Pontifícia Universidade Católica do Paraná

Escola POLITÉCNICA

BACHARELADO EM SISTEMAS DE INFORMAÇÃO

Hamilton santos

Talles borges

F.A.D.D.V.M.

documento de design

CURITIBA

2013

Hamilton santos

talles borges

F.A.D.D.V.M.

documento de design

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Sistemas de Informação da Pontifícia Universidade Católica do Paraná, como requisito parcial à obtenção do título de Bacharel em Sistemas de Informação.

Orientadores:

Prof. Msc. Attilio Zanelatto Neto

Prof. Msc. Everson Mauda

Prof. Msc. Mario Luiz Bernert

CURITIBA

2013

SUMÁRIO

[1 Introdução 3](#_Toc358578098)

[2 Estudo Bibliográfico da Tecnologia Utilizada 3](#_Toc358578099)

[2.1 Objetivos 3](#_Toc358578100)

[2.2 LINGUAGENS DE PROGRAMAÇÃO 4](#_Toc358578101)

[2.3 bancos de dados 4](#_Toc358578102)

[2.4 outras tecnologias 5](#_Toc358578103)

[3 Arquitetura 6](#_Toc358578104)

[3.1 visao geral 6](#_Toc358578105)

[3.2 decisões e justificativas 7](#_Toc358578106)

[4 realização dos casos de uso 8](#_Toc358578107)

[**4.1** **Caso de Uso 02 – Manter histórico do paciente** 8](#_Toc358578108)

[4.1.1 Funcionamento geral 9](#_Toc358578109)

[4.1.2 Especificação das interfaces visuais 10](#_Toc358578110)

[4.1.3 Especificacão dos serviços 10](#_Toc358578111)

[4.1.4 Especificação da camada de persistência 10](#_Toc358578112)

[**4.2** **Caso de Uso 05 – Manter categoria** 12](#_Toc358578113)

[4.2.1 Funcionamento Geral 12](#_Toc358578114)

[4.2.2 Especificaçao das interfaces visuais 13](#_Toc358578115)

[4.2.3 Especificaçoes dos serviços 13](#_Toc358578116)

[4.2.4 Expecificacao da camada de persistência 14](#_Toc358578117)

[5 Componentes comuns 14](#_Toc358578118)

[6 Diagrama de classes geral da aplicação 15](#_Toc358578119)

[7 modelo físico de dados 16](#_Toc358578120)

[8 Protótipo das interfaces 16](#_Toc358578121)

[9 Considerações Finais 16](#_Toc358578122)

[Referências 17](#_Toc358578123)

# Introdução

A ferramenta de auxilio na decisão do desmame de ventilação mecânica ou F.A.D.D.V.M. é um sistema especialista que irá apoiar o profissional fisioterapeuta intensivista na tarefa de decisão quanto à retirada de um paciente da ventilação mecânica, a fim de reduzir os riscos que a decisão precipitada nesse momento pode causar ao paciente, a intensão do sistema é também reduzir os gastos extras que podem ser causados por esta decisão.

Atualmente a decisão quanto à este procedimento é feita de forma empírica pelo fisioterapeuta, a partir de dados coletados tanto em exames como a gasometria quanto na anamnese, porém, a não padronização deste processo pode causar uma diferença nos resultados da análise de fisioterapeuta para fisioterapeuta.

Além da padronização dos resultados, o sistema tem por objetivo, auxiliar o profissional em início de carreira, que ainda não possui tanta experiência garantindo que certos parâmetros da decisão serão sempre analisados, e analisados de forma correta.

Este documento irá apresentar o sistema em desenvolvimento do ponto de vista técnico, apresentando os detalhes sobre as tecnologias que serão utilizadas, detalhes gerais da arquitetura do sistema, realização de casos de uso e conterá também um protótipo de como ficará a aplicação com o seu comportamento simulado.

# Estudo Bibliográfico da Tecnologia Utilizada

## Objetivos

Esta seção tem por objetivo apresentar um estudo bibliográfico sobre as tecnologias utilizadas neste projeto, permitindo aos envolvidos no projeto uma visão geral da tecnologia utilizada.

