FELIPE RODRIGUES DO PRADO
JOÃO PAULO NAKAJIMA PEREIRA

MODULARIZAÇÃO DE SOFTWARE

UNIVERSIDADE DO VALE DO SAPUCAÍ
POUSO ALEGRE
2015

FELIPE RODRIGUES DO PRADO
JOÃO PAULO NAKAJIMA PEREIRA

MODULARIZAÇÃO DE SOFTWARE

Pré-projeto de desenvolvimento apresentado à disciplina de TCC 1 do curso de Sistemas de Informação como requisito parcial para obtenção de créditos.

UNIVERSIDADE DO VALE DO SAPUCAÍ
POUSO ALEGRE
2015

SUMÁRIO

QUADRO TEÓRICO

Para que se possa desenvolver qualquer solução, é necessário o uso de algumas ferramentas, teorias e tecnologias. Abaixo serão apresentadas algumas delas usadas no desenvolvimento desta pesquisa, bem como sua utilidade.

1.1 Java

A linguagem de programação Java é composta por coleções de classes existentes nas bibliotecas de classe Java, conhecidas como Java APIs. Classes são partes que consistem uma aplicação Java, sendo que essas classes são compostas por outras partes chamadas métodos, que são responsáveis por realizar tarefas e retornar informações. É possível criar classes para formar uma aplicação Java e também utilizar as coleções de classes do Java API. Portanto, para se desenvolver uma aplicação utilizando essa linguagem, é preciso entender como criar classes próprias que irão compor a aplicação e como trabalhar utilizando as classes do Java API.

Java foi lançada em 1996 pela Sun Microsystems[1] com a finalidade de desenvolver aplicativos para diferentes plataformas (Windows – Linux – Web – Mobile). Qualquer dispositivo que possua a JVM[2] é capaz de executar o software desenvolvido em Java (SOBRAL; CLARO, 2008).

O objetivo da linguagem Java é uma plataforma para o desenvolvimento de sistemas de médio a grande porte. Uma das muitas vantagens de se utilizar a plataforma Java é a grande quantidade de bibliotecas gratuitas que podem ser utilizadas em diversos tipos de projetos. Como cada linguagem é apropriada para uma determinada área, a utilização de Java é melhor para o desenvolvimento de aplicações que tenham tendência a crescer (CAELUM, 2015).

Para desenvolver aplicações em Java é necessário instalar um *Kit* de desenvolvimento,

1 Fabricante de computadores, semicondutores e *software* adquirida pela Oracle Corporation em 2009.
2 JVM – Java *Virtual Machine*.

o Java *Development Kit* - JDK, o qual pode ser obtido no próprio site da Oracle – empresa mantenedora da plataforma. Ele é composto de compilador, máquina virtual, bibliotecas e utilitários.(CAELUM, 2015).

E assim essa linguagem demonstra ser de imprescindível utilidade para nosso projeto, pois além de várias vantagens propiciadas ao projeto citadas anteriormente temos ainda a facilidade da manutenção e alteração do código e o seu uso em diversas plataformas.

1.2 OSGi

Modularização é o desenvolvimento de *software* de maneira desacoplada, separando o *software* em partes independentes ou *bundles*, ou seja, é o particionamento do *software* em partes menores, com o objetivo de gerenciar e controlar melhor o desenvolvimento e manutenção do código.

Knoernschild (2012) afirma que para que uma aplicação seja modularizada, qualquer módulo dela deve ser instalável, gerenciável, reutilizável, combinável, não guardar estado e oferecer uma interface clara.

A tecnologia *Open Services Gateway iniciative* – OSGi é um conjunto de especificações o qual segue um modelo de desenvolvimento em que as aplicações são dinamicamente compostas de componentes distintos e reutilizáveis (OSGI ALLIANCE, 2015).

De acordo com OSGi Alliance (2015) os utilizadores dessa tecnologia obtém como benefício a redução da complexidade do desenvolvimento de modo em geral, como por exemplo: aumento da reutilização de módulos, incremento da facilidade de codificação e teste, gerenciamento total dos módulos, sem a necessidade de reiniciar a aplicação, aumento do gerenciamento da implantação e detecção antecipada de *bugs.* Possui como uma de suas aplicações mais populares a IDE Eclipse e o *framework* Spring.

