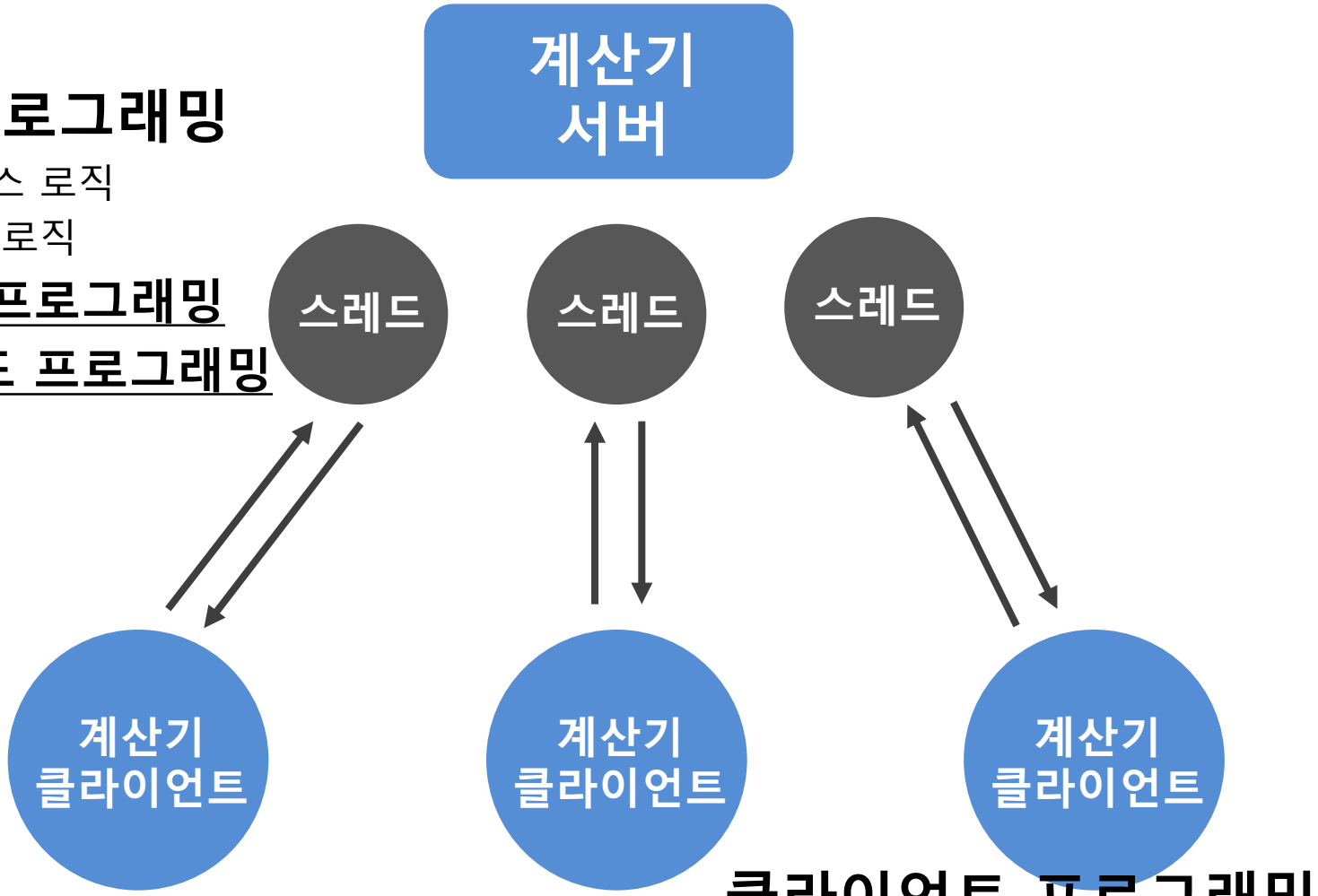
클라이언트 프로그래밍

- 프리젠테이션 로직
- **소켓 프로그래밍**

1.5 웹 애플리케이션 아키텍처의 특징

서버 프로그래밍

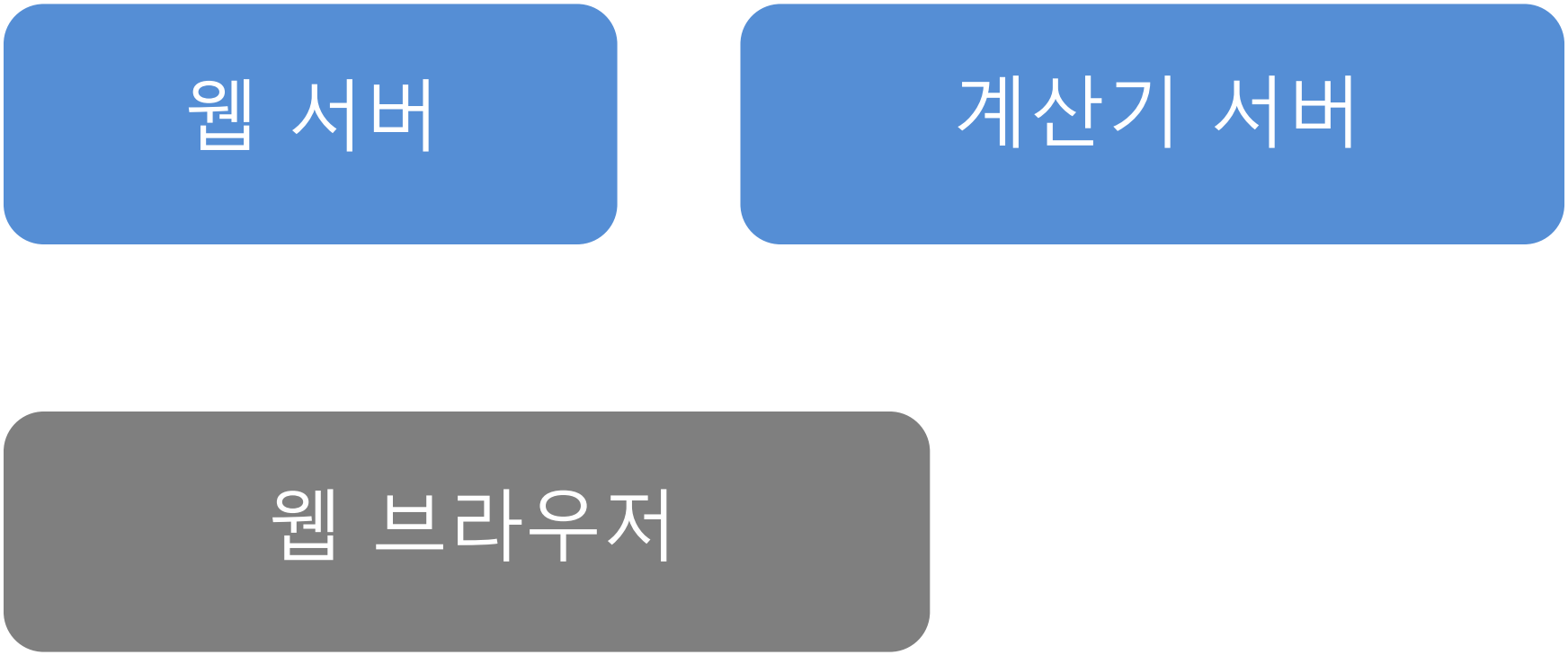
- 비즈니스 로직
- 데이터 로직
- **소켓 프로그래밍**
- **스레드 프로그래밍**



