

# 네트워크 기초

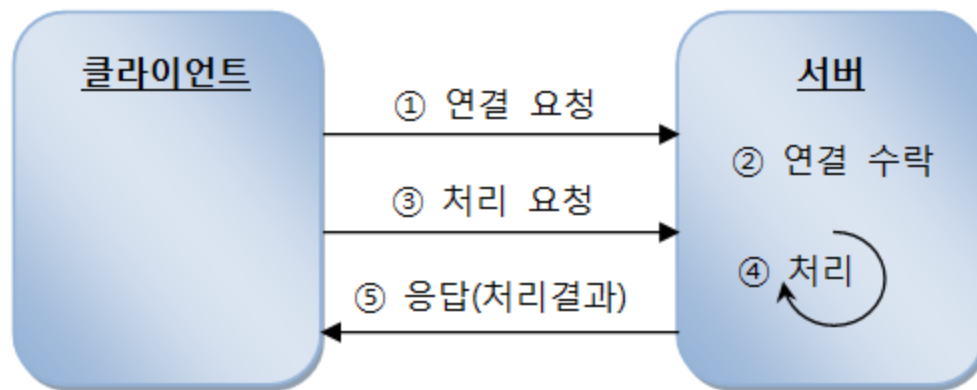
## ❖ 네트워크

- 여러 대의 컴퓨터를 통신 회선으로 연결한 것
  - 홈 네트워크: 컴퓨터가 방마다 있고, 이들 컴퓨터를 유·무선 등의 통신 회선으로 연결
  - 지역 네트워크: 회사, 건물, 특정 영역에 존재하는 컴퓨터를 통신 회선으로 연결한 것
  - 인터넷: 지역 네트워크를 통신 회선으로 연결한 것

# 네트워크 기초

## ❖ 서버와 클라이언트

- 서버: 서비스를 제공하는 프로그램
  - 웹 서버, **FTP**서버, **DBMS**, 메신저 서버
  - 클라이언트의 연결을 수락하고, 요청 내용 처리한 후 응답 보내는 역할
- 클라이언트: 서비스를 받는 프로그램
  - 웹 브라우저, **FTP** 클라이언트, 메신저
  - 네트워크 데이터를 필요로 하는 모든 애플리케이션이 해당(모바일 앱 포함)



