

클라우드 애플리케이션 개발 (Cloud Application Development)

가상환경에서의 Web Application 개발 환경 구성

Wonjo Yoo
Principal Sales Consultant
Cloud Infrastructure / aPaaS
Oracle Korea



Oracle Korea

유 원 조

Principal Sales Consultant

경력: 18 years

- Java 기반의 Web Application Server
- Cloud Native, AppDev 담당

LinkedIn : <https://www.linkedin.com/in/wonjo-yoo-b9109048/>

클라우드 애플리케이션 개발(Cloud App. Development)

- 클라우드 환경에서 애플리케이션 개발과 관리

- 강의 목표 : 클라우드 환경 및 애플리케이션 개발에 대한 이해

8 가상환경에서의 Web Application 개발 환경 구성	유원조	10/25	10/26
9 Container 애플리케이션 개발 개요 및 환경 구성 (Local Docker 환경 구성, Dockerfile 이해)	김영규	11/01	11/02
10 Single Container 애플리케이션 구성(Git, Docker hub를 이용한 빌드/배포 자동화)	김영규	11/08	11/09
11 Multi Container 애플리케이션 구성(WEB-WAS-DB로 구성된 Multi-tier 애플리케이션 환경 구성 및 개발)	유원조	11/15	11/16
12 Opensource 기반의 Multi Container 애플리케이션 구성(Spring/Tomcat/MySQL)	김영규	11/22	11/23
13 Git, Wercker를 이용한 Container 애플리케이션 구축(node.js-cassandra / node.js-MongoDB)	유원조	11/29	11/30
14 애플리케이션 개발 트렌드(Microservices, Polyglot)	이미남	12/06	12/07
15 기말고사		12/13	12/14

Software Is Eating the World

Every company needs to become a software company

Marc Andreessen

모든 기업들은 Software를 필요로 한다



Speed of Business Innovation, Enabled by Software

Modern software의 중심은 WEB

WWW = HTTP

OSI 7 Layer중 7계층 Application

HTTP(HyperText Transfer Protocol)

GET / POST / PUT / DELETE / HEAD / OPTIONS Method

Request – Response 방식 → Connection less

80포트

Header / Body

HTTP is essential

Using HTTP is the modern application's trend

기존 웹사이트의
서비스의 재활용

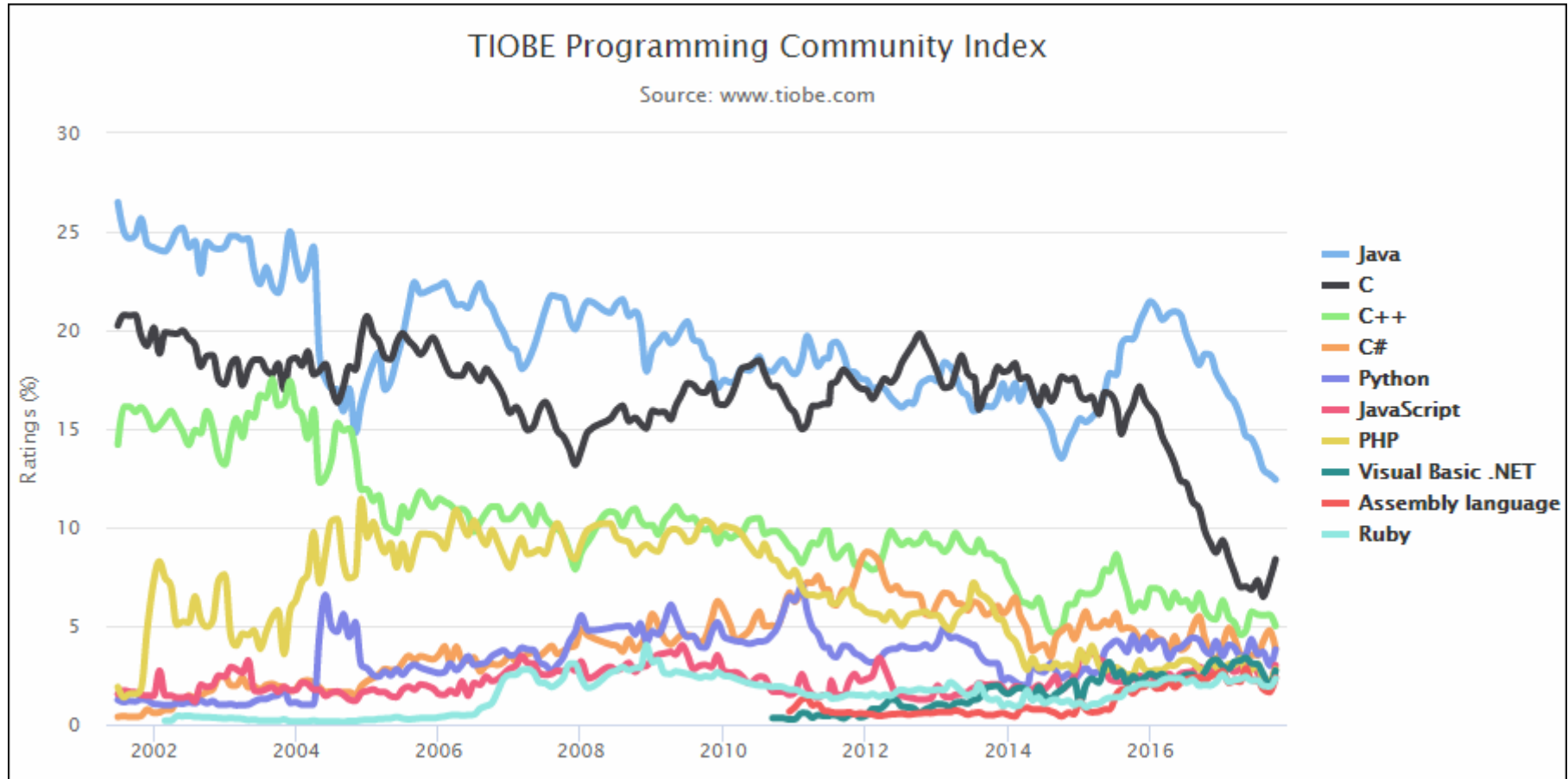
REST

Hybrid Mobile App

하드웨어의 발전

프로그래밍 언어 인기순위

Java is the most popular language.

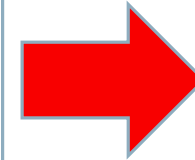


Web Application

Web application 개발하기 위해 필요한 것은?

- 직접 모든 것을 개발할 경우
브라우저가 해석할 수 있는 HTTP protocol 에 따른 처리
Request / Response (HTTP Header, HTTP Body)

동시 다중 사용자 처리 Thread Pool 관리
Database와의 연동
로그파일 관리



Application 개발

Web Application
Server

- WAS 사용



WAS (Web Application Server)

Java SE, EE



Java SE : JDK, Core

Java EE : 기업/서버

Java ME : 모바일 / 무선

Java SE, EE, ME

Java EE 주요기능

기능	J2EE
트랜잭션 관리	JTA
데이터베이스 접속	JDBC
웹 프레젠테이션	Java Servlet, JSP, JSF
비즈니스 로직 컴포넌트	EJB
비동기 메시징	JMS
시스템 통합	JCA
서버 운용 관리	JMX
메일 송수신	Java Mail
웹서비스	JAXP, JAX-RPC, JAX-WS
데이터 영속화, O/R매핑	JPA

Java EE specification

Java Standards	Version
Batch Application Processing (JSR 352)	1
Contexts and Dependency Injection for Java EE	1.1
Dependency Injection for Java EE	1
Concurrent Managed Objects (JSR 236)	1
Expression Language (EL)	3.0, 2.2, 2.1, 2.0
Java API for JSON Processing (JSR-353)	1
Java API for XML-Based Web Services (JAX-WS)	2.2, 2.1, 2.0
Java API for RESTful Web Services (JAX-RS)	2
Java API for WebSocket	1.1
JavaBeans Activation Framework	1.1
Java EE	7
Java EE Application Deployment	1.2
Java EE Bean Validation	1.1
Java EE Common Annotations	1.2
Java EE Connector Architecture	1.7
Java EE EJB	3.2, 3.1, 3.0, 2.1, 2.0, and 1.1
Java EE Enterprise Web Services	1.3, 1.2, 1.1
Java EE Interceptors	1.1
Java EE JDBC	4.0, 3.0
Java EE JMS	2.0, 1.1, 1.0.2b
Java EE JNDI	1.2
Java EE JSF	2.2, 2.1.*, 2.0, 1.2, 1.1
Java EE JSP	2.3, 2.2, 2.1, 2.0, 1.2, and 1.1
Java EE Managed Beans	1
Java EE Servlet	3.1, 3.0, 2.5, 2.4, 2.3, and 2.2
Java RMI	1
JavaMail	1.4
Java Transaction API	1.2
JAX-B	2.2, 2.1, 2.0
JAX-P	1.3, 1.2, 1.1
JAX-R	1
JAX-RPC	1.1
JDKs	8.0 (8.0 and 7.0 for clients)
JMX	2.0, 1.4
JPA	2.1, 2.0., 1.0

