SGA
Documento de Arquitetura

Arquitetos Responsáveis: Adriano de Paula Costa, Pedro Taylon de Faria

Histórico de Revisões

Data	Versão	Descrição	Autor
27/07/2015	0.3	Versão Final do Documento de Arquitetura – Revisado.	Costa, Adriano de Paula; Carvalho, Pedro Taylon
18/08/2015	1.0	Versão Final do Documento de Arquitetura – Atualização.	Costa, Adriano de Paula; Carvalho, Pedro Taylon

1. Introdução

A arquitetura de software de um sistema consiste na definição dos componentes, propriedades extrenas, e seus relacionamentos com outros softwares.

Neste documento buscamos detalhar os principais componentes da arquitetura proposta para o desenvolvimento de sistemas SGA (Sistema de Gerenciamento Acadêmico). Definimos a arquitetura seguindo diversos padrões de projeto já consolidados pela comunidade, dentre estes, buscamos pricipalmente os padrões Orientados a Objetos com destaque no mercado. Cada parte da arquitetura será destacada, buscando mostrar o motivo da sua criação.

2. Objetivos

O Documento de Arquitetura do Software provê uma visão geral da arquitetura, usando um conjunto de visões arquiteturais para tratar aspectos diferentes do software.

Este documento serve como um meio de comunicação entre o Arquiteto de Software e outros membros da equipe de projeto sobre as decisões significativas que forem tomadas durante o projeto.

3. Considerações Gerais

As definições arquiteturais de um projeto de desenvolvimento de software em geral seguem as definições necessárias aos vários projetos de uma organização ou instituição e que atenda a todas as necessidades do projeto, desde a segurança, regras de negócio, até a persistência dos dados. Essas definições do projeto em geral estão em um documento à parte, elaborado durante um trabalho de consultoria arquitetural.

As definições do projeto já documentadas até o presente momento devem guiar as primeiras versões do Documento de Arquitetura do Software, que é desenvolvido durante a fase de Elaboração, uma vez que o propósito dessa fase é estabelecer os fundamentos arquiteturais para o projeto do software.

4. Responsabilidades

O Arquiteto de Software é o responsável por elaborar este documento e por manter a integridade do mesmo durante o processo de desenvolvimento do software. Ele deve:

- Aprovar todas as mudanças arquiteturais significativas e documentá-las.
- Fazer parte do comitê que decide sobre os problemas que tenham algum impacto arquitetural.

5. Referências

1. Bourning, A. and Travers, M. Two approaches to casual Interaction on Computer and Video Networks. Proceedings of International Networking Conference, 1991.

2. Dourish, P. and Bly, S. Portholes: Supporting Awareness in distributed Work Group. Proceedings CHI, 1992.

3. Ellis, C.A., Barthelmess, P., Quan, B. e Wainer, J. NEEM: An Agent Based Meeting Augmentation System. Technical Report CU-CS-937-02, University of Colorado at Boulder, Computer Science Department, 2002.

6. Arquitetura

O que é? E como é composta?

A arquitetura foi desenvolvida para ser totalmente de alta coesão e baixo acoplamento.

6.1. Elementos que compõe a Arquitetura

Quais são os principais elementos da arquitetura?

A arquitetura é composta por alguns elementos, entenda-se classes, que em conjunto produzem o efeito desejado pela arquitetura como um produto final para o desenvolvimento..

Elementos pertencentes à arquitetura:

- Banco de dados
- WEB
    - JSF
    - TagLibs
    - Spring-Security

Nos próximos capítulos iremos discutir cada componente listado anteriormente e qual o seu papel dentro da arquitetura como um todo, além de discutir de forma sucinta a tecnologia e/ou o padrão adotado para a implementação do mesmo.

6.2. **Banco de Dados**

O que é essa camada? E pra que serve?

No desenvolvimento de sistemas precisamos em muitas vezes fazer acesso a uma determinada base de dados. Em algumas linguagens de programação, principalmente as estruturadas, a separação real de funcionalidades na programação é muito complexa de forma que acessos a dados, consultas, entre outros elementos do sistema, acabam se misturando, o que ocasiona um alto acoplamento para termos alta coesão. Com a programação orientada a objetos isso já não ocorre em modelagens mais elaboradas. No caso da arquitetura desenvolvida temos uma camada apenas de banco de dados ou camada de persistência.

Nessa camada estão os modelos de dados mapeados segindo anotações JPA, gerenciamento da conexão com o SGBD.

Com esse tipo de solução conseguimos realmente separar das regras de negócio a implementação de funcionalidades básicas de acesso à base de dados, controles transacionais e gerenciadores de conexão.

6.3. **WEB**

Como é dividida? Quais seus principais elementos?

Já detalhamos bem a camada de persistência de dados, agora iremos detalhar a camada de interface com o usuário. Nesse primeiro momento iremos falar sobre interfaces web, como ela foi dividida e o que foi envolvido na criação da mesma.

