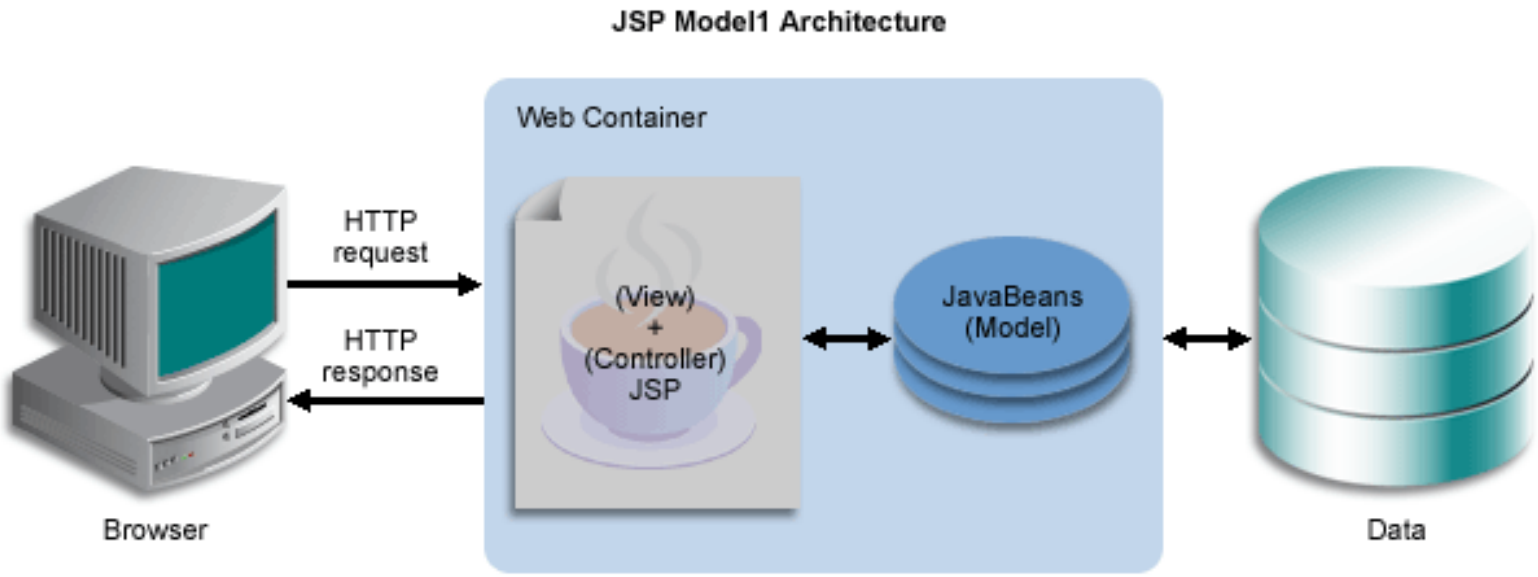
Introduction to MVC

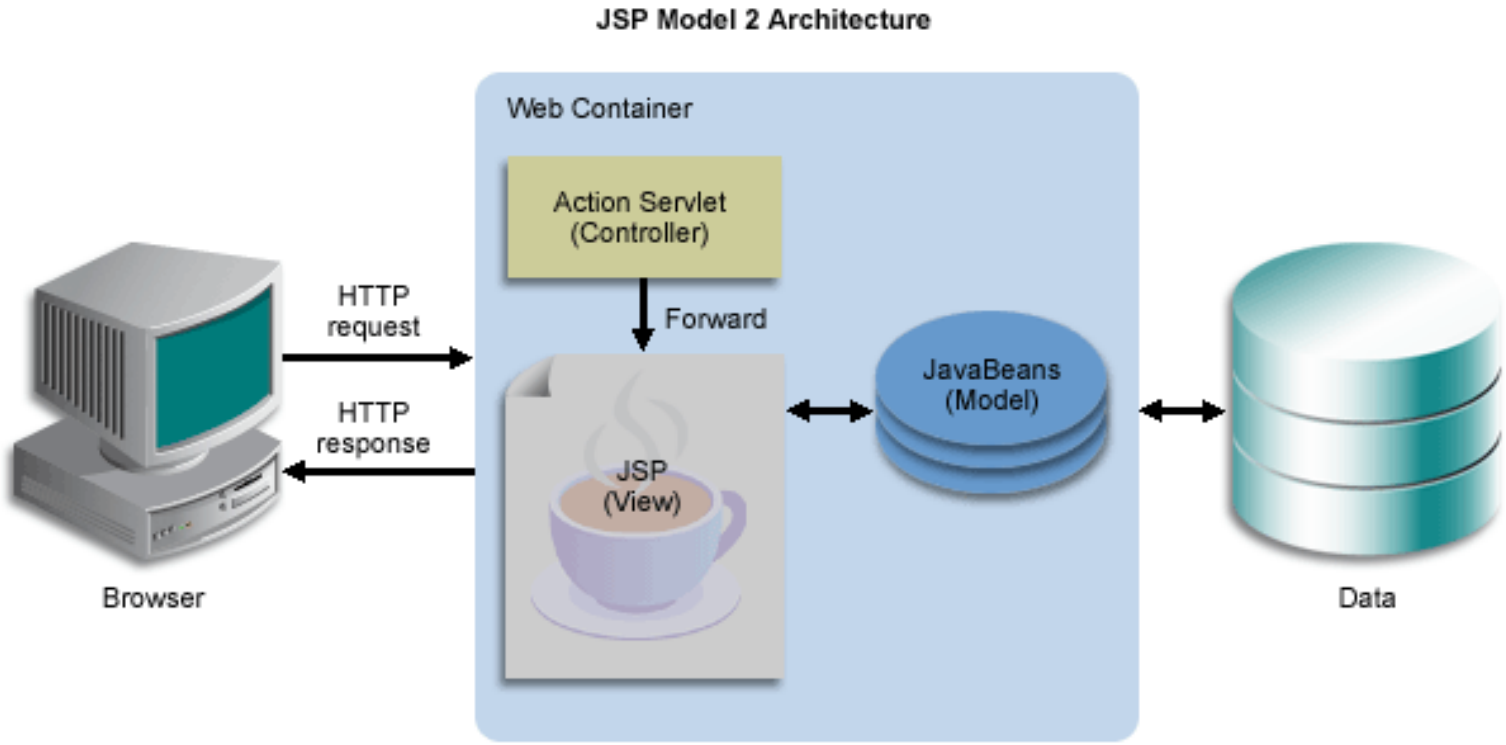
Model 1 vs Model 2