A especificação teve início em março de 1999 pela OSGi Alliance. Seus principais desafios na época não era desenvolver uma solução para a execução de diferentes versões de um mesmo projeto na mesma aplicação, mas sim de elaborar uma maneira que diferentes componentes que não se conhecem possam ser agrupados dinamicamente sob o mesmo

projeto (OSGI ALLIANCE, 2015).

Segundo Gama (2008) a OSGi Alliance surgiu a partir do desinteresse da Sun Microsystems[3] em manter o projeto, mesmo esse projeto possibilitando que diferentes *deploys* fossem hospedados de maneira simples, isolada e segura numa única JVM[4].

O OSGi está estruturado da seguinte maneira ilustrada a seguir:

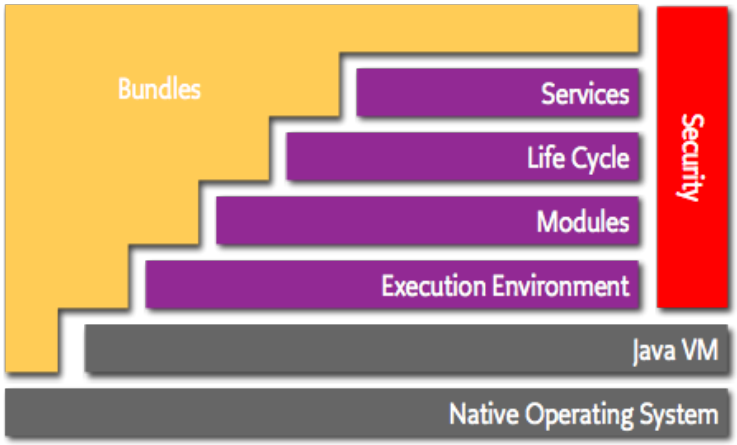

Fig 1 - Estrutura de Camadas. Fonte: OSGi Alliance (2015)

Segundo OSGI ALLIANCE (2015) "a camada de serviços conecta pacotes de uma forma dinâmica, permitindo melhor reutilização de código, utilizando-se de objetos isolados por interfaces para isolar o módulo que usará o serviço das classes que o implementam."

Os *bundles* (como os módulos são chamados no contexto da OSGi) consomem ou disponibilizam serviços. Eles estão organizados de uma maneira que formam a tríade consumidor, fornecedor e registro dos serviços, na qual pode-se consumir serviços ou disponibilizá-los. Para a liberação de um novo serviço, deve-se registrá-lo em um catálogo, onde este teria visibilidade por parte de *bundles* externos que consumiriam esses serviços disponibilizados (OSGI ALLIANCE, 2015).

OSGI ALLIANCE (2015) diz que a API[5] disponibilizada permite o fácil gerenciamento (instalação, inicialização, parada e atualização) dos pacotes, bem como possibilita enumerar os pacotes e utilizar de seus serviços.

A especificação OSGi requer uma implementação de referência, sendo assim diversas implementações foram criadas, dentre elas podemos destacar Equinox (Projeto Eclipse) e Felix (Apache) (MADEIRA, 2009).

3 Fabricante de computadores, semicondutores e *software* adquirida pela Oracle Corporation em 2009.
4 JVM – Java *Virtual Machine*.
5 API – Abreviação para *Application Programming Interface*.

Segundo LUCENA (2010) “OSGi é uma especificação, ou seja, documenta e atribui funções precisas a tais serviços, mas não os implementa. Equinox, por outro lado, é uma implementação desta especificação”.

Um aspecto importante é que um *bundle* pode ser executado em qualquer implementação de OSGi. Ou seja, um *bundle* desenvolvido e testado em uma implementação pode ser executado em qualquer outra implementação de OSGi (LUCENA, 2010).

Segundo Gama (2008) “existe uma curva de aprendizado que não vale a pena e nem faz sentido se você está desenvolvendo aplicações que não precisam das vantagens do OSGi”.