Java Standards	Version
JSR 77: Java EE Management	1.1
JSTL	1.2
Managed Beans	1
OTS/JTA	OTS 1.2 and JTA 1.2
RMI/IIOP	1
SOAP Attachments for Java (SAAJ)	1.3, 1.2
Streaming API for XML (StAX)	1
Web Services Metadata for the Java Platform	2.0, 1.1

Web Services Standards	Version
JSR 109: Implementing Enterprise Web Services	1.3
Java API for XML-based Web Services	2.2
Java Architecture for XML Binding	2.2
Web Services Description Language (WSDL)	1.1
Web Services Policy Framework (WS-Policy), WS-PolicyAttachment	1.5 and 1.2
Simple Object Access Protocol (SOAP)	1.2 and 1.1
SOAP with Attachments API for Java (SAAJ) 1.3	1.3
Web Services Security (WS-Security)	1.1 and 1.0
Web Services Security Policy (WS-SecurityPolicy)	1.3
Security Assertion Markup Language (SAML)	2.0 and 1.1
Security Assertion Markup Language (SAML) Token Profile	1.1 and 1.0
Web Services Addressing (WS-Addressing)	1.0 and 2004/08
Web Services Reliable Messaging (WS-ReliableMessaging), WS-RM Policy	1.2, 1.1
Web Services Trust Language (WS-Trust)	1.4 and 1.3
Web Services Secure Conversation Language (WS-SecureConversation)	1.4
WS-MakeConnection	1.1
Web Services Atomic Transaction, Web Services Coordination	1.2, 1.1, and 1.0

Other Standards	Version
X.509	v3
LDAP	v3
TLS	v1.1, v1.2
HTTP	1.1
SNMP	SNMPv1, SNMPv2, SNMPv3
xTensible Access Control Markup Language (XACML)	2.0
Partial implementation of Core and Hierarchical Role Based Access Control	2.0
Internet Protocol (IP)	V6, V4

Java EE 스펙 구현 인증

www.oracle.com/technetwork/java/javaee/overview/compatibility-jsp-136984.html

- Java ME
- Java SE Advanced & Suite
- Java Embedded
- Java DB
- Web Tier
- Java Card
- Java TV
- New to Java
- Community
- Java Magazine

Java EE Compatibility


Java EE Compatibility



In every industry, businesses face the challenge of accommodating ever greater demands for high-speed data access, diverse clients, and secure transactions without incurring extensive additional costs. To extend existing IT investments while meeting these demands, developers have consistently adopted the Java Platform, Enterprise Edition.

The de-facto standard for delivering secure, robust, scalable multi-platform applications and services, Java EE technology and its success is predicated on compatibility which brings Java technology's mission of "Write Once, Run Anywhere" capability to the server. Developers can write applications to the Java EE specification -- and companies can purchase such applications -- and be assured that they are portable across all the Java compatible, Enterprise Edition products available today.

Java EE 7 Full Platform Compatible Implementations




GlassFish Server Open Source Edition
4.0

Tested Configuration




TMAX JEUS 8

Tested Configuration




Wildfly 8.x

Tested Configuration



Cosminexus: Hitachi Application Server
v10.0

Tested Configuration



IBM WebSphere Application Server
Version 8.5.5.6(Liberty Profile)

