



오픈소스 데이터센터 포럼 기술세미나

1주차 모임

# 순서



1. 모임 소개
2. 스터디 방법
3. 스터디



# 1. 모임 소개

# 1. 모임 소개

교육  
공개



개설자 정보

정보 | 다른모임

유명환 (뻔뻔강사) [HOME](#)

[funfun.yoo@gmail.com](mailto:funfun.yoo@gmail.com)

010-7212-3929

엑세스 주식회사

문의사항은 메일 / 전화 / 댓글을 이용해주세요.

모임수정

모임관리

이메일

SMS

## 오픈소스 데이터센터 포럼 기술세미나 1주차 모임

모임기간 4월 5일 (수) 19시 30분 ~ 22시 00분

모임장소 [토즈 강남토즈타워점] 서울 강남구 토즈타워 1층

신청인원 총 20명 | 0명 신청가능

페이스북 [오픈소스 데이터센터 포럼] 그룹에 참석하기

#오픈소스데이터센터포럼

신청기간 3월 28일 (화) 22시 00분

### 커본그룹

선착순 | 총 20명

마감

<http://onoffmix.com/event/95822>

NAVER D2

본 모임은 [네이버 D2] 후원으로 [오픈스택 한국 커뮤니티]와 협동으로 진행되는 스터디 모임입니다.

무료 · 1명

참여 신청이 마감되었습니다.



12

# 1. 모임 소개

<https://www.facebook.com/groups/opensourcedatacenterforum/>



페이스북 그룹 [오픈소스 데이터센터 포럼]



왜 그룹을 만들었을까?

유명환님이 링크를 공유했습니다.  
18시간

★★★ 스터디 모임 1주차 안내 ★★★

<http://onoffmix.com/event/95822...> 더 보기

설명 수정  
페이스북의 OCP (Open Compute Project) 와 같이 데이터센터 관련 인프라 구조와 기술들을 같이 교류하고 공유하는 인프라 전문 커뮤니티입니다. 😊

그룹 유형  
스터디 그룹

# 1. 모임 소개

The screenshot shows the homepage of the Open Compute Project (OCP) website. At the top, there is a navigation bar with links: About (highlighted with a green underline), Learn, Buy, Participate, Projects, News, Contact, Sign In, and a search icon. To the left, there is a logo for "OPEN Compute Project" featuring a blue and yellow circular graphic. On the right side of the page, there is a large black circle containing text: "Hundreds of active member companies" and "Collaborating to improve infrastructure design". The main content area features a large image of a server room with rows of server racks. Overlaid on this image is a large white text block that reads: "About OCP" followed by a horizontal line, and then "The Open Compute Project (OCP) is a collaborative community focused on redesigning hardware technology to efficiently support the growing demands on compute infrastructure." A red dotted rectangle highlights the word "technology" in this text.

<http://opencompute.org/about/>

About OCP

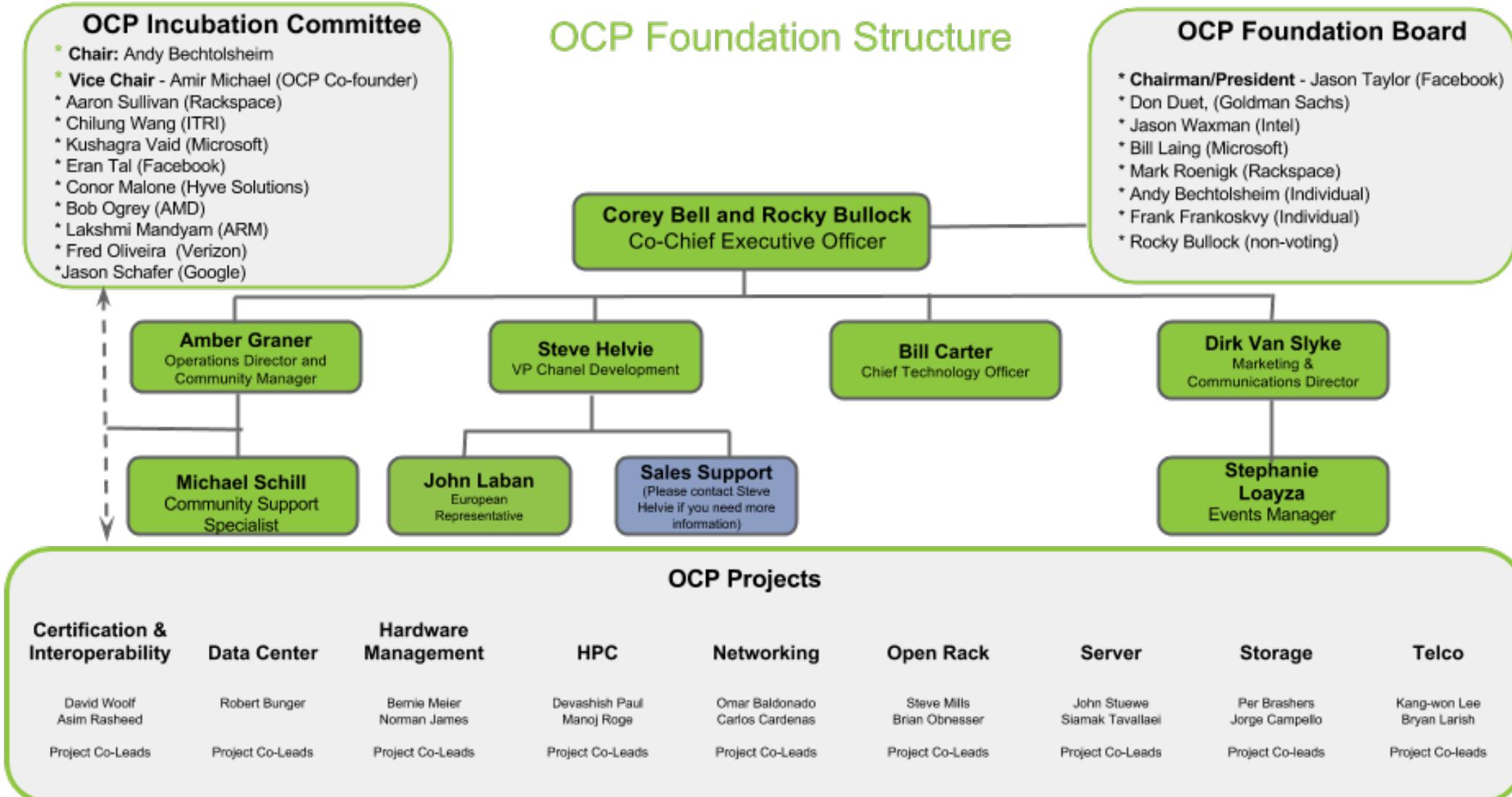
The Open Compute Project (OCP) is a collaborative community focused on redesigning hardware technology to efficiently support the growing demands on compute infrastructure.

Hundreds  
of active member companies

Collaborating to  
improve  
infrastructure  
design

# 1. 모임 소개

<http://opencompute.org/assets/assets/OCP-Org-Chart-13-OCT-2.png>



\*IC Chair and Vice Chair are Board appointed - all other seats will be community elected \*\*While currently staffed by Facebook, all board members companies may give staffing resources to the Foundation.

# 1. 모임 소개

Prineville Data Center

좋아요 ▾ 팔로잉 ▾ 공유하기 ...

메시지 보내기

## Prineville, OR Data Center

Dashboard: PUE & WUE

Next update:  
51 seconds

2.5 hour delay  
24 hour period

11:50 AM, Today

Power Usage Effectiveness (PUE) **1.07**

Water Usage Effectiveness (WUE) **0.00 L/kWh**

Humidity (Outdoors) **51% /100%**

Temperature (Outdoors) **56° F /13.3°C**

Annualized Numbers — The chart above shows real-time PUE, WUE, temperature and humidity for our data center. The numbers to the right are the trailing 12-month PUE and WUE as of the end of the last quarter.

PUE **1.09** TTM

WUE **0.18** TTM

www.facebook.com/PrinevilleDataCenter/

페이스북이 9억명 가입자에게 서비스를 제공하기 위해 운영중인 서버가 18만대 이상인 것으로 예상됐다. 2009년 3만대에서 3년만에 6배 늘어났다.

13일(현지시간) 외신에 따르면, 아마존웹서비스의 제임스 해밀턴 부사장은 최근 페이스북이 발표한 데이터 센터 에너지 효율성을 근거로 운영 서버의 규모를 18만대로 계산했다.

# 1. 모임 소개

facebook Community Update

1.27.2016



1.59 Billion

on Facebook each month



900 Million

on WhatsApp each month



800 Million

on Messenger each month



400 Million

on Instagram each month



950+ Million

people notified by  
Safety Check in 2015



1 Billion

people use Groups  
each month



500 Million

people use Events  
each month



50 Million

small businesses  
use Pages



19 Million

people connected via  
Internet.org



Shipped

Samsung Gear VR  
with Oculus software





## 2. 스터디 방법

## 2. 스터디 방법



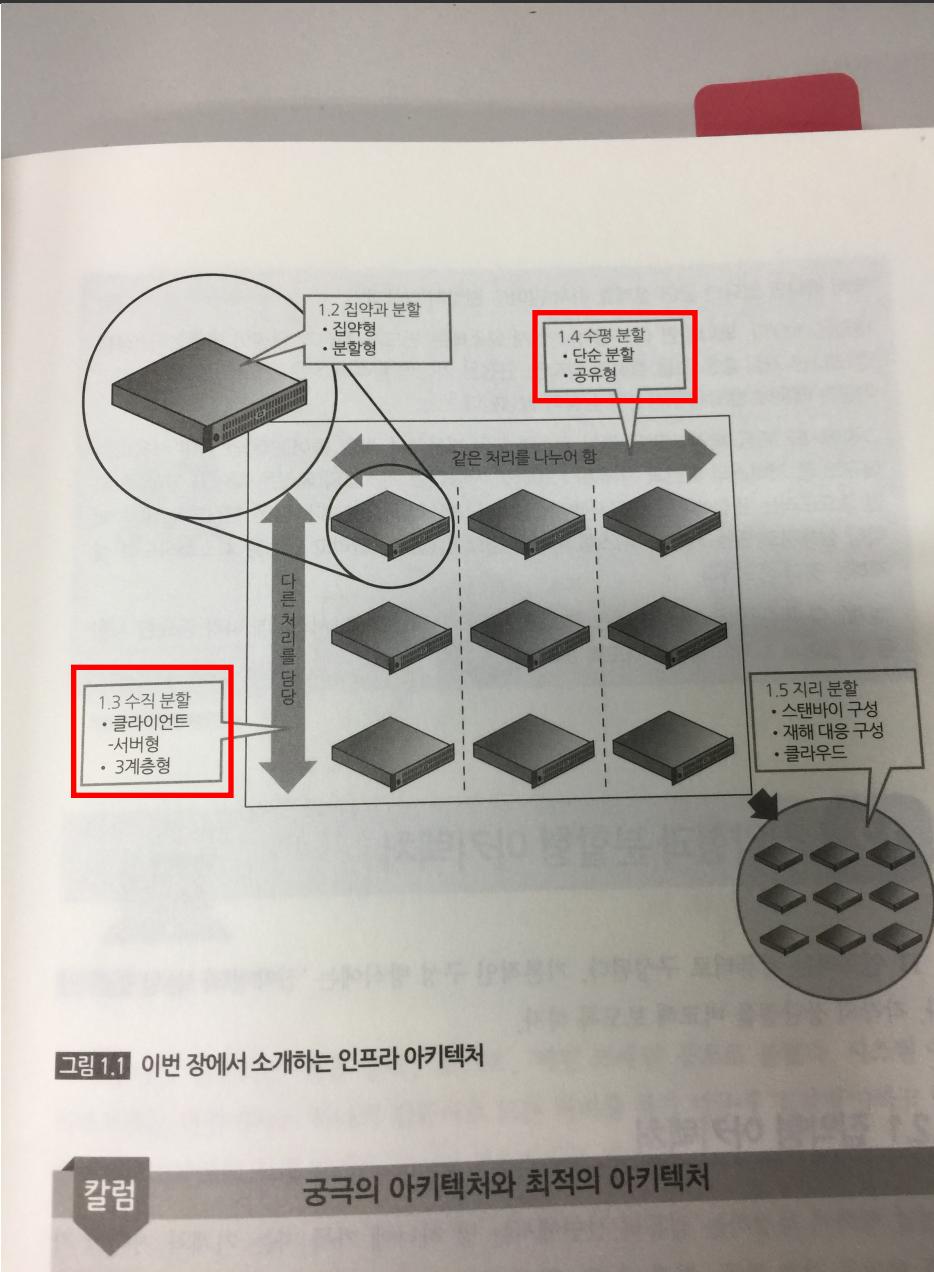
# 3. 스터디





# Ch.1 인프라 아키텍처를 살펴보자

# Ch.1 인프라 아키텍처를 살펴보자



# Ch.1 인프라 아키텍처를 살펴보자

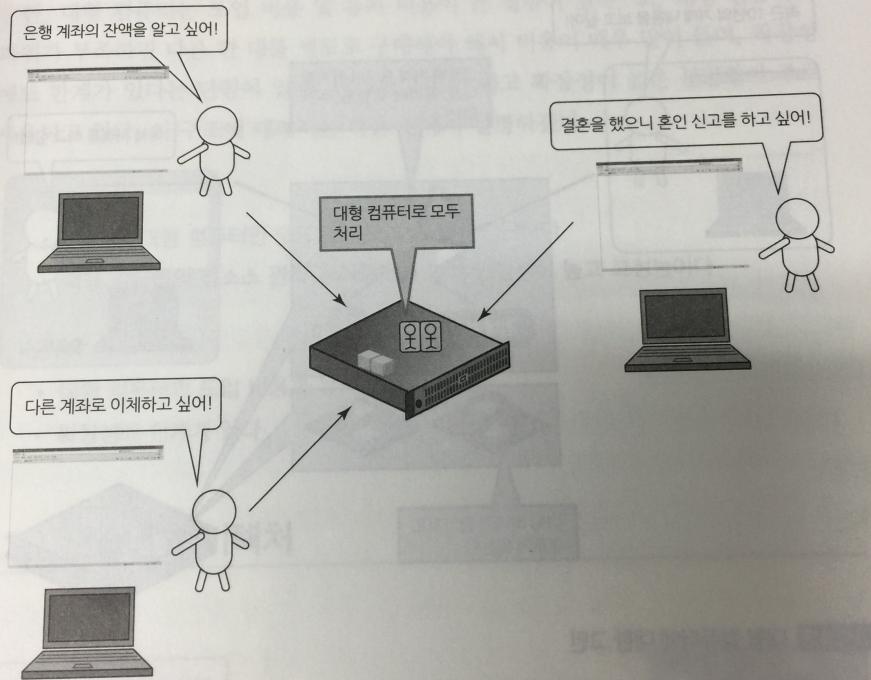


그림 1.2 집약형 아키텍처

이런 대형 컴퓨터는 ‘범용 장비’, ‘호스트’, ‘메인 프레임’ 등으로 불렸다. 시스템 아키텍처라는 관점에서는 하나의 컴퓨터로 모든 처리를 하기 때문에 ‘집약형’이라고 할 수 있다. 집약형의 최대 장점은 구성이 간단하다는 것이다.

