

Spring 프레임워크 중요구성원리

2018. 5. 16

조효은

honnynoop@naver.com

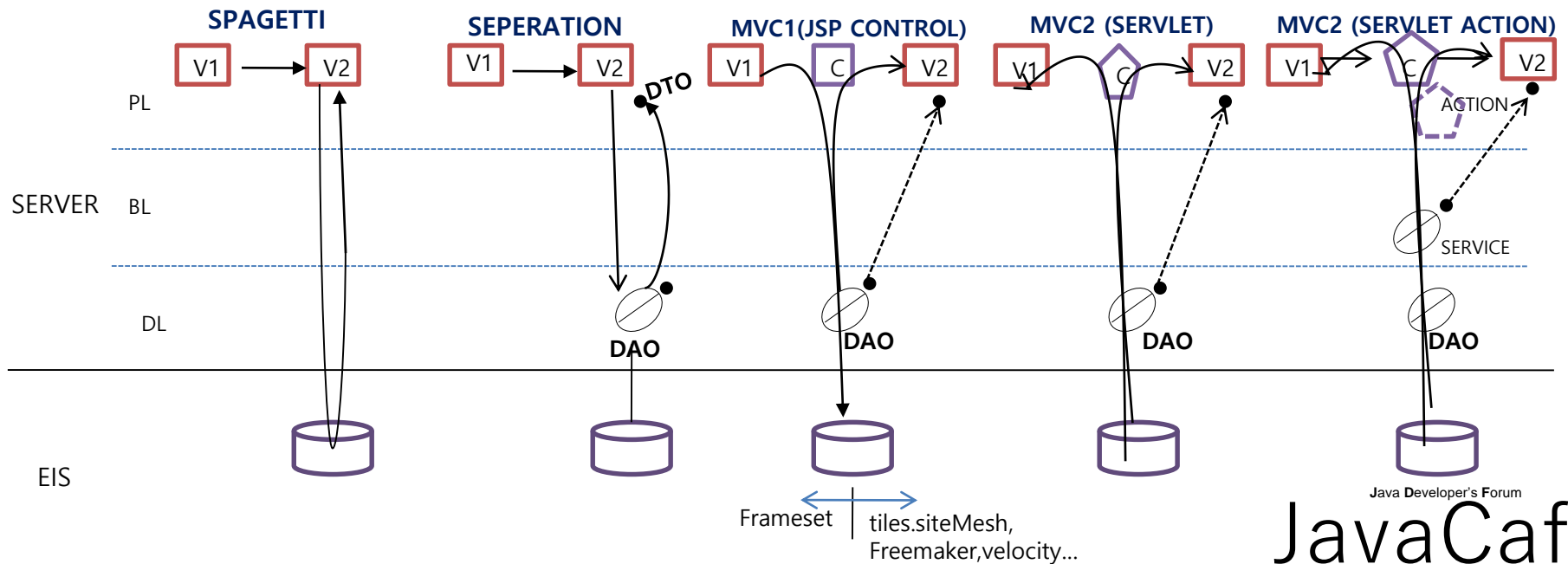
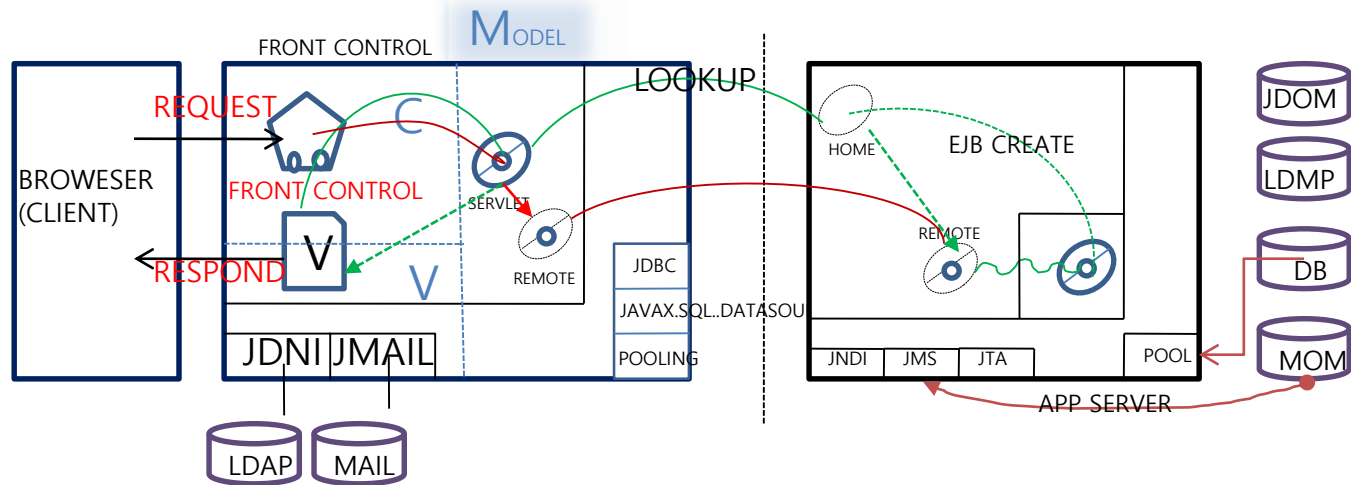
Today Story ...

1. 티어와 레이어
2. 웹 프로그래밍과 엔터프라이즈 프로그래밍
3. MVC 모델과 웹 개발의 흐름
4. Spring 3대 구성원리와 디자인패턴 5대 원리
5. AJAX와 데이터 처리

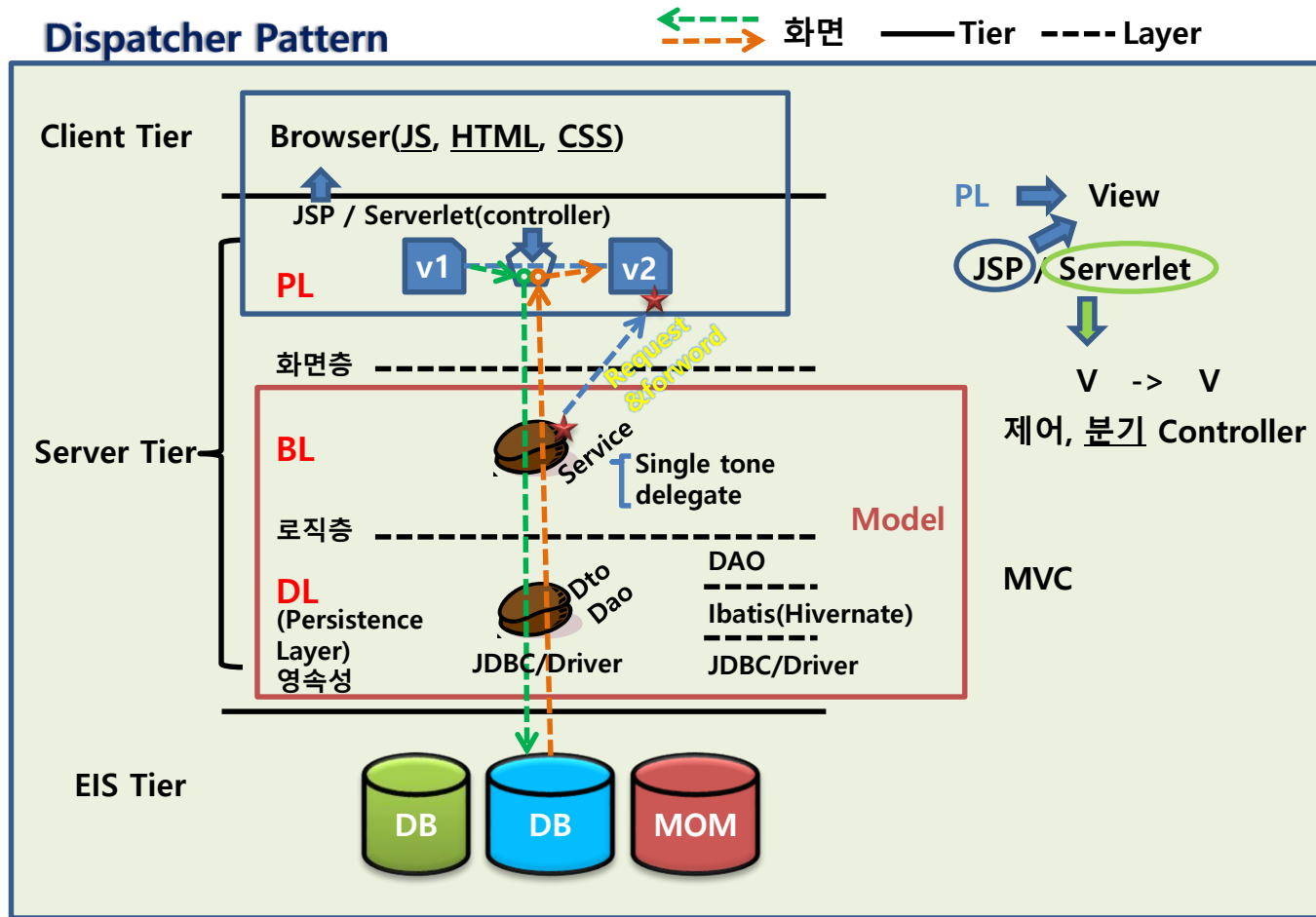
Today Story ...

1. 티어와 레이어
2. 웹 프로그래밍과 엔터프라이즈 프로그래밍
3. MVC 모델과 웹 개발의 흐름
4. Spring 3대 구성원리와 디자인패턴 5대 원리
5. AJAX와 데이터 처리

Tier & MVC ...

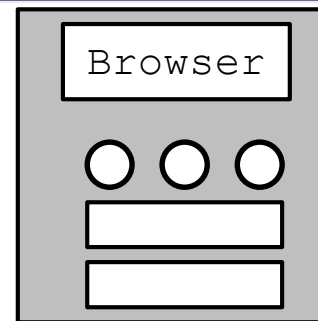
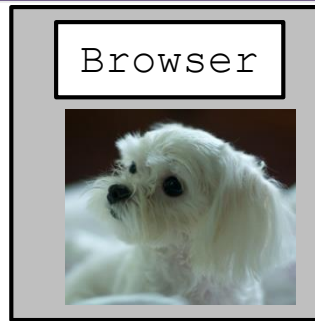


Tier/Layer



3 Tier...

User



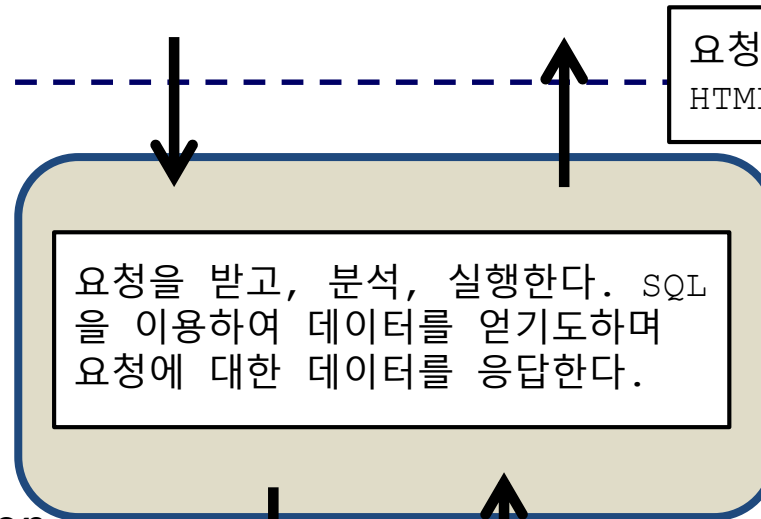
Client Tier

Presentation Layer
(JSP/Servlet)

Business Layer
(Service, EJB)

Data Layer
(Persistence, Integration
Layer : DAO, ORM)

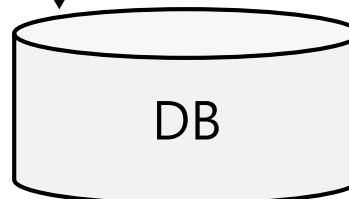
Resource Layer



요청을 동적으로 처리하여
HTML로 응답한다.

Server Tier

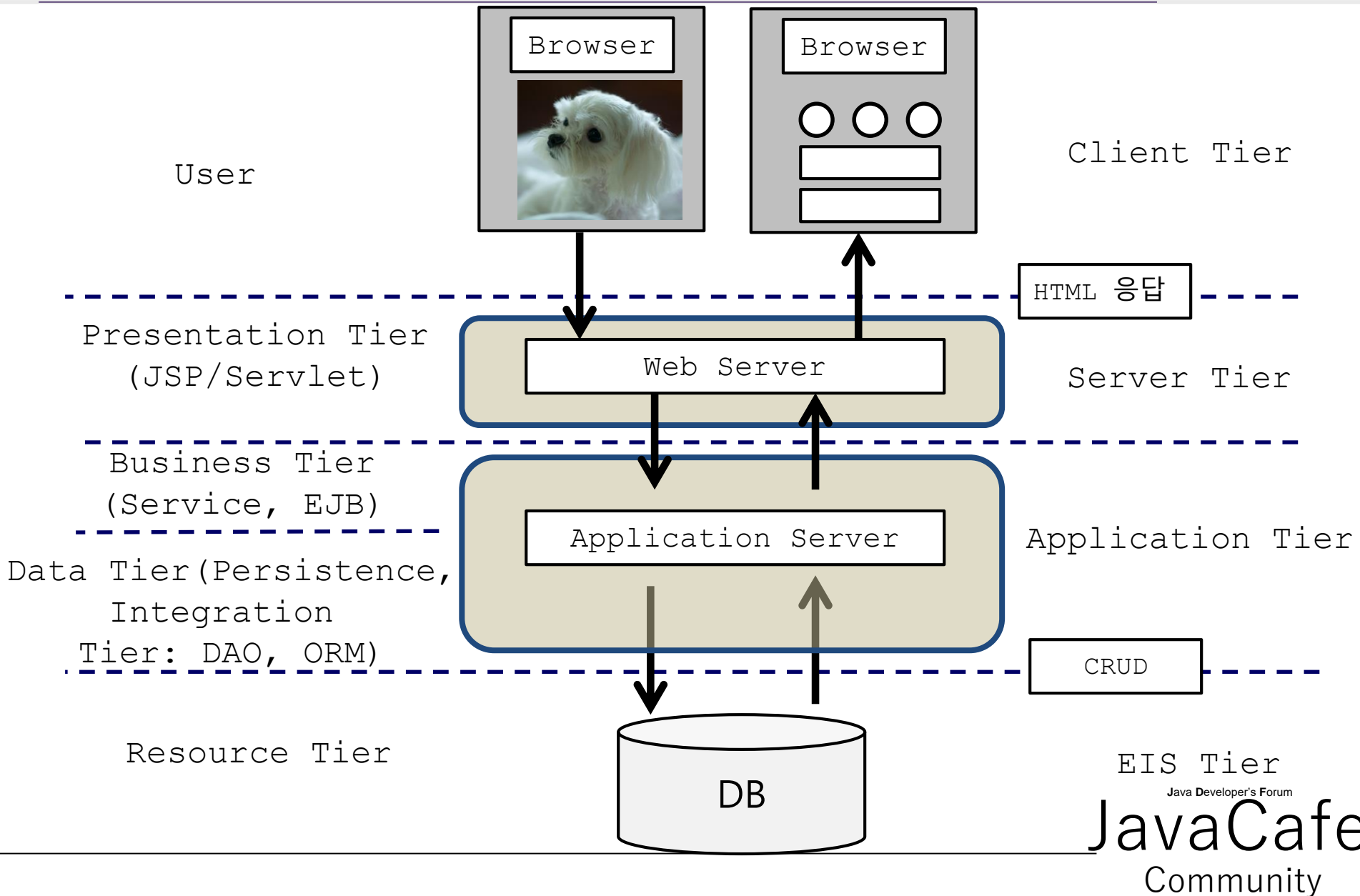
SQL을 이용하여 정보 검색, 추
가, 수정, 삭제 (CRUD)를 한다.