1.3 Spring Framework

Spring Framework é um *framework* que disponibiliza um modelo de programação e configuração para o desenvolvimento de aplicações corporativas para a plataforma Java, com o objetivo de estruturar a aplicação de maneira que o desenvolvedor tenha foco na lógica de negócios em nível de aplicativo, enquanto o *framework* gerencia implementações específicas (PIVOTAL SOFTWARE, 2015).

De acordo com DevMedia (2015), o Spring é um *framework Open Source* e foi criado por Rod Johnson no ano de 2002, com o objetivo de simplificar o desenvolvimento utilizando a plataforma Java, possibilitando a criação de *software* que só poderia ser criado anteriormente através de EJB's[6].

Das funcionalidades do Spring Framework, podemos destacar como principais a injeção de dependências, programação orientada a aspectos, aplicação *web* Spring MVC, *framework* de serviços *web* RESTfull e suporte para JDBC[7], JPA[8] e JMS[9] (PIVOTAL SOFTWARE, 2015).

O Spring Framework é integrado com a plataforma OSGi[10] através do Spring Dynamic Modules que gerencia o ciclo de vida, controla e permite exportar e importar serviços OSGi de forma transparente (PIVOTAL SOFTWARE, 2015).

6 EJB – Abreviação para *Enterprise JavaBeans.*
7 JDBC – Abreviação para Java *Database Connectivity.*
8 JPA – Abreviação para Java *Persistence* API.
9 JMS – Abreviação para Java *Message Service.*
10 OSGi – Abreviação para *Open Services Gateway iniciative*

1.4 Hibernate

Segundo Bauer e King (2005, p.19), Hibernate é um *framework* de persistência que possui como propósito fornecer uma visão orientada a objetos sobre banco de dados relacionais, ou seja, permite persistir e gerenciar objetos de Java para as tabelas dos bancos de dados relacionais de forma simplificada. Para se realizar essa conversão de dados é utilizado arquivos XML[11] que contém as configurações para se fazer o mapeamento dos dados contidos da classe em Java para as colunas do Sistema Gerenciador de Banco de Dados ou SGBD.

Utilizar-se dos benefícios oferecidos pelo Hibernate para o desenvolvimento de uma aplicação que utilize banco de dados relacional e linguagem orientada a objetos é de grande estima, pois segundo Durham e Johnson (1996, pg.34) no desenvolvimento de *software*, um *framework* é uma estrutura de suporte definida em que um outro projeto de *software* pode ser organizado e desenvolvido. Tipicamente, um *framework* pode incluir programas de apoio, bibliotecas de código, linguagens de *script* e outros *softwares* para ajudar a desenvolver e juntar diferentes componentes do seu projeto.

1.5 PostgreSQL

Segundo Neto (2003 apud Souza et al,2012,p. 2), o PostgreSQL é um SGBDR - Sistema Gerenciador de Banco de Dados Relacional que está baseado nos padrões SQL[12] ANSI-92, 96 e 99, possui alta performance, de fácil administração e utilização em projetos por especialistas *Database Administrators* (DBAs) e Projetistas de Sistemas.

PostgreSQL teve origem em um projeto chamado de POSTGRES na Universidade Berkeley, na Califórnia (EUA), em 1986. Sua equipe fundadora foi orientada pelo professor Michael Stonebraker com o apoio de diversos órgãos, entre eles o Army Research Office (ARO) e o National Science Foundation (NSF). Atualmente, o SGBD encontra-se em sua versão 9.4 estável, contendo todas as principais características que um SGBD pode

11 XML – Abreviação para *Extensible Markup Language*.
12 SQL – Abreviação para *Structured Query Language*.

disponibilizar (MILANI,2008).

Seu código é livre e há um grupo responsável pela sua validação. Grandes empresas como a Fujitsu, NTT Group, Skype, Hub.org, Red Hat e SRA são financiadoras do PostgreSQL que, além disso, recebe doações. É utilizado por multinacionais, órgãos governamentais e universidades. Recebeu vários prêmios como melhor sistema de banco de dados *Open Source* (Souza et al; 2011).

1.6 HyperText Markup Language (HTML)

HyperText Markup Language, é o significado da sigla HTML, que, em português, significa linguagem para marcação de hipertexto. Foi criada por Tim Berners-Lee na década de noventa tornando-se um padrão internacional. De modo geral, o hipertexto é todo o conteúdo de um documento para *web*, com a característica de se interligar a outros documentos da *web* através de *links* presentes nos hipertextos (Silva, 2011).