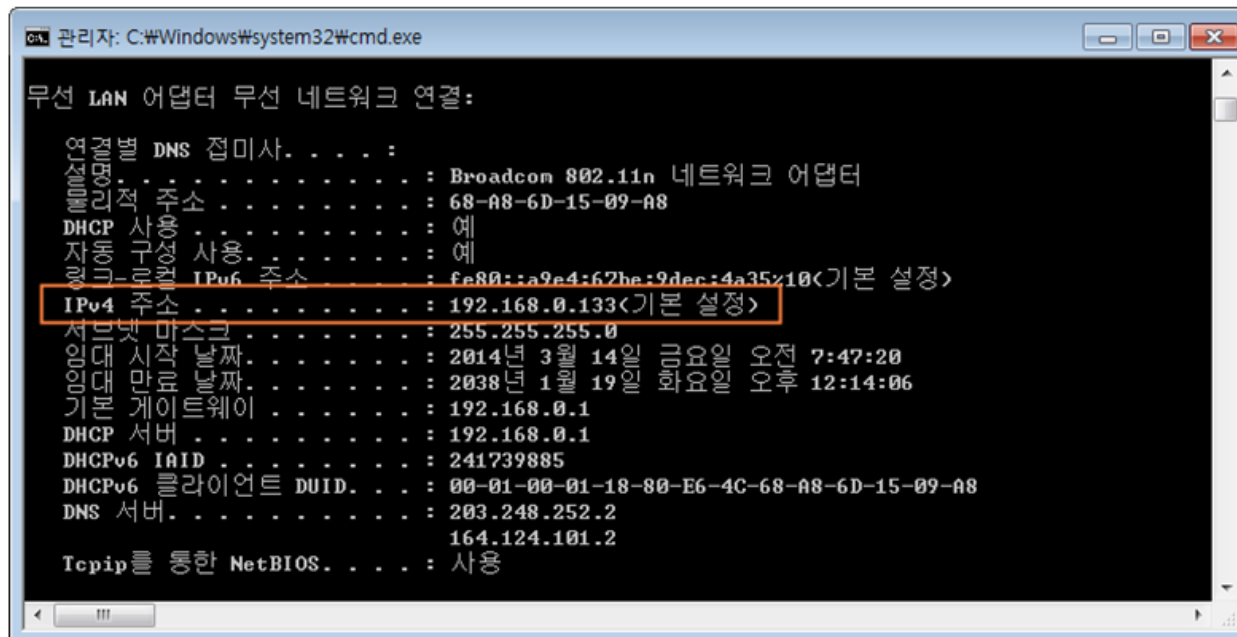
# 네트워크 기초

## ❖ IP 주소와 포트(port)

### ■ IP(Internet Protocol) 주소

- 네트워크상에서 컴퓨터를 식별하는 번호
- 네트워크 어댑터(랜 (Lan) 카드) 마다 할당
- IP 주소 확인 법 - 명령 프롬프트 (cmd.exe) 사용
- **xxx.xxx.xxx.xxx** 형식으로 표현 (**xxx**는 **0~255** 사이의 정수)

```
C:\W>ipconfig /all
```



```
관리자: C:\Windows\system32\cmd.exe

무선 LAN 어댑터 무선 네트워크 연결:

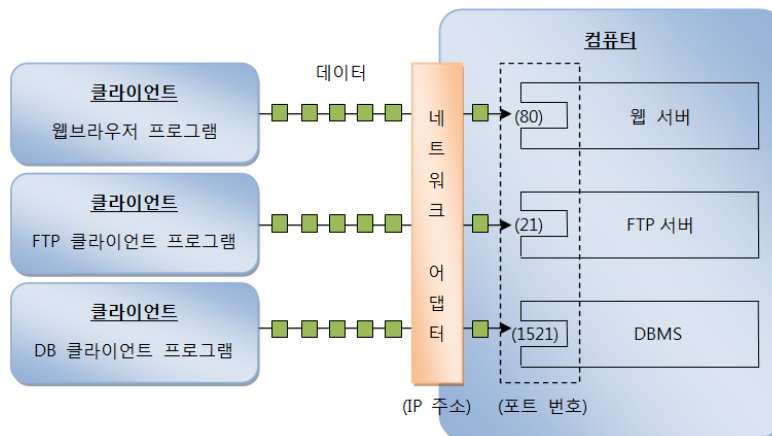
연결별 DNS 접미사. . . . . :
설명. . . . . : Broadcom 802.11n 네트워크 어댑터
물리적 주소. . . . . : 68-A8-6D-15-09-A8
DHCP 사용. . . . . : 예
자동 구성 사용. . . . . : 예
링크-로컬 IPv6 주소. . . . . : fe80::a9e4:67be:9dec:4a35%10<기본 설정>
IPv4 주소. . . . . : 192.168.0.133<기본 설정>
서브넷 마스크. . . . . : 255.255.255.0
임대 시작 날짜. . . . . : 2014년 3월 14일 금요일 오전 7:47:20
임대 만료 날짜. . . . . : 2038년 1월 19일 화요일 오후 12:14:06
기본 게이트웨이. . . . . : 192.168.0.1
DHCP 서버. . . . . : 192.168.0.1
DHCPv6 IAID. . . . . : 241739885
DHCPv6 클라이언트 DUID. . . : 00-01-00-01-18-80-E6-4C-68-A8-6D-15-09-A8
DNS 서버. . . . . : 203.248.252.2
                  164.124.101.2
Tcpip를 통한 NetBIOS. . . . : 사용
```

# 네트워크 기초

## ■ 포트(Port)

- 같은 컴퓨터 내에서 프로그램을 식별하는 번호
- 클라이언트는 서버 연결 요청 시 **IP** 주소와 **Port** 같이 제공
- **0~65535** 범위의 값을 가짐
- 포트 범위는 세 가지로 구분

구분명	범위	설명
Well Know Port Numbers	0~1023	국제인터넷주소관리기구(ICANN)가 특정 애플리케이션용으로 미리 예약한 포트
Registered Port Numbers	1024~49151	회사에서 등록해서 사용할 수 있는 포트
Dynamic Or Private Port Numbers	49152~65535	운영체제가 부여하는 동적 포트 또는 개인적인 목적으로 사용할 수 있는 포트