## LINGUAGENS DE PROGRAMAÇÃO

**Java EE 6**, esta linguagem foi definida a pedido do cliente, pois será de mais fácil manutenção e futuros aprimoramentos pelos desenvolvedores e analistas que ficarão responsáveis pelo mantenimento do mesmo.

Ela é uma linguagem com suporte a multicamadas, distribuída e multiusuário, rodando em um servidor de aplicação, podendo ser acessado de qualquer computador com acesso ao mesmo.

A utilização desta linguagem dará ao cliente uma maior flexibilidade, pois é uma linguagem multiplataforma, podendo rodar em qualquer SO, e sendo assim, pode rodar em qualquer configuração de servidor, desde que o mesmo atenda as especificações de performance aceitáveis para o funcionamento do sistema.

A Linguagem é Orientada a Objetos, o que a torna uma ferramenta extremamente poderosa. E é também Multi-Threaded, o que significa dizer que ela suporta processamento paralelo múltiplo podendo ter melhoramentos de performance.

## bancos de dados

O banco de dados que iremos utilizar é o **MySQL**, também a pedido do cliente, pois é o banco de dados atualmente utilizado pelo sistema de prontuário eletrônico e o cliente tem planos de integrar os dois sistemas no futuro.

MySQL é um dos bancos de dados opensource mais utilizados devido à sua robustez e flexibilidade, ele é amplamente utilizado por grandes empresas como Wikipedia, Google, Facebook e Twitter e possui, versões gratuitas e pagas, dependendo dos fins do software que irá fazer uso dele e das necessidades da empresa.

Escrito em C e C++ o MySQL é bastante performático, Funciona em diversas plataformas. APIs para C, C++, Eiffel, Java, Perl, PHP, Python, Ruby e Tcl estão disponíveis.

Suporte total a multi-threads usando threads diretamente no kernel. Isto significa que se pode facilmente usar múltiplas CPUs, se disponível.

Tabelas em disco (MyISAM) baseadas em árvores-B extremamente rápidas com compressão de índices.

Um sistema de alocação de memória muito rápido e baseado em processo(thread).

Joins muito rápidas usando uma multi-join de leitura única otimizada.

Tabelas hash em memória que são usadas como tabelas temporárias.

Funções SQL são implementadas por meio de uma biblioteca de classes altamente otimizada e com o máximo de performance. Geralmente não há nenhuma alocação de memória depois da inicialização da pesquisa.

## outras tecnologias

**JSF**, um framework MVC de aplicações web baseado em Java, que se destina a simplificar o desenvolvimento de interfaces de usuário baseadas em web, iremos usar seus componentes visuais, suas regras de navegação, conceitos de backbeans e regras de validação que serão bastante uteis no projeto.

**CSS,** uma linguagem de fácil configuração e entendimento para alteração e definição de parâmetros de aparência em websites, permitindo com facilidade, que alterando apenas um arquivo, se consiga redefinir a aparência de todas as paginas da aplicação web.

**Bootstrap**, desenvolvida pela equipe do Twitter para criação de websites, ela possui uma ampla biblioteca de componentes visuais e ferramentas baseadas em HTML, CSS e javascript, para a configuração visual do website.

**JQuery**, biblioteca javascript desenvolvida para simplificar os scripts cliente-side que interagem com o HTML, possui vários componentes visuais interessantes para serem usados em aplicações web.

**JDBC**, a aplicação utilizará JDBC para se comunicar com o banco de dados, ele é uma API com várias classes e métodos para envio de intruções SQL para qualquer banco de dados, cada banco de dados fornece um driver que implementa métodos definidos pelo JDBC para realizar a comunicação da aplicação que utiliza o JDBC e o banco de dados.

Servidor de aplicação **Glassfish,** utilizaremos o servidor de aplicação Glassfish, por ser totalmente compatível com Java EE, incluindo JSF, EJB e JPA, sendo assim, recomendado e pouco provável de causar problemas com a configuração do nosso projeto, outro motivo importante para termos escolhido Glassfish foi o fato de ele utilizar a licença GPL, mesma licença que pretendemos dar a nosso projeto, sendo assim, necessária em todas as ferramentas utilizadas.