- Model 1은 단일 처리기가 컨트롤러 역할과 뷰 역할 수행



Model 1 vs Model 2

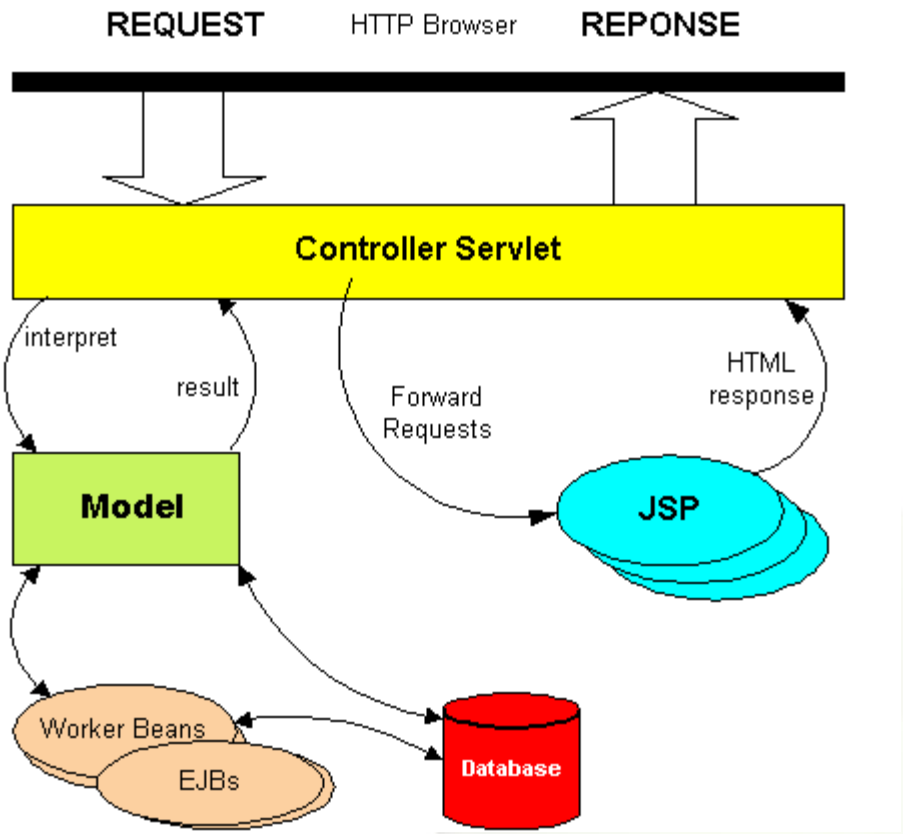
- Model 2는 컨트롤러 역할과 뷰 역할을 별개의 처리기에서 수행