EIS Tier
Java Developer's Forum

JavaCafe
Community

4 Tier ...

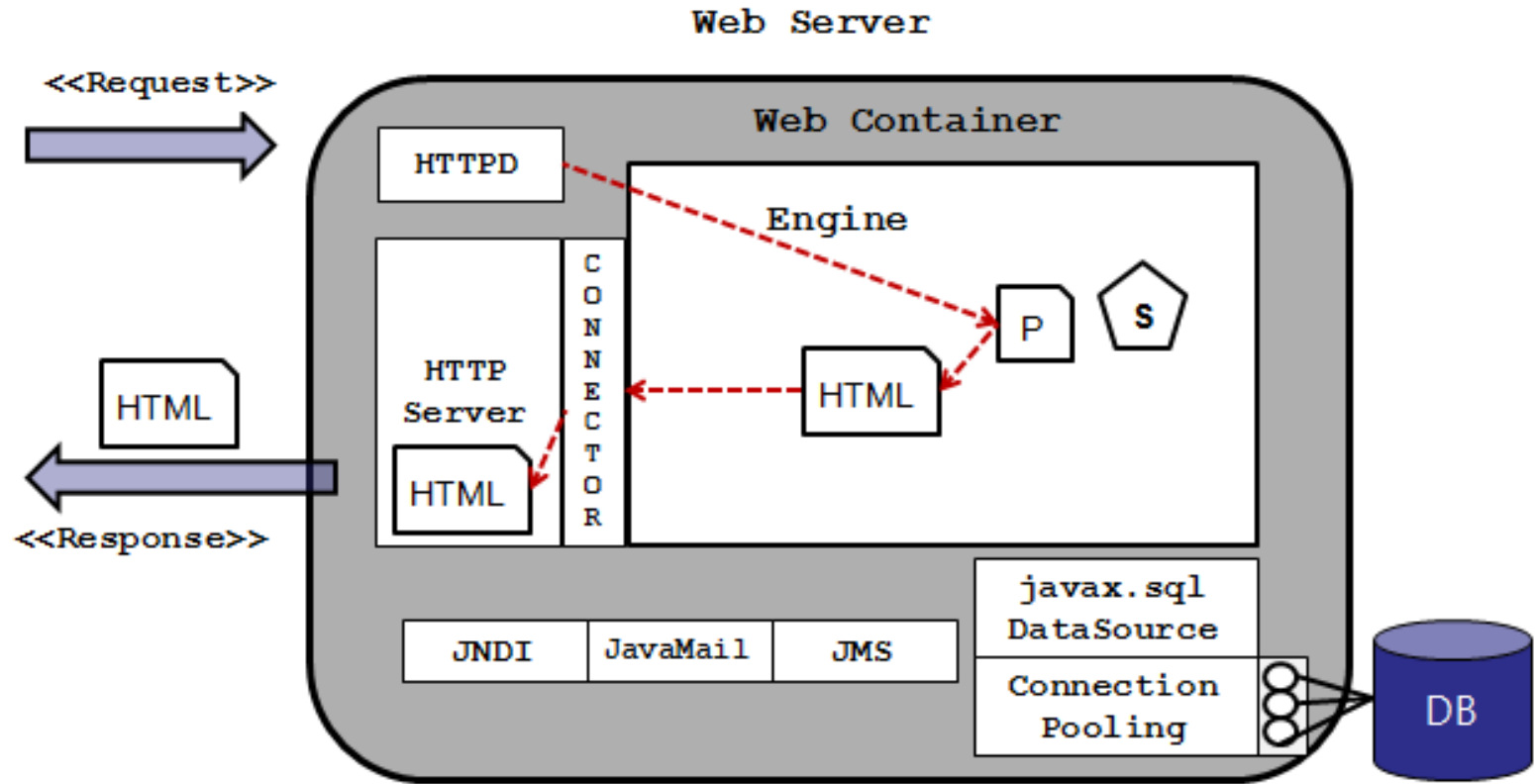


Today Story ...

1. 티어와 레이어
2. 웹 프로그래밍과 엔터프라이즈 프로그래밍
3. MVC 모델과 웹 개발의 흐름
4. Spring 3대 구성원리와 디자인패턴 5대 원리
5. AJAX와 데이터 처리

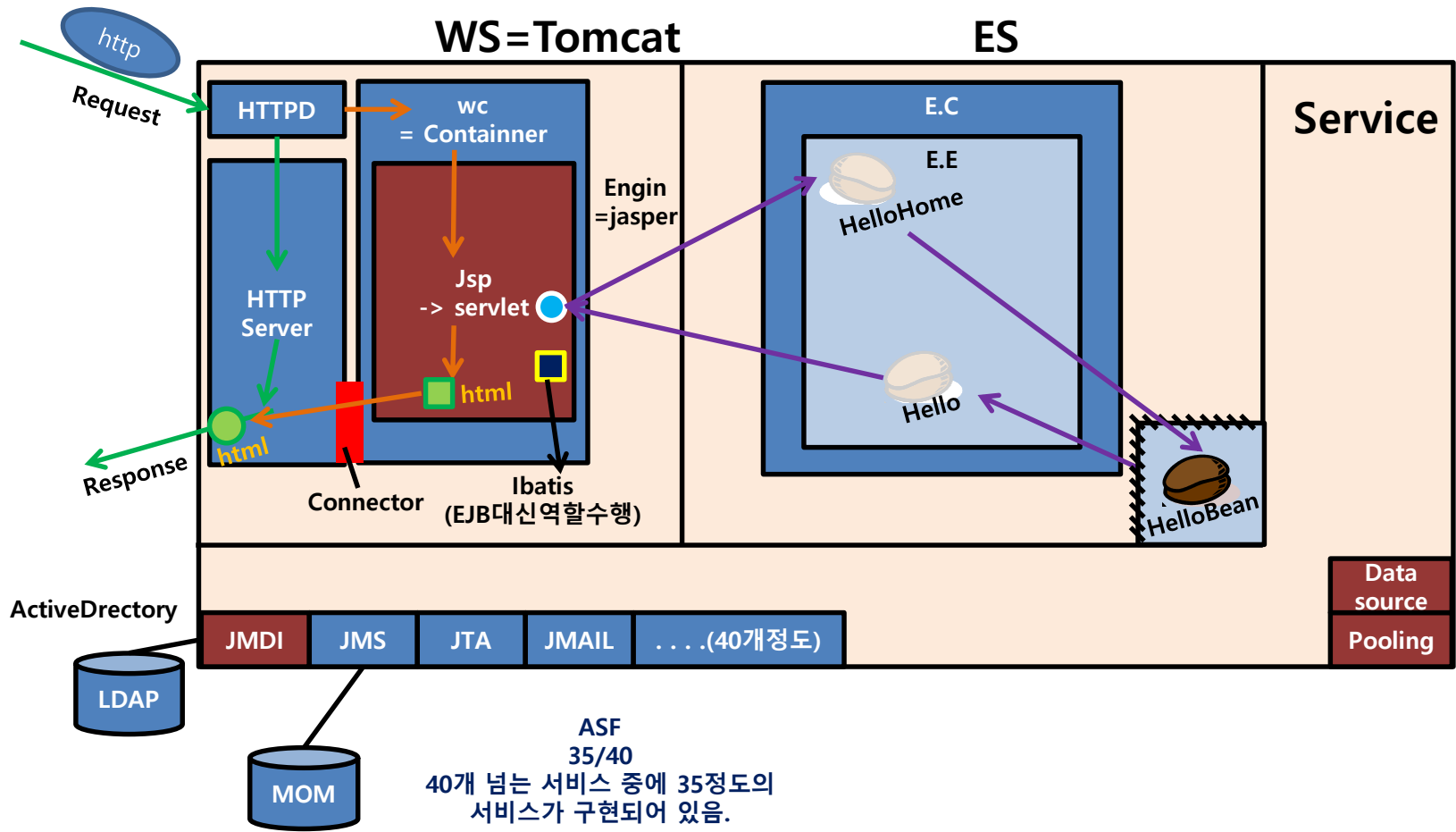
Web Server...

WS = HTTP Server + Web container + Web Engine +Connector



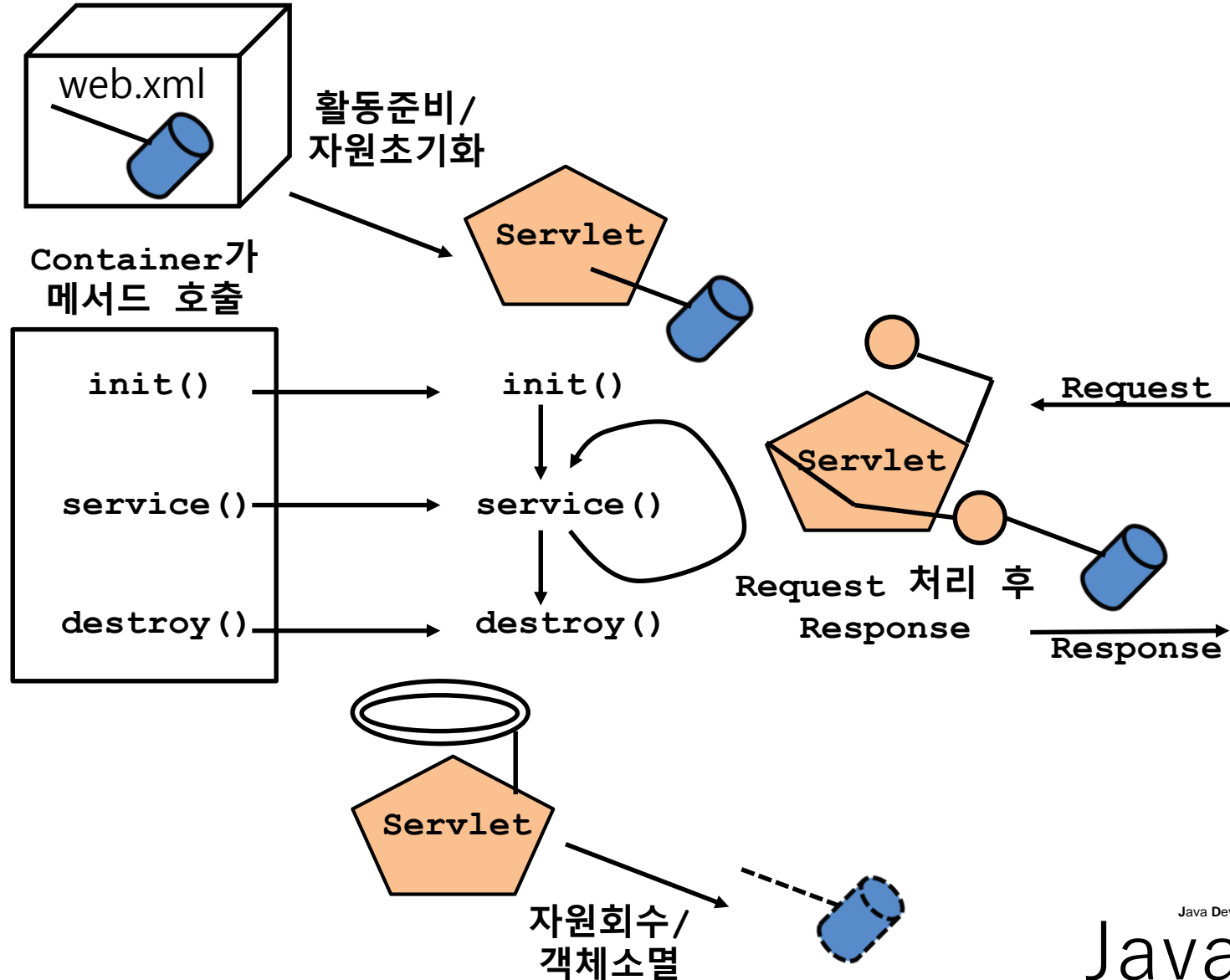
WAS ...

WAS = WS + ES + SERVICE



ASF
35/40
40개 넘는 서비스 중에 35정도의
서비스가 구현되어 있음.

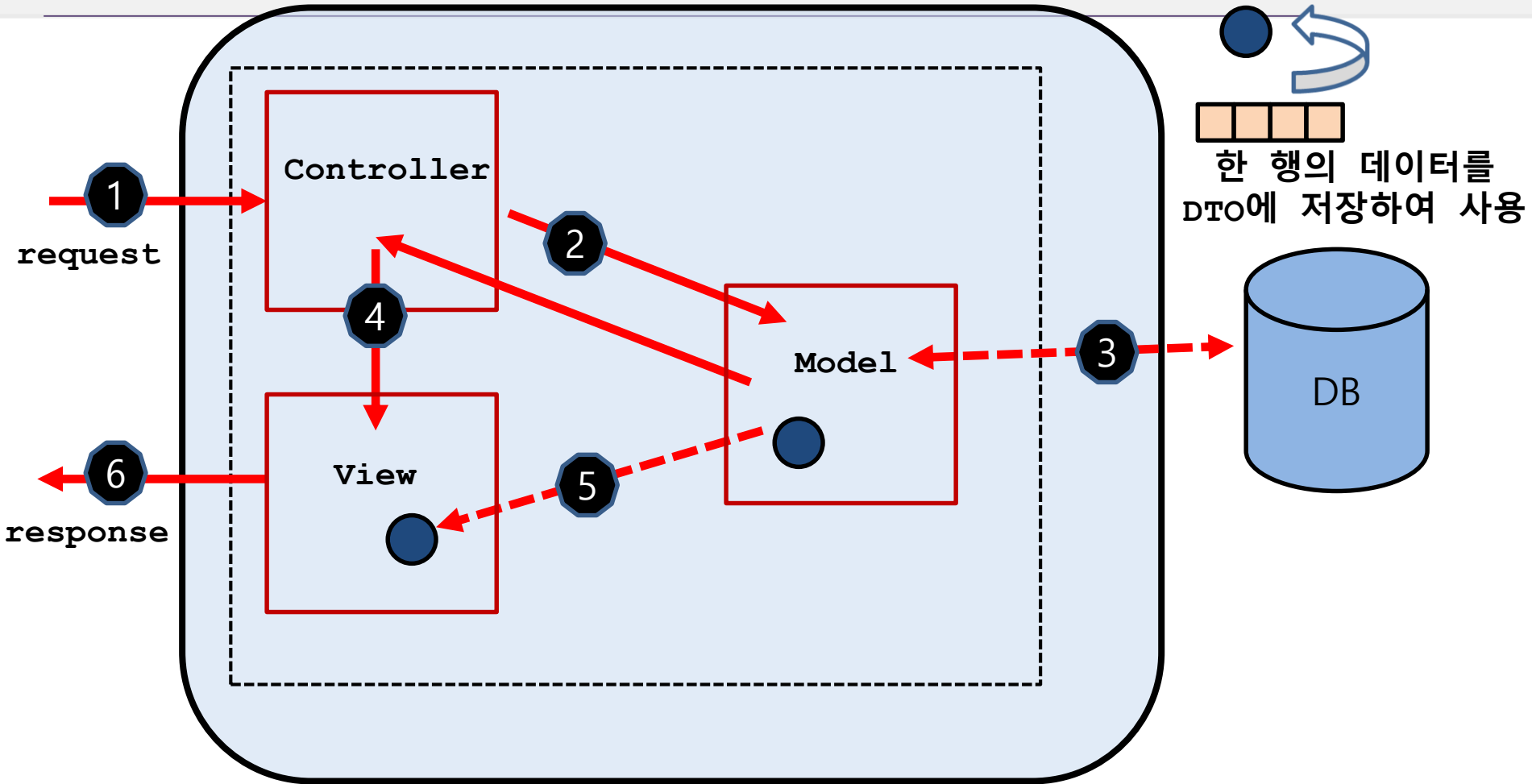
Life Cycle ...



Today Story ...

1. 티어와 레이어
2. 웹 프로그래밍과 엔터프라이즈 프로그래밍
3. MVC 모델과 웹 개발의 흐름
4. Spring 3대 구성원리와 디자인패턴 5대 원리
5. AJAX와 데이터 처리

MVC ...