단, 대형 컴퓨터는 도입 비용 및 유지 비용이 큰 경향이 있다. 또, 대형 컴퓨터의 파워가 부족하면 다른 한 대를 별도로 구매해야 해서 비용이 매우 많이 들며, 확장성에도 한계가 있다는 단점이 있다. 현재는 가격이 싸고 확장성이 높은 분할형이 주로 사용되고 있다. 이 구조에 대해서는 다음 절에서 설명하겠다.

## 장점

- 한 대의 대형 컴퓨터만 있으면 되므로 구성이 간단하다
- 대형 컴퓨터의 리소스 관리나 이중화에 의해 안정성이 높고 고성능이다

## 단점

- 대형 컴퓨터의 도입 비용과 유지 비용이 크다
- 확장성에 한계가 있다

## 1.2.2 분할형 아키텍처

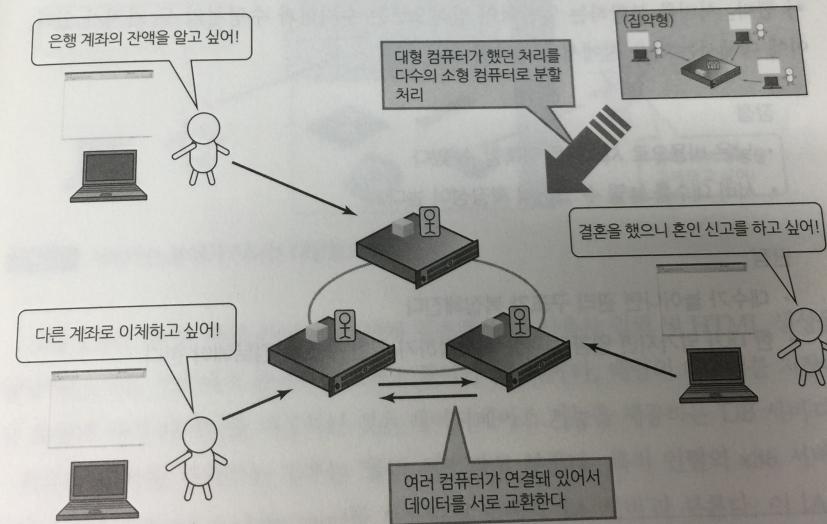
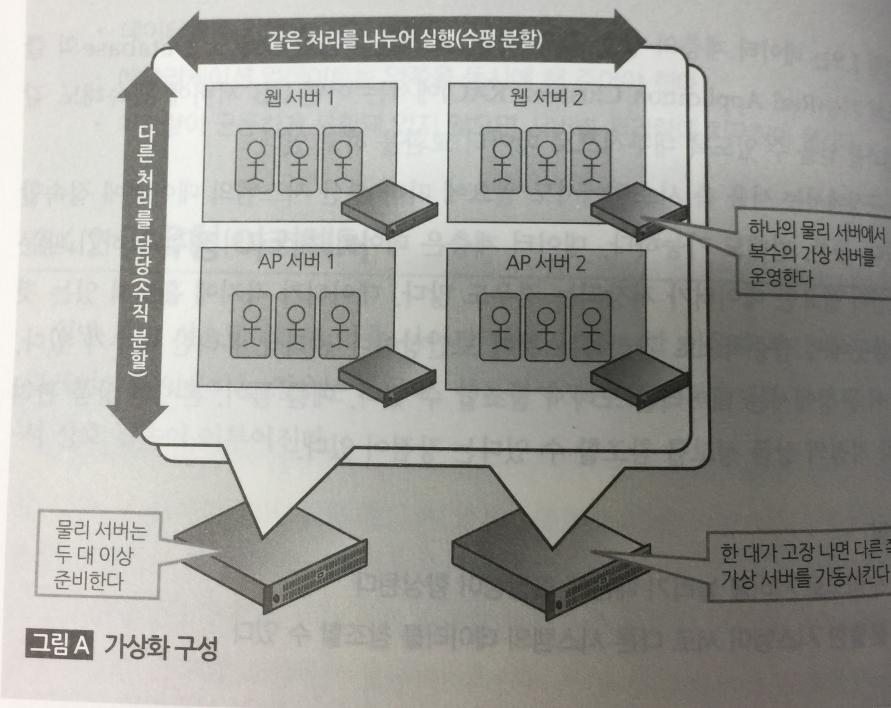


그림 1.4 분할형 아키텍처

# Ch.1 인프라 아키텍처를 살펴보자

대형 컴퓨터는 높은 비용이 문제였지만, 서버는 서가 서버도 충분한 성능을 가지고 있기 때문에 시스템 전체를 한 대로 집약해도 별다른 문제가 없을 수 있다. 하지만 서버 한 대에 지나치게 집약하면 확장성 문제가 불거질 수 있다. 또한, 저가 서버는 안정성에 문제가 있을 수 있다.

집약과 분할형 양쪽의 장점을 취하는 접근법이 '가상화'다. 물리 서버를 가상화 기능으로 여러 대의 가상 서버로 분할하는 방식이다. 그럼 A에 있는 예처럼, 두 대의 웹 서버와 두 대의 AP 서버, 총 네 대의 가상화 서버를 하나의 물리 서버에서 운영하고 있다. 또한, 물리 서버를 이중화해서 한 대가 망가지더라도 계속해서 운영할 수 있다(이중화에 대해서는 7장에서 상세히 설명하겠다).



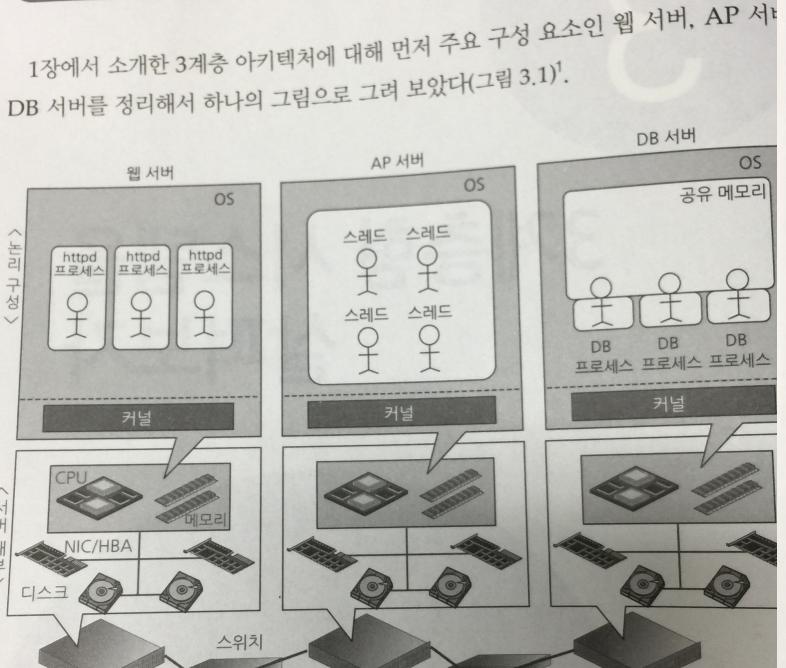


# Ch.3 3계층형 시스템을 살펴보자

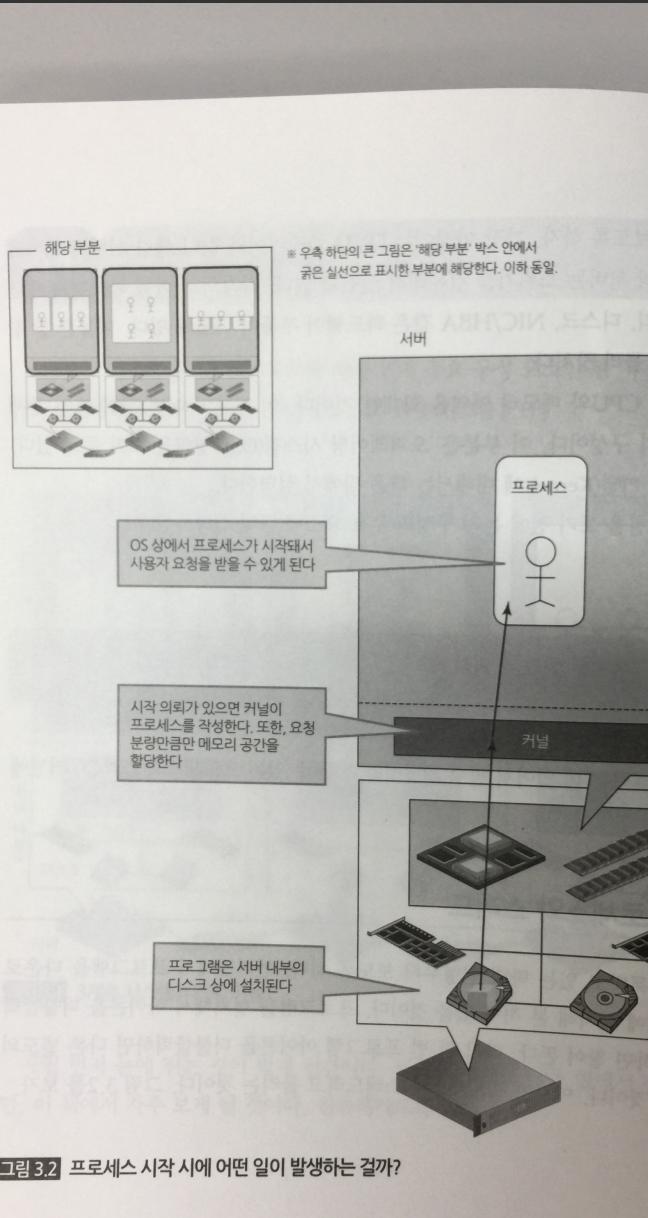
# Ch.3 3계층형 시스템을 살펴보자

## 3.1

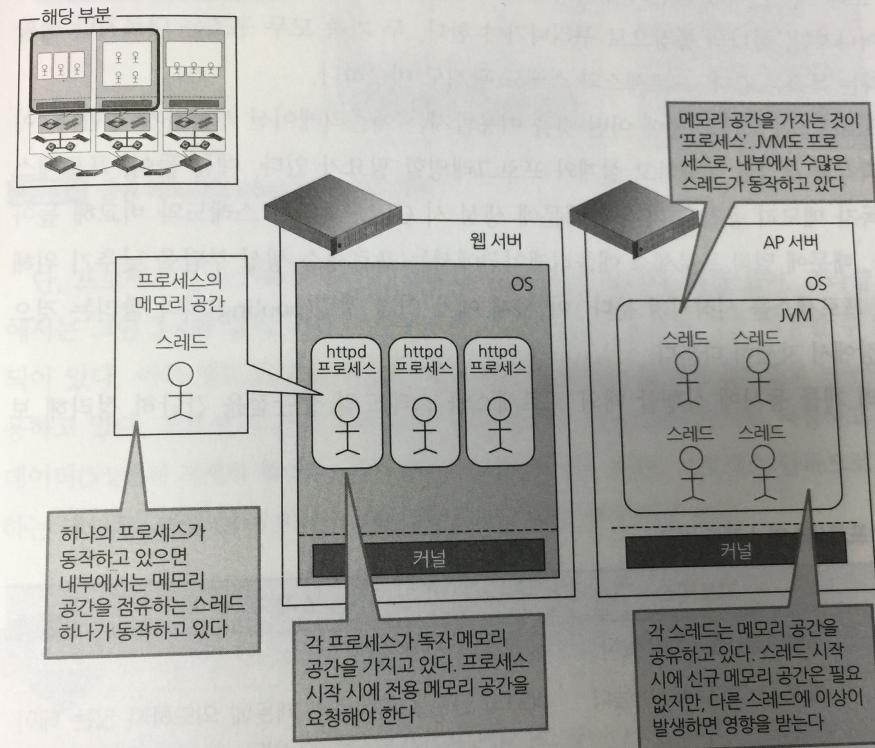
### 3계층형 시스템의 구성도



제일 먼저 눈에 뛰는 것이 막대 인간이다. 이것이 정체는 다음 절에서 설명하는 막대 인간이지만, 이 책에서 자주 보게 될 것이다. 친숙해지도록 하자.