Model-View-Controller Pattern

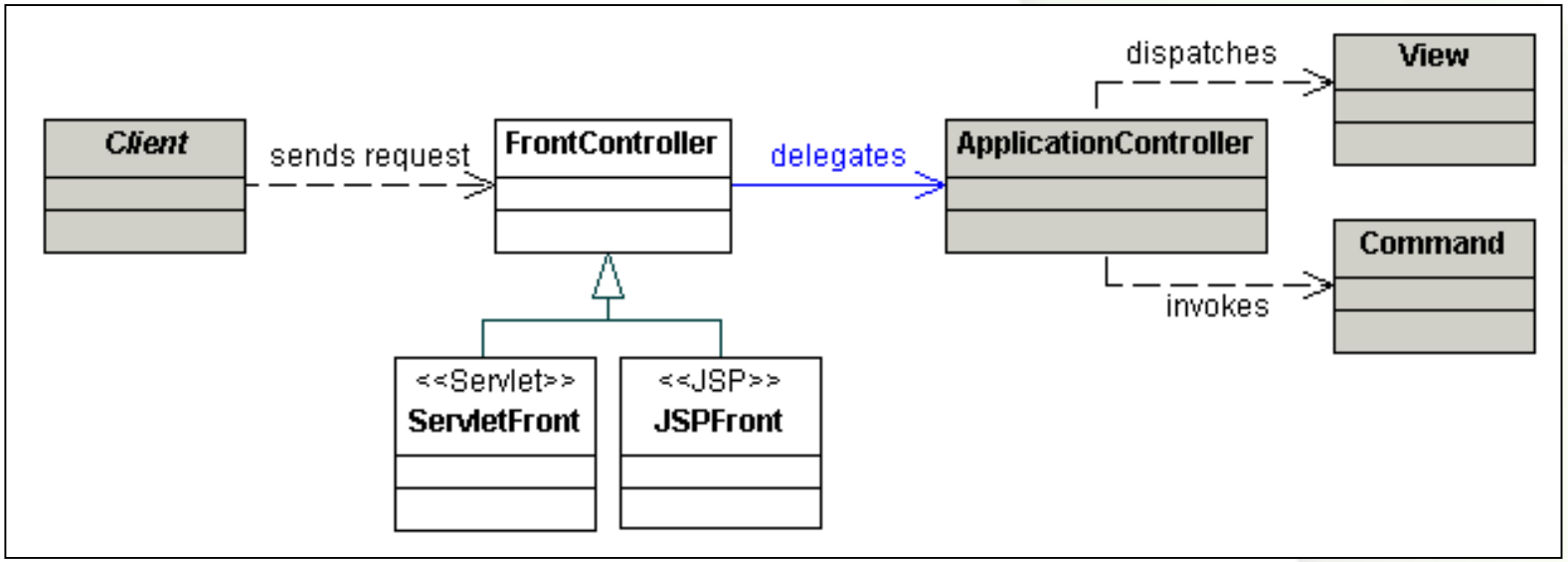
- 아키텍처 패턴의 하나로 애플리케이션 전체를 모델, 뷰, 컨트롤러의 관점으로 구분하는 패턴

