# **DEFPROC**

# **Documento de Arquitetura de Software**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nome** | **Alterações** | **Data** | **Versão** |
| Jonathan Silva Pereira | Criação do Documento | 10/05/2013 | 0.01 |
| Jonathan Silva Pereira | Adicionado visões arquiteturais | 21/05/2013 | 0.02 |
|  |  |  |  |

**1 - Introdução**

**1.1 Finalidade**

Este documento oferece uma visão geral arquitetural abrangente do projeto **DEFPROC**, usando visões arquiteturais para representar diferentes aspectos do sistema. O objetivo deste documento é capturar e comunicar as decisões arquiteturais significativas, tomadas em relação ao sistema, além de alguns padrões que devem ser seguidos durante a implementação da solução e seus requisitos não funcionais.

**1.2 Escopo**

O Documento de Arquitetura de Software influencia diretamente a implementação dos requisitos não-funcionais.

**1.3 Definições, Acrônimos e Abreviações**

**1.4 - Referências**

**1.5 - Visão Geral**

As seções e sub-seções do Documento de Arquitetura de Software apresentam diferentes visões da arquitetura, com suas camadas e componentes, seus objetivos e restrições, organização das classes em pacotes e outros padrões de implementação.

**2 - Representação Arquitetural**

A documentação e modelagem da arquitetura do sistemas seguem o estilo de design em múltiplas camadas, que são representadas neste documento através de visões lógicas e de implementação.

**3 - Visão de Casos de Uso**

A Visão de Casos de Uso descreve o conjunto de cenários e/ou casos de uso que são o foco de uma iteração. Ela descreve o conjunto de cenários que representam alguma funcionalidade central e significativa. Também descreve o conjunto de cenários que possuem cobertura arquitetural substancial.

Os casos de uso deste sistema estão listados a seguir. Uma descrição desses casos de uso pode ser encontrada posteriormente nesta seção.

