DESENVOLVIMENTO COM JAVA EE E SUAS ESPECIFICAÇÕES

Hugo Henrique Rodrigues Correa[1], Jaime Willian Dias [1]

Universidade Paranaense (Unipar)

Paranavaí – PR–Brasil

hugohrcorrea@gmail.com, jaime@unipar.br

Resumo. De acordo com as instruções sobre abstract, vamos construir o resumo e em seguida o Abstract

Abstract. *According to the summary instructions, we will construct the summary then the Summary*

1. Introdução:

O Java EE são várias especificações que quando implementadas, que auxiliam o desenvolvimento da uma aplicação, além de não se preocupar com código grande, sendo ele um grande PDF, uma especificação, mostrando quais fazem parte deste. Para que possamos usar algum software, teríamos que baixar uma implementação dessas descrições. Exemplo o Glassfish, desenvolvida pela Sun/Oracle, é um servidor open source e gratuito, porém não é o líder de mercado apesar de ganhar força nos últimos tempos [Caelum, 2015].

Existem umas diversas variação de aplicação web, que são compatíveis com a cômputo do J2EE 1.4, Java EE 5 e alguns já do Java EE 6. O JBoss tem sido uns dos primeiros que estão no topo do mercado e além de ser uma ferramenta gratuito e open source. Algumas aplicações são implementam apenas por uma dessas especificações do Java EE, como o Apache Tomcat, que só implementa JSP e Servlets, além de afirmar que não é correto chamá-lo de servidor de aplicação. A partir do Java EE 6, existe o termo "application server web profile", onde se referência os servidores que não oferecem tudo, mas um grupo menor de especificações, consideradas essenciais para o desenvolvimento web [Caelum, 2015].

O objetivo desse artigo é mostrar que a aplicação web hoje, está sendo muito valorizada, por causa de suas capacidades multi-plataforma, o seu modelo distribuído para a interação da Web, as suas capacidades de multi -threading, e sua independência de plataforma, além da sua rapidez. Além de não estar preso a um código, a especificação garante que sua aplicação funcionará com a implementação de outro fabricante, sendo um

atrativo muito grande para empresas e governos, que querem evitar o vendor lock-in [Caelum, 2015].

2. Metodologia

A metodologia usada para executar este artigo foi uma revisão bibliográfica em materiais como livros, artigos, apostila e sites da Internet. O passo posterior será gerado a partir da matriz simplificada, ou seja, em partes menores, que ao final, serão unidas para compor o projeto final.

3. Desenvolvimento 3. 1 Como o Java EE pode ajudar a enfrentar problemas

Hoje as aplicações Web possuem regras de negócio completamente complexa e ainda codificá-las já representam um grande trabalho.

Como requisitos funcionais, existem outros requisitos que precisam ser atingidos através de planejamento: persistência em banco de dados, transação, gerenciamento de conexões HTTP, acesso remoto, entre outros. São chamados de requisitos nãofuncionais. Se fossemos responsáveis por escrever código que se trata desses outros tipos de requisitos, teria mais trabalho a fazer. Visando isso, a Sun criou uma série de especificações que, quando implementadas, pode ser retirada um bom proveito e reutilizar toda essa infraestrutura. O Java EE (Java Enterprise Edition) consiste em uma série de especificações, de como deve ser implementado um software que faz cada um desses serviços [Caelum, 2015].

3.2 Containers/Arquitetura Java EE

Segundo Gonçalves, cada container tem vários de especificações implementadas no Java EE. Containers são ambientes que executam o Java EE, que provê os componentes construídos como Servlets, objetos dentro de outros objetos, que o desenvolvedor passa incluir ou remove-los, além de trazerem benefícios aos componentes que eles hospedam, gerenciando um ciclo de vida, injeção de dependências, e assim por diante. Estes componentes usam contratos bem definidos para se comunicarem com a infraestrutura do Java EE e com outros componentes.

Cada container tem uma função própria, que suporta várias APIs juntas, oferecendo serviço aos componentes. Cada função tende de ocultar a complexidade técnica e melhorando a sua portabilidade [Gonçalves, 2011].

3.3 Serviços

Java Transaction API (JTA): oferece uma API de demarcação de transações usada pelo container e pela aplicação [Gonçalves, 2011].

Java Persistence API (JPA): API padrão da linguagem Java que faz um mapeamento objeto-relacional (ORM), para frameworks de persistência de dados. Com sua Java

Persistence Query Language (JPQL), pode consultar entidades armazenadas no banco de dados subjacente [Gonçalves, 2011].