Conforme Mozilla Developer Network (2014), a HTML é uma linguagem de marcação que estrutura o conteúdo de um documento da *web*. O conteúdo visto ao acessar uma página através do navegador é descrito utilizando a HTML, tornando-se a principal linguagem para conteúdo da *web* mantida pelo *World Wide Web Consortium* (W3C).

O W3C é uma comunidade internacional liderada pelo criador da *web* Tim Berners-Lee, é formada por organizações, profissionais e público em geral com o objetivo de conduzir a *web* ao seu potencial máximo, através do desenvolvimento de padrões e especificações (W3C, 2015).

Segundo Silva (2011), desde a criação da *web* em 1992, a HTML passou pelas versões HTML, HTML+, HTML 2.0, HTML 3.0, HTML 3.2, HTML 4.0, HTML 4.01 e HTML 5. Entre essas versões, o W3C, considera somente as versões HTML 2.0, HTML 3.2, HTML 4.0, HTML 4.01 e HTML 5 oficialmente pois as outras versões são anteriores à criação da W3C.

Mesmo sendo uma linguagem destinada à criação de documentos, a HTML não tem como objetivo definir estilos de formatação, como por exemplo, nomes e tamanhos de fontes, margens, espaçamentos e efeitos visuais. Também não possibilita adicionar funcionalidades de interatividade avançada à página. A linguagem HTML destina-se somente a definir a

estrutura dos documentos *web*, fundamentando dessa maneira os princípios básicos do desenvolvimento seguindo os padrões *web* (Silva, 2011).

1.7 JavaScript

JavaScript é uma linguagem de programação criada pela Netscape em parceria com a Sun Microsystems. Sua primeira versão, definida como JavaScript 1.0, foi lançada em 1995 e implementada em março de 1996 no navegador Netscape Navigator 2.0 (Silva, 2010).

Segundo Silva (2010), o JavaScript tem como finalidade fornecer funcionalidades para adicionar interatividades avançadas a uma página *web*. É desenvolvido para ser executada no lado do cliente, ou seja, é interpretada pelo navegador do usuário. Os navegadores possuem funcionalidades integradas para realizar a interpretação e o funcionamento da linguagem JavaScript.

Conforme Mozilla Developer Network (2015), JavaScript é baseado na linguagem de programação ECMAScript, que é padronizado pela Ecma International na especificação ECMA-262 e ECMA-402.

JavaScript é uma linguagem interpretada, baseada em objetos com funções de primeira classe e com a característica de ser uma linguagem leve para a execução. Funções de primeira classe em linguagem de programação, são funções que podem ser passadas como argumentos para outras funções, retornadas de outras funções, atribuídas a variáveis ou armazenadas em estrutura de dados. Além de ser executada nos navegadores, o JavaScript é utilizada em outros ambientes, como por exemplo, node.js ou Apache CouchDB (Mozilla Developer Network, 2015).

De acordo com Caelum (2015), JavaScript é responsável por aplicar qualquer tipo de funcionalidade dinâmica em páginas *web*. É uma linguagem poderosa, que possibilita ótimos resultados. Podemos citar o Gmail, Google Maps e Google Docs como exemplos de aplicações *web* desenvolvidas utilizando JavaScript.

1.8 Cascading Style Sheet (CSS)

Cascading Style Sheet, que traduzido para o português significa folhas de estilo em cascata com a abreviação CSS, tem a finalidade de definir estilos de apresentação para um documento HTML[13] (Silva, 2012).

De acordo com a W3C[14] (2015), "*Cascading Style Sheets (CSS) is a simple mechanism for adding style (e.g., fonts, colors, spacing) to Web documents*[15]".

Tim Berners-Lee considerava que os navegadores eram responsáveis pela estilização de uma página *web*, até que em setembro de 1994, surge como proposta a implementação do CSS. Em dezembro de 1996, foi lançada oficialmente pela W3C as CSS1. O W3C utiliza o termo nível em vez de versão, tendo dessa maneira CSS nível 1, CSS nível 2, CSS nível 2.1 e atualmente CSS nível 3, conhecida também como CSS3 (Silva, 2012).