- 구성요소
  - 모델
    - 개발자가 처리할 데이터를 표현하는 클래스
    - 데이터의 변경과 조작을 위한 비즈니스 규칙을 표현하는 클래스
  - 뷰
    - 애플리케이션의 사용자 인터페이스 영역
  - 컨트롤러
    - 사용자와의 커뮤니케이션 (외부의 요청 수신)
    - 애플리케이션의 전체적인 흐름 관리



Front Controller Pattern

- 모든 요청을 단일 또는 소수의 FrontController를 통해 처리하는 구조
- 주로 웹 애플리케이션 구현에 적용
- 작동 방식
  - 요청이 발생하면 프런트 컨트롤러에게 전달
  - 프런트 컨트롤러는 요청 분석 결과에 따라 해당하는 개별 컨트롤러 호출
  - 개별 컨트롤러는 모델 영역의 객체와 연동해서 요청 처리
  - 처리 결과에 따라 뷰 선택 및 호출
  - 뷰는 요청 로직 처리 결과를 표시하는 화면을 생성하고 클라이언트로 반환



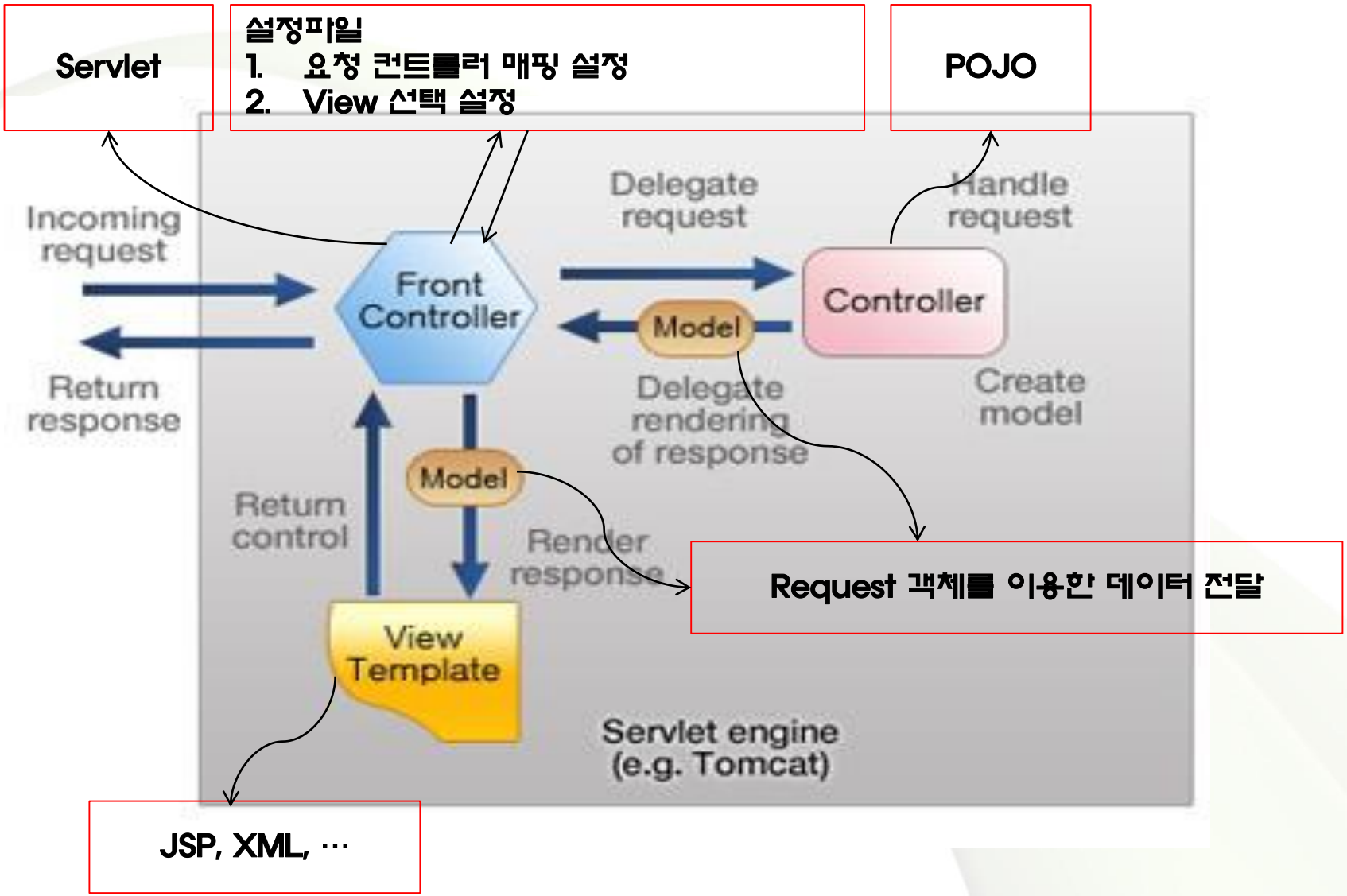
MVC 패턴의 장/단점

- 장점
  - 재사용성 향상
    - 예를 들어 하나의 비즈니스 로직 컴포넌트를 사용하는 웹 페이지와 윈도우 애플리케이션 개발 가능
  - 유연성과 확장성 향상
    - 한 영역의 변화가 다른 영역에 미치는 영향을 최소화 해서 변경이 용이한 구조
    - 새로운 기능을 쉽게 추가하고 적용할 수 있는 구조

- 단점
  - 복잡성의 증가로 구현의 난이도가 높은 구조
  - 모든 컴포넌트간의 상호작용을 고려할 필요가 있는 구조

- 단점을 보완하는 다양한 프레임워크 개발
  - 스트럿츠 / 스프링
  - 루비 온 레일즈 / 장고와 파이썬, PHP용 젠드 프레임워크
  - 모노레일 (ASP.NET을 MVC 프레임워크) / ASPNET MVC

MVC 구현 – 구조 및 관련 구현 객체



MVC 구현 – Front Controller Servlet 선언 및 요청 매핑

- web.xml을 이용한 서블릿 등록 및 요청 매핑

```
public class FrontController extends HttpServlet {
        @Override
        public void doGet(HttpServletRequest req, HttpServletResponse res)
        throws ServletException, IOException {
                //Code will go here
        }
        @Override
        public void doPost(HttpServletRequest req, HttpServletResponse res)
        throws ServletException, IOException {
                doGet(req, res);