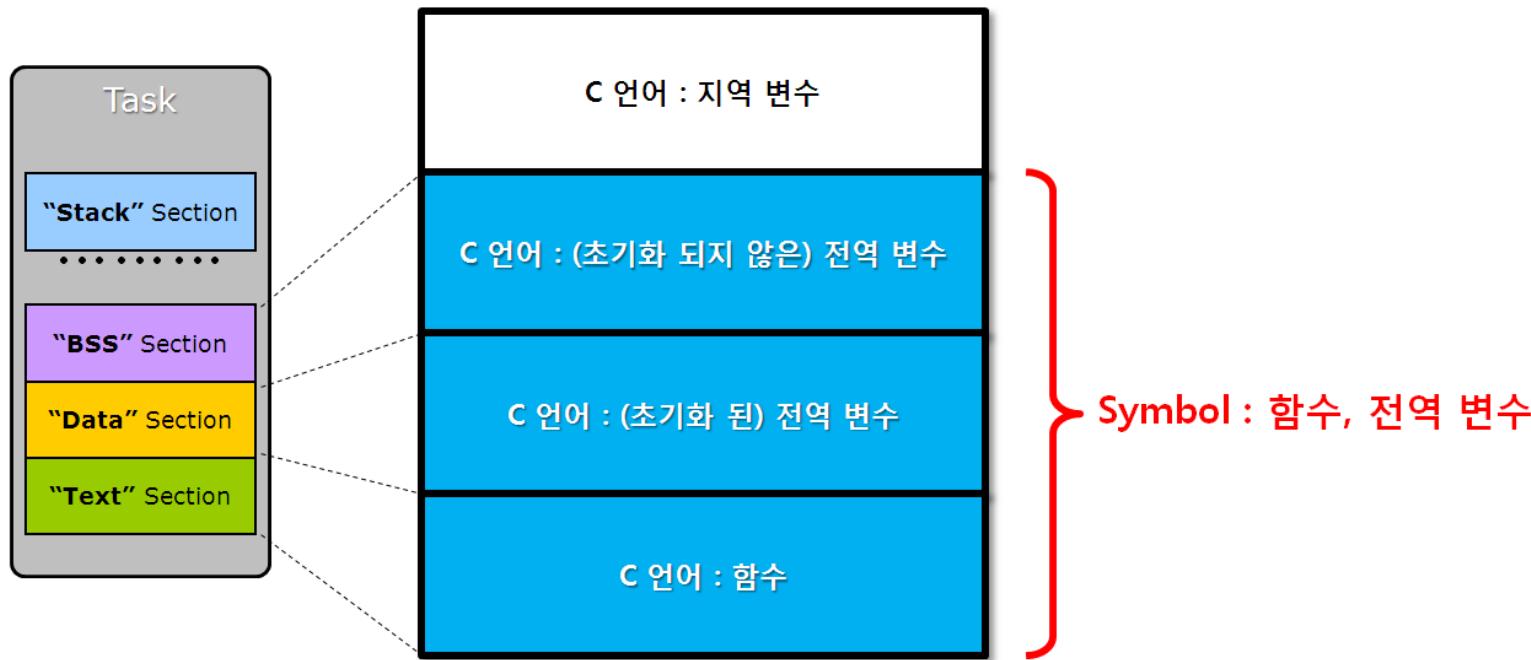


명)에 의해 메모리 상에 확보된다. 이 메모리 공간은 막대 인간이 자신을 위해 소유하는 공간으로, 개인 공간이라 할 수 있다. 다양한 처리를 하면서 데이터를 주고받기 위해 이 공간을 사용한다. 그럼 3.3에 있는 것처럼 프로세스 시작 시에 이 공간이 확보된다.



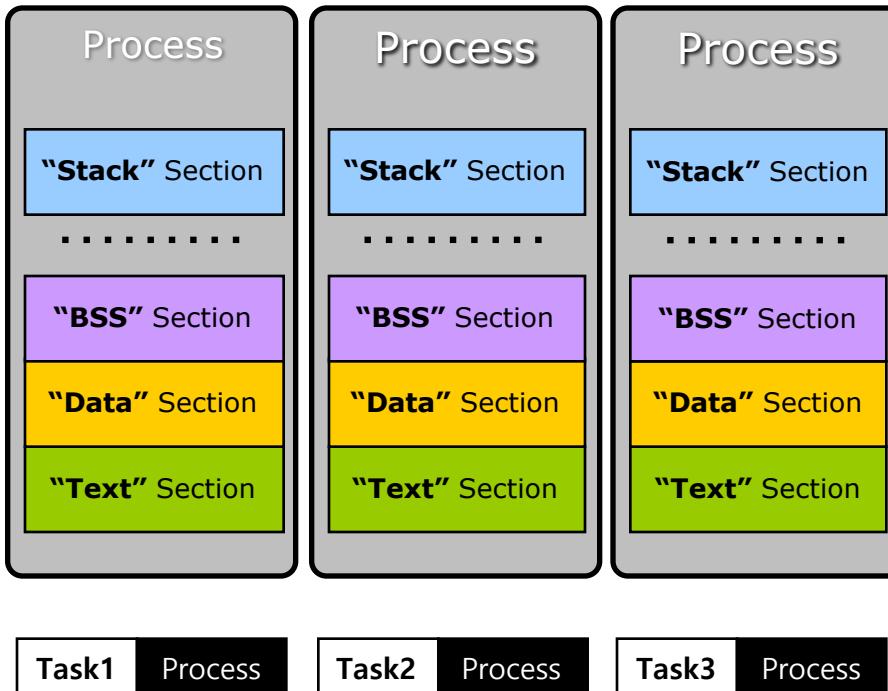
먼저, 웹 서버를 보자. ‘httpd 프로세스’라고 쓰여 있는 막대 인간이 프로세스 그 주변을 감싸고 있는 하얀 공간이 있는데, 이것이 프로세스의 메모리 공간을

# Ch.3 3계층형 시스템을 살펴보자

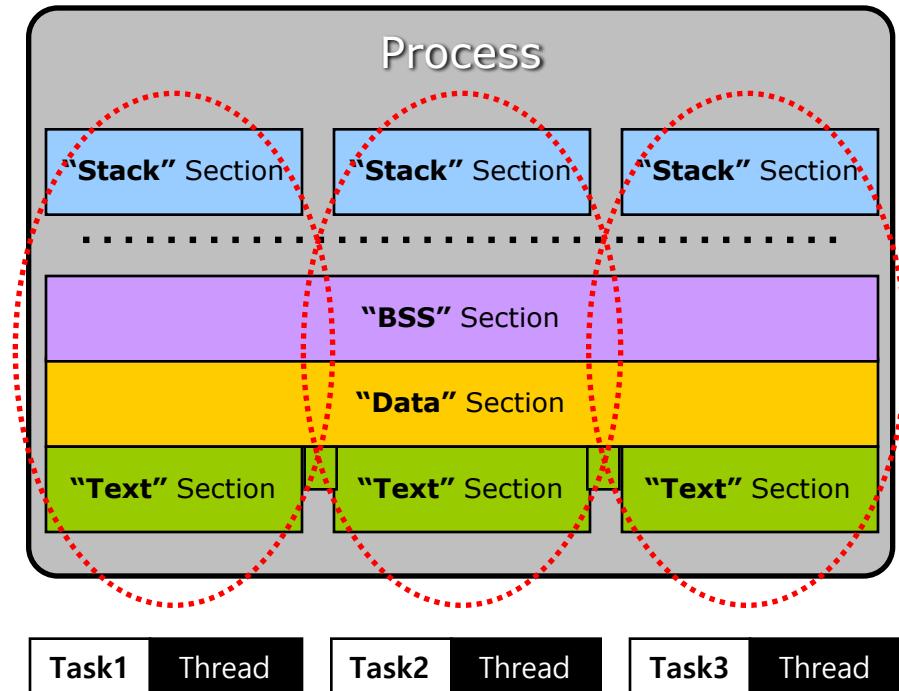


# Ch.3 3계층형 시스템을 살펴보자

## Multi-Process (Non-RTOS)



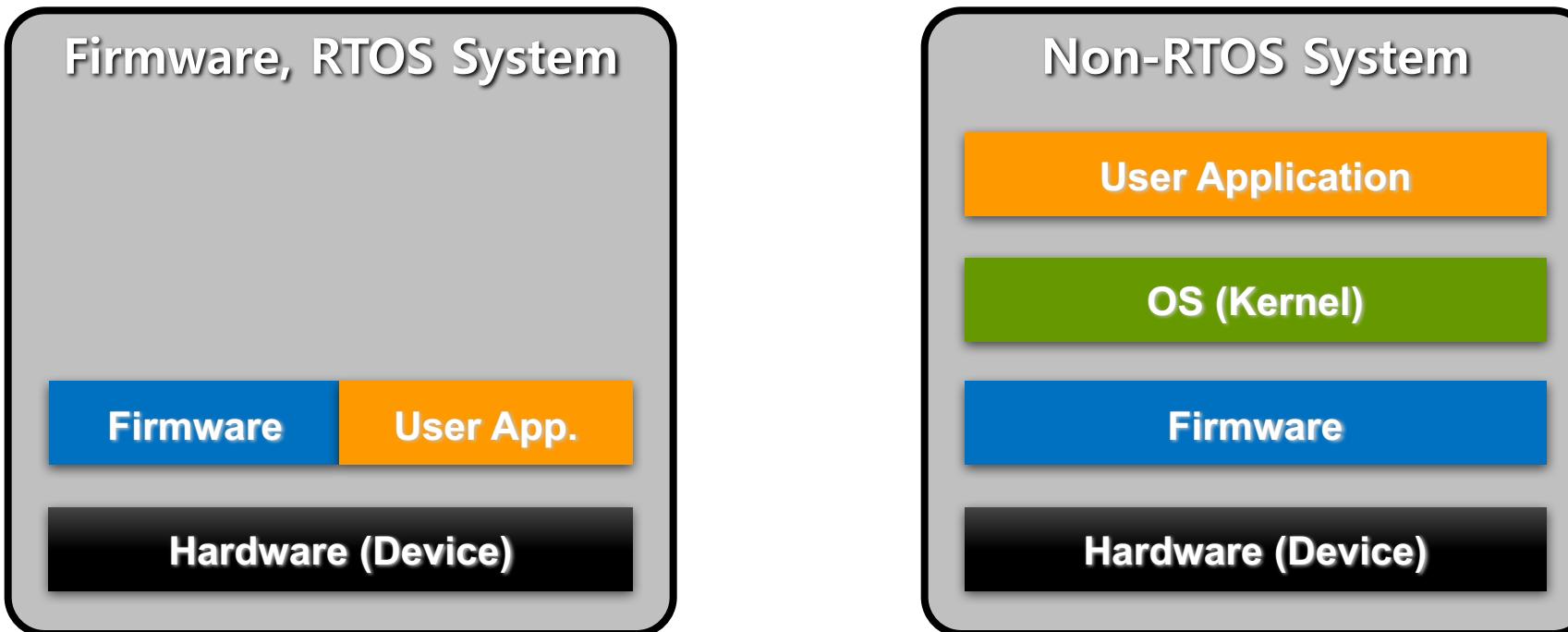
## Multi-Thread (RTOS)



- Linux, Windows
- 프로세스(Process)들마다 독립된 메모리 영역
- 프로세스의 生死 여부가 다른 프로세스에게 영향을 미치지 않는다!
- User Level 영역과 Kernel Level 영역으로 구분된다!

- VxWorks, pSOS, eCOS, MicroC/OS-II, TinyOS
- 쓰레드(Thread)들이 일부 영역(Data, BSS)을 서로 공유
- 쓰레드의 生死 여부가 다른 쓰레드에게 영향을 미친다!
- User, Kernel Level 영역의 구분이 없다!