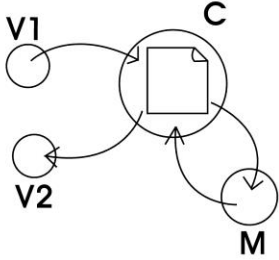
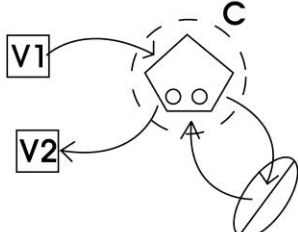
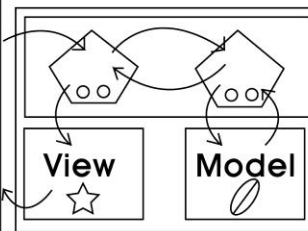
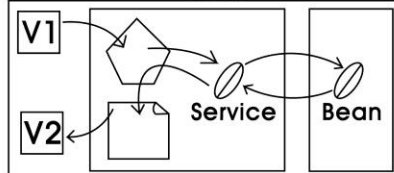


① 요청 ⑥ 응답 ② 데이터를 처리하거나 얻는다 ③ CRUD 실행

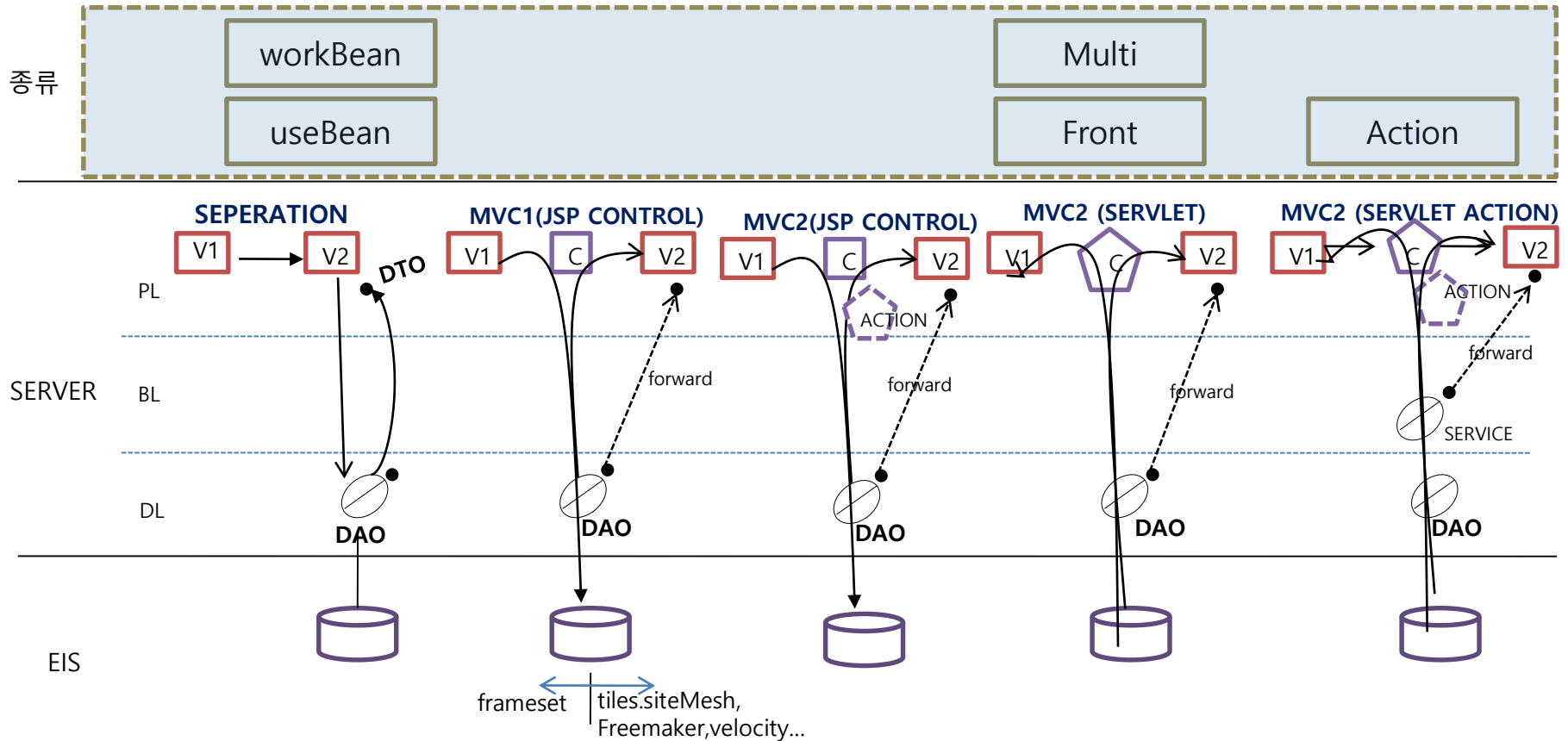
④ 처리결과에 따라 해당화면으로 forward

⑤ 처리결과에 따라 해당 Business 객체를 넘겨서 화면에 출력한다

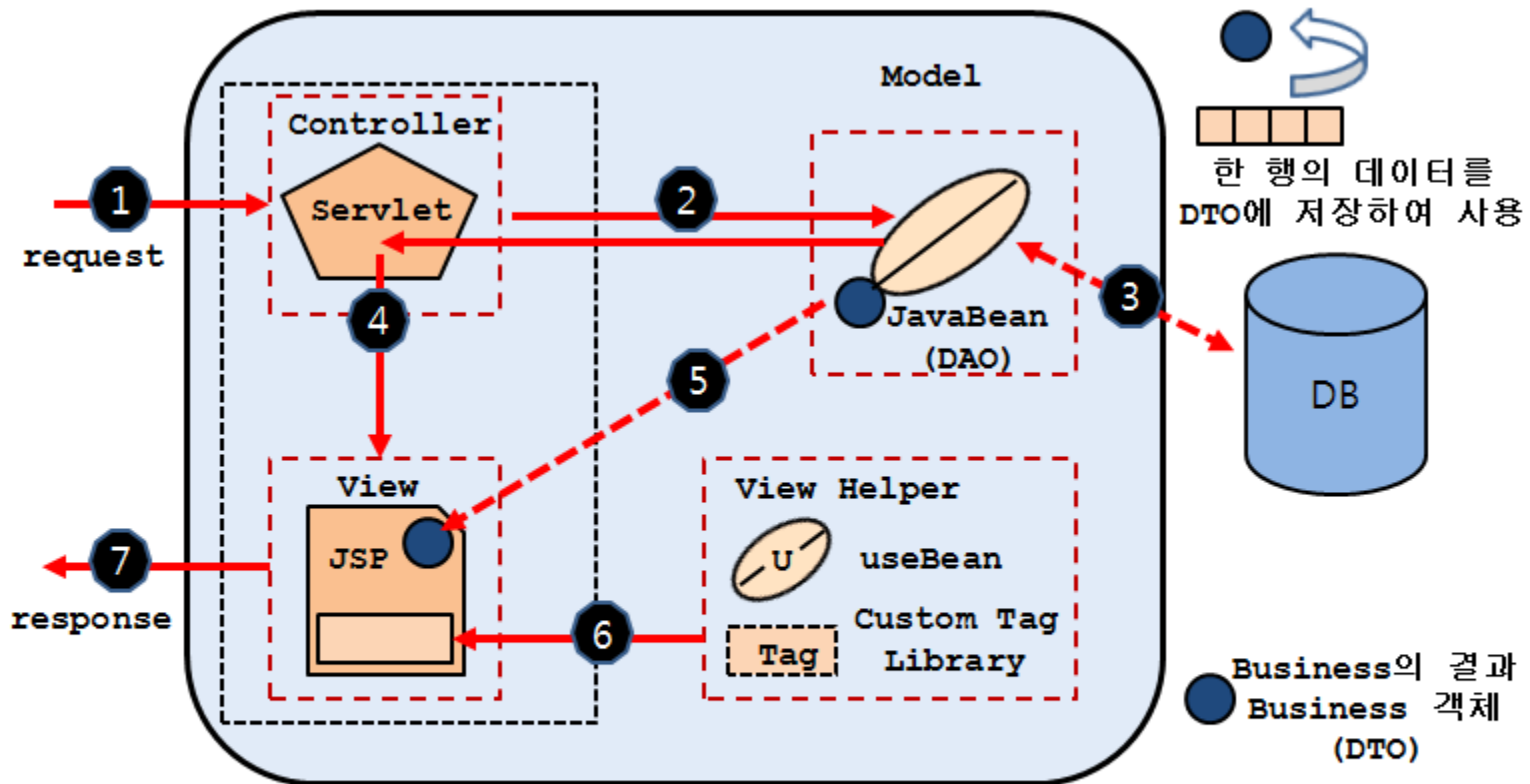
Web History...

1997	1999 JSP		2002	2004	2005
only servlet	spagetti	(old MVC1) separation	MVC JSP *scope (old MVC2) Jsp Centric	MVC2 Servlet Centric	FrameWork
<ul style="list-style-type: none"> * 계층구조 * life cycle * url mapping 	<ul style="list-style-type: none"> * scriptElement view + viewlogic 	<ul style="list-style-type: none"> * scriptlet 줄이자 view + viewlogic <p>DL { WorkBean DAO => DTO</p>	<p>EL JSTL \${cust.id} OGNL controller 7가지</p> 	 <p>Multi Controller ↓ Action ↓ Struts ⇒ Spring</p>	<p>FrontControl</p>  <p>WAS</p>  <p>MVC EJB Model2</p>
<p>HelperView 패턴</p> <p>XML < UseBean -> Action CustomTag -> 사용자 정의</p> <ol style="list-style-type: none"> ① Class xxx extends TagSupport ② xxx.tld ③ <%@taglib prefix="xxx" url="/xxx.tld"%> ④ <yyy:show/> 					

SoC(Separation of Concern)...

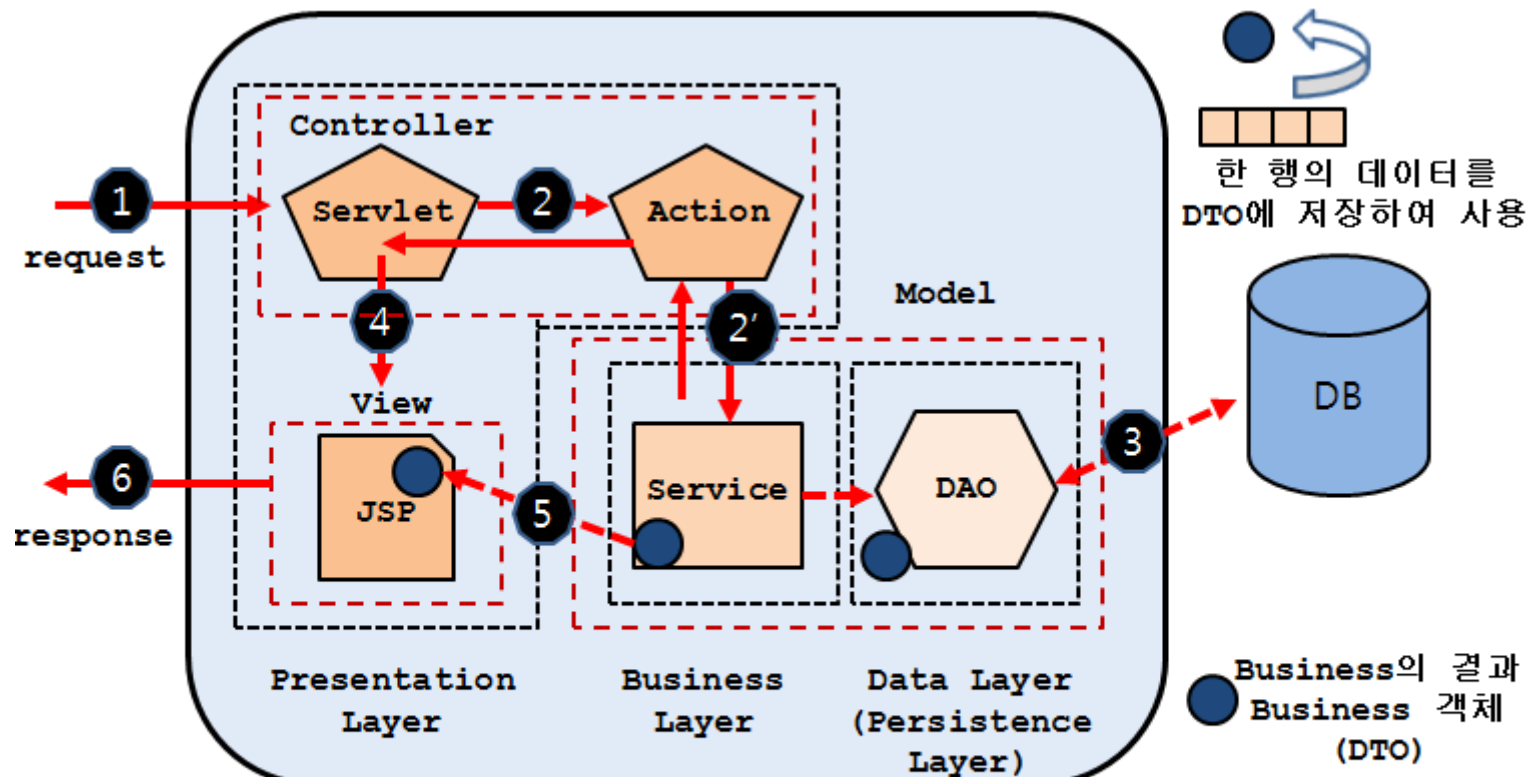


MVC...



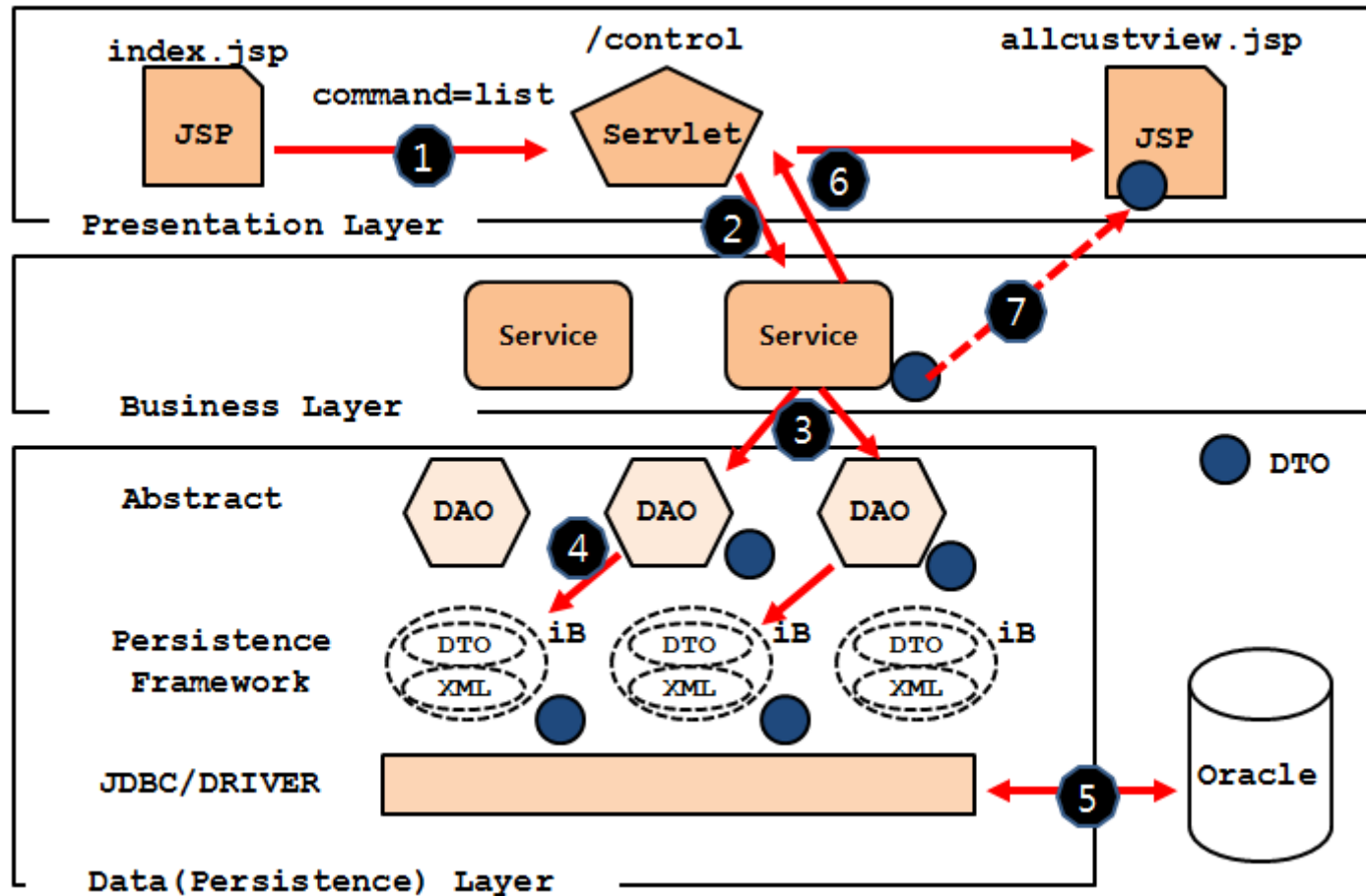
- ① 요청 ⑦ 응답 ② 데이터를 처리하거나 얻는다 ③ CRUD 실행
④ 처리결과에 따라 해당화면으로 forward ⑤ 처리결과에 따라 해당 Business 객체를 넘긴다. ⑥ 부분화면생성 Business 객체 출력

MVC...