Tested Configuration



ORACLE
FUSION MIDDLEWARE
WEBLOGIC SERVER

Oracle Weblogic Server 12.2.1

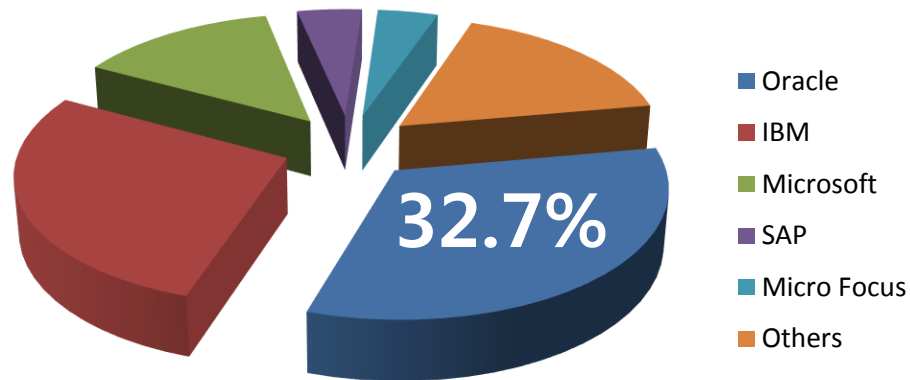
Tested Configuration

Java EE WAS

Popular Java EE middleware

- Oracle WebLogic, IBM WebSphere, Redhat JBoss, TmaxSoft JEUS

2015년 상반기 IDC Report :
Worldwide Application Server Market Share



ORACLE®

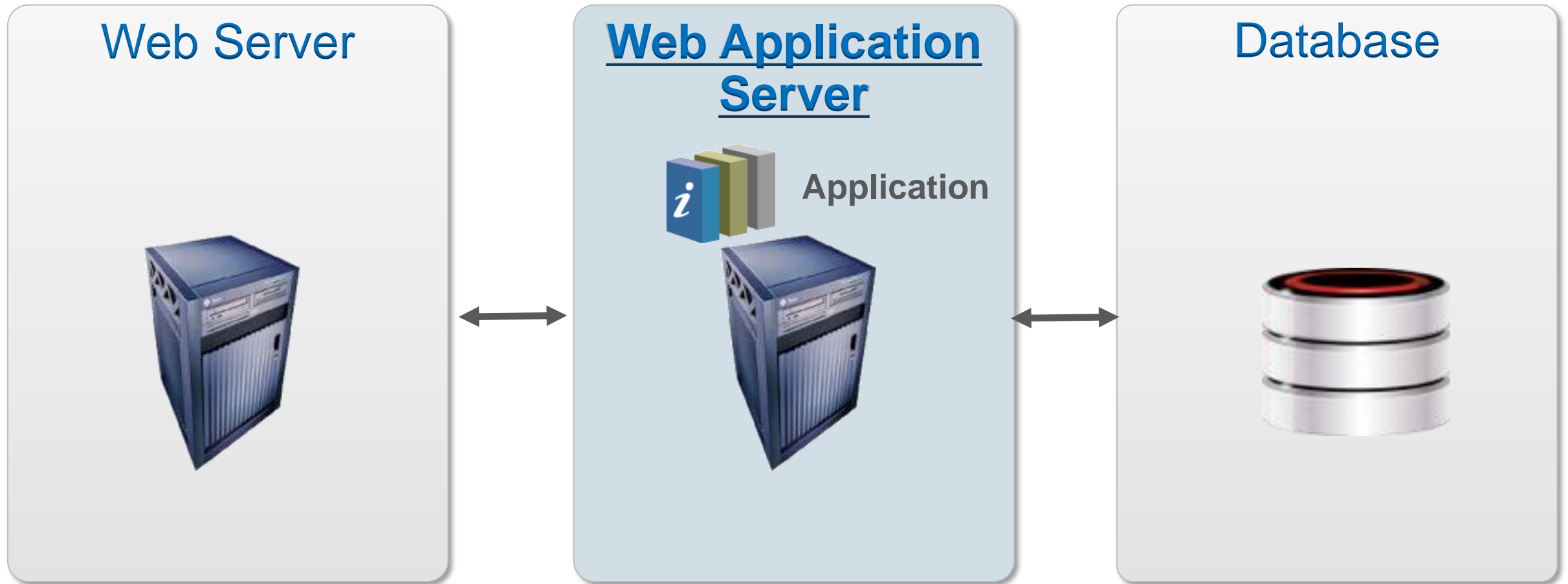
FUSION MIDDLEWARE
WEBLOGIC SERVER



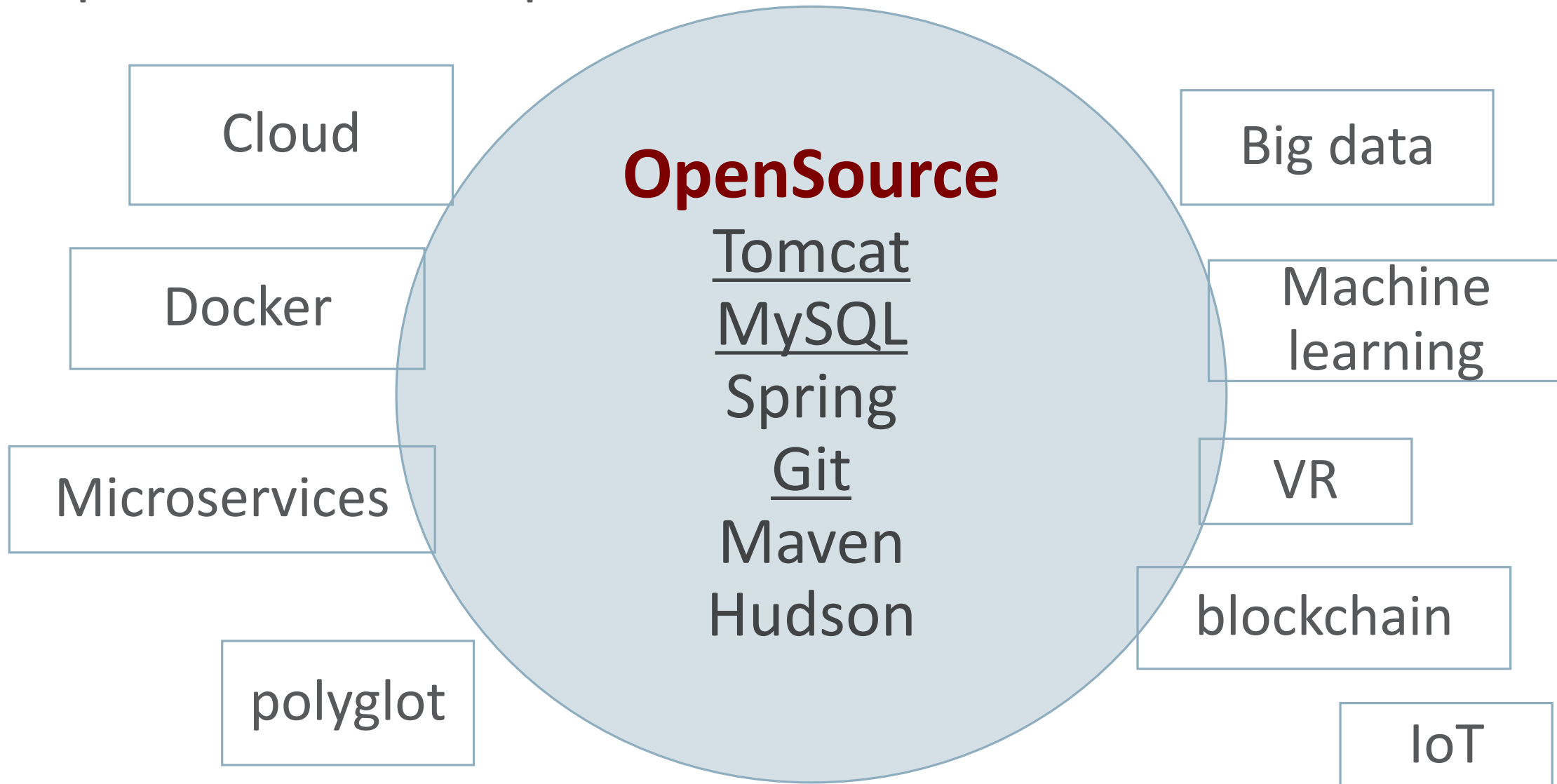
RED HAT® JBOSS®
ENTERPRISE
APPLICATION PLATFORM 7

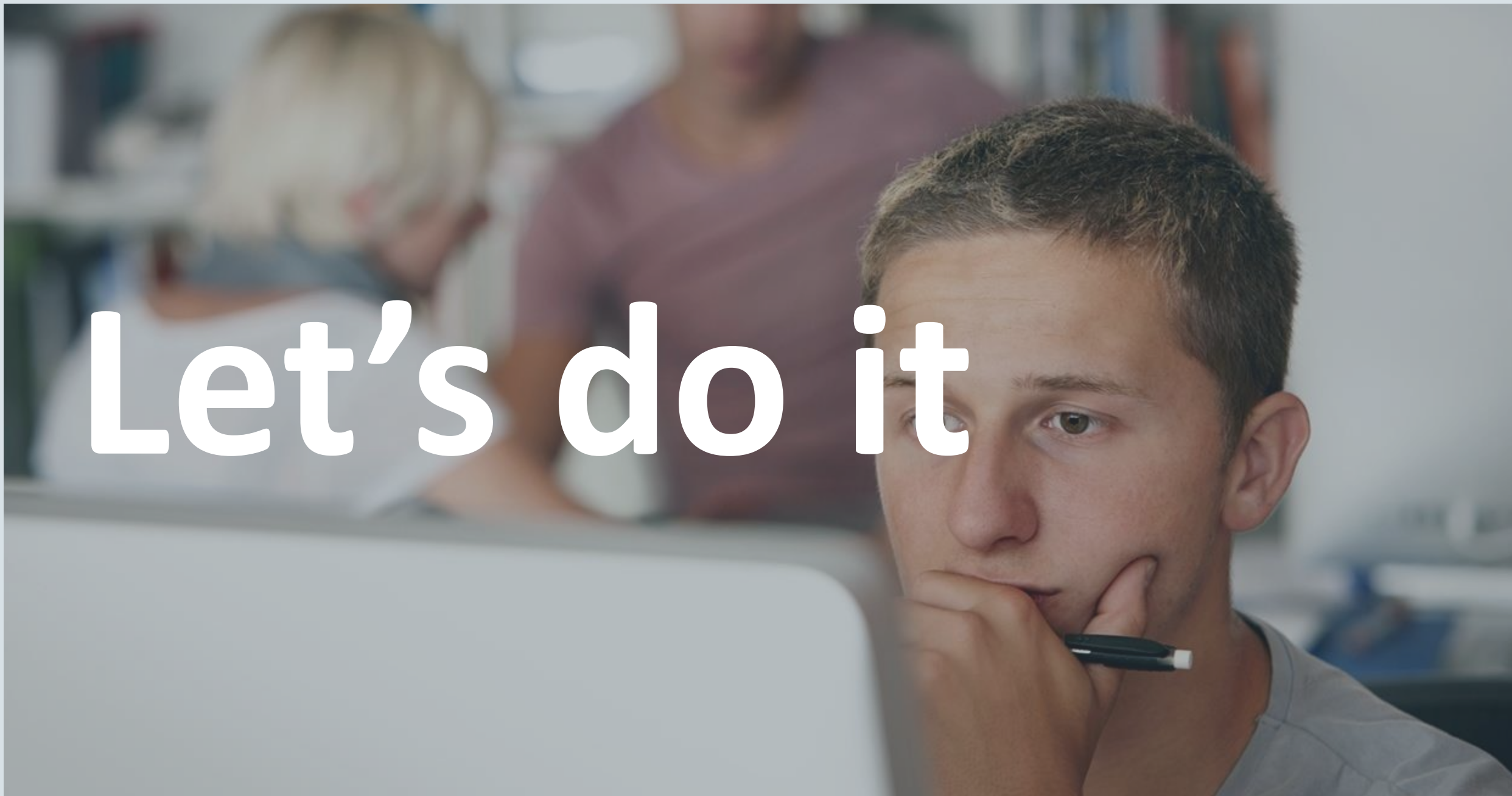
WAS (Web Application Server) 구성 Architecture

3 Tier Architecture



Open Source is important trends.