# Ch.3 3계층형 시스템을 살펴보자



# Ch.3 3계층형 시스템을 살펴보자

[http://www.allsoft.co.kr/bbs/board.php?bo\\_table=tip\\_tech&wr\\_id=114](http://www.allsoft.co.kr/bbs/board.php?bo_table=tip_tech&wr_id=114)

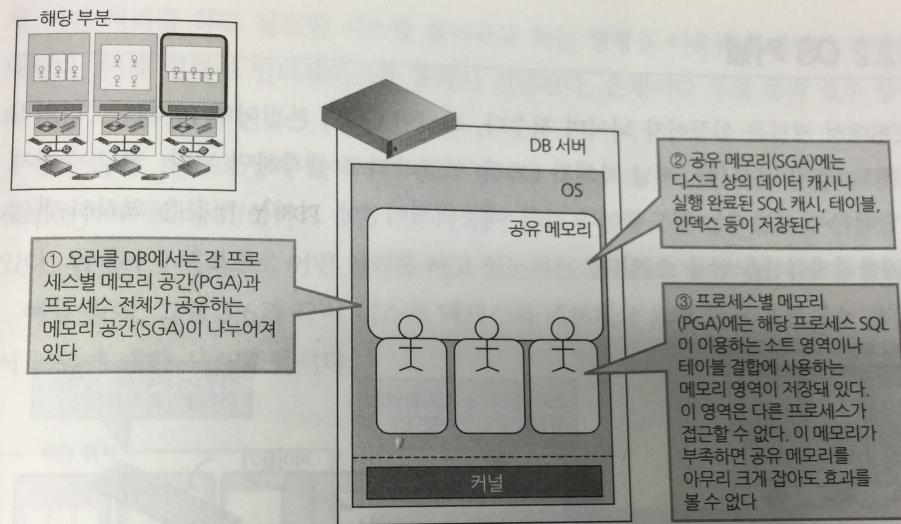


그림 3.4 공유 메모리형 DBMS의 메모리 관리

단, 프로세스가 메모리 공간을 공유할 수 없는 것은 아니다. 예를 들어, 오라클에서는 그림 3.4와 같이 여러 프로세스가 ‘공유 메모리 공간’을 상호 이용할 수 되어 있다. 이와 별도로 프로세스별 독자 메모리 영역도 있어서 용도별로 나누어 용하고 있다. 프로세스 간에 공유하고 싶은 데이터, 예를 들어 캐시로 저장하고 데이터(5장에서 자세히 설명)는 공유 메모리 상에 둔다. 한편, 프로세스 단독으로 하는 데이터, 예를 들어 자신이 계산할 결과는 전용 메모리에 둔다.

## PGA (Program Global Area)

- 서버 프로세스에 생성되며 오라클에서 사용하는 메모리 영역
- 데이터베이스에 접속하는 모든 유저에게 할당되어 각각의 서버 프로세스가 독자적으로 사용하는 오라클 메모리 영역
- 하나의 유저 프로세스에 하나의 서버 프로세스가 할당되며, 또한 하나의 서버 프로세스는 하나의 PGA를 생성한다.

## SGA (System Global Area)

- 오라클이 데이터를 읽거나 변경하기 위해 사용하는 공용 메모리 영역
- 공용 메모리 영역이므로 동일 데이터베이스에 접속하는 모든 사용자는 동일 SGA를 사용한다.
- 인스턴스가 시작될 때 시스템 메모리에서 할당 받으며 종료될 때 다시 시스템 메모리 영역으로 반환한다.

# Ch.3 3계층형 시스템을 살펴보자

OS에서 커널은 심장이자 뇌이며 척수다. 커널이 OS의 본질이며 나머지는 틈이 농하고 해도 과언이 아니다. 커널 자체가 OS의 '인프라'라고 생각하면 된다. 커널이 존재하기 때문에 개발자는 하드웨어나 다른 애플리케이션에 끼치는 영향을 의식하지 않고 애플리케이션을 만들 수 있다.

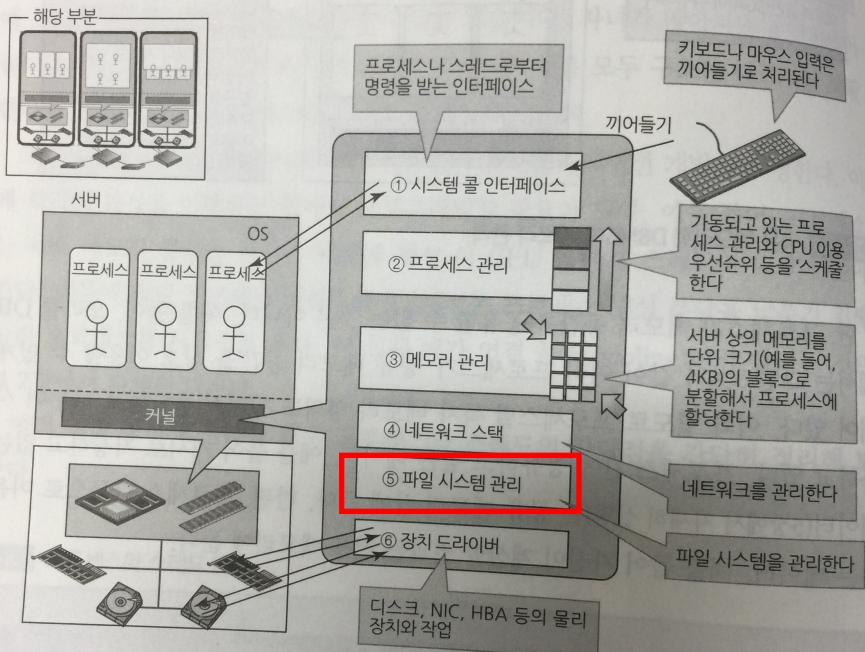


그림 3.5 커널의 여섯 가지 역할

OS 처리는 원칙적으로는 커널을 통해 이루어진다. 커널의 역할에는 여러 가지가 있지만, 그림 3.5와 같이 여섯 가지로 정리할 수 있다. 이번 장에서는 이 중에서 다섯 개를 선별해서 소개한다(④ 네트워크 스택은 6장에서 다룬다).

## ① 시스템 콜 인터페이스

예를 들어, 디스크 상의 데이터를 읽고 싶거나 네트워크 통신을 하고 싶을 때, 또는 새로운 프로세스를 생성하고 싶은 경우에 해당 시스템 콜을 호출하면 기능을 이용할 수 있다. 뒤에서 구체적으로 어떤 처리를 하고 있는지는 프로세스가 의식할 필요가 없다. 예를 들어, 그림 3.6과 같이 디스크 액세스와 네트워크 요청 모두 프로세스 관점에서 커널에 대한 시스템 콜이다.

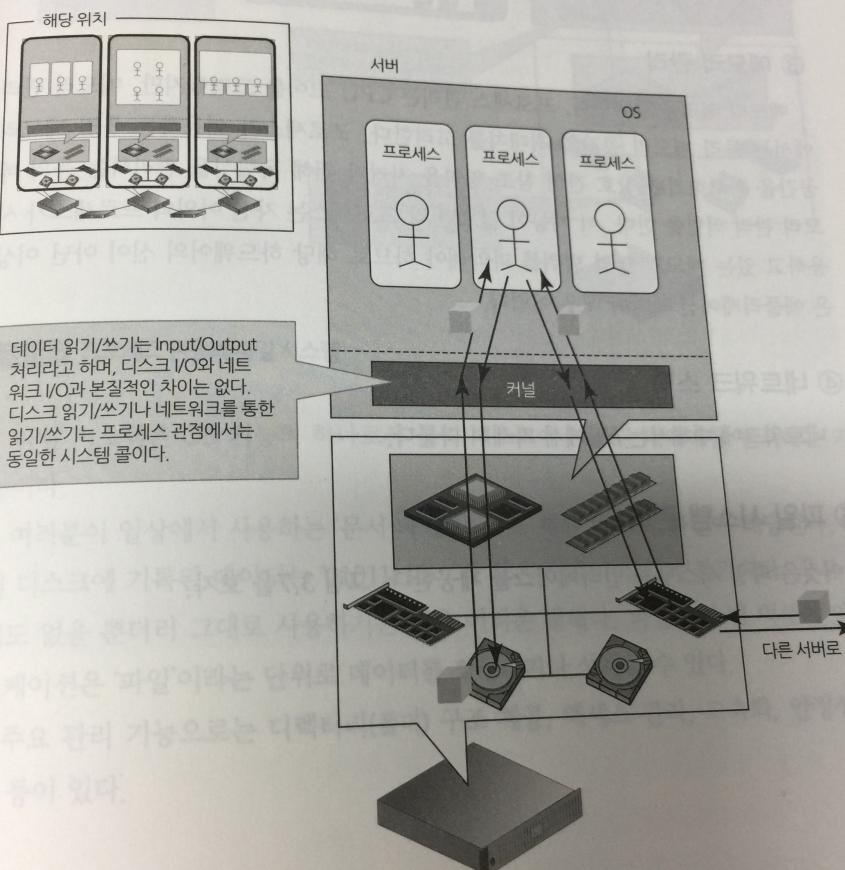


그림 3.6 디스크 I/O와 네트워크 I/O 모두 시스템 콜이다

# Ch.3 3계층형 시스템을 살펴보자

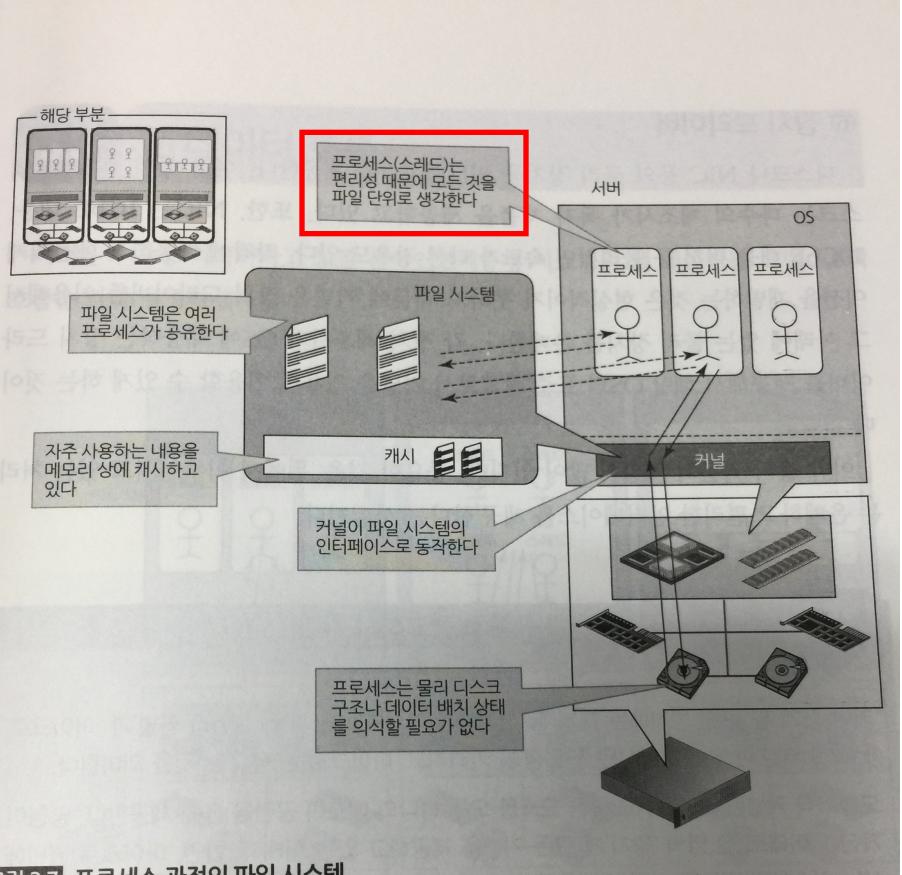


그림 3.7 프로세스 관점의 파일 시스템

파일 시스템은 OS 기능의 하나로서 물리 디스크에 제공된 데이터를 관리하는 기능이다.

여러분이 일상에서 사용하는 ‘문서 파일’이나 ‘표 계산 파일’이 파일에 해당한다. 디스크에 기록된 데이터는 ‘01011110…’과 같은 숫자의 집합에 불과하다. 구분 없을 뿐더러 그대로 사용하기는 매우 어려운 형태다. 파일 시스템 덕분에 파일 케이션은 ‘파일’이라는 단위로 데이터를 작성하거나 삭제할 수 있다.

## 3.3 웹 데이터 흐름

여러분이 가장 익숙하다고 느낄 분야인 3계층형 시스템의 웹 데이터를 그림 3.8의 흐름을 따라 설명하겠다.