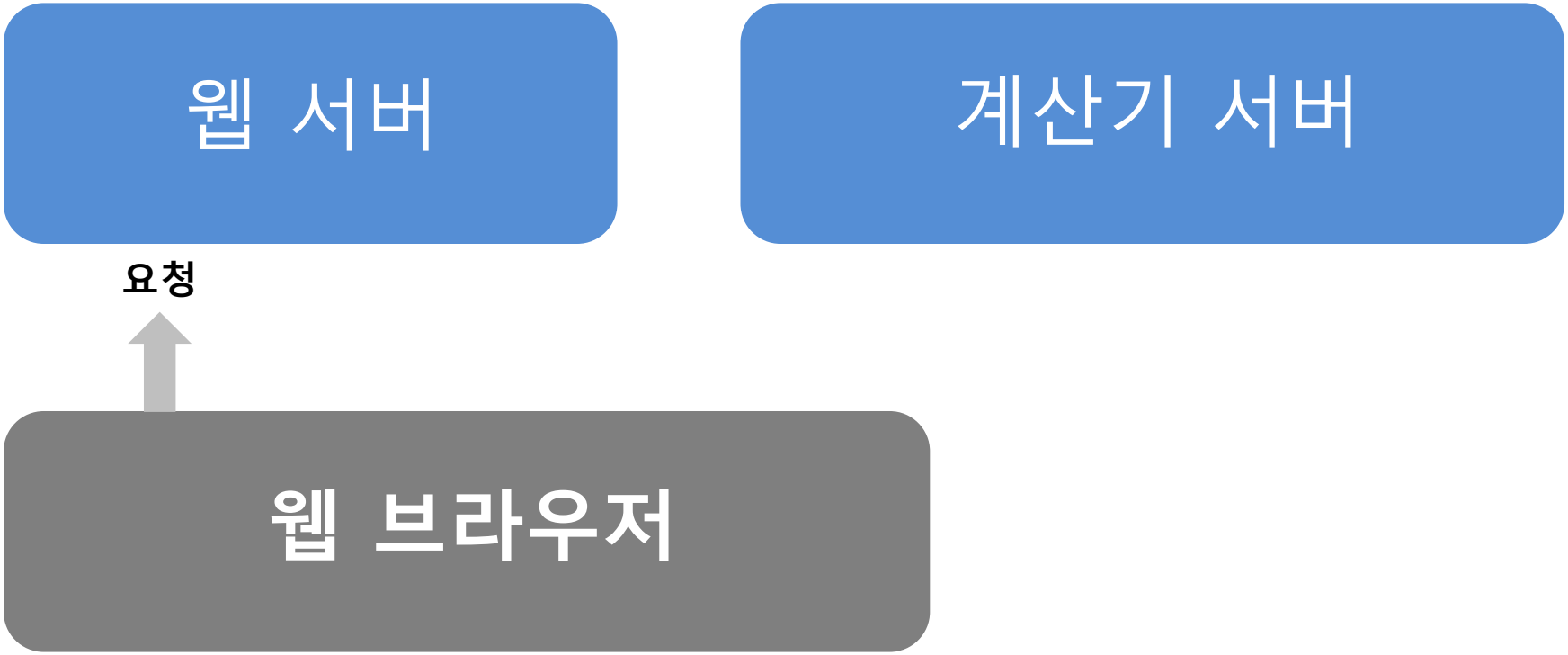
클라이언트 프로그래밍

- 프리젠테이션 로직
- **소켓 프로그래밍**

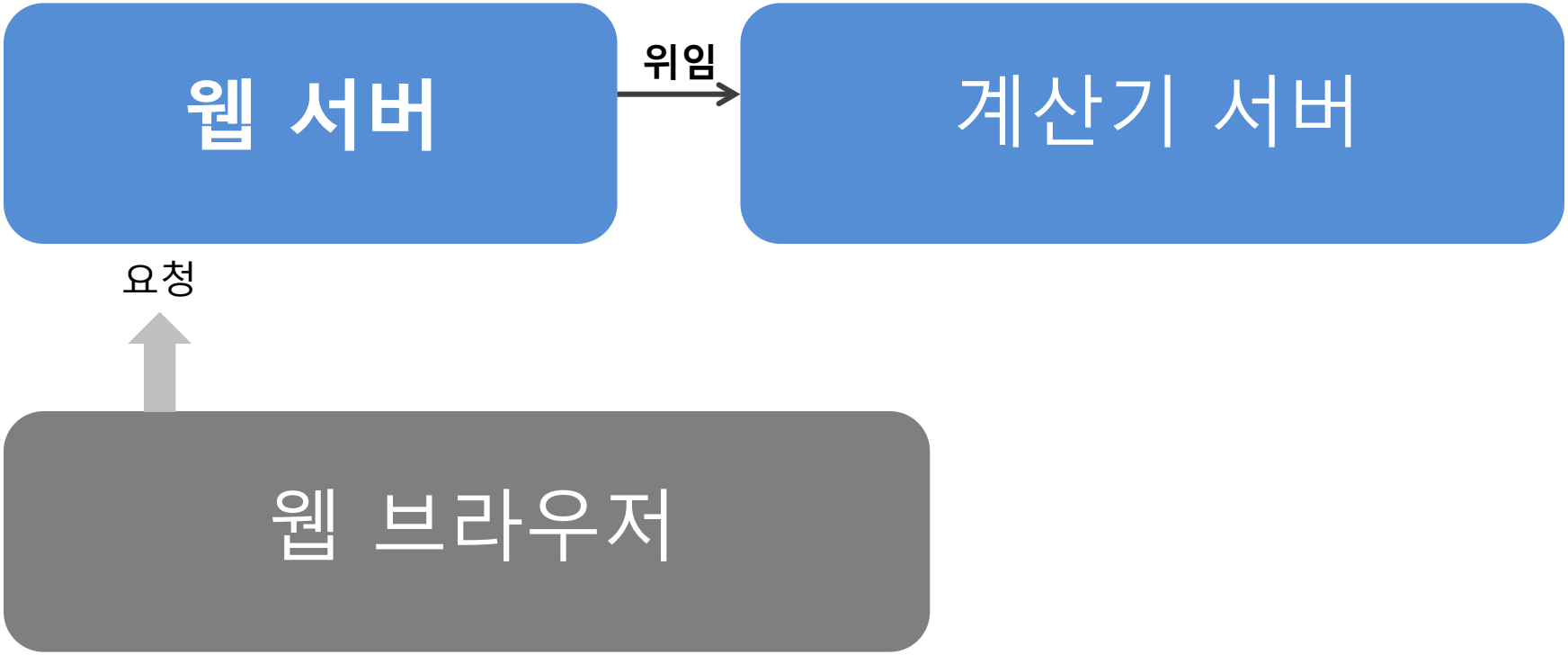
웹 애플리케이션 구조



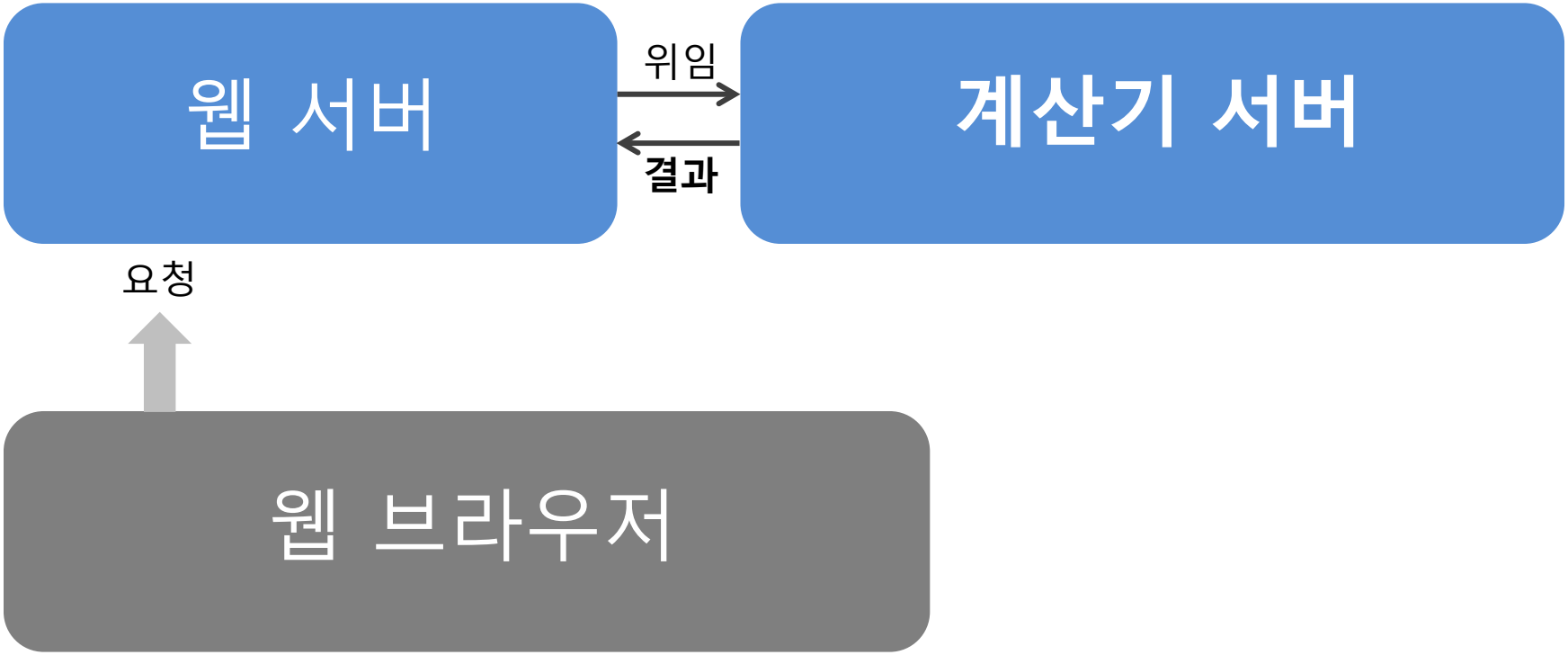
웹 애플리케이션 구조



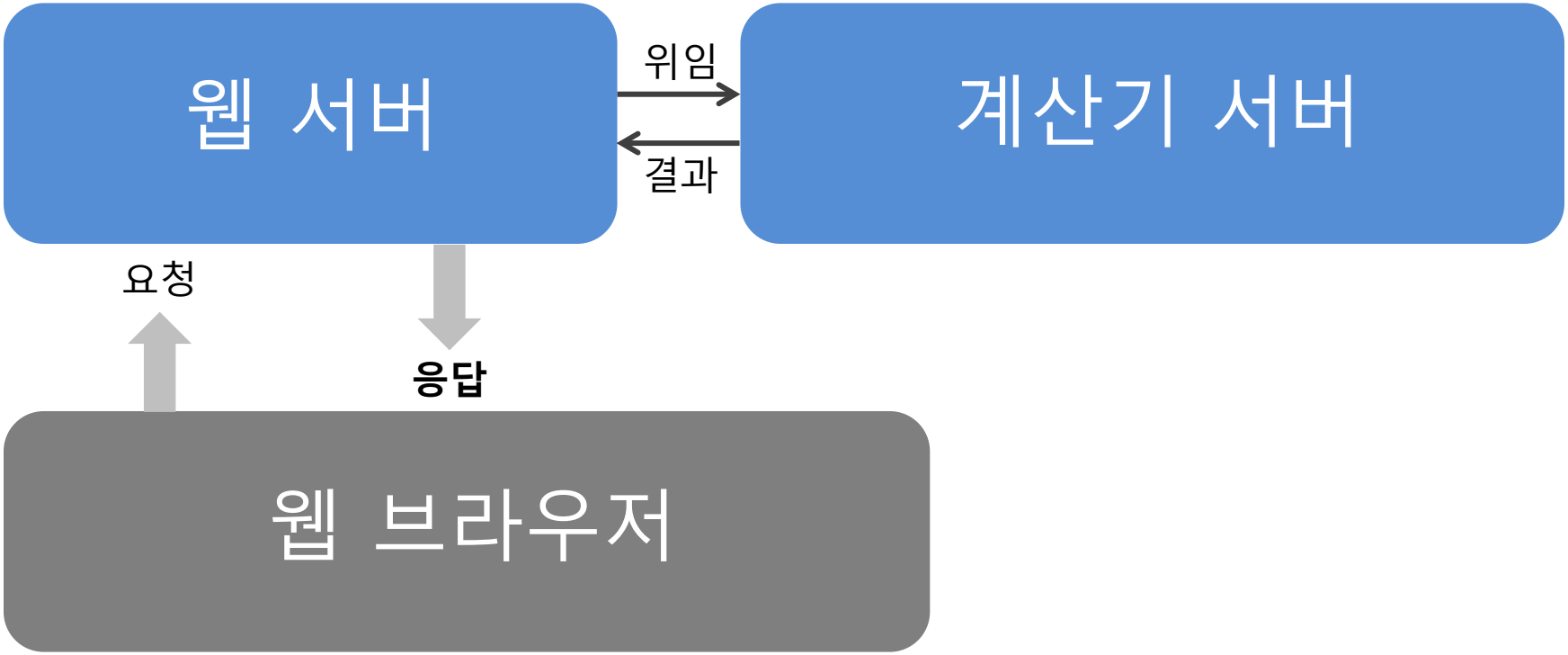
웹 애플리케이션 구조



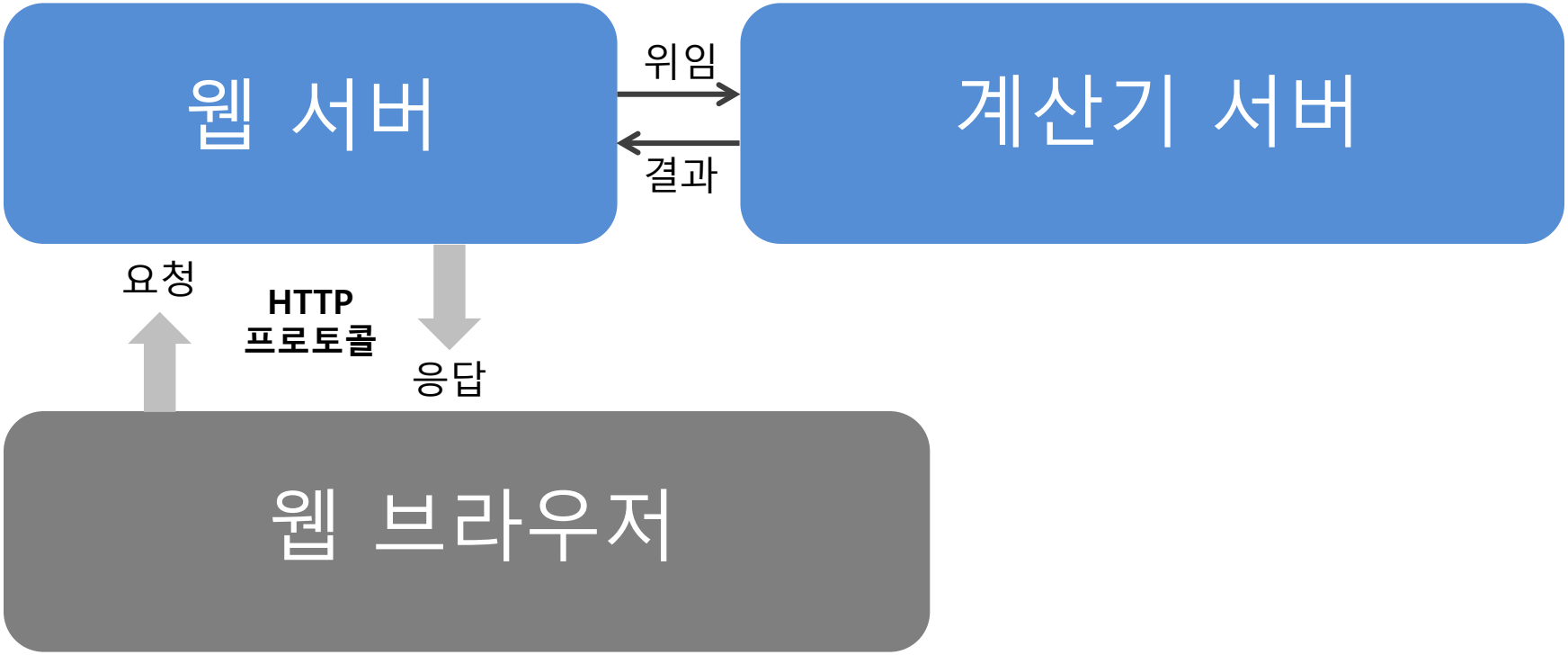
웹 애플리케이션 구조



웹 애플리케이션 구조

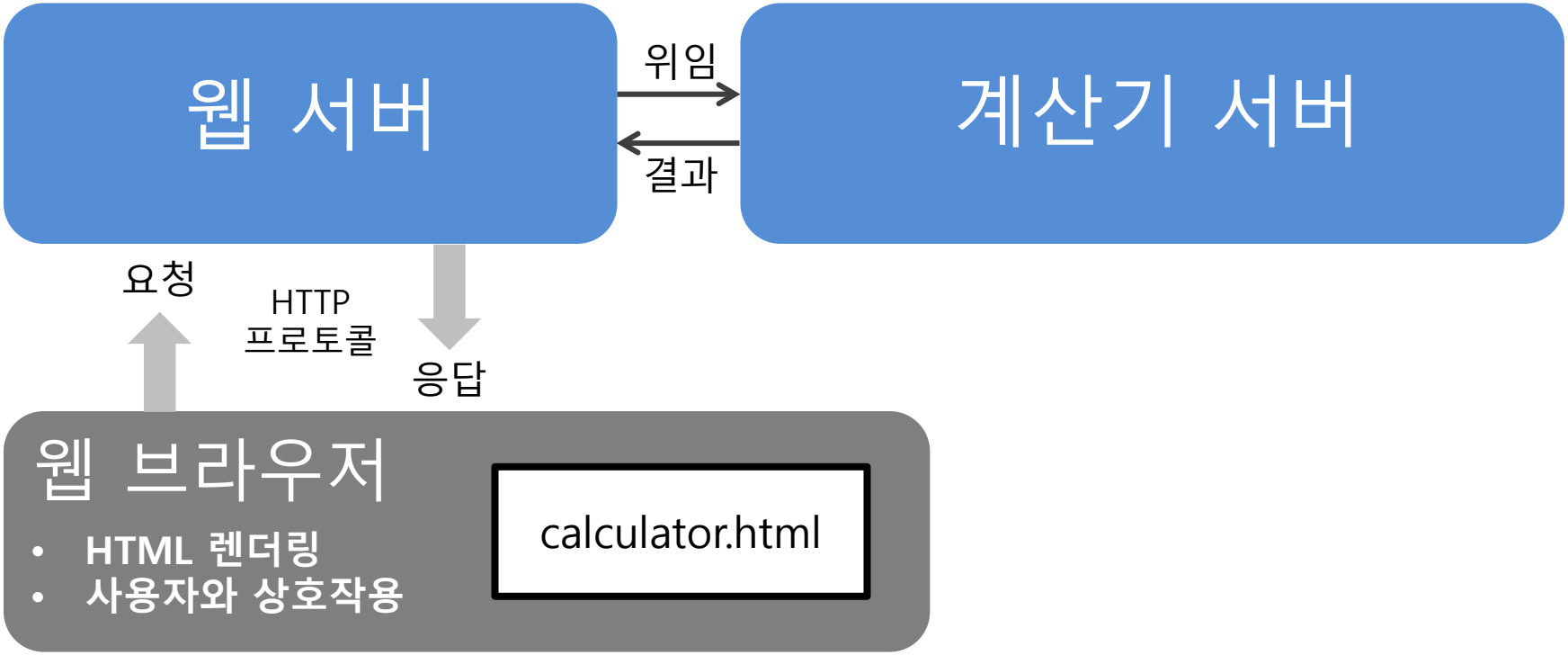


웹 애플리케이션 구조

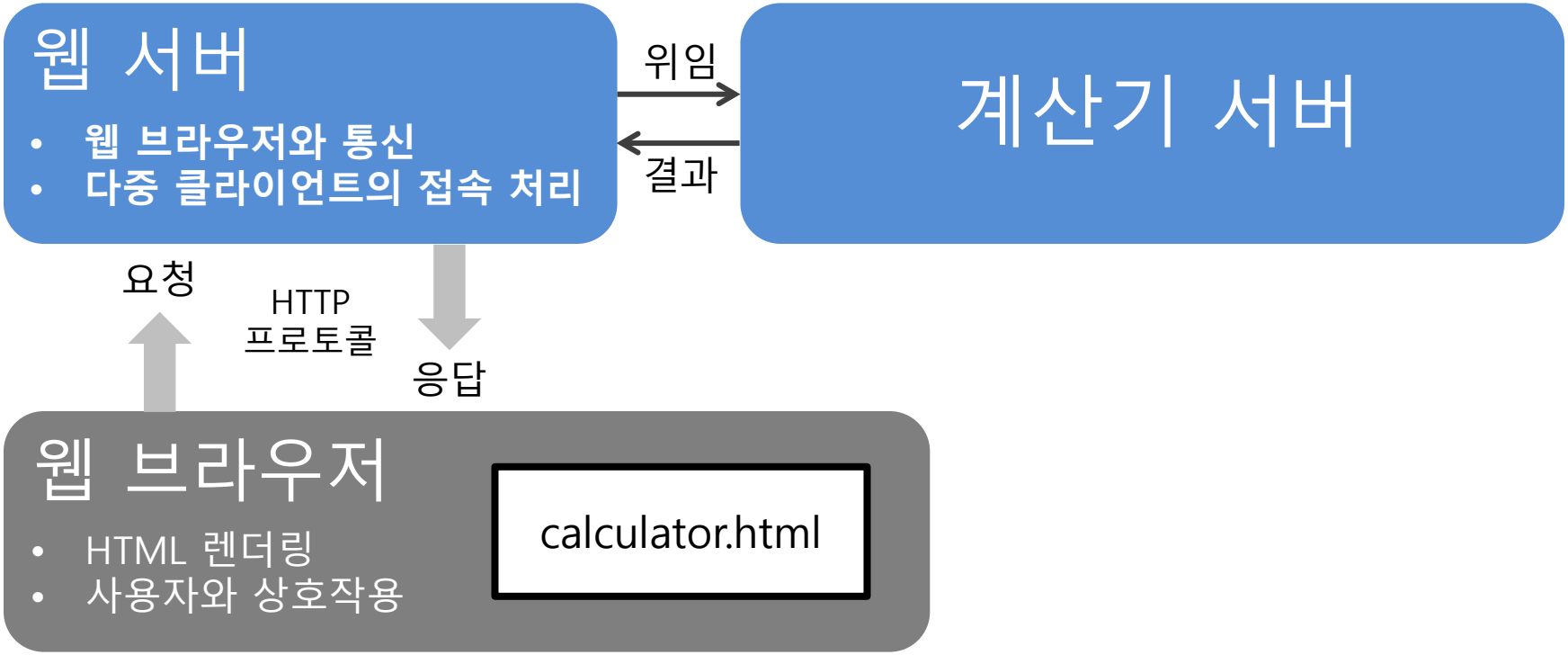


웹 애플리케이션

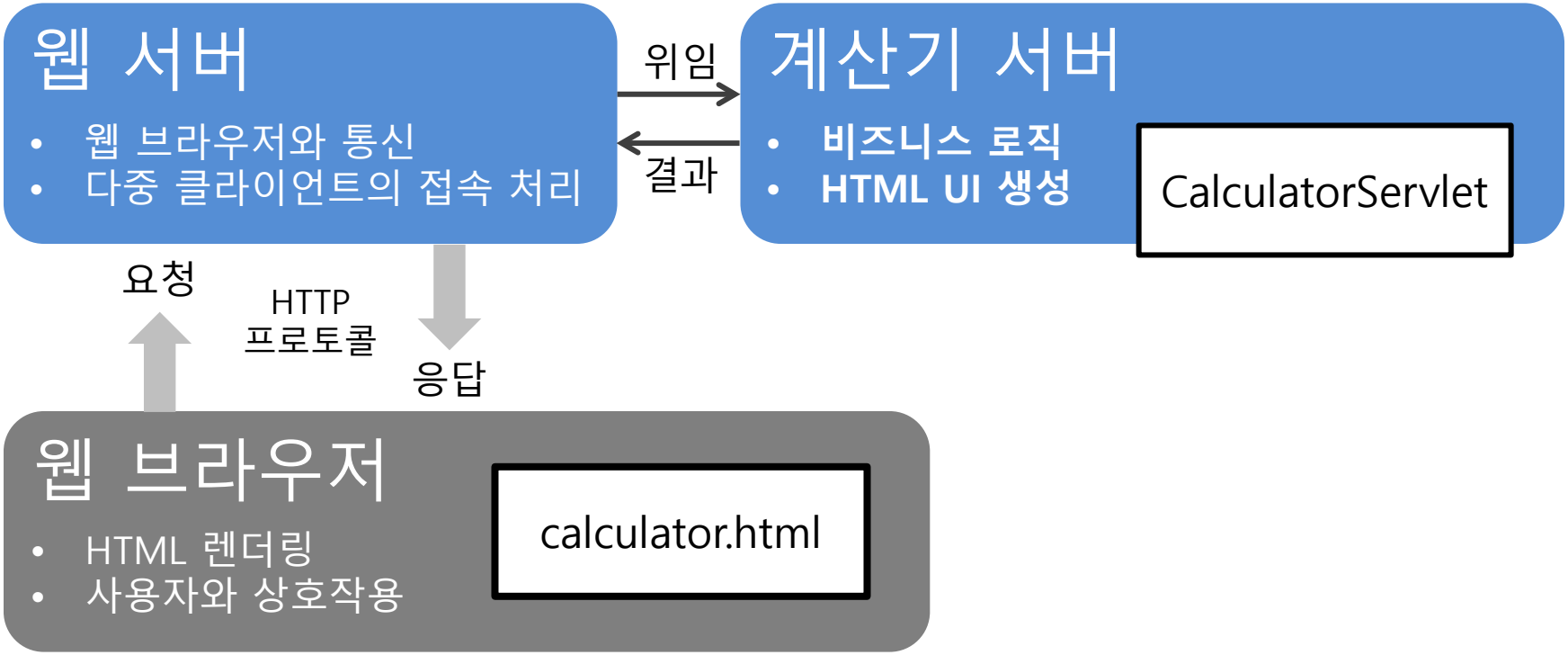
웹 애플리케이션 구조



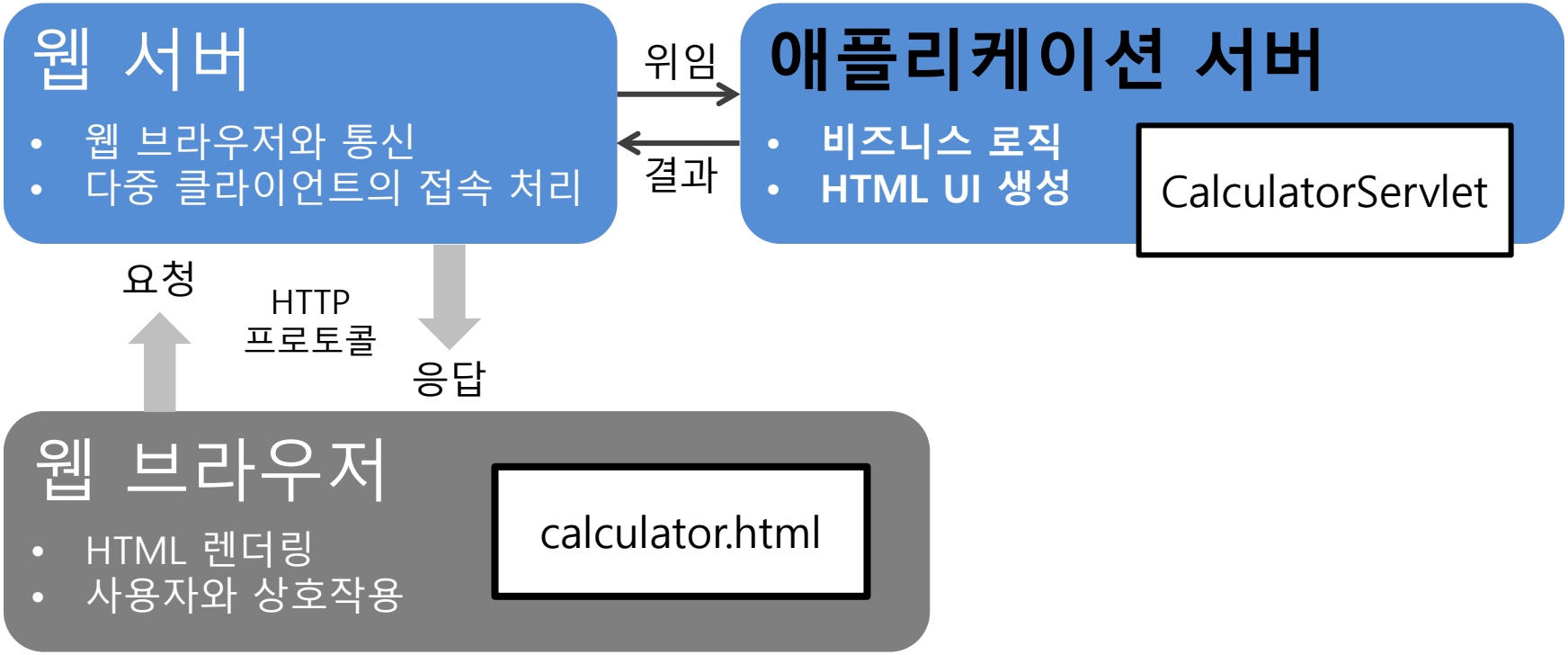
