**SIB – Sistema Integrado de Biblioteca**

## **Documento de Arquitetura de Software**

**Alunos:**

**Eduardo Lapa**

**Gisana Cristina**

**Lucas Orion**

**Matheus França**

**Israel Alves**

**Rogério Amorim**

**Sumário**

[**Documento de Arquitetura de Software** 1](#_gjdgxs)

[**1 - Introdução** 3](#_1fob9te)

[**1.1 - Finalidade** 3](#_3znysh7)

[**1.2 – Escopo** 3](#_2et92p0)

[**1.3 - Definições, Acrônimos e Abreviações** 3](#_tyjcwt)

[**1.4 - Visão Geral** 3](#_3dy6vkm)

[**2 - Representação da Arquitetura** 3](#_1t3h5sf)

[**3 - Visão Lógica** 4](#_4d34og8)

[**3.2 - Visão Geral** 4](#_2s8eyo1)

[**4 - Visão de Processos** 4](#_26in1rg)

[**5 - Visão de Implantação** 4](#_lnxbz9)

[**6 - Visão de Implementação** 5](#_35nkun2)

[**6.1 - Visão Geral** 5](#_1ksv4uv)

[**6.2 - Camadas** 5](#_44sinio)

[**6.2.1 - View** 5](#_2jxsxqh)

[**6.2.2 - Model** 5](#_z337ya)

[**6.2.3 – Controller** 5](#_3j2qqm3)

[**7 - Tamanho e Desempenho** 5](#_1y810tw)

[**8 - Qualidade** 6](#_4i7ojhp)

# **1 - Introdução**

Uma vez terminada a elicitação de requisitos de um sistema, obtém-se a necessidade de se ter uma solução de como deve ser estruturado o software de modo que venha atender as restrições e funcionalidades que o software necessita. E esta solução se define em partes neste documento de arquitetura de software.

## **1.1 - Finalidade**

Este documento apresenta uma visão geral abrangente da arquitetura do sistema e utiliza uma série de visões arquiteturais diferentes para ilustrar os diversos aspectos do sistema. Sua intenção é capturar e transmitir as decisões significativas do ponto de vista da arquitetura que foram tomadas em relação ao sistema.

## **1.2 – Escopo**

Este Documento de Arquitetura de Software se aplica a denominada SBI (Sistema Integrado de Biblioteca).

## **1.3 - Definições, Acrônimos e Abreviações**

Elicitação: Técnica de obtenção de dados junto aos usuários detentores das informações.

## **1.4 - Visão Geral**

Apos a definição da finalidade e escopo deste documento, abaixo vamos descrever a arquitetura escolhida que melhor soluciona os problemas e necessidade encontrados neste projeto. Assim vamos estar apresentando em 5 visões sendo elas casos de uso, logica, processo, implantação e implementação.

# **2 - Representação da Arquitetura**

Este documento apresenta a arquitetura como uma série de visões: visão de casos de uso, visão lógica, visão de processos, visão de implantação e visão de implementação.

# **3 - Visão Lógica**

Descreve as classes mais importantes, sua organização em pacotes e subsistemas de serviço, e a organização desses subsistemas em camadas.

## **3.2 - Visão Geral**

A visão geral da aplicação é composta por 3 pacotes:

* View: Este não esta preocupado em como a informação foi obtida ou onde ela foi obtida, apenas exibe a informação ao usuário.
* Controller: Este determina o fluxo de apresentação servindo como uma camada intermediária, ou seja, o intercessor entre o View e o Model.
* Model: Este é responsável por tudo o que a aplicação irá fazer. Modelando os dados e o comportamento do sistema, bem como preocupa com o armazenamento, manipulação e geração de dados, sendo um encapsulador de dados e de comportamento independente da apresentação (view).

**4 - Visão de Processos**

O sistema é gerenciado por meio de processos, esses processos podem ser divididos com base na sua capacidade de influência para o sistema como um todo, podendo ser classificados em dois tipos :

* Processos leves: São processos de baixa importância dentro do sistema, tais como threads de baixa prioridade criados para processamento paralelo.
* Processos pesados: São processos de alto impacto dentro do sistema como um todo, em que o má gerenciamento pode comprometer outras áreas do sistema ,tais como threads criadas para a interação com o usuário.

# **5 - Visão de Implantação**

O sistema é construído na linguagem java ,com o foco voltado para web. Disponibilizando a portabilidade do sistema a qualquer terminal web ao usuário.

# **6 - Visão de Implementação**

## **6.1 - Visão Geral**

Seguindo o padrão arquitetural MVC ,o sistema pode ser dividido em 3 camadas : Model, view, controller. Nessas camadas são separados as classes do código com base em sua influência no sistema. As classes da camada view são responsáveis apenas pelo front-end do sistema. As classes da camada model são as classes atômicas e as classes compostas essenciais para construção do sistema. As classes da camada controller são responsáveis pela construção do back-end do sistema.

## **6.2 - Camadas**

### **6.2.1 - View**

A camada view é responsável pela interação do usuário com o sistema ,nessa camada são realizados etapas como construção de interface de usuário e interação para as configurações das funcionalidades.

### **6.2.2 - Model**

A camada model é responsável pelo armazenamento das classes mais atômicas do projeto ,nessa camada são armazenados estados de cada objeto.

**6.2.3 – Controller**

A camada controller é responsável pela execução de algoritmos complexos como interpretação dos dados da interface para o sistema e o controle do ciclo de vida da aplicação.

# **7 - Tamanho e Desempenho**

O sistema proposto possui tamanho em disco de 30MB e é desenvolvido para uso em servidor propagado ao usuário pela web. A criação de um sistema com muitas classes e arquivos avulsos devem ser ponderados de forma a não afetar (ou afetar o mínimo possível) o desempenho do sistema no servidor durante a execução.

# **8 - Qualidade**

Sabendo que qualidade e seus atributos é a base para as estratégias e decisões de design da arquitetura. O padrão de arquitetura MVC foi a solução mais satisfatória para atender a qualidade esperada do sistema. Sistema este que deverá ser desenvolvido na linguagem java, visando uma interface que seja interativa e fácil de se usar.