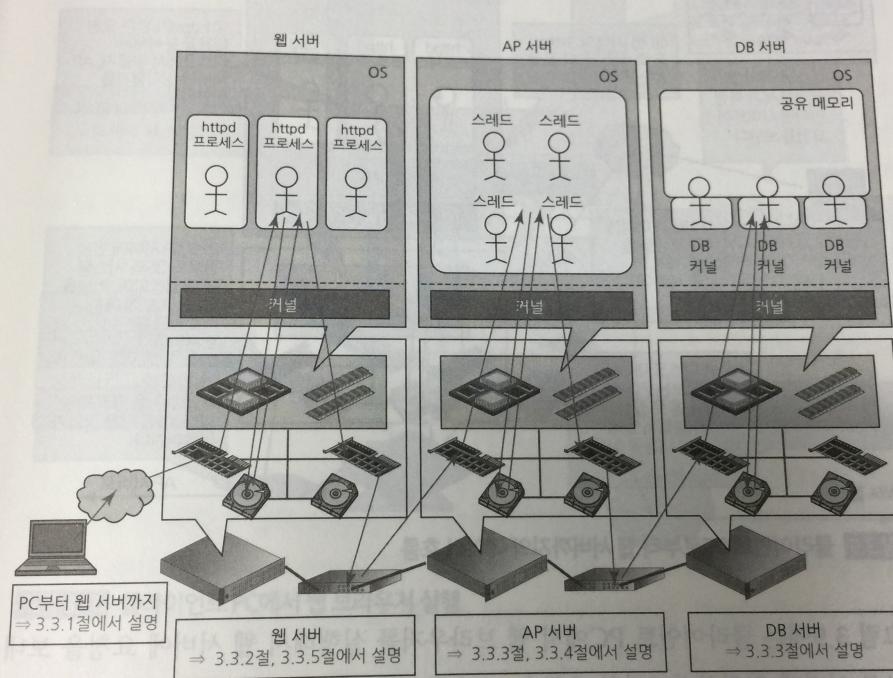
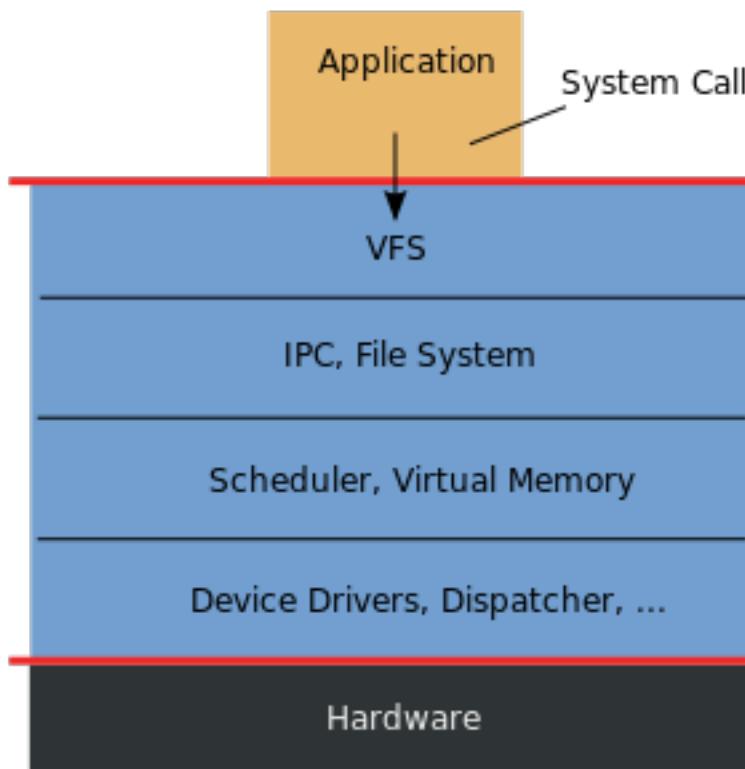


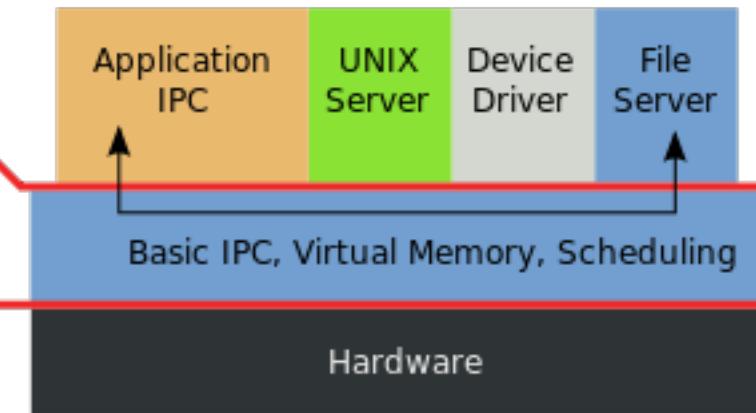
그림 3.8 이번 절에서 설명하는 내용

# Ch.3 3계층형 시스템을 살펴보자

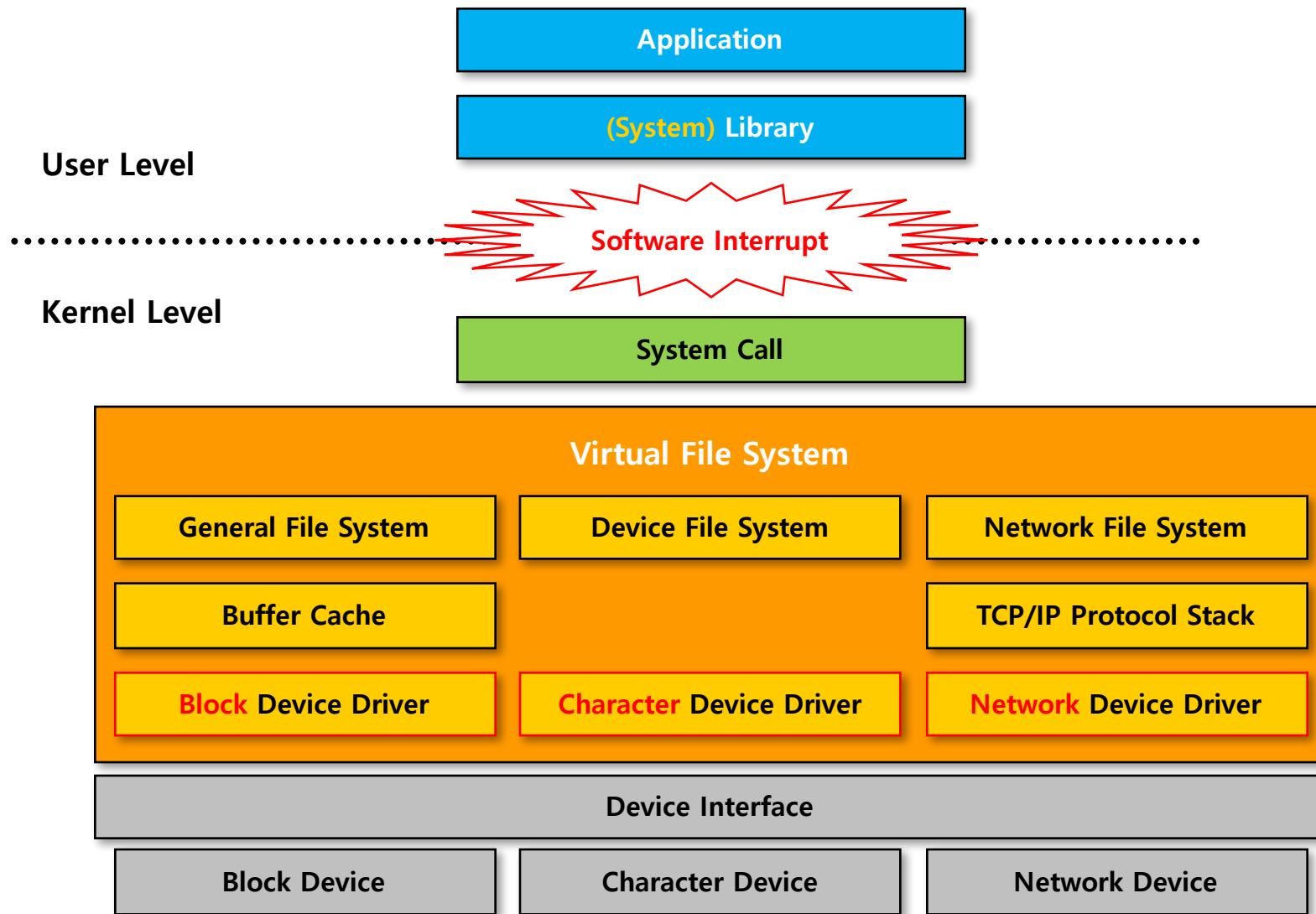
Monolithic Kernel  
based Operating System