웹 애플리케이션 구조


웹 서버
웹 브라우저와 통신
다중 클라이언트의 접속 처리
위임
결과
계산기 서버
요청
HTTP
프로토콜
응답
웹 브라우저
HTML 렌더링
사용자와 상호작용
calculator.html

웹 애플리케이션 구조

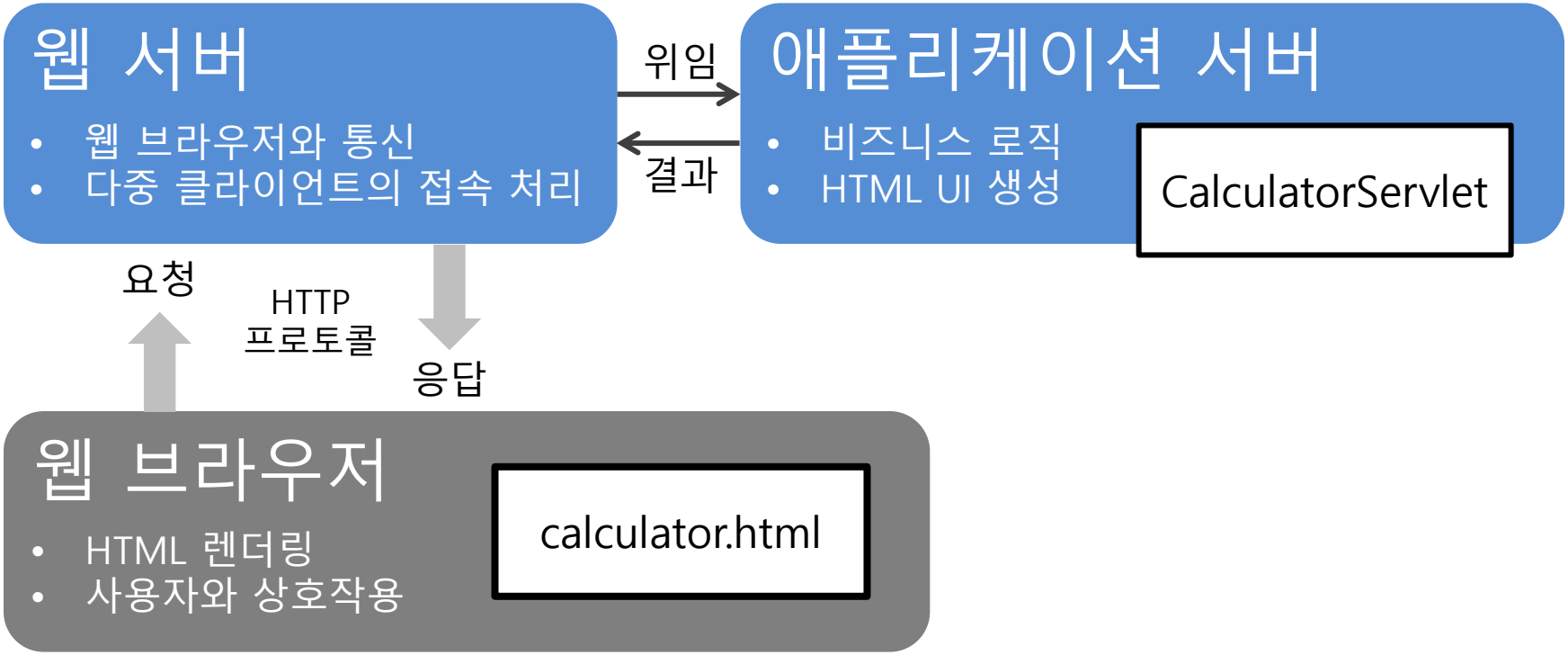

웹 서버
• 웹 브라우저와 통신
• 다중 클라이언트의 접속 처리
위임
결과
계산기 서버
• 비즈니스 로직
• HTML UI 생성
CalculatorServlet
요청
HTTP
프로토콜
응답
웹 브라우저
• HTML 렌더링
• 사용자와 상호작용
calculator.html

웹 애플리케이션 구조


웹 서버
• 웹 브라우저와 통신
• 다중 클라이언트의 접속 처리
위임
결과
애플리케이션 서버
• 비즈니스 로직
• HTML UI 생성
CalculatorServlet
요청
HTTP
프로토콜
응답
웹 브라우저
• HTML 렌더링
• 사용자와 상호작용
calculator.html