Efetuar Login

Criar processo de software

Alterar processos de software

Definir atividades

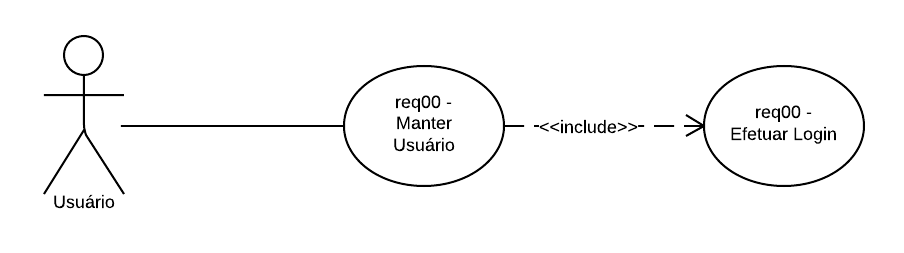
Definir detalhes de atividades

Definir execução concorrente de atividades

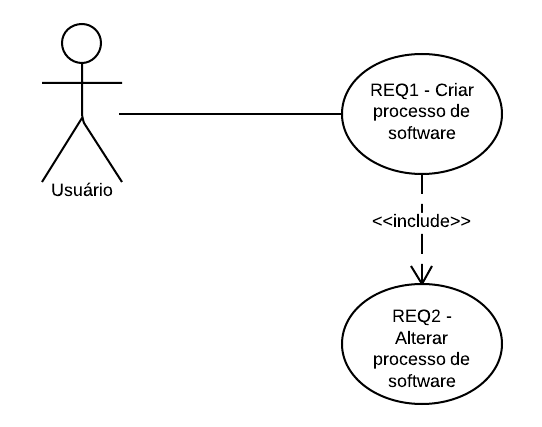
Definir fluxo de atividades

Definir execução de atividades

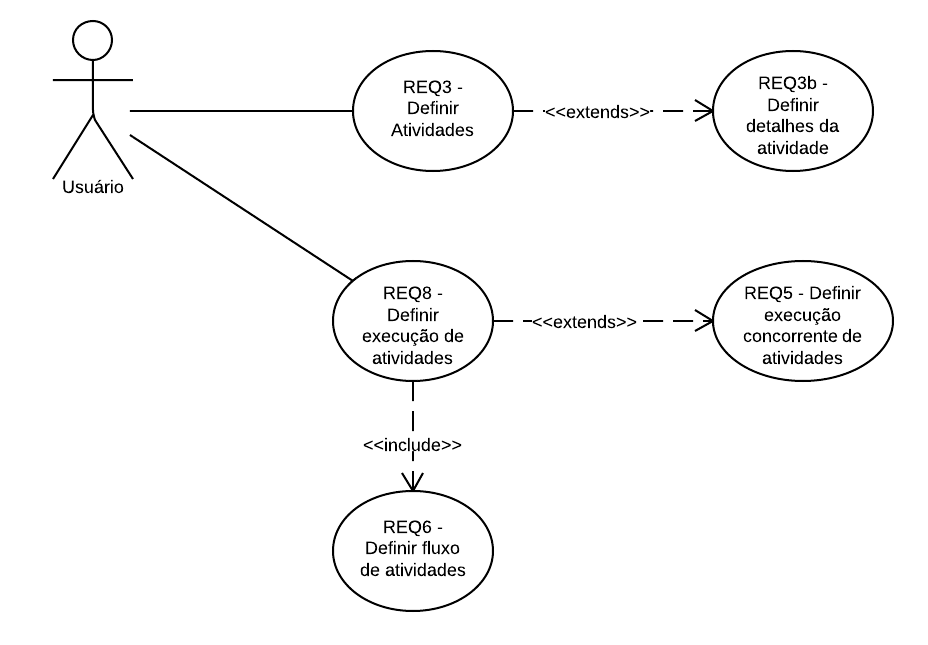
**3.1 - Efetuar Login**



**3.2 - Criar Processo de Software**



**3.2 - Definir Atividades**



**4 - Visão Lógica**

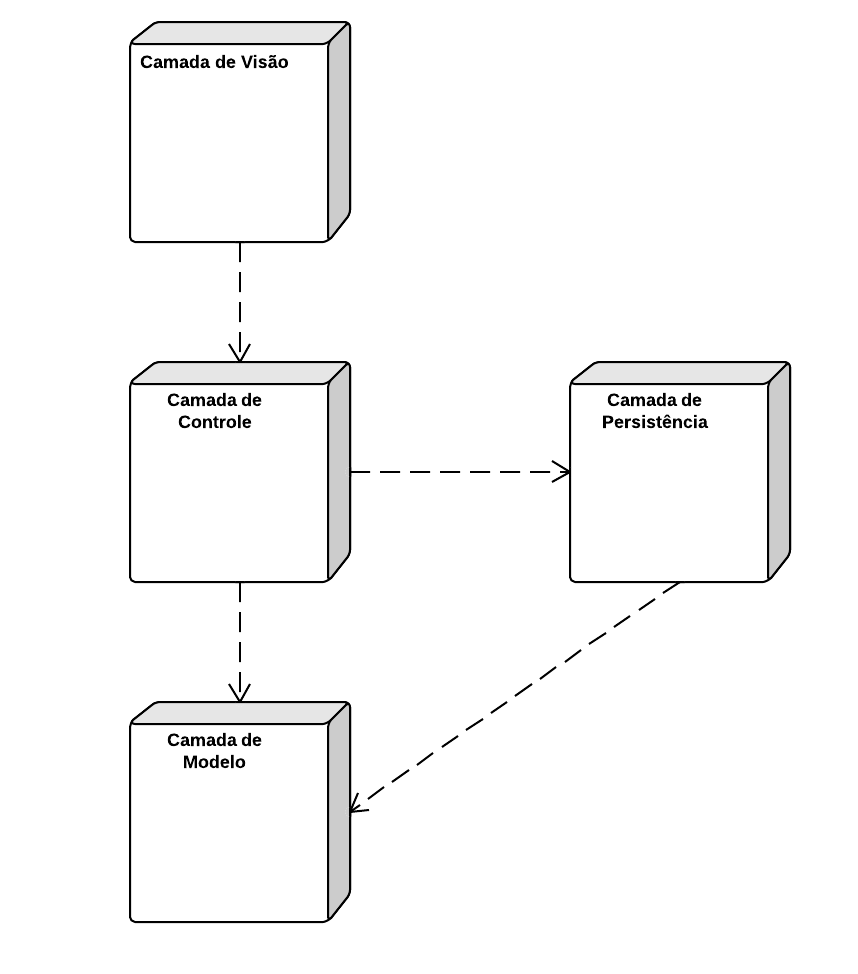