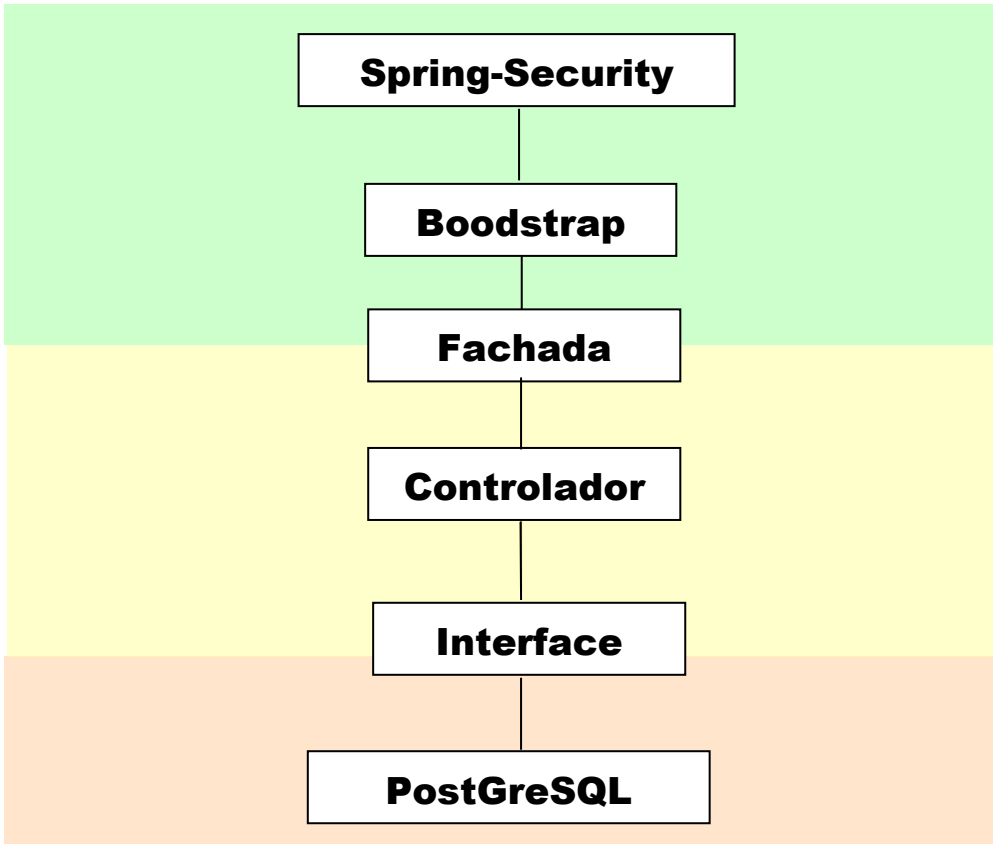
O Jboss Widfly é um servidor de aplicações Java para web. O WildFly como o sucessor do projeto JBoss Application Server. O WildFly representa tanto uma atualização da marca para o projeto, quanto uma renovação da sua visão para estimular a próxima geração de tecnologias de servidor de aplicativos. O nome foi escolhido pelos membros da comunidade de código aberto no website JBoss.org, durante uma votação especial no final de 2012.

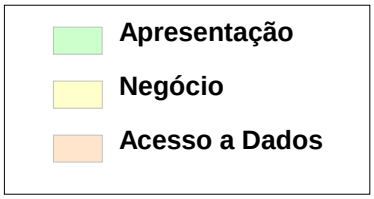
A tecnologia continua a servir como o projeto de destaque para o JBoss Enterprise Application Platform da Red Hat, e aborda algumas das principais forças que dão forma ao middleware hoje, incluindo a mudança de abordagens mais flexíveis e modernas para o desenvolvimento de aplicativos, habilitação de nuvem híbrida aberta e Java Enterprise Edition 7 (Java EE 7).

A interface web foi desenvolvida utilizando-se JFS e as taglibs, também foi utilizado Boodstrap e o spring-security.

As taglibs são tags criadas para facilitar a criação das interfaces gráficas web. Com isso conseguimos criar tags com lógicas que facilitam a programação e a manutenção das JSP’s (Java Server Pages, as páginas dinâmicas do J2EE).

6.4. **Desenho GERAL da Arquitetura**





7. Padrões de Projeto

Os padrões de projeto possibilitam a reutilização de técnicas comprovadas com o objetivo de atender requisitos técnicos relacionados ao projeto de um sistema. Para o SSAA, foram analisados alguns padrões de projeto e selecionados aqueles que poderiam ser satisfatoriamente aplicados.

7.1. Facade

O padrão de projeto Facade oferece um ponto centralizado e unificado para um conjunto de interfaces em um subsistema ou do sistema como um todo, que representa o conjunto de serviços oferecidos. O SGA implementa a Fachada como um ponto de acesso único para as funcionalidades, isolando os diversos componentes do sistema.

7.2. MVC

Padrão de arquitetura de software que separa a representação da informação da interação do usuário com ele. O modelo (model) consiste nos dados da aplicação, regras de negócios, lógica e funções. Uma visão (view) pode ser qualquer saída de representação dos dados, como uma tabela ou um diagrama. É possível ter várias visões do mesmo dado, como um gráfico de barras para gerenciamento e uma visão tabular para contadores. O controlador (controller) faz a mediação da entrada, convertendo-a em comandos para o modelo ou visão.

8. Objetivos e Restrições Arquiteturais

Esta seção descreve os requisitos e objetivos do software que têm algum impacto na arquitetura, tais como: segurança, proteção de dados, privacidade, portabilidade, distribuição, reuso. Também são descritos nesta seção restrições arquiteturais que se aplicam ao projeto, tais como: estratégias de modelagem e implementação, ferramentas de desenvolvimento, sistemas legados.

8.1. Requisitos básicos

- A arquitetura deve seguir o padrão J2EE/J2SE.
- Utilização de componentes opensource.
    - Jboss Wildfly como servidor WEB (contêiner WEB).
- O modelo de interface deverá ser WEB.