Validação: a validação de beans, permite escrever, expressar restrições, além de facilitar validação e declaração de restrições em nível de classe [Gonçalves, 2011].

Java Message Service (JMS): é uma linguagem Java orientada às mensagens, que permite que os componentes se comunicar assincronamente através de mensagens. Tendo a suporta cessão confiável de mensagens ponto-a-ponto (P2P) [Gonçalves, 2011].

Java Naming and Directory Interface (JNDI): é utilizada para o acesso de recursos externo de diretórios e de names services. É usado para ligar/associar nomes a atributos e encontrar esses objetos em um diretório [Gonçalves, 2011].

Java Mail: oferece uma estrutura independente de plataforma e protocolo, que não exige habilidade de envio de e-mails, o que pode ser implementado pelo uso da API do JavaMail [Gonçalves, 2011].

Java Beans Activation Framework: oferece um serviço de padrões que determina uma estrutura para tratamento de dados em diferentes tipos, para descrever o conteúdo das mensagens e download, que bem usada pelo Java Mail [Gonçalves, 2011].

Processamento de XML: a API do Java para processamento de XML (JAXP) fornece suporte para gerar linguagem de marcação para o processamento de documentos com as APIs de SAX e DOM, bem como para XSLT. A Streaming API for XML (StAX) fornece uma API de pull-parsing para XML [Gonçalves, 2011].

Java EE Connector Architeture (JCA): é considerado um componente que conecta um servidor de aplicação a um EIS, de um componente Java EE. Este pode ser um programa de base de dados, de mainframes, ou de Enterprise Resource Planning (ERP) [Gonçalves, 2011].

Serviços de segurança: o Java Autthentication and Authorization Service (JAAS) permite aplicações escrita na plataforma usar serviços que faça a autenticação e reforça os controles de acesso de autorização dos usuários. O Contrato do Provedor de Serviços de Autorização para Containers do Java (JACC) de acordo com a documentação oficial do Java EE “define um contrato entre um servidor de aplicação Java EE e um provedor de serviços de autorização”, permitindo que provedores de serviços sejam autorizados e conectados a qualquer produto Java EE [Gonçalves, 2011].

Serviços Web: o Java EE provê componentes que permitem receber dados em XML, podendo ser acessado por outras aplicações para serviços web como SOAP e RESTful.

A API do Java para Serviços Web por XML (JAX-WS), substituindo a API do Java para RPC baseado em XML (JAX-RPC), fornece suporte para criação de serviços web usando o protocolo SOAP/HTTP. A API do Java para Serviços Web RESTful (JAXRS) tende a fornece suporte para aplicação web usando o estilo REST [Gonçalves, 2011].

Injeção de dependências: basicamente o ele define um mecanismo seguro de injeção de dependência na plataforma, o Java EE 5, já usa esses tipos de recursos (fontes de dados,

produtores de JMS, unidades de persistência, EJBs.), podem ser emitidos em componentes. O Java EE 6 e o mais atualizado que temos hoje, ele vem usando as especificações de CDI (Injeção de dependência e contexto) bem como a DI (Injeção de dependência para Java) [Gonçalves, 2011].

Gerenciamento: o Java EE define APIs para gerenciando os dados utilizando o mapeamento relacional de objetos. A API Extensões de Gerenciamento do Java (JMX) também usa suporte de gerenciamento para gerenciar e monitorar as aplicações [Gonçalves, 2011].

Abaixo as imagens com as especificações dos serviços contidos em cada container. Estas imagens foram extraídas do livro de Gonçalves