**4.1 Visão Geral**

A visão lógica define a estrutura lógica da arquitetura. Ela consiste em descrever as camadas que compõem a aplicação definindo as responsabilidades de cada camada. Nesta

**4.2 Definição das Camadas**

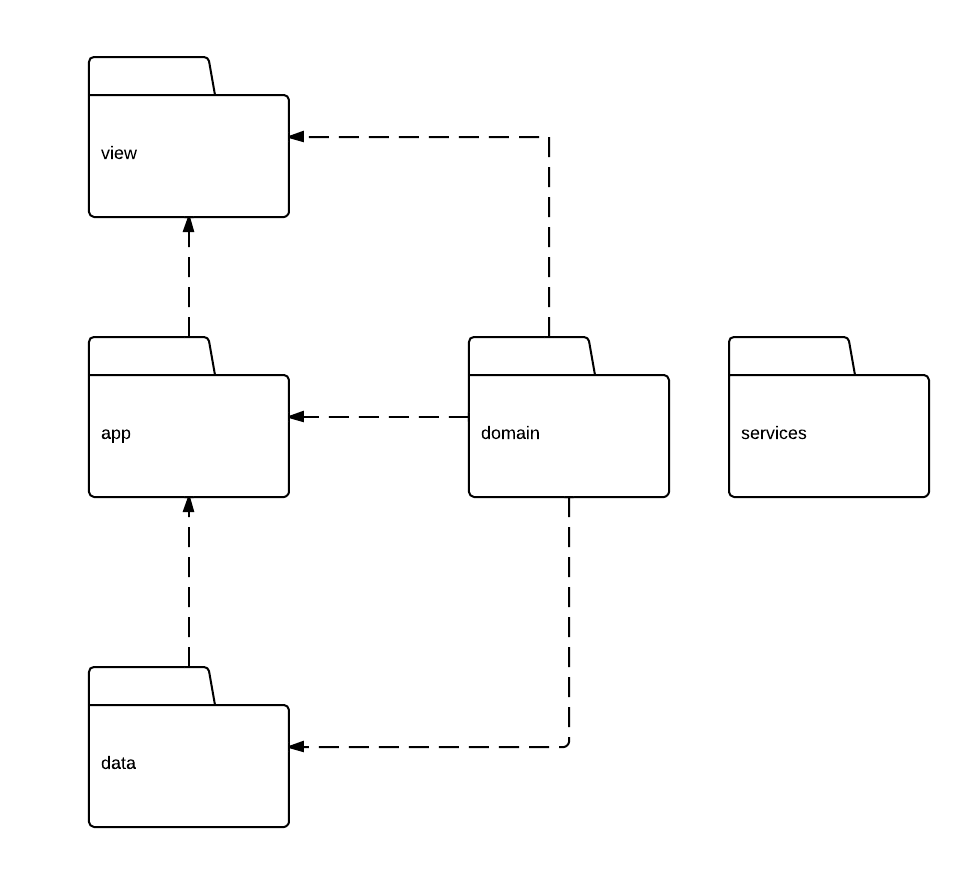
Visando atender uma necessidade de fácil criação, manutenção e a separação das camadas do aplicativo. Adotou-se como base o padrão de projeto MVC, para a plataforma Java.