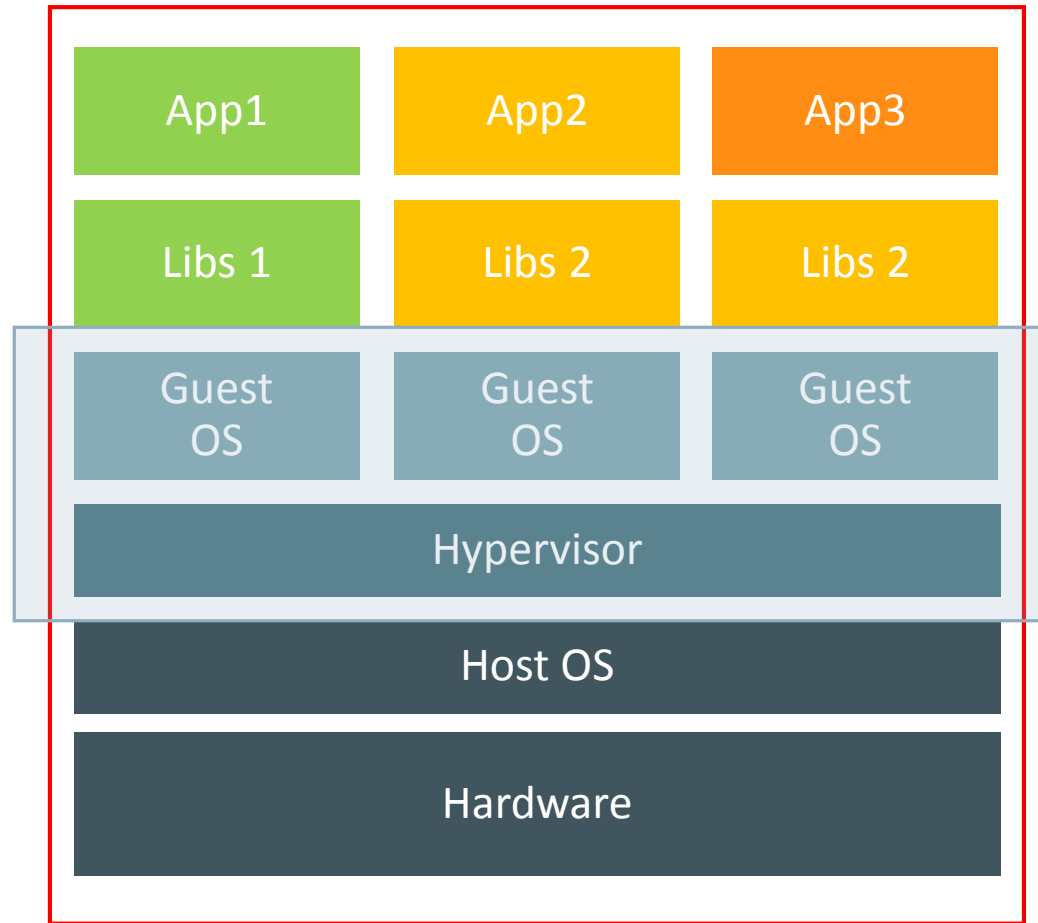


실습 Agenda

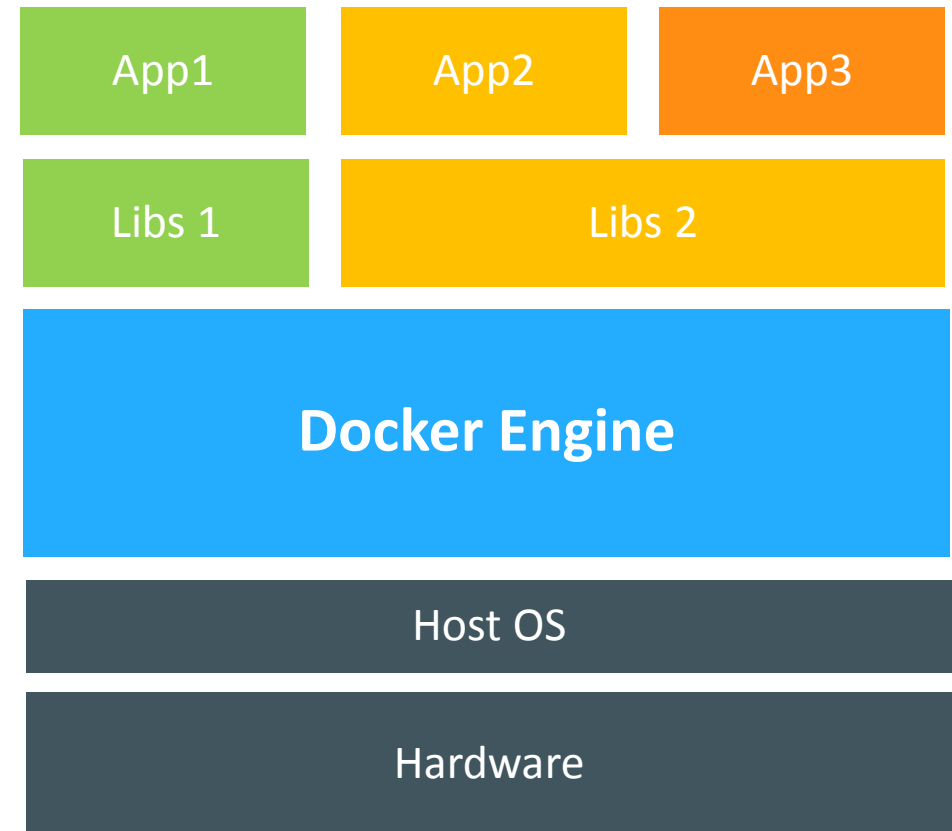
목표: 가상환경으로 일반적인 Web Application에 대한 개발 환경을 구성한다.

- VirtualBox를 구성하여 Linux가상환경 구성
- Java EE 기반 Web Application Server인 Tomcat 설치
- 데이터베이스 MySQL 설치
- 간단한 Web Application 배포 및 테스트
- Web Server Apache 설치
- Apache + Tomcat 연동 구성
- VM별로 Apache/Tomcat/MySQL 구성하기

Virtual Machines vs. Containers



Virtual Machine



Container

Lab 1-1 가상환경 구성 (20분)

Virtual Box 설치

- <https://www.virtualbox.org/>
 - x86 and AMD64/Intel64 기반의 가상화 기술
 - OpenSource
 - 오라클 Lead
-
- 1. VirtualBox 설치
 - 2. 가상 시스템 가져오기

Lab 1-2 WAS(Web Application Server) 설치 (15분)

Tomcat 설치 (<https://tomcat.apache.org/>)

- 1. JDK 설치
 - tar xvfz jdk-8u144-linux-x64.tar.gz
- 2. 환경변수 변경
 - vi .profile 실행 후 아래부분에 JAVA_HOME 추가하고 PATH 앞쪽 부분에 추가
set PATH so it includes user 's private bin directories
JAVA_HOME=/home/dku/jdk1.8.0_144
PATH=\$JAVA_HOME/bin:\$HOME/bin:\$HOME/.local/bin:\$PATH
 - ./profile로 적용 (. 다음에 공백 주의)
- 3. Tomcat 설치
 - tar xvfz apache-tomcat-8.5.23.tar.gz
 - cd apache-tomcat-8.5.23/bin
 - ./startup.sh
 - 브라우저에서 호출 http://localhost:8080/

Java EE Web Application

Web App 디렉토리 구조

- App Root 디렉토리 : jsp 파일
- 하위 디렉토리
 - WEB-INF : web.xml 이 있어야 함
 - WEB-INF/lib/
 - application에서 사용하는 클래스를 압축하여 jar 파일 형태로 library를 넣는 곳
 - WEB-INF/classes
 - application에서 사용하는 클래스를 넣는 곳 (servlet 클래스, 공용 클래스)
- 위 디렉토리를 war 파일로 압축하여 배포 가능

Tomcat App 배포

Web App 디렉토리 구조

- apache-tomcat-8.5.23/webapps/
- 위 webapps 경로에 web app 을 넣을 경우 자동으로 배포됨
- webapps에 web app 인 app1 이 있을 경우 context 는 app1 이 됨 `http://{tomcatip}/app1/` 으로 접속
- Context 는 application 단위를 의미하고, 여기에서 /app1은 context-path를 의미
- ROOT 디렉토리는 / context를 의미

Lab 1-3 간단한 Web Application 배포하기 (5분)

Tomcat에 Java EE Application 배포

- `cd ~/apache-tomcat-8.5.23/webapps`
- `mkdir test`
- test 폴더에 index.jsp 를 하나 만들고 브라우저에 확인
- <http://localhost:8080/test/index.jsp>

Lab 1-4 Simple Servlet 작성(10분)

간단한 Servlet 작성하기

- Servlet 3.0에서 @WebServlet 추가

```
cd ~/apache-tomcat-8.5.23/webapps/test
```

```
mkdir WEB-INF
```

```
cd WEB-INF
```

```
mkdir classes
```

```
cd classes
```

```
vi TServlet.java → 클래스 작성
```

```
javac -cp ./~/apache-tomcat-8.5.23/lib/servlet-api.jar TServlet.java
```

- <http://localhost:8080/test/simple>

Lab 1-5 데이터베이스 설치 (10분)

My-SQL 설치

- <https://dev.mysql.com/downloads/>
- **MySQL Community Server** (GPL)
- <https://dev.mysql.com/doc/mysql-apt-repo-quick-guide/en/>
- `cd /home/dku/install`
- `sudo dpkg -i mysql-apt-config_0.8.8-1_all.deb`
- `sudo apt-get update`
- `sudo apt-get install mysql-server`
 - 새로 설치하는 MySQL의 Root password 입력 (이후 계속 사용하므로 기억)

Lab 1-6 데이터베이스 툴 설치 (10분)

My-SQL 기동/종료, workbench

- `sudo service mysql status`
- `sudo service mysql stop`
- `sudo service mysql start`

My-SQL workbench 툴 설치

- `sudo apt-get install mysql-workbench-community`
- mysql-workbench 실행

Lab 1-7 데이터베이스 테이블 생성 (10분)

Schema 생성

- 왼쪽 하단 메뉴에서 오른쪽 마우스 클릭 -> Create Schema → dku로 생성
 - ('Apply' 버튼이 안보일 경우: VirtualBox에서 전제화면 모드 선택 필요)

테이블 만들기

- Schema선택 후 오른쪽 마우스 클릭 -> Create Table
- 테이블명 : test
컬럼 : ID varchar(10), name varchar(100) 으로 생성

임시 데이터 넣기

- File->New Query Tab 선택하여 창을 연 후 아래 쿼리 실행(번개모양)
use dku;
insert into test values('test1', 'this is test data');

JDBC (Java Database Connectivity)

4가지 JDBC Driver의 Types

- Type 1: JDBC-ODBC bridge
- Type 2: Java driver → Native DB Driver
(예: Oracle OCI)
- Type 3: Net protocol Driver
- Type 4: 100% 순수 Java Driver