Estamos utilizando também a ferramenta **GitHub** para versionar nossos arquivos, códigos e documentos do projeto, podendo assim controlar com segurança o versionamento e tendo assim uma maior rastreabilidade dos avanços, além de uma garantia de não haver perda de informação durante o avanço do projeto.

# Arquitetura

## visao geral

O padrão arquitetural que utilizaremos é o **MVC**, padrão mais utilizado para aplicações java web, por separar as camadas de apresentação, controle e modelo, sendo assim de fácil compreensão e manutenção.

Com o aumento da complexidade das aplicações desenvolvidas, sempre visando a programação orientada a objeto, torna-se relevante a separação entre os dados e a apresentação das aplicações. Desta forma, alterações feitas no layout não afetam a manipulação de dados, e estes poderão ser reorganizados sem alterar o layout.

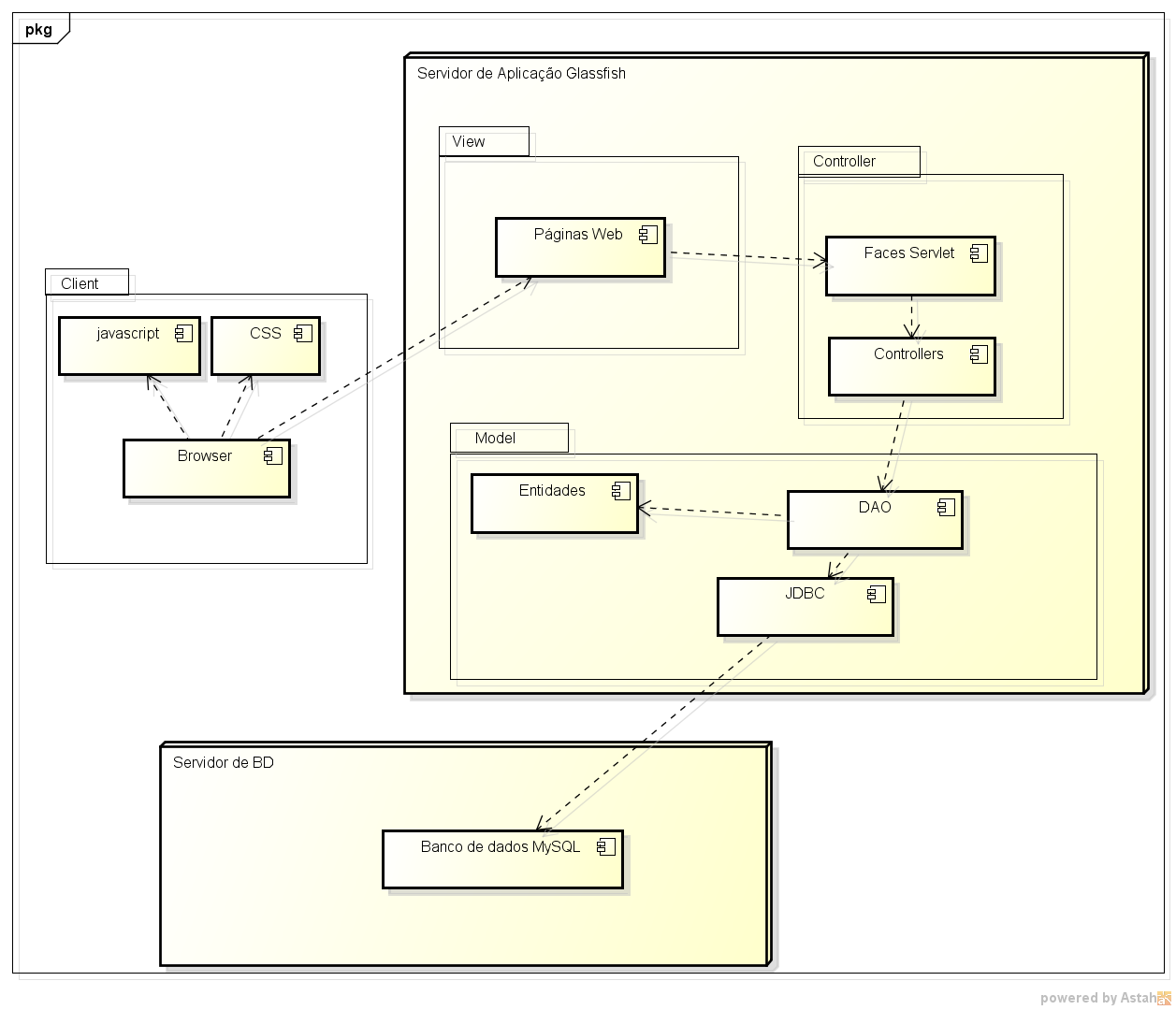
Esse padrão resolve este problema através da separação das tarefas de acesso aos dados e lógica de negócio, lógica de apresentação e de interação com o utilizador, introduzindo um componente entre os dois: o controlador.