        }
}
```

```
<servlet>
        <servlet-name>FrontController</servlet-name>
        <servlet-class>
                com.examples.mvc.controllers.FrontController
        </servlet-class>
 </servlet>
 <servlet-mapping>
          <servlet-name>FrontController</servlet-name>
          <url-pattern>/controller/*</url-pattern>
 </servlet-mapping>
```

- 어노테이션을 이용한 요청 매핑

```
@WebServlet( "/controller/*" )
public class FrontController extends HttpServlet {
}
```

MVC 구현 – Front Controller Servlet 구현

- Servlet 클래스의 요청 처리 메서드 구현

```
public void doPost(HttpServletRequest req, HttpServletResponse res)
throws ServletException, IOException {

    String action = req.getParameter("action");
    if (action.equals("action1")) {
            //Model 객체 생성 및 메서드 호출
            //결과에 따라 View 페이지 선택 및 forward
    } else if (action.equals("action2")) {
            //Model 객체 생성 및 메서드 호출
            //결과에 따라 View 페이지 선택 및 forward
    }
    //...
    else {
            //기타 요청 처리
    }

}
```

MVC 구현 – View 구현 및 Controller/View 간 데이터 전달

- Model 또는 Controller영역의 데이터 전달
  - Controller

```
request.setAttribute( “key” , value);
```

  - View

```
Object data = request.getAttribute( “key” );
```

- View 페이지 구현

```
<%@ page language="java" contentType="text/html; charset=EUC-KR"
    pageEncoding="EUC-KR"%>

<html>
<head>
        <title>제목</title>
</head>
<body>
        요청 처리 결과 표시
</body>
</html>
```