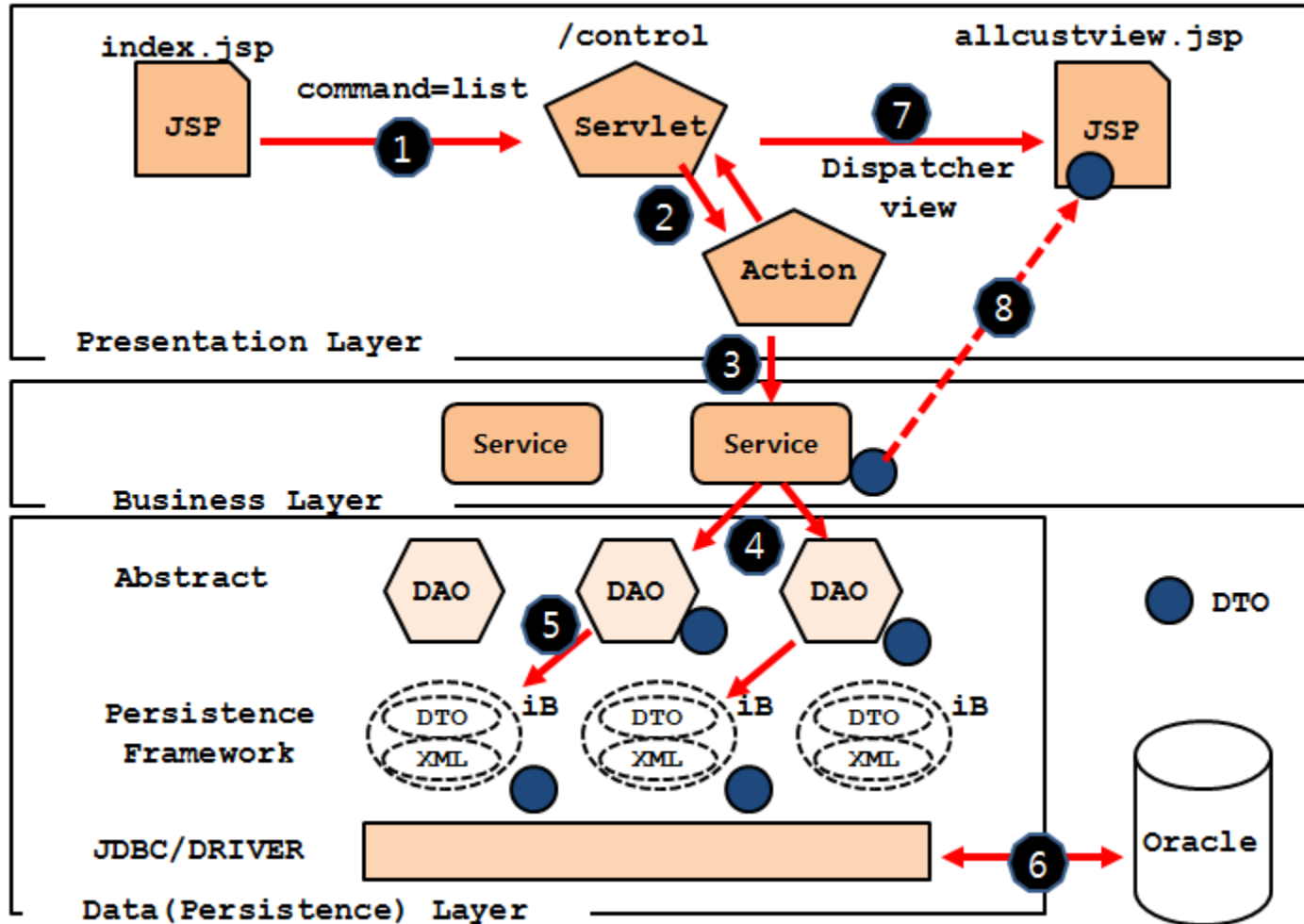


- 1 요청 6 응답 2 데이터를 처리하거나 얻는다 3 CRUD 실행
4 처리결과에 따라 해당화면으로 forward 5 처리결과에 따라 해당 Business 객체를 넘겨서 화면에 출력한다

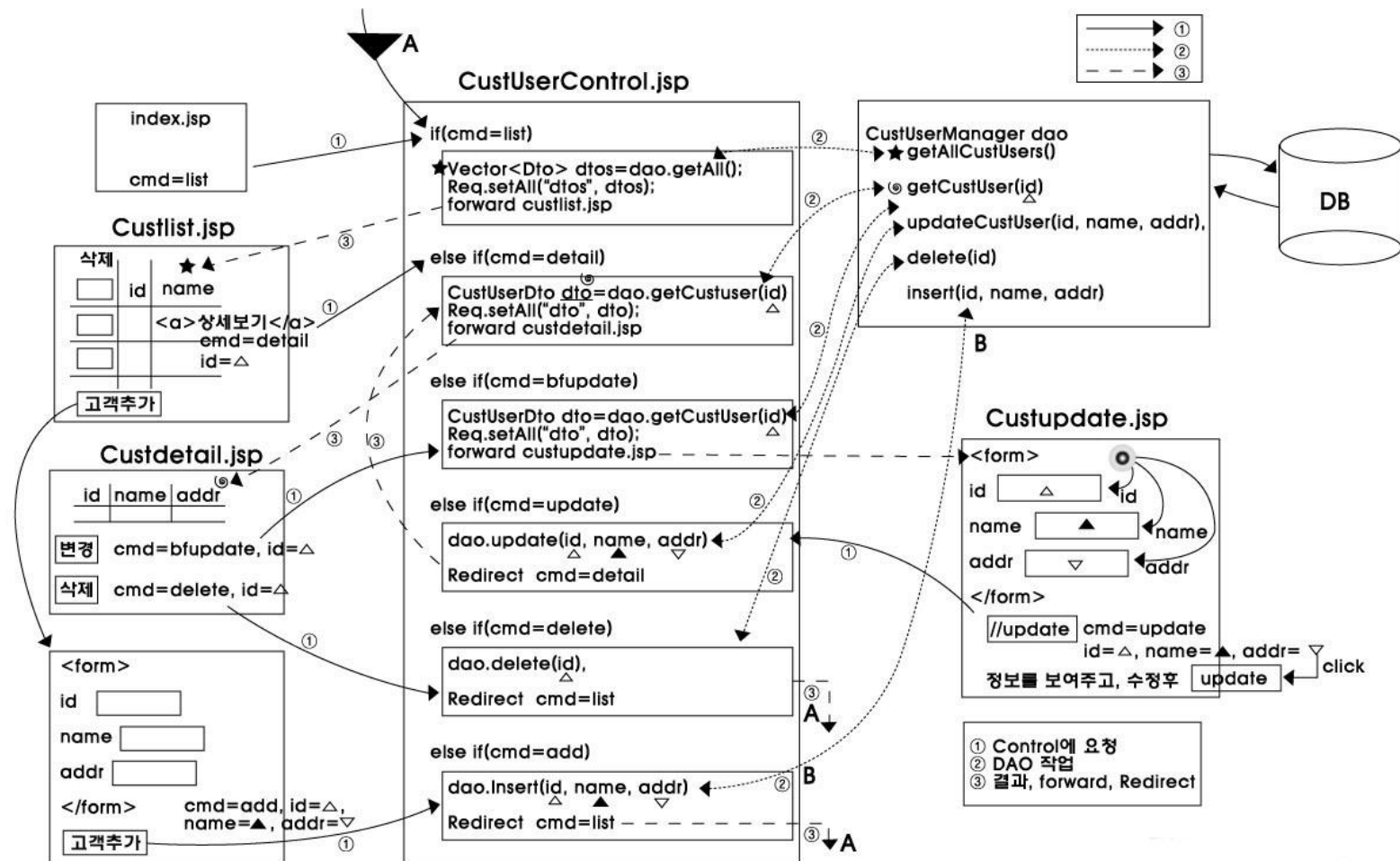
Front Servlet...



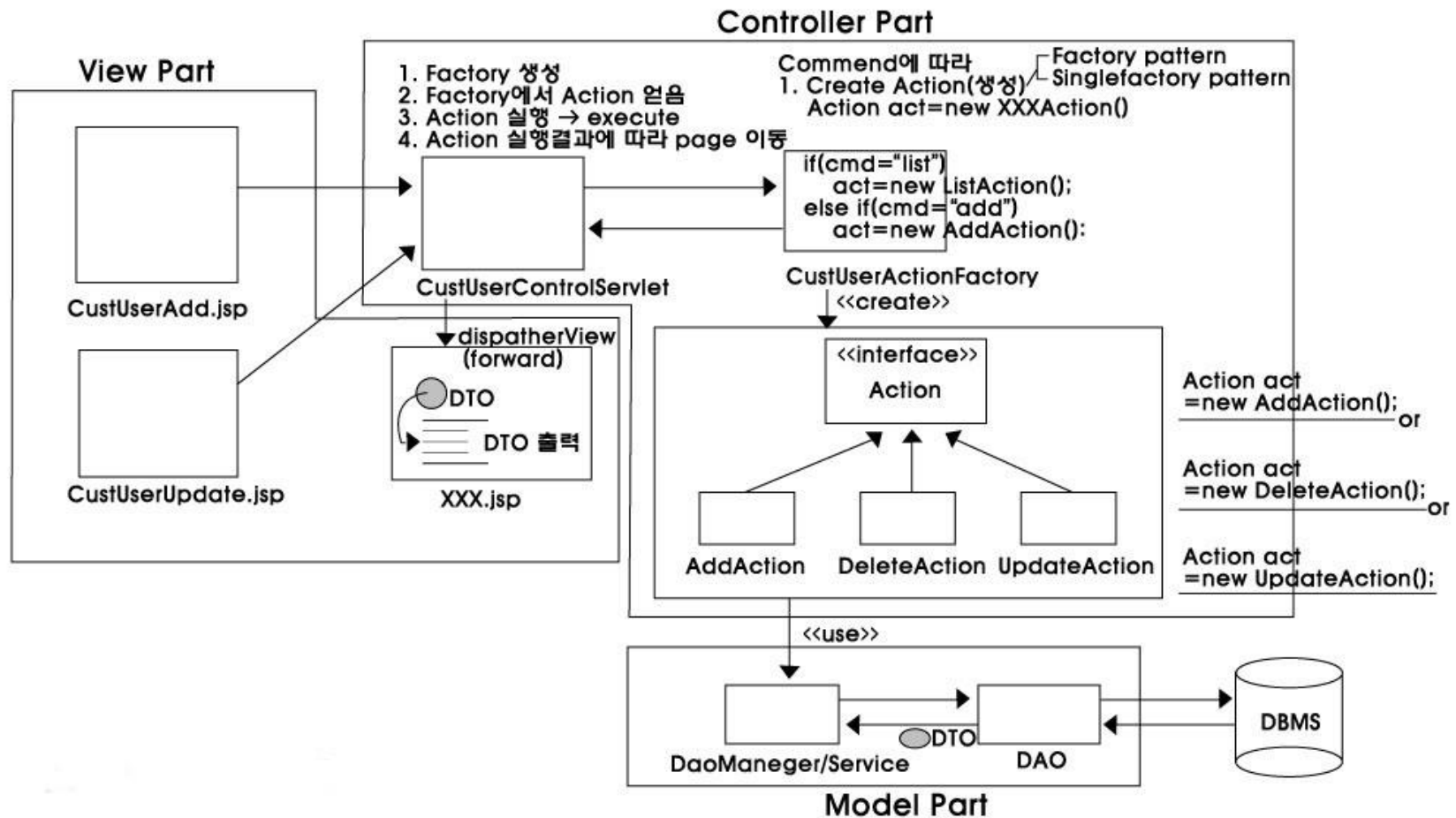
Action Servlet...



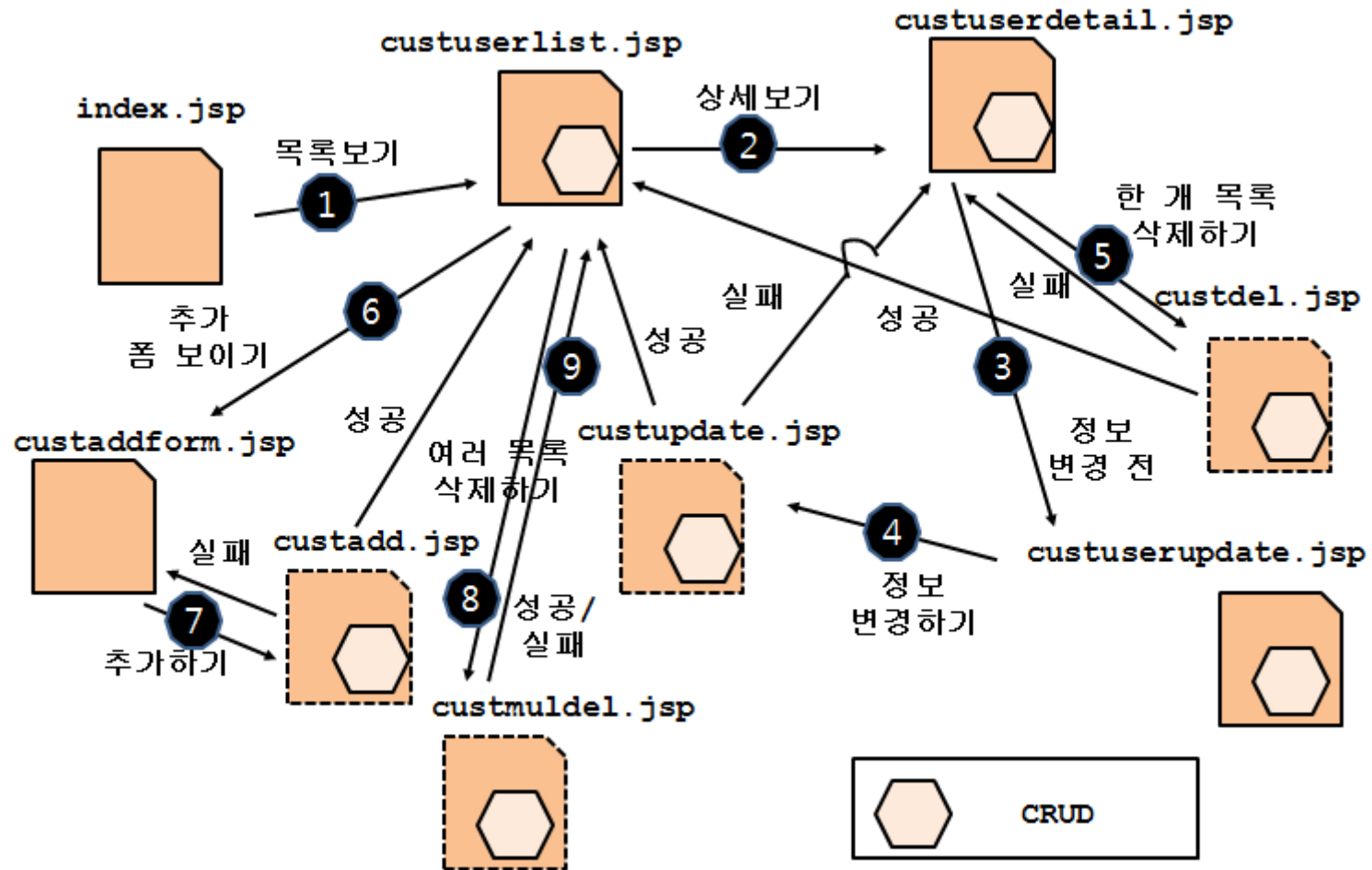
MVC JSP Centric...



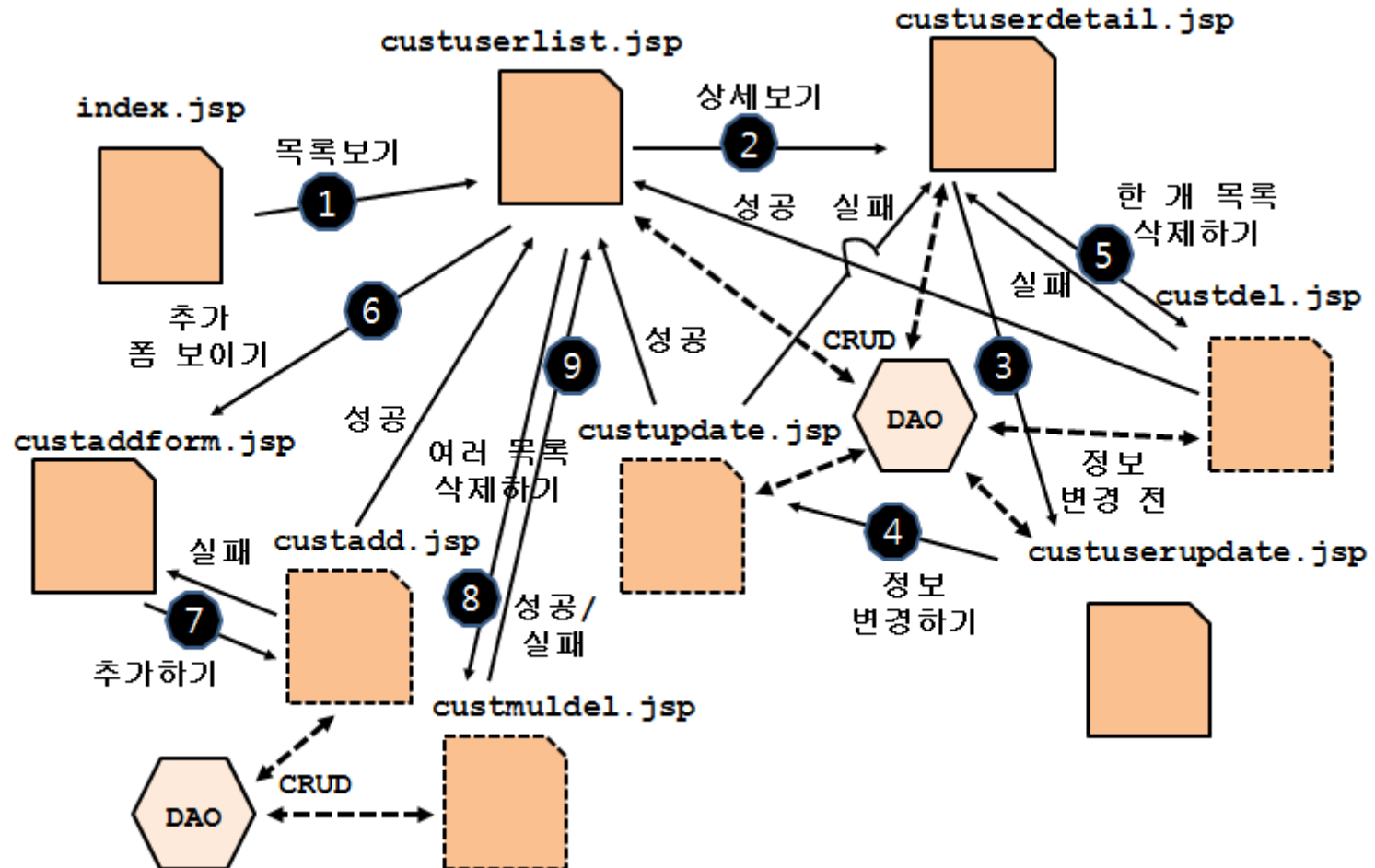
MVC Servlet Centric(Action Servlet)...