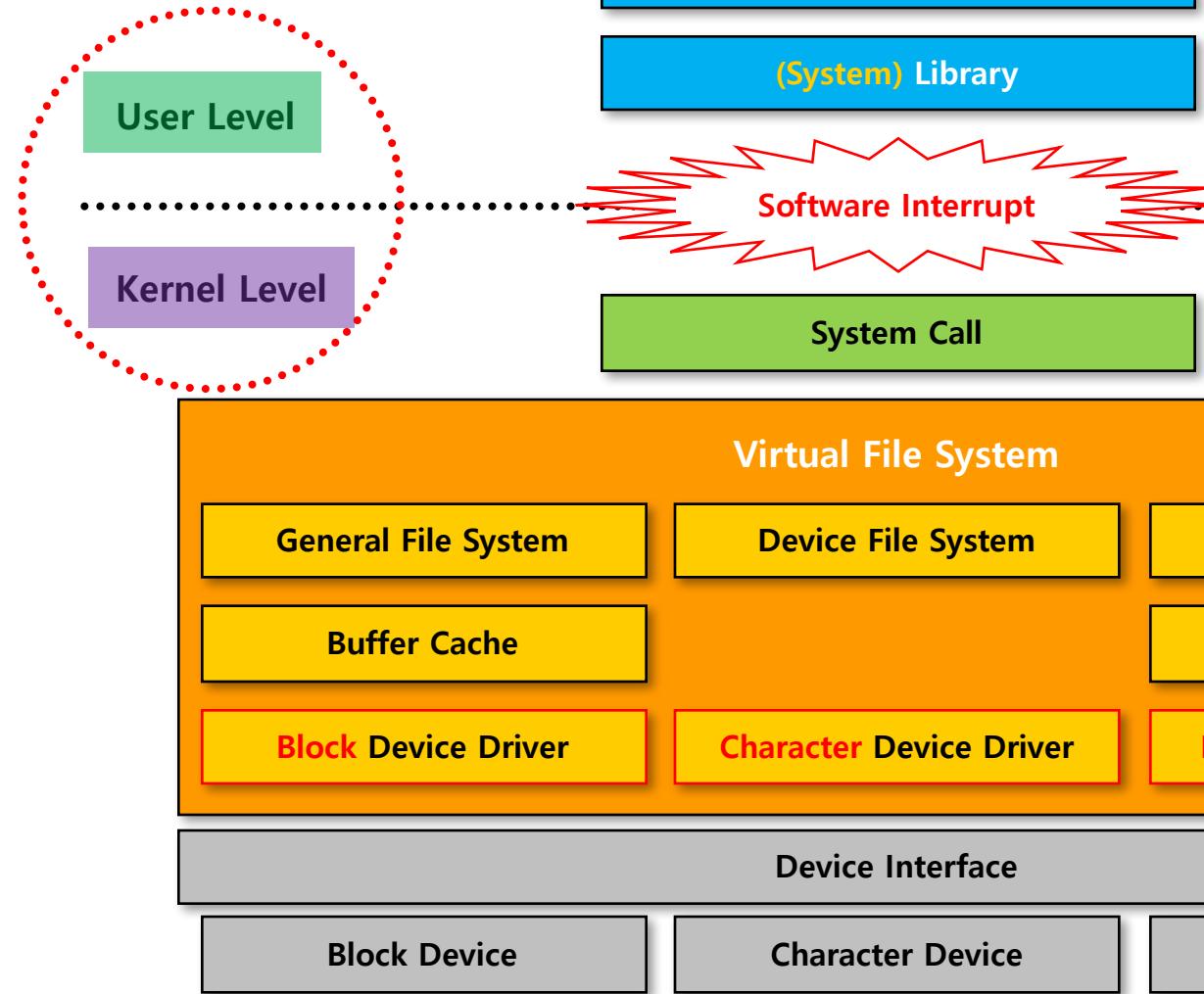
Microkernel  
based Operating System



# Ch.3 3계층형 시스템을 살펴보자



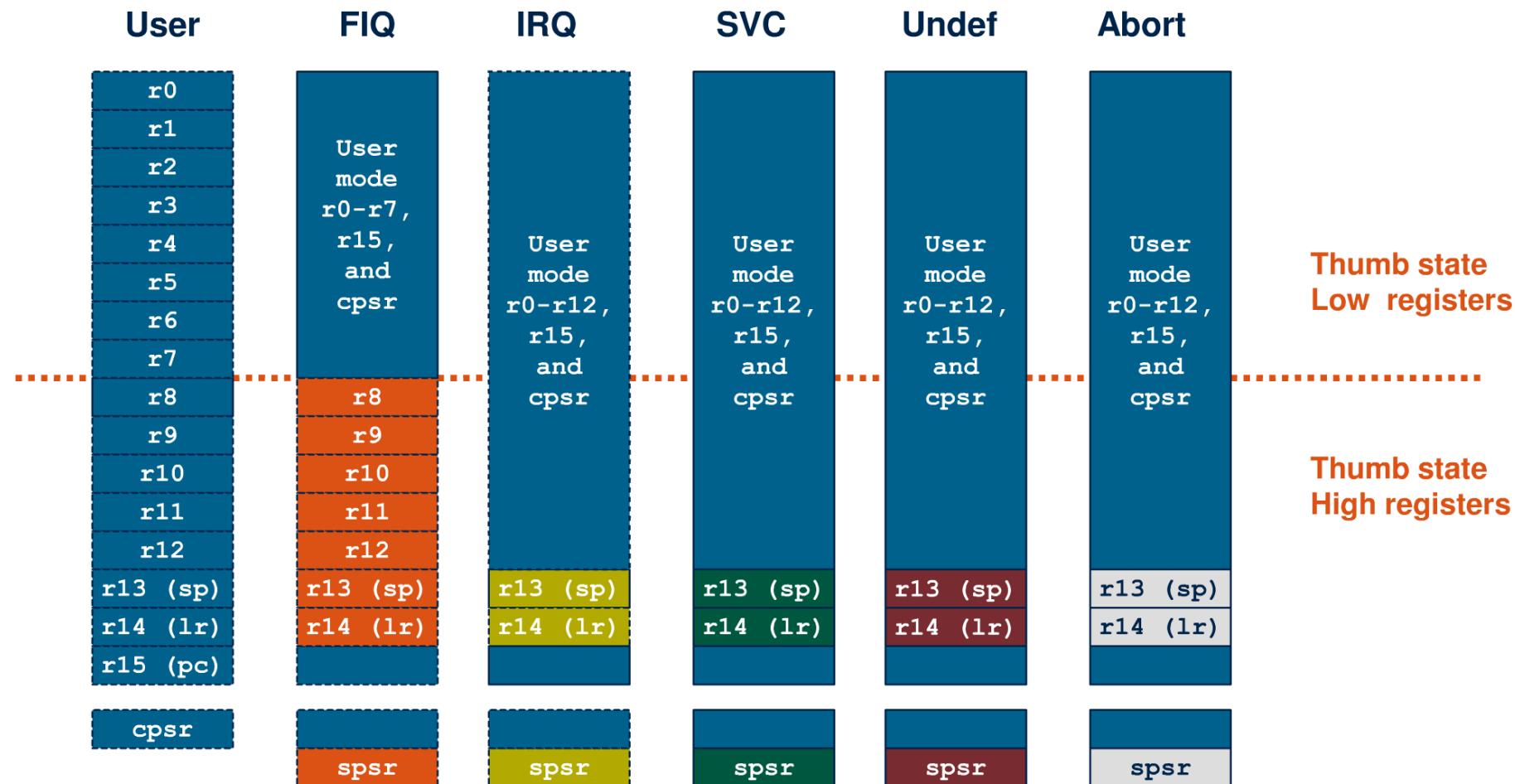
# Ch.3 3계층형 시스템을 살펴보자



Modes							
		Privileged modes		Exception modes			
User	System	Supervisor	Abort	Undefined	Interrupt	Fast interrupt	
R0	R0	R0	R0	R0	R0	R0	
R1	R1	R1	R1	R1	R1	R1	
R2	R2	R2	R2	R2	R2	R2	
R3	R3	R3	R3	R3	R3	R3	
R4	R4	R4	R4	R4	R4	R4	
R5	R5	R5	R5	R5	R5	R5	
R6	R6	R6	R6	R6	R6	R6	
R7	R7	R7	R7	R7	R7	R7	
R8	R8	R8	R8	R8	R8	R8	R8_fiq
R9	R9	R9	R9	R9	R9	R9	R9_fiq
R10	R10	R10	R10	R10	R10	R10	R10_fiq
R11	R11	R11	R11	R11	R11	R11	R11_fiq
R12	R12	R12	R12	R12	R12	R12	R12_fiq
R13	R13	R13_svc	R13_abt	R13_und	R13_irq	R13_fiq	
R14	R14	R14_svc	R14_abt	R14_und	R14_irq	R14_fiq	
PC	PC	PC	PC	PC	PC	PC	PC
CPSR	CPSR	CPSR	CPSR	CPSR	CPSR	CPSR	CPSR
		SPSR_svc	SPSR_abt	SPSR_und	SPSR_irq	SPSR_fiq	

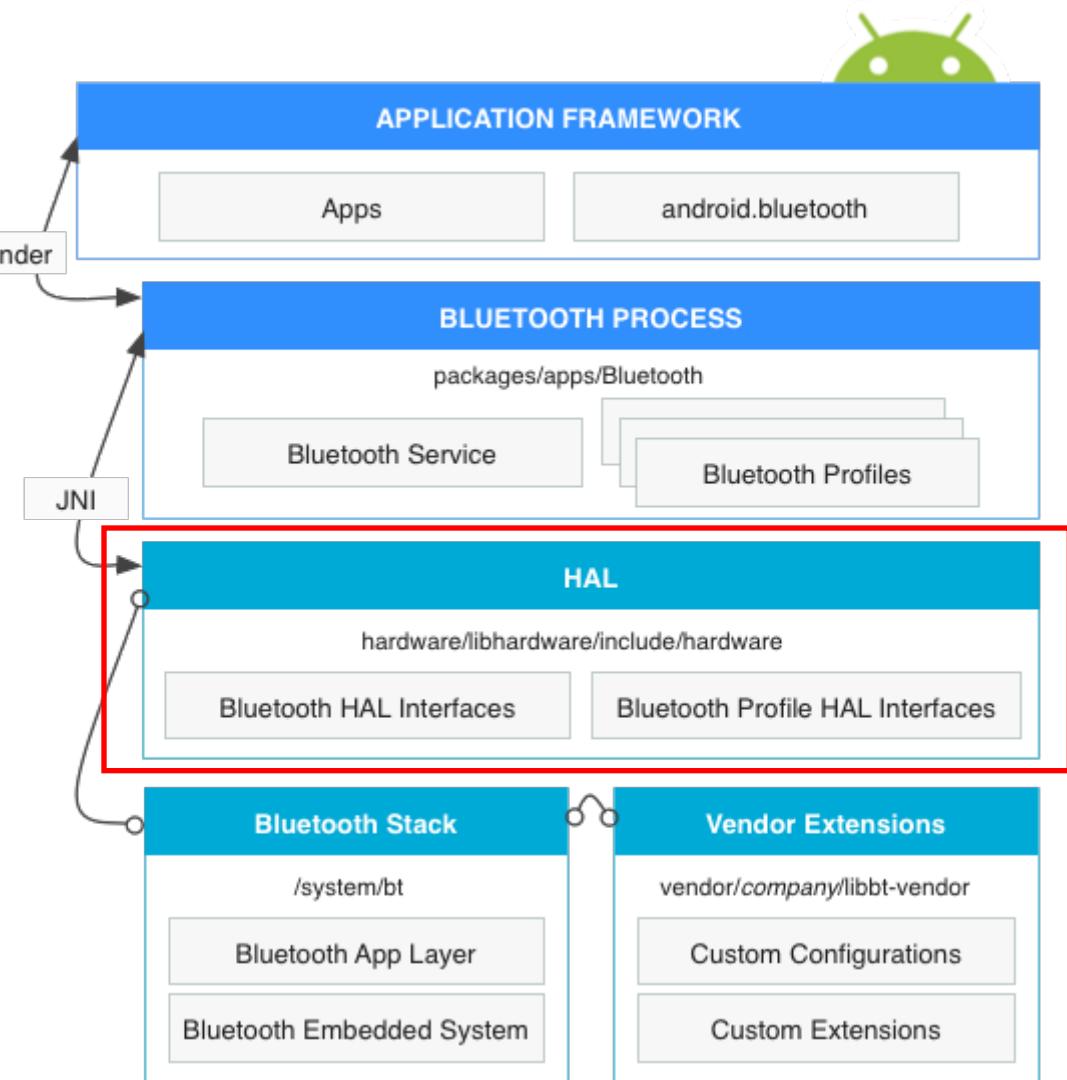
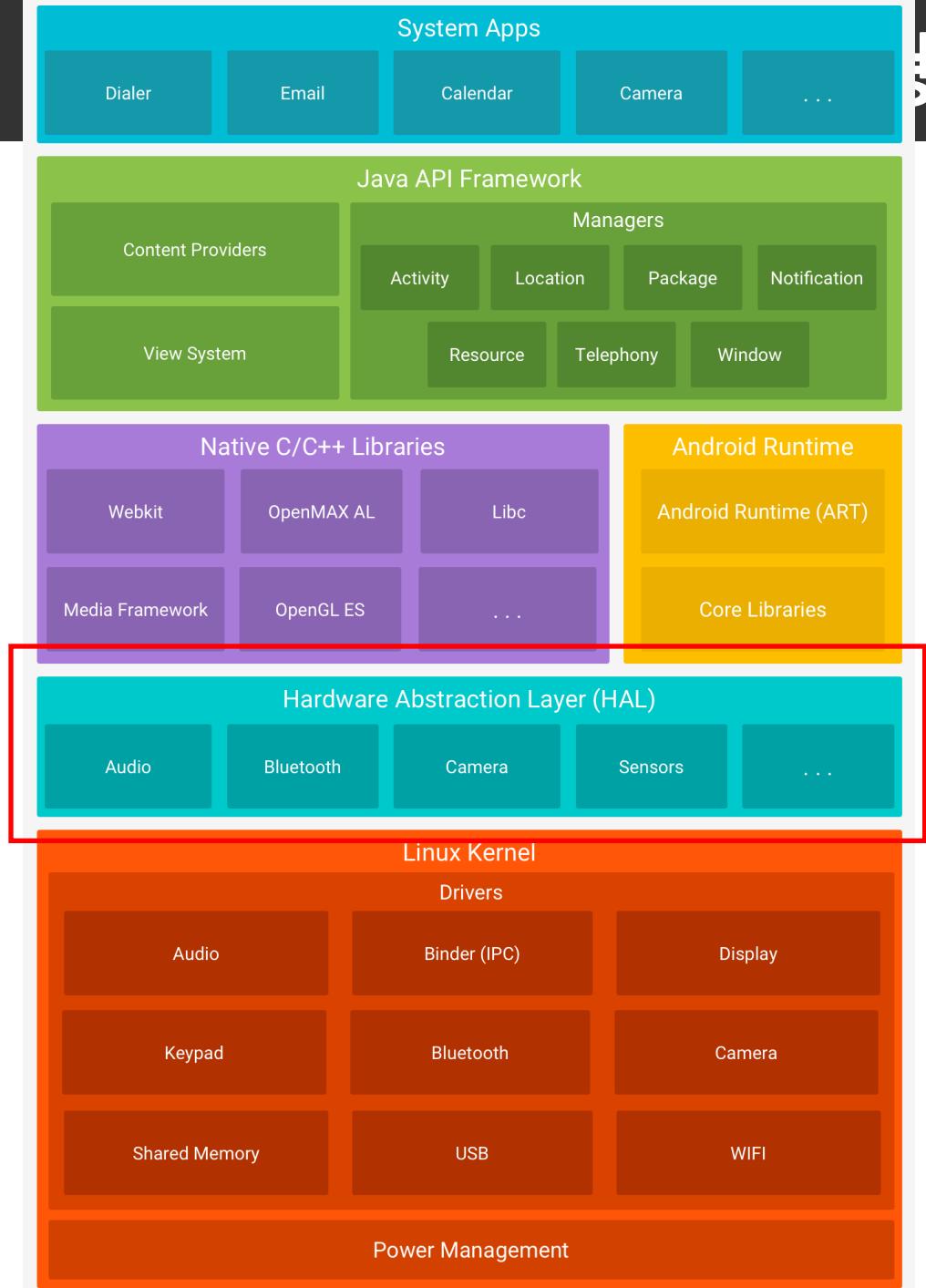
△ indicates that the normal register used by User or System mode has been replaced by an alternative register specific to the exception mode

# Ch.3 3계층형 시스템을 살펴보자



Note: System mode uses the User mode register set

# 안드로이드 시스템을 살펴보자



# Ch.3 3계층형 시스템을 살펴보자

## 3.3.1 클라이언트 PC부터 웹 서버까지

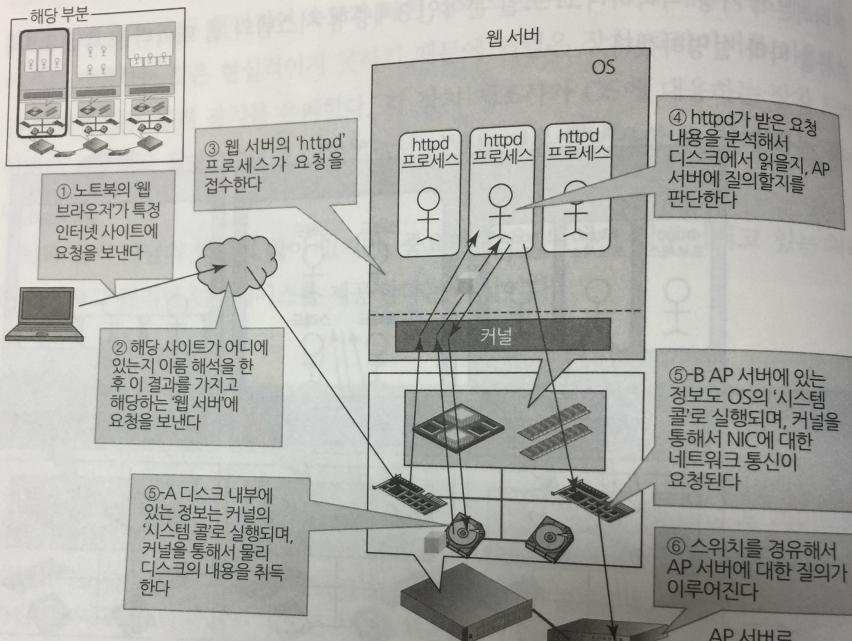


그림 3.9 클라이언트 PC로부터 웹 서버까지의 데이터 흐름

그림 3.9에는 클라이언트 PC에서 웹 브라우저를 실행해서 웹 서버에 요청을 보내고 AP 서버에 질의하기까지의 흐름을 보여 주고 있다. 각 단계에서 어떤 일이 발생하고 있는지 자세히 보도록 하자.

전체 흐름은 다음과 같다.

1. 웹 브라우저가 요청을 발행한다
2. 이름 해석을 한다

먼저, 인터넷에 접속돼 있는 환경에서 웹 브라우저를 실행해 본다.