1.5 웹 애플리케이션 아키텍처의 특징

웹 애플리케이션 구조

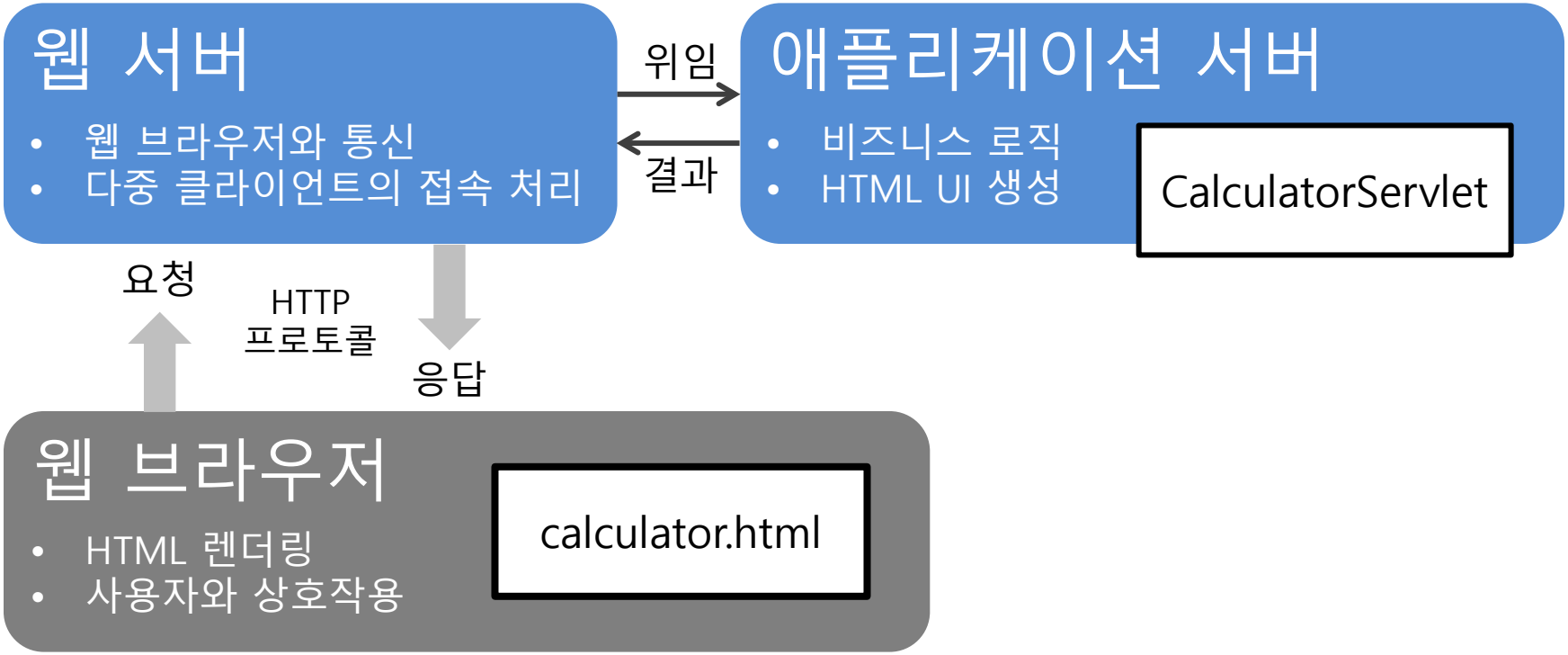


특징

- **웹 기술 도입**

1.5 웹 애플리케이션 아키텍처의 특징

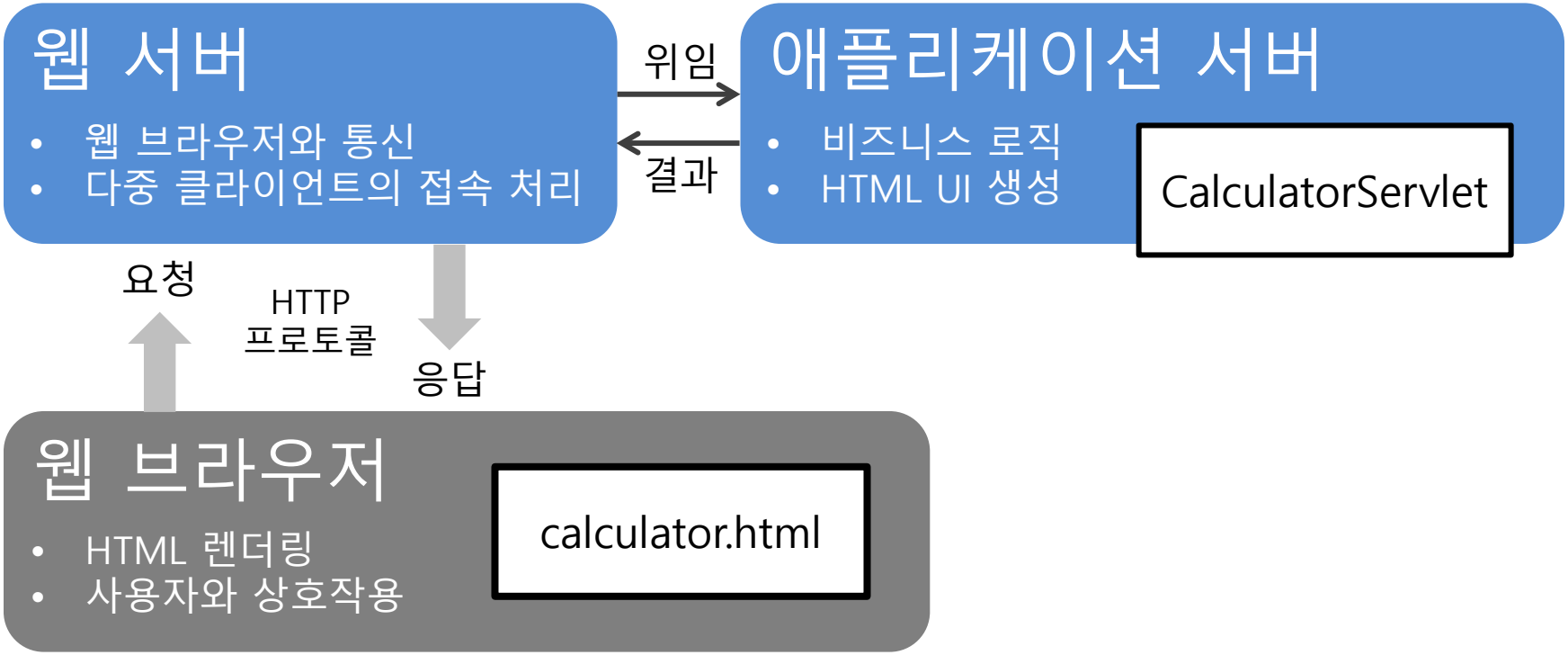
웹 애플리케이션 구조



특징

- 웹 기술 도입 ➔ **소켓 & 멀티 스레드 프로그래밍에서 탈출**

웹 애플리케이션 구조

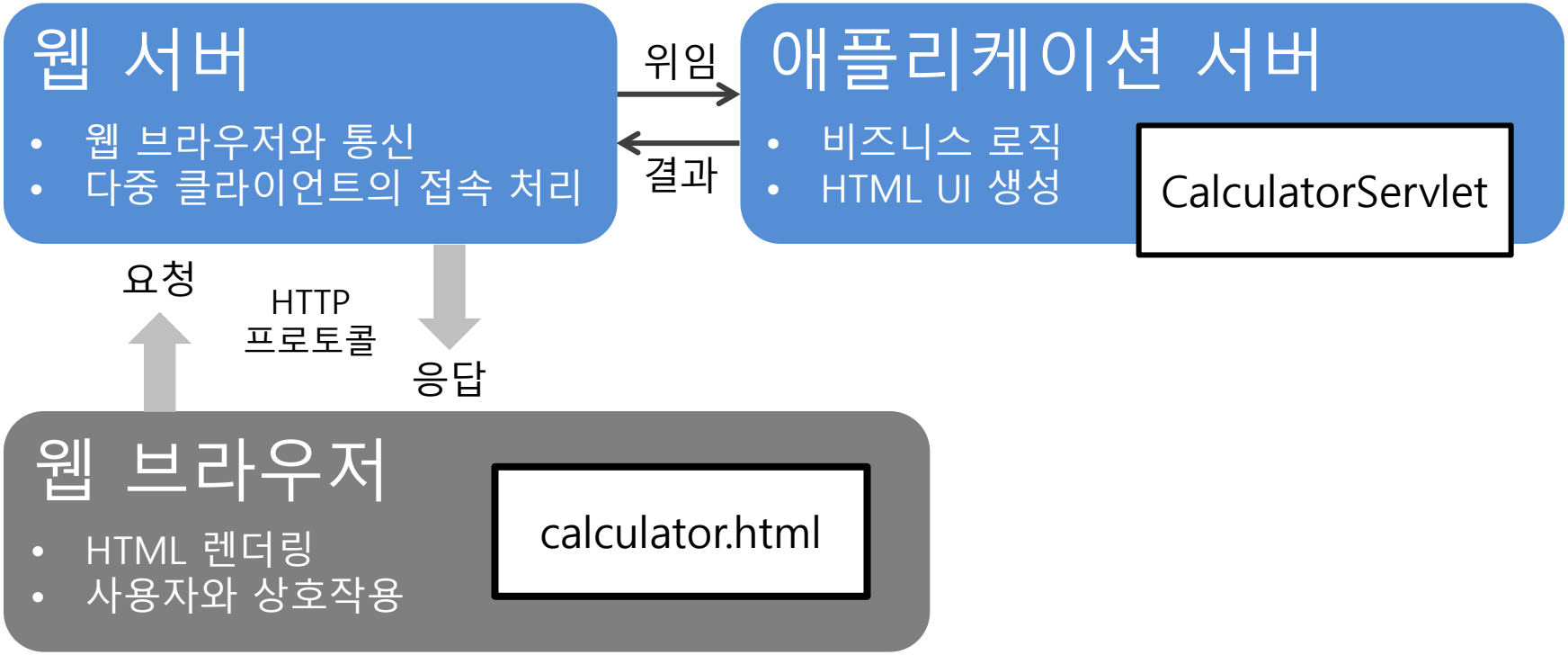


특징

- 웹 기술 도입 ➔ 소켓 & 멀티 스레드 프로그래밍에서 탈출
- **서버에 애플리케이션 배포 및 실행**

1.5 웹 애플리케이션 아키텍처의 특징

웹 애플리케이션 구조



특징

- 웹 기술 도입 ➔ 소켓 & 멀티 스레드 프로그래밍에서 탈출
- 서버에 애플리케이션 배포 및 실행 ➔ **기능 변경 및 추가가 용이**

1.4 클라이언트·서버 아키텍처의 진화

1.4 클라이언트·서버 아키텍처의 진화

전통적인 클라이언트·서버 구조

Server

- 데이터 처리

1.4 클라이언트·서버 아키텍처의 진화

전통적인 클라이언트·서버 구조

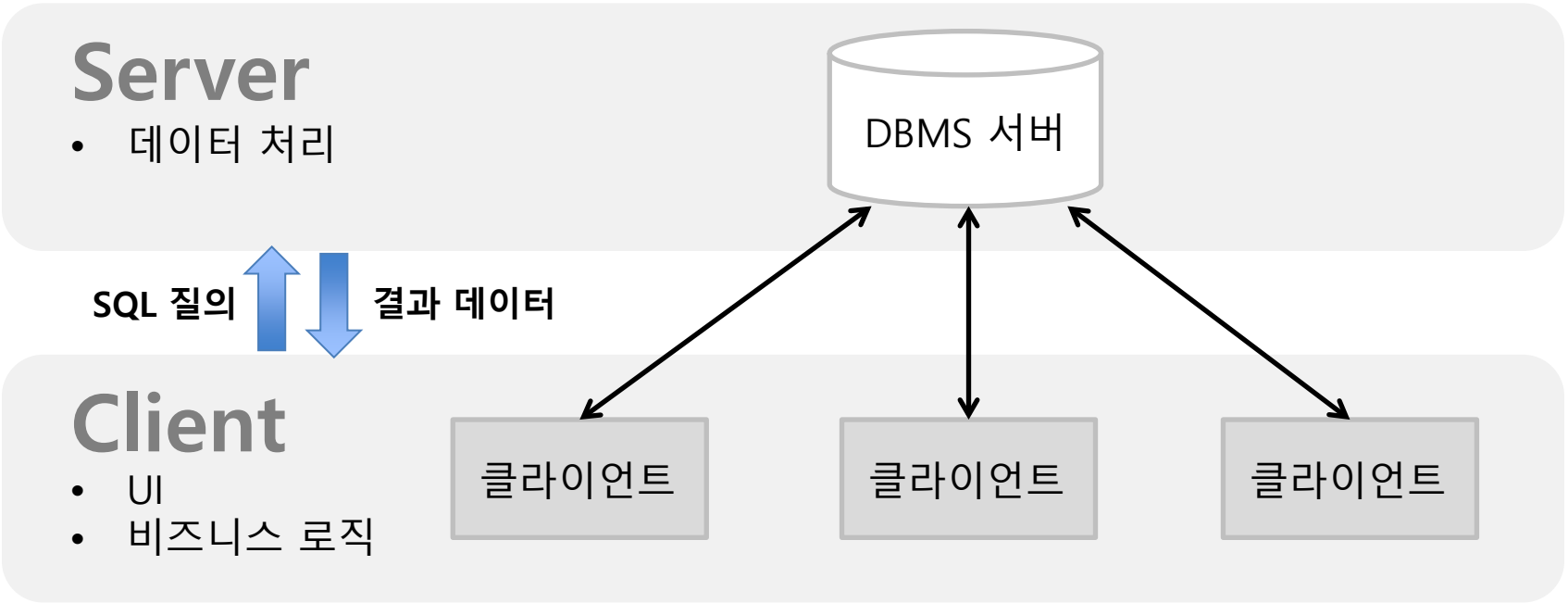
Server
- 데이터 처리

Client
- **UI**
- **비즈니스 로직**

클라이언트 **클라이언트** **클라이언트**

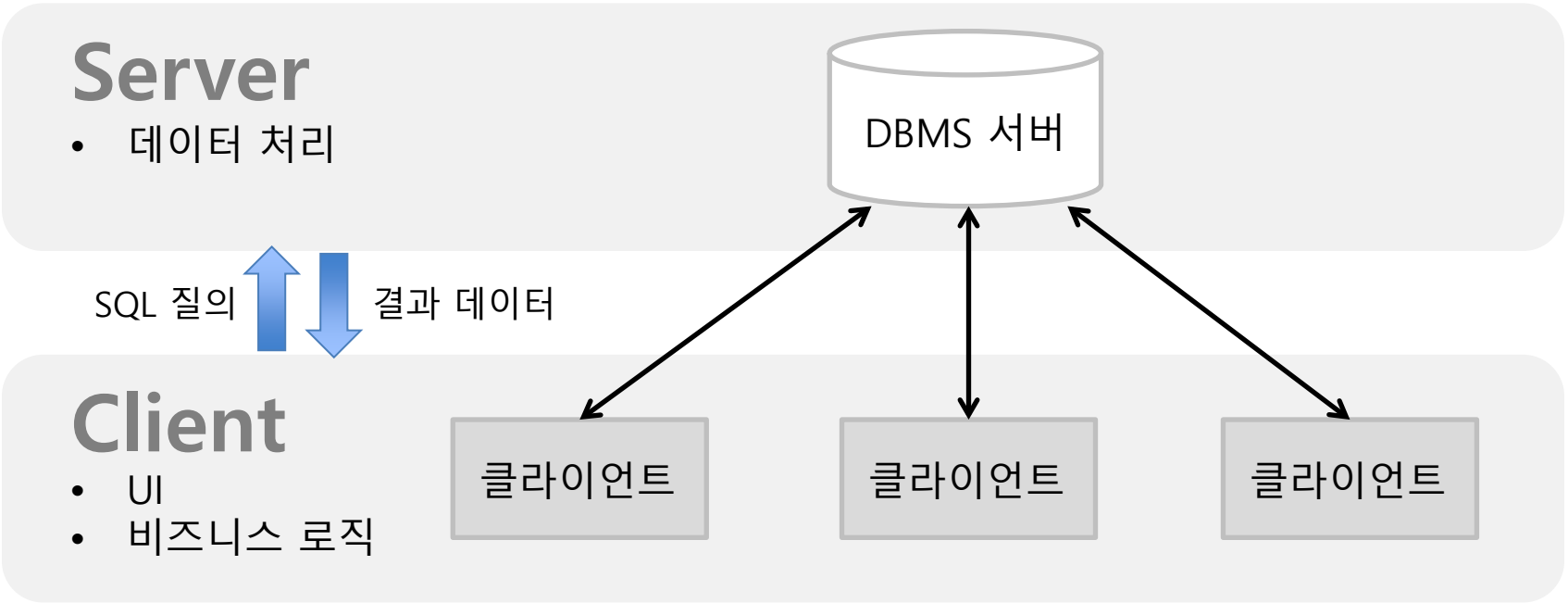
1.4 클라이언트·서버 아키텍처의 진화

전통적인 클라이언트·서버 구조



1.4 클라이언트·서버 아키텍처의 진화

전통적인 클라이언트·서버 구조

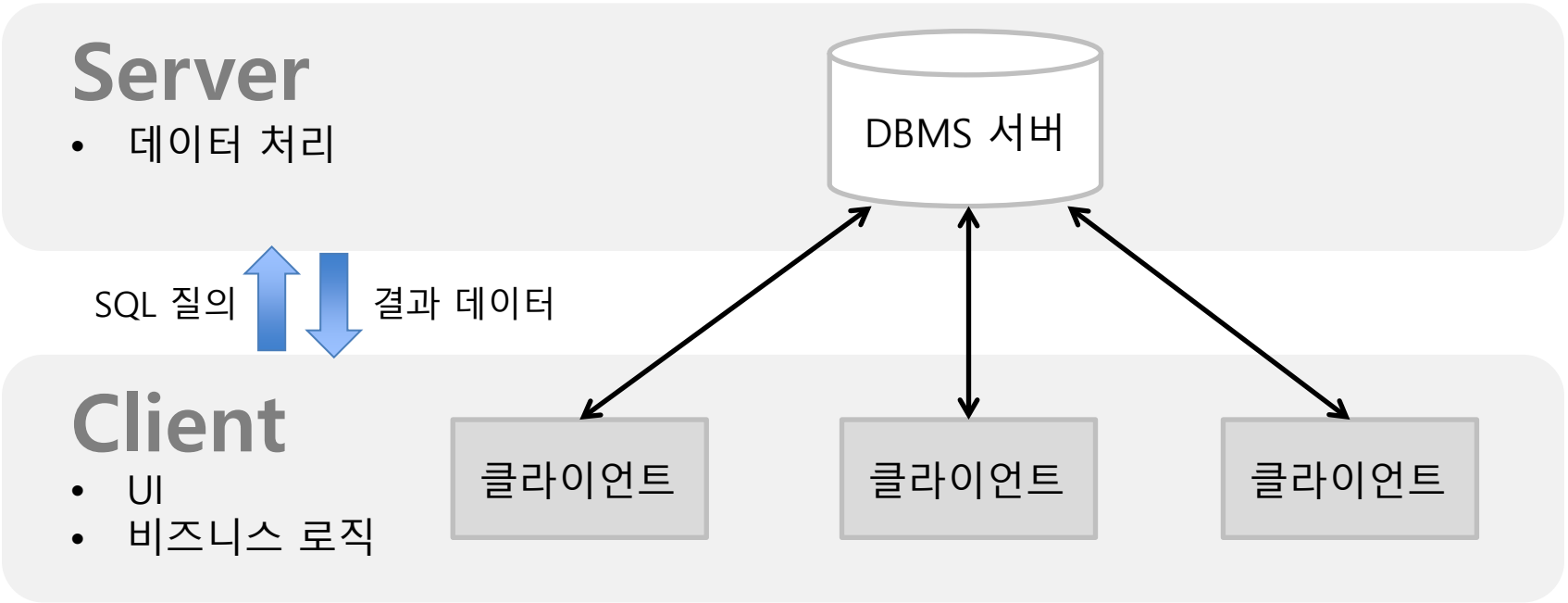


특징

- **서버에서 데이터 처리**

1.4 클라이언트·서버 아키텍처의 진화

전통적인 클라이언트·서버 구조

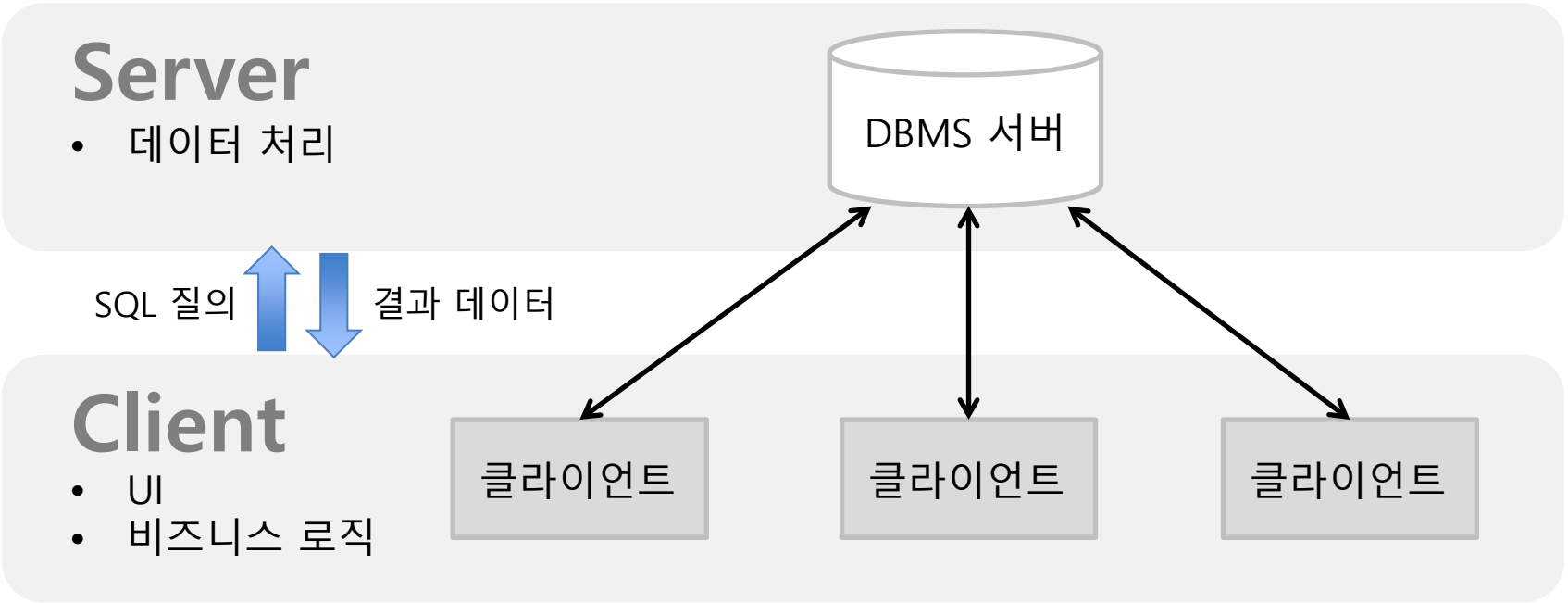