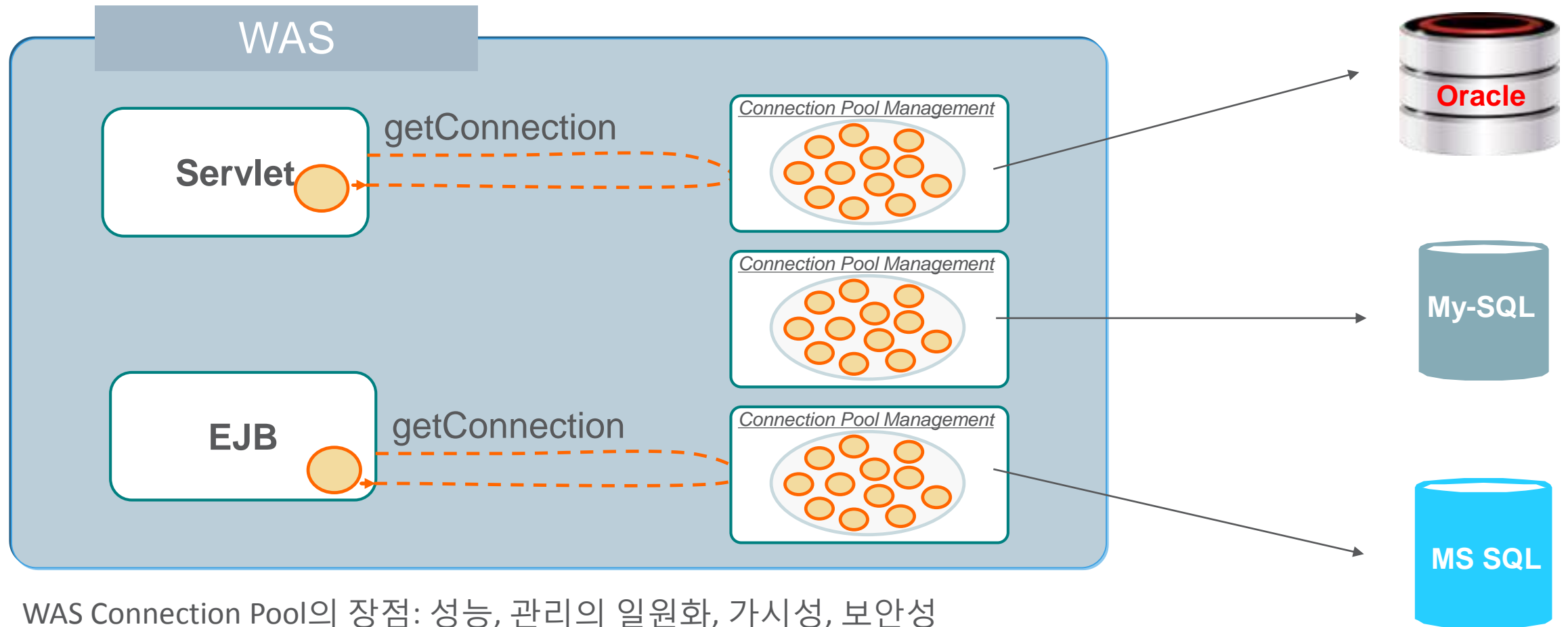
Lab 1-8 Web Application에서 데이터 조회하기 (5분)

Tomcat - MySQL연동 구성

- JDBC Driver 파일 library에 복사하기
- <https://dev.mysql.com/downloads/connector/j/> 에서 다운로드
- Tomcat의 webapp에 jdbc driver 복사하기

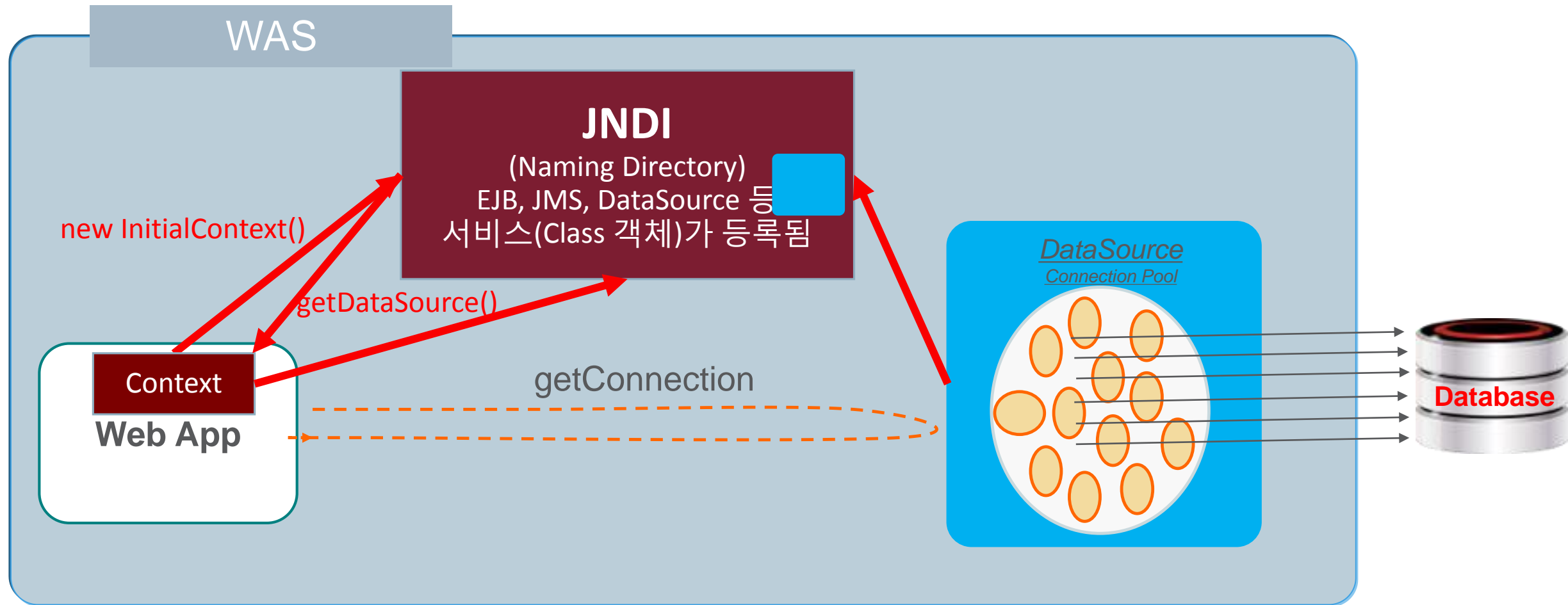
```
cd /home/dku/install/mysql-connector-java-5.1.44  
cp mysql-connector-java-5.1.44-bin.jar ~/apache-tomcat-8.5.23/webapps/test/WEB-INF/lib/
```
- dku 계정 home에 있는 db.jsp 파일 복사해서 실행하기
- <http://localhost:8080/test/db.jsp>

JDBC Connection Pool



WAS Connection Pool의 장점: 성능, 관리의 일원화, 가시성, 보안성

JDBC Connection Pool



Lab 1-9 JDBC Connection Pool 사용하기 (15분)

Tomcat - MySQL 연동 구성

1. jdbc파일 복사 -> ~/apache-tomcat-8.5.23/lib

```
cp mysql-connector-java-5.1.44-bin.jar ~/apache-tomcat-8.5.23/lib/
```

2. apache-tomcat-8.5.23/conf/context.xml 에 추가

```
<Resource name="jdbc/TestDB" auth="Container" type="javax.sql.DataSource"  
    maxTotal="100" maxIdle="30" maxWaitMillis="10000"  
    username="root" password="welcome1" driverClassName="com.mysql.jdbc.Driver"  
    url="jdbc:mysql://localhost:3306/dku"/>
```

3. Web App 에서 reference하도록 web.xml 에 추가

```
<resource-ref>  
    <res-ref-name>jdbc/TestDB</res-ref-name>  
    <res-type>javax.sql.DataSource</res-type>  
    <res-auth>Container</res-auth>  
</resource-ref>
```

Lab 1-9 JDBC Connection Pool 사용하기 (15분)

Tomcat - MySQL 연동 구성

1. 소스에서 수정

```
cp db.jsp db2.jsp
```

2. Connection 얻는 부분 소스에서 수정

```
Context context = new InitialContext();
```

```
DataSource ds = (DataSource) context.lookup(" java:/comp/env/jdbc/TestDB");
```

```
Connection conn = ds.getConnection();
```