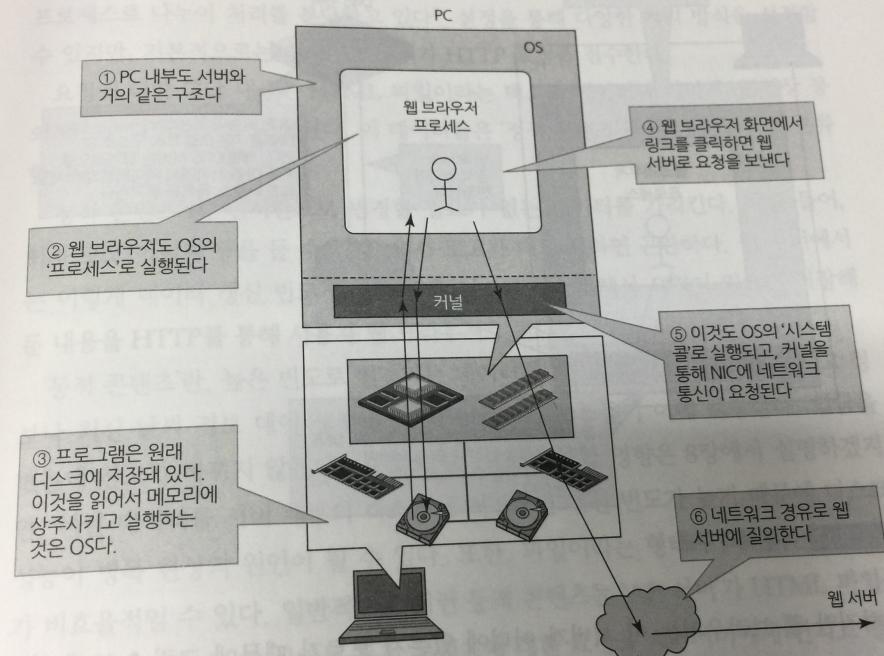


그림 3.10 클라이언트 PC에서 웹 브라우저 실행

그림 3.10은 서버가 아닌 PC에서의 처리 흐름이다. 디스크에서 프로그램을 읽어서 프로세스를 시작하고, 메모리 공간을 확보한다. 이 흐름은 PC와 서버에서 기본으로 같은 동작이다. 앞 절에서 소개한 시스템 콜이 이용되고 있음에 주목하자.

# Ch.3 3계층형 시스템을 살펴보자

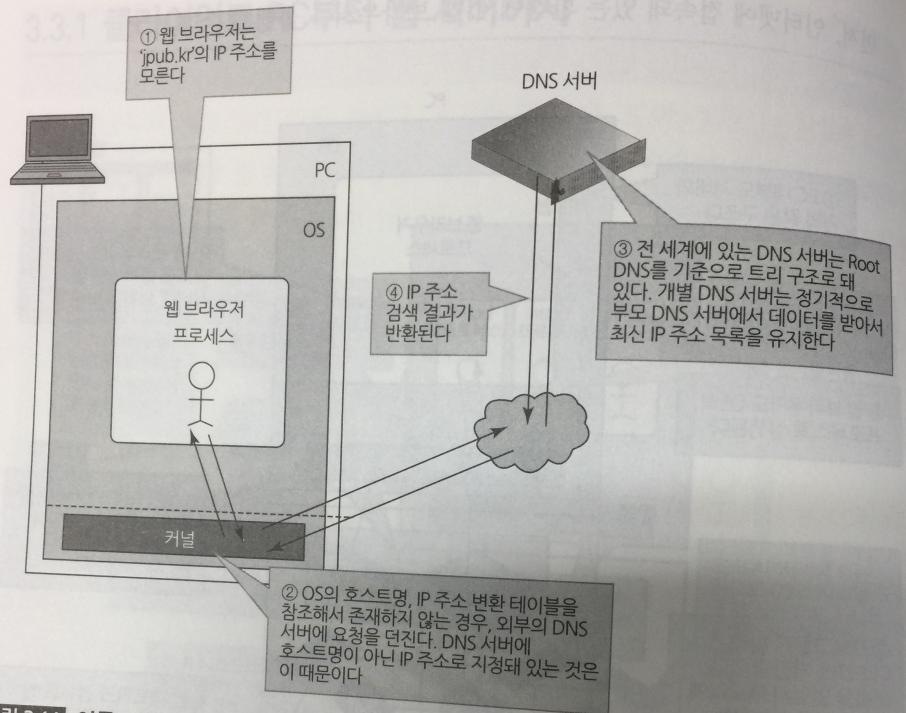


그림 3.11 이름 해석 구조

하지만 웹 브라우저는 이 서버가 어디에 있는지 모르기 때문에 그림 3.11에 있는 것처럼 조사를 하기 시작한다. 이 구조를 ‘이름 해석(name resolution)’이라고 한다. 이런 구조가 필요한 걸까? 인터넷 상의 주소는 ‘IP’라는 숫자로 표현돼 있어서 문어열인 URL과 IP를 연결시키지 않으면 통신이 되지 않기 때문이다. 이런 네트워크 이제 웹 서버까지 도착했다. 웹 서버의 역할은 HTTP 요청에 대해 적절한 파일이 콘텐츠를 반환하는 것이다. HTTP는 ‘HyperText Transfer Protocol’이다. 트콜을 가리킨다. 프로토콜이라는 용어는 원래 어떤 규칙이나 표준을 말하는 용어였다. 예전에는 전화선을 통해 데이터를 전송하는 표준이나, 전자우편을 전송하는 표준 등이 있다. 웹 서버는 이런 표준에 맞춰 데이터를 전송하는 역할을 한다.

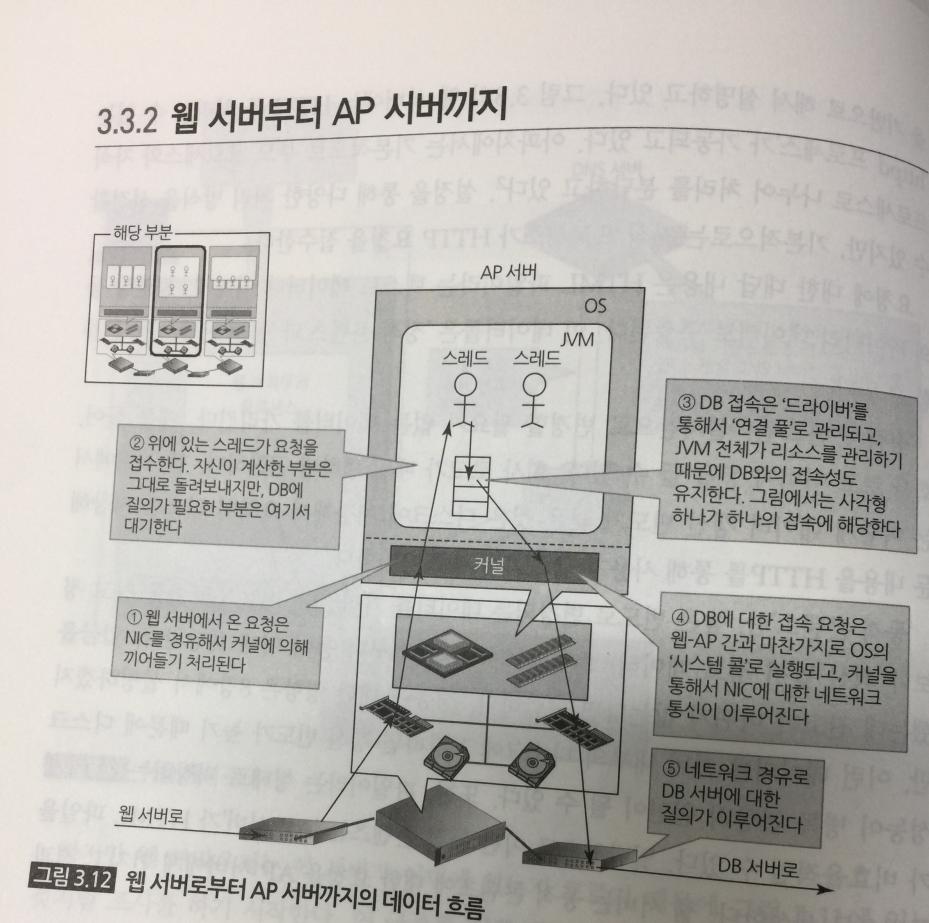


그림 3.12 웹 서버로부터 AP 서버까지의 데이터 흐름

‘동적 콘텐츠’에 대한 요청을 처리하는 것이 AP 서버다. 그림 3.12를 통해 구체적인 처리 내용을 살펴보자.

1. 웹 서버로부터 요청이 도착한다
2. 스레드가 요청을 받으면 자신이 계산할 수 있는지, 아니면 DB 접속이 필요한지를 판단한다
3. DB 접속이 필요하면 연결 풀에 액세스한다

# Ch.3 3계층형 시스템을 살펴보자

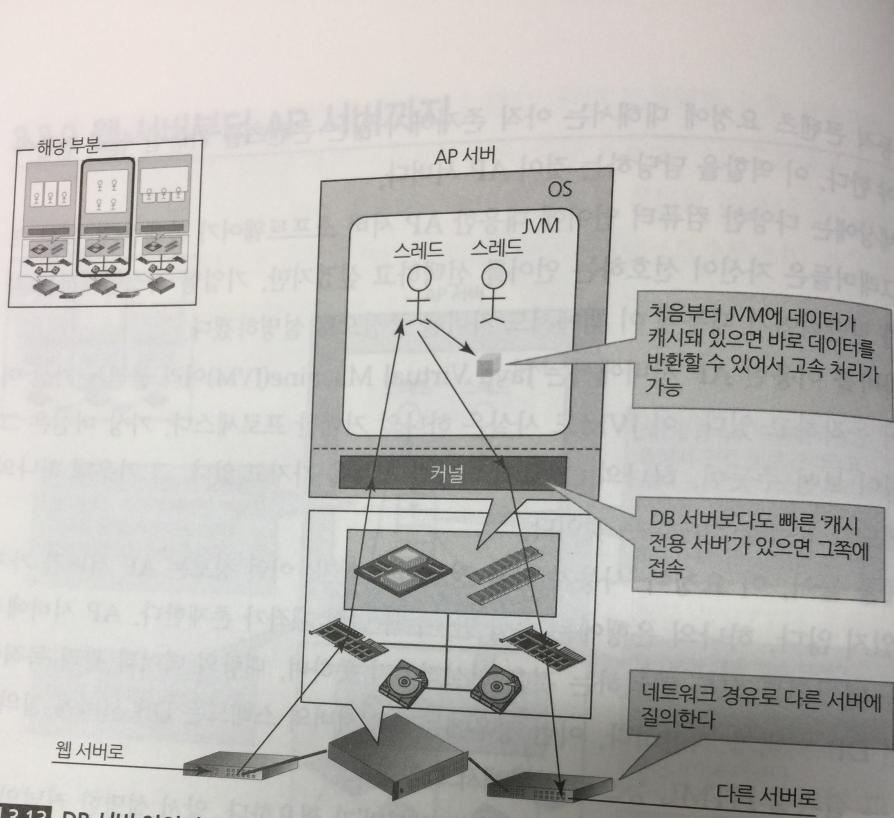


그림 3.13 DB 서버 이외의 옵션

데이터가 필요하면 DB 서버에 접속하는 것이 일반적이지만, 이것이 항상 효율적이라고 할 수는 없다. 예를 들어, '대한민국 행정 경계 정보'는 자주 바뀌는 것이 아니기 때문에 이것을 매번 데이터베이스에 질의할 필요는 없다. 이렇게 규모가 작고 반복성이 낮은 정보는 그림 3.13에 있는 것처럼 JVM 내부에 캐시로 저장해 두었다. 반대로, 규모가 큰 정적 데이터 전송 시에는 DB 서버 이외에 CDN(Content Delivery Network)이라 불리는 데이터 전송 전용 서버를 이용하

나. CDN은 대량의 데이터 전송에 특화된 것으로, 전 세계에 있는 데이터 복사본(캐시)을 배치하는 기술과 병렬 기술을 활용해서 처리를 효율화하고 있다. 특징이 '하나의 시스템에 대한 사용자 수가 제한돼 있다'. '침조뿐만 아니라 데이터를 생산하는 업무가 많다'와 같은 이유 때문이다.