# 네트워크 기초

## ❖ InetAddress로 IP 주소 얻기

### ■ java.net.InetAddress

- IP 주소 표현한 클래스
- 로컬 컴퓨터의 IP 주소
- 도메인 이름을 **DNS**에서 검색한 후 IP 주소를 가져오는 기능 제공

# TCP 네트워킹

## ❖ TCP(Transmission Control Protocol)

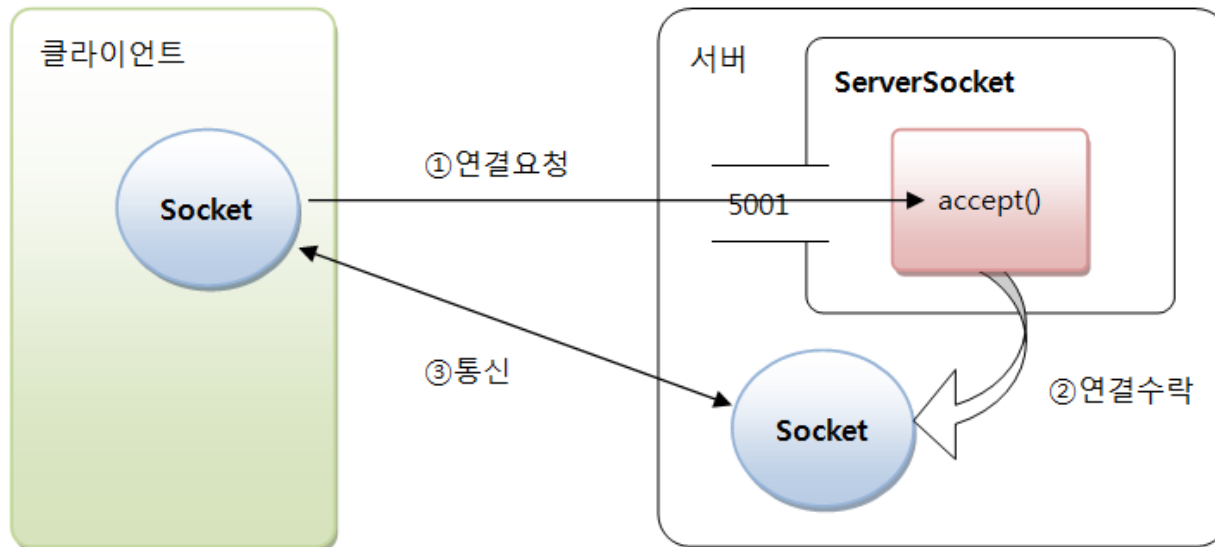
### ■ 특징

- 연결 지향적 프로토콜 -> 시간 소요
- 통신 선로 고정 -> 전송 속도 느려질 수 있음
- 데이터를 정확하고 안정적으로 전달

### ■ java.net API

- **ServerSocket, Socket**

### ■ ServerSocket과 Socket 용도



# TCP 네트워킹

## ❖ **ServerSocket** 생성과 연결 수락

### ■ **ServerSocket** 생성과 포트 바인딩

- 생성자에 바인딩 포트 대입하고 객체 생성

### ■ 연결 수락

- **accept()** 메소드는 클라이언트가 연결 요청 전까지 블로킹 → 대기
- 연결된 클라이언트 **IP** 주소 얻기

```
InetSocketAddress socketAddress = (InetSocketAddress) socket.getRemoteSocketAddress();
```

리터타입	메소드명(매개변수)	설명
String	getHostName()	클라이언트 IP 리턴
int	getPort()	클라이언트 포트 번호 리턴
String	toString()	"IP:포트번호" 형태의 문자열 리턴