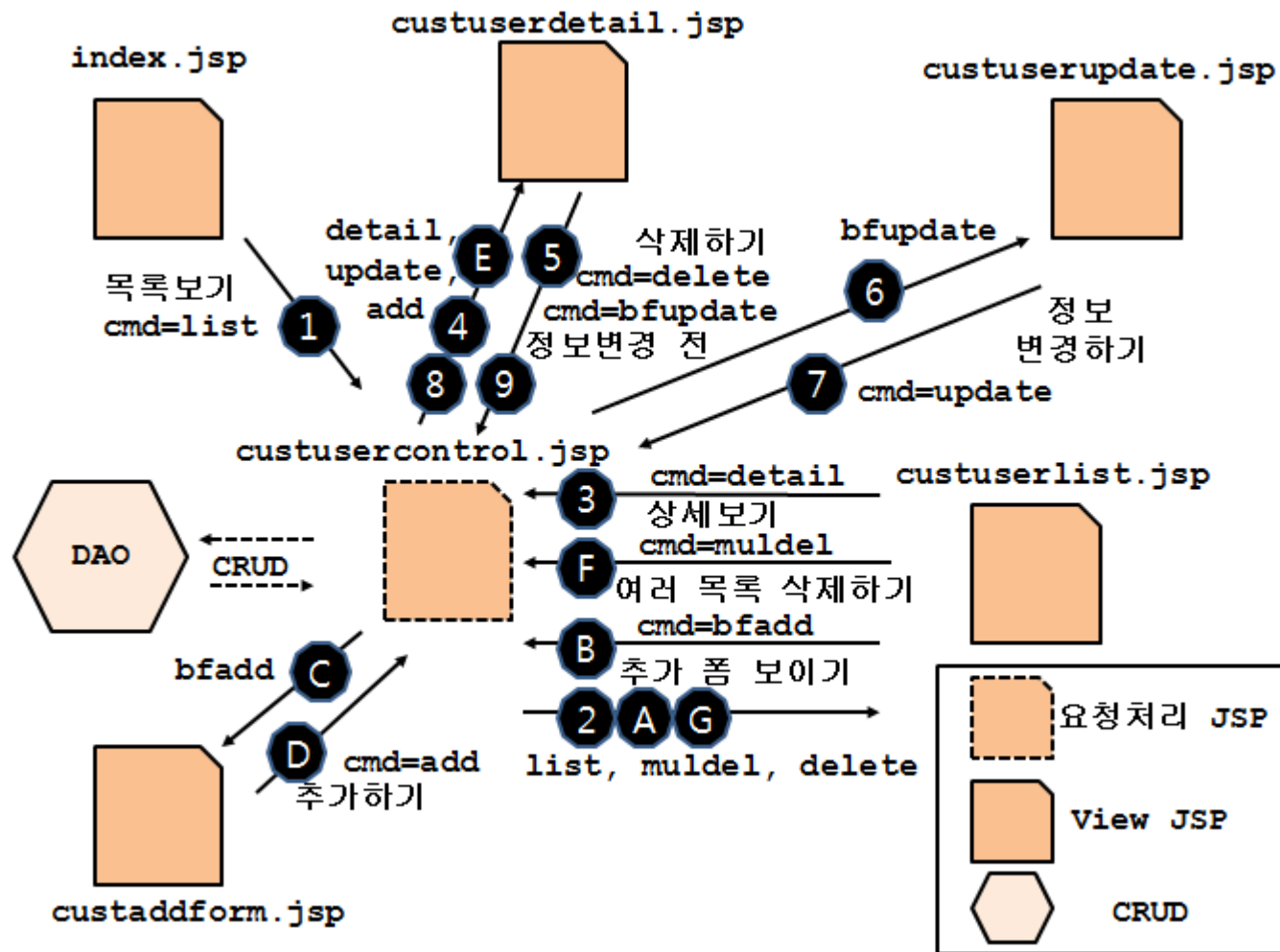
Spagetti...



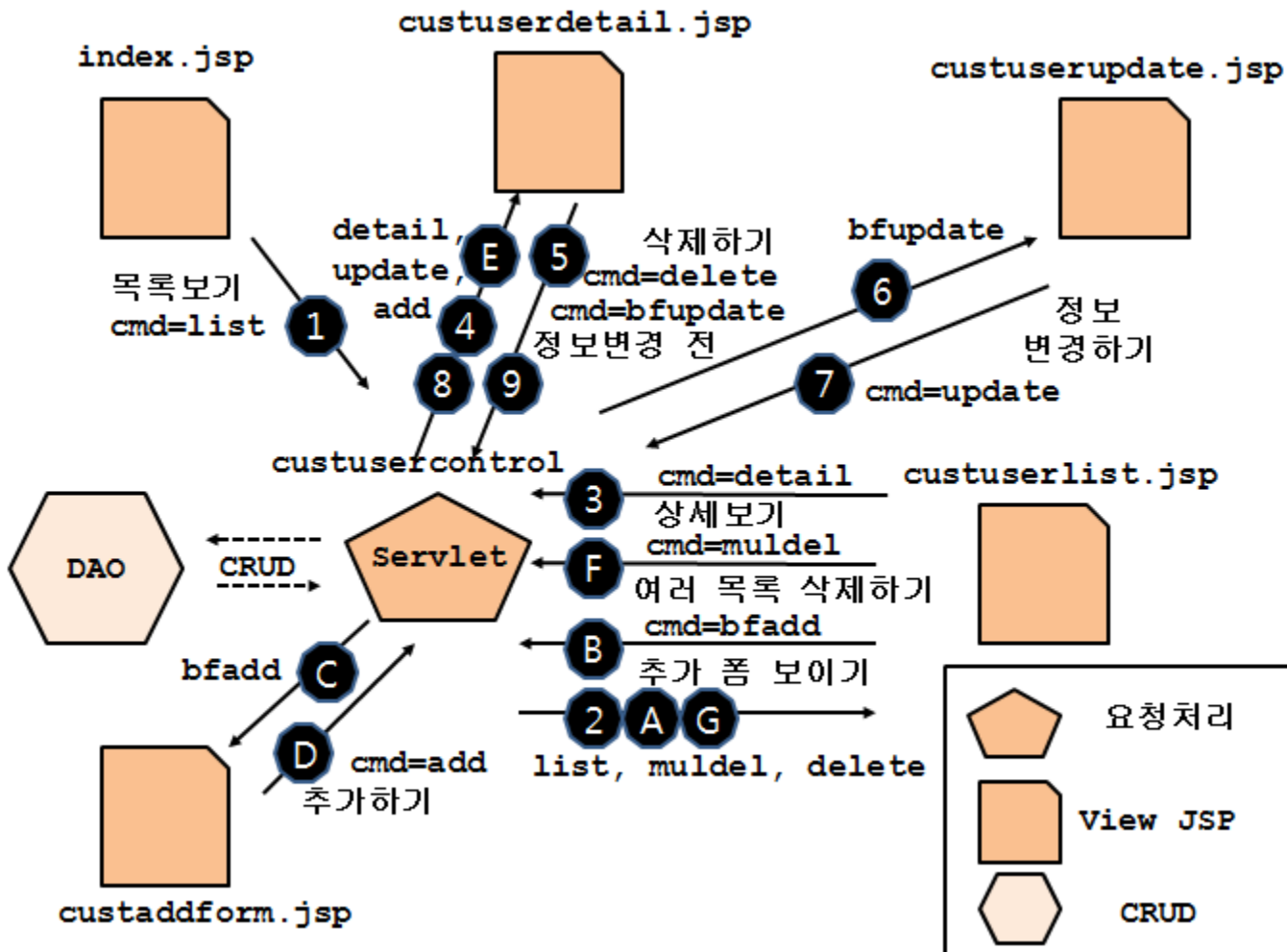
Separation...



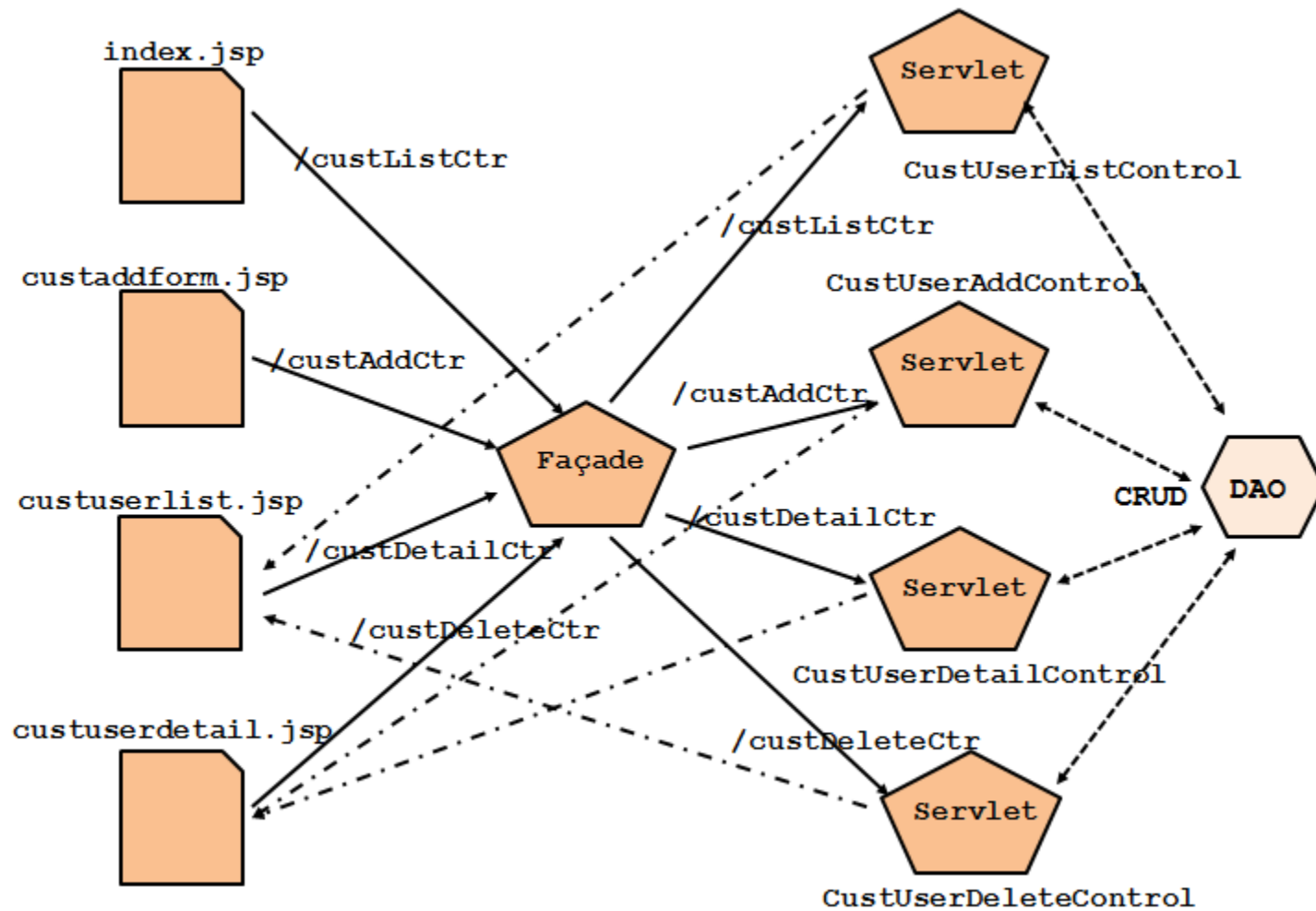
MVC Model 1 JSP Centric...



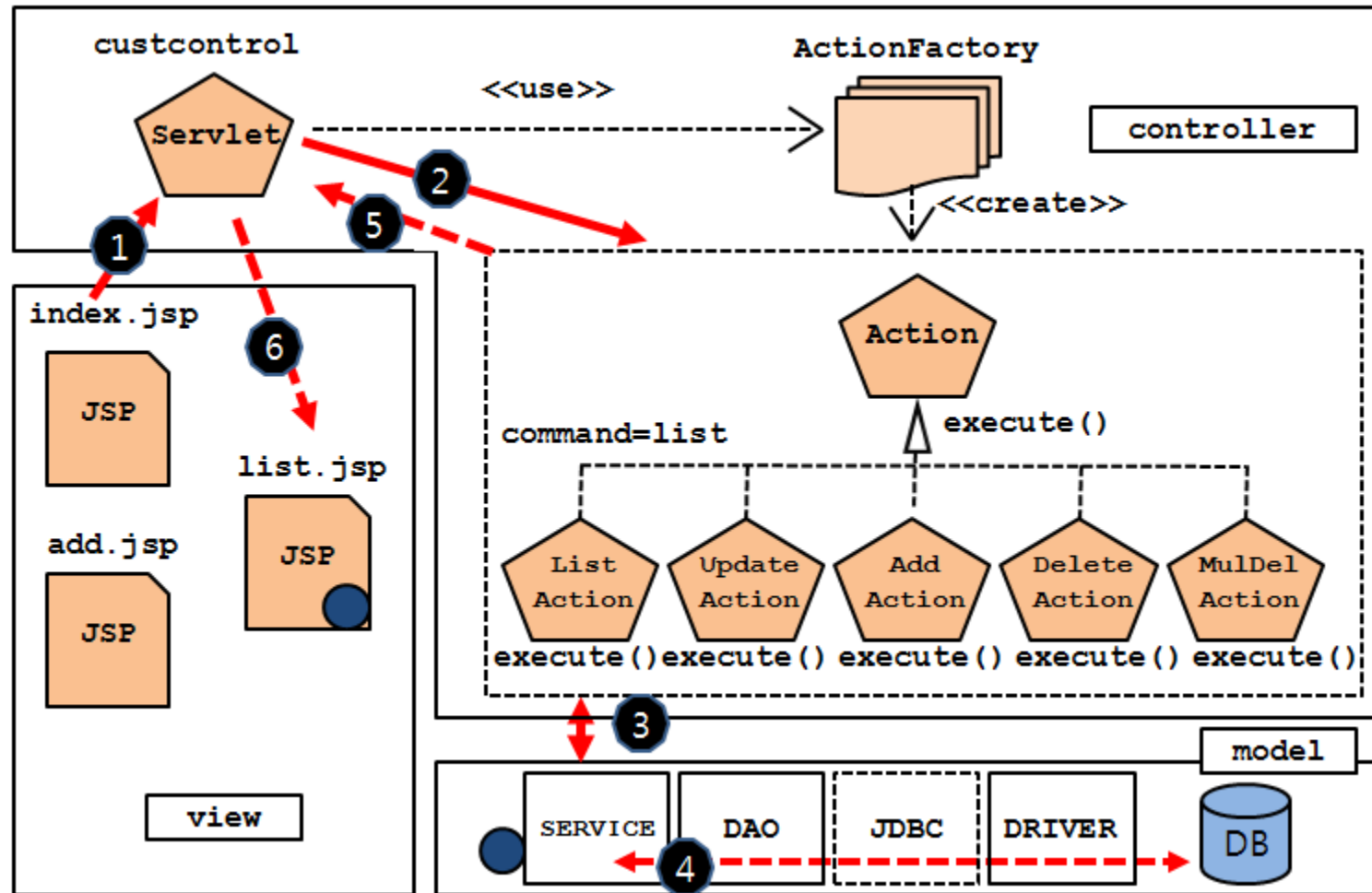
MVC Model 2 Servlet Centric...



Facade...



Action...



Listener/Filter ...

