A Web Container with Event Driven, Adaptive, Differentiated Services Web Container Architecture

By Lakshman Kishore
UTD ID: 2021126336

Under the guidance of
Dr. Lawrence Chung
as part of the course on
Advanced Software Architecture and Design
Introduction

A **Web Container** provides a runtime environment for Servlets, JSP files, according to JEE specification. It does the task of loading, initializing and terminating Servlet instances. Also manages the creation and life cycle of the Request and Response objects.

**Request Processing:**

Upon the arrival of a request from a client, a processing thread is assigned to the request. The thread then encapsulates the request into a Request Object, which is then passed through a predefined request processing flow. Response object is returned to the client.

**Disadvantages of the Traditional architecture:**

1. Just one processing thread is used, therefore if blocked, results in **poor concurrency**.
2. Does not differentiate the requests.
3. Cannot **reconfigure** request flows for various requests flexibly.
4. The **Processing modules** themselves lack the ability to provide service with different QoS.

Features of the new Architecture:

1. Means to classify Client Requests
2. Provides Different Request processing flow to different types of Requests
3. Provides differentiated services for different processing modules

The new Architecture EADWA adopts a Service Level Agreement (SLA) for providing the above features. The SLA stipulates:

1. Service Levels
2. Classification rules
3. Request types and priorities
Components of a new Architecture

1. Self Managed unit
   It encapsulates the old processing module to work for requests but also manages its own thread pool and processing events. It also configures the differentiated service strategy which is based on a SLA descriptor.

2. Request event
   It is a 2-tuple, defined as
   \[ \text{request-event} = \langle \text{request-type}, \text{request-object} \rangle \]

   It connects different SMUs in creating a request processing flow.

3. SMU Manager
   This manages all the SMUs in the web container and creates, configures all SMUs and SMU chains. The first SMU in a chain will create a Request-event.
Elements of an SMU

1. In-Event queue
2. Event Manager: creates Classification Event Queues for each request type.
3. Thread pool
4. Context setter: sets the context of a processing thread based on its type.
5. Scheduler: manages the thread pools, and assigns thread from the pool to each request in the queue.
6. Dispatcher: dispatches events to the next SMU, based on dispatch-rules, which are as:

\[ \text{dispatch-rule} = \langle \text{request-type, target-SMU} \rangle \]

By adopting different strategies for Access Control, Thread scheduling and context setting using the SMU Differentiated services can be implemented which overcomes the aforementioned deficiencies with the Traditional Web container Architecture.
References:

1. An Event-Driven Adaptive Differentiated Service Web Container Architecture by
   Yang Li, Chinese Academy of Sciences, China
   Ningjiang Chen, Guangxi University, China
   Tao Huang, Chinese Academy of Sciences, China

   http://doi.ieeecomputersociety.org/10.1109/QSIC.2006.12

2. Java EE technologies