### ■ **ServerSocket** 포트 언바인딩

- 더 이상 클라이언트 연결 수락 필요 없는 경우

# TCP 네트워킹

## ❖ Socket 생성과 연결 요청

### ■ Socket 생성 및 연결 요청

- **java.net.Socket** 이용
- 서버의 **IP** 주소와 바인딩 포트 번호를 제공하면 생성과 동시에 사용가능

### ■ 연결 끊기

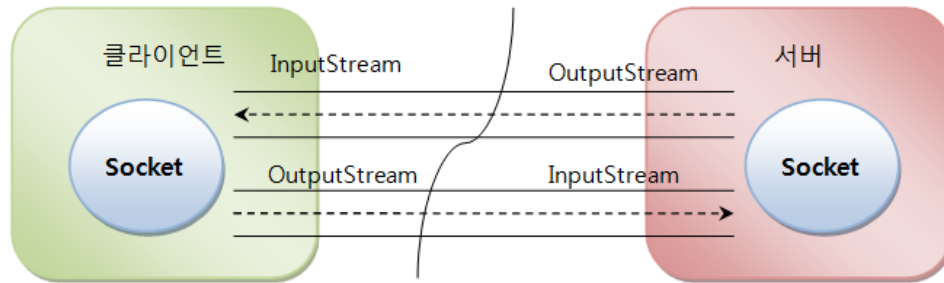
- **Exception** 처리 필요



# TCP 네트워킹

## ❖ Socket 데이터 통신

### ■ Socket 객체로 부터 입출력 스트림 얻기



- 입출력 스트림 구현 예제
  - 연결 성공 후 클라이언트가 서버에 **“Hello Server”**
  - 서버가 데이터 받음
  - 서버가 클라이언트에 **“Hello Client”** 보냄
  - 클라이언트가 데이터 받음
- **read()**의 블로킹 해제

블로킹이 해제되는 경우	리턴값
상대방이 데이터를 보냄	읽은 바이트 수
상대방이 정상적으로 Socket 의 close()를 호출	-1
상대방이 비정상적으로 종료	IOException 발생

# TCP 네트워킹

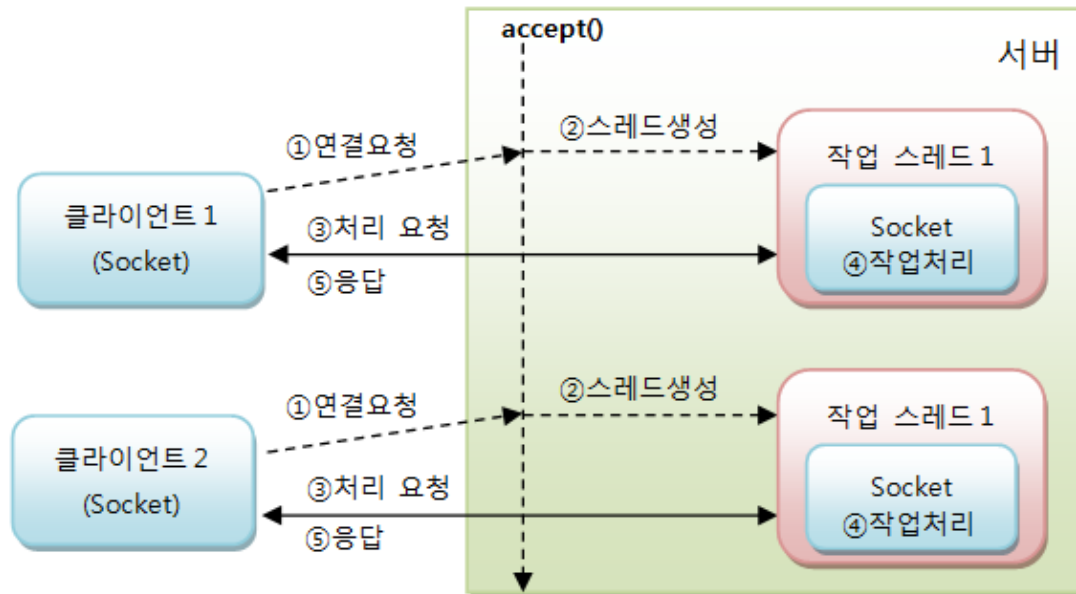
## ❖ 스레드 병렬 처리

- 블로킹(대기 상태)가 되는 메소드
  - **ServerSocket**의 **accept()**
  - **Socket** 생성자 또는 **connect()**
  - **Socket**의 **read()**, **write()**
- 병렬 처리의 필요성
  - 스레드가 블로킹되면 다른 작업을 수행하지 못한다.
    - 입출력 할 동안 다른 클라이언트의 연결 요청 수락 불가
    - 입출력 할 동안 다른 클라이언트의 입출력 불가
  - **UI** 생성/변경 스레드에서 블로킹 메소드를 호출하지 않도록
    - **UI** 생성 및 변경이 안되고 이벤트 처리 불가