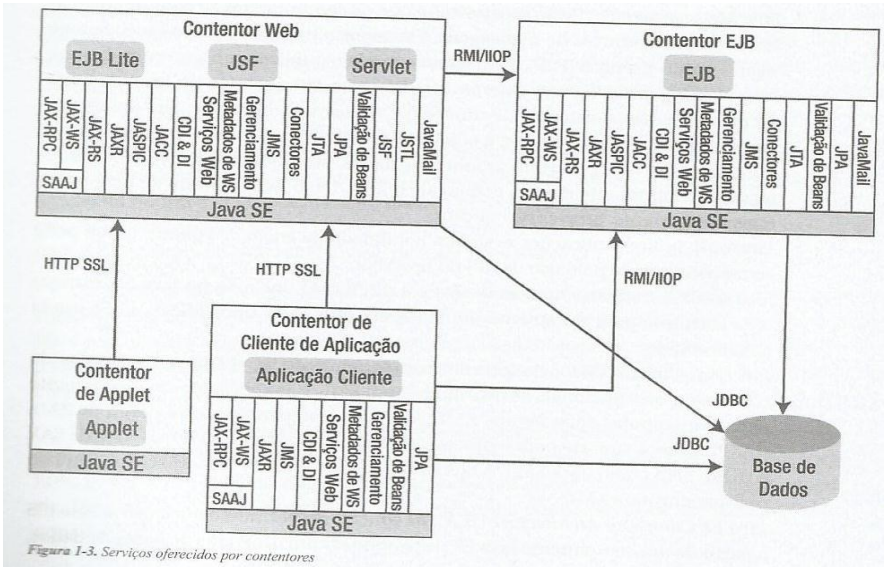
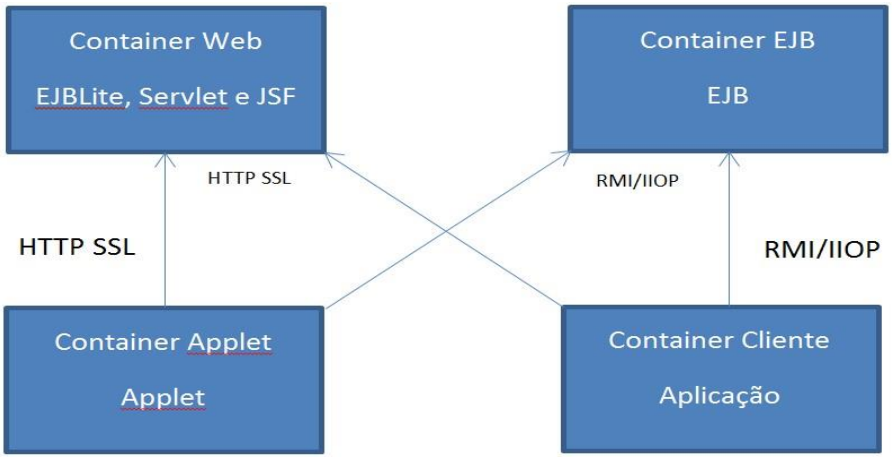


Figura 1-3. Serviços oferecidos por contentores

Diagrama dos serviços contidos na Plataforma Java EE 6



3.4 Servletes

As Servlets é a primeira forma para uma criação de páginas dinâmicas com Java. É próprio da linguagem Java criar uma classe que gera conteúdo HTML. O nome "servlet" vem da ideia de um pequeno cujo seu objetivo é receber chamadas HTTP, para que possa ser processadas e devolver uma boa resposta ao cliente [Caelum, 2015].

Cada servelets é responsável por uma página, que vê os dados do cliente e responde com outros dados (uma página HTML, uma imagem GIF, etc.). As principais vantagens são herdadas da própria linguagem Java, como: Portabilidade, Facilidade de programação e Flexibilidade. No Java muitas das vezes nós tentamos sempre que possível programar em orientado ao objeto, nada mais natural que uma servlet que é representada como um objeto a partir de uma classe Java. Cada servlet é, portanto, um objeto Java que recebe tais requisições e produz algo, como uma página HTML. Além disso, a linguagem oferece algumas facilidades, como por exemplo, o gerenciamento automático de memória [Caelum, 2015].

4 Considerações Finais

O conhecimento das diferentes formas de operação de cada ferramenta e o correto planejamento da estratégia de busca são fundamentais para a recuperação das informações que se deseja. E para atender a necessidade do usuário de forma rápida e segura, os motores de busca na web tentam cada vez mais desenvolver ferramentas que sejam de fácil uso, a fim de facilitar cada vez mais a busca.

Como produto final deste artigo, ficou melhor a interpretação e compreensão sobre o Java Enterprise Edition, diferenciando um Servidor de Aplicação de um Servlet, fazendo com que uma classe seja acessível via navegador, criando páginas e contendo formulários, recebendo e convertendo parâmetros enviados por uma página a fim de executar suas lógicas e regras de negócio.

5 Referências:

Harvey M. Deitel. Java: **Como Programar**. 6 ed. São Paulo: Pearson education do Brasil, 2005. 1097 p.

Apostila Java e Orientação a Objetos, licença Creative Commons, 7ª edição, pela Caelum - Ensino e Inovação.

Patrick Naughton, **Dominando o Java**, Guia Autorizado da Sun Microsystems, Editora Makron Books, 1997, Osborne

Gonçalves, Antônio. **Introdução à Plataforma Java EE 6 com o Glassfish** – Segunda edição. Rio de Janeiro: Editora Ciência Moderna LTDA, 2011.

Caelum. et al. (2015) **Java para Desenvolvimento Web.** Disponível em: <http://www.caelum.com.br/apostila-java-web>. Acesso em: 05/08/2015