특징

- 서버에서 데이터 처리 ➔ **자료 중복 및 자료 불일치 문제 해결**

1.4 클라이언트·서버 아키텍처의 진화

전통적인 클라이언트·서버 구조



특징

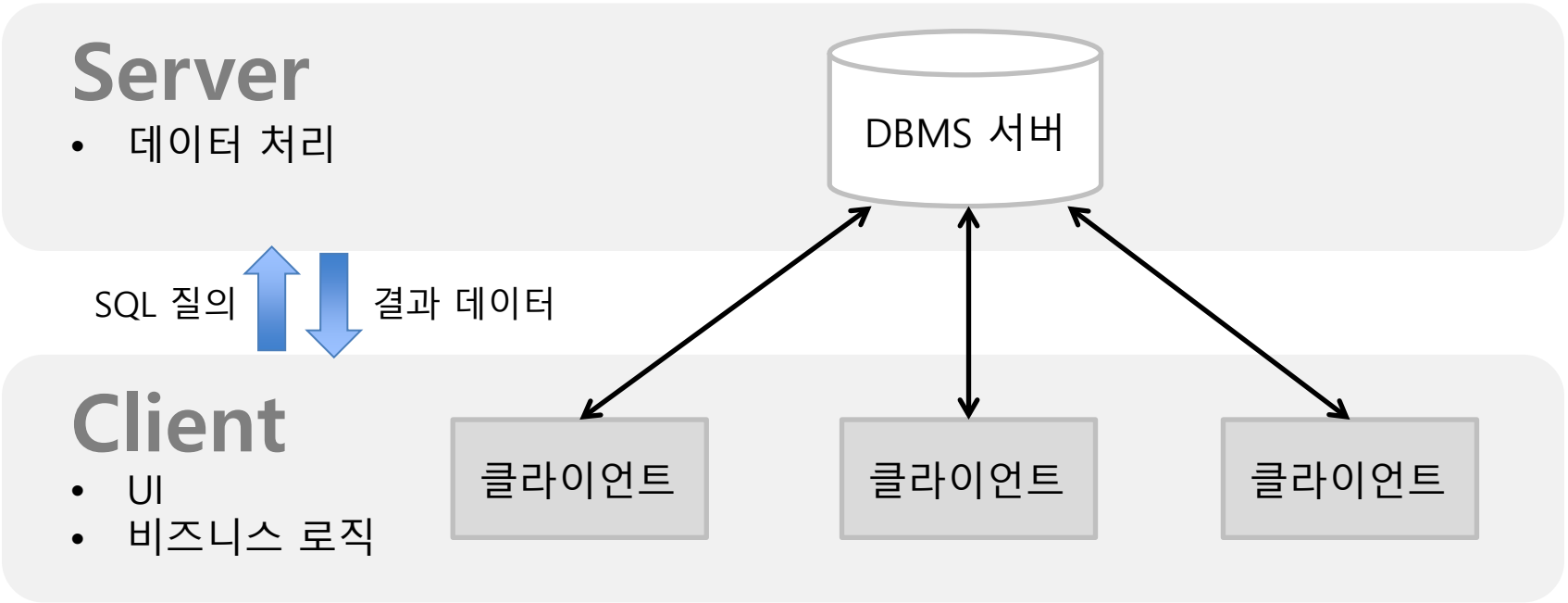
- 서버에서 데이터 처리 ➔ 자료 중복 및 자료 불일치 문제 해결

문제점

- **애플리케이션 변경 시 재배포 필요**

1.4 클라이언트·서버 아키텍처의 진화

전통적인 클라이언트·서버 구조



특징

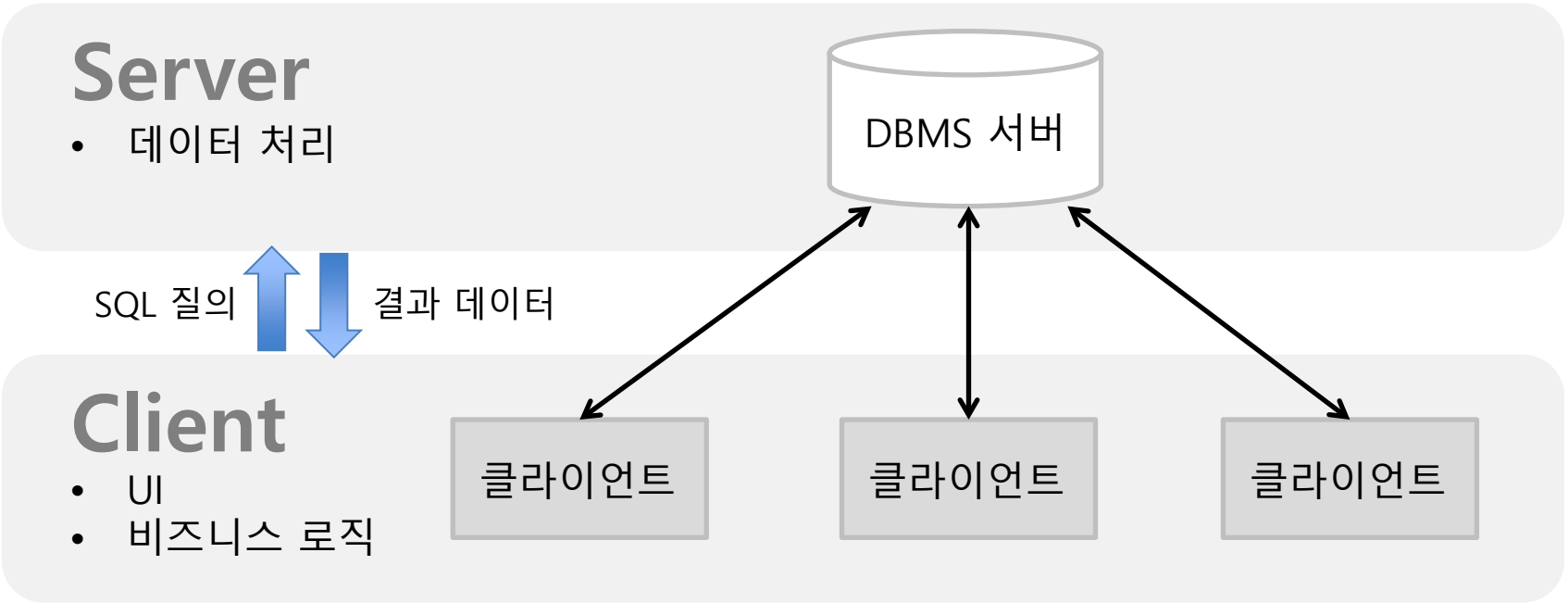
- 서버에서 데이터 처리 ➔ 자료 중복 및 자료 불일치 문제 해결

문제점

- 애플리케이션 변경 시 재배포 필요
- **클라이언트에서 DBMS에 직접 접속**

1.4 클라이언트·서버 아키텍처의 진화

전통적인 클라이언트·서버 구조



특징

- 서버에서 데이터 처리 ➔ 자료 중복 및 자료 불일치 문제 해결

문제점

- 애플리케이션 변경 시 재배포 필요
- 클라이언트에서 DBMS에 직접 접속 ➔ **접속 아이디, 암호 노출 위험**

1.4 클라이언트·서버 아키텍처의 진화

애플리케이션 서버 도입

Server
- 데이터 처리

DBMS 서버

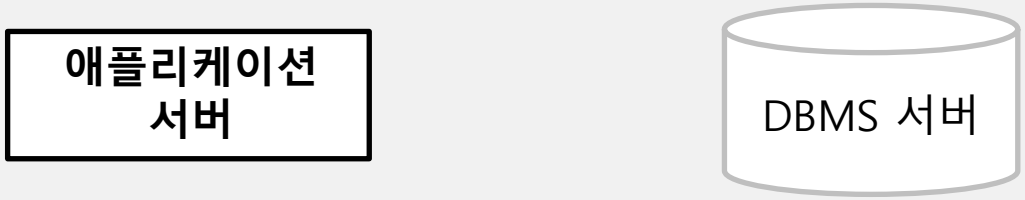
Client
- UI
- ~~비즈니스 로직~~

클라이언트 클라이언트 클라이언트

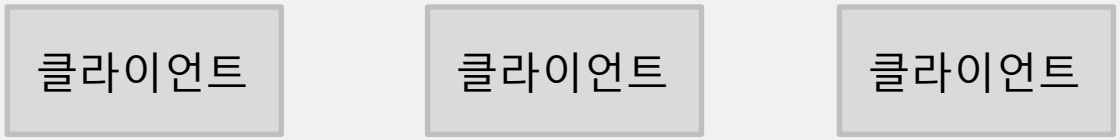
1.4 클라이언트·서버 아키텍처의 진화

애플리케이션 서버 도입

Server
- 데이터 처리
- **비즈니스 로직**

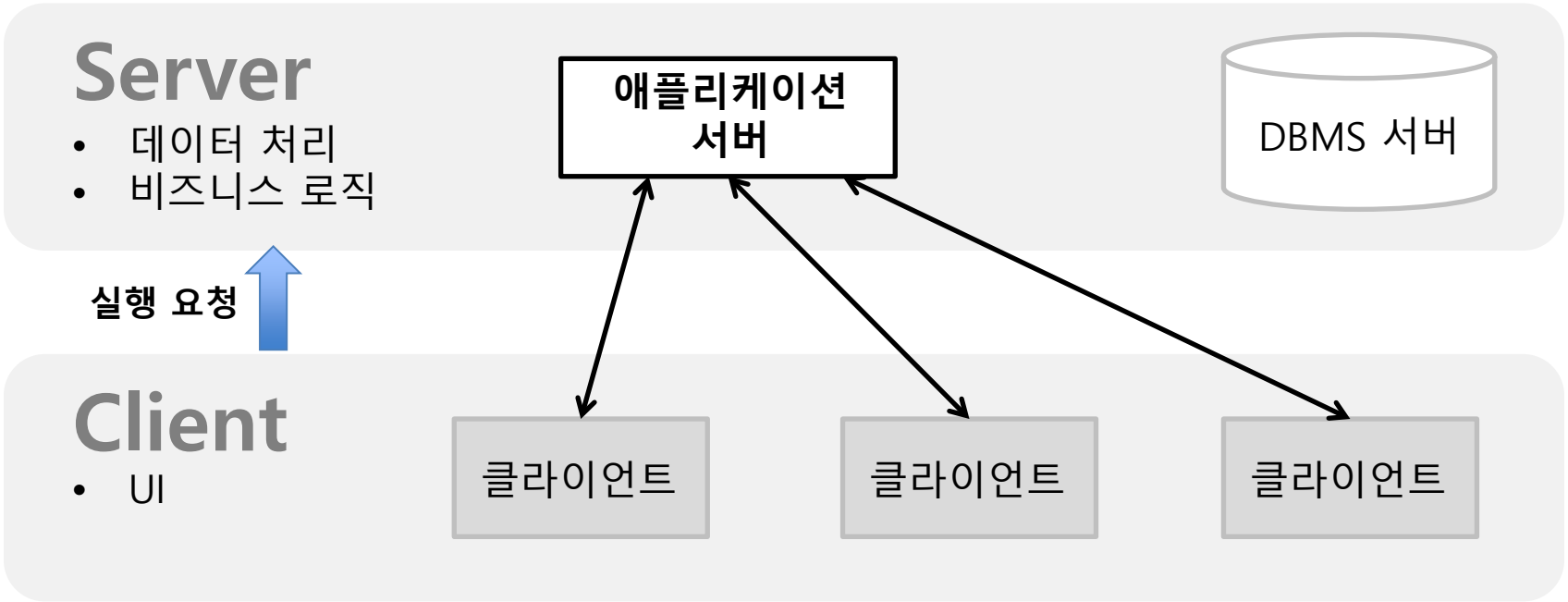


Client
- UI



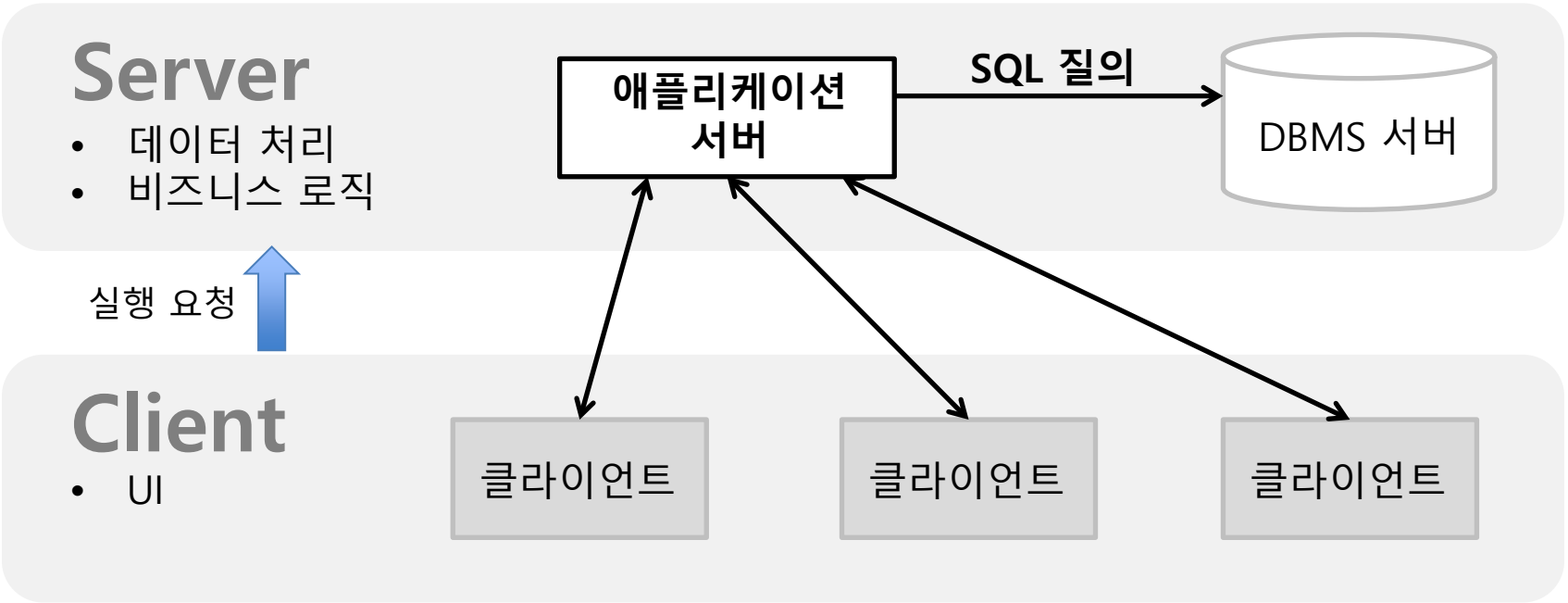
1.4 클라이언트·서버 아키텍처의 진화

애플리케이션 서버 도입



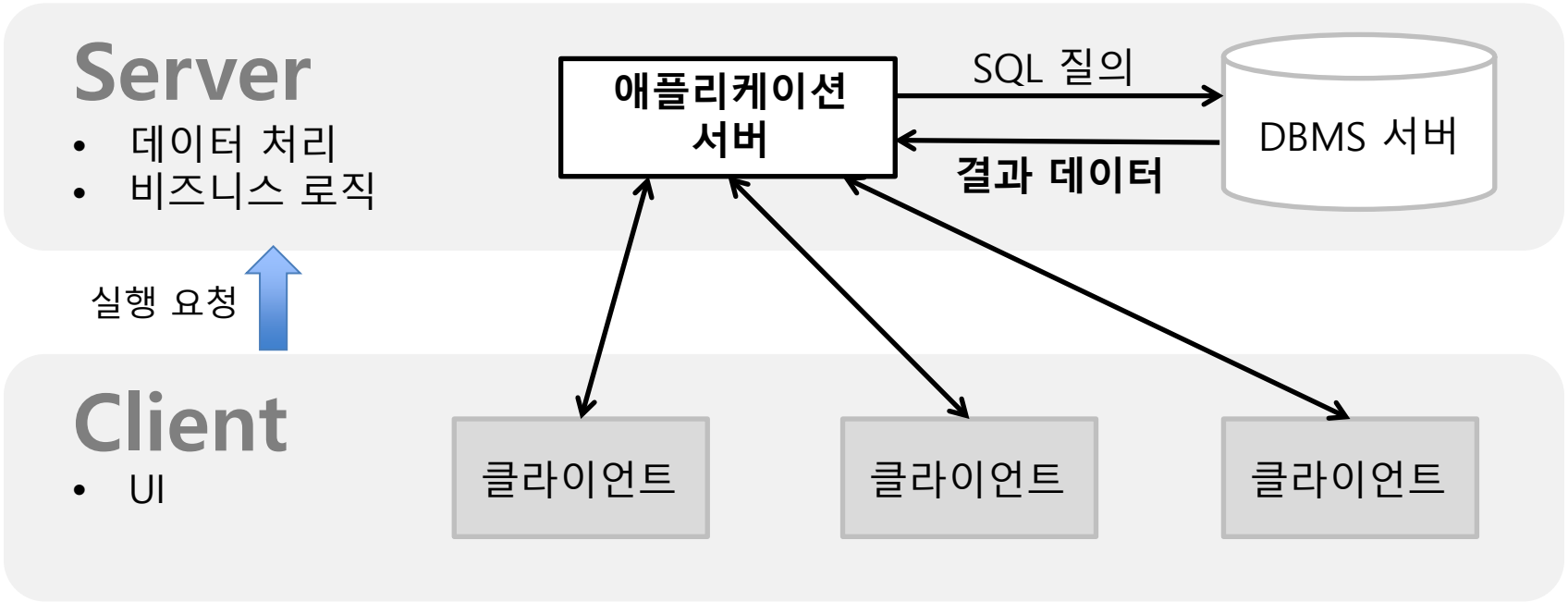