De acordo com Mozilla Developer Network (2015), a CSS descreve a apresentação de documentos HTML, XML[16] ou XHTML[17]. É padronizada pelas especificações da W3C e já contém seus primeiros rascunhos do nível CSS4.

Enquanto o HTML é uma linguagem destinada para estruturar e marcar o conteúdo de documentos na *web*, o CSS fica responsável por definir cores, nome e tamanho das fontes, espaçamentos e outros atributos relacionados a apresentação visual da página. Essa separação da marcação e estrutura de um documento do seu estilo de apresentação torna o uso do CSS uma grande vantagem no desenvolvimento de documento para a *web* (Maujor, 2015).

1.9 Test-Driven Development (TDD)

Test-Driven Development ou em português desenvolvimento guiado por testes é o significado da abreviação TDD. É sem dúvida uma das práticas mais populares utilizadas no desenvolvimento de software, que traz uma ideia bem simples: escrever os testes antes mesmo

13 HTML – Abreviação para *HiperText Markup Language.*
14 W3C – Abreviação para *World Wide Web Consortium.*
15 Folha de estilo em cascata é um mecanismo simples para adicionar estilos (por exemplo: fontes, cores, espaçamentos) para documentos *web* (tradução nossa).
16 XML – Abreviação para *Extensible Markup Language.*
17 XHTML – Abreviação para *Extensible HyperText Markup Language.*

de escrever o código de produção (ANICHE, 2014).

Automatizar os testes de um *software* é a atividade de submeter o código desenvolvido para outro *software* testá-lo e assim obter o apontamento de erros no código e indicação das falhas para futura correção.

Qualidade do *software*, fácil manutenção e evolução são uma das vantagens que se conquista testando um *software* (VAZ, 2003).

Teste é a atividade do ciclo de desenvolvimento de s*oftware* na qual pode ser observada maior distância entre a teoria (técnicas de teste propostas) e a prática (aplicação destas técnicas) (MYERS, 2004).

Conforme ANICHE (2012), quando a produtividade é medida através do número de linhas de código escrito por dia, o rendimento será menos produtivo. Todavia, se produtividade for a quantidade de linhas de código final sem defeitos escritos por dia, provavelmente o rendimento será mais produtivo ao usar testes automatizados.

Segundo VAZ (2003), para facilitar a execução das rotinas dos testes automatizados foram criados *frameworks* como o JUnit, que através da instalação de seu *plug-in* na ferramenta de desenvolvimento Eclipse, disponibiliza uma infraestrutura para se realizar os testes do código.

Dentre as atividades de um processo de desenvolvimento, as atividades de testes têm uma importância fundamental para a garantia de qualidade do *software* que está sendo desenvolvido, a qual é aplicada no decorrer de todo o projeto.

2 REFERÊNCIAS

ANICHE, Maurício. **Test-Driven Development:** Teste e Design no Mundo Real. São Paulo: Casa do Código, 2012.

ANICHE, Maurício. **TDD**. 2014. Disponível em http://tdd.caelum.com.br/. Acesso em 11 de março, 2015.

BAUER, Christian; KING, Gavin. **Hibernate in action**. Greenwitch: Manning Publications, 2005.

CAELUM. **Apostila Desenvolvimento Web com HTML, CSS e JavaScript**. 2015. Disponível em https://www.caelum.com.br/apostila-html-css-javascript/javascript-e-interatividade-na-web/. Acesso em 08 de março, 2015.

CAELUM. **Apostila Java e Orientação a Objetos**. 2015. Disponível em http://www.caelum.com.br/apostila-java-orientacao-objetos/. Acesso em 08 de março, 2015.

CLARO, Daniela Barreiro; SOBRAL, João Bosco Mangueira. **Programação em Java**. Santa Catarina: Copyleft Pearson Education, 2008.

DEITEL, Harvey Matt; DEITEL, Paul John.**Java How to Program**. 8. ed. Edson Furmankiewicz. São Paulo: Pearson Prentice Hall, 2010.