## 3.3.3 AP 서버부터 DB 서버까지

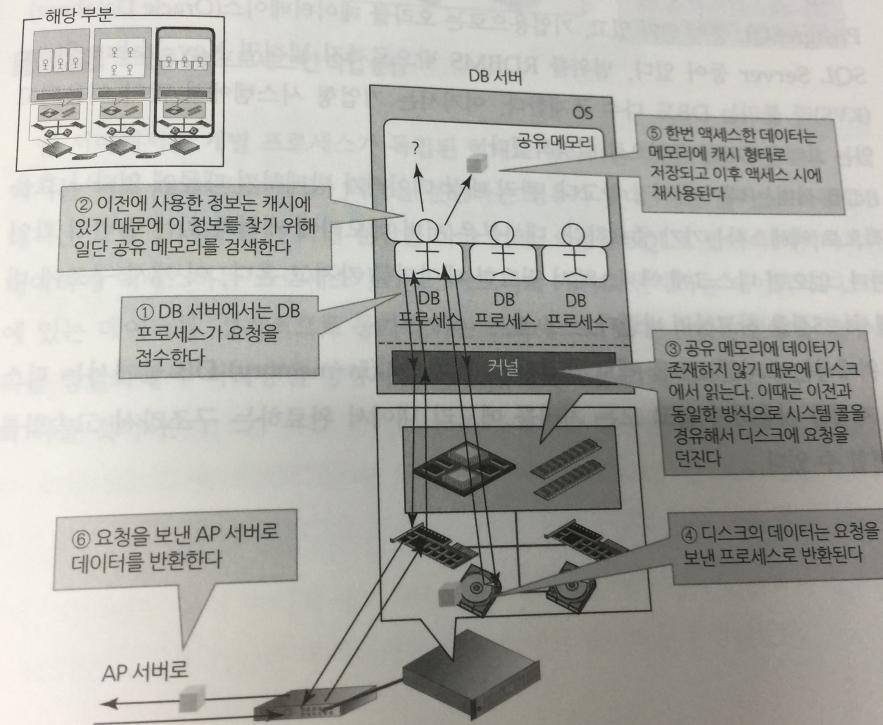


그림 3.14 AP 서버로부터 DB 서버까지의 데이터 흐름

3.3 웹 데이터 흐름

# Ch.3 3계층형 시스템을 살펴보자

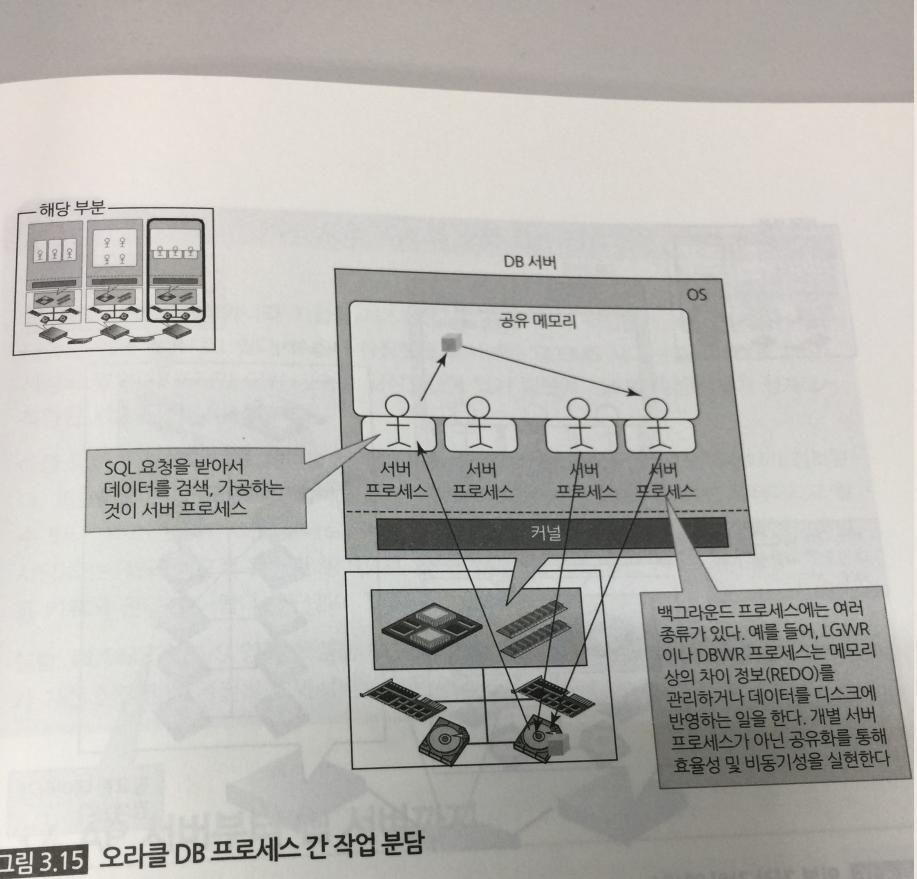


그림 3.15 오라클 DB 프로세스 간 작업 분담

웹 서버에서는 개별 프로세스가 독립된 형태로 동작하는 것을 보았지만, DB에서는 여러 개의 프로세스가 역할을 분담하는 경우가 있다. 예를 들어, 오라클에서는 그림 3.15와 같이 작업 분담을 하고 있다. 요청을 SQL로 받아서 해석하는 데이터에 액세스하는 프로세스가 있고, 메모리에 캐시로 존재하는 데이터와 디스크에 있는 데이터를 '정기적으로 동기화'하는 프로세스도 있다. 이것은 분업을 통한 병렬화해서 '처리량'을 향상시킬 수 있기 때문이다. 이에 대해서는 8장에서

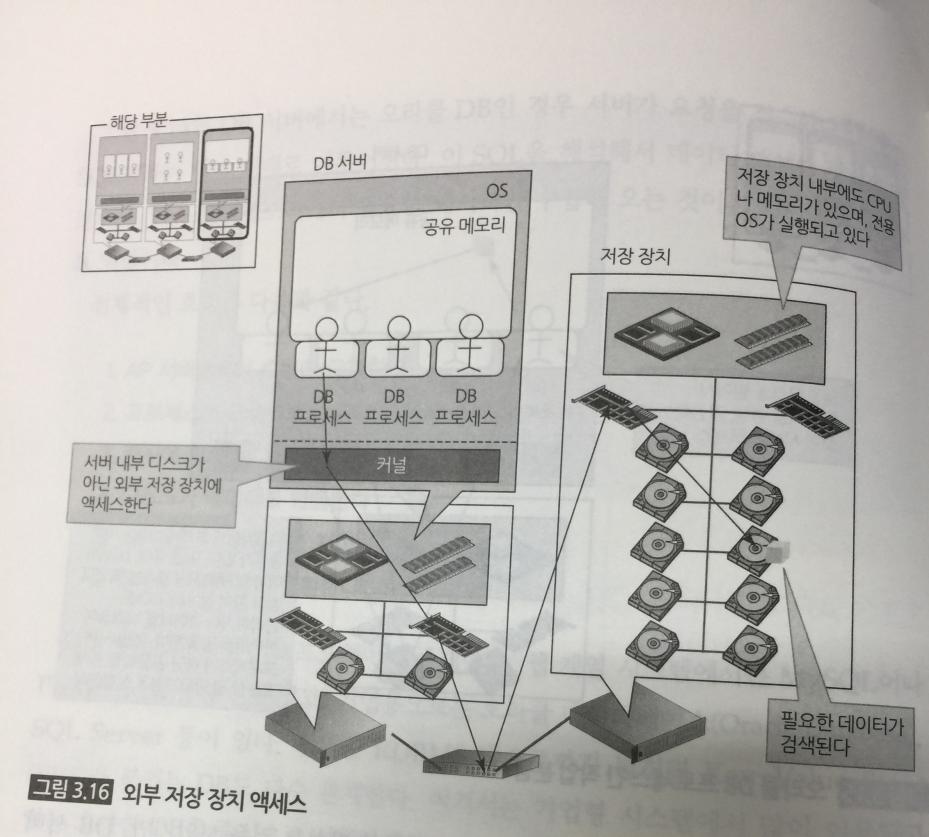
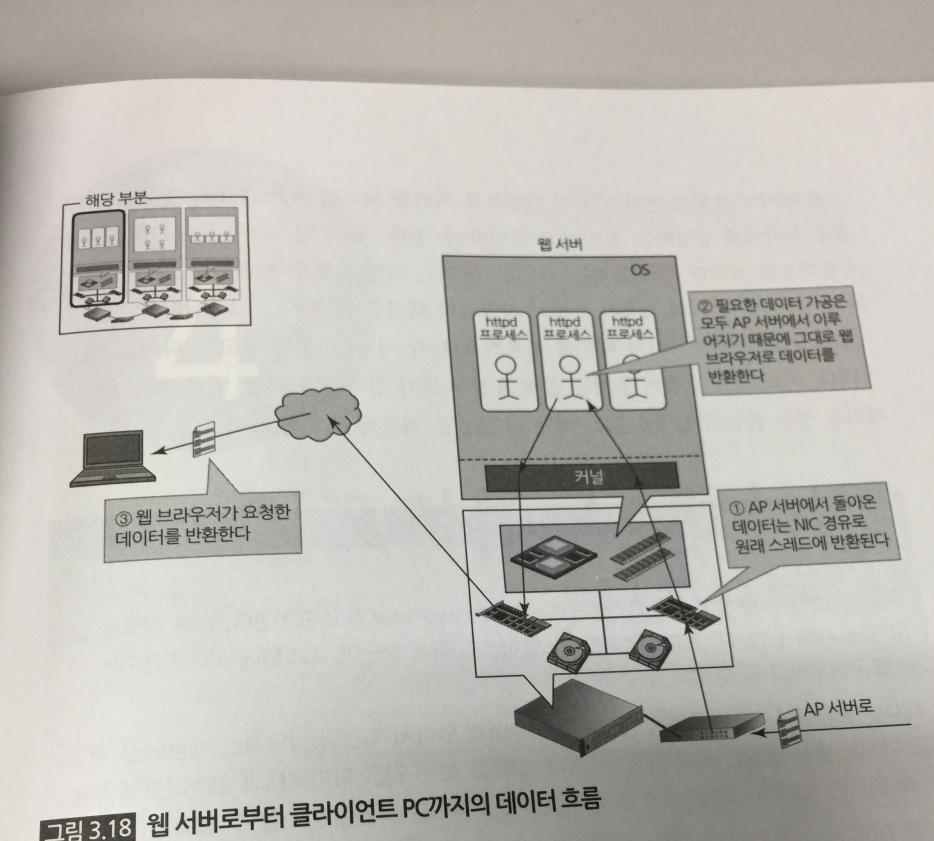
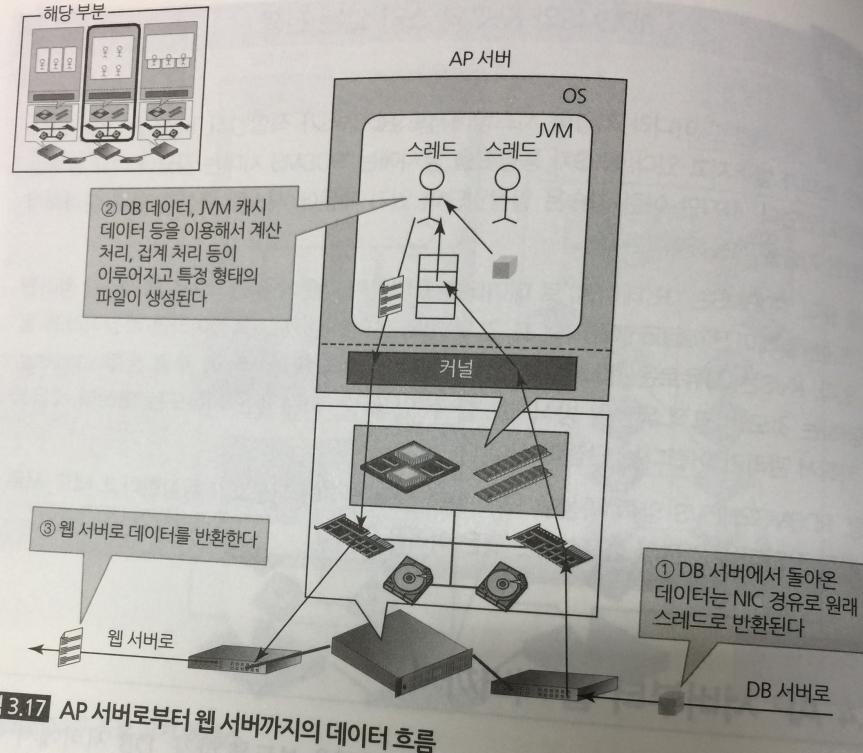


그림 3.16 외부 저장 장치 액세스

앞서 설명한 그림들에서는 DB 서버의 디스크 액세스 부분이 간략화되어 있어서 많은 기업형 시스템의 설정이 잘 반영돼 있지 않았다. 실제로는 DB 서버 내부의 디스크는 이중화 관점에서 뒤떨어지기 때문에 직접 사용하는 경우는 드물다. 대부분은 그림 3.16에 있는 것처럼 별도 저장 장치를 이용한다. 저장 장치에는 다수의 디스크가 설치돼 있다. 하지만 본질적인 구조는 지금까지 통장한 웹 서버, AP 서버, DB 서버와 큰 차이가 없다. CPU나 메모리가 있고 OS가 작하고 있는 구조다. 메모리에는 데이터를 캐시 형태로 저장하는 구조도 있다. 대의 데이터에 고속 액세스하기 위한 전용 서버라고 생각하면 좋다. 예전에는 대체로

# Ch.3 3계층형 시스템을 살펴보자



긴 여정이었지만, 요청 결과가 웹 브라우저까지 반환돼 왔다. 구체적으로는 하나의 요청에 하나의 데이터가 반환된다. 일반적인 웹 페이지에는 페이지 HTML 파일과 다수의 이미지 파일 등이 있기 때문에 복수의 요청으로 할돼서 웹 서버에 도착하고, 각 요청별로 데이터를 반환한다.

## 3.3.5 웹 서버부터 클라이언트 PC까지

AP 서버에서 돌아온 데이터를 받아서 웹 서버의 httpd 프로세스가 PC의 웹 브라우저로 반환한다(그림 3.18). 전체 흐름은 다음과 같다.

1. AP 서버로부터 데이터가 도착한다
2. 프로세스는 받은 데이터를

## 3.3.6 데이터 흐름 정리