# TCP 네트워킹

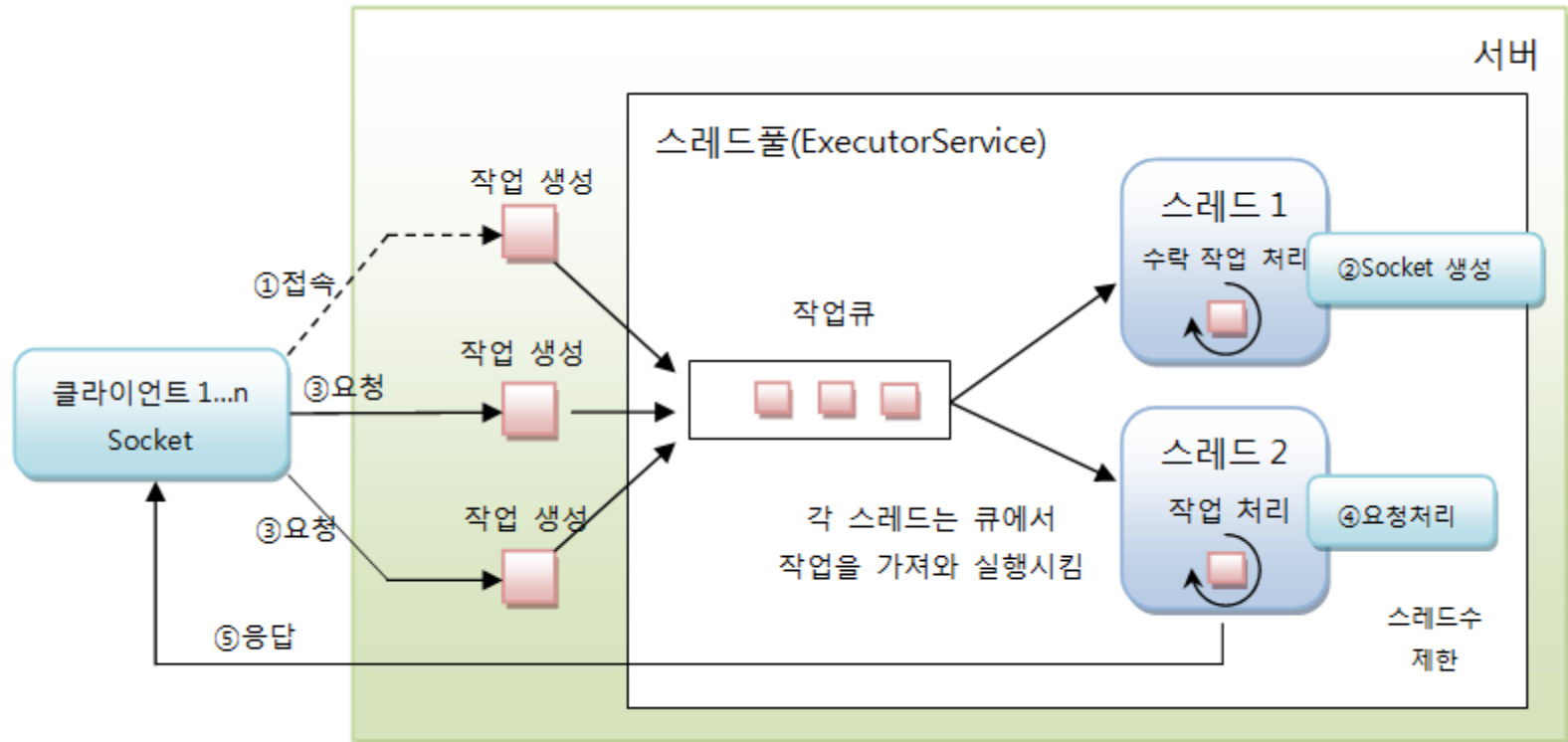
## ■ 스레드 병렬 처리

- **Accept(), connect(), read(), write()**는 별도 작업 스레드 생성



# TCP 네트워킹

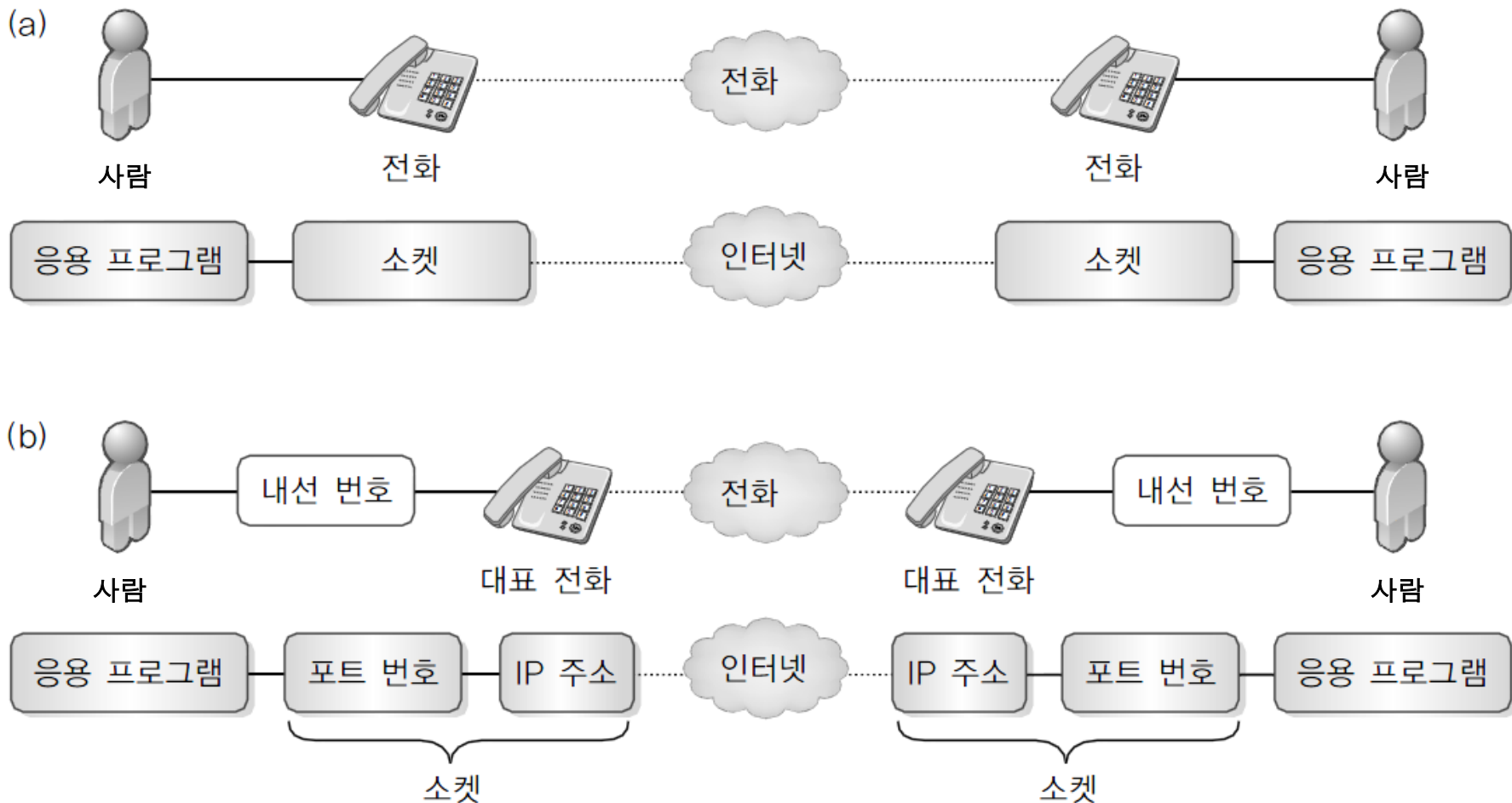
## ■ 스레드풀 사용해 스레드 수 관리



- 스레드풀은 스레드 수 제한해 사용
- 갑작스런 클라이언트의 폭증은 작업 큐의 작업량만 증가
  - 서버 성능은 완만히 저하
  - 대기하는 작업량 많아 개별 클라이언트에서 응답을 늦게 받기도

# 소켓의 개념

## 전화 통신과 소켓 통신 비교



# TCP 기반 네트워크 프로그램 작동 구조

## SERVER

1. 소켓 생성
2. Bind (ip, port 할당)
3. Listen (설정 - Backlog)
4. Accept (연결 요청 대기)
5. 통신 (데이터 보내기 / 받기)
6. 소켓 종료

연결된 통신 소켓 생성

## CLIENT

1. 소켓 생성
2. Connect (서버에 연결 요청)
3. 통신 (데이터 보내기 / 받기)
4. 소켓 종료