Na camada de apresentação será onde utilizaremos boa parte dos frameworks, CSS, Bootstrap e JQuery serão usados nesta camada para tornar a aparência da aplicação mais agradável e mais fácil de modificar, esta é a camada que o terá interação direta com o usuário, é a camada que é apresentada do browser.

A camada de controle será a camada que irá implementar as regras do sistema e regras de negócio, assim como regras de navegação e validação de dados, Ela é a camada que reage a entrada de dados do usuário, chamando o metidi apropriado na cmada Model, é esta camada que define a navegação na aplicação e altera as informações

A camada modelo será a camada da aplicação que irá representar os valores que estão no banco de dados durante sua utilização e alteração, esta camada que irá se comunicar com o banco de dados e utilizar suas informações para repassá-las às outras camadas do sistema.

Abaixo está o diagrama de implantação da F.A.D.D.V.M. para melhor compreensão do padrão que adotamos.



## decisões e justificativas

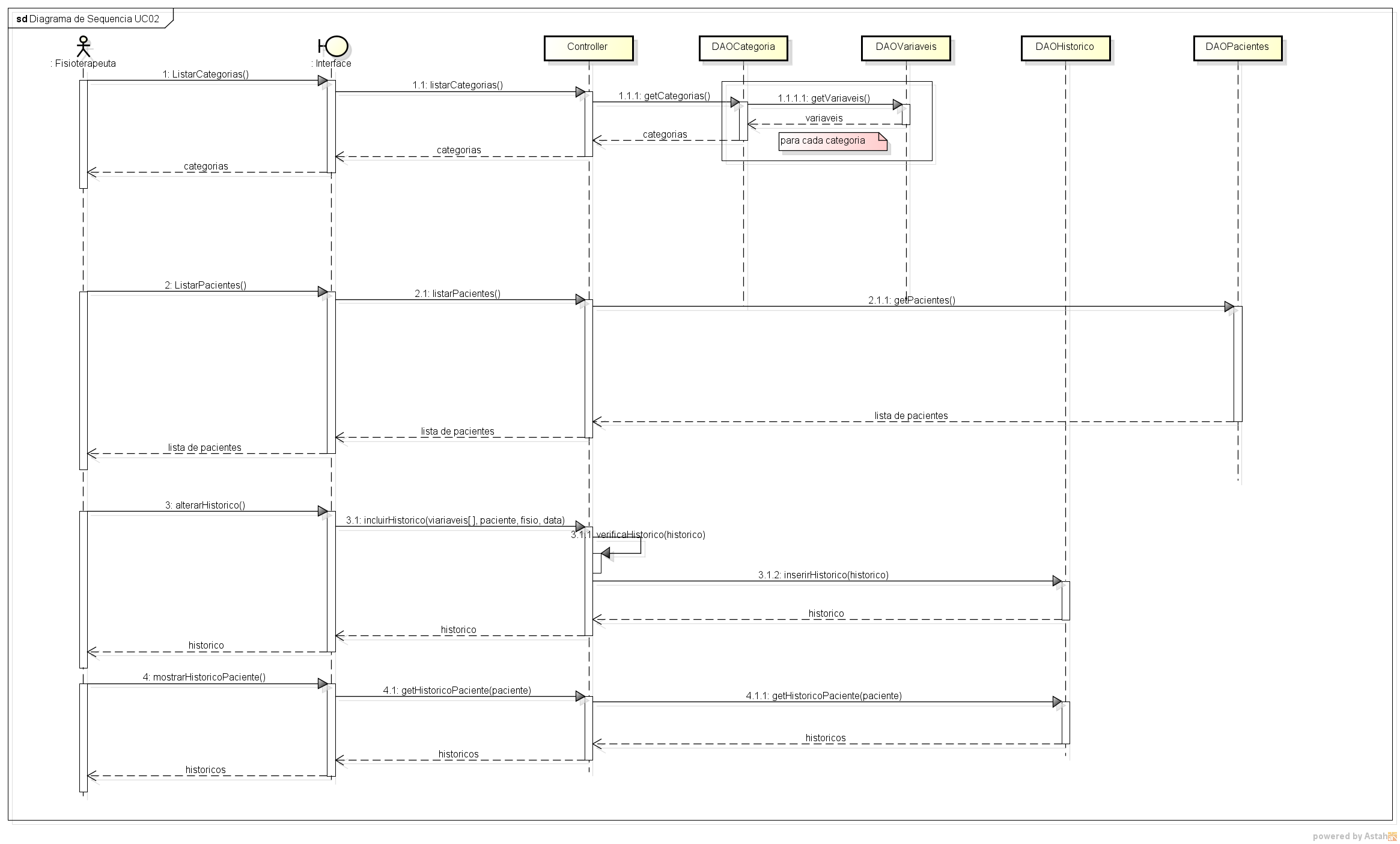
* Linguagem escolhida em reunião com o cliente, devido ao amplo uso de java e maior facilidade de manutenção do código por outros desenvolvedores que possam assumir o desenvolvimento de possíveis correções e melhoramentos.
* MVC escolhido por ser um framework amplamente usado e considerado uma boa prática para o mantenimento do sistema e para o fácil entendimento do código por programadores que possam precisar analisá-lo.
* JSF foi escolhido por ser o framework mais indicado para aplicações java EE, ele é um dos mais atuais e possui fácil manutenção e compreensão, possui controle de navegação entre paginas, gerenciamento de objetos, validação de tratamento de erros entre outras vantagens que foram levadas em consideração.
* Utilizaremos os componentes mais simples que encontramos para a manipulação da interface gráfica, utilizando as tags fornecidas pelo JSF assim como leves modificações utilizando CSS, javastcript e componentes fornecidos por JQuery e Bootstrap.
* Decidimos por MySQL também em reunião com o cliente, ele foi o BD escolhido para ser utilizado no nosso sistema por ser um robusto banco de dados open source, por ser utilizado no prontuário eletrônico do hospital, sistema este que o cliente pretende integrar com o F.A.D.D.V.M. em um futuro próximo e por atender as licenças que pretendemos utilizar.