2. Tomcat 재기동

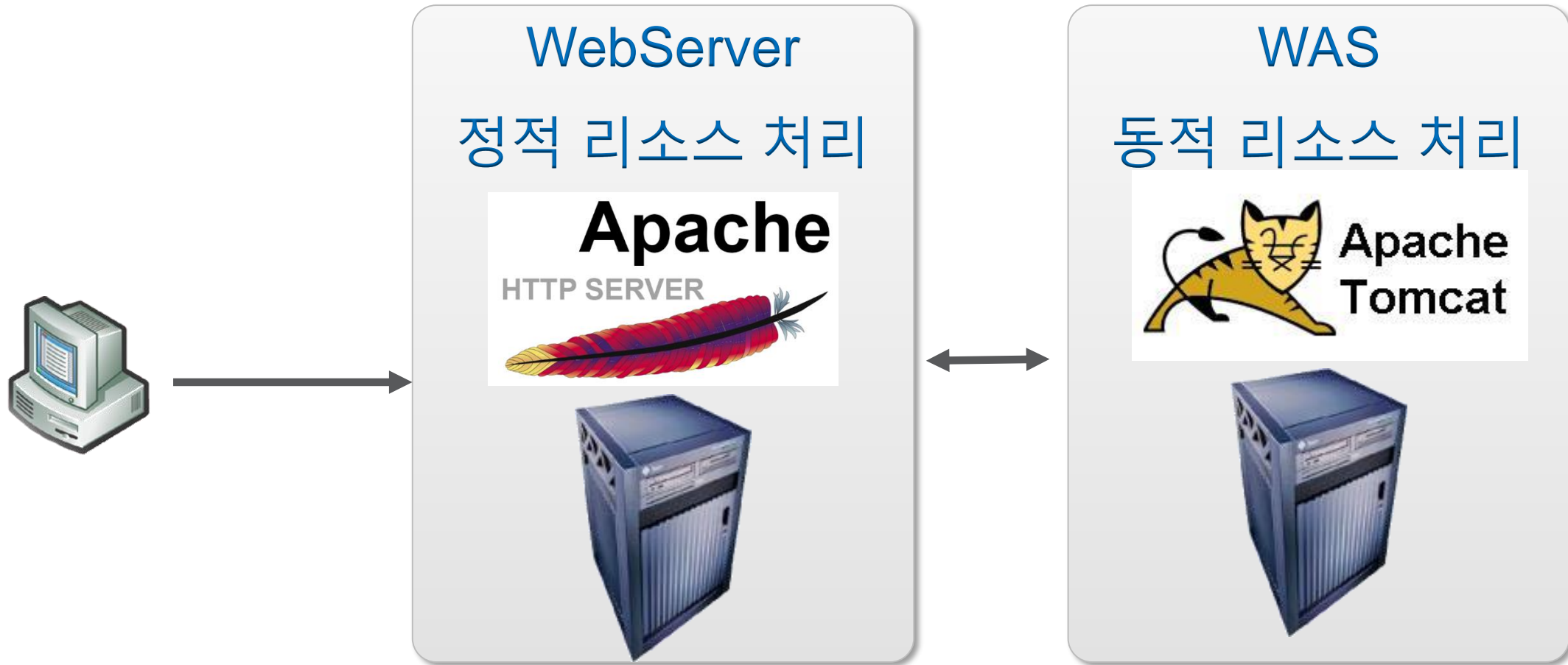
```
cd ~/apache-tomcat-8.5.23/bin
```

```
./shutdown.sh → ./startup.sh
```

3. 브라우저에서 Test <http://localhost:8080/db2.jsp>

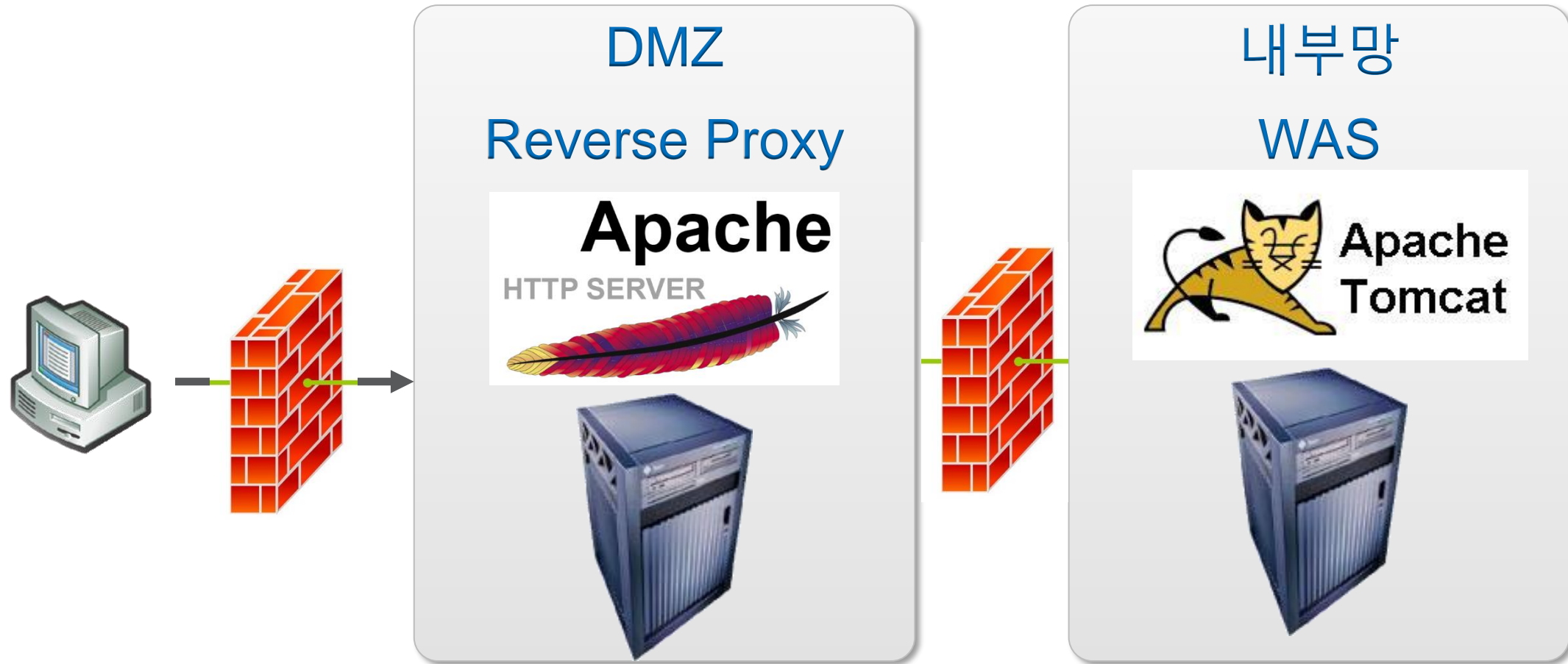
WebServer 와 WAS

업무 분담을 통한 성능 분산



WebServer 와 WAS

Security



Lab 1-10 웹서버 설치 (5분)

Apache 설치

- `sudo apt-get install apache2`
- 브라우저에서 확인 <http://localhost/>
- 설정 파일 위치 `/etc/apache2.conf`
 - `sudo service apache2 stop`
 - `sudo service apache2 start`
- Document Root : `/var/www/html`

Lab 1-11 Apache + Tomcat 연동하기 (10분)

mod_proxy

- 설정파일 복사 : `cd /etc/apache2/mods-available`
`sudo cp proxy.conf ../mods-enabled/`
`sudo cp proxy.load ../mods-enabled/`
`sudo cp proxy_http.load ../mods-enabled/`
- 연동설정 : `sudo vi /etc/apache2/mods-enabled/proxy.conf`
`ProxyPass / http://localhost:8080/`
`ProxyPassReverse / http://localhost:8080/`
- 재시작 `sudo service apache2 restart`
- 브라우저에서 확인 <http://localhost/test/>

Lab 1-12 VM 별로 구동하기 (30분)

VM 별로 Apache, Tomcat, MySQL 별도로 구동

1. VM 3개로 복사
2. 방화벽 해제
3. MySQL VM 에서 모든 ip에 대해 접속 권한 변경

use mysql;

```
GRANT ALL PRIVILEGES ON *.* TO 'root' @ '%' IDENTIFIED BY 'welcome1' ;
```

4. MySQL Listen 주소 설정 변경

```
sudo vi /etc/mysql/mysql.conf.d/mysqld.conf
```

Tomcat쪽 app (db.jsp)에서 DB 접속 IP 변경

5. Apache Tomcat 연동 IP 변경

Lab 1-13 부하테스트 (추가 실습)

Jmeter를 활용한 부하테스트

1. JMeter 설치 및 설정

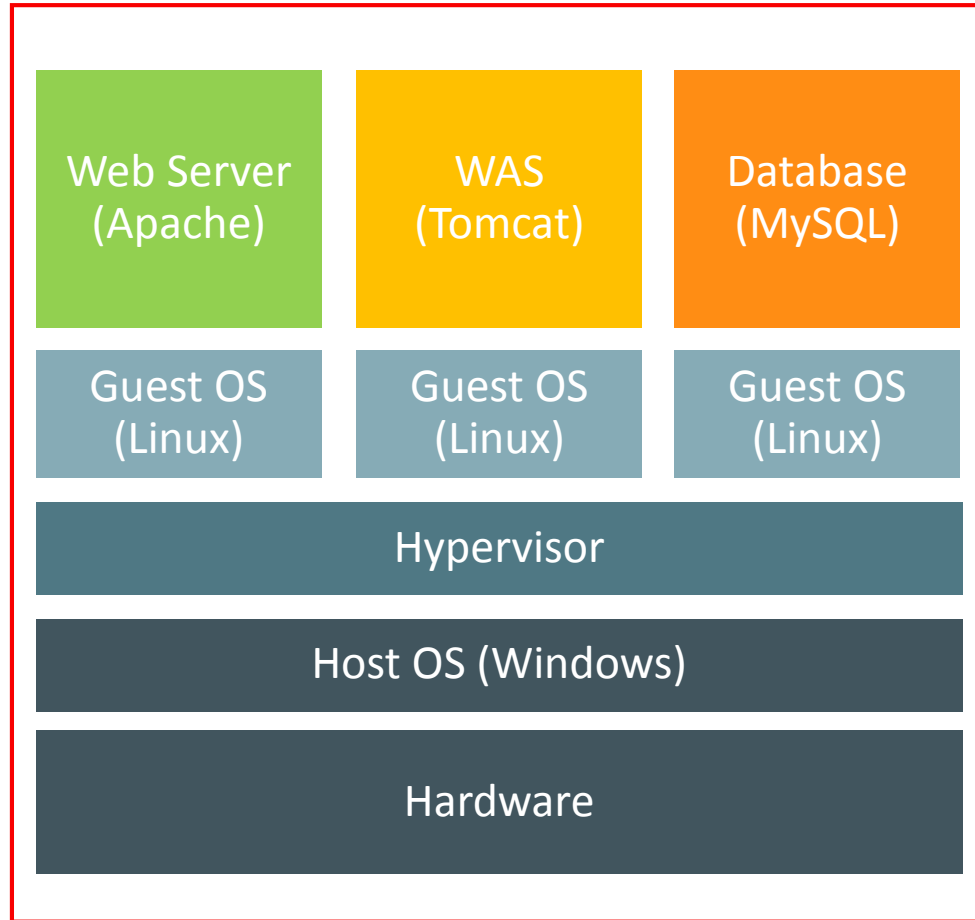
- sudo apt-get install jmeter
- Jmeter 실행 후 아래 생성
- Plan생성 : Test Plan 오른쪽 마우스 → Add → Threads → Thread Group
 - Number of Threads : 10, Loop Count : 10
- ThreadGroup 오른쪽 마우스 → Add → Sampler → Http Request
 - Name : db Server Name : localhost Path: /test/db.jsp
- ThreadGroup 오른쪽 마우스 → Add → Sampler → Http Request
 - Name : db_pool Server Name : localhost Path: /test/db2.jsp
- ThreadGroup 오른쪽 마우스 → Add → Listener → Summary Report