Listener

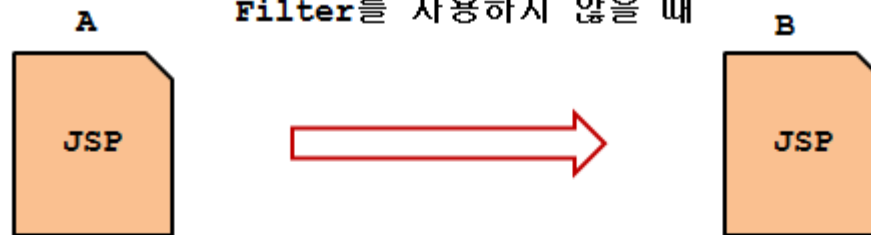
특정 행위가 발생하면
등록된 메서드를 호출



Context가 호출되기 직전→

ContextListener의
initializedContext() 메서드 호출

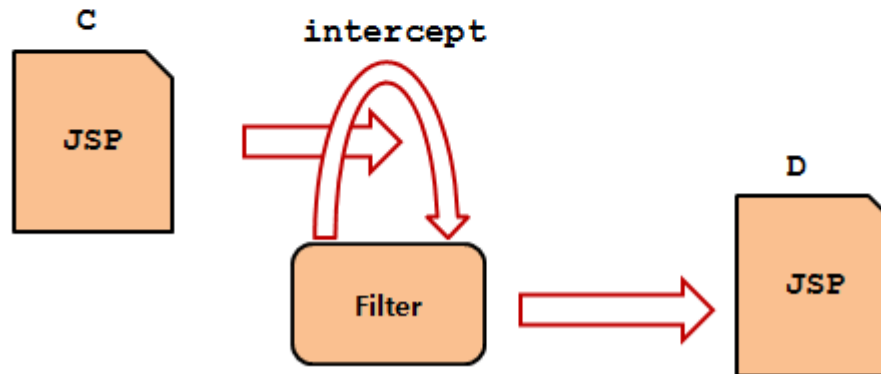
Filter를 사용하지 않을 때



Filter

가로채기

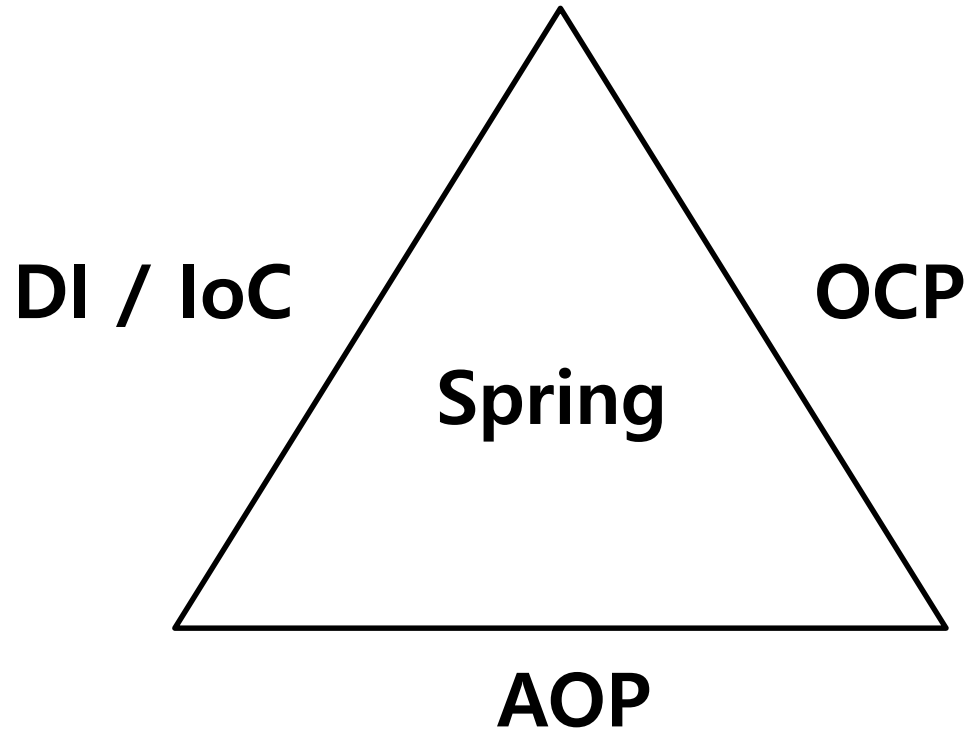
Filter를 사용할 때
intercept



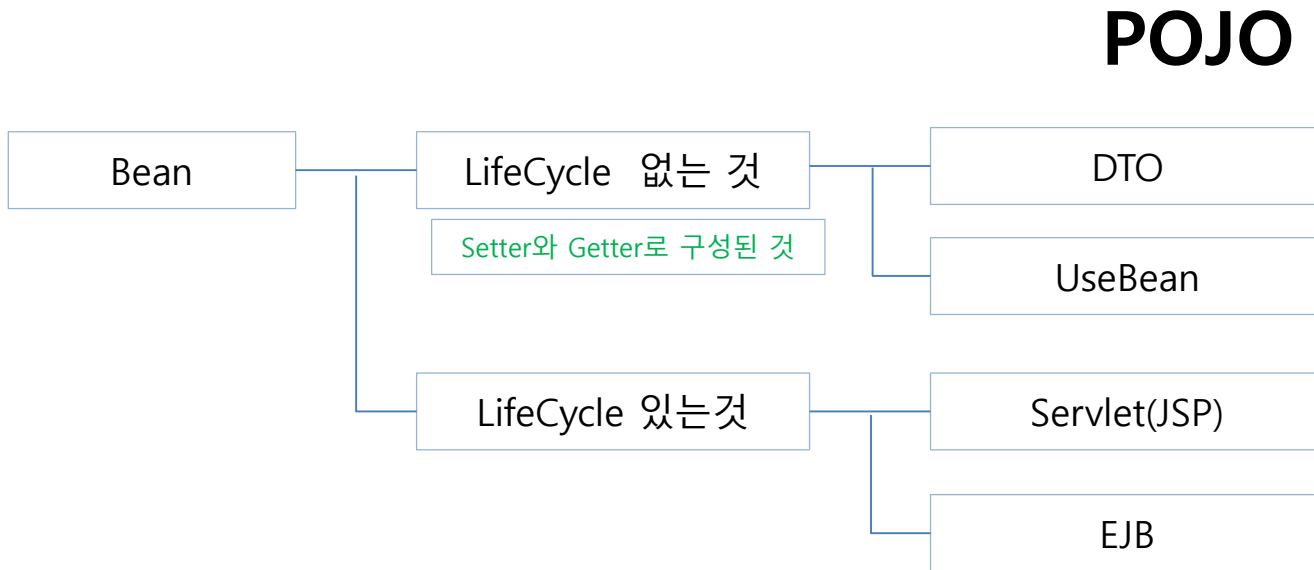
Today Story ...

1. 티어와 레이어
2. 웹 프로그래밍과 엔터프라이즈 프로그래밍
3. MVC 모델과 웹 개발의 흐름
4. Spring 3대 구성원리와 디자인패턴 5대 원리
5. AJAX와 데이터 처리

Spring 3대 원리...

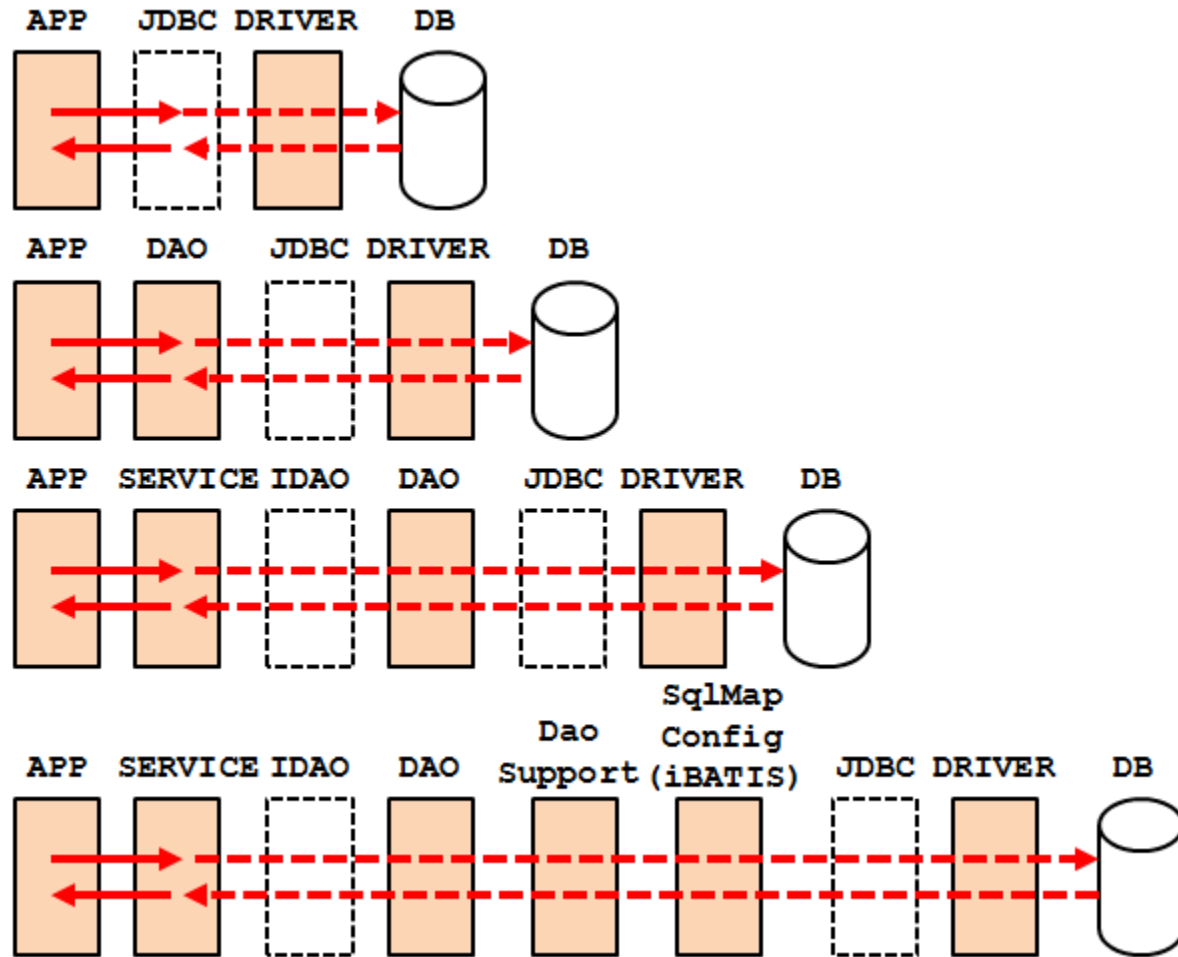


Bean ...

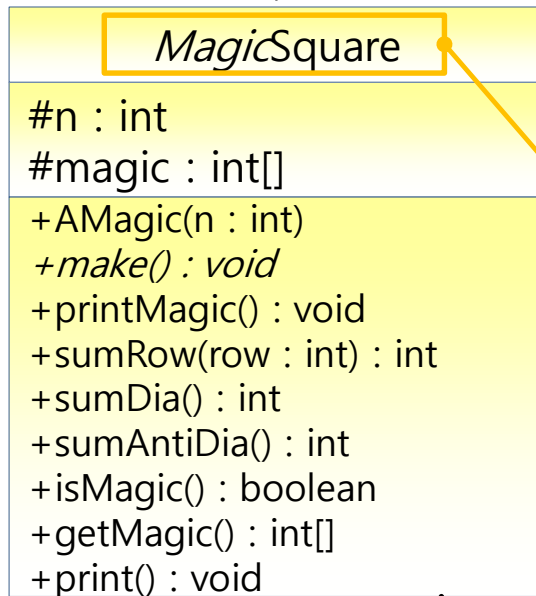
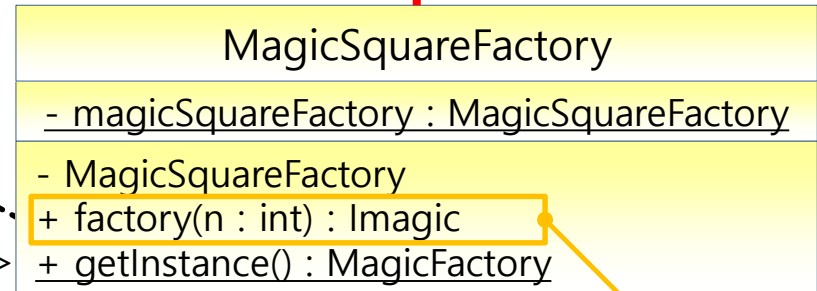
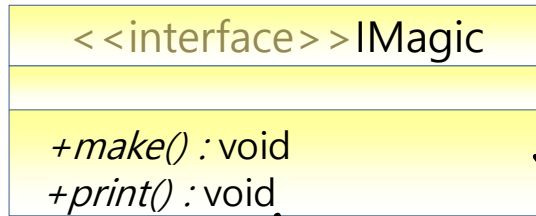


LCO

Abstraction Programming ...



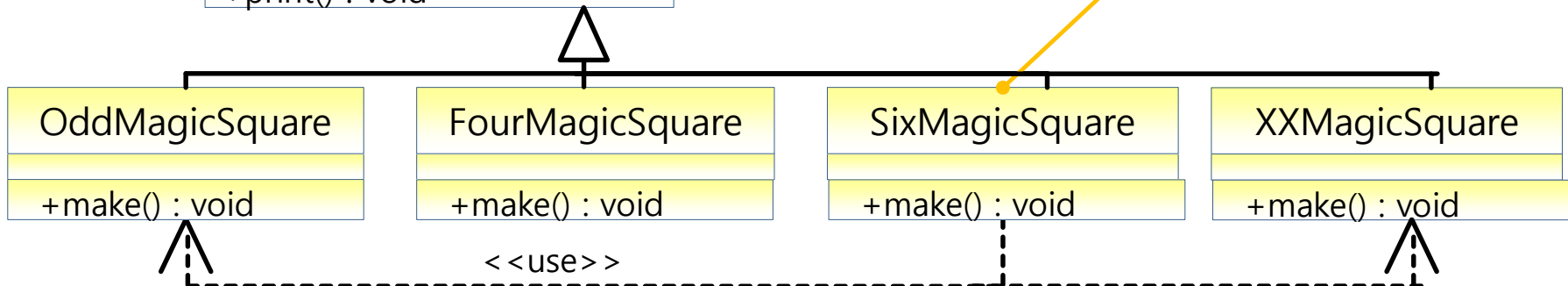
Singleton



factory(n) : IMagic
n값에 따라 Odd·Four·SixMagicSquare를 생성

MagicSquare abstract class : 이탤릭체로 표기
추상클래스-MagicSquare, 추상메서드-make()

make()메서드 내부에서 OddMagicSquare를
생성 후 getMagic()을 이용하여 OddMagicSquare
의 int[][]를 얻어온다



SixMagicSquare...

```
private void addMagic( ){
    int n=magic.length;

    OddMagicSquare odd=new OddMagicSquare(n/2);
    odd.make();

    int [][] mogi=odd.getMagic();
    for(int i=0 ; i<n/2;i++){
        for(int j=0;j<n/2;j++){
            magic[i][j] +=mogi[i][j];
            magic[i][j+n/2] +=mogi[i][j];
            magic[i+n/2][j] +=mogi[i][j];
            magic[i+n/2][j+n/2] +=mogi[i][j];
        }
    }
}
```

SixMagicSquare...

```
private void addMagic( ){
    int n=magic.length;

    XXMagicSquare odd=new XXMagicSquare(n/2);
    odd.make();

    int [][] mogi=odd.getMagic();
    for(int i=0 ; i<n/2;i++){
        for(int j=0;j<n/2;j++){
            magic[i][j] +=mogi[i][j];
            magic[i][j+n/2] +=mogi[i][j];
            magic[i+n/2][j] +=mogi[i][j];
            magic[i+n/2][j+n/2] +=mogi[i][j];
        }
    }
}
```

SixMagicSquare...

```
private void addMagic(IMagic odd){  
    int n=magic.length;  
  
    odd.make();
```

....

```
IMagic odd;  
public void setIMagic(IMagic odd){  
    this.odd=odd;  
}
```

```
private void addMagic( ){  
    int n=magic.length;  
  
    odd.make();
```

....

Java Developer's Forum

JavaCafe
Community

OOP 5대원리 ...

1. SRP(The Single Responsibility Principle)
 - 단 하나의 책임원리 → 응집도 관련
2. OCP(The Open-Closed Principle)
 - 개방 폐쇄의 원리 → 엔터티, 상속(부모 변경 지양)
 - A---> B (A를 변경하지 않고 B를 변경)
3. LSP(The Liskov Substitution Principle)
 - 리스코프 교체원리
 - 다형성 관련(부모로 자식을 사용-> 인터페이스 사용권장)
4. DIP(The Depend Inversion Principle)
 - 의존의 대상을 추상클래스나 인터페이스 권장
5. ISP(The Interface Segregation Principle)
 - 인터페이스 격리원칙
 - 필요한 메서드만 갖는 인터페이스를 구현하라.

OCP...

OCP

StudentController -----> StudentServiceImple

LSP...