# realização dos casos de uso

## **Caso de Uso 02 – Manter histórico do paciente**

Este caso de uso tem por objetivo, realizar as interações com o histórico do paciente, cadastros de novas entradas no histórico e consultas de valores, assim como alterações ou remoções em caso de entradas incorretas.

### Funcionamento geral



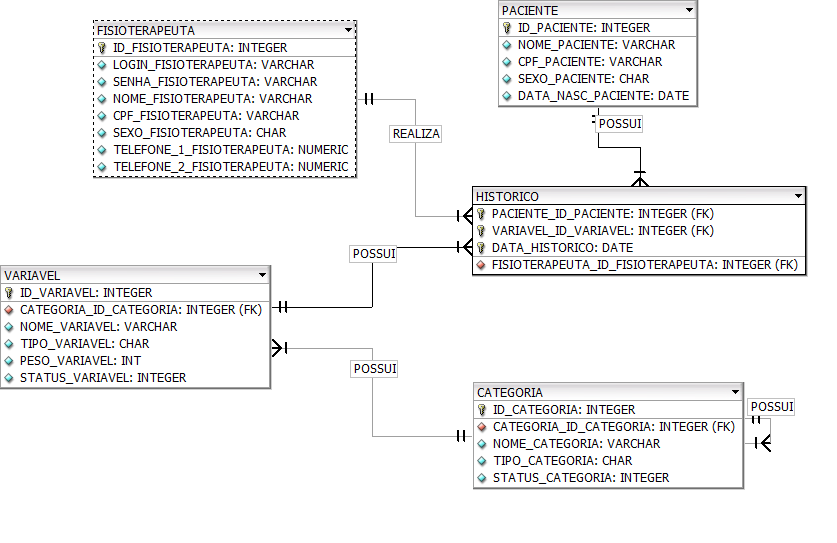
### Especificação das interfaces visuais

