**SPRING-WEB-MVC:**

The SPRING-WEB module serves as an environment that facilitates the integration of other MVC-based frameworks and applications with Spring applications.

**Example:** To integrate Struts applications with the Spring Framework, the SPRING-WEB module is utilised. Similarly, to integrate JSF applications with the Spring Framework, the SPRING-WEB module is employed.

**Note:** It is important to note that the SPRING-WEB module is considered an outdated module within the Spring Framework, as most MVC-based frameworks, such as Struts, JSF, and Xwork 2, have been superseded.

The Spring Framework provides its own implementation of MVC through the SPRING-WEB MVC module, which can be directly utilised to develop MVC-based applications.

**Question:** Given that we already have established MVC frameworks like Struts, JSF, etc., what is the necessity to employ the SPRING-WEB-MVC module?

**Answer:**

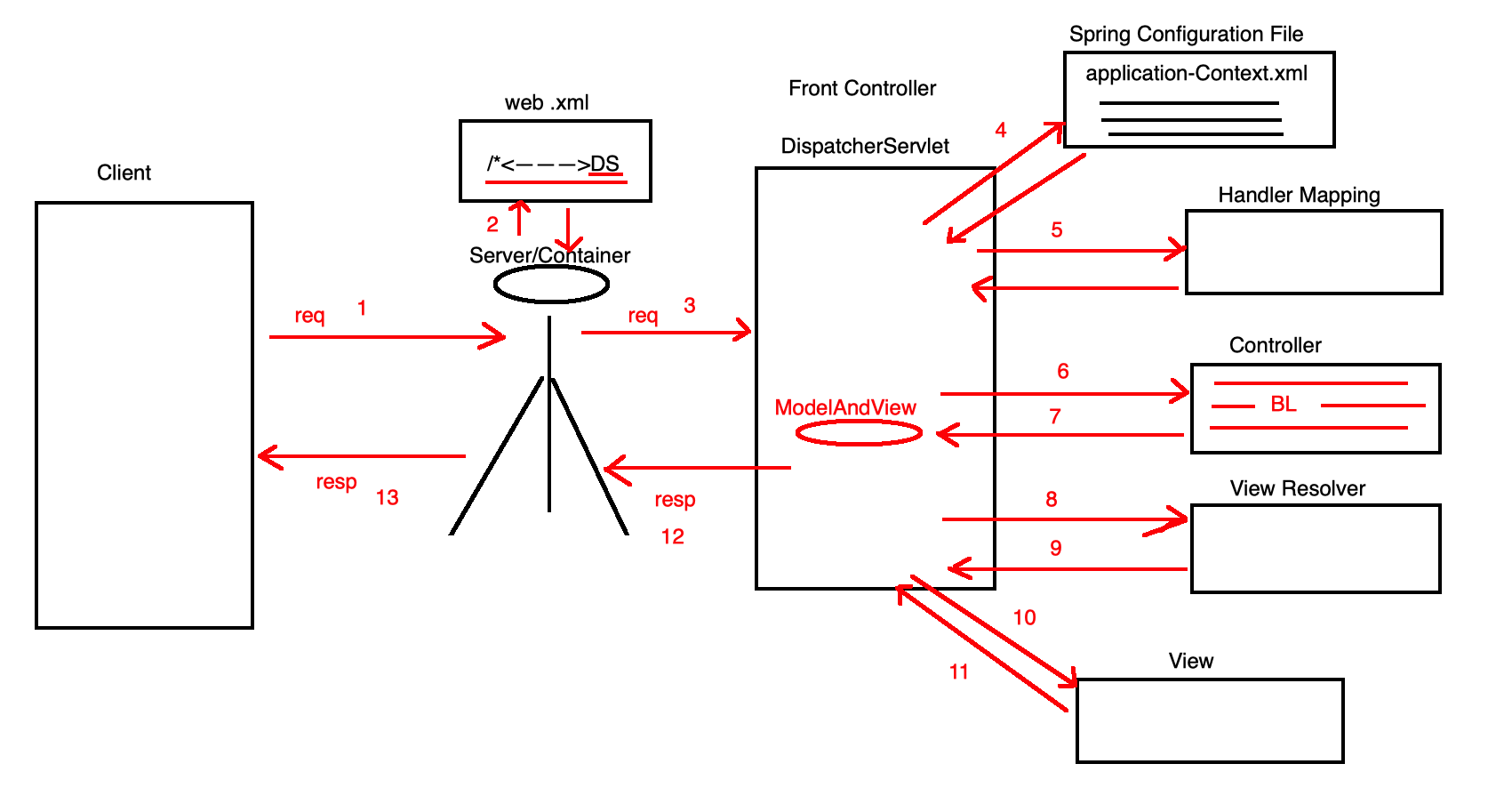
1. Struts and JSF are two web frameworks that are capable of focusing solely on the web layer in enterprise applications. Spring Framework, on the other hand, is an application framework that can focus on all layers, including the user interface layer, business processing layer, and persistence layer, of enterprise applications.
2. Struts and JSF are considered heavyweight frameworks, while Spring Framework is a lightweight framework.
3. Struts and JSF employ tightly coupled design, while Spring Framework employs loosely coupled design for applications.
4. Struts and JSF adhere exclusively to the MVC design pattern, whereas the Spring Framework incorporates a diverse range of design patterns, including MVC.
5. 6. While Struts and JSF do not provide a distinct separation between the Controller, View, and Model layers, the Spring Framework offers a clear distinction between these layers.
6. In terms of API dependency, Struts and JSF exhibit higher dependency, whereas the Spring Web MVC framework is less API-dependent.
7. Although Struts and JSF lack integration layers for integrating other technology applications such as EJBs and Hibernate, the Spring Framework provides dedicated integration layers for these applications.
8. AOP implementations are not available in Struts and JSF, but the Spring Framework incorporates AOP internally.
9. Struts and JSF are not modularized frameworks, whereas the Spring Framework is a modularized framework.

**Spring WEB MVC Features:**

1. Spring WEB MVC has a number of components like Controller, Service, Model, Command objects,..., all these components have their own functionalities.
2. Spring WEB MVC Framework is allowing the command objects in order to manage the user form data and in order to manage the reusable business code instead of extending the Frameworks provided libraries.
3. Spring WEB MVC has very good Data binding mechanisms and the Data Validations for the user data.
4. Spring WEB MVC is providing customizable Handler Mappings, ViewResolvers,...
5. Spring WEB MVC is supporting the View technologies like JSTL, EL, Velocity, Freemarker,...

**SPRING WEB MVC COmponents:**

1. View
2. web.xml
3. DispatcherServlet
4. HandlerMapping
5. Controller Component
6. Command Classes
7. ViewResolver
8. Spring COnfiguration File



1. Client Sends a request to the Server.
2. Server or Container will goto the web.xml file and find the name and location of the Front controller that is DispatcherServlet.
3. Server/Container will send the request to the DispatcherServlet.
4. DispatcherServlet will goto the Spring Configuration File and find the name and location of the Handler mapping.
5. DispatcherServlet will goto the Handler Mapping to find the name and location of the Controller component.
6. DispatcherServlet will execute the Business logic which is defined in the controller component.
7. Controller Component will return a Model and View object with the View logical name.
8. DispatcherServlet will goto the ViewResolver to find the name and location of the View component.
9. ViewResolver will send the name and location of the View component to teh DispatcherServlet.
10. DispatcherServlet forward request to the View component, execute the view component.
11. View component will send the generated response to the DispatcherServlet.
12. DispatcherServlet will send the response to the Server or Container.
13. Server or Container will dispatch the response to the respective client.