StudentController -----> **StudentService**



LSP

StudentServiceImpl

DIP

StudentController -----> **StudentServiceImpl**

setStudentService(IService iss)

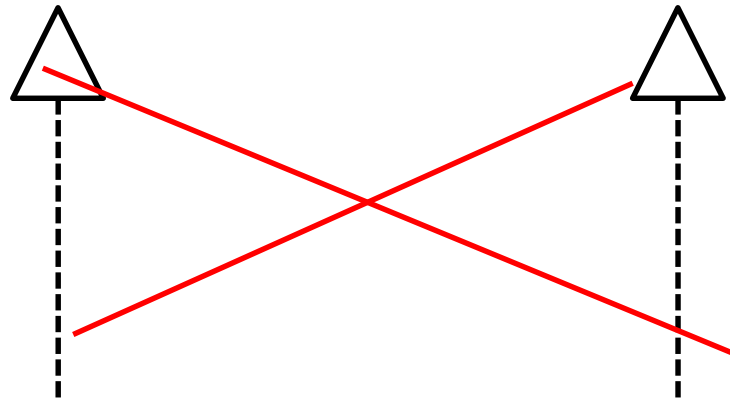
Java Developer's Forum

JavaCafe
Community

ISP...

TeacherService

StudentService



PeopleServiceImpl

ISP

ISP ...

StudentService



StudentServiceImpl

TeacherService



TeacherServiceImpl

Relation... 2.x

StudentController

SqlMapClientTemplate

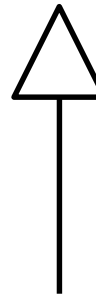
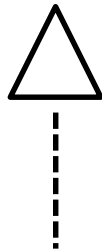
StudentService

StudentDao

SqlDaoSupport

StudentServiceImpl

StudentDaoImpl



Relation... 3.x

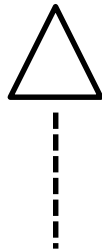
StudentController



StudentService

StudentDao

SqlSession



StudentServiceImpl

StudentDaoImpl

Controller ...

```
public class BoardListController extends AbstractController {
```

```
    private BoardRepService boardRepService;  
    public void setBoardRepService(BoardRepService boardRepService)  
    {  
        this.boardRepService = boardRepService;  
    }  
}
```

```
    protected ModelAndView handleRequestInternal(  
        HttpServletRequest request, HttpServletResponse response) throws  
        Exception {  
        List<BoardRepDto> list=null;  
        list=boardRepService.getAllBoards();  
        return new ModelAndView("list", "lists", list);  
    }  
}
```

Service...

```
public class BoardRepServiceImp implements BoardRepService {
```

```
    private BoardRepManager boardRepManager;
```

```
    public void setBoardRepManager(  
        BoardRepManager boardRepManager ) {  
        this.boardRepManager = boardRepManager;  
    }  
}
```

```
    public int deleteBoard(int seq) {  
        return boardRepManager.deleteBoard(seq);  
    }  
}
```

```
    public int deleteBoards(int[] seq) {  
        return boardRepManager.deleteBoards(seq);  
    }  
}
```

Dao ...

```
public class BoardRepManagerImp  
    extends SqlMapClientDaoSupport implements BoardRepManager {
```

```
    public BoardRepDto getBoard(int seq) {  
        BoardRepDto dto=new BoardRepDto();  
        dto=  
            (BoardRepDto)this.getSqlMapClient().  
                queryForObject("board.getBoard", seq);  
        return dto;  
    }
```

Wiring ...

```
<bean id="boardListController"  
class="com.board.controller.board.BoardListController"  
p:boardRepService-ref="boardRepService">  
</bean>
```

```
<bean id="boardUserService"  
class="com.board.servicImp.BoardUserServicImp"  
p:boardUserManager-ref="boardUserManager">  
</bean>
```

```
<bean id="boardRepManager"  
class="com.board.daolmp.BoardRepManagerImp"  
p:sqlMapClient-ref="sqlMapClient" />
```

```
<bean id="sqlMapClient"  
class="org.springframework.orm.ibatis.SqlMapClientFactoryBean"  
p:dataSource-ref="dataSource"  
p:configLocation="/WEB-INF/sqlMap/sqlMapConfig.xml"/>
```

3.X Controller ...

```
@Controller
public class JHBoardController {

@Autowired
private IUserService iUserService;
```

```
@RequestMapping(value = "/registry.do", method = RequestMethod.POST)
public String registry(JUser dto, Model model) {
    logger.info("Welcome JHBoardController registry.do !" + new Date());
    logger.info("Welcome JHBoardController " + dto);
    iUserService.addJUserAndRoll(dto);
    return "redirect:/index.do";
}
}
```


3.X Service...

@Service

public class JUserServiceImple implements IUserService {

@Autowired

private IUserDao iJUserDao;

@Override

@Transactional

public void addJUserAndRoll(JUser juser) {

 iJUserDao.addJUser(juser);

 iJUserDao.addJRollMap(juser);

}

3.x Dao ...

@Repository

public class JUserDaoImple implements IUserDao {

@Autowired

private SqlMapClientTemplate sqlMapClientTemplate;public

**SqlMapClientTemplate getSqlMapClientTemplate() {
 return sqlMapClientTemplate;**

}

@Override

**public void addJUser(JUser dto) throws DataAccessException{
 getSqlMapClientTemplate().insert("juserMap.addJUser",dto);**

}

@Override

**public void addJRollMap(JUser dto) throws DataAccessException{
 getSqlMapClientTemplate().insert("juserMap.addJRollMap",dto);**

}

Java Developer's Forum

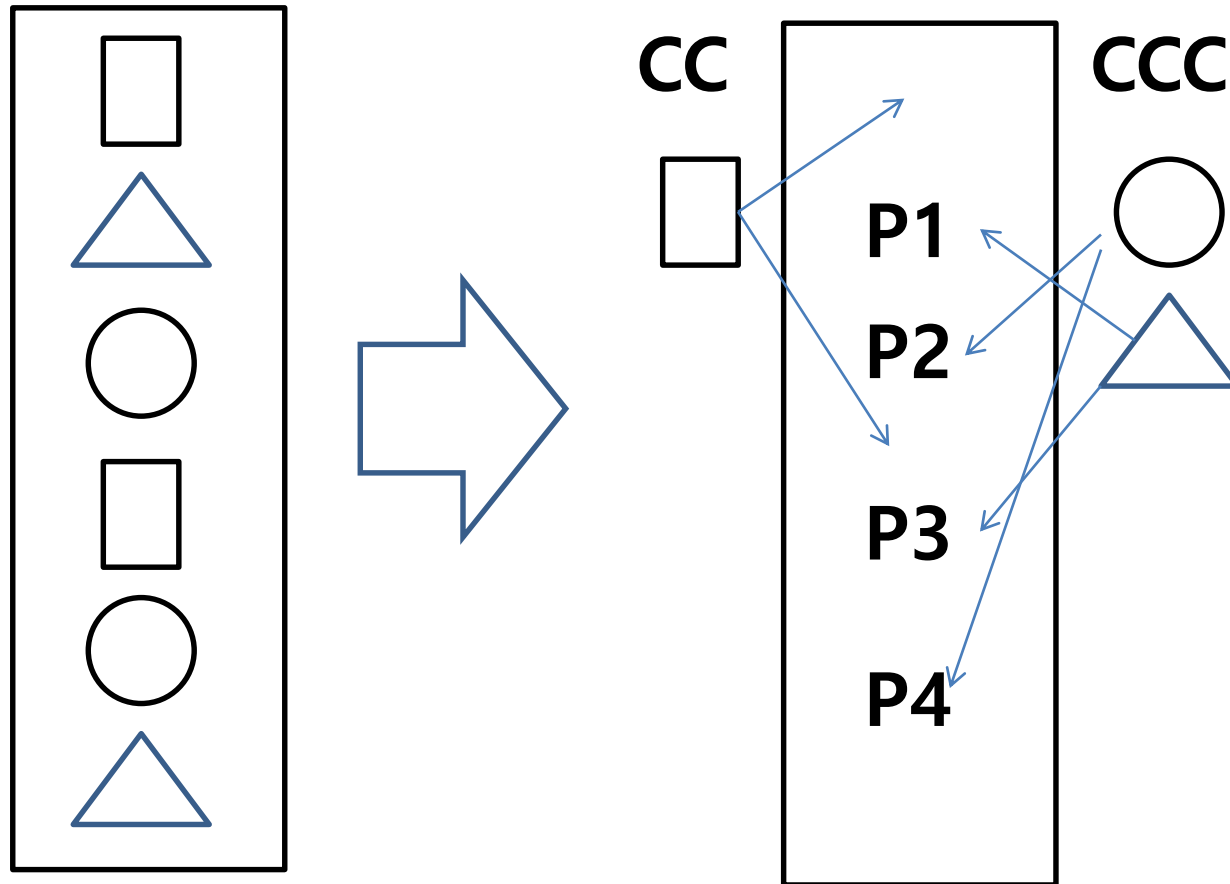
JavaCafe
Community

Auto Wiring Config...

```
<mvc:annotation-driven />
```

```
<context:component-scan base-package="com.hankyung.boards  
com.hankyung.datemans" />
```

AOP ...



weaving

JDBC Transaction...

```
public boolean addJUserAndRoll(JUser juser) {
    Connection conn=null;
    boolean isS=false;
    boolean isT=false;
    try {
        conn=getConnection();
        conn.setAutoCommit(false);
        isS=addJUser(juser,conn);
        isT=addRoll(juser,conn);
        if(isS&&isT){
            conn.commit();
        else{
            throw new SQLException("등록실패");
        }
    } catch (SQLException e) {
        try {
            conn.rollback();
        } catch (SQLException e1) {
        }
    } finally{
        try {
            conn.setAutoCommit(true);
        } catch (SQLException e) {
        }
        this.close(conn, null, null);
    }

    return isS&&isT;
}
```

```
public boolean addJUser(JUser juser,Connection  
con) throws SQLException{
```

```
    int count=0;
    String sql=" INSERT INTO
    J_USER(ID,NAME,PWD,ADDRESS,PHONE)" +
    " VALUES(?,?,?,?,?)";
    Connection conn=null;
    PreparedStatement psmt=null;
```

```
    try {
        conn=con;
        log("2/6 S addJUser ");
        psmt=conn.prepareStatement(sql);
```

```
public boolean addRoll(JUser juser,Connection  
con) throws SQLException{
```

```
    int count=0;
    String sql=" INSERT INTO
    J_ROLLMAP(ROLLMAPID,ID,ROLLID)" +
    " VALUES(J_ROLLMAP_SEQ.NEXTVAL,?,3)";
    Connection conn=null;
    PreparedStatement psmt=null;
```

```
    try {
        conn=con;
        log("2/6 S addRoll ");
        psmt=conn.prepareStatement(sql);
        int i=1;
        psmt.setString(i++, juser.getId());
        log("3/6 S addRoll ");
        count=psmt.executeUpdate
```

Transaction 3.x...

@Override

@Transactional

```
public void addJUserAndRoll(JUser juser) {  
    iJUserDao.addJUser(juser);  
    iJUserDao.addJRollMap(juser);  
}
```

```
<bean id="transactionManager"  
class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
```

```
<property name="dataSource" ref="dataSource"/>  
</bean>
```

```
<tx:annotation-driven transaction-manager="transactionManager"/>
```

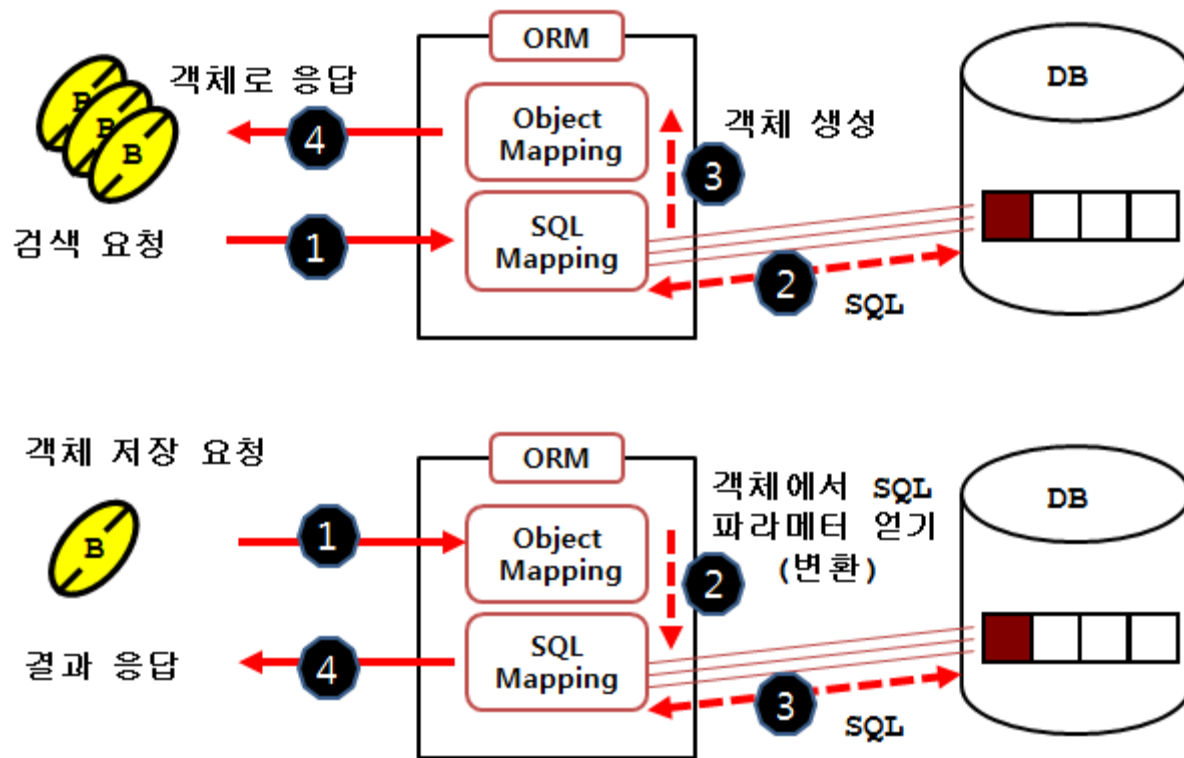
Java Developer's Forum

JavaCafe
Community

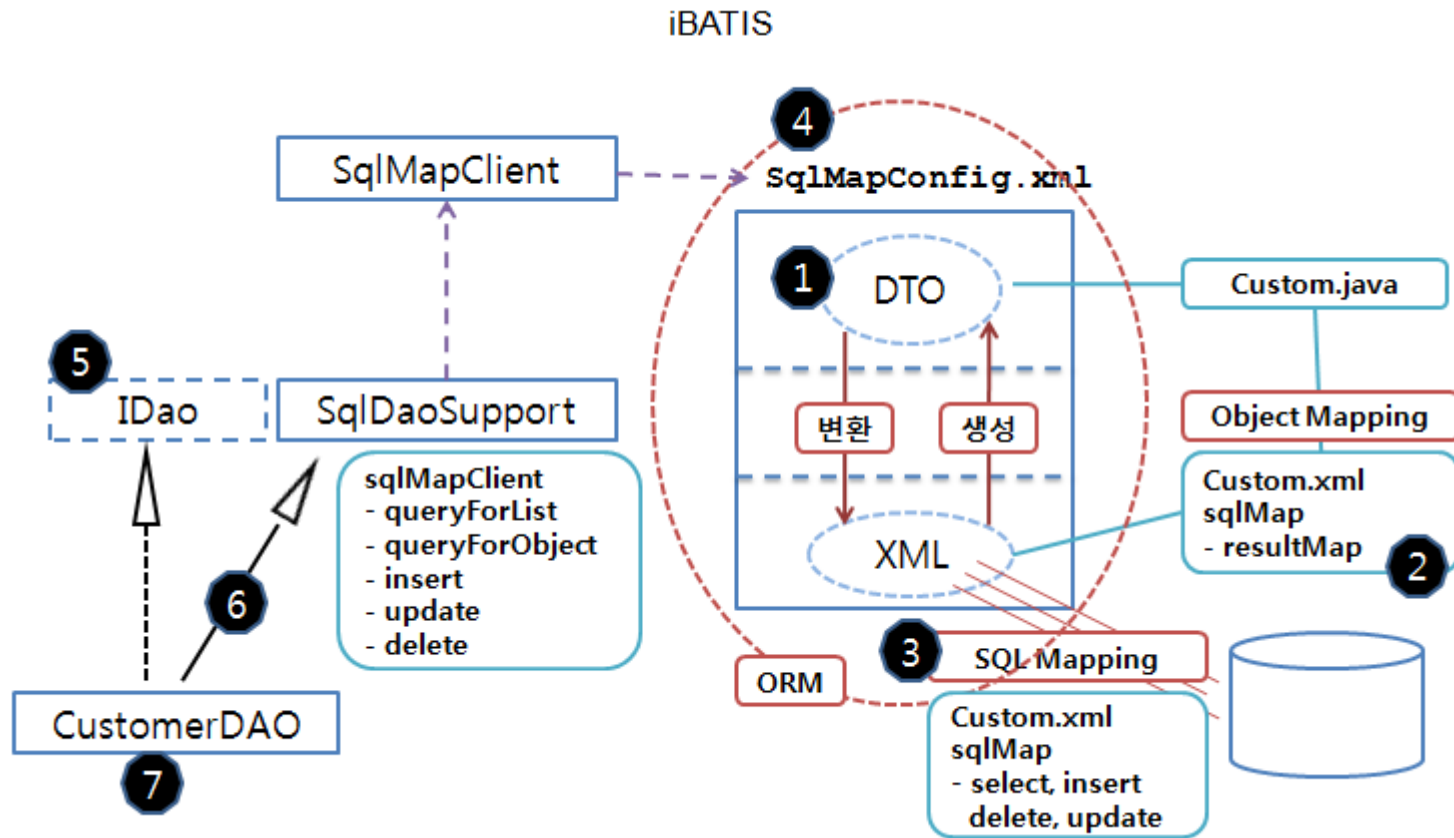
Transaction 2.5 ...

```
<tx:advice id= "transactionAdvice" transaction-  
manager="transactionManager">  
  <tx:attributes>  
    <tx:method name= "get*" read-only="true" />  
    <tx:method name= "*" propagation="REQUIRED" rollback-  
for="org.springframework.dao.DataAccessException" />  
  </tx:attributes>  
</tx:advice>  
  
<aop:config proxy-target-class= "true">  
  <aop:advisor advice-ref= "transactionAdvice" pointcut="execution(*  
com.hankyung.boards.model.*Service.*(..))" />  
</aop:config>
```