### Especificacão dos serviços

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Descriçao | Nome do Serviço | Entrada | Saída | Pré-cond | Pós-cond |
| Incluir nova entrada no histórico do paciente | incluirHistorico | Variáveis, fisioterapeuta, data, paciente | Histórico | Paciente cadastrado | Entrada de histórico |
| Mostrar histórico do paciente | mostrarHistorico | Paciente | Históricos do paciente | Paciente cadastrado | Histórico apresentado |

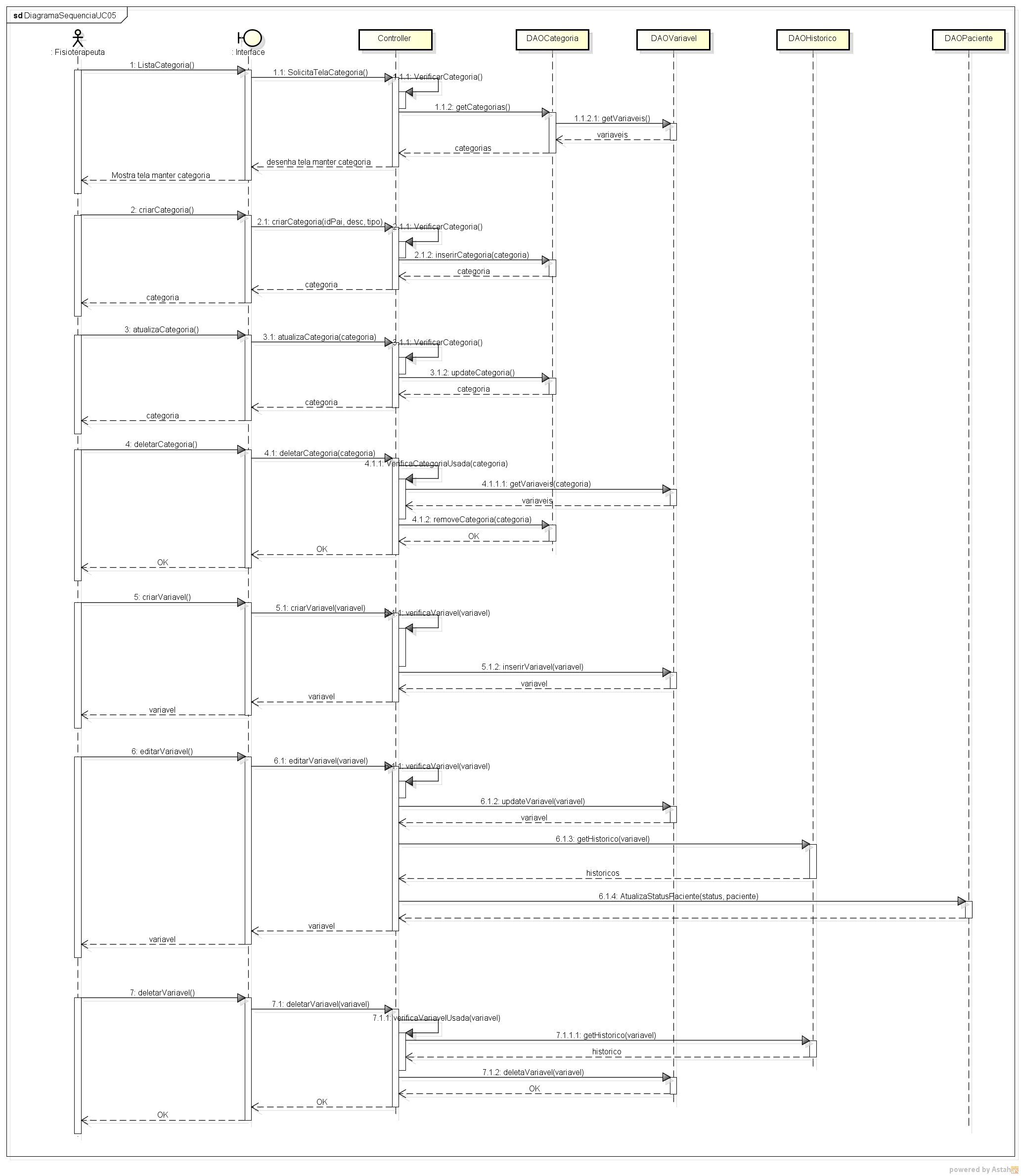
### Especificação da camada de persistência

Na persistência de histórico foram utilizadas todas as tabelas listadas na imagem abaixo.