2. 부하 실행 (Ctrl + R)

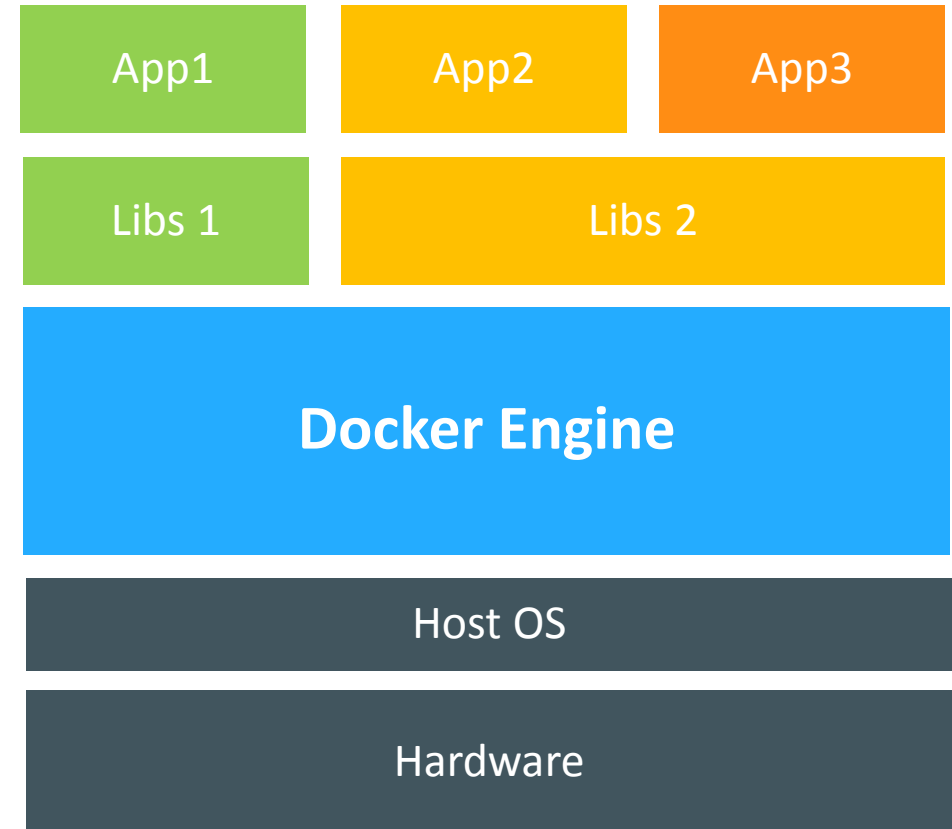


Wrap Up

Wrap Up



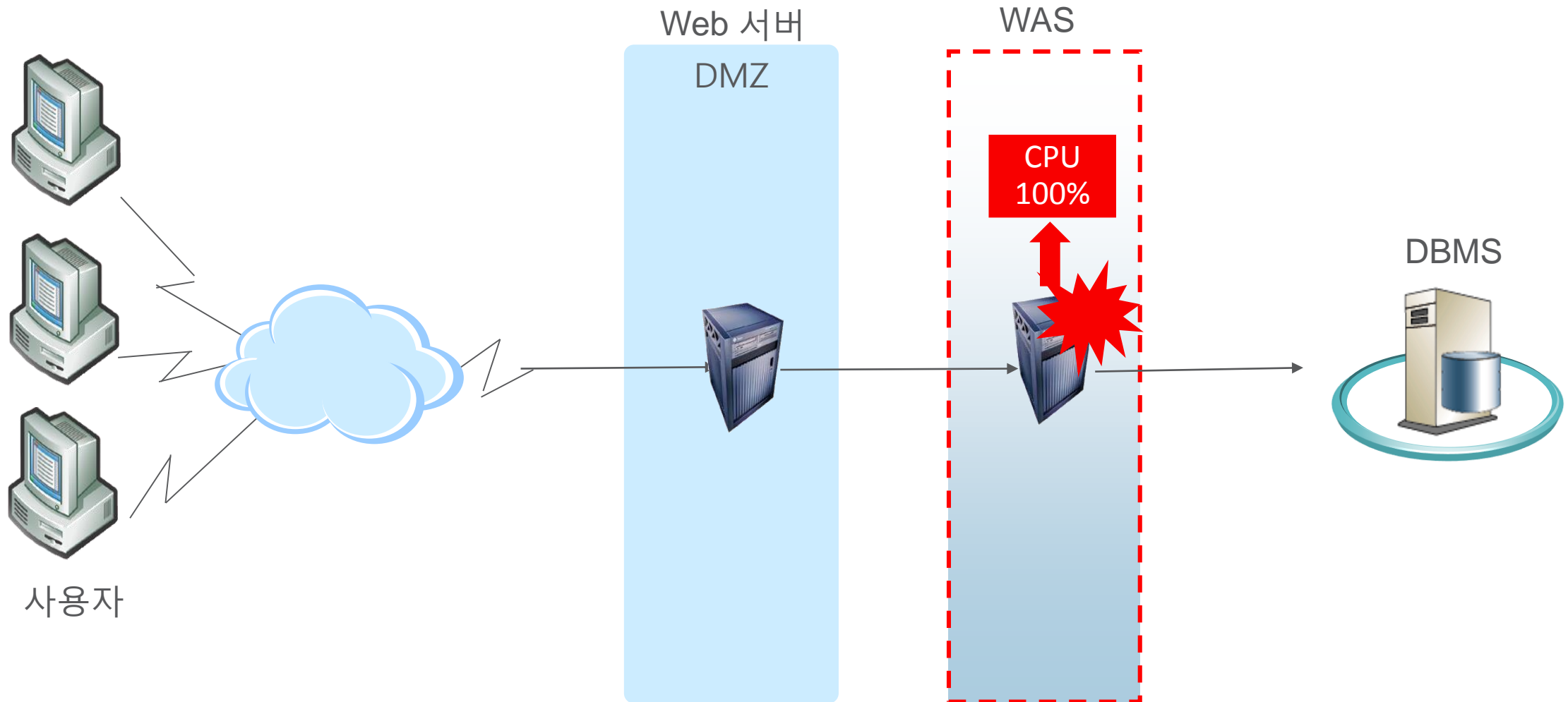
Virtual Machine



Container

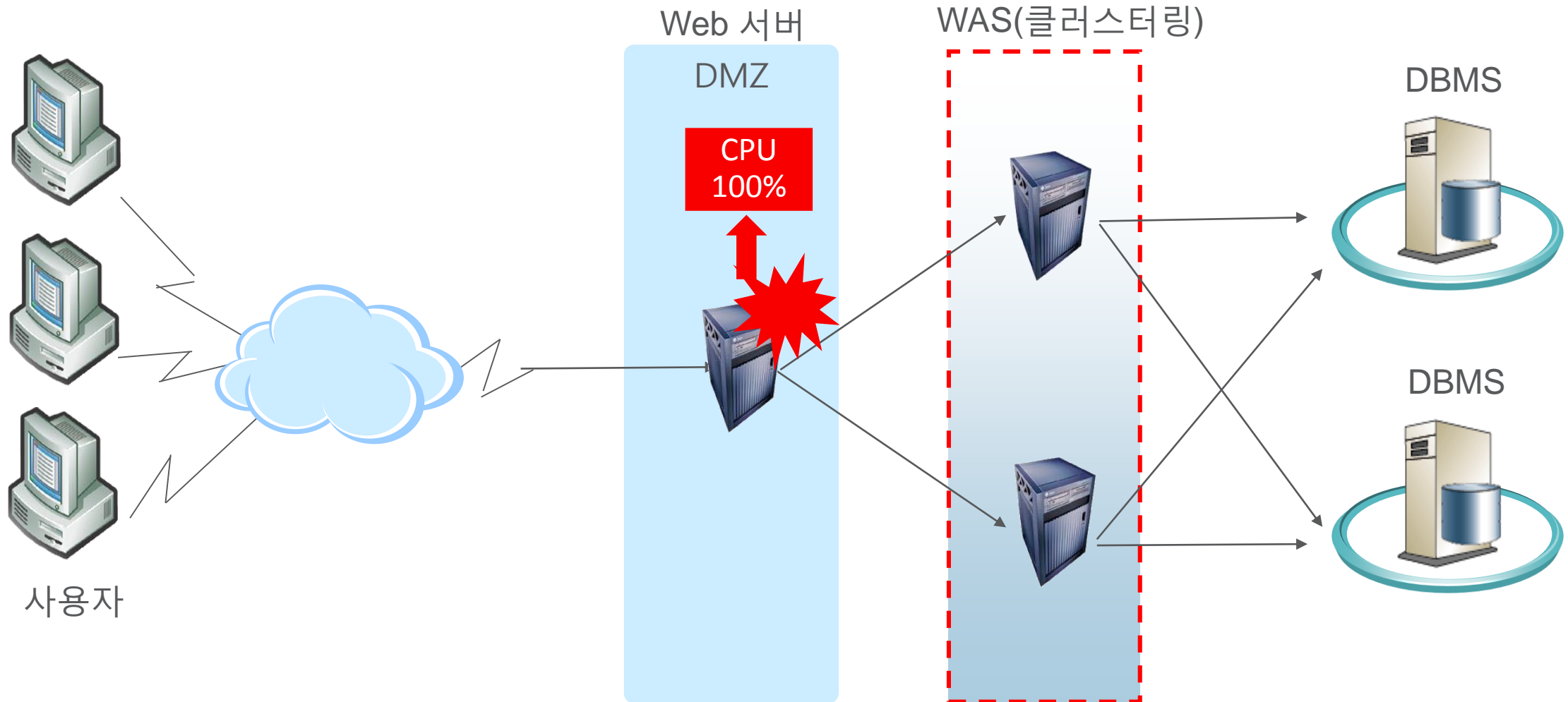
WAS (Web Application Server) 구성 Architecture

3 Tier Architecture