No diagrama abaixo estão ilustradas cada uma das camadas que compõem a arquitetura básica proposta

:

**4.3 Pacotes Significativos**

A estrutura é composta por cinco pacotes significativos, *view, app, domain, data, services:*



* **view**

Contém as classes responsáveis pela comunicação dos atores com o Sistema. As classes desse pacote existem para oferecer.

* **app**

Contém as classes para a funcionalidade de processamento principal dentro do sistema. As classes de controle são responsáveis pela comunicação entre as camadas de Visão, Persistência e Modelo.

* **domain**

Contém pacotes com classes que fornecerão os modelos como Atividades, Projetos, Usuários, etc.

* **data**

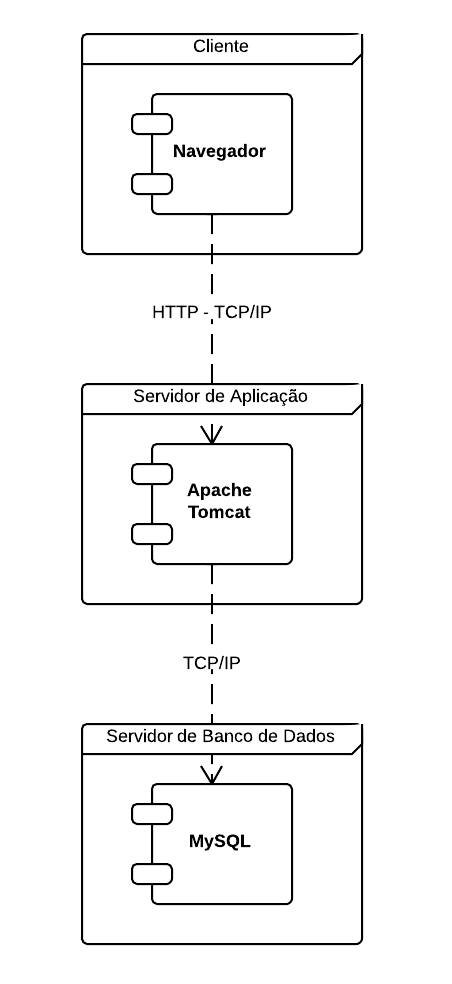
Contém classes que persistirão em objetos específicos dentro do sistema.

* **services**

Contém pacotes que fornecerão classes em nível de sistema para fins de manutenção.

**5 Visão de Implantação**

**5.1 Visão Geral**

Esta visão define o ambiente de implantação onde a aplicação será publicada/instalada.

**6 Visão de Implementação**

A visão de implementação aborda a arquitetura sobre a perspectiva do projeto estrutural dos componentes do sistema, como o sistema e cada um dos seus componentes serão organizados em termos de diretórios/pacotes.

A implementação da arquitetura segue o padrão de projeto MVC, dividido em três camadas, Visão, Controle e Modelo.

**Visão** - Camada responsável pela apresentação da interface da aplicação ao usuário. As páginas implementadas utilizam Html, CSS, JavaServer Faces (JSF), PrimeFaces;

**Controle** - Camada responsável pela comunicação entre as camadas de Visão e Modelo. Implementa também as regras de negócio pertinentes a requisitos funcionais. São utilizados Servlets para implementação das classes além do Framework SpringMVC.

**Modelo** - Camada responsável pelo acesso a base de dados. É utilizado o padrão de projeto DAO (Data Access Object) para implementação das classes.

**7 - Qualidade**

A arquitetura adotada, com separação em camadas, busca dar mais produtividade às implementações e otimizar a manutenibilidade do sistema. Além disso, garante a confiabilidade, a integridade e a disponibilidade das informações, através da segurança de controle de acesso.