## **Caso de Uso 05 – Manter categoria**

### Funcionamento Geral



### Especificaçao das interfaces visuais

### Especificaçoes dos serviços

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Descriçao | Nome do Serviço | Entrada | Saída | Pré-cond | Pós-cond |
| Criar categoria nova | criarCategoria | Categoria pai, descricao, tipo, | Categoria | Fisioterapeuta precisa ter perfil de adm | Categoria criada |
| Editar categoria | editarCategoria | Categoria | Categoria | Fisioterapeuta precisa ter perfil de adm, | Categoria alterada |
| Excluir categoria | excluirCategoria | Catgoria | Mensagem de sucesso | Fisioterapeuta precisa ter perfil de adm, categoria não pode ter variáveis | Categoria excluída |
| Criar variavel | criarVariavel | Variavel | Variavel | Fisioterapeuta precisa ter perfil de adm, tipo da categoria pai deve ser grupo | Variavel cadastrada |
| Editar variavel | editarVariavel | Variavel | Variavel | Fisioterapeuta precisa ter perfil de adm | Variavel alterada, status dos pacientes alterado |
| Excluir variavel | excluirVariavel | Variavel | Mensagem de sucesso | Fisioterapeuta precisa ter perfil de adm, variavel não pode ter sido usada em histórico | Variavel excluída |

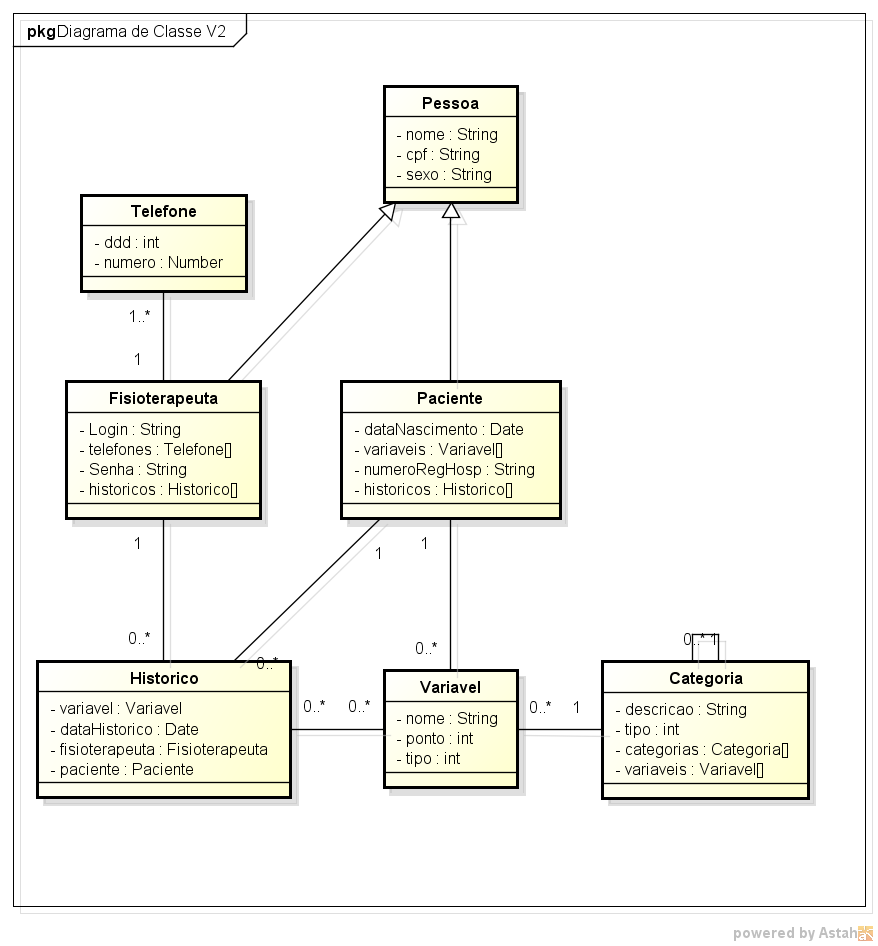
### Expecificacao da camada de persistência