1.4 클라이언트·서버 아키텍처의 진화

애플리케이션 서버 도입



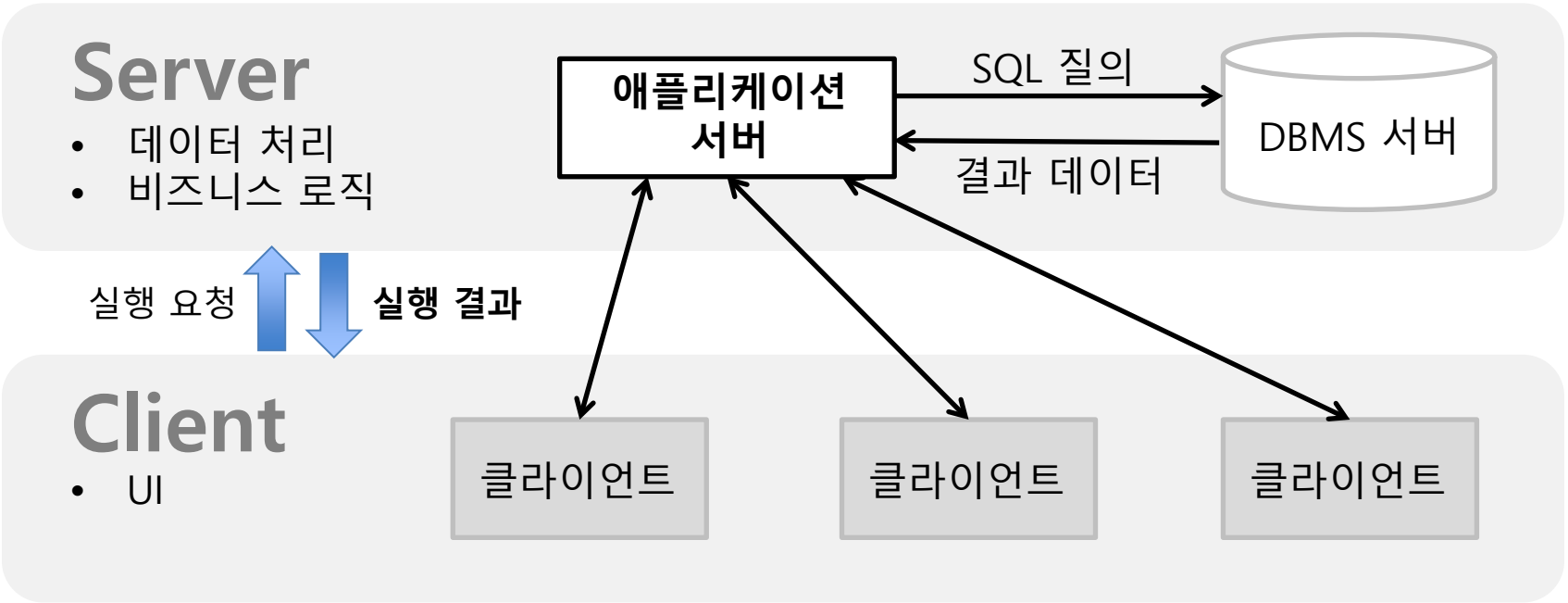
1.4 클라이언트·서버 아키텍처의 진화

애플리케이션 서버 도입



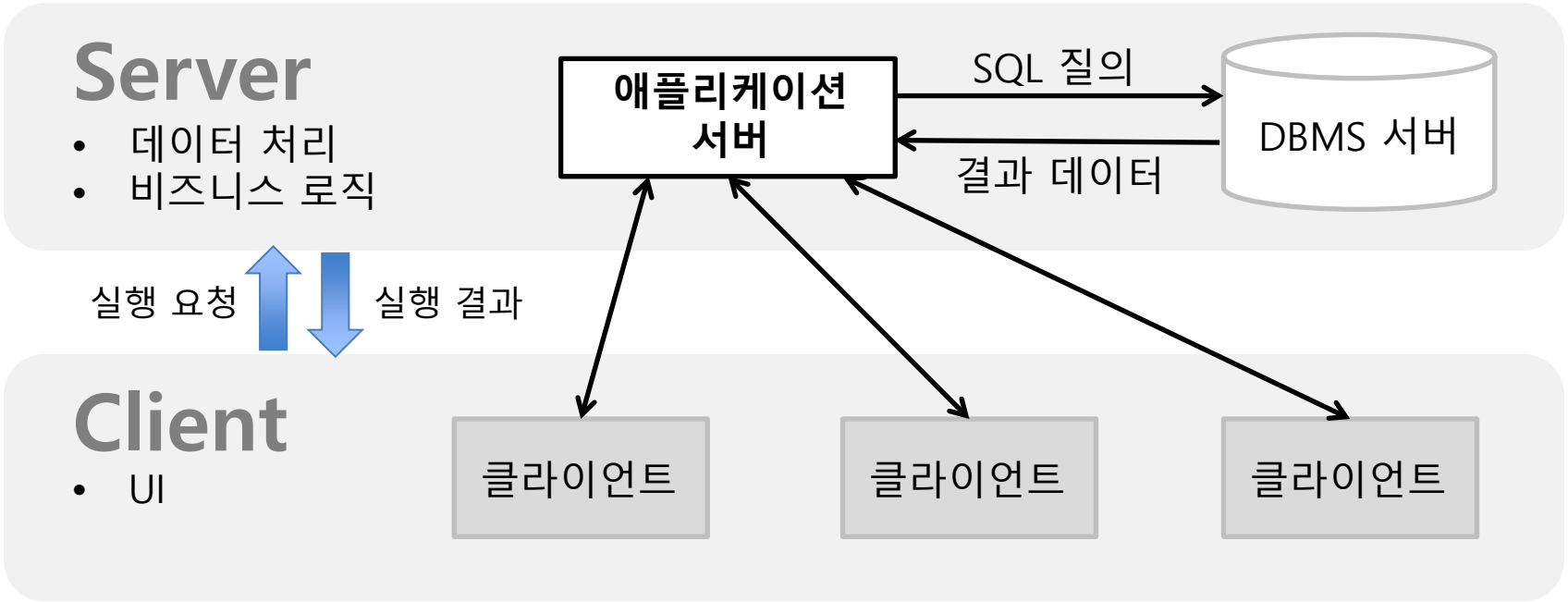
1.4 클라이언트·서버 아키텍처의 진화

애플리케이션 서버 도입



1.4 클라이언트·서버 아키텍처의 진화

애플리케이션 서버 도입

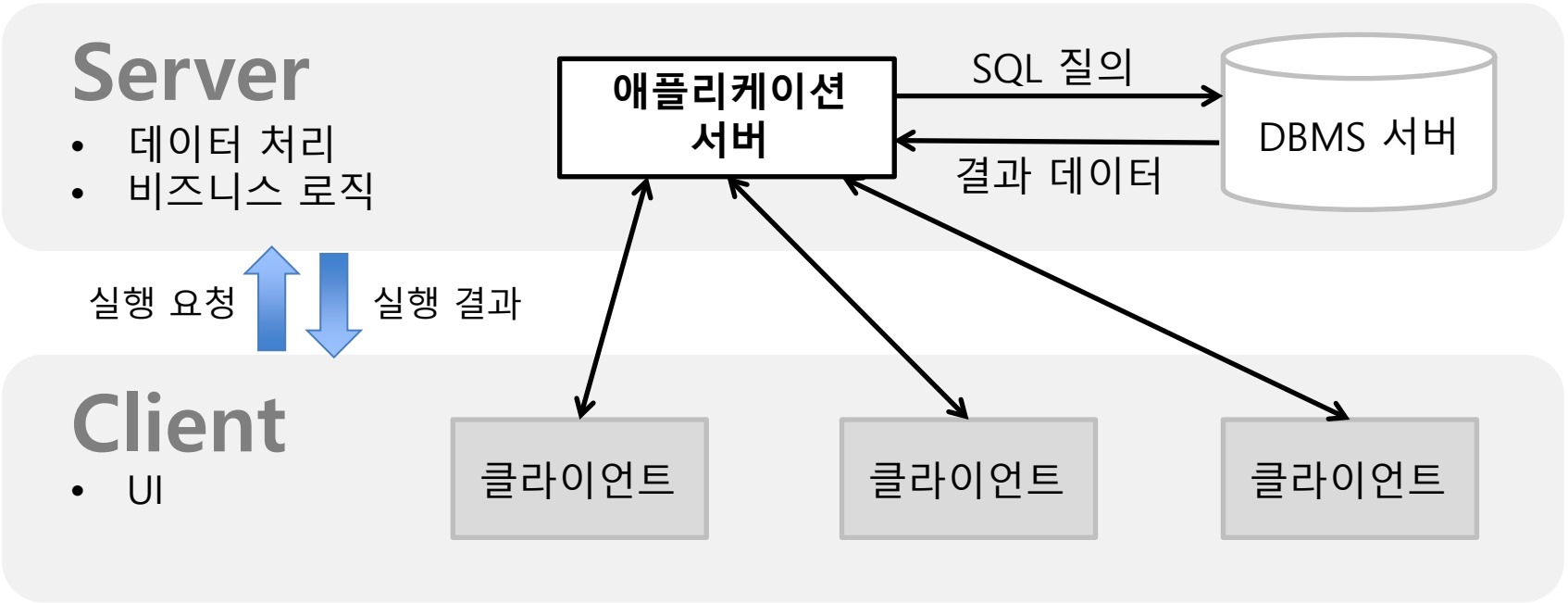


특징

- **비즈니스 로직을 전문으로 처리하는 서버를 둠**

1.4 클라이언트·서버 아키텍처의 진화

애플리케이션 서버 도입

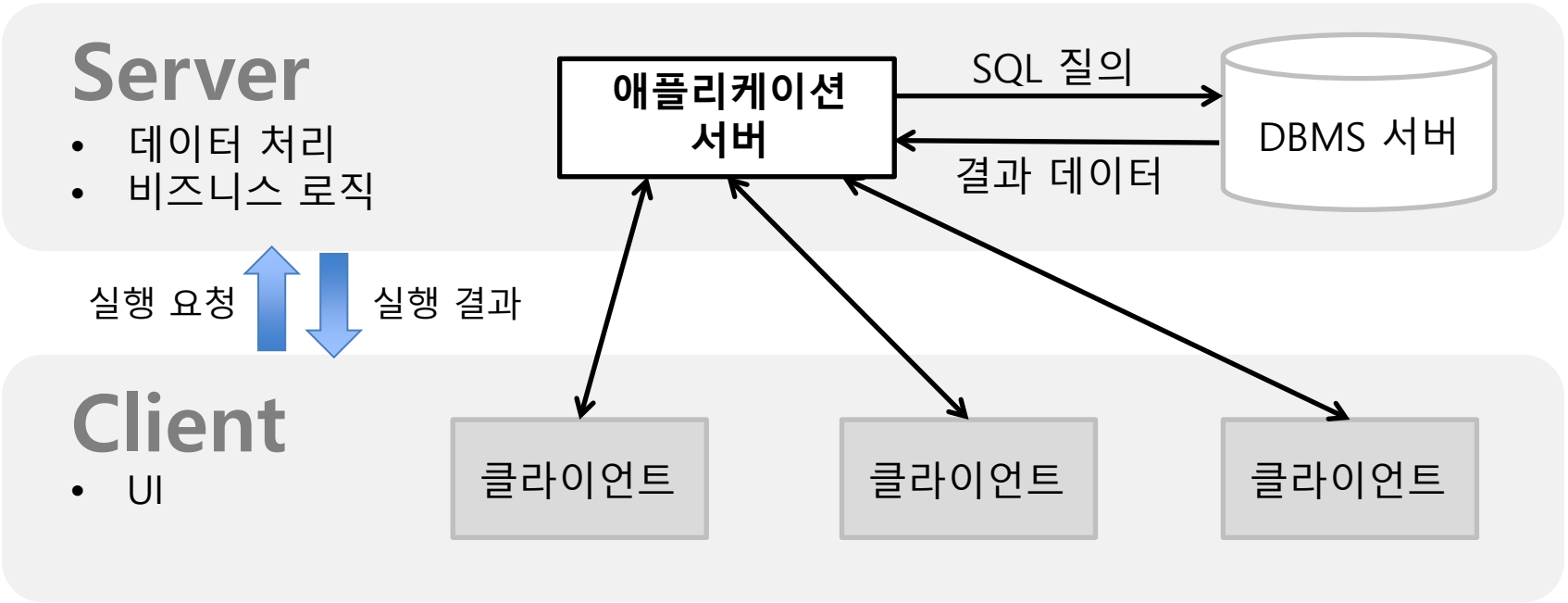


특징

- 비즈니스 로직을 전문으로 처리하는 서버를 둠 ➔ **애플리케이션 서버**

1.4 클라이언트·서버 아키텍처의 진화

애플리케이션 서버 도입

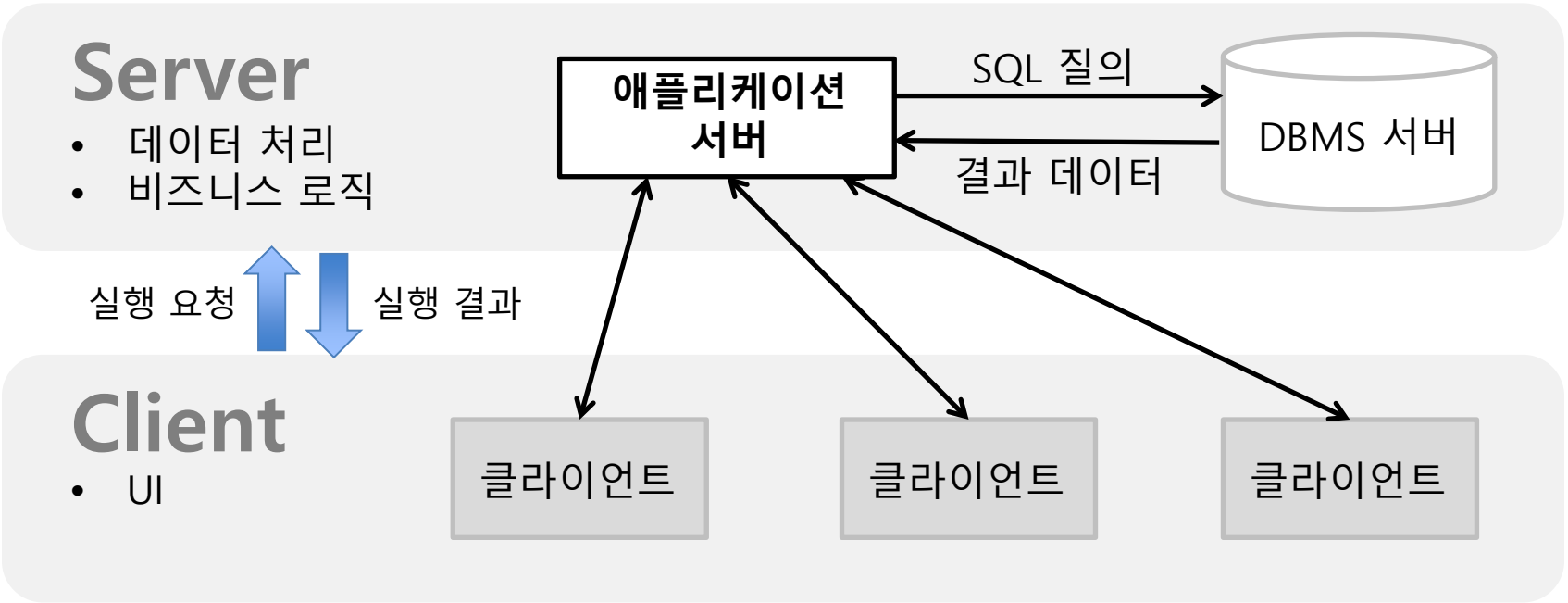


특징

- 비즈니스 로직을 전문으로 처리하는 서버를 둠 ➔ 애플리케이션 서버
- **클라이언트는 실행 결과 출력만을 담당**

1.4 클라이언트·서버 아키텍처의 진화

애플리케이션 서버 도입

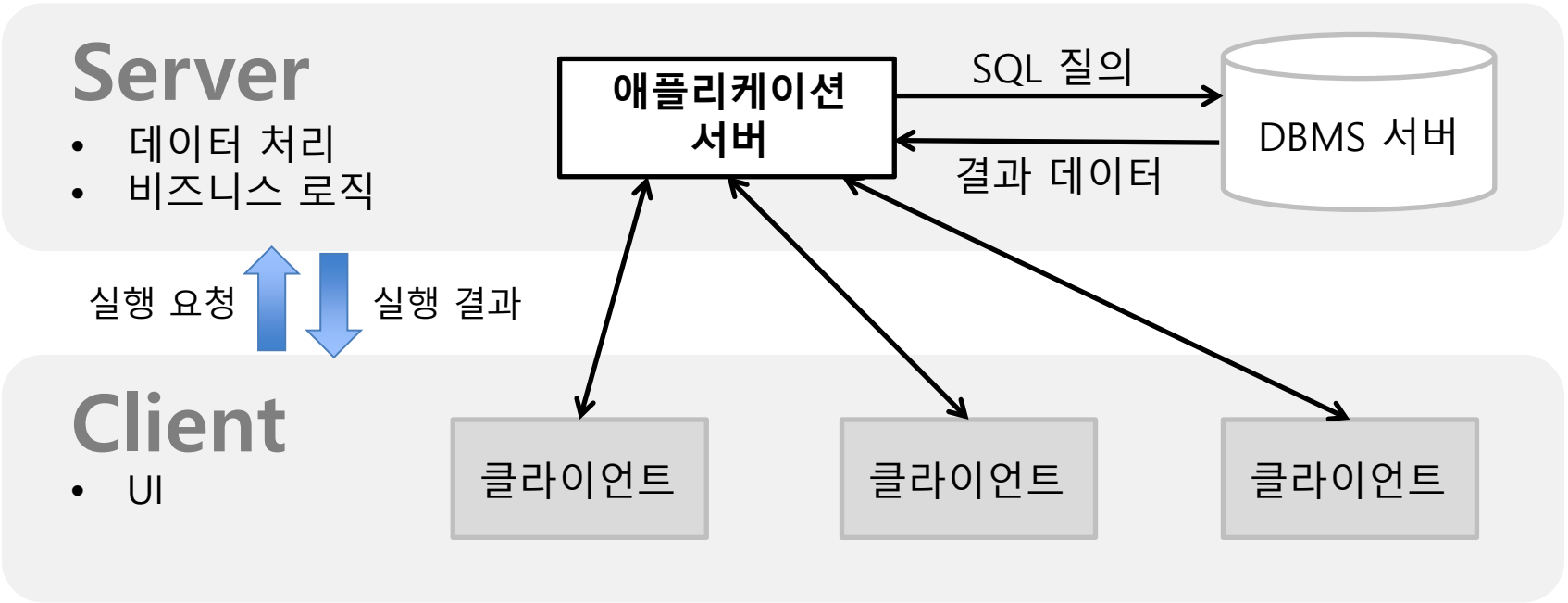


특징

- 비즈니스 로직을 전문으로 처리하는 서버를 둠 ➔ 애플리케이션 서버
- 클라이언트는 실행 결과 출력만을 담당 ➔ **씬 클라이언트(thin client) 가능**

1.4 클라이언트·서버 아키텍처의 진화

애플리케이션 서버 도입

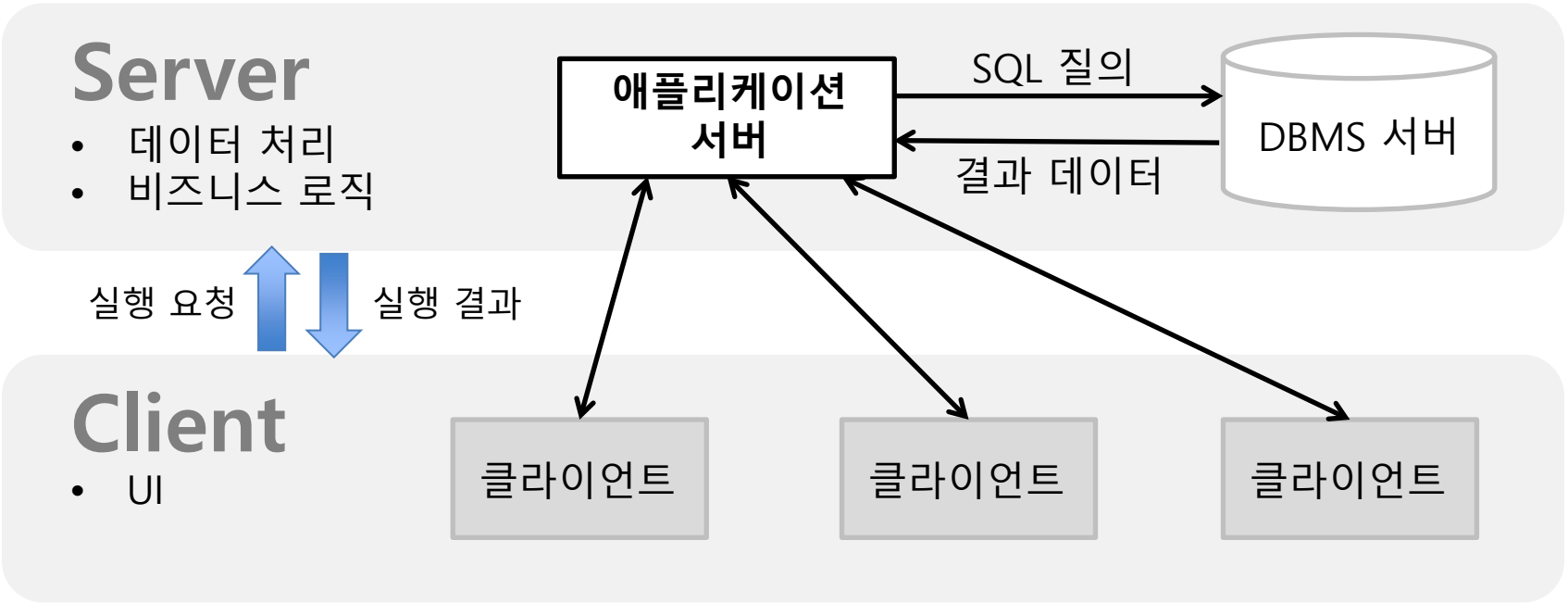


특징

- 비즈니스 로직을 전문으로 처리하는 서버를 둠 ➔ 애플리케이션 서버
- 클라이언트는 실행 결과 출력만을 담당 ➔ 씬 클라이언트(thin client) 가능
- **애플리케이션 서버에서 DBMS 접근**