DEVMEDIA. **Introdução ao Spring Framework**. 2015. Disponível em http://www.devmedia.com.br/introducao-ao-spring-framework/26212. Acesso em 10 de março, 2015.

DURHAM, Alam; JOHNSON, Ralph. **A Framework for Run-time Systems and its Visual Programming Language.** In: OBJECT-ORIENTED PROGRAMMING SYSTEMS, LANGUAGES, AND APPLICATIONS. San Jose, CA, 1996, p. 20-25.

Javascript Brasil. **Funções de primeira classe**. 2012. Disponível em: http://javascriptbrasil.com/2012/09/05/funcoes-de-primeira-classe/. Acesso em 08 de março, 2015.

KIEVGAMA. **Uma visão geral sobre a plataforma OSGi**. 2008. Disponível em https://kievgama.wordpress.com/2008/11/24/um-pouco-de-osgi-em-portugues/. Acesso em 09 de março, 2015.

KNOERNSCHILD, Kirk. **Java application architecture:** modularity patterns with examples using OSGi**.** Crawfordsville: Pearson Education, 2012.

LUCENA, Fábio Nogueira de. **Introdução ao Equinox**. 2010. Disponível em https://code.google.com/p/exemplos/wiki/equinox. Acesso em 09 de março, 2015.

MADEIRA, Marcelo. **OSGi – Modularizando sua aplicação**. 2009. Disponível em https://celodemelo.wordpress.com/2009/11/12/osgi-modularizando-sua-aplicacao/. Acesso em 09 de março, 2015.

MAUJOR. **Site sobre CSS e Padrões Web:** Por que CSS?. 2015. Disponível em: http://www.maujor.com/index.php. Acesso em: 08 de março, 2015.

MYERS, Glenford John. **The Art of Software Testing**. Hobokey: Word Association, 2004.

MILANI, André. **PostgreSQL:** Guia do Programador. São Paulo: Novatec.Editora, 2008.

Mozilla Developer Network. **HTML**. 2014. Disponível em: https://developer.mozilla.org/pt-BR/docs/Web/HTML. Acesso em 07 de março, 2015.

Mozilla Developer Network. **JavaScript language resources**. 2014. Disponível em: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Language_Resources. Acesso em 08 de março, 2015.

Mozilla Developer Network. **CSS**. 2015. Disponível em: https://developer.mozilla.org/pt-BR/docs/Web/CSS. Acesso em 08 de março, 2015.

Mozilla Developer Network. **JavaScript**. 2015. Disponível em: https://developer.mozilla.org/pt-BR/docs/Web/JavaScript. Acesso em 08 de março, 2015.

OSGI ALLIANCE. **OSGi**. 2015. Disponível em http://www.osgi.org/Main/HomePage. Acesso em 08 de março, 2015.

Pivotal Software. **Preface**. 2015. Disponível em http://docs.spring.io/osgi/docs/current/reference/html/preface.html. Acesso em 10 de março, 2015.

Pivotal Software. **Spring Framework**. 2015. Disponível em http://projects.spring.io/spring-framework/. Acesso em 10 de março, 2015.

SILVA, Maurício Samy. **JavaScript:** Guia do Programador. São Paulo: Novatec Editora, 2010.

SILVA, Maurício Samy. **HTML5:** A linguagem de marcação que revolucionou a web. São Paulo: Novatec Editora, 2011.

SILVA, Maurício Samy. **CSS3:** Desenvolva aplicações web profissionais com uso dos poderosos recursos de estilização das CSS3. São Paulo: Novatec Editora, 2012.

SOUZA, Arthur Câmara; AMARAL, Hugo Richard; LIZARDO, Luis Eduardo O. **PostgreSQL:** uma alternativa para sistemas gerenciadores de banco de dados de código aberto. In: Anais do Congresso Nacional Universidade, EAD e Software Livre, 2012.

VAZ, Rodrigo Cardoso. **JUnit - Framework para testes em Java**. Palmas: Centro Universitário de Palmas, 2003.

W3C. **About W3C**. 2015. Disponível em http://www.w3.org/Consortium/. Acesso em 07 de março, 2015.

W3C. **What is CSS?**. 2015. Disponível em http://www.w3.org/Style/CSS/. Acesso em 08 de março, 2015.