Blablablablabla

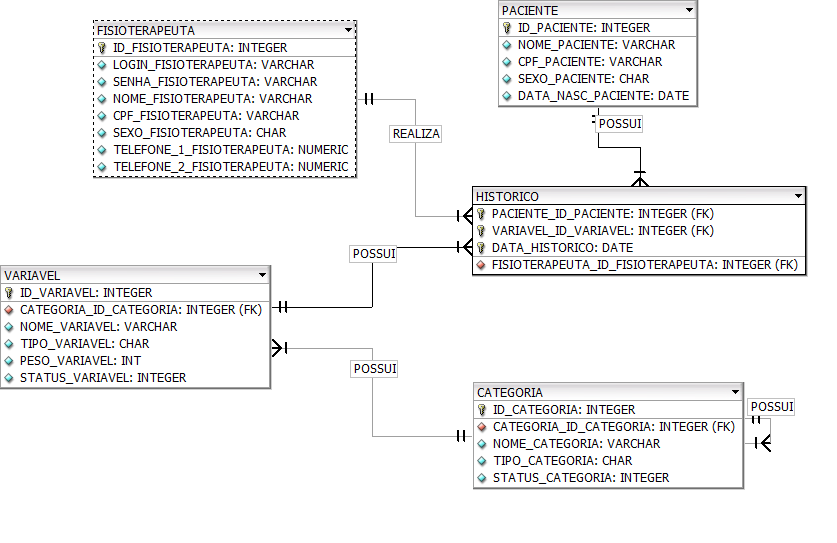
# Componentes comuns

O aluno deverá descrever aqui os componentes que são comuns aos casos de uso realizados na seção anterior. Desde que reusáveis, os componentes especificados aqui podem ser dos mais diversos tipos. Alguns exemplos: serviços, funções e componentes de tela.

# Diagrama de classes geral da aplicação



# modelo físico de dados



# Protótipo das interfaces

Durante a banca de análise será definido quais interfaces deverão compor o protótipo do sistema, o qual deverá permitir a navegação entre as funcionalidades mas não precisará realizar os tratamentos de campos e inserções no banco de dados.

# Considerações Finais

Uma breve conclusão mencionando o trabalho desenvolvido, questões chaves encontradas nesta fase, assim como uma breve descrição dos próximos passos.

# Referências

[Perrone, Paul J.](http://pt.wikipedia.org/w/index.php?title=Paul_J._Perrone&action=edit&redlink=1); Chaganti, Krishna. [**J2EE Developer's Handbook**](http://www.informit.com/store/product.aspx?isbn=0672323486). Indianapolis, Indiana: Sam's Publishing, 2003.

Bodoff, Stephanie; **The J2EE Tutorial**. Boston: Addison-Wesley, 2004.

Kumaraswamipillai, Arulkumaran; [**Java/J2EE Job Interview Companion**](http://books.google.com/books?id=saF6F-sO_H0C&lpg=PP1&pg=PA5#v=onepage&q=&f=false). [S.l.: s.n.], 2007.

["Market Share"](http://www.mysql.com/why-mysql/marketshare/). Why MySQL?. Oracle. Retrieved 17 September 2012.

["DB-Engines Ranking"](http://db-engines.com/en/ranking). Retrieved 26 February 2013.

SCHUMACHER, Robin; Lentz, Arjen. ["Dispelling the Myths"](http://web.archive.org/web/20110606013619/http:/dev.mysql.com/tech-resources/articles/dispelling-the-myths.html).**[MySQL AB](https://en.wikipedia.org/wiki/MySQL_AB" \o "MySQL AB)**. Archived from [the original](http://dev.mysql.com/tech-resources/articles/dispelling-the-myths.html) on 6 June 2011. Retrieved 17 September 2012.