1.4 클라이언트·서버 아키텍처의 진화

애플리케이션 서버 도입

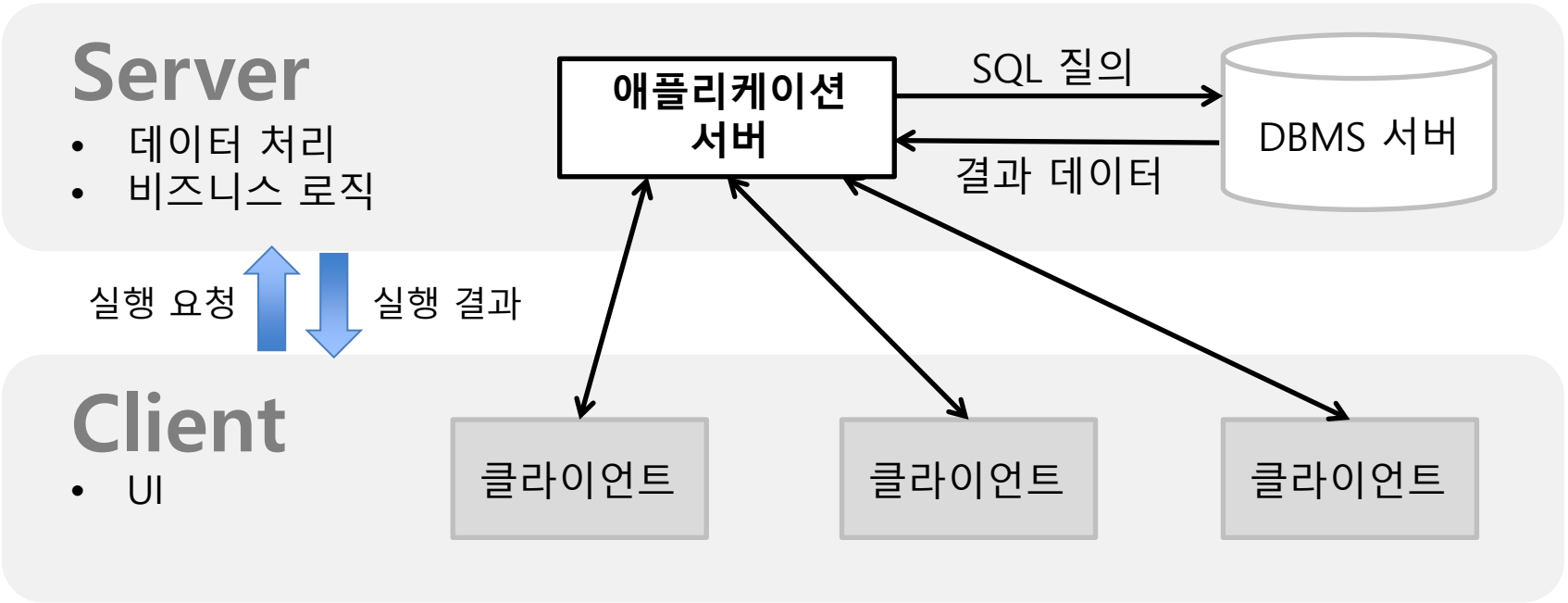


특징

- 비즈니스 로직을 전문으로 처리하는 서버를 둠 ➔ 애플리케이션 서버
- 클라이언트는 실행 결과 출력만을 담당 ➔ 씬 클라이언트(thin client) 가능
- 애플리케이션 서버에서 DBMS 접근 ➔ **보안 강화**

1.4 클라이언트·서버 아키텍처의 진화

애플리케이션 서버 도입

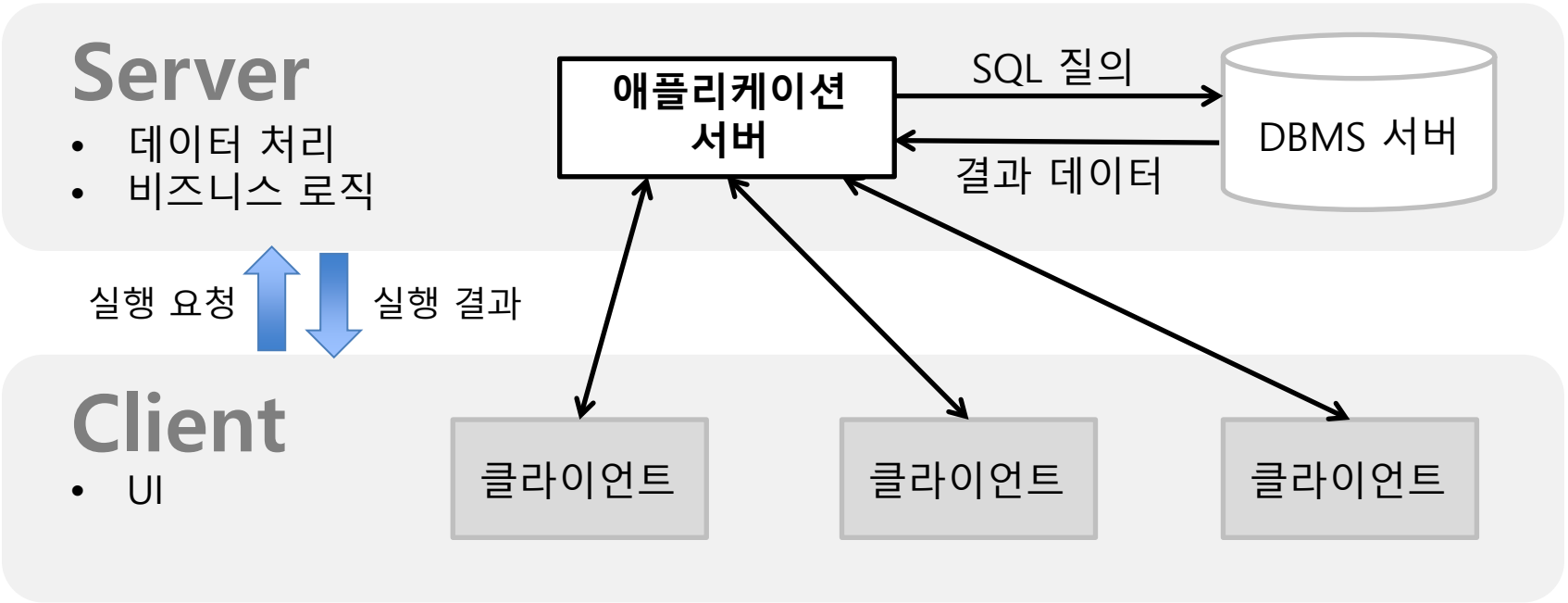


특징

- 비즈니스 로직을 전문으로 처리하는 서버를 둠 ➔ 애플리케이션 서버
- 클라이언트는 실행 결과 출력만을 담당 ➔ 씬 클라이언트(thin client) 가능
- 애플리케이션 서버에서 DBMS 접근 ➔ 보안 강화
- **기능 추가 또는 변경 시에 서버쪽만 변경**

1.4 클라이언트·서버 아키텍처의 진화

애플리케이션 서버 도입



특징

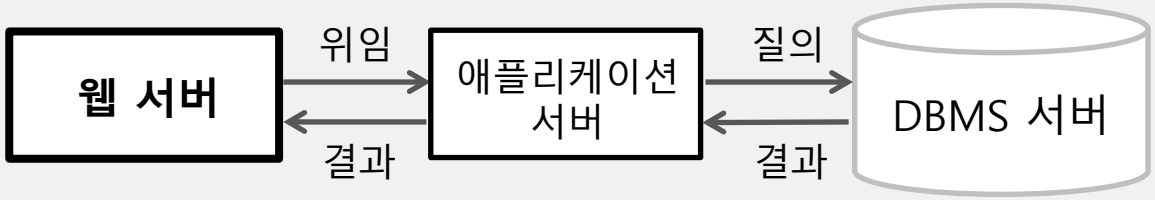
- 비즈니스 로직을 전문으로 처리하는 서버를 둠 ➔ 애플리케이션 서버
- 클라이언트는 실행 결과 출력만을 담당 ➔ 씬 클라이언트(thin client) 가능
- 애플리케이션 서버에서 DBMS 접근 ➔ 보안 강화
- 기능 추가 또는 변경 시에 서버쪽만 변경 ➔ **배포가 쉬움**