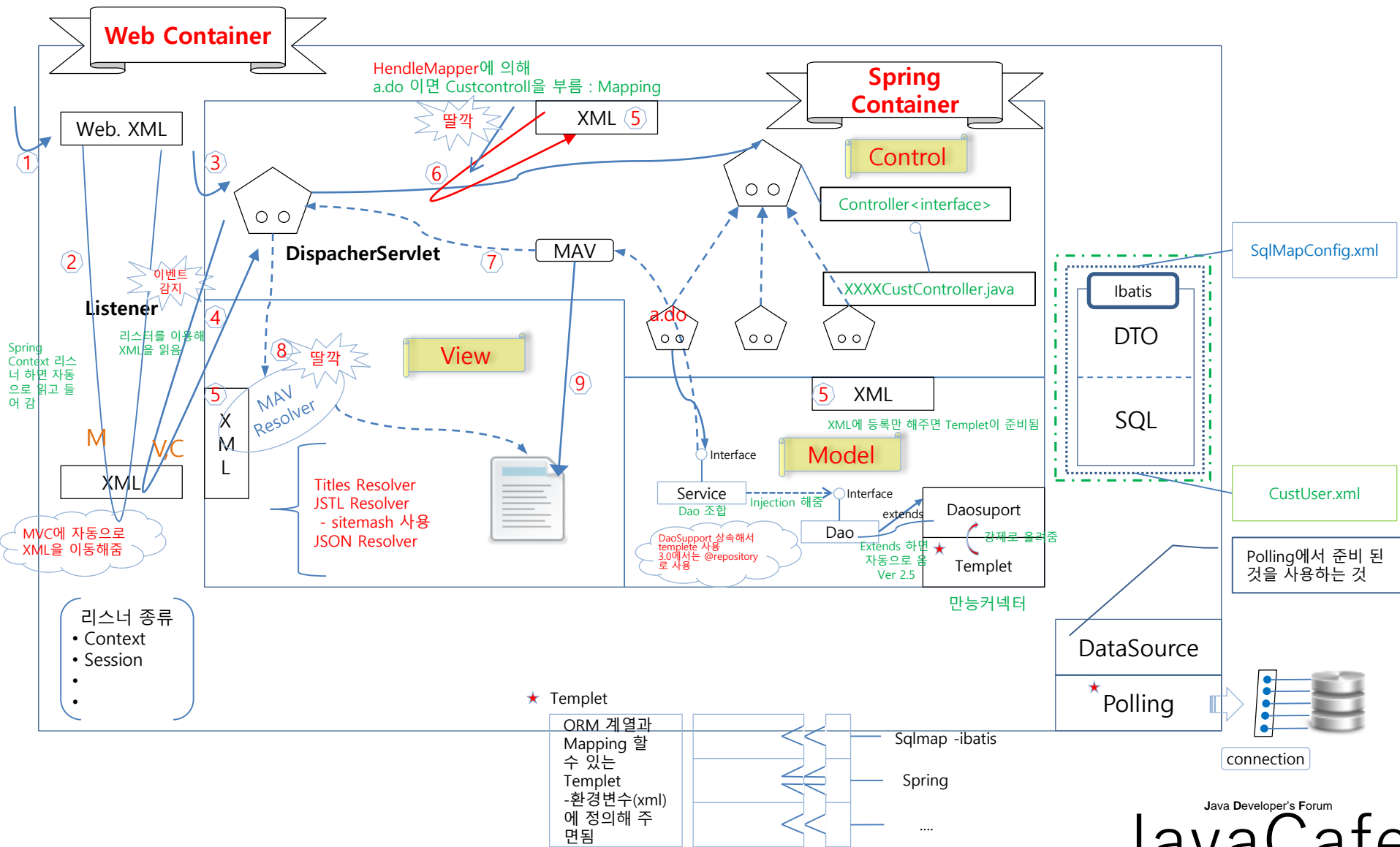
ORM...



MyBatis...



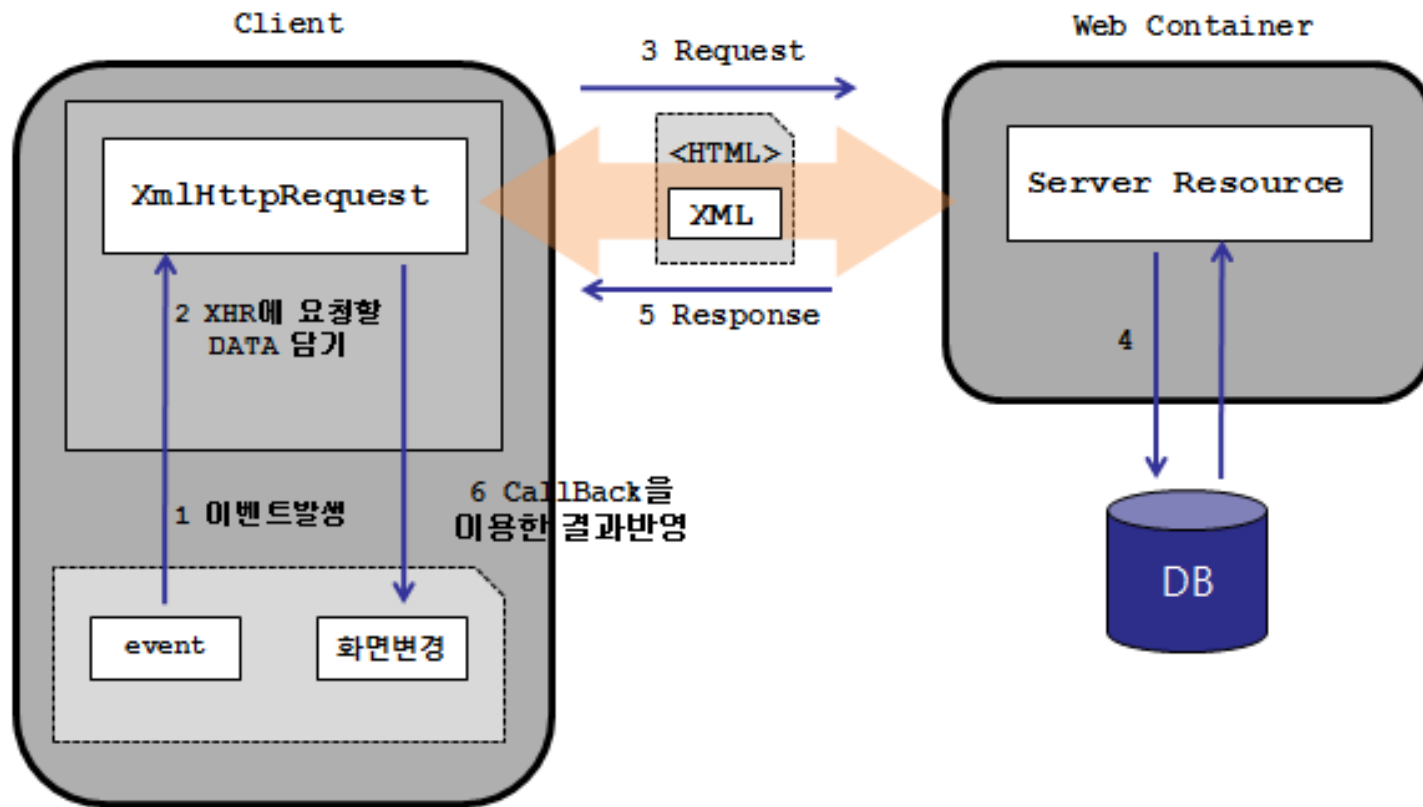
Spring MVC...



Today Story ...

1. 티어와 레이어
2. 웹 프로그래밍과 엔터프라이즈 프로그래밍
3. MVC 모델과 웹 개발의 흐름
4. Spring 3대 구성원리와 디자인패턴 5대 원리
5. AJAX와 데이터 처리

Round Trip ...



AJAX Data...

CSV

@@@Jungbo,15400,1100@@Hankyung,26200,1000@@@

XML

```
<kosdaq>
  <stock>
    <stockname>Jungbo</stockname>
    <stockprice>15400</stockprice>
    <prevstockprice>1100</prevstockprice>
  </stock>
  <stock>
    <stockname>Hankyung</stockname>
    <stockprice>26200</stockprice>
    <prevstockprice>200</prevstockprice>
  </stock>
</kosdaq>
```

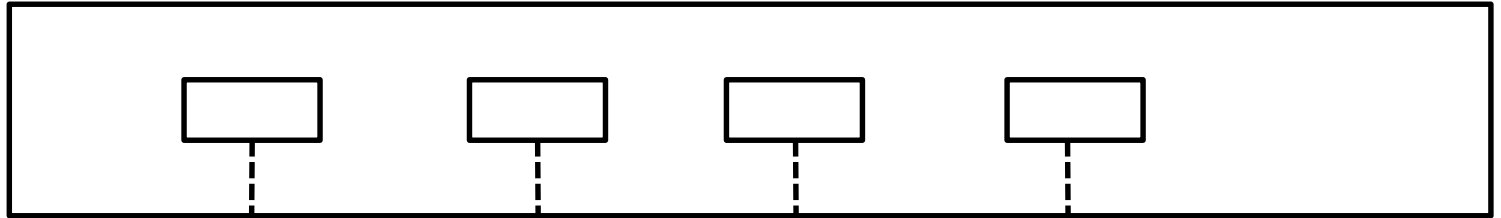
AJAX Data...

JSON

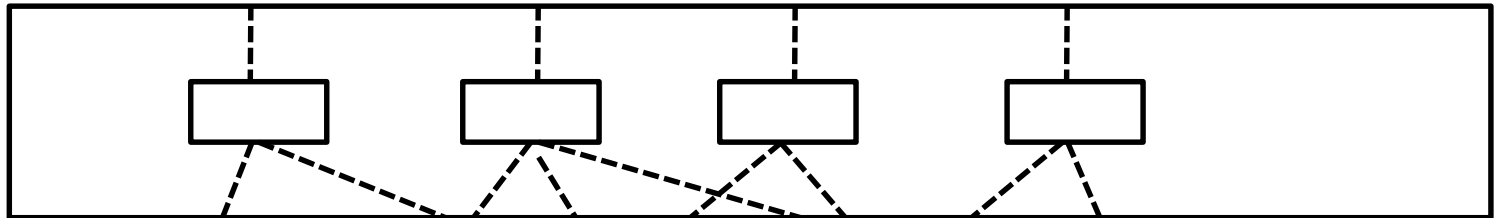
```
{ "kosdaq": { "stock" : [  
    { "stockname": "Jungbo",  
      "stockprice": "15400",  
      "prevstockprice": "1100" },  
    { "stockname": "Hankyung",  
      "stockprice": "26200",  
      "prevstockprice": "1000" }  
  ]  
}
```

CBD ...

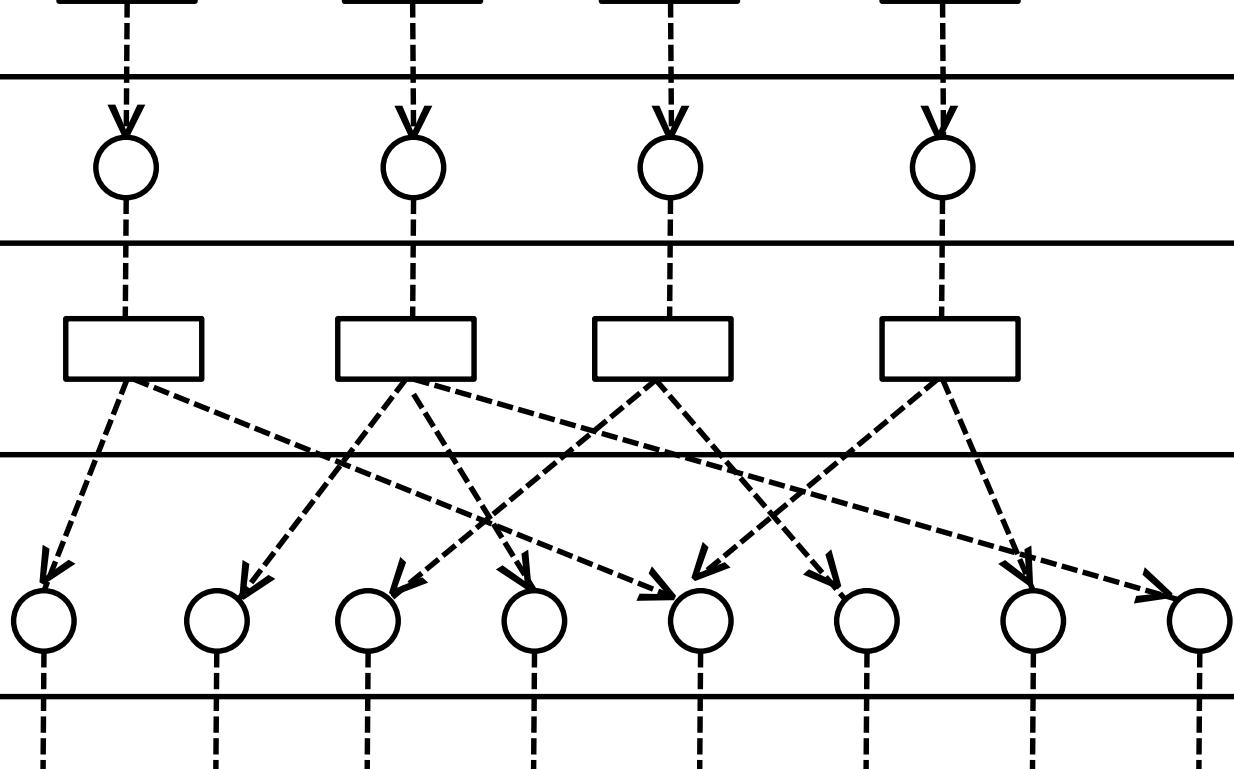
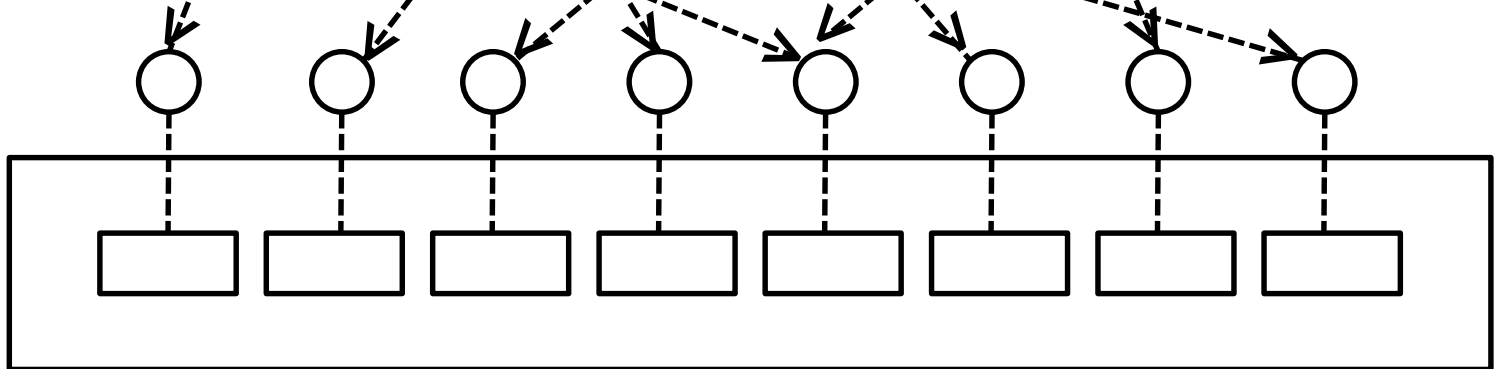
Control



Service



DL



Java Developer's Forum

JavaCafe
Community