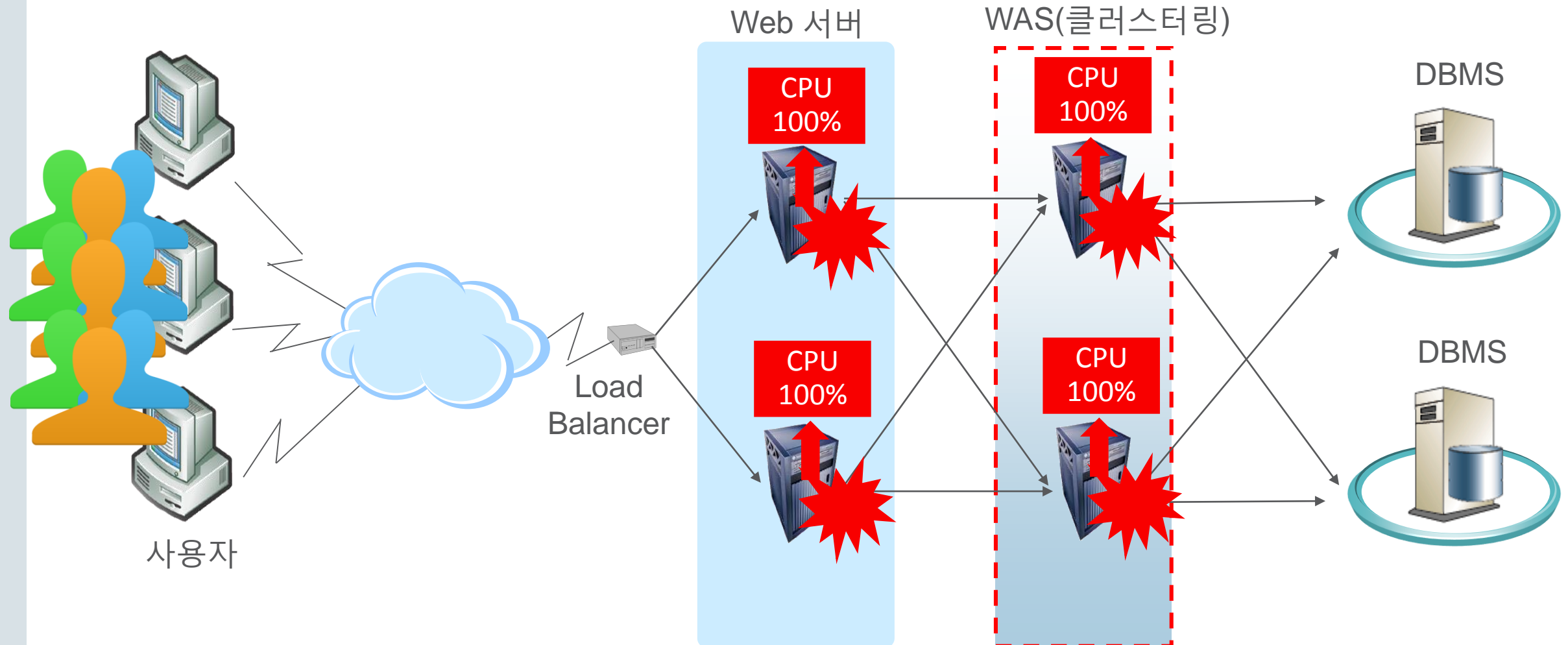
WAS (Web Application Server) 구성 Architecture

3 Tier Architecture



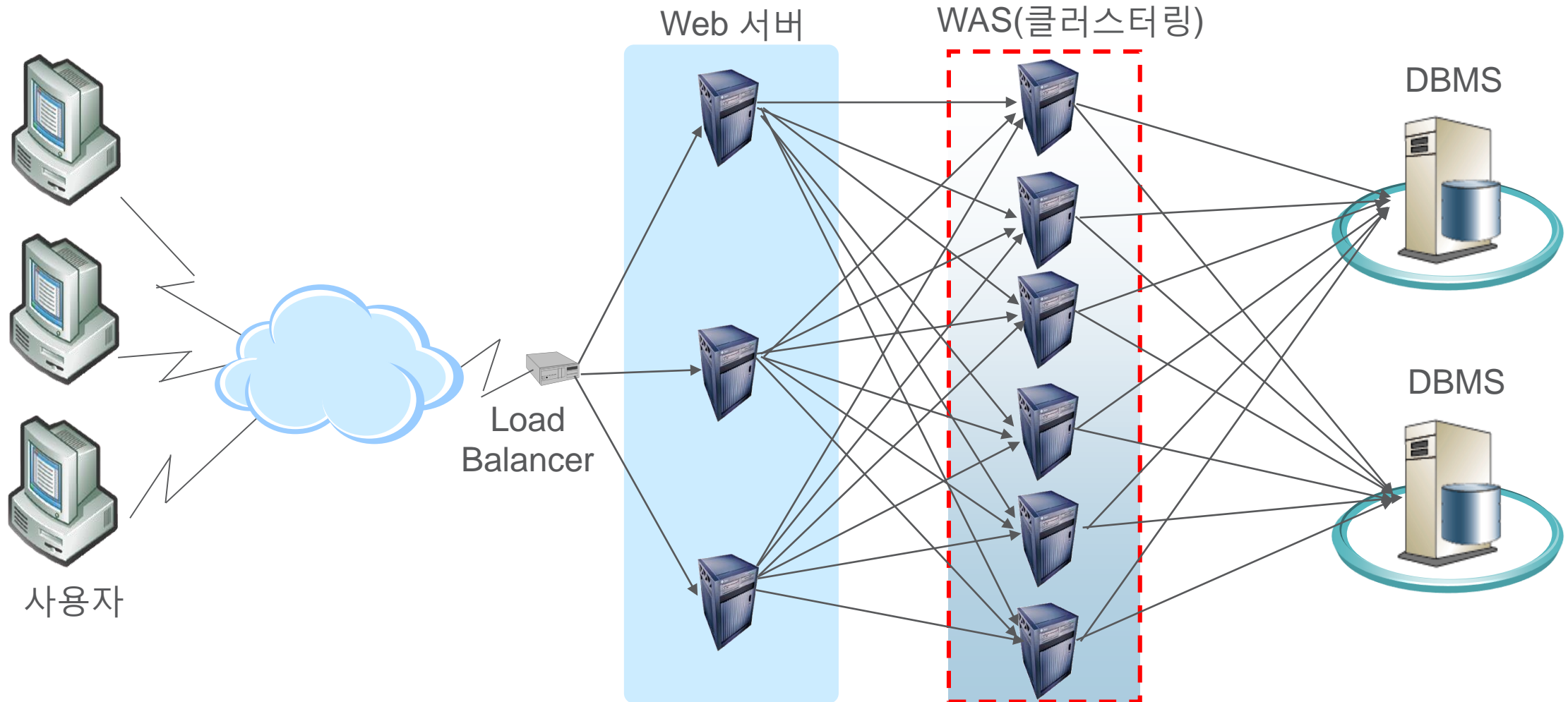
WAS (Web Application Server) 구성 Architecture

3 Tier Architecture



WAS (Web Application Server) 구성 Architecture

3 Tier Architecture





ORACLE®