1.4 클라이언트·서버 아키텍처의 진화

웹 기술 적용

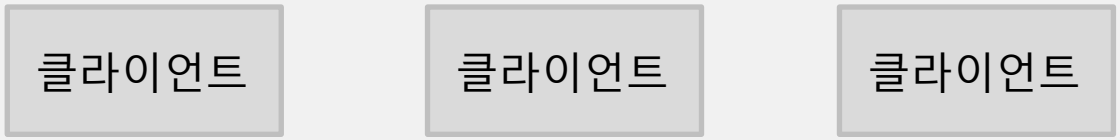
Server

- 데이터 처리
- 비즈니스 로직
- **UI 생성**



Client

- UI

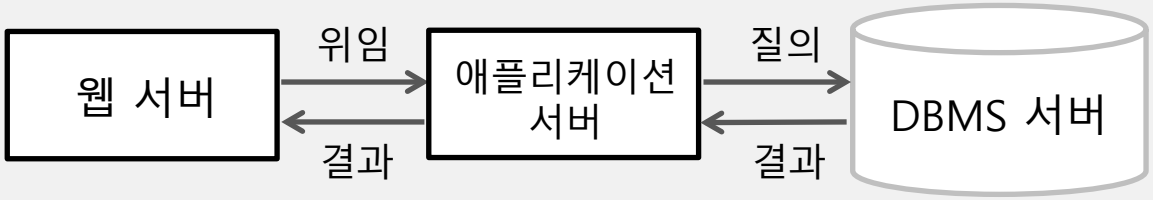


1.4 클라이언트·서버 아키텍처의 진화

웹 기술 적용

Server

- 데이터 처리
- 비즈니스 로직
- UI 생성



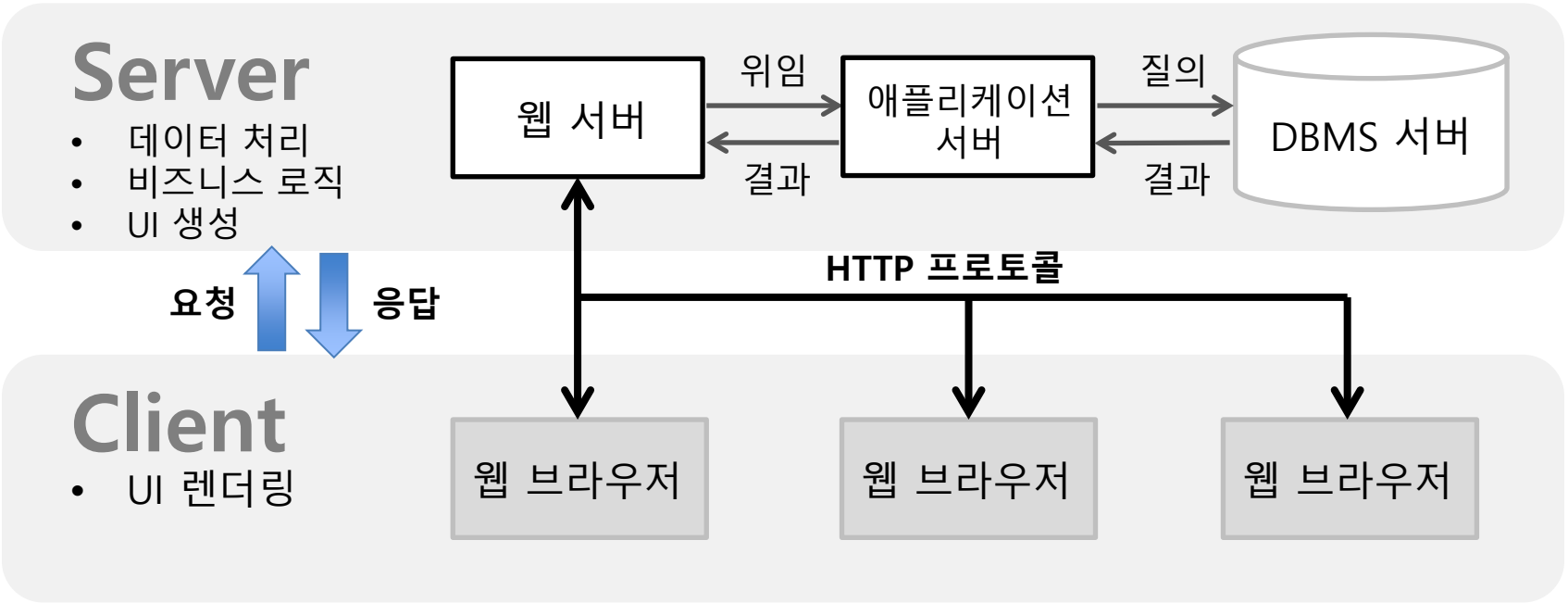
Client

- **UI 렌더링**

웹 브라우저 **웹 브라우저** **웹 브라우저**

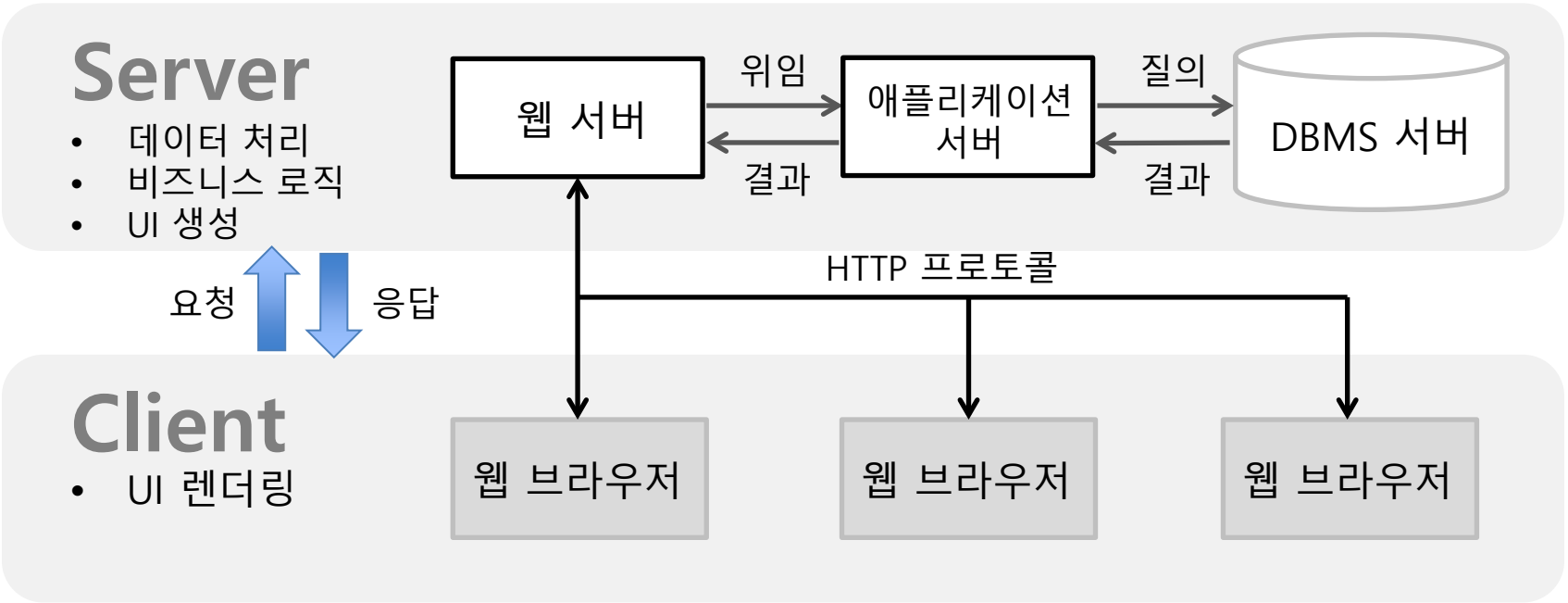
1.4 클라이언트·서버 아키텍처의 진화

웹 기술 적용



1.4 클라이언트·서버 아키텍처의 진화

웹 기술 적용

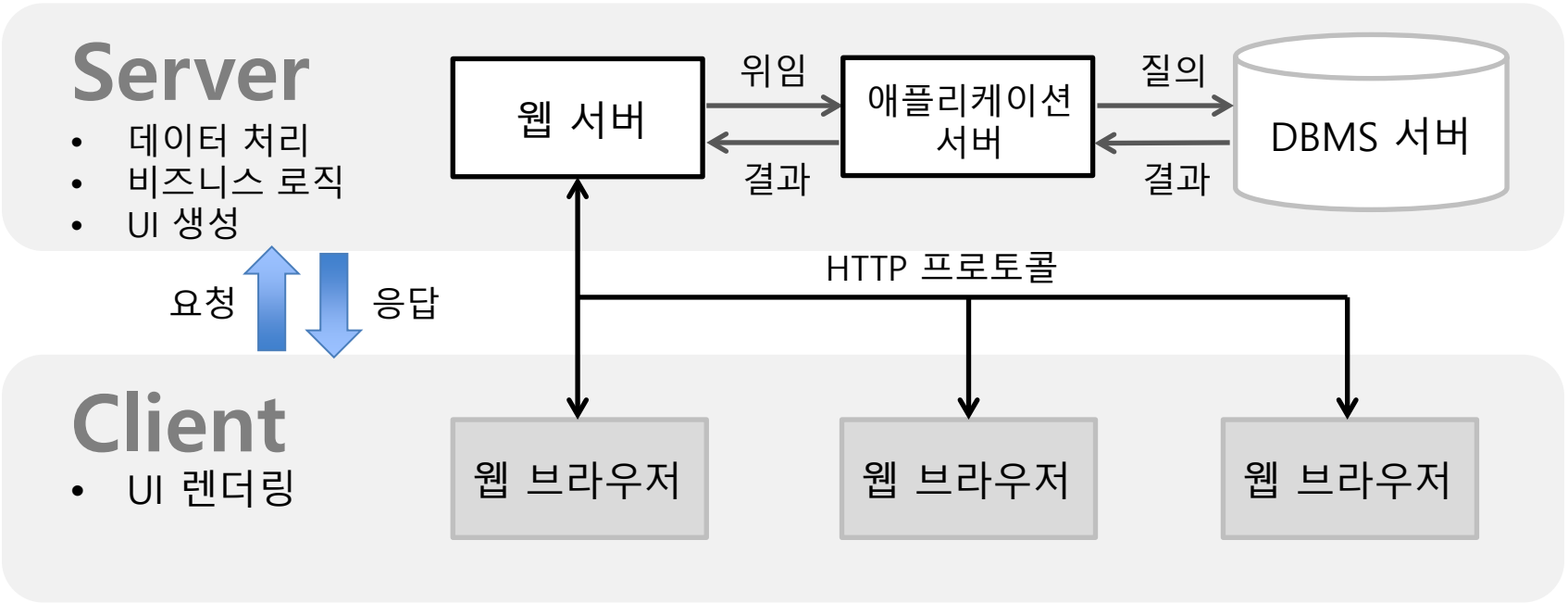


특징

- **웹 표준 기술을 활용하여 클라이언트와 서버 간에 통신**

1.4 클라이언트·서버 아키텍처의 진화

웹 기술 적용

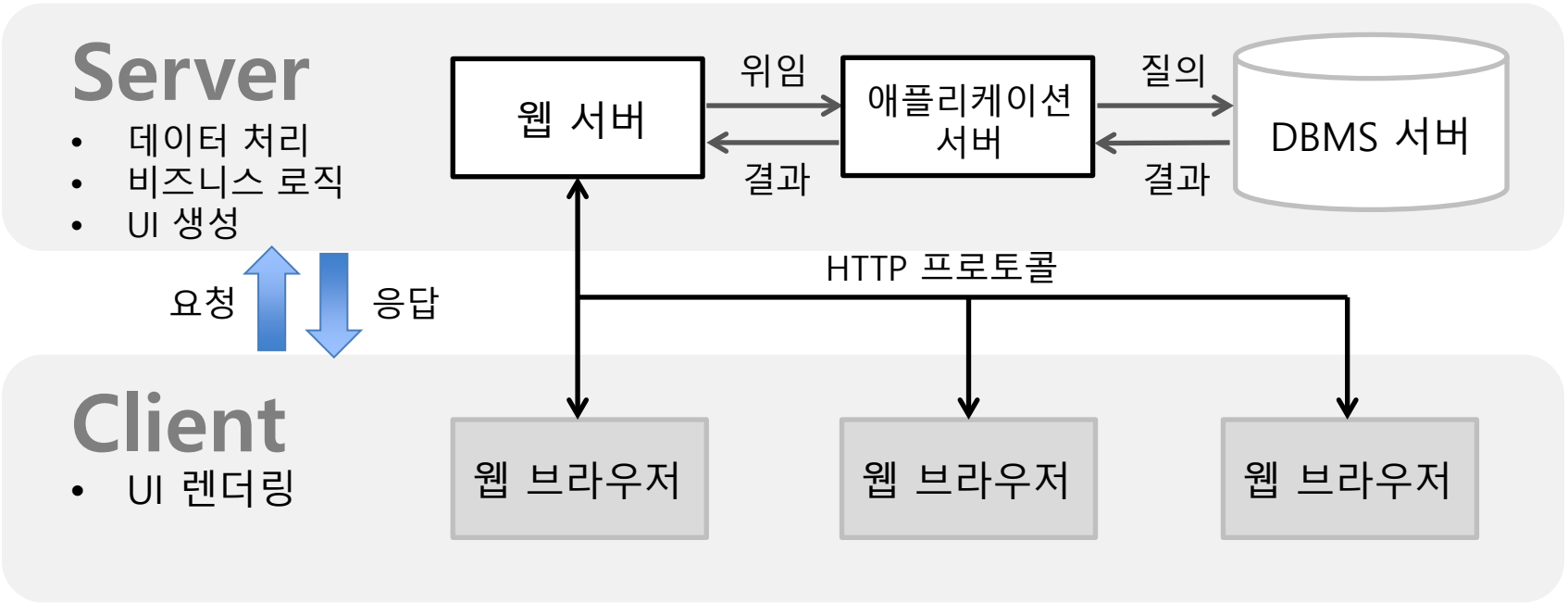


특징

- 웹 표준 기술을 활용하여 클라이언트와 서버 간에 통신
  ➔ **클라이언트 애플리케이션을 배포할 필요 없음**

1.4 클라이언트·서버 아키텍처의 진화

웹 기술 적용

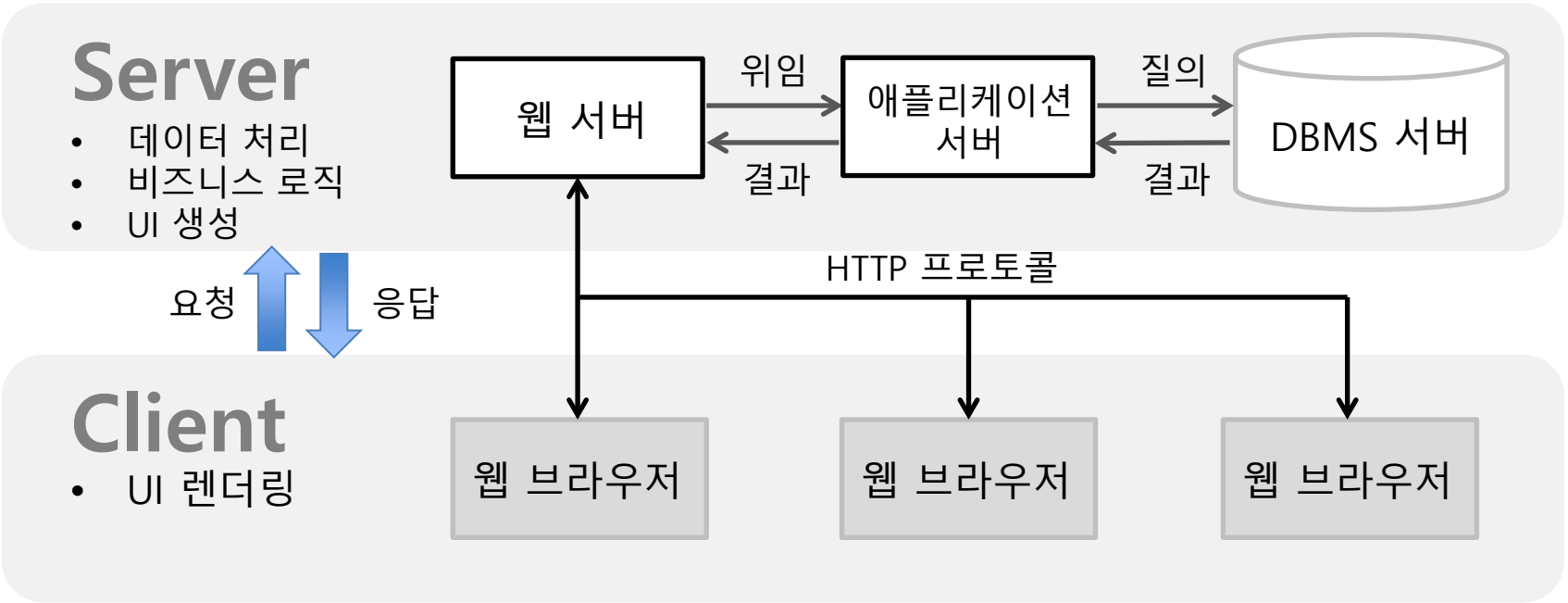


특징

- 웹 표준 기술을 활용하여 클라이언트와 서버 간에 통신
  ➔ 클라이언트 애플리케이션을 배포할 필요 없음
  ➔ **소켓 프로그래밍과 멀티 스레드 프로그래밍에서 탈출**

1.4 클라이언트·서버 아키텍처의 진화

웹 기술 적용

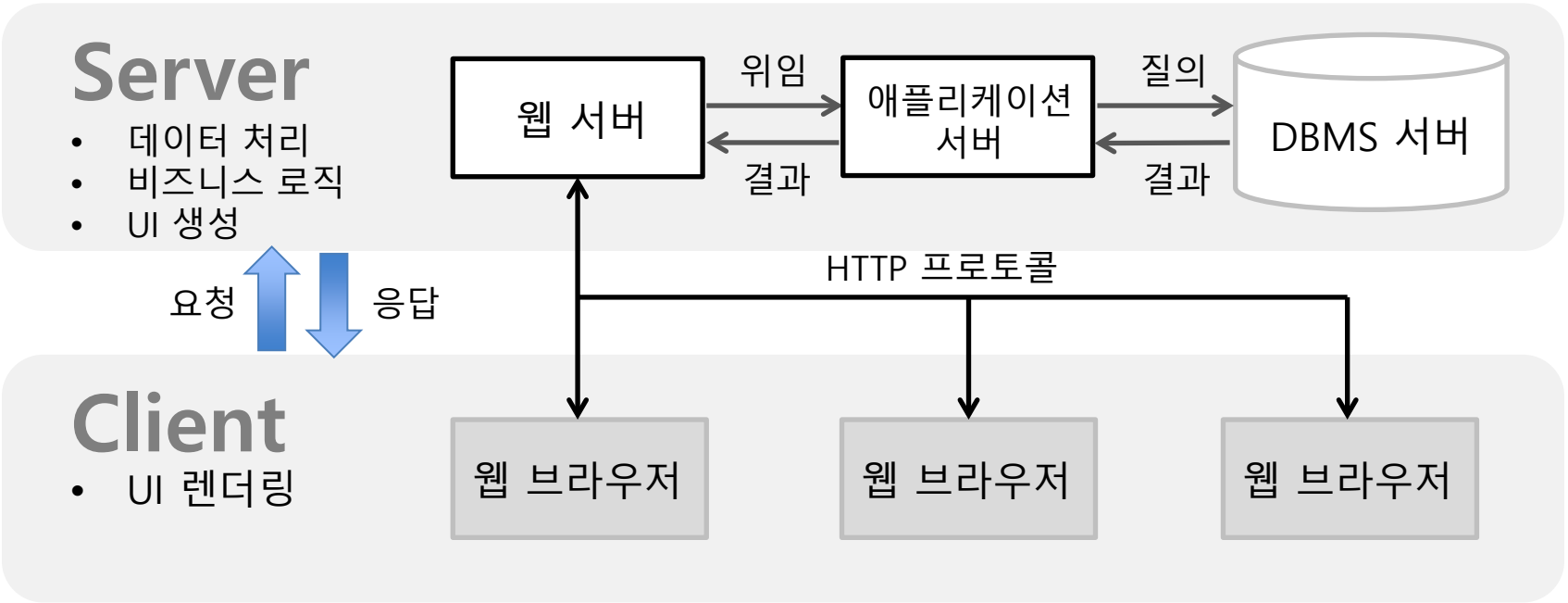


특징

- 웹 표준 기술을 활용하여 클라이언트와 서버 간에 통신
  ➔ 클라이언트 애플리케이션을 배포할 필요 없음
  ➔ 소켓 프로그래밍과 멀티 스레드 프로그래밍에서 탈출
  ➔ **이기종 시스템 간에 매끈한 연결 지원**

1.4 클라이언트·서버 아키텍처의 진화

웹 기술 적용

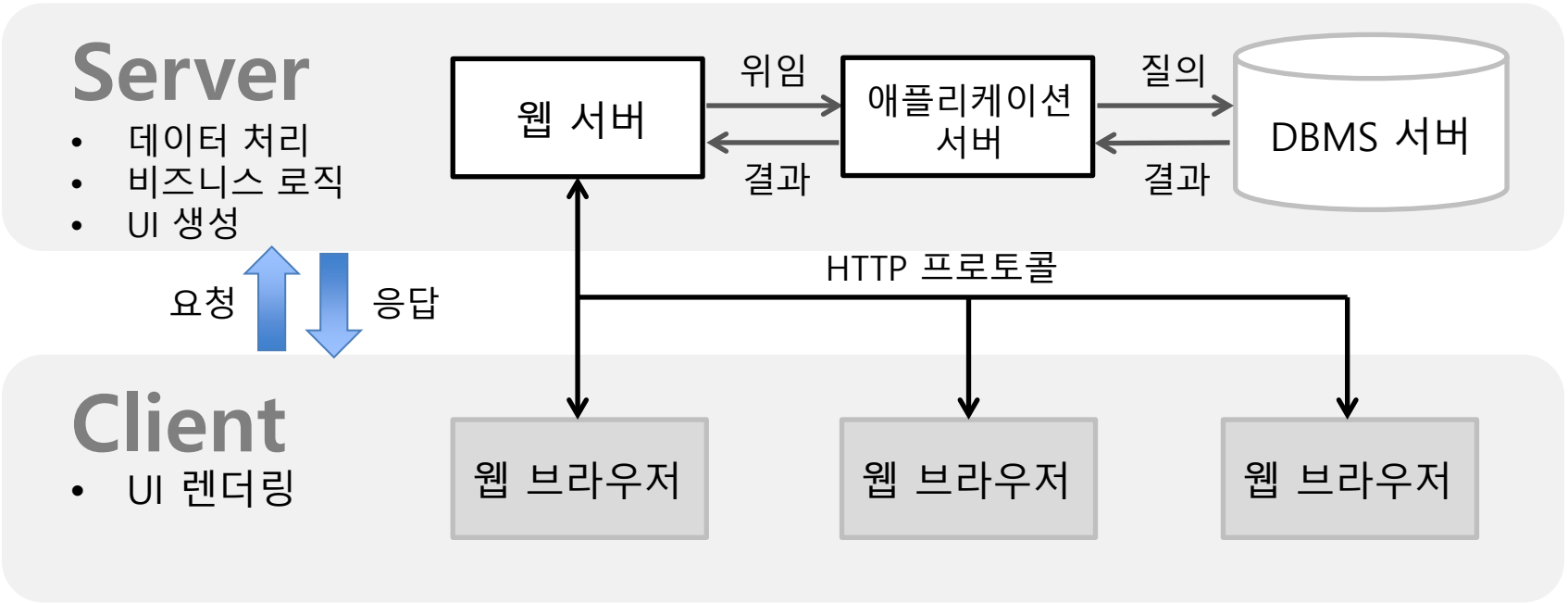


특징

- 웹 표준 기술을 활용하여 클라이언트와 서버 간에 통신
  ➔ 클라이언트 애플리케이션을 배포할 필요 없음
  ➔ 소켓 프로그래밍과 멀티 스레드 프로그래밍에서 탈출
  ➔ 이기종 시스템 간에 매끈한 연결 지원
- **서버쪽에서 UI 생성**

1.4 클라이언트·서버 아키텍처의 진화

웹 기술 적용



특징

- 웹 표준 기술을 활용하여 클라이언트와 서버 간에 통신
  ➔ 클라이언트 애플리케이션을 배포할 필요 없음
  ➔ 소켓 프로그래밍과 멀티 스레드 프로그래밍에서 탈출
  ➔ 이기종 시스템 간에 매끈한 연결 지원
- 서버쪽에서 UI 생성 ➔ **씬